Digital Ecosystems and Business Intelligence Institute

A Customized Semantic Service Retrieval Methodology for the Digital Ecosystems Environment

Hai Dong

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University of Technology

June 2010

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Signature: ………………………………………….

Date: ………………………...

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THESIS SUMMARY

With the emergence of the Web and its pervasive intrusion on individuals, organizations, businesses etc., people now realize that they are living in a digital environment analogous to the ecological ecosystem. Consequently, no individual or organization can ignore the huge impact of the Web on social well-being, growth and prosperity, or the changes that it has brought about to the world economy, transforming it from a self-contained, isolated, and static environment to an open, connected, dynamic environment. Recently, the European Union initiated a research vision in relation to this ubiquitous digital environment, known as Digital (Business) Ecosystems. In the Digital Ecosystems environment, there exist ubiquitous and heterogeneous species, and ubiquitous, heterogeneous, context-dependent and dynamic services provided or requested by species. Nevertheless, existing commercial search engines lack sufficient semantic supports, which cannot be employed to disambiguate user queries and cannot provide trustworthy and reliable service retrieval. Furthermore, current semantic service retrieval research focuses on service retrieval in the Web service field, which cannot provide requested service retrieval functions that take into account the features of Digital Ecosystem services. Hence, in this thesis, we propose a customized semantic service retrieval methodology, enabling trustworthy and reliable service retrieval in the Digital Ecosystems environment, by considering the heterogeneous, context-dependent and dynamic nature of services and the heterogeneous and dynamic nature of service providers and service requesters in Digital Ecosystems.

The customized semantic service retrieval methodology comprises: 1) a service information discovery, annotation and classification methodology; 2) a service retrieval methodology; 3) a service concept recommendation methodology; 4) a quality of service (QoS) evaluation and service ranking methodology; and 5) a service domain knowledge updating, and service-provider-based Service Description Entity (SDE) metadata publishing, maintenance and classification methodology.

The service information discovery, annotation and classification methodology is designed for discovering ubiquitous service information from the Web, annotating the discovered service information with ontology mark-up languages, and classifying the annotated service information by means of specific service domain knowledge, taking into account the heterogeneous and context-dependent nature of Digital Ecosystem services and the heterogeneous nature of service providers. The methodology is realized by the prototype of a Semantic Crawler, the aim of which is to discover service advertisements and service provider profiles from webpages, and annotating the information with service domain ontologies.

The service retrieval methodology enables service requesters to precisely retrieve the annotated service information, taking into account the heterogeneous nature of Digital

Ecosystem service requesters. The methodology is presented by the prototype of a Service Search Engine. Since service requesters can be divided according to the group which has relevant knowledge with regard to their service requests, and the group which does not have relevant knowledge with regard to their service requests, we respectively provide two different service retrieval modules. The module for the first group enables service requesters to directly retrieve service information by querying its attributes. The module for the second group enables service requesters to interact with the search engine to denote their queries by means of service domain knowledge, and then retrieve service information based on the denoted queries.

The service concept recommendation methodology concerns the issue of incomplete or incorrect queries. The methodology enables the search engine to recommend relevant concepts to service requesters, once they find that the service concepts eventually selected cannot be used to denote their service requests. We premise that there is some extent of overlap between the selected concepts and the concepts denoting service requests, as a result of the impact of service requesters' understandings of service requests on the selected concepts by a series of human-computer interactions. Therefore, a semantic similarity model is designed that seeks semantically similar concepts based on selected concepts.

The QoS evaluation and service ranking methodology is proposed to allow service requesters to evaluate the trustworthiness of a service advertisement and rank retrieved service advertisements based on their QoS values, taking into account the context-dependent nature of services in Digital Ecosystems. The core of this methodology is an extended CCCI (Correlation of Interaction, Correlation of Criterion, Clarity of Criterion, and Importance of Criterion) metrics, which allows a service requester to evaluate the performance of a service provider in a service transaction based on QoS evaluation criteria in a specific service domain. The evaluation result is then incorporated with the previous results to produce the eventual QoS value of the service advertisement in a service domain. Service requesters can rank service advertisements by considering their QoS values under each criterion in a service domain.

The methodology for service domain knowledge updating, service-provider-based SDE metadata publishing, maintenance, and classification is initiated to allow: 1) knowledge users to update service domain ontologies employed in the service retrieval methodology, taking into account the dynamic nature of services in Digital Ecosystems; and 2) service providers to update their service profiles and manually annotate their published service advertisements by means of service domain knowledge, taking into account the dynamic nature of service providers in Digital Ecosystems. The methodology for service domain knowledge updating is realized by a voting system for any proposals for changes in service domain knowledge, and by assigning different weights to the votes of domain experts and normal users.

In order to validate the customized semantic service retrieval methodology, we build a prototype – a Customized Semantic Service Search Engine. Based on the prototype, we test the mathematical algorithms involved in the methodology by a simulation approach and validate the proposed functions of the methodology by a functional testing approach.

ACKNOWLEDGEMENT

I would like to acknowledge the Grace of the God, for giving me this opportunity to complete my Doctoral Dissertation under the supervision of Dr. Farookh Khadeer Hussain and Professor Elizabeth Chang.

Additionally, I wish to acknowledge the efforts of my family without whose support and sacrifices I would never have been the person that I am today. To my parents, Mr. Jianwen Dong and Mrs. Aiying Yang, who have always been there to support me and have sacrificed a lot for me, I can never ever thank you enough for all that you have done for me.

Next, I owe my deepest gratitude to my supervisor, Dr. Farookh Khadeer Hussain, whose endless encouragement, superb supervision and never-ending support from the preliminary to the concluding levels enabled me to understand the genuine meaning of undertaking a research and being a researcher.

I would also like to express my sincere thanks to Professor Elizabeth Chang; and without her great support and patient guidance I could not have embarked on my PhD life and reached this stage.

I also extend my sincere thanks to Mr. Wei Liu, our Web designer and programmer who dedicated his precious time to help me to build the system prototype and to implement the system evaluation task.

Finally, I offer my regards and blessings to all of those who have supported me in any respect in the past three years.

LIST OF PUBLICATIONS

Refereed Journal Articles

1. Dong, H., Hussain, F.K., Chang, E.: A context-aware semantic similarity model for ontology environments. Concurrency and Computation: Practice and Experience **In Press**

2. Dong, H., Hussain, F.K., Chang, E.: A framework for discovering and classifying ubiquitous services in digital health ecosystems. Journal of Computer and System Sciences **In Press**

3. Dong, H., Hussain, F.K.: Focused crawling for automatic service discovery, annotation and classification in industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

4. Dong, H., Hussain, F.K., Chang, E.: A service search engine for the industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

5. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

6. Dong, H., Hussain, F.K., Chang, E.: A QoS-based service retrieval methodology for digital ecosystems. International Journal of Web and Grid Services **5** (2009) 261-283

Book Chapters

1. Dong, H., Hussain, F.K., Chang, E.: A hybrid concept similarity measure model for ontology environment. In: Meersman, R., Herrero, P., Dillon, T. (eds.): On the Move to Meaningful Internet Systems: OTM 2009 Workshops. Springer-Verlag, Vilamoura, Portugal (2009) 848–857

2. Dong, H., Hussain, F.K., Chang, E.: State of the art in semantic focused crawlers In: Gervasi, O., Taniar, D., Murgante, B., Laganà, A., Mun, Y., Gavrilova, M. (eds.): Computational Science and Its Applications – ICCSA 2009. Springer-Verlag, Seoul, Korea (2009) 910-924

3. Dong, H., Hussain, F.K., Chang, E.: Paradox in applications of semantic similarity models in information retrieval. In: M. Ulieru, P.P., and R. Doursat (ed.): IT Revolutions. Springer-Verlag, Venice, Italy (2009) 60-68

4. Dong, H., Hussain, F.K., Chang, E.: Semantic service search, service evaluation and ranking in service oriented environment. In: Ulieru, M., Palensky, P., Doursat, R. (eds.): IT Revolutions. Springer-Verlag, Venice, Italy (2009) 111-117

5. Dong, H., Hussain, F.K., Chang, E.: A semantic crawler based on an extended CBR algorithm. In: Meersman, R., Tari, Z., Herrero, P. (eds.): On the Move to Meaningful Internet Systems: OTM 2008 Workshops. Springer-Verlag, Monterrey, Mexico (2008) 1084-1093

Refereed Conference Proceedings

1. Dong, H., Hussain, F.K., Chang, E.: A framework enabling semantic search in health service ecosystems. Proceedings of the 6th International Conference on Semantics, Knowledge and Grids (SKG '10). IEEE Computer Society, Ningbo, China (2010) Accepted

2. Dong, H., Hussain, F.K., Chang, E.: Semantic service retrieval and QoS measurement in the Digital Ecosystem environment. Proceedings of the 4th International Conference on Complex, Intelligent and Software Intensive Systems (CISIS 2010). IEEE Computer Society, Krakow, Poland (2010) 153-160

3. Dong, H., Hussain, F.K., Chang, E.: An ontology-based webpage classification approach for the knowledge grid environment. Proceedings of the Fifth International Conference on Semantics, Knowledge and Grid (SKG '09). IEEE Computer Society, Zhuhai, China (2009) 120-127

4. Dong, H., Hussain, F.K., Chang, E.: A hybrid service metadata clustering methodology in the Digital Ecosystem environment. Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops (WAINA '09). IEEE Computer Society, Bradford, UK (2009) 238-243

5. Dong, H., Hussain, F.K., Chang, E.: Quality of service (QoS) based service retrieval engine. Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM '08). ACM, Linz, Austria (2008) 405-408

6. Dong, H., Hussain, F.K., Chang, E.: A transport service ontology-based focused crawler. Proceedings of the Fourth International Conference on Semantics, Knowledge and Grid (SKG '08). IEEE Computer Society, Beijing, China (2008) 49-56

7. Dong, H., Hussain, F.K., Chang, E.: A survey in Semantic Web technologies-inspired focused crawlers. Proceedings of the Third International Conference on Digital Information Management (ICDIM 2008). IEEE, London, UK (2008) 934-936

8. Dong, H., Hussain, F.K., Chang, E.: Transport service ontology and its application in the field of semantic search. Proceedings of the 2008 IEEE International Conference on Service Operations and Logistics, and Informatics (IEEE SOLI 2008). IEEE, Beijing, China (2008) 820-824

9. Dong, H., Hussain, F.K., Chang, E.: Developing a semantic service transaction system in the DE environment. Proceedings of the 6th IEEE International Conference on Industrial Informatics (INDIN 2008). IEEE, Daejeon, Korea (2008) 1353-1357

10. Dong, H., Hussain, F.K., Chang, E.: State of the art in metadata abstraction crawlers. Proceedings of the 2008 IEEE International Conference on Industrial Technology (IEEE ICIT 2008). IEEE, Chengdu, China (2008) 1-6

11. Dong, H., Hussain, F.K., Chang, E.: A survey in traditional information retrieval models. Proceedings of the 2008 2nd IEEE International Conference on Digital Ecosystem and Technologies (DEST 2008). IEEE, Phitsanulok, Thailand (2008) 397-402

12. Dong, H., Hussain, F.K., Chang, E.: Developing a conceptual framework of semantic search engine to promote the collaborations between SMEs in the Digital Ecosystems environment. Proceedings of the 2008 2nd IEEE International Conference on Digital Ecosystem and Technologies (DEST 2008). IEEE, Phitsanulok, Thailand (2008) 409-412

13. Dong, H., Hussain, F.K., Chang, E.: A survey in semantic search technologies. Proceedings of the 2008 2nd IEEE International Conference on Digital Ecosystem and Technologies (DEST 2008). IEEE, Phitsanulok, Thailand (2008) 403-408

14. Dong, H., Hussain, F.K., Chang, E.: Ontology-based digital ecosystem conceptual representation. Proceedings of the Third International Conference on Autonomic and Autonomous Systems (ICAS '07). IEEE Computer Society, Athens, Greek (2007) 42-42

15. Dong, H., Hussain, F.K., Chang, E.: An integrative view of the concept of Digital Ecosystem. Proceedings of the Third International Conference on Networking and Services (ICNS '07). IEEE Computer Society, Athens, Greek (2007) 42-42

Chapter 1 - Introduction

1.1 Introduction

In this chapter, we present an overview of Digital Ecosystems and the features of services in Digital Ecosystems. In Section 1.3.1, we present a case study to survey the service retrieval function in some of the commercial search engines, and present their shortcomings. Additionally, in Section 1.3.2 we give a brief introduction to the research works on service retrieval and semantic search, and point out their shortcomings. Subsequently, in Section 1.3.3, we present the pressing issues related to service retrieval that need to be addressed so as to realize a precise and trustworthy service retrieval in the Digital Ecosystems environment. Section 1.4 lists the objectives of this particular research. The subsequent section discusses the scope of the thesis and outlines clearly what does and does not lie within its scope. Section 1.6 presents the importance or significance of the objectives of the thesis in the context of Digital Ecosystems. In Section 1.7, we give a very brief introduction to each of the remaining ten chapters of this thesis. Finally, Section 1.8 concludes this chapter and sets the scene for the second chapter.

1.2 Digital Ecosystems and Services in Digital Ecosystems

1.2.1 Digital Ecosystems

With the emergence of the Web and its pervasive intrusion on individuals, organizations, businesses etc., people now realize that they are living in a digital environment which can be construed as being analogous to the ecological ecosystem [3]. Consequently, no individual or organization can ignore the huge impact of the Web on social well-being, growth and prosperity, or the changes that it has brought to the world economy, transforming it from a self-contained, isolated, and static pattern to an open, connected, dynamic pattern [4].

Recently, the European Commission initiated a research vision in relation to the ubiquitous digital environment, known as Digital (Business) Ecosystems [5]. Since the end of last century, the barriers that existed in the world economic environment have gradually collapsed. Worldwide industries, economic sectors, organizations and functions of organizations are linked and networked as a collaborative environment, which Moore has named a "*Business Ecosystem*" [12], from an ecological perspective. The Business Ecosystem is founded by interacting organizations and individuals, most of whose participants are small and medium enterprises (SMEs) [13]. The SMEs rely heavily on the social and business activities with their suppliers, clients, and business partners. The current information and communication technologies (ICT), however, cannot sufficiently facilitate such activities. In order to resolve this troublesome situation of SMEs, the European Union initiated Digital Ecosystems, the purpose of which is to provide a favourable ICT environment for the sustainable development of SMEs [6].

A Digital Ecosystem is defined as "*an open, loosely coupled, domain clustered, demand-driven, self-organising agents' environment, where each species is proactive and responsive for its own benefit or profit*" [4]. The species are the entities with common interests that participate in Digital Ecosystems. These contain biological species such as people, economic species such as organizations, and digital species such as software, hardware and agents [1]. These species need to interact with others in order to gain benefits and profits. For example, a company needs to implement a series of business transactions with customers who could be individuals or other companies in order to gain profits [8]. The environment refers to the underlying ICT and services that support species' activities within Digital Ecosystems [2]. Additionally, the species are the providers and requesters of the technologies and services. It is obvious that Digital Ecosystems constitute a widespread computing environment comprised of heterogeneous, geographically dispersed and ubiquitous species, technologies and services [9].

1.2.2 Features of Services in Digital Ecosystems

Services involved in the Digital Ecosystems environment have the characteristics of heterogeneity, contextual dependence and dynamism. These three features are explained as follows:

Heterogeneity. Digital Ecosystem services can be divided into three main categories according to the classification schema of Digital Ecosystem species: 1) individual services provided by biological species, such as food and beverage services etc., 2) economic/organizational services provided by economic species, such as business services etc., and 3) digital services provided by digital species, such as Web services etc. [4]. The information regarding Digital Ecosystem services, including service advertisements and service provider profiles, are mingled together with other Web information in the Web environment [10].

Contextual Dependence. A Digital Ecosystem service entity may have different content in different contexts [7]. For example, a car cleaning service for trucks has different content from that of a car cleaning service for small cars. Moreover, the evaluation criteria for the quality of services (QoS) may vary with the variation of contexts. In the above example, the QoS evaluation criteria for a car cleaning service for trucks may be different from the criteria for a car cleaning service for small cars. Analogously, the QoS status of a service entity is dynamic along with its changing contexts. For example, a company may have a good reputation as a vehicle cleaning service for small cars, but not an equally good reputation as a vehicle cleaning service for trucks.

Dynamism. In Digital Ecosystems, the content of a service entity is dynamic. The content of a service entity can be altered by its provider at any time. This dynamic nature is also revealed in service domain knowledge [10]. It is well recognized that knowledge is not static. In Digital Ecosystems, service domain knowledge needs to be altered in order to allow it to be consistent with people's understanding of service domains. There are many reasons for the change of domain knowledge, such as changing service requirements, dynamic service environments, agreements made between knowledge creators and knowledge users, etc... Furthermore, the QoS evaluation criteria of a service field and QoS status of a service entity, which can be regarded as parts of service domain knowledge, may evolve or change over time as a result of people's dynamic perceptions of the service field or the service entity.

From the perspective of services, species in Digital Ecosystems can play one or both of these roles: service provider (server) that provides services, and service requester (client) that needs services [4]. Service providers and service requesters are heterogeneous and dynamic, which can be expressed as follows:

Heterogeneity. As introduced in Section 1.2.1, Digital Ecosystem species are heterogeneous. Since service requesters and service providers are roles of species, they have the feature of heterogeneity. This would lead to heterogeneity in terms of their service requests. Therefore, from the perspective of service requests, we classify service requesters into two broader categories – the service requesters who do not have relevant domain knowledge about their service requests and the service requesters who have relevant domain knowledge about their service requests.

Dynamism. In Digital Ecosystems, a species can play the role of service provider and service requester simultaneously [4]. In other words, a service provider can be a service requester at the same time. Therefore, the services role of species is dynamic in Digital Ecosystems. Furthermore, the profile of a service provider (or a service requester), including provided service entities, can be altered by the service provider (or the service requester) anytime, in order to indicate the change of the service provider's conditions. The change of service providers' profiles also reflects the feature of dynamism.

In the next section, we will point out the issues related to service retrieval in commercial search engines and in the research community.

1.3 Issues with Service Retrieval in Digital Ecosystems

In this section, we will discuss some of the issues related to service retrieval in the Digital Ecosystems environment. It should be noted here that all the issues related to service retrieval in Digital Ecosystems are outside the scope of the thesis. We discuss only the chief or pressing issues regarding service retrieval in Digital Ecosystems that need to be addressed. The discussion is divided into three parts. In section 1.3.1, we discuss some of the chief service retrieval issues related to commercial search engines. In Section 1.3.2, we discuss some of the chief issues related to service retrieval in the research community. In Section 1.3.3, we point out and discuss at an abstract level some of service retrieval issues that need to be addressed in the Digital Ecosystems environment.

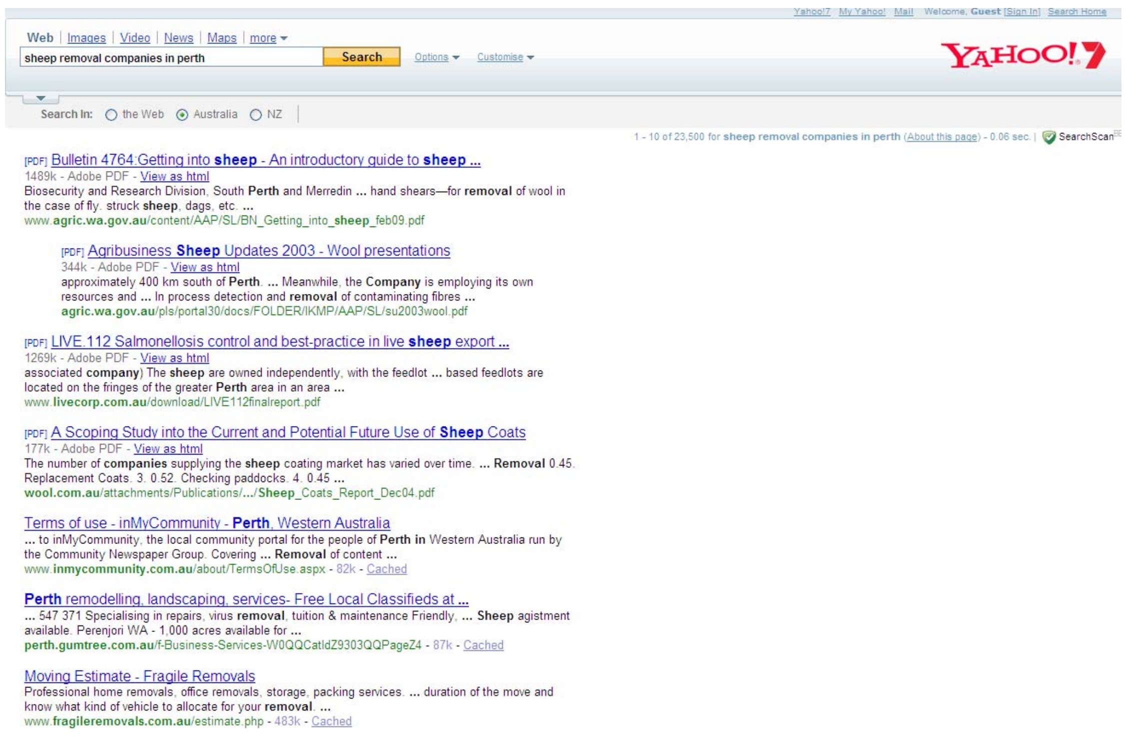
1.3.1 Issues with Service Retrieval with Commercial Search Engines

In this section, by means of a case study, we will analyse the service retrieval issues with commercial search engines.

John is a farmer who lives in Perth (capital city of Western Australia) and desires a sheep removal service provided by a local company, in order to help him to move sheep from Perth to City B. In addition, John intends to find out the ranking of all available sheep removal companies in Perth based on QoS.

From the perspective of the internet services, there are two primary categories of service search engines that can be found by John [8].

The first category is generic search engines, such as Yahoo! and Google^{TM}. For example, John can enter "*sheep removal companies in Perth*" into a generic search engine (here the example is Yahoo! – http://au.yahoo.com/). From the retrieved results from the search engine (Fig. 1.1), it is observed that most of the retrieved results do not match John's search intention – sheep removal companies in Perth, and the service information is difficult to be distinguished and identified from the results. Thus, it is asserted that the performance of the generic search engine is poor in this case study [8].


Yahoo!7 My Yahoo! Mail Welcome, Guest [Sign In] Search Home
Web | Images | Video | News | Maps | more
Search
Options
Customise
YAHOO!7
Search In: the Web Australia NZ

sheep removal companies in perth

1 - 10 of 23,500 for **sheep removal companies in perth** (About this page) - 0.06 sec. | SearchScan

[PDF] Bulletin 4764:Getting into **sheep** - An introductory guide to **sheep** ...
1489k - Adobe PDF - View as html
Biosecurity and Research Division, South **Perth** and Merredin ... hand shears—for **removal** of wool in the case of fly. struck **sheep**, dags, etc. ...
www.**agric.wa.gov.au**/content/AAP/SL/BN_Getting_into_**sheep**_feb09.pdf

[PDF] Agribusiness **Sheep** Updates 2003 - Wool presentations
344k - Adobe PDF - View as html
approximately 400 km south of **Perth**. ... Meanwhile, the **Company** is employing its own resources and ... In process detection and **removal** of contaminating fibres ...
agric.wa.gov.au/pls/portal30/docs/FOLDER/IKMP/AAP/SL/su2003wool.pdf

[PDF] LIVE.112 Salmonellosis control and best-practice in live **sheep** export ...
1269k - Adobe PDF - View as html
associated **company**) The **sheep** are owned independently, with the feedlot ... based feedlots are located on the fringes of the greater **Perth** area in an area ...
www.**livecorp.com.au**/download/LIVE112finalreport.pdf

[PDF] A Scoping Study into the Current and Potential Future Use of **Sheep** Coats
177k - Adobe PDF - View as html
The number of **companies** supplying the **sheep** coating market has varied over time. ... **Removal** 0.45. Replacement Coats. 3. 0.52. Checking paddocks. 4. 0.45 ...
wool.com.au/attachments/Publications/.../**Sheep**_Coats_Report_Dec04.pdf

Terms of use - inMyCommunity - **Perth**, Western Australia
... to inMyCommunity, the local community portal for the people of **Perth in** Western Australia run by the Community Newspaper Group. Covering ... **Removal** of content ...
www.**inmycommunity.com.au**/about/TermsOfUse.aspx - 82k - Cached

Perth remodelling, landscaping, services- Free Local Classifieds at ...
... 547 371 Specialising in repairs, virus **removal**, tuition & maintenance Friendly, ... **Sheep** agistment available. Perenjori WA - 1,000 acres available for ...
perth.gumtree.com.au/f-Business-Services-W0QQCatIdZ9303QQPageZ4 - 87k - Cached

Moving Estimate - Fragile Removals
Professional home removals, office removals, storage, packing services. ... duration of the move and know what kind of vehicle to allocate for your **removal**. ...
www.**fragileremovals.com.au**/estimate.php - 483k - Cached

Figure 1.1: Retrieved results of "*sheep removal companies in Perth*" from Yahoo search engine [8]

The reasons for the poor performance of the search engine can be concluded as follows [8, 11]:

- The search engine uses a traditional keyword-based search strategy without incorporating or taking into account Semantic Web technologies to assist it to fully understand the sense of the user's query words. Moreover, the search engine does not take into account the heterogeneous service requesters. These two reasons lead to the poor performance of the search engine in terms of precision.

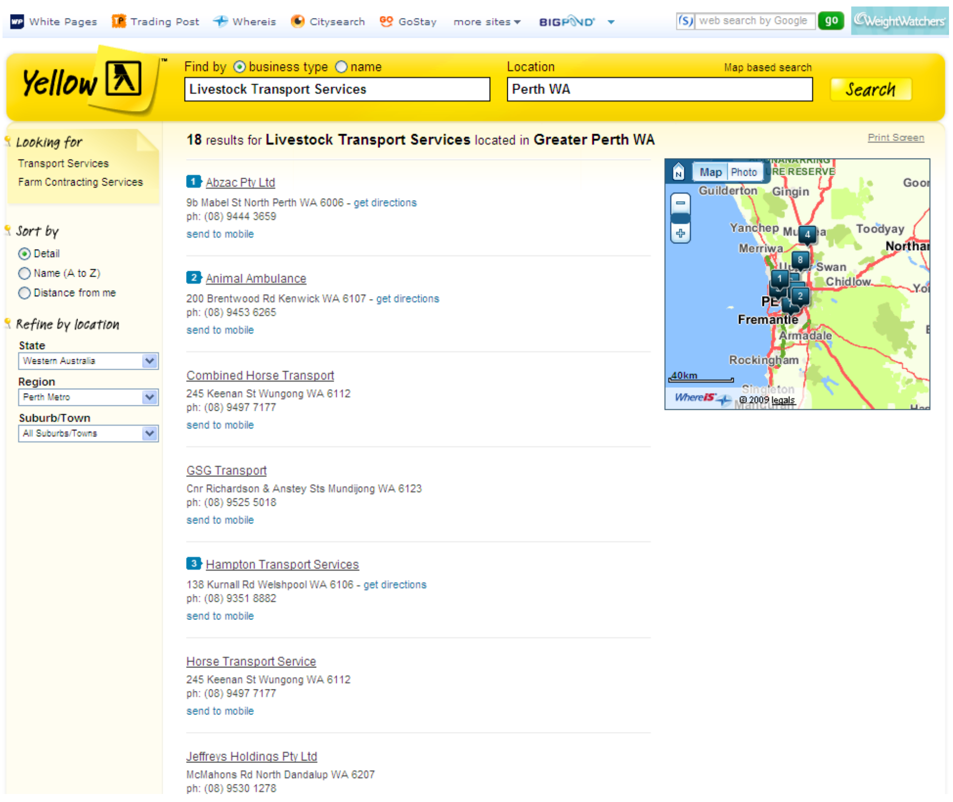
- The generic search engine is not specially designed for the purpose of service retrieval. It has no means of distinguishing the heterogeneous service information from other Web information. Hence, the search process has to be carried out against a much larger information source and most of the retrieved results are irrelevant with respect to services.

- The format of the retrieved service information is not standardized, which makes it difficult for users to read and comprehend the retrieved service information. The search engine does not offer any means for clarifying the heterogeneous service information.

- The generic search engine does not provide any methodologies for evaluating and ranking the quality or trustworthiness of retrieved Web information, taking into account the untrustworthy nature of Web information.

An enhanced approach is that John can gain access to the repository of a local business directory such as Yahoo! or Google^{TM} local search, online Yellowpages®. These local search engines (here the example is Australian online Yellowpages® – http://www.yellowpages.com.au/) can usually provide John with two options of service retrieval as follows [8]:

- One option is that John can browse businesses under the "*livestock transport services*" category in the location "*Perth WA*", by following the "*browse by category*" (Fig. 1.2). This method can provide John with more precise search results and structured service information. The disadvantage is that John will need to peruse the entire category of the website step-by-step, which is expensive in terms of time and effort. Moreover, if John does not have knowledge about the category, he can easily become lost in the searching process. This indicates that the search engine does not consider the heterogeneous nature of service requesters.



Find by [✓ business type] name: Livestock Transport Services
Location: Perth WA

Looking for
Transport Services
Farm Contracting Services

Sort by
[✓ Detail]
Name (A to Z)
Distance from me

Refine by location
State: Western Australia
Region: Perth Metro
Suburb/Town: All Suburbs/Towns

18 results for **Livestock Transport Services** located in **Greater Perth WA**

1 Abzac Pty Ltd
9b Mabel St North Perth WA 6006 - get directions
ph: (08) 9444 3659
send to mobile

2 Animal Ambulance
200 Brentwood Rd Kenwick WA 6107 - get directions
ph: (08) 9453 6265
send to mobile

Combined Horse Transport
245 Keenan St Wungong WA 6112
ph: (08) 9497 7177
send to mobile

GSG Transport
Cnr Richardson & Anstey Sts Mundijong WA 6123
ph: (08) 9525 5018
send to mobile

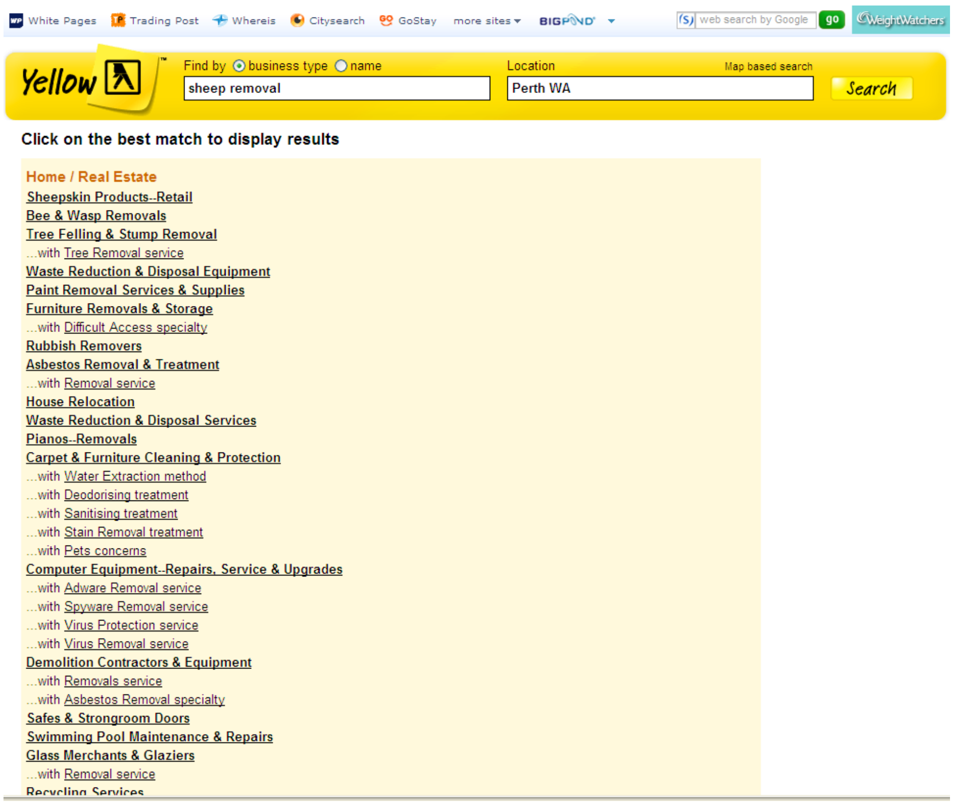
3 Hampton Transport Services
138 Kurnall Rd Welshpool WA 6106 - get directions
ph: (08) 9351 8882
send to mobile

Horse Transport Service
245 Keenan St Wungong WA 6112
ph: (08) 9497 7177
send to mobile

Jeffreys Holdings Pty Ltd
McMahons Rd North Dandalup WA 6207
ph: (08) 9530 1278

Figure 1.2: Businesses under the category of "*livestock transport service in Perth*" in Australian Yellowpages® website [8]

- Another option is that John can directly enter "*sheep removal*" into the business type box and "*Perth*" into the location box of the search engine provided by the website (Fig. 1.3). While this approach can save searching time, it also has its own disadvantages – the search engine cannot understand the user's query intention and thus returns non-relevant results. Similar to the generic search engines, the reason for this is that the local search engine does not propose any solutions to the problem of the heterogeneity of service requesters.



Find by [✓ business type] name: sheep removal

Location: Perth WA

Click on the best match to display results

Home / Real Estate
Sheepskin Products--Retail
Bee & Wasp Removals
Tree Felling & Stump Removal
...with Tree Removal service
Waste Reduction & Disposal Equipment
Paint Removal Services & Supplies
Furniture Removals & Storage
...with Difficult Access specialty
Rubbish Removers
Asbestos Removal & Treatment
...with Removal service
House Relocation
Waste Reduction & Disposal Services
Pianos--Removals
Carpet & Furniture Cleaning & Protection
...with Water Extraction method
...with Deodorising treatment
...with Sanitising treatment
...with Stain Removal treatment
...with Pets concerns
Computer Equipment--Repairs, Service & Upgrades
...with Adware Removal service
...with Spyware Removal service
...with Virus Protection service
...with Virus Removal service
Demolition Contractors & Equipment
...with Removals service
...with Asbestos Removal specialty
Safes & Strongroom Doors
Swimming Pool Maintenance & Repairs
Glass Merchants & Glaziers
...with Removal service
Recycling Services

Figure 1.3: Retrieved results from online Australian Yellowpages® search engine based on query words "sheep removal"

Apart from the lack of Semantic Web technology support, another limitation of the local service search engines is that John cannot determine which company performs best as a sheep transport service. Similarly, John cannot evaluate the performance of a service provider once he finishes an interaction with this service provider. The reason for this is that these search engines do not provide QoS methodologies for service ranking and service evaluation. Whilst some commercial search engines, such as Google^{TM} Local Business Centre, Yahoo! Local, Yellowpages^{TM} etc., provide a simple evaluation system (star-rating or comment) for users to evaluate QoS without criteria or with unified criteria, these evaluations cannot objectively indicate the QoS, owing to a lack of knowledge about the context-dependent nature of QoS evaluation criteria and QoS [10].

In this section, we discussed two popular commercial search engines for service retrieval. It should be noted that these commercial search engines are not meant to represent all the commercial search engines currently available.

As can be seen from the aforementioned, the commercial search engines do not provide a service retrieval function that takes into account the heterogeneous, context-dependent and dynamic nature of services, and the heterogeneous and dynamic nature of service providers and service requesters in the Digital Ecosystems environment.

1.3.2 Issues with Service Retrieval in Research Literature

Given the importance of service, in the research world much attention has been given to the issue of service retrieval. Nevertheless, the current service retrieval research primarily focuses on Web services and ignores the generic service information in the Web environment; on the other hand, there is vast generic service information in the Web without methodologies for discovery, annotation and classification. At the time of writing this thesis, significant research has been carried out in the field of semantic search, which makes use of Semantic Web technologies for disambiguating user queries and denoting user query intentions. However, little of the existing literature concentrates on semantic search in the generic service field; on the other hand, the mainstream commercial search engines cannot be used for genuine service retrieval. Further details regarding the two issues can be found in Chapter 2. From Chapter 4 to Chapter 9, we will present a solution which addresses this issue. In the next section, we identify several issues regarding service retrieval in Digital Ecosystems.

1.3.3 Issues with Service Retrieval in Digital Ecosystems

Subsequent to the discussion in Section 1.3.1 and Section 1.3.2, in this section we summarize some of the pressing issues regarding service retrieval in the Digital Ecosystems environment that need to be addressed, and that can be expressed as follows:

1. How can we identify the information regarding all available service advertisements and service provider profiles over the Internet from vast Web information?

2. How can we integrate, clarify and annotate all available service advertisements and service provider profiles over the Internet based on specific service domain knowledge?

3. How can a service requester, without relevant domain knowledge about his/her service request, precisely retrieve an available service advertisement by means of relevant service domain knowledge?

4. How can a service requester evaluate the trustworthiness of a service advertisement by means of context-specific QoS evaluation criteria?

5. How can a service requester rank the trustworthiness of service advertisements by context-specific QoS evaluation criteria?

6. How can service domain knowledge be addressed and duly updated to maintain its accuracy and to represent most people's understanding of domain knowledge?

7. How can a service advertisement be duly updated to reveal its present status?

8. How can a service provider profile be duly updated to reveal his/her present status?

1.4 Objective of the Thesis

The previous sections outlined the features of Digital Ecosystem services and some of the issues regarding service retrieval in the Digital Ecosystems environment. This thesis is a step in that direction as it presents solutions for some of the service retrieval issues in Digital Ecosystems by proposing a methodology to allow species to semantically discover, annotate, classify, retrieve, rank, evaluate and maintain service information in the Digital Ecosystems environment. The objectives of this thesis are summarized as follows:

1. To develop a methodology by which all available service advertisements over the Internet can be identified, annotated and classified by specific service domain knowledge, taking into account the heterogeneous and context-dependent nature of Digital Ecosystem services.

2. To develop a methodology by which all available service provider profiles over the Internet can be identified, annotated and classified by specific service domain knowledge, taking into account the heterogeneous nature of Digital Ecosystem service providers.

3. To develop a methodology by which a service requester can precisely retrieve a service advertisement by means of specific service domain knowledge, taking into account the heterogeneous nature of Digital Ecosystem service requesters.

4. To develop a methodology by which a service requester can evaluate the trustworthiness of a service advertisement by context-specific QoS evaluation criteria, taking into account the context-dependent nature of Digital Ecosystem services.

5. To develop a methodology by which a service requester can rank the service advertisements based on their trustworthiness according to context-specific QoS evaluation criteria, taking into account the context-dependent nature of Digital Ecosystem services.

6. To develop a methodology by which service domain knowledge can be addressed and duly updated to maintain its accuracy and to represent most people's understandings of domain knowledge, taking into account the dynamic nature of Digital Ecosystem services.

7. To develop a methodology by which a service advertisement can be duly updated to reveal its present status, taking into account the dynamic nature of Digital Ecosystem services.

8. To develop a methodology by which a service provider profile can be duly updated to reveal his/her present status, taking into account the dynamic nature of Digital Ecosystem service providers.

1.5 Scope of the Thesis

This thesis presents a methodology that will enable a service requester to precisely retrieve over the Internet a service advertisement with the highest level of trustworthiness.
It should be noted here that this thesis focuses only on semantically searching for generic services, since many service retrieval/discovery/matchmaking methodologies have been developed in the Web service field. Thus, in this thesis when we refer to "services", we mean generic services.

In addition, this research is not concerned with assessing the availability of services in different time slots. Whilst our proposed methodology enables service providers to duly update their service advertisements, we cannot guarantee that service providers can duly notify the availability of their services at any time.

1.6 Significance of the Thesis

To the best of our knowledge, at the time of writing, this thesis is the first and only attempt in the literature to semantically and qualitatively retrieve service information in the Digital Ecosystems environment, by taking into account the heterogeneous, context-dependent and dynamic nature of services, and the heterogeneous and dynamic nature of service providers and requesters. Specifically, the significance of this thesis arises from the following:

1. This thesis proposes a methodology by which all available generic service information over the Internet can be identified, annotated and classified by specific service domain knowledge, taking into account the heterogeneous and context-dependent nature of Digital Ecosystem services. To the best of our knowledge, such a methodology that focuses on discovering and annotating generic service information has not been proposed in the literature.

2. This thesis proposes a methodology by which a service requester can semantically and qualitatively retrieve a service advertisement by means of specific service domain knowledge, taking into account the heterogeneous nature of Digital Ecosystem service requesters. To the best of our knowledge, such a methodology that focuses on semantically and qualitatively searching generic service advertisements has not been proposed in the literature.

3. This thesis proposes a methodology by which a service requester can evaluate the trustworthiness of service advertisements and rank service advertisements based on their trustworthiness according to context-specific QoS evaluation criteria, taking into account the context-dependent nature of Digital Ecosystem services. To the best of our knowledge, such a methodology concerning Digital Ecosystem service trustworthiness measurement has not been proposed in the literature.

4. This thesis proposes a methodology by which service domain knowledge can be addressed and duly updated to maintain its accuracy and to represent most domain experts' understanding of domain knowledge, taking into account the dynamic nature of Digital Ecosystem services. To the best of our knowledge, such a methodology concerning Digital Ecosystem service domain knowledge evolution has not been proposed in the literature.

1.7 Plan of the Thesis

In this thesis, we provide a complete methodology for service retrieval in the Digital Ecosystems environment. In order to achieve its objectives, this thesis is organised in eleven chapters. In the section, we give a brief summary of each chapter.

Chapter 2: Chapter 2 provides an extensive review of the current literature on existing methods for semantic search. The problems associated with the current literature with regard to semantic search are identified in this chapter. Additionally, this chapter lists the problems that we intend to address in this thesis. The aim of this chapter is to illustrate that the problems that we intend to address via this thesis have not been previously addressed and resolved in the literature.

Chapter 3: Chapter 3 formally defines each of the problems that we intend to address in this thesis. Furthermore, here we present definitions of those terminologies that will be used to define the problems addressed in this thesis. Additionally, this chapter discusses the various research methodologies and we choose the one that is most pertinent to this research.

Chapter 4: Chapter 4 gives an overview of the solution for each of the issues identified in Chapter 3. Chapter 4 also contains pointers to the chapters containing the detailed solution to the identified research issues.

Chapter 5: Chapter 5 presents a service information discovery, annotation and classification methodology, by means of which service information embedded in the Web environment can be automatically discovered, and annotated by specific service domain knowledge. This methodology is presented by introducing the conceptual framework and workflow of a semantic crawler and its inclusive mathematical models.

Chapter 6: Chapter 6 proposes a service retrieval methodology, by means of which a service requester can precisely retrieve a service advertisement with the assistance of specific service domain knowledge. This methodology is presented by introducing the system architecture and workflow of a service search engine and its inclusive mathematical models.

Chapter 7: Chapter 7 presents a service concept recommendation methodology, by means of which a service requester without relevant knowledge about his/her service queries can be recommended with several service concepts for the query disambiguation. This methodology is presented by introducing the system architecture and workflow of a service concept recommendation system and its inclusive mathematical models.

Chapter 8: Chapter 8 presents a QoS evaluation and service ranking methodology, by means of which the trustworthiness extent of service advertisements can be measured by means of domain-specific QoS evaluation criteria, and service advertisements under the same contexts can be ranked based on their trustworthiness values. This methodology is presented by introducing the system architecture and workflow of QoS evaluation and service ranking system and its inclusive mathematical models.

Chapter 9: Chapter 9 provides a methodology for updating service domain knowledge and a methodology for service-provider-based service (description) metadata publishing, maintenance and classification. By means of the former, Digital Ecosystem service domain knowledge can be updated to maintain its accuracy and to represent most domain experts' understanding of domain knowledge. By means of the latter, service providers can duly update their profiles, publish and maintain their advertised service profiles, and more importantly, manually annotate service profiles by means of specific service domain knowledge. The two methodologies are presented by introducing their respective workflows.

Chapter 10: Chapter 10 describes the prototype that we engineered in order to validate the service retrieval methodology. For the mathematical models involved in the methodology, we validate their performance in two different service domains by means of the simulation approach; and we validate the workflows and functions involved in the methodology by running the prototype to test its functions.

Chapter 11: Chapter 11 concludes the thesis with a summary of the results achieved in this thesis, along with suggestions for future research direction.

1.8 Conclusion

In this chapter, we provided an introduction to Digital Ecosystems and the features of the services in Digital Ecosystems. We then presented a case study to examine the function of service retrieval in some of the commercial search engines, namely Yahoo! and Australian Yellowpages®. The shortcomings associated with the methods employed by these commercial search engines were pointed out and subsequently discussed. Specifically, it was noted that these commercial search engines do not provide a particular function for service retrieval which takes into account the heterogeneous, context-dependent and dynamic nature of services, and the heterogeneous and dynamic nature of service providers and service requesters in the Digital Ecosystems environment. Moreover, it was noted that similar shortcomings were associated with the research being carried out in relation to service retrieval and semantic search. Subsequently, some of the pressing issues pertaining to service retrieval in the Digital Ecosystems environment were identified. Next, the objectives of undertaking this study were presented and discussed, followed by a description of the scope and significance of this thesis in enabling precise and trustworthy service retrieval in the Digital Ecosystems environment. Finally, the plan of this thesis was presented.

In the next chapter, we present an overview of the existing literature on semantic search. The objective is to ensure that the problems that we intend to address via this thesis have not been addressed previously.

1.9 References

1. Boley, H., Chang, E.: Digital Ecosystems: Principles and semantics. Proceedings of the 2007 Inaugural IEEE International Conference on Digital Ecosystems and Technologies (DEST 2007). IEEE, Cairns, Australia (2007) 398-403

2. Briscoe, G., Wilde, P.D.: Digital Ecosystems: Evolving service-oriented architectures. Proceedings of the 1st International Conference on Bio Inspired Models of Network, Information and Computing Systems. ACM, Cavalese, Italy (2006) 17

3. Chang, E., West, M.: Digital ecosystems and comparison to existing collaboration environment. WSEAS Transactions on Environment and Development **2** (2006) 1396-1404

4. Chang, E., West, M.: Digital Ecosystem - A next generation of the collaborative environment. Proceedings of the 8th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2006). ACM, Yogyakarta, Indonesia (2006) 3-24

5. Corallo, A., Passiante, G., Prencipe, A.: The Digital Business Ecosystem. Edward Elgar, Cheltenham Glos, UK (2007)

6. Dini, P., Rathbone, N., Vidal, M., Hernandez, P., Ferronato, P., Briscoe, G., Hendryx, S.: The digital ecosystems research vision: 2010 and beyond. Creative Commons, Stanford, California, USA (2005), http://www.digital-ecosystems.org/events/2005.05/de_position_paper_vf.pdf

7. Dong, H., Hussain, F.K., Chang, E.: A QoS-based service retrieval methodology for digital ecosystems. International Journal of Web and Grid Services **5** (2009) 261-283

8. Dong, H., Hussain, F.K., Chang, E.: Semantic service retrieval and QoS measurement in the digital ecosystem environment. Proceedings of the 4th International Conference on Complex, Intelligent and Software Intensive Systems (CISIS 2010). IEEE Computer Society, Krakow, Poland (2010) 153-160

9. Dong, H., Hussain, F.K., Chang, E.: A framework for discovering and classifying ubiquitous services in digital health ecosystems. Journal of Computer and System Sciences **In Press**

10. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

11. Dong, H., Hussain, F.K., Chang, E.: A service search engine for the industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

12. Moore, J.F.: Predators and prey: a new ecology of competition. Harvard Business Review **71** (1993) 75-86

13. Nachira, F., Nicolai, A., Dini, P., Louarn, M.L., Leon, L.R.: Digital Business Ecosystems. European Commission Information Society and Media, Luxembourg (2007)

Chapter 2 – Literature Review

2.1 Introduction

In this chapter, we present an overview of the existing literature on semantic search. In 2003, Guha et al. [68] provided a new terminology – "*semantic search*" for the field of Web search, which is distinct from the traditional information retrieval methods. The purpose of the semantic search is to assist users to denote their search intentions and assist search engines to understand the meaning of users' queries in terms of Semantic Web technologies. We understand this concept from a broader sense, in that Semantic search refers to using semantic technologies for information retrieval. In this chapter, we discuss and evaluate four important application areas of semantic search that relate to our research, namely: semantic crawlers, semantic service discovery, semantic search engines, and semantic similarity models.

The rest of the chapter is organized as follows: in Section 2.2 we analyse several approaches in the field of semantic crawlers; in Section 2.3 we present the existing literature in the field of semantic service discovery; in Section 2.4 we discuss the researches within the domain of semantic search engines; in Section 2.5 we evaluate the contemporary semantic similarity models; in Section 2.6 we provide an integrative view of all the existing proposals in the literature; finally, Section 2.7 concludes this chapter.

2.2 Semantic Crawlers

A crawler is an agent which can automatically search and download webpages [23, 32]. Nowadays, the research of crawlers moves closer to the Semantic Web, along with the an increase in the appearance of XML/RDF/OWL files and the rapid development of ontology mark-up languages [82, 92].

A semantic (Web) crawler refers to a series of Web crawlers designed for harvesting Semantic Web documents [55]. Semantic Web documents are those marked by the ontology mark-up languages (e.g. RDF(S), XML, OWL etc.).

In general, semantic crawlers can be categorized into three primary types: metadata abstraction crawlers, semantic focused crawlers, and metadata harvesting crawlers. This categorization is based on the classification framework proposed by Dong et al. [49, 50, 52]. Metadata abstraction crawlers are those which are able to abstract information from normal Web documents, and generate metadata based on the abstracted information [49]. Semantic focused crawlers refer to the crawlers which are able to use ontologies to define their crawling scope [52]. Metadata harvesting crawlers refer to the crawlers which are able to directly retrieve from the Internet Semantic Web documents and Semantic Web information pieces annotated by ontology mark-up languages.

In fact, these crawlers have no clearly defined boundaries. Fig. 2.1 shows the boundaries of semantic crawlers from the perspective of functionalities. As can be seen, each type of crawler may have some properties or features in common with other types. For example, metadata harvesting crawlers can also have the features of metadata abstraction and semantic focusing, and thus, it has the variation of metadata harvesting and abstraction crawlers, focused metadata harvesting crawlers, and focused metadata harvesting and abstraction crawlers. There are seven types of semantic crawlers summarized in our research, including:

- Three main types
  - Metadata harvesting crawlers
  - Semantic focused crawlers
  - Metadata harvesting crawlers
- Four cross-bounded types
  - Metadata abstraction focused crawlers
  - Metadata harvesting and abstraction crawlers
  - Metadata harvesting focused crawlers
  - Metadata harvesting and abstraction focused crawlers

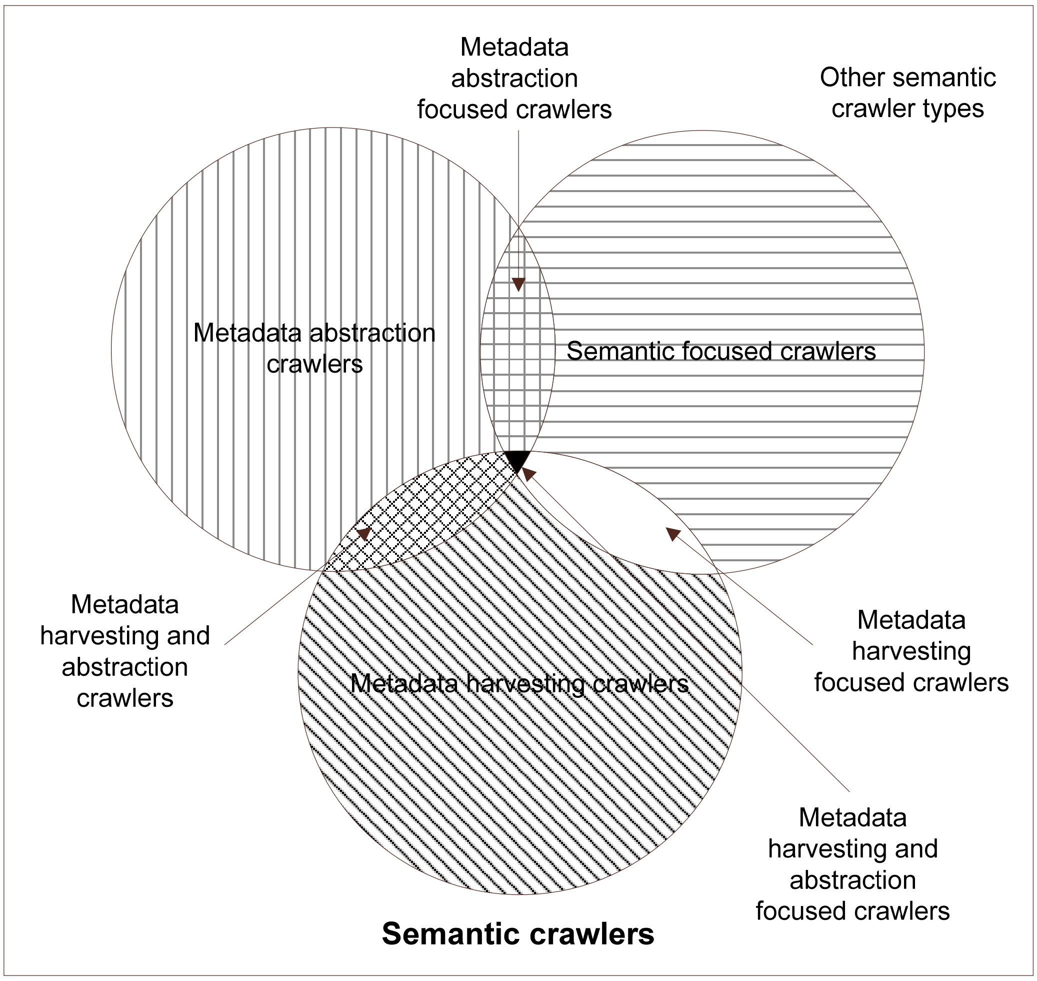

Metadata abstraction focused crawlers
Other semantic crawler types
Metadata abstraction crawlers
Semantic focused crawlers
Metadata harvesting and abstraction crawlers
Metadata harvesting focused crawlers
Metadata harvesting crawlers
Metadata harvesting and abstraction focused crawlers
Semantic crawlers

Figure 2.1: Boundaries of semantic crawlers

There may well be other types of semantic crawlers that have not been included in the semantic crawlers identified in this survey.

Whilst we define the main types of semantic crawlers, in actual fact, not all crawlers have been developed at present. In this section, we briefly introduce the following types of crawlers that have available examples – metadata abstraction crawlers, semantic focused crawlers, metadata harvesting crawlers, metadata abstraction focused crawlers, and metadata harvesting and abstraction crawlers. Furthermore, the first two types of crawlers have their subtypes, as shown in Fig 2.2.

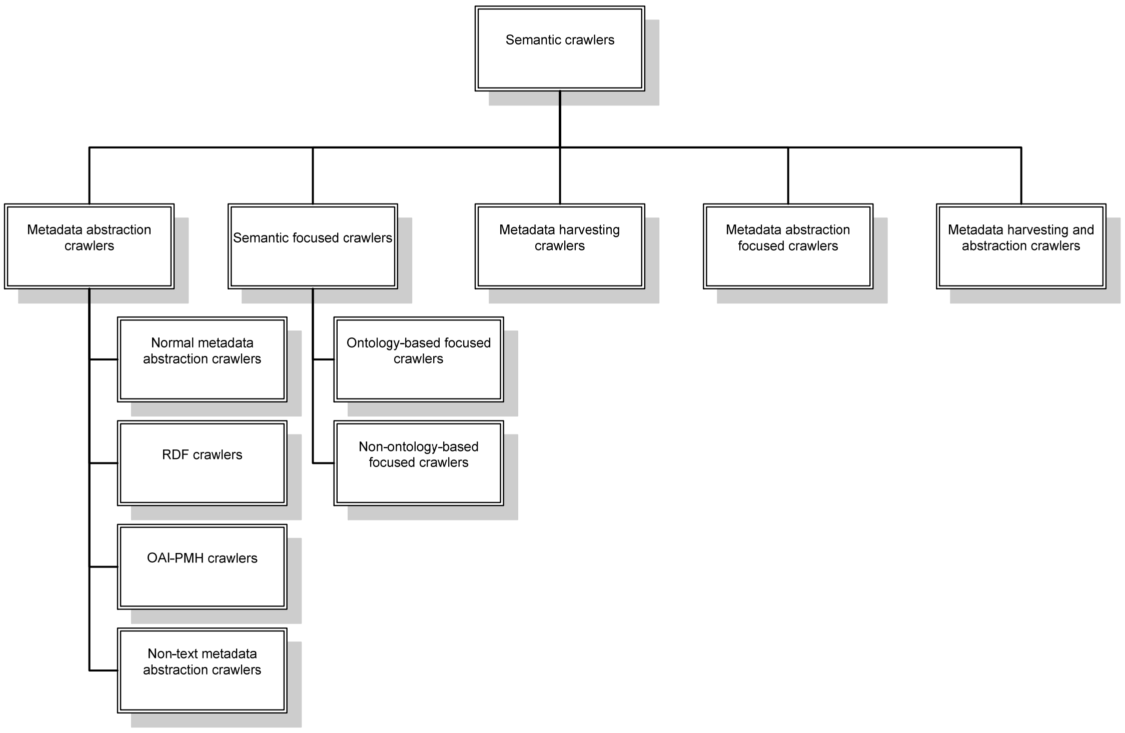

Semantic crawlers
Metadata abstraction crawlers
Semantic focused crawlers
Metadata harvesting crawlers
Metadata abstraction focused crawlers
Metadata harvesting and abstraction crawlers
Normal metadata abstraction crawlers
RDF crawlers
OAI-PMH crawlers
Non-text metadata abstraction crawlers
Ontology-based focused crawlers
Non-ontology-based focused crawlers

Figure 2.2: Further classification of semantic crawlers

In this section, we present and subsequently discuss five types of semantic crawlers from the current literature. Each subsection deals with a different type of existing crawler. Section 2.2.1 deals with metadata abstraction crawlers; Section 2.2.2 presents and discusses semantic focused crawlers; Section 2.2.3 examines metadata harvesting crawlers; Section 2.2.4 presents and discusses metadata abstraction focused crawlers; and Section 2.2.5 describes and discusses the existing metadata harvesting and abstraction crawlers.

2.2.1 Metadata Abstraction Crawlers

According to their distinctive features, metadata abstraction crawlers can be sub-categorized into four groups of crawlers: metadata abstraction crawlers, RDF crawlers, OAI-PMH crawlers and non-text metadata abstraction crawlers.

Normal metadata abstraction crawlers do not have distinct functions, apart from the function of metadata abstraction [49].

Davulcu et al. [37] proposed an OntoMiner system, with the purpose of organizing the overlapping websites provided by users, based on automatically generated ontologies. First of all, a Web crawler fetches all webpages from a given website. A Semantic

Partitioner then analyses the labels in the webpages and builds a hierarchical tree of labels. Next, a Taxonomy Miner clusters the frequent labels into several concepts as the concepts' attributes, by means of the Frequent Tree Mining algorithm, in order to build a conceptual hierarchy. For each concept in the hierarchy, an Instance Miner associates the concept with the potential webpage instances, and computes the labelled and unlabeled attribute values for the instances.

Panayiotou and Samaras [148] proposed a personalized knowledge portal – mPERSONA – for the collaboration between wireless users and content providers. When the content providers join in this system, they need to submit their URLs and characteristic keywords. To semanticize the content providers' websites, a specialized crawler is designed to convert the contents of each page to metadata, in order to build a semantic tree. Each node of the tree is represented by characteristic keywords. A Thesaurus is used to find the synonyms for each keyword, in order to enrich each node's semantic meaning. Thereafter, the metadata fetched by the crawler are linked to the nodes, which are groups of topics, thereby clarifying the semantic meaning of each node.

Topic maps are a semantic technology which classifies knowledge by topics. Roberson and Dicheva [158] propose an approach to create the drafts of topic maps for websites. A crawler is used to download all webpages in a website, and then extract the semantic information regarding topic by means of a set of heuristics.

Shimazu and Arisawa [170] proposed a content management system for interdisciplinary metadata exchange. A crawler is used to collect source files from a local network. A Natural Language Analyser is used to parse, identify and annotate the name entities from the source files. Thereafter, metadata in the format of Dublin Core are abstracted from the annotated source files, based on the method of 5W1H (when, who, what, where, why and how). Finally, these metadata are indexed and stored in index files for further content search.

The disadvantages of these metadata abstraction crawlers are as follows:

1. They use data mining techniques for information exploitation. Some data mining techniques need to define domain-specific rules and heuristics, which could be time-consuming.

2. They mostly use non-semantic techniques for metadata classification. This could give rise to errors when using non-semantic techniques to process huge amounts of semantic information.

3. They do not define crawling boundaries apart from natural boundaries, e.g., within a website. This often causes information redundancy when dealing with large-scale websites.

4. They do not propose any means by which metadata can be generated from service fields.

5. They do not provide any means for qualitatively ranking the crawled service information.

6. Most of them are still in the conceptual phase without enough practical application in the real environment.

The term *RDF crawler* originates from Ontobroker, and is a series of crawlers with the objective of generating RDF metadata [39]. All of the following systems use the RDF crawler as their technical backbone to achieve different goals.

Decker et al. [39] proposed an Ontobroker system, with the purpose of extracting, reasoning and generating RDF-annotated metadata. The system has five major components. First of all, the domain-specific ontologies are stored in a knowledge base. Subsequently, an Ontocrawler is designed to extract the formal knowledge from HTML webpages. Two different approaches are implemented here. For the similarly structured HTML files, a wrapper is used to generate their formal descriptions, by means of referring to an ontology in the knowledge base; and for the specially structured HTML files, an annotation language is used. The descriptions are reasoned by an inference engine. Next, a RDF-Maker converts the reasoned descriptions to the metadata in the form of RDF. Finally, a query interface is designed to allow users to browse the ontologies and metadata.

Handschuh and Staab [73-76] designed a framework of metadata creator – CREAM. A RDF crawler is utilized to find references for created metadata, with the purpose of avoiding duplication. With the CREAM, when the metadata creator wants to determine whether or not an instance already exists, the RDF crawler retrieves the instance from the local knowledge base, which stores the RDF files harvested from the Semantic Web. If a URI relevant to the instance is returned by the RDF crawler, the creator will then be aware that the relational metadata has been created.

Stojanovic et al. [180, 182] proposed a platform – SEAL, for semantic portal development. Ontobroker is the backbone of the platform, which works as a middleware between the Web server and knowledge warehouse for the purpose of RDF generation, knowledge portal template generation, ontology query and ranking and so forth. A RDF crawler is used to build the knowledge warehouse by generating RDF documents from the Internet.

OntoKhoj is a Semantic Web portal for ontology searching, ranking and classification. The task of ontology searching is executed by an RDF crawler, realized in the platform of Java or Linux. The crawler can retrieve RDF codes embedded in HTML and DAML+OIL documents, by means of physical links (URIs or URLs). The fetched ontologies are then stored in a repository, and ranked based on an OntoRank algorithm, where the ontology's weight depends on the number and the weight of incoming references from other ontologies, and the total number of reference and ontologies. Meanwhile, the weight of references is in light of their reference types. Then the ontologies are clustered by means of DMOZ Open Directory Project model, which is a set of manually classified data [150].

The limitations of these RDF crawlers can be concluded as follows:

1. They generate metadata by means of domain-specific ontologies. The design of domain-specific ontologies could be costly in terms of time.

2. The performance of webpage classification relies heavily on the quality of ontologies.

3. No ontology evolution mechanism is provided by these crawlers.

4. The ontologies are designed by domain experts and sometimes may not match actual users' subjective perceptions of domain knowledge.

5. They are able to generate only RDF-annotated metadata, which do not support an OWL-based knowledge environment.

6. They do not propose any means by which metadata can be generated from service fields.

7. They do not provide any means for qualitatively ranking the crawled service information.

8. Most of them are still in the conceptual phase without enough practical application in the real environment.

OAI-PMH (Open Archives Initiative Protocol for Metadata Harvesting) is a HTTP-based protocol which can be used to retrieve XML metadata [143]. The following crawlers utilize this protocol in order to harvest XML metadata.

Nelson et al. [143] utilized the OAI-PMH to enhance the competence of normal Web crawlers, with the purpose of crawling metadata from webpages. OAI-PMH is based on a data model composed of three layers – resource, item and records. The OAI-PMH identifier can identify items which are "*entry point to all records (metadata) pertaining to the resource (Web documents)*". To enhance the accuracy of metadata searching, the crawler uses an XML-based complex object format – MPEG-21 Digital Item Declaration Language (MPEG-21 DIDL) for formatting the retrieved digital objects (items), which consists of the concept of item (a group of items/components), component (a group of resource), resource (an individual data stream), container (a group of containers/items), and descriptor/statement (information pertaining a item, a component or a container). In addition, Smith and Nelson [172] proposed to use OAI-PMH to convert Web information into the format of CRATE – a self-contained preservation-ready version of the resource, in which an entity is composed of resource and its associated metadata in the format of XML.

The challenges of the OAI-PMH crawlers are as follows:

1. They do not propose any means for generating ontology mark-up languages (RDF or OWL)-based metadata. The XML metadata need to be further semanticized to make them available in the Semantic Web environment.

2. They do not propose any means for semantically classifying metadata, which may give rise to errors when searching for metadata.

3. They do not propose any means by which metadata can be generated from service fields.

4. They do not provide any means for qualitatively ranking the crawled service information.

5. Most of them are still in the conceptual phase without enough practical application in the real environment.

Some metadata abstraction crawlers are designed not only to retrieve plain text documents. The following crawlers exploit Semantic Web technologies to enhance their ability to search non-text files.

Liu et al. [123] proposed a media agent for managing personal multimedia files. An online crawler and an offline crawler are introduced in the system, in order to collect the metadata regarding multimedia files. The difference between the two crawlers is that the former can work when a user is operating an online multimedia file; and the latter works only according to a user's predefined preference data, when the user is offline. Both of the crawlers contain three sub-components – semantic collection, features extraction and multimedia file description indexing. The semantic collection is for collecting the URLs and semantic descriptions of multimedia files. The latter are extracted from the titles, surrounding texts of the multimedia objects. The feature extraction is used to extract feature keywords from the semantic descriptions. Subsequently, each multimedia file description is seen as a collection of keywords, and thus can be ranked by means of the tf-idf algorithm from the Vector Space Model (VSM).

Liu et al. [124, 125] proposed a specific table search engine – TableSeer. A table crawler is used in the system to crawl PDF documents with tables in a digital library. Then, a Doc Classifier classifies the documents into six categories and discards the documents without tables. For each identified table, a table metadata is created. And a specific table rank algorithm – Table Term Frequency – Inverse Table Term Frequency (TTF-ITTF) is used to index the metadata.

The shortcomings of the non-text metadata abstraction crawlers are as follows:

1. They do not propose any means for processing the information extracted from text descriptions of non-text files that may exist noise information.
2. They do not propose any means for semantically classifying the generated metadata, which may affect the precision of the metadata search.
3. They do not define crawling boundaries except for natural boundaries, e.g., within a website. This often causes information redundancy when dealing with large-scale websites.
4. They do not propose any means by which metadata can be generated from service fields.
5. They do not provide any means to qualitatively rank the crawled service information.
6. Most of them are still in the conceptual phase without enough practical application in the real environment.

2.2.2 Semantic Focused Crawlers

Focused (topical) crawlers are a group of distributed crawlers that specialize in certain specific topics [6]. Each crawler analyses its topical boundary when fetching webpages. Semantic focused crawlers are types of crawlers that use Semantic Web technologies [52]. We briefly classify them into two sub-categories – ontology-based focused crawlers and non-ontology-based focused crawlers.

Generally speaking, ontology-based focused crawlers are a series of focused crawlers which utilize ontologies to link the fetched Web documents with the ontological concepts (topics), with the purpose of organizing and categorizing Web documents, or filtering irrelevant webpages with regards to the topics [50, 132].

Ehrig and Maedche [55] proposed a semantic focused crawling framework. They divided the focused crawling process into two interconnected cycles. The first is an ontology cycle driven by human engineers, which provides domain ontologies and crawling outputs (ontology-annotated webpages) to users. The second is a crawling cycle driven by Web crawlers, which provides inputs (webpages) for the system and connects inputs with relevant ontology concepts. They used three steps for relevance computation as follows: 1) building references between terms in webpages and terms in ontologies; 2) computing the relevance of terms by consider their three relative relations – same class/subclass and other relations; and 3) summarizing the scores of all related terms to obtain a total similarity value. Analogously, Fang et al. [56] proposed a similar ontology-based Deep Web sources focused crawler framework. The particular features of this framework are that a set of classifiers is employed to guide its search. First, a link classifier is employed to analyse the relation between URLs and ontology concepts by a Navies Bayes algorithm. Second, a page classifier is used to compute the similarity between webpages and ontology concepts by a Navies Bayes algorithm. Finally, a form classifier is designed to filter out non-searchable forms. The relevance computation uses Ehrig and Maedche's approach [55]. Similarly, Batsakis et al. [7] proposed a semantic crawler. First of all, terms similar to topic terms are retrieved from the WordNet in order to semantically describe and extend crawling topics. Later, the semantic similarity between terms in topics and webpages is computed by referring to their relations in the WordNet.

Yuvarani et al. [206] proposed a new generation of focused crawler – LSCrawler – by using ontologies to analyse the semantic similarity between URLs and topics. With the LSCrawler, an ontology base is built to store ontologies. For each query keyword, a Relevant Ontology Extractor retrieves the ontology base to find the compatible ontology. Then the matched ontology is passed to a Crawler Manager. Meanwhile, a Seed Detector sends the keyword to the three most popular search engines, and returns the retrieved seed URLs to the URL Buffer of the Crawler Manger. Based on the matched ontology and the retrieved URLs, the Crawl Manager then generates a multi-threaded crawler to fetch webpages by these URLs. Meanwhile, a Busy Server is configured to prevent repeatedly visiting URLs. The fetched webpages are then stored into a Document Repository, and the fetched URL database is updated. Following this, a Link Extractor extracts all URLs and their surrounding texts from the fetched webpages, and sends them to a Hypertext Analyser. Meanwhile, the Porter Stemmer algorithm is used to remove stop keywords and extract terms from the texts. The Hypertext Analyser then removes the URLs found in the fetched URL Database, and the extracted terms are matched with the concepts in the ontology, in order to determine the relevance of webpages to the keyword. Based on the relevance values, the URLs are ranked and then stored in the URL Repository for further visit. Tane et al. [188] proposed a new ontology management system – Courseware Watchdog. One important component of the system is an ontology-based focused crawler. By means of the crawler, a user can specify his/her preference, by assigning weights to the concepts of an ontology. By means of the interrelations between concepts within the ontology, the weights of other concepts can be calculated. Once a webpage has been fetched, its text and URL descriptions are matched with the weighted ontological concepts. Thus, the weights of the webpage and its URLs are measured, ranked and clustered according to the concepts. In addition, the webpage relations can be viewed by linking the webpages to the ontology concepts that appear in the webpages. Similarly, Kozanidis [103] proposed an ontology-based focused crawler in which a topical ontology is built. The crawler then analyses the relevance of the ontology concepts to the downloaded webpages by computing the occurrence of terms within them. Ardö [4] used the similar technologies to collect ontology-classified Web documents for a Superpeer Semantic Search Engine – ALVIS.

Ganesh [60] proposed an association metric, with the purpose of optimizing the order of visited URLs for Web crawlers. For each URL, an association metric evaluates its semantic content based on a reference domain-specific ontology. In addition, the metric of URL can analyse the strength of the link between parent and children webpages after the latter have been downloaded, in order to refine it.

THESUS aims to organize online documents by linking their URLs to hierarchical ontology concepts, which are seen as thematic sub-sets. A Web crawler is used in the document acquisition component of the system. The mechanism of this crawler is as follows: first, the crawler extracts the URLs and their descriptive texts from the initial set of documents; and then the descriptive text of one URL is matched with one of the ontological concepts, and the URL is linked to the concept. A threshold of maximum times of recursions or maximum number of documents is set as an ending requirement [70].

The technology of ontology-based focused crawlers has also been applied to several particular fields. Toch et al. [191] proposed a semantic search engine for the approximate Web service retrieval. A Web crawler is used here with the purpose of discovering, analysing and indexing the semantic descriptions of Web services. The premised working environment is the ontology-based Web services with semantic properties, such as input and output. In the e-commerce field, Huang et al. [88] proposed a semantic focused crawler for retrieving e-commerce information. The authors designed an e-commerce ontology by which relevant Web documents can be classified.

Su et al. [184] proposed an ontology-based focused crawler which uses an ontology learning approach to compute the real-time weights between ontology concepts and topics when computing the relevance between URLs and topics. The weight computation takes into account the number of URLs crawled, the number of URLs crawled when a topic occurs, the number of webpages crawled when a concept occurs, and the number of webpages crawled when a concept and a topic co-occur. Zheng et al. [209] provided a similar framework. The major difference is that they used an artificial neural network (ANN) for ontology concept-based crawling topic classification. Luong et al. [131] designed a Web crawler for ontology learning. First of all, the crawler is used to fetch Web documents based on initial ontologies. For each ontology concept, a Support Vector Machine (SVM) model is used to filter the irrelevant terms appeared in its associated Web documents to the concept. The reserved terms can then be used to enrich the vocabulary of each concept.

The drawbacks of these ontology-based focused crawlers are as follows:

1. The crawlers do not propose a means for generating metadata by annotating extracted features from Web documents.

2. The design of ontologies needs the participation of domain experts and IT staff, as well as long-term validation, which is costly in terms of labour and time.

3. The performance of webpage classification relies heavily on the quality of ontologies.

4. No ontology evolution mechanism is provided by these crawlers.

5. The ontologies may not always match users' subjective perceptions of domain knowledge.

6. The extraction of descriptive texts for URLs may generate errors without enough semantic supports.

7. They do not provide any means for qualitatively ranking the retrieved service information.

8. Most of the crawlers are still in the conceptual phase without sufficient testing results to consolidate this approach.

The non-ontology-based focused crawlers utilized Semantic Web technologies other than ontologies for metadata classification. Thus, we cluster them for convenience of description.

Can and Baykal [25] proposed a medical search engine – MedicoPort – with the use of a topical Web crawler – Lokman. The Lokman collects medical information while limiting the scope of linking URLs. By means of the concepts from a Unified Medical Language System (UMLS), the Lokman can identify the links relevant to the medical domain. For each fetched document, a Document Parser extracts the links from it. For each fetched webpage, its relevance value to the UMLS concepts is estimated, based on the concept frequencies, concept weights, and the relevance value of its contained URLs. The URL relevance values are evaluated by a Link Estimator, based on the relevance between the texts within the URLs and the UMLS concepts. Then a URL Frontier determines the order of URL queue, based on their relevance values. The Lokman then fetches the URLs within the URL queue. The performance of the Lokman is tested by comparing two re-evaluation algorithms – IncrementValues which regards the sum of link relevance values for a link as the link value, and GetGreater which regards the maximum value as the link value. Two situations, which are direct links included and excluded out of the seed URLs, are tested by means of the two re-evaluation algorithms. In comparison with a simple best search crawler, the Lokman shows significant improvement in both situations.

Liu et al. [119, 120] proposed a learned user model-based approach to assist focused crawlers to predict relevant links based on users' preferences. Three components are involved in the architecture: User Modelling, Pattern Learning, and Focused Crawling. In the User Modelling, the system observes the sequence of user-visited pages with regards to a specific topic. A Web graph is drawn, which consists of nodes that represent the user-visited webpages and edges that represent the links among the webpages, in order to analyse user browsing pattern. In addition, the nodes are highlighted when users regard them as relevant. In the Pattern Learning, the Latent Semantic Indexing (LSI) model is adopted to cluster the documents into several groups, and to reveal the topic for each cluster and the relationship between the topics. Meanwhile, an EM algorithm is used to optimize the clusters. Then a Hidden Markov Model (HMM) algorithm is used to estimate the likelihood of the topics directly or indirectly leading to a target topic. In the Focused Crawling, a focused crawler downloads the page linked to the first URL in its URL queue, and computes the page's reduced LSI representation. It then downloads all the children pages and clusters them by means of the K-Nearest Neighbourhood algorithm to obtain the corresponding Visit Priority Value based on the learned HMM. In comparison with a Best-First search crawler, the crawler shows significant advantage in terms of precision.

Cesarano et al. [28] proposed an agent-based semantic search engine. The query keywords are sent to a traditional search engine and the retrieved URLs are returned. One of the components of the search engine – Web Spider – can download all pages by URLs and then visit all children pages in the same website, which traditional search engines cannot reach. The Web Spider uses a Web Catcher which follows links to visit webpages. Then the webpages are stored in a Web Repository, and the unvisited links parsed from the webpages are visited next time. The whole crawling procedure stops when a predefined depth parameter is reached. Then a Document Postprocessor extracts the useful information for each downloaded page, including the title, contents and description; and a Miner Agent ranks these pages according to the similarities between the pages' information and a user-predefined search context. The tool used for computing similarity values is the group of ontologies stored in a Semantic Knowledge Base, which has weighted relations between concepts.

Zhuang et al. [211] proposed to use publication metadata to guide focused crawlers to collect the missing information in digital libraries. The whole procedure is as follows: when a request for retrieving the publications in a specific venue is sent by a user, a Homepage Aggregator queries a public metadata repository, and returns some Metadata Heuristics for a focused crawler to locate the authors' homepages, and also returns a list of URLs to a Homepage URL Database; and then the focused crawler fetches the publications by means of the seed URLs and stores them in a Document Database.

Batzios et al. [8] proposed a vision of crawler – BioCrawler – working in the environment of Semantic Web. BioCrawler extends from the focused crawler, which is a group of distributed crawlers over the Web, which is seen as an entity of "vision, moving, communication abilities", and an up-to-date knowledge model when browsing Web content. Vision is the scope of domains which one BioCrawler can visit, in the form of webpage link vectors. Thus, BioCrawlers' movement is controlled by their visions. A Rule Manager agent is configured to determine the best rule (route) upon a crawler's request, based on the strength parameter of each available route plan. The knowledge model mechanism in BioCrawler is composed of a classifier that stores the information regarding rules, and a classifier evaluator which calculates the amount of semantic content grabbed by following the rules, also called the rules' strength.

Some challenges of the non-ontology-based focused crawlers are concluded as follows:

1. The non-ontology-based focused crawlers mostly do not have the function of metadata abstraction.

2. No experiment has been provided to evaluate the efficiency of these crawlers compared with Web crawlers.

3. They do not propose any means by which metadata can be generated from service fields.

4. They do not provide any means to qualitatively rank the crawled service information.

5. Most of the crawlers are still in the conceptual phase without sufficient testing results to consolidate this approach.

2.2.3 Metadata Harvesting Crawlers

Slug is a Web crawler for harvesting RDF documents from the Internet. It supports multi-thread retrieval of data via a fixed list of hyperlinks. The controller is responsible for a list of tasks which are carried out by a number of worker instances. The controller uses a factory class to create worker instance at application start-up and on demand during application processing, and each worker runs as a thread coordinated by the controller, which works as a master-slave design pattern. Then the crawled results are processed through the multi-stages that are abstracted by the task itself by a producer-consumer design pattern, which means that each work produces results handled by a customer instance. This allows alternate processing behaviour to be substituted in the framework without changing the core functionality that deals solely with multi-threaded retrieval of content via HTTP. Finally, to allow for modular extension of the consuming of task results, the framework provides an implementation of the Consumer interface that co-ordinates several other Consumer instances. This delegation model allows multiple Consumer instances to be involved in processing a single Response generated by each Worker [48].

The limitations of the metadata harvesting crawlers are as follows:

1. They do not propose any means by which metadata can be generated by extracting features from non-semantic Web documents.

2. They do not propose any means by which harvested metadata can be semantically classified by domain-specific ontologies.

3. They do not propose any means by which metadata can be generated from service fields.

4. They do not provide any means to qualitatively rank the crawled service information.

5. They are still in the conceptual phase without enough practical application in the real environment.

2.2.4 Metadata Abstraction Focused Crawlers

Metadata abstraction focused crawlers are the focused crawlers that can abstract metadata from the fetched Web documents, in addition to fetching relevant documents [52].

Yang [202, 203] proposed a Semantic Web crawler program working in an ontology-based Web environment. First of all, a knowledge base is designed, which stores ontologies. A Web crawler then obtains all data from a given Web site. Next, the Web information is modelled, which contains a website profile and all associated webpage profiles. Each profile includes the basic description, static information, and ontological information regarding a corresponding webpage. To realize this objective, a DocExtractor program is designed to extract the basic information from a webpage for the first section, calculating statistical data for the second section and remove all HTML tags. Subsequently, an OntoAnnotator is used to annotate the Web metadata for the third section. Within the DocExtractor, a HTML Analyser is utilized to analyse the webpages from a DocPool which contains the webpages from the retrieved website, then extracts the information regarding URLs, titles, anchors and headings, and calculates the statistical data regarding tags. Thereafter, a HTML Tag Filter is used to remove all tags from the analysed webpages, and a Document Parser converts the tag-free webpages into a list of keywords. These keywords are passed to an OntoAnnotator. In the OntoAnnotator, an OntoClassifier is used to describe each webpage with the mostly matched classes of domain ontology based on the tf-idf algorithm. Following this, an Annotator is used to annotate the webpage with the classes and their frequencies, and a Domain Marker is used to determine the belonged domain, based on the class frequencies for the webpage. Moreover, the framework of OntoCrawler is employed for crawling services from the ubiquitous computing environment [205], and crawling academic documents [204].

Navas-Delgado [141, 142] proposed a Deep Semantic Crawler framework. First of all, they proposed a methodology for annotating both dynamic and static Web documents and computing the scores of ontologies from a knowledge base on each Web document. The scores and related ontology concepts are then annotated within each Web document.

Francesconi and Peruginelli [59] proposed a Vertical Portal system, with the purpose of providing both resources and available solutions and services to satisfy users' requirements, within a legal domain. A focused crawler is adopted in the system, to crawl the domain-specific Web documents. Thereafter, a metadata generator automatically transforms the Web documents into metadata, by means of extraction. The focused crawler is implemented by computing the possibility of URLs in regards to the predefined topics. The metadata format is in accordance with the Dublin Core (DC) scheme in its XML version. Subsequently, the tf-idf model is used in order to extract the terms which can represent the documents. Next, two algorithms – Naive Bayes (NB) and Multiclass Support Vector Machines (MSVM) are adopted respectively for the documents' classification.

Giles et al. [64] proposed a niche search engine for retrieving e-business information, with the integration of the CiteSeer technique. A set of crawling strategies, including Brute Force, inquiries-based and focused crawlers are used to fetch Web documents. The CiteSeer technique is used to parse citations from the downloaded documents, and subsequently creates metadata based on the documents. To enhance the quality of metadata, the Support Vector Machine (SVM) algorithm is chosen to extract metadata, by the comparison with the Hidden Markov Model (HMM) algorithm.

The disadvantages of these metadata abstraction focused crawlers are as follows:

1. They use domain-specific ontologies to generate metadata. The design of domain-specific ontologies could be time-costly.

2. The performance of webpage classification relies heavily on the quality of ontologies.

3. No ontology evolution mechanism is provided by these crawlers.

4. The ontologies are mostly designed by domain experts, and thus sometimes may not match actual users' subjective perceptions of domain knowledge.

5. They do not propose any means by which metadata can be generated from service fields.

6. They do not provide any means to qualitatively rank the crawled service information.

7. Most of the crawlers are still in the conceptual phase without sufficient testing results to consolidate this approach.

2.2.5 Metadata Harvesting and Abstraction Crawlers

The metadata harvesting and abstraction crawler, as its name implies, is the semantic crawler that can execute dual tasks, namely metadata harvesting and metadata abstraction. In this section, we will introduce an instance of metadata harvesting and abstraction crawler.

SWoogle is a crawler-based indexing and retrieval system for the Semantic Web. It can index and retrieve metadata including ontologies, instances from SWDs. SWoogle deploys a series of crawlers for metadata harvesting, including a Google crawler for retrieving SWDs by identifying file type extensions. A focused crawler is also used to retrieve SWDs by given URLs. Then a SWoogle crawler is designed to harvest, parse and analyse SWDs or Semantic Web information pieces embedded in Web documents by utilizing several heuristics through semantic relations. A SWoogle Analyser is used to analyse the content of harvested Semantic Web documents for indexing and ranking purpose. The metadata harvested by SWoogle can be classified into three categories – metadata regarding syntactic and semantic features of SWDs, relations between SWDs, and analytical results regarding SWO/SWDB classifications, and SWDs ranking. The SWDs ranking algorithm is deducted from the PageRanks algorithm, computing amount of references instead of hyperlinks [43-47].

The drawbacks of the SWoogle crawler are as follows:

1. It does not provide any means for semantically classifying metadata.
2. It does not provide any approach for service fields.
3. It does not provide any means to qualitatively rank the crawled service information.
4. It does not provide sufficient testing results to consolidate this approach.

2.3 Semantic Service Discovery/Matchmaking

There are two definitions of service discovery/matchmaking, both of which are in the Web service field. A Web service refers to "*a software system designed to support interoperable machine-to-machine interaction over a network*" [18]. The first definition of service discovery is a process of finding an appropriate service provider for a service requester through a middle agent, from the perspective of matchmaking [38]. W3C [18] produced another definition with a broader meaning, which is "*the act of locating a machine-processable description of a Web service that may have been previously unknown and that meets certain functional criteria*".

The mechanisms of Web service discovery can be primarily divided into three categories – registries, indexes and peer-to-peer (P2P) [1, 63]. The registry mechanism is a centralized approach to control Web service publishing and discovery. A typical example is Universal Description, Discovery and Integration (UDDI) [1]. The index mechanism is a compilation or guide approach for Web services that exist elsewhere. A typical example is Google^{TM}. The P2P mechanism allows decentralized Web services to discover each other dramatically.

In order to realize Web service discovery, a mechanism should have the ability to search for Web services based on their functional descriptions. In the past, usually a Web service has been described by Web Service Description Language (WSDL) [33]. However, there are two challenges for WSDL:

1. Although WSDL is annotated by Service Modelling Language (SML) and is able to specify the operations and data structure of a Web service, as a result of insufficient semantic supports, it is not able to specify the meaning and semantic constraints of data involved in Web services, which may generate ambiguities during the discovery process [183].

2. WSDL is not able to represent the capabilities of a Web service, and therefore it is not able to recognize the similarity between the capabilities of a Web service being provided and the functionalities of a Web service being requested in the matchmaking process [33].

These two challenges highlight the need for semantics in Web service discovery. Hence, semantics can be used for describing and reasoning the capabilities of a Web service so as to match the functionalities specified in a service request. This gives rise to the vision of Semantic Web Services (SWS)-based Web service discovery, which integrates the service metadata, domain ontologies, formal tools and Web service architecture [137]. SWS are built around universal standards for the interchange of semantic metadata, which makes it easy for programmers to combine data from different sources and services without losing meaning. Currently, there are several ontology mark-up languages which have been employed for annotating SWS, including DARPA Agent Mark-up Language for Services (DAML-S) [149], Web Service Modelling Ontology (WSMO) [21], Web Ontology Language for Services (OWL-S) [136], Web Services Semantics (WSDL-S) [2], etc. Within them, DAML-S is substituted by OWL-S as OWL is one of the mainstream ontology mark-up languages; OWL-S and WSMO respectively propose new models for developing Web service ontologies; and WSDL-S proposes to add a semantic layer to the existing Web service descriptions.

Based on the SWS frameworks, to date many approaches have been developed for semantic service discovery. Since the literature regarding Web service discovery occupies the greater proportion of our survey, and the design of Web service discovery mechanism must be in accordance with the features of their application environment, in addition to the classification of SWS frameworks, we further divide the SWS-based service discovery methodologies in terms of their application environments, which are centralized, P2P, Grid computing, and ubiquitous computing [53]. Table 2.1 shows the classification of the existing semantic service discovery methodologies according to our survey.

Table 2.1: Classification of existing semantic service discovery methodologies

Service types	DAML-S	OWL-S	SWS WSMO	WSDL-S	Other SWS
Centralized	×	×	×	×	×
P2P	×	×	×	×	×
Grid Computing	×	×	×	×	×
Ubiquitous Computing	×	×	×	×	×

In this section, we present and subsequently discuss the existing semantic service discovery methodologies from the perspectives of services types and application environments. Section 2.3.1 reviews the existing semantic service discovery methodologies in centralized environments, Section 2.3.2 examines semantic service discovery methodologies in P2P environments, Section 2.3.3 gives an overview and discussion of existing semantic service discovery methodologies in grid computing environments, and Section 2.3.4 presents and discusses existing semantic service discovery methodologies in ubiquitous computing environments.

2.3.1 Semantic Service Discovery in Centralized Environments

The registry-based discovery approach is the dominant technology in centralized Web service environments [63]. UDDI established the first and uniform standard for Web service registry and discovery. However, as the only discovery mechanism provided by UDDI is the keyword-based search, which matches service advertisements with service requests by matching the features of business and service descriptions, UDDI fails to recognize the similarities and differences between the capabilities provided by Web services [97]. Consequently, much research has been undertaken in order to address this problem.

In 2003, McIlraith et al. [137, 138] were the first to bring semantics to Web services, by designing DAML-S for describing Web services, in order to describe the capabilities of Web services by means of ontologies. DAML-S is built on industry efforts such as Simple Object Access Protocol (SOAP), WSDL, Web Services Flow Language (WSFL), extension of WSDL (XLANG), and Business Process Execution Language for Web Services (BPEL4WS) by adding class information into service profiles in order to describe and constrain the range of Web service capabilities. Domain-specific Web service ontologies need to be built by DAML or DAML+OIL, in order to clearly represent capabilities of Web services and user constraints. This enables automated Web service discovery in terms of matching Web service descriptions with service-requester-specified properties in DAML-S.

Li and Horrocks [110] introduced the use of the DAML-S ontologies in order to semantically represent the descriptions of services, service advertisements and service requests. A matching algorithm is developed to match service advertisements (abbreviated as A) with service requests (abbreviated as R), which includes five digressive match levels, namely: Exact ($R \equiv A$), Plug-in ($R \subseteq A$), Subsume ($A \subseteq R$), Intersection ($\neg(A \cap R) \subseteq \bot$), and Disjoint ($A \cap R \subseteq \bot$). Chiat et al. [31] designed a Description Logic (DL)-based matchmaker for DAML-S annotated Web services. A matching algorithm is designed to match service advertisements with service requests, which follows the same match levels as Li and Horrocks' algorithm. Similarly, Sycara et al. [186] utilized DAML-S to annotate Web service profiles, including the capabilities, non-functional properties (NFPs) and provider profiles of Web services. Then, a match engine is employed to find a match between service advertisements and Web services. The match algorithm includes four digressive match levels of Exact, Plug-in, Subsume and Disjoint.

Liu et al. [118] designed an e-service platform integrated with semantic search for e-service metadata. E-service metadata refers to descriptions of e-services and providers, the aim of which is to publish and to discover e-services. There are two types of metadata in the system: business level metadata – the description of e-service providers, and service level metadata – the description of basic information about e-service. The authors adopted UDDI which is a Web service standard to register and search e-services. Three means for searching service and business are provided: find_business, find_service and XQuery. Find_business returns a list of service providers for specific conditions; find_service returns the information for a list of services who match customized conditions; and XQuery queries the extended metadata added in a businessService list.

Kawamura et al. [97] designed the framework for a Semantic Service Matchmaker (SSM) in order to search services in the UDDI. The services are coded with the Web Service Semantic Profile (WSSP), inspired by the DAML-S Service Profile, in order to encode semantic information into the WSDL of services. Five ordinal filters are employed for the matchmaking process, including a namespace filter which is used to check whether or not the requested service and provided service have at least one shared namespace; a text filter which uses the tf-idf to pre-check the human-readable service explanation parts; a domain filter which is used to check whether or not the requested service and provided service belong to an ontology domain; an input/output (I/O) type filter which is used to check whether or not the I/O parameters match; and a constraint filter which is used to determine whether or not a service request can be subsumed by a service.

Li et al. [111] proposed an ontology-based matchmaking approach for web services in e-markets. This approach contains a DAML-based domain ontology and a decision making cycle-based semantic matchmaking mechanism, which includes the steps of problem re-organization, service search, pre-evaluation, decision making and post evaluation.

The disadvantages of these DAML-S-based Web service discovery approaches in centralized environments are:

1. Since DAML has been substituted by OWL, the DAML-based Web service ontologies also need to be re-annotated by OWL.

2. These approaches need service providers to register their services with the centralized registries, which is difficult to achieve in the real-world environment, considering the physically dispersed service providers and the need for a high volume of storage space.

3. They do not propose any means for discovering Web services or resources outside registries.

4. It is difficult for the service requesters who do not have enough domain knowledge about their requested services, to configure the constraints for precisely locating a service.

5. The employed Web service ontologies are designed by domain experts, which may not be consistent with actual users' perceptions of Web service domain knowledge.

6. They do not provide ontology evolution mechanisms to allow Web service ontologies to adapt to the dynamic nature of service domain knowledge.

7. They do not propose any means by which service requesters can choose a Web service with the best QoS.

8. The DAML-S-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

9. They do not propose any means for generic service discovery/matchmaking.

10. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Along with the emergence and increasing popularity of OWL, many researchers have shifted their focus from DAML to OWL for describing Web services. Consequently, Marin et al. [136] proposed an OWL-based Web service ontology – OWL-S. Based on OWL-S, much research has been undertaken and is discussed in what follows.

Klusch et al. [100, 101] designed an OWL-S service matchmaker (OWLS-MX) to match Web services with service requests. OWLS-MX integrates the logic-based matching and non-logic-based matching. The former determines the similarity based on the relationships between concepts within service ontologies respectively, referring to service advertisements and service requests, which includes seven match levels of Exact, Plug-in, Subsumes, Subsumed By, Nearest Neighbour and Fail. The latter uses mathematical models to determine the syntactic similarity between domain-specific concepts referring to the I/O of service advertisements and service requests. Similarly, Bianchini et al. [14] proposed a hybrid matchmaking approach for OWL-S annotated SWS. First of all, a service can be described by the conjunction of OWL-DL-based concepts from a service ontology. Thus, the DL-based classification is used to precisely establish the kind of matching between a request and an advertised service, by means of deducing their relationship in the ontology. The similarity between the request and its partially matched services are then computed and ranked based on a hybrid approach – first estimating their degree of similarity based on conceptual relationships in local Web service ontologies, second computing the similarity based on the similarity coefficients of I/O parameters and the similarity coefficients of the operations conjunct with the I/O parameters.

Sriharee and Senivongse [175] defined a series of service profile-based mechanisms for Semantic Web service matchmaking. They built an OWL-based upper ontology for modelling the profile of Web services. The profile integrates the information with regard to attributes, capability, structure, behaviour and constraints of Web services. Based on the integrated Web service profile, they provided multiple matching criteria for Web service matchmaking, including ontological concept matching, numerical constraints matching, sets of ontological values matching, service constraints matching, behaviour constraints matching and simple attributes matching. The matched Web services can then be ranked based upon the integrative values from all matchmaking criteria.

The limitations of these OWL-S-based Web service discovery approaches in centralized environments are:

1. These approaches need service providers to register their services to the centralized registries, which is difficult to achieve in the real environment, considering the physically dispersed service providers and the need for a high volume of storage space.

2. They do not propose any means for discovering the Web services or resources outside registries.

3. It is difficult for the service requesters who do not have enough domain knowledge about their requester services, to configure the constraints for precisely locating a service.

4. The employed Web service ontologies are designed by domain experts, which may not be consistent with actual users' perceptions of Web service domain knowledge.

5. They do not provide ontology evolution mechanisms to allow Web service ontologies to adapt to the dynamic service domain knowledge.

6. They do not propose any means by which service requesters can choose a Web service with the best QoS.

7. The OWL-S-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

8. They do not propose any means for generic service discovery/matchmaking.

9. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Apart from OWL-S for service discovery, many researchers have taken the initiative of using WSMO to assist in service discovery. WSMO consists of the following components [106]:

- Goals – a service requester's objectives when consulting a Web service,

- Ontologies – a formal semantic description of the information shared by all other components,

- Mediators – connectors between components, which provide interoperability between different ontologies,

- Web services – semantic descriptions of Web services, which provide an ontology-based framework that describes the capacity and functionalities of SWS.

Keller et al. [98] proposed a WSMO-based service discovery engine. The engine uses a DL reasoner to match WSMO-annotated service advertisements with service requests, which includes five levels of matching: Equivalence, Plug-in, Subsume, intersection and disjoint.

Keller et al.'s approach has the following possible drawbacks:

1. These approaches need service providers to register their services with the centralized registries, which is difficult to achieve in the real environment, considering the physically dispersed service providers and the need for a high volume of storage space.

2. They do not propose any means for discovering the Web services or resources outside registries.

3. It is difficult for the service requesters who do not have enough domain knowledge about their requester services, to configure the constraints for precisely locating a service.

4. The employed Web service ontologies are designed by domain experts, which may not be consistent with actual users' perceptions of Web service domain knowledge.

5. They do not provide ontology evolution mechanisms that allow Web service ontologies to adapt to the dynamic service domain knowledge.

6. They do not propose any means by which service requesters can choose a Web service with the best QoS.

7. The WSMO-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

8. They do not propose any means for generic service discovery/matchmaking.

9. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Some researchers also have employed other ontology mark-up languages to support the deployment and interoperability of SWS, such as RDF.

Lord et al. [128] proposed a user-oriented semantic service discovery architecture – Feta, for the myGrid project in the bioinformatics domain. The aim of the myGrid project is to develop programmatically accessible middleware to enable the transfer of information and composition of data and tool services to large workflows to address the real needs of lab biologists [178]. Due to the special needs of SWS in bioinformatics – user transparency and messaging opacity - Lord et al. designed a data model for service descriptions and created a myGrid service ontology with RDF. Feta annotates the SWS stored in a UDDI registry with XML, owing to two reasons: 1) the use of XML schema validation can ensure that Feta documents can alleviate the need for further error checking of discovery process; and 2) the service metadata can be mapped into the Feta schema for service discovery. Service requests are then annotated with RDF in order to compare them with the I/O, operation names, and resources of services.

The disadvantages of Lord et al.'s approach are as follows:

1. These approaches need service providers to register their services with the centralized registries, which is difficult to achieve in the real environment, considering the physically dispersed service providers and the need of high volume of storage spaces.

2. They do not propose any means for discovering the Web services or resources outside registries.

3. It is difficult for the service requesters who do not have enough domain knowledge about their requester services, to configure the constraints for precisely locating a service.

4. The employed myGrid service ontologies are designed by domain experts, which may not be consistent with actual users – that is, lab biologists' perceptions of domain-specific service knowledge.

5. They do not provide ontology evolution mechanisms to allow Web service ontologies to adapt to the dynamic service domain knowledge.

6. They do not propose any means by which service requesters can choose a Web service with the best QoS.

7. The RDF-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

8. They do not propose any means for generic service discovery/matchmaking.

9. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

The above approaches all attempt to use ontology mark-up languages to directly annotate descriptions of Web services in order to support their interoperability. However, two limitations are observed from these approaches: 1) the annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service; and 2) on account of the employment of multiple ontology mark-up languages for Web service annotation, it is difficult to exchange meanings between the multi-language-annotated Web services. This leads to the emergence of WSDL-S, which directly embeds semantic annotations in WSDL-encoded service descriptions. WSDL-S provides a lightweight mechanism by which Web services can be semantically expressed by multi-fold models or languages, such as UML or OWL. This allows existing semantics to be re-used for service interpretation, and the update of the existing tools with WSDL specifications [2]. No WSDL-S-based service discovery approach has been identified in this survey.

The following semantic service discovery methodologies have their own special features and thus we discuss them separately.

Talantikite et al. [187] used OWL-S to annotate the I/O and quality of Web services and the I/O of service requests. A semantic network is constructed by connecting the semantically similar Web services. The semantic similarity is measured by comparing the I/O of Web services based on domain-specific Web service ontologies. Therefore, a service request can easily find semantically similar services by means of the semantic network.

The limitations of Talantikite et al.'s approach are:

1. They do not propose any means by which the semantic network can evolve when the information of its participants is updated.

2. They do not propose any means for discovering the Web services or resources outside registries.

3. It is difficult for the service requesters who do not have enough domain knowledge about their requester services, to configure the constraints for precisely locating a service.

4. The employed Web service ontologies are designed by domain experts, which may not be consistent with actual users' perceptions of Web service domain knowledge.

5. They do not provide ontology evolution mechanisms to allow Web service ontologies to adapt to the dynamic service domain knowledge.

6. The OWL-S-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

7. They do not propose any means for generic service discovery/matchmaking.

8. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Fenza et al. [58] designed a hybrid approach which combines an agent-based paradigm and fuzzy modelling for SWS matchmaking. They used OWL-S and a fuzzy multiset to represent service profiles, which forms a knowledge layer for Web services. Based on the knowledge layer, they designed an agent layer which comprises broker agents and advertiser agents. The broker agents turn a service query into a fuzzy multiset and Web services into a fuzzy matrix by respectively comparing the query and profiles of Web services with concepts from a local service ontology. By means of a Fuzzy C-Means (FCM) algorithm, Web services are clustered and distances from the Web services to the query are obtained. By configuring different threshold values for the distance, the Web services that match the query are retrieved.

The drawbacks of Fenza et al.'s approach are as follows:

1. They do not propose any means for evolving service ontologies to enable them to adapt to the dynamic service environment and service domain knowledge.

2. They do not consider the cost of recomputing the fuzzy matrix when updating service ontologies.

3. They do not propose any means for discovering the Web services outside registries.

4. For the service requesters who do not have enough domain knowledge about their requester services, their queries could be incorrect and thus cannot be used to discover services.

5. They do not propose any means by which service requesters can choose a Web service with the best QoS.

6. They do not propose any means for establishing agreements between domain experts and service requesters for designing service domain ontologies.

7. The OWL-S-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

8. They do not propose any means for generic service discovery/matchmaking.

9. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Bianchini et al. [15] attempted to use the ontology approach to support e-service publication and discovery in the UDDI environment. The authors designed a service provider ontology that defines service providers, functionalities of services (I/O and operations), available channels and quality information, and a service requester ontology that defines profiles and contexts of service requests (characterized by devices, networks, network interfaces and application protocols) and requested quality information. The quality information includes QoS, quality of providers and quality of channels. Additionally, a domain service ontology is created, which includes a concrete service layer, an abstract service layer and a subject category layer, which are described as follows:

- A concrete service layer contains WSDL interfaces of services clustered by their functional similarities. The similarity is obtained by aggregating the two similarity coefficients as follows: 1) entity-based similarity coefficients which is the measure of affinity between I/O names of services in a service domain ontology; and 2) functionality-based similarity coefficient which is the measure of affinity between operation names and I/O parameter names of services in a service domain ontology.

- An abstract service layer contains abstract services which define the common functionalities of services in each cluster. Mapping rules are defined to relate operations in abstract services and in concrete services.

- A subject category layer contains a standard taxonomy of services. Each abstract service is associated with one or more classes in the taxonomy with "*is-a*" or "*is-composed-of*" relationships.

This 3-layer structure enables the service search by category and functionality. Moreover, the authors proposed a full search module, which compares a service request and a service advertisement from the perspective of functions restricted by their context and quality information.

Bianchini et al.'s approach has the following drawbacks:

1. For the service requesters who do not have enough domain knowledge about their requester services, their queries could be incorrect and thus cannot be used to discover services.
2. They do not propose any means for evolving service ontologies, thereby allowing them to adapt to the dynamic service environment and service domain knowledge.
3. They do not propose any means for discovering the Web services outside registries.
4. They do not propose any means for making agreements between domain experts and service requesters for designing service domain ontologies.
5. They do not propose any means for generic service discovery/matchmaking.
6. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

2.3.2 Semantic Service Discovery in P2P Environments

At stated previously, it is difficult to realize the centralized registry-based service discovery approach in the open environment where all the resources are connected and widely distributed, as a result of distributed resources and services. P2P overlay networks provide a solution to this problem. A P2P network is a distributed network architecture where resources and services are all decentralized, and participants (peers) are all decentralized and self-organized [165]. The P2P-based Web service discovery (PWSD) is built upon the prerequisite that every peer can directly access other peers' resources and services without the need for intermediate entities [63]. In the past, PWSD relied on the keywords-based matching between service profiles and service requests described by WSDL. Since WSDL is purely syntactic, PWSD fails to find the similarity and differences between the capabilities provided by Web services. Therefore, many Web semantics-based PWSD approaches have been designed in this field.

Verma et al. [197] proposed a METEOR-S Web Services Discovery Infrastructure (MWSDI) in order to use DAML-S ontologies to organize the distributed service registries. One feature of the registries requires that the ontologies should be distributed to users for service discovery and publication. The MWSDI architecture consists of four layers. The first layer is a data layer which contains Web service registries. The second layer is a semantic specification layer designed to enable the use of semantic metadata, which involves: 1) a registry ontology which maps between registries and domains, and indicates registry-registry relationships; and 2) Web service profiles represented by domain-specific ontologies. The third layer is a communication layer which consists of a P2P network, which provides the infrastructure for the MWSDI. The fourth layer is an operator service layer which contains all the services provided by the operator peers. The operator services are value-added services, such as semantic service discovery and publication. Therefore, the service discovery can firstly follow the registry ontology to find the appropriate registry in the requested service domain. A service request can then be matched with service profiles by means of mapping operations and I/O of operations of service profiles into domain-specific service ontologies.

The challenges for Verma et al.'s approach are as follows:

1. DAML-S has already been replaced by OWL-S.

2. They do not propose any means for discovering the unknown services in P2P environments.

3. It is difficult for service requesters who do not have enough domain knowledge about their requester services, to configure the requested service profiles for precisely locating a service.

4. The employed Web service ontologies are designed by domain experts, which may not be consistent with actual users' perceptions of Web service domain knowledge.

5. They do not provide ontology evolution mechanisms to allow Web service ontologies to adapt to the dynamic nature of service domain knowledge.

6. They do not propose any means by which service requesters can choose a Web service with the best QoS.

7. The DAML-S-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

8. They do not propose any means for generic service discovery/matchmaking.

9. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Zhang et al. [208] proposed an OWL-based resource and service discovery system. Four ontologies are designed respectively for resource domain, thesaurus, service description and QoS of Web service in P2P networks. The resource domain ontology consists of classes that represent resource domains and instances that represent resources. They proposed three mathematical models – a group locating algorithm and a resource locating algorithm for locating resources, and a service matching algorithm for locating service in the P2P networks, which are described as follows:

- The group locating algorithm aims at finding related friend groups for a given resource. This is realized by converting a resource domain ontology into a tree, and finding the nearest neighbour for the node that represents the resource.

- The resource locating algorithm aims at locating the related resources for a given resource, by means of extracting all the instances of the related friend groups in the resource domain ontology and finding the nearest neighbour for the node that represents the requested resource.

- The service matching algorithm aims at matching a service request with available services, by matching the requested service profile and available service profiles by mapping the I/O and QoS information of service profiles to the service description ontology and QoS ontology.

By means of these models, the resources services in the neighbour peers can be retrieved. If a user chooses a service of interest, s/he needs to upload an OWL-S-annotated service query description profile, including the I/O and QoS information for the matching between the service and the query.

The disadvantages of Zhang et al.'s approach are as follows:

1. As service requesters must upload an OWL-S annotated service query description profile before executing a service query, for the service requesters who do not have enough domain knowledge about their requester services, it is difficult to configure the profile.

2. They do not propose any means for discovering the unknown services in P2P environments.

3. The employed Web service ontologies are designed by domain experts, which may not be consistent with actual users' perceptions of Web service domain knowledge.

4. They do not provide ontology evolution mechanisms to allow Web service ontologies to adapt to the dynamic nature of service domain knowledge.

5. The OWL-S-annotated descriptions duplicate the descriptions of Web services in WSDL, which leads to the inconvenience of creating multiple definitions for a Web service.

6. They do not propose any means for generic service discovery/matchmaking.

7. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Liu et al. [121] proposed to construct a semantic service link network to semantically organize Web services for the convenience of the service discovery in the P2P networks. The semantic service link network is built by multi-dimensionally clustering services based on operation names, categories, and keywords. The clustering dimensions are the types of relations between the attributes of the services, which include Equal-to, Partial-Specialization, Extension-Specialization, Revision-Specialization, Non-Specialization and Unknown. The clusters are determined by the fuzzy integration of the operation similarity, category similarity, and keyword similarity between services.

The shortcomings of Liu et al.'s approach are as follows:

1. It is difficult for the service requesters who do not have enough domain knowledge about their requester services, to configure the requested service profiles.

2. They do not propose any means for discovering the unknown services in P2P environments.

3. Their approach is still in the conceptual phase.

4. They provide insufficient testing results to consolidate their approach.

5. They do not provide any performance comparison with other service discovery approaches.

6. They do not propose any means by which a service requester can find a service with the best quality.

7. They do not propose any means for generic service discovery/matchmaking.

8. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

2.3.3 Semantic Service Discovery in Grid Computing Environments

A grid computing environment is a distributed, large-scale cluster computing environment which uses software to divide and apportion pieces of a program among several computers, in order to process the program in parallel [10]. Resource and service discovery is a crucial issue in the grid computing environment, by means of which a service provider can decide how to find resources to implement a particular task and how many potential resources a service provider can offer to a service requester. Service discovery in the grid computing environment refers to the process of locating service providers and retrieving service profiles [129]. The problem of service discovery arises due to the heterogeneous, distributed resources spanning different organizations, when assigning resources to tasks in order to meet different task requirements and resource policies. These requirements and policies are often expressed in disjoint applications and resource models; hence, the emergence of semantic service discovery [79].

Ludwig et al. [129, 130] proposed a semantic service discovery framework in the grid computing environment. This is realized by building ontologies for grid applications and services, and designed a three-stage discovery framework for grid service discovery, which includes: 1) context selection, which matches service requests with context semantics defined in the application ontologies; 2) semantic selection, which matches service requests with DAML+OIL annotated service attributes; and 3) registry selection where a lookup is performed. They designed a matching engine that implements the matching tasks according to predefined matching rules, and a reference engine for reasoning instances of classes and relationships between classes with DAML+OIL ontologies. In addition, the matching engine can rank semantically similar services with a service request based on a similarity metrics, which is the aggregation of the similarity values of attributes, descriptions and metadata description between service advertisements and service requests.

The limitations of Ludwig et al.’s approach are as follows:

1. DAML+OIL has already been substituted by OWL.

2. As service requesters must upload an ontology-based service query description profile before executing a service query, if service requesters do not have enough domain knowledge about their requested services, it is difficult for them to configure the profile.

3. They do not provide any means for annotating and discovering unknown services in grid computing environments.

4. The application ontologies and grid service ontologies are all designed by relevant domain experts. When end users use these ontologies to annotate their requested service profiles, there could be discrepancies between the ontologies and their own perceptions of domain knowledge.

5. They do not propose any solutions to enable the evolution of domain-independent ontologies in order to maintain consistency between the ontologies and dynamic domain knowledge.

6. They do not propose any means by which a service requester can find a service with the best quality.

7. They do not propose any means for generic service discovery/matchmaking.

8. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Tangmunarunkit et al. [190] proposed a semantic service matchmaker in the grid environment, which is comprised of RDF ontologies, domain background knowledge and matchmaking rules. There are three categories of ontologies: 1) resource ontologies for describing resources on the Grid; 2) resource request ontologies for describing requests; and 3) policy ontologies for describing the authorization and usage policy of resources. In addition, matchmaking rules are designed with the ontologies and the ontology-annotated domain background knowledge, in order to match resource providers with requests. The matchmaker provides a matchmaking service and a brokering service to carry out a task, which are described as follows:

- Upon receiving an ontology-based resource request, the matchmaker returns a list of resource providers based on matching rules and the requester's preference criteria.

- The matchmaker sends a negotiation message to a highly ranked resource provider by using predefined negotiation protocols. If the resource provider accepts the request, the matchmaker will contact the provider by using predefined claiming protocols.

Analogously, Harth et al. [79] designed an ontology-based matchmaker (OMM) for performing resource selection in the grid, by using terms defined in ontologies to form a loose coupling between resources and request descriptions. Their matchmaker uses RDF-encoded ontologies to describe requests, resources and usage policies. Inference rules are designed to reason about the characteristics of requests, resources and usage policies in order to find resources that can satisfy requesters' requirements.

The challenges for the RDF-based service discovery approaches in grid computing environments are as follows:

1. RDF has already been replaced by OWL.

2. It is difficult for the service requesters who do not have enough domain knowledge about their requester services, to configure ontology-based request profiles.

3. They do not provide any means for discovering unknown services in grid computing environments.

4. The resource, request and policy ontologies are all designed by relevant domain experts. When end users use these ontologies to annotate their requested service profiles, there could be discrepancies between the ontologies and their own perceptions of domain knowledge.

5. They do not propose any solutions to enable the evolution of the ontologies in order maintain the consistency of the ontologies with the dynamic domain knowledge.

6. They do not propose any means by which a service requester can find a service with the best quality.

7. They do not propose any means for generic service discovery/matchmaking.

8. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Han et al. [72] defined a semantic-similarity-model-oriented heuristic algorithm in order to help agents to discover neighbour agents with requested services in a grid environment, based on the prerequisite that each agent stands at one node in the grid and is responsible for the resources in that node. The resources can be denoted by a domain ontology. Consequently, the similarity of agents can be estimated by considering the similarity of resources between each node, namely the similarity between groups of concepts that represent resources, which can be calculated by a semantic similarity model. Based on the similarity values, the neighbours of each node can be decided. Subsequently, the authors designed a heuristic algorithm for dynamic resource discovery, which carries out the following steps to find agents who can provide requested resources:

Step 1. Randomly selecting an agent from the grid who has at least one requested resource and computing the similarity between its resources and one of the requested resources for a task. If the similarity value is beyond a predefined threshold value, the agent will be assigned with the task. The algorithm will then check the neighbours of the agent who have at least one of the requested resources and obtain their similarity values with the requested resources. This is a recursive process until all the neighbours have been checked.

Step2. If some of the requested resources are not found, Step 1 will be executed again from another random node from the unchecked set.

The limitations of Han et al.'s approach are as follows:

1. They do not propose any means for dealing with new nodes in a grid environment.

2. If the resources of the nodes in a grid are highly dynamic, the update of semantic similarity models could be time-costly.

3. They do not provide any means for discovering unknown services in grid computing environments.

4. The resource, request and policy ontologies are all designed by relevant domain experts. When end users take means of these ontologies to annotate their requested service profiles, there could be gaps between the ontologies and their own perceptions of domain knowledge.

5. They do not propose any solutions to enable the evolution of the ontologies in order to maintain the consistency of the ontologies with the dynamic domain knowledge.

6. They do not propose any means by which a service requester can find a service with the best quality.

7. They do not propose any means for generic service discovery/matchmaking.

8. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Vega-Gorgojo et al. [196] designed a semantic approach for discovering Computer Supported Collaborative Learning (CSCL) services. CSCL system is a grid-based collaborative learning environment which provides services to support CSCL activities [102]. They developed an ontology for describing the CSCL tools, and then introduced an ontology-enabled service discovery infrastructure. In this infrastructure, all CSCL services are annotated by the ontology concepts. An ontology-compliant reasoner is then utilized to enable the retrieval of service instances by a racer query language (RQF) [69].

Vega-Gorgojo et al.'s approach may meet the following challenges:

1. They do not propose any means for dealing with unknown services in a grid environment.

2. For the service requesters who do not have enough domain knowledge about their requester services, their queries could be incorrect and thus cannot be used to discover services.

3. They do not propose any means for making agreements between domain experts and real users for designing service domain ontologies.

4. They do not propose any solutions to enable the evolution of the ontologies in order to maintain the consistency of the ontologies with the dynamic domain knowledge.

5. They do not propose any means by which a service requester can find a service with the best quality.

6. They do not propose any means for generic service discovery/matchmaking.

7. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

2.3.4 Semantic Service Discovery in Ubiquitous Computing Environments

Ubiquitous (pervasive) computing refers to a post-desktop model of Human Computer Interaction (HCI) in which information processing has been thoroughly integrated into everyday objects and activities [77]. It enables users to access information and computational resources anytime and anywhere in an intuitive and natural way, by embedding computers and networks in physical environments [163]. Users can take part in the ubiquitous computing environment by bringing mobile devices that integrates seamlessly in the existing infrastructure. Services provided by the ubiquitous computing environment are context-aware, and therefore keep changing to adapt to users' positions, preferences, requirements and existing infrastructure [9]. Therefore, service discovery is critical and challenging in the ubiquitous computing environment, where devices are constantly aware of each other and the network topology [195]. In the ubiquitous computing environment, since resources and access points are highly dynamic, current researches in this field focus primarily on the development of Service Discovery Protocol (SDP). In order to precisely match a service request with a context-aware service, many researchers tend to use Semantic Web technologies to semantically interpret service contexts.

Vazquez et al. [195] developed a UDP/HTTP-based Multicast Resource Discovery Protocol (mRDP). The mRDP is built on the prerequisite that all resources in the ubiquitous computing environment are annotated with RDF/OWL. The mRDP architecture comprises mRDP clients and mRDP servers. When a semantic powered request is disseminated from an mRDP client to all mRDP servers in the network, each server will model the request with its semantic information models by SPARQL, and the Uniform Resource Identifiers (URIs) of the matched resources are returned.

The drawbacks of Vazquez et al.'s approach are as follows:

1. They do not propose any solutions to enable the evolution of the service capacity ontologies in order to maintain the consistency of the ontologies with the dynamic environment.

2. They do not provide any means for discovering non-semantic-supported devices in ubiquitous computing environments.

3. The ontologies are designed by domain experts, which may not agree with actual users' perceptions of domain knowledge.

4. They do not propose any means for generic service discovery/matchmaking.

5. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Mokhtar et al. [140] developed an Efficient Semantic Service Discovery (EASY) approach, in order to enhance the existing SDPs for the semantic, context-aware and QoS-aware service discovery. EASY can be employed on top of the existing SDPs by adding semantics to their syntactic descriptions. The EASY contains an EASY Language (EASY-L) which originates from OWL-S for describing capacities of semantic service, including the attribute of I/O, category and properties. There are two types of properties – context properties and QoS properties. Based on these ontologies, an EASY Matching (EASY-M) is designed which is a set of conformance relations (Exact, Inclusive and Weak) for matching between requested service capacities and advertised service capacities. In the instance of an inclusive relation between a request and an advertisement, their similarity value can be calculated by considering the relations (Subsume and Plug-in) of the concepts denoting their respective attributes.

The disadvantages of Mokhtar et al.'s approach are as follows:

1. They do not propose any solutions to enable the evolution of the service capacity ontologies in order to maintain the consistency of the ontologies with the dynamic environment.

2. They do not provide any means for discovering non-semantic-supported devices in ubiquitous computing environments.

3. The ontologies are designed by domain experts, which may not agree with actual users' perceptions of domain knowledge.

4. They do not propose any means for generic service discovery/matchmaking.

5. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Borens [20] proposed a context-aware, ontology-based, semantic service discovery approach (COSS) for ubiquitous computing environments. The input of COSS is a set of advertised services, a set of context providers and ontologies. There are seven OWL-encoded ontologies for annotating service advertisements and service requests, which are service, service type, product, attribute, context and payment ontologies. The author defined five levels for the matching of service advertisements with service requests, which are Exact, Plug-in, Subsume, Intersection and Disjoint.

Borens’ approach may meet the following challenges:

1. This approach is built upon the prerequisite that service requesters are aware of the context of their requested services. They do not propose any means by which service advertisements can automatically be aware of their contexts.

2. They do not propose any means by which the designed ontologies can evolve to adapt the dynamic environment and service domain knowledge.

3. They do not propose any means by which a service request or a service advertisement can be automatically annotated.

4. They do not propose any means by which the designed ontologies can achieve agreements between ontologies designers and users.

5. They do not propose any means by which a service requester can choose an advertised service that will deliver the best quality.

6. They do not propose any means for generic service discovery/matchmaking.

7. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

Toninelli et al. [192] provided the framework of a middleware – Adaptable Intelligent Discovery of context-Aware Services (AIDAS), for user-centric semantic service discovery in a mobile environment. AIDAS adds semantics to the properties of interacting entities and the environment by annotating the profiles of services, users and devices by OWL-DL. Each service is described by a static profile which contains static information such as service name, service functions etc., and a dynamic profile which contains dynamic information, such as location and state of applications. AIDAS creates a service capacity/requirement ontology by OWL-DL. User metadata include user profiles and preferences. Device descriptions include device characteristics and operation conditions. The service discovery process is implemented by the following five agents:

- A Metadata Manager which provides supports for the publishing and maintenance of service, user and device metadata,

- A Discovery Manager which is employed to match user and services based on their metadata.

- A Context Manager which monitors the changes of user contexts and environments.

- A Query Manager which is in charge of collecting and processing user requests.

- A Profile Matching Engine which uses a matching algorithm to match user/device requirements with service capabilities by taking into account user preferences. The matching algorithm is realized by determining the relationships (Exact, Subsume, Plug-in, and Fail) between the annotated user/device requirements and service capabilities in the service capacity/requirement ontology. If some user preferences overlap some user/device requirements, the overlapped user preferences can substitute the corresponding user/device requirements to compare service capabilities.

The limitations of Toninelli et al.'s research approach are as follows:

1. They do not propose any solutions to enable the evolution of the service capacity/requirement ontologies in order to maintain the consistency of the ontologies with the dynamic environment.

2. They do not propose any means by which a user/device can find a service with the best quality.

3. They do not provide any means for discovering non-semantic-supported devices in ubiquitous computing environments.

4. Efficiency is a key factor when evaluating a service discovery approach in the ubiquitous environment. There is no performance data about the time cost of this approach.

5. They do not provide any evaluation results to consolidate this approach.

6. They do not make any comparisons with existing non-semantic-supported approaches.

7. They do not propose any means for generic service discovery/matchmaking.

8. They do not propose any means for non-WSDL-encoded service discovery/matchmaking.

2.4 Semantic Search Engines and Related Technologies

Semantic search, as an application of Semantic Web in the field of information retrieval, has shown significant potential for improving the performance of retrieval. Compared with the traditional keyword-based search engines that focus on the frequency of word appearance, semantic search engines are more likely to try to understand the meanings hidden in retrieved documents and users' queries, by means of adding semantic tags to texts, in order to structuralize and conceptualize objects within documents [68]. the researches regarding semantic search are still in the initial stage because traditional keyword-based search engines such as Google®, Yahoo® and so forth still dominate the search engine market.

In 2008, we conducted a survey on current semantic search technologies, which comprised twenty-seven research achievements. We classified the research achievements by means of product types, technological features and retrieved objects. Six categories of semantic search technologies were identified, which include semantic search engines, semantic search technologies, hybrid semantic (keyword-based enhanced by semantic) search engines, XML (files) search engines, ontology (files) search engines and semantic multimedia (files) search engines. The survey enabled us to identify four fundamental drawbacks of current semantic search technologies including: ignorance of ontological differences between designers and users' perceptions, ignorance of evolving knowledge structure, low precision and high recall, and lack of evaluations [51].

In 2010, Grimes [65] suggested a new scheme for classifying existing semantic search technologies, which categorizes the semantic search technologies into 2+9 views, which are: 1) related searches/queries, 2) reference results, 3) semantically annotated results, 4) full-text similarity search, 5) search on semantic/syntactic annotations, 6) concept search, 7) ontology-based search, 8) Semantic Web search, 9) faceted search, 10) clustered search, and 11) natural language search.

In this section, we discuss and analyse the existing literature regarding semantic search engines and technologies based on Grimes' classification scheme. However, it needs to be noted that we classify the existing literature by means of its major technical features and the literature under each category may have features in common with other categories.

2.4.1 Related Searches Engines and Technologies

As stated by Grimes [65], related searches engines and technologies refer to searching for related terms of query terms.

Duke et al. [54] proposed a semantic search and browse tool – Squirrel. By means of this system, once a user submits a query, the query terms will be matched with concepts (topics) from PROTON – a lightweight general purpose ontology [154]. The query terms and matched topics are sent to Web search engines. The documents that match the query terms and topics are classified according to their types. Under each topic, documents are ranked following the combination of the degree of relevance to query terms and the level of interest to user profile (a list of topics).

The limitations of Duke et al.'s approach are as follows:

1. The performance of this approach relies on the quality of ontologies and queries.

2. The expanded query terms could burden Web search engines and cause query flooding.

3. The keyword-based query term-ontological concept matching could lead to a mismatch or incomplete matching.

4. User profile-based document ranking cannot work when users search for documents with new topics.

5. They do not propose any means for ontology evolution.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.2 Semantic Search Engines and Technologies for Reference Results

Semantic search engines and technologies for reference results refer to the semantic search engines and technologies which respond with materials that define search terms [65].

Kandogan et al. [96] developed a semantic search engine – Avatar, which enhances traditional text search engines by using ontology annotations. Avatar has two main functions: (1) extraction and representation – by means of the Unstructured Information Management Architecture (UIMA) framework, which is a workflow consisting of a chain of annotators extracted from documents and stored in the annotation store; and (2) interpretation – a process of automatically transforming a keyword search to several precise searches. Avatar consists of two main parts: a semantic optimizer and a user interaction engine. When a query is entered into the former, it will output a list of ranked interpretations for the query; and then the top-ranked interpretations are passed to the latter, which will display the interpretations and the retrieved documents from the interpretations.

Kandogan et al.'s semantic search engine has the following limitations:

1. The interpretations with fewer keywords do not have a direct relationship with their level of importance, which could be ignored and the results could be incomplete.

2. The performance of the search approach relies on the quality of queries.

3. The expanded queries could affect the efficiency of the search engine.

4. The efficiency of the search engine could be challenged when searching in a large-scale information base.

5. They do not propose any means for semantic search in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.3 Search Engines and Technologies for Semantically Annotated Results

Semantically annotated results refer to the annotation of retrieved objects by highlighting text features of especially named or pattern-defined entities that are semantically relevant to search terms [65].

Wang et al. [199] projected a semantic search methodology to retrieve information from normal tables, which has three main steps: identifying semantic relationships between table cells; converting tables into data in databases; and retrieving objective data by query languages. The research objective defined by the authors is how to use a given table and a given domain ontology to convert a table into a database table with semantics. The authors' approach is used to denote the layout-by-layout syntax grammar, and match these denotations with given templates that can be used to analyse the semantics of table cells. Semantic-preserving transformation is then used to transform tables to database format.

Wang et al.'s approach could have the following limitations:

1. Reorganizing cells in tables cannot completely solve the ambiguity problem in template matching, which needs proper semantic approaches.

2. There are no success criteria given in the experiment. As a result, it is not possible to evaluate the performance of this method.

3. The performance of the search methodology relies on the quality of domain ontologies and queries.

4. They do not propose any means for evolving the employed ontologies.

5. Their approach may be challenged by the addition of new tables.

6. They do not propose any means for semantic searches in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Zhu and Hu [210] designed a semantic search methodology on Distributed Hash Tables (DHT) overlays in the environment of P2P system. DHT is the control mechanism to overlay (network) topology and data placement, which utilizes an exact-match lookup algorithm to retrieve files. The issues of DHT are that it cannot be adopted for multi-keywords search and semantic search, and has low recall and query flooding. To solve the problematic situation, the authors proposed a semantic search technique based on LSH (Locality Sensitive Hash) function. With this function, each file and query can be represented as a term vector (TV) in a VSM, then LSH generates a set of semantic keys for each TV and index files by these keys; and LSH locates each index in same nodes with high probability in P2P. Different from traditional DHT structure, the designers add extractor layer and semantic indexing and locating layer between their communications. Two approaches are utilized in the latter layer: LSH-based Semantic Indexing which is to assign the same semID for semantically similar files and to cluster semID and nodes with high probability, and LSH-based Semantic Locating which matches a query with visiting limited nodes.

Zhu and Hu's approach has the following limitations:

1. They do not propose any means for updating the semantic indexes in order to allow them to adapt to the changes of DHT files.
2. The performance of the search approach relies on the quality of queries.

The approach could be challenged by large-scale file systems.

3. They do not propose any means for semantic search in service domains.
4. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.4 Full-text Similarity Search Engines and Technologies

Full-text similarity search engines and technologies refer to the search engines and technologies designed to calculate the similarity between documents by means of semantic technologies [65].

Zhang et al. [207] delivered a semantic search methodology in a semantic portal. The search model is based on Description Logics (DL)-based information retrieval. In DL-based information retrieval, the content, the structure, the layout and the Thesaurus of a document are stored in the form of DL in a knowledge base. A Retrieval Status Value (RSV) is used to measure the relevance between two documents; a fuzzy-DL is used to assess the uncertainty between the relevance; and the concept of fuzzy interpretation is introduced to illustrate a concept and the role of concept in DL. The problem with this model is that there are no text representations for nodes in semantic graphs and thus no RSV is stored in knowledge bases. To solve the problem, the designers create a simple text representation for illustrating each node in semantic graph, and a text representation of the node is then extracted from simple text representation of surrounding nodes.

Zhang et al.'s approach may encounter the following challenges:

1. End users may find it difficult to convert an information need in a search to a well-formulated query by this model.

2. The performance of the search approach relies on the quality of queries.

3. They do not provide any means for computing the degree of similarity between queries and documents.

4. This approach is challenged by large-scale knowledge bases.

5. They do not propose any means for semantic searches in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.5 Search Engines and Technologies on Semantic/Syntactic Annotations

Search engines and technologies for semantic/syntactic annotations refer to the fact that users must specify the syntactic role of query terms when using search engines and technologies [65].

Heflin and Hendler [81] proposed the SHOE project which classifies webpages by domain ontologies. The ontologies contain properties of concepts that denote webpage information. A user needs to configure specific values of properties to search for a webpage. Similarly, Garcia et al. [61] proposed an OntoIR system and took a similar approach for semantic search. The difference is that they used DAML+OIL and RDF for ontology construction. In QuizRDF, Davies et al. [36] enhanced the ontology-based search by indexing RDF resources with full-text-based descriptions. As a result, QuizRDF enables both the ontology-based search and keyword-based search. Burton-Jones et al. [24] proposed a semantic network-based search engine, in which a self-defined semantic network is employed to obtain the synonyms, hyponyms, and antonyms of query terms. A user needs to select the correct connotation of a query term from candidates provided by the semantic network. Therefore, each query is transformed to a Boolean query and sent to a search engine. Analogously, Liang et al. [114] projected a DAML+OIL-based semantic search model. The whole design process has four phases: (1) designing domain ontologies by DAML+OIL; (2) annotating webpages; (3) collecting and storing annotation in webpages; and (4) executing searching algorithm. Searching algorithm involves two steps as follows:

- Mapping items to concepts – to determine whether a concept is the concept of an item.

- Concept-based search – Two steps are involved in concept-based search: (1) using *GetProperties* to return properties for a given concept; and (2) searching concepts and subconcepts for a given concept and using *CoreSearch* to search properties of obtained concepts for given properties. The returned results are then shown in the form of ontologies.

The limitations of the above proposals are concluded as follows:

1. The performance of the search engines relies on the quality of domain ontologies and queries.

2. They do not propose any means for evolving the domain ontologies to make them consistent with the dynamic domain knowledge.

3. They do not propose any means for dealing with the perceptual differences between ontology designers and users toward the ontologies.

4. The retrieved results could be incomplete due to the limited search scope of the navigational search and keyword-based search.

5. These approaches could be challenged by large-scale knowledge bases.

6. They do not propose any means for semantic searches in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Sheth et al. [169] proposed a Semantic Content and Retrieval Engine (SCORE). Two types of metadata are extracted from documents, which are syntactic metadata describing non-contextual information about documents and semantic metadata describing domain-specific information about documents. Based on the semantic metadata, documents are associated with context ontologies by means of automatic classification technologies. Users send queries to the system by specifying contexts and attributes values of metadata.

Sheth et al.'s approach has the following limitations:

1. The keyword-based matching style could produce incomplete results.

2. The performance of the approach relies on the quality of ontologies and queries.

3. The approach could be challenged by large-scale knowledge bases.

4. They do not propose any means for ranking retrieved results.

5. They do not propose any means for evolving the domain ontologies to make them consistent with the dynamic domain knowledge.

6. They do not propose any means for dealing with the perceptual differences between ontology designers and users toward the ontologies.

Ning et al. [145] presented a set of ranking algorithms and corresponding search algorithms for the ranked search in the environment of Semantic Web. The data representation in this research focus on a weighted directed graph G = (V, E, f), where V is a set of nodes representing resources or literals; E is a set of edges which represent properties; and f is a function which indicates strength of properties. Based on the graph, the authors developed an extended RDF tuple (s, p, o, w), where s is subject, p is property, o is object, and w is weight of p. For each node, the sum of weight probabilities is no more than 1. For global resource in the Semantic Web, a random surfer can browse objects based on the ranking of w. Based on the basic query and answer search model R = (Q; C), Q – query requirement, C – answer constraint, the authors developed a general semantic search algorithm based on the graph G, which contains three main procedures: first, the algorithm determines start nodes for spreading search in graphs based on weights of properties to request; by means of C, the scope of spreading search is constricted and divided to concept constraint search and non-concept constraint search; and queried nodes are then ranked non-increasingly according to their activation values which combine global importance values and query-dependent relevance values.

Ning et al.'s approach could have the following limitations:

1. Transition probabilities distribution is a subjective process, relying on the designers' own perception of the relevant importance of relationship between subjects and objects. Designers' perceptions and users' perceptions of such relationships cannot be reconciled due to the disparities between human understandings of any items.

2. Since global ranking values of resources are configured by designers, there is the probability of unacceptance by users, since there may be disparity between the users' perception of what constitutes importance and that of the designers. Without the function of customization, the global ranking value cannot be convincingly built.

3. The performance of the search approach relies on the quality of queries.

4. The search approach may be challenged by the large-scale documents in the Semantic Web.

5. Some inferences in the graph are uncertain, which means some statements returned from a semantic search cannot be proven by logic in practice.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Jin et al. [94] addressed a semantic search schema in a P2P network. In a P2P environment, each node contains a number of documents which relate to certain topics. Because of the semantic similarity of documents, there are links between nodes. The search engine is designed to observe the short-range links, whose nodes are semantically similar and the long-range links whose nodes have some probable relationship. In the search process, each query is defined as Q = [K, t] where K is keywords, and t is search topics. There are two search methods for different situations. When there are documents from peer matching K, the peer forwards the query to every short-range links and long-range links with high-proportional t; and when there are no documents matching K, the node forwards the query to its direct neighbours and to its long-range links with high-proportional t.

Jin et al.'s approach has the following defects:

1. They do not propose any means for updating the semantic links between nodes when documents are changed.

2. The approach could be challenged by large-scale semantic networks.

3. They do not propose any means for helping users to denote their queries with semantic technologies.

4. The performance of the search approach relies on the quality of queries.

5. The system is challenged by the distributed P2P environments.

6. The keyword-based search could omit semantically similar results.

7. They do not propose any experiments for evaluating their approach.

8. They do not propose any means for semantic search in service domains.

9. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Cesarano et al. [28] developed an agent-based semantic search engine. The whole search procedure is as follows: (1) users submit a query to a user interface combined with a depth parameter which includes the search scope, a language selection of queried webpages plus a context option for search area; (2) these parameters are then passed on to a traditional search engine to execute the search; (3) results involving hyperlinks are returned by the traditional search engine; (4) a Web Spider (WS) downloads all pages by the hyperlinks and then visits the pages which the traditional search engine cannot reach; (5) the WS stops when a predefined depth parameter is reached; (5) the document pre-processor extracts useful information for each downloaded page; and (6) the miner agent ranks these pages by referring them to a semantic knowledge base according to the similarities between the pages' information and predefined search context.

Cesarano et al.'s approach could have the following limitations:

1. The performance of the approach relies on the quality of ontologies and queries.
2. This approach could be challenged by extensive Web information.
3. They do not propose any means for disambiguating user queries by means of semantic technologies
4. They do not propose any means for updating the semantic knowledge base to enable it to adapt to the dynamic nature of the domain knowledge.
5. They do not propose any means for making agreements between ontology designers and users for ontology design.
6. They do not propose any experiments for evaluating their approach.
7. They do not propose any means for semantic search in service domains.
8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.6 Concept Search Engines and Technologies

Concept search engines and technologies refer to the semantic search engines and technologies which are designed to pursue concepts in order to denote users' query intentions [65]. The relationships between concepts are specified in taxonomies or inferred by statistical co-occurrences or similar techniques.

Lee and Tsai [109] designed a semantic search engine which can interact with users in order to better capture users' query concepts. Four agents are involved in the system. The Interface Agent interacts with users by sending users' queries and feedbacks to other agents and presenting query results to users. The Information Agent extracts keywords from webpages collected. The Discovery Agent formulates queries. The Filtering Agent re-ranks and recommends webpages to users. The four agents work continuously until users are satisfied by final search results. Meanwhile, a profile is created to record keywords extracted from webpages for automatic query generation. An evolutionary algorithm, as an iteratively cyclic mechanism, is utilized in this model to formulate queries. After a user query has been submitted, the top retrieved webpages are delivered to the Filtering Agent and re-ranked by calculating their similarities with the query by a cosine algorithm. The average value of the similarity values of the top k (k < l) webpages is then defined as the fitness of an individual. The best fitted individuals are then selected from the t+s candidates and propagated to the next generation. After a predefined number of generations, the top k pages derived from the final query of iteration are sent to the user for evaluation. With the user's feedback, the user profile is updated, and a new population is initialized and the evolutionary process starts again. This process continues until the user terminates the query.

Lee and Tsai's approach could have the following limitations:

1. The user query denotation process is an iteratively interactive process, which is time-costly for users when applying it in real-world environments.

2. The performance of the search approach relies on the quality of queries.

3. For users who often query new concepts, building a user profile is not necessary.

4. Term frequencies are not directly relevant to users' concept in mind, thus the ranking of webpages based on word lists in profiles may not match users' requests.
5. Since the matching algorithm (cosine coefficient) cannot completely reflect the semantic similarity of webpage contents, users' requests may not be fulfilled.

6. They do not propose to use well-defined domain ontologies to represent users' query intentions.

7. The methodology could miss some terms when capturing users' concepts due to retrieving limited webpages.

8. They do not propose any means for semantic search in service domains.

9. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Kwon et al. [105] proposed a meta-search engine, which provides a category-based intelligent product search for customers to retrieve products from e-commerce websites. Category is the taxonomy of products, and a number of predefined providers' categories are stored in a knowledge base. To retrieve products, a buyer needs to define a search category represented by a superclass/subclass relationship, which contains an inference node that reveals the buyer's search intention. WordNet is then utilized to extend the query terms by returning their synonyms. Next, the defined category is matched with the product categories by a category-matching algorithm, and the relevance between them is measured. Finally, the providers' categories whose relevance values are above a threshold value are ranked and displayed to buyers. A linguistic and a structural measure algorithm are integrated to measure the semantic relevance between buyers and suppliers' categories. The linguistic model determines the relevance by assigning values to different matching situations of each supplier's category. The structural model is based on computing the weights of nodes that are the distances away from their root.

The limitations of Kwon et al.'s approach can be summarized as follows:

1. If a user wants to find a general product with the name which WordNet does not commonly cite, the category search engine cannot guarantee a better result than can a keyword search engine.

2. The performance of the search approach relies on the quality of queries.

3. A user needs to define a category before searching a similar category, which is time-costly and inconvenient.

4. They do not propose any means for semantic search in service domains.

5. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Nikravesh [144] proposed a fuzzy-based framework to logically reason queries before search. A fuzzy conceptual model (FCM) is presented by the author, which is represented as a fuzzy conceptual similarity matrix consisting of: Text_sim which is a fuzzy set based on conceptual term-doc matrix (tf-idf); In_link and Out_link which are a fuzzy set to measure the number of conceptual links including both actual and virtual links; rules which are extracted from data or configured by users; and a concept referring to a precise concept extracted from data or configured by users. Based on FCM, a conceptual fuzzy match (CFM) can be used for text and image retrieval. In CFM, an ambiguous concept can be defined by a set of imprecise concepts; each imprecise concept can be defined by a set of fuzzy concepts; the fuzzy concepts relate to a set of imprecise concepts; finally, the imprecise concepts can be interpreted to precise concepts by an ontology-based ambiguity clarification dialog. The precise concepts are then passed to search engines.

Nikravesh's approach has the following limitations:

1. FCM could generate a huge number of related concepts for a given concept, which could increase the burden of search engines and the response time.

2. The ontology-based ambiguity clarification dialog needs the involvement of users, which is time-costly.

3. The disambiguation performance of FCM relies on the quality of ontologies and queries.

4. They do not propose any means for updating their ontologies so that they can adapt to the dynamic nature of domain knowledge.

5. They do not propose any means for making agreements about ontologies between

ontology designers and users of the ontologies.

6. They do not propose any experiments for evaluating their approach.
7. They do not propose any means for semantic search in service domains.
8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Han and Chen [71] presented a hybrid Web search methodology – HWS, combining traditional text search and semantic search, to improve the performance of traditional search engines. Three algorithms are adopted in the search engine, namely: BAS which is used to mine associations from existing user profile ontologies; BCH which is used to construct class hierarchies by means of a hierarchy clustering method; and MCH which merges classes into class hierarchies. A ranking algorithm utilized in HWS concerns all entities and relations, and contextual similarities between two entities. Han and Chen's methodology has the following disadvantages:

1. Class hierarchies cannot be modified by either users or designers, and so cannot adapt to changing knowledge in real environments and satisfy users' requests for structure change.
2. The semantic search is based on traditional keyword-based search, which has the issue of query flooding for each keyword in one query and thus could increase the cost of search time and resources, and increase the browsing time of users.
3. The class hierarchies cannot represent more complicated relationships between entities in domain knowledge.
4. The performance of the search approach relies on the quality of queries.
5. The approach could be challenged by large numbers of documents.
6. They do not propose any means for semantic search in service domains.
7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.7 Ontology-based Search Engines and Technologies

Ontology-based search engines and technologies refer to the semantic search engines and technologies designed to denote users' query intentions by means of ontology concepts [65]. The relationships between concepts are specified by ontologies, which are more complicated than those by concept search.

Chiang et al. [30] presented a semantic search engine based on the smart Web query

(SWQ) method for Web data retrieval. The SWQ architecture contains three main parts: SWQ search engine and its subcomponents: "*query parser*" and "*context ontology determination engine*"; context ontologies for domains of application; and a semantic search filter to improve search precision based on retrieving term properties in context ontologies. After the user enters keywords into a user interface, the SWQ engine builds a parse tree for the user's query. The search engine provides multiple relevant ontologies from context ontologies for users to select, in order to guide underlying queries. The ontologies are ranked according to the similarity between ontological terms and keywords entered by the user. The synonyms of keywords are obtained from selected ontologies, and these synonyms are then sent to traditional search engines. SWQ ranks retrieved webpages from traditional search engines based on their relevance to keywords. The pages whose ratios are below a threshold value are removed from the returned result. The slimed list of webpages is passed to three semantic search filters which utilize different domain semantics from context ontologies to evaluate the webpages – "*readability, document structure (layout) and word sense*". If the number of returned pages is greater than user's the initial configure, only top-ranked webpages are shown to the user.

The limitations of Chiang et al.'s approach are as follows:

1. The experiments show that search results from SWQ tend to be biased towards particular pages with a great number of professional terminologies, which is not appropriate for generic searches.

2. The number of domain-specific filters is limited, and the evaluation criteria are not comprehensive.

3. The performance of the SWQ search engine relies on the quality of context ontologies and queries.

4. They do not propose any means for evolving the context ontologies to make them consistent with the dynamic domain knowledge.

5. They do not propose any means for dealing with the perceptual differences between ontology designers and users toward the context ontologies.

6. The retrieved results could be incomplete due to the limited amount of information retrieved from traditional search engines.

7. They do not propose any means for semantic search in service domains.

8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Stojanovic [181] proposed a librarian agent system for analysing query ambiguity and providing corresponding refinements. When a user sends a query to the system, the system will determine the ambiguity status of the query terms and query context, and recommend some solutions to the user (senses and contexts) based on a domain ontology.

The limitations of Stojanovic's approach are as follows:

1. The performance of the search engines relies on the quality of domain ontologies and queries.
2. They do not propose any means for ranking recommended solutions.
3. The recommended solutions could flood a large-scale semantic network.
4. They do not propose any means for evolving the domain ontologies to make them consistent with the dynamic domain knowledge.
5. They do not propose any means for dealing with the perceptual differences between ontology designers and users toward the ontologies.
6. The retrieved results could be incomplete due to the limited search scope of the navigational search and keyword-based search.
7. They do not propose any means for semantic search in service domains.

Celino et al. [26, 27] proposed a semantic search engine – Squiggle, which uses ontologies for user query disambiguation. When a user submits a plain text-based query, Squiggle's Semantic Searcher will analyse it and identify the ontological elements within it by referring it to domain ontology repositories. Next, a MeaningSuggester will suggest to the user the meanings obtained from domain ontologies and the user can make a selection from them. The selected meanings will be sent to Web search engines in order to retrieve results.

Celino et al.'s approach could have the following drawbacks:

1. The performance of this approach relies on the quality of domain ontologies and queries.
2. They do not propose any means for evolving ontologies if users are unsatisfied with domain ontologies.
3. The keyword-based query term-ontological concept matching could cause retrieved concepts to be incomplete.
4. The users without enough domain knowledge could select multiple meanings, which could cause query flooding.
5. They do not propose any means for ranking retrieved results.

6. They do not propose any means for semantic search in service domains.

Bratsas et al. [19] proposed a semantic Medical Computational Problem (MCP) management system – KnowBaSICS-M, integrated with a semantic MCP repository, a semantic search engine and an ontology VSM. The repository contains an MCP ontology and associated MCP database. The search engine consists of three parts – query formulator, query processor and resultset retrieval. The formulator formats users' queries based on the attributes of ontological classes. The processor reads the formatted query from the former and then creates an instance of an execution query. The retrieval part then retrieves instances from the MCP database in terms of the query instance. The instances are later passed to the VSM which consists of a VSM constructor and a calculator. The constructor regards the retrieved instances as documents, and the instances of IndexTerm class as term vectors, which contain keywords drawn from UMLS and their weights. The similarity value of each retrieved instance is calculated by VSM, and the instances are ranked and displayed to users.

Bratsas et al.'s approach may have the following drawbacks:

1. The performance of the semantic search engine relies on the quality of MCP ontologies and queries.

2. They do not propose any means for evolving the MCP ontology in order to allow it to adapt to the dynamic domain knowledge.

3. They do not propose any means for making agreements between ontology designers and users of ontologies regarding ontology design.

4. Their approach may be challenged by the increase in the number of new documents in the MCP repository.

5. They do not propose any means for semantic search in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Liang et al. [113] proposed an inference-based association search methodology. The problem with the traditional association search is that if two instances do not have a direct relationship, the semantic search engine will not find their associations. The authors propose to solve the problem by means of Bayesian networks, which is a compact graph of node probability distribution. A Bayesian network consists of a node association graph and parameters between node associations. Ontologies can automatically provide Bayesian networks with a graphic structure. The association search has three phases: finding instances by keywords; finding association between two instances; and measuring probability of the association. The association has three instances – null, direct and indirect. The authors focused on the indirect association search and developed an algorithm for this instance.

Liang et al.'s approach has the following defects:

1. The search methodology could be challenged by the large-scale networks.
2. The keyword-based association matching approach could omit semantically similar instances.
3. The performance of the association search approach relies on the quality of ontologies and queries.
4. They do not propose to deal with the update of the associated networks.
5. They do not propose any means for evolving ontologies in order to allow them to adapt to the dynamic domain knowledge.
6. They do not propose any means for making agreements between ontology designers and users regarding ontology design.
7. They do not propose any means for semantic search in service domains.
8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

We discuss and evaluate the existing literature with regard to ontology-based multimedia searches in the rest of this subsection.

Linckels et al. [117] proposed an ontology-based lecture audio search engine in the field of e-librarian services. The e-librarian service platform consists of a domain ontology, a mechanism for identifying learning objects (recorded lectures) and a semantic search engine which returns answers of natural language for given questions. The authors deploy OWL-DL to represent the domain knowledge. In DL, the representation form of knowledge consists of concepts, roles (relations) and other constructors such as inheritance, conjunction and extensional restriction. A mechanism is developed to identify learning objects in lectures. The whole process is as follows: speech recognition software converts all lecture audios to transcripts; the transcripts are transformed to a unified format with part-of-speech tags; and the learning objects are identified from the unified transcript. The learning objects can be queried by the semantic search engine.

Linckels et al.'s approach could be challenged by the following shortcomings:

1. The DL-based queries could lead to the incompleteness of results.
2. They do not propose any means for ranking the retrieved objects by means of their similarity values with queries.
3. The performance of the approach relies on the quality of ontologies and queries.

4. They do not propose any means for updating the ontology to make it to adapt the dynamic domain knowledge.

5. They do not propose any means for making agreements between ontology designers and users of ontology design.

6. They do not propose any experiments to evaluate their approach.

7. They do not propose any means for semantic search in generic service domains.

Ding et al. [42] proposed a semantic flash search engine, by means of creating an expressive semantics for flash movies. The ranking algorithm is based on an eigenvector model which consists of two concepts: common expression and well-defined movie. Thus, the authors deduced that the above concepts can mutually and positively enhance each other. In addition, search engine can query movies by keywords from the former; and rank the movies based on the latter. An expression matrix, composed of expression vectors and movies, is designed to analyse the relationship between them and is thus used for semantic ranking.

Ding et al.'s approach could have the following defects:

1. The keyword-based query module could lead to the incompleteness of results.

2. The performance of the approach relies on the quality of ontologies and queries.

3. The resource of the search engine is based on top-50 returned results from Google^{TM}, which could lead to the incompleteness of results.

4. They do not propose any means for updating the ontology to make it to adapt the dynamic domain knowledge.

5. They do not propose any means for making agreements between ontology designers and users regarding ontology design.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.8 Semantic Web Search Engines and Technologies

Semantic Web search engines and technologies refer to the search engines and technologies that aim to capture data relationships and make resulting data queryable [65].
Guha et al. [67, 68] delivered a semantic search engine in TAP – a comprehensive Semantic Web system. The query language for semantic search in TAP is called the

GetData interface, which allows programs to visit properties of a resource in a semantic graph. Each graph is referenced by a URL, and GetData specifies resource names and property names to access to the value of property. Two additional search interfaces are provided by TAP, which are "*Search*" which searches for any properties with titles containing a given string, and "*Reflection*" which searches for incoming and outgoing tracks for a given node in a semantic graph. Two steps are executed to realize the query of these two systems: choosing a denotation – matching query terms with nodes in Semantic Web graphs based on term popularity, user profile and search context, which is done by TAP search interface; and determining what to show – determining which part (properties of nodes) in Semantic Web graphs should be shown with query results in which order. Two approaches are provided by the designers. One approach sees properties as equally relevant, and then walks through Semantic Web graphs in a "*breadth first order*", namely starting from an anchor node to a node of predefined limit. In the second approach, users manually choose nodes. The third one is a hybrid approach, whereby the second approach is executed first, and then if the nodes selected are not enough, the first approach is implemented.

The drawbacks of Guha et al.'s approach are as follows:

1. No ranking method is given when using the second approach for the step of determining which results are shown to users.

2. The results retrieved from TAP are poorly organized and non-structured, which make it difficult for users to understand.

3. The performance of the search approach relies on the quality of queries.

4. Their approach may be challenged by an increase in the number of new semantic graphs.

5. The scalability of this approach could be challenged by large-scale semantic networks.

6. Some inferences in RDF graphs are uncertain, which means some statements returned from semantic search cannot be proven by logic in practice.

7. They do not propose any means for semantic search in service domains.

Tran et al. [193] proposed an ontology-based information retrieval model, which is quadruple $[D_\Theta, Q_\Theta, F_\Theta, R_O(q_i, d_j)]$, where Θ is an ontology, D_Θ is a resource represented through a set of ontology classes $o \in \Theta$, Q_O is a user information need which can be represented by ontology elements Q'_O, F_O is an entailment operation that checks whether the resource entails the information need, and $R_O(q_i, d_j) \in [0,1]$ is a ranking function. This model enables two sorts of queries – user queries (keyword-based queries) and system queries (conjunctive queries). The user queries can be transformed to SPARQL queries with respect to a domain ontology. Relevant resources can be retrieved from RDF graphs by SPARQL queries and ranked as well as represented by this ontology-based information retrieval model.

Tran et al.'s approach could have the following limitations:

1. The performance of the search approach relies on the quality of ontologies and queries.

2. Their approach may be challenged by the increase in the number of new semantic graphs.

3. The scalability of this approach could be challenged by large-scale semantic networks.

4. Some inferences in RDF graphs are uncertain, which means some statements returned from the model cannot be proven by logic in practice.

5. They do not propose any means for evolving employed ontologies.

6. They do not propose any means for evaluating this model.

7. They do not propose any means for semantic search in service domains.

8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Bhagwat and Polyzotis [13] proposed a semantic-based file system search engine – Eureka, which uses an inference model to build the links between files and a FileRank metric to rank the files according to their semantic importance. Eureka has two main parts: (1) a crawler which extracts files from a file system and generates two kinds of indices: keywords' indices that record the keywords from the crawled files, and a rank index that records the FileRank metrics of the files; and (2) when search terms are entered, the query engine will match the search terms with keywords' indices, and determine the matched file sets and their ranking order using an information retrieval-based metrics and FileRank metrics. There are three kinds of semantic links between files defined by authors – content overlap links, name overlap links and name reference links. The authors capture the links to infer the semantic relationship between files. FileRank is the probability of visiting a file in a random order, which depends on the number of links to the file, the importance of neighbour files and the weight of links.

Bhagwat and Polyzotis could have the following shortcomings:

1. They do not propose any means for helping users to denote their query intentions by means of semantic technologies.

2. The performance of the search approach relies on the quality of queries.

3. The algorithm for computing FileRank is not given, which is an essential part of the author's prototype.

4. No implementation detail is given about how the crawler captures files and builds the file link network.

5. The authors do not propose any means to validate Eureka.

6. Their approach may be challenged by the increase in the number of new files in the file system.

7. They do not propose any means for semantic search in service domains.

8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Anyanwu et al. [3] proposed a relevance-based ranking algorithm – SemRank, for ranking complex relationships search results on the Semantic Web. Fundamental to the ranking approach is "*the ability to measure how much information is conveyed by a result thereby giving a sense of how much information a user would gain by being informed about the existence of the result*". The variables utilized for measuring the predictability of a result include: the "*uniqueness*" and "*specificity*" of the result; and how different the result is from the possibilities gained from the schema. This ranking algorithm is used in Semantic Search of a Different Kind (SSARK) system, which has three main phases. First, RDF documents are loaded by a loader then processed by pre-processor as a path sequence which is a group of RDF graphs extracted from RDF documents for evaluation; and the storage manager then stores the path sequence in a database. When a query is given, the query processor chooses proper path sequences and groups them as an Annotated Path Expression Tree (APET) to users. Second, the paths in an APET are annotated by SemRank values computed from three main indexes. Third, the paths in a returned APET are ordered by counting their annotated values and then shown to users.

Anyanwu et al.'s approach could have the following limitations:

1. They do not propose any means for retrieving other ontology mark-up language-annotated documents.

2. The retrieved RDF graphs may be incomplete due to the limited capability of the loader.

3. The performance of the search approach relies on the quality of queries.

4. The scalability of their approach could be challenged by large-scale semantic networks.

5. Some inferences in RDF graphs are uncertain, which means some statements returned from semantic search cannot be proven by logic in practice.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

GiveALink is a public website where users can share their favourite bookmarks. Stoilova et al. [179] proposed a semantic search engine for bookmark search. A URL similarity measure algorithm combines both hierarchical structures of bookmark ontologies and collaborative filtering, thereby integrating common patterns in users' preferences. When users submit URL queries in GiveALink query interface, the search engine will return the URLs with high similarity values based on a similarity matrix. If the search engine cannot find related bookmarks, the query URL is sent to Google^{TM} and the top ten returned URLs will be used to look for bookmarks in the database again. The queried results are ranked according to the combination of bookmark similarities, generality, prestige and novelty. Prestige originates from PageRank, and the prestige of a URL depends on the total prestige of its neighbours. Here, the authors computed the prestige of a bookmark based on the indirect similarity values between the bookmark and other bookmarks. Generality is central to a bookmark in the similarity matrix, which is the average of the shortest-path similarities between the bookmark and other bookmarks. Novelty is used to recommend those bookmarks whose indirect shortest-path similarities are higher than the direct similarities.

Stoilova et al.'s approach could suffer the following drawbacks:

1. The search engine retrieves only the top ten results from Google^{TM}, thereby possibly missing many valuable bookmarks.

2. The performance of the URL similarity measure algorithm relies on the quality of bookmark ontologies and queries.

3. They do not propose any means for evolving the bookmark ontologies.

4. They do not propose any means for making agreements about the bookmark ontologies between ontology designers and users.

5. They do not propose any means for semantic search in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Rocha et al. [159] built a hybrid search engine combining traditional text search methodology and a spread activation algorithm over the Semantic Web. Three different measures are counted for weighting relationships between nodes in a semantic graph: 1) a cluster measure which assesses the similarities between nodes; 2) a specificity measure which assesses the differentiation or specificity between nodes; and, 3) a combined measure which integrates the former two measures. A spread activation algorithm is utilized to find related nodes based on the principle that from some starting nodes which have an "*initial activation value*" namely the weight of a relation to the searching task, the instance is chosen which has the highest value and is matched with constraints; and then its neighbours are activated. Constraints have three basic categories – concept type constraints on which activated nodes must be given types, fan-out constraints on which the spread must stop when more than one node are connected to the processed nodes, and distance constraint which specifies the depth of spread from initial instance.

Rocha et al.'s approach may meet the following challenges:

1. The approach could be challenged by large-scale semantic networks.
2. The performance of the search approach relies on the quality of queries.
3. Spread activation algorithm has the issue of query incompleteness and query flooding.
4. There is no semantic interpretation of the activation value flowing through the network.
5. Some inferences in the graph are uncertain, which means some statements returned from semantic search cannot be proven by logic in practice.
6. They do not propose any means for semantic search in service domains.
7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Maedche et al. [133] designed an integrated approach for ontology searching, reusing and updating. In its architecture, an ontology registry is designed to store the metadata about ontologies and an ontology server stores the ontologies. The ontologies in distributed ontology servers can be created, replicated and evolved. Ontology metadata in an ontology registry can be queried and registered when a new ontology is created. Ontology search in ontology registry is executed under two conditions: query-by-example which restricts search fields and search terms, and query-by-term which restricts the hyponyms of terms for search.

Maedche et al.'s approach could have the following shortcomings:

1. The large-scale semantic networks may make it difficult to improve the efficiency of this approach.
2. The performance of the search approach relies on the quality of queries.

3. Incompleteness of search result is another issue because the search could miss some results when searching in a huge semantic tree.

4. Halting problem – all relations in semantic search have the feather of uncertainty, which means some statements returned from semantic search cannot be proven by logic in practice.

5. This approach could be challenged by distributed ontology registries.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Swoogle is a crawler-based semantic search engine. Three main functions are provided by Swoogle, which are: finding appropriate ontologies for specific terms involved; finding instance data – Semantic Web documents (SWDs) defined by specific classes and properties; and characterizing the Semantic Web – by establishing interrelationships among metadata in Semantic Web, Swoogle is able to answer the questions about Semantic Web structure. The architecture of Swoogle consists of three basic parts. The SWD discovery searches SWDs and maintains the latest record of these SWDs. The metadata creation stores SWDs, and caches and creates metadata about semantics and syntax information for each SWD. The data analysis analyses the SWDs and metadata to generate a report about classification of Semantic Web ontologies (SWOs), ranking and indexing of SWDs. Swoogle uses Sire as its indexing and retrieval method. Sire is a custom indexing and retrieval engine which uses typical words and N-gram – "a*n n-character segment of the text which spans inter-word boundaries*" combined with a tf-idf model with a standard cosine similarity metric [47]. Similarly, Hogan et al. [80, 86] proposed a framework of Semantic Web Search Engine (SWSE), which enables search over RDF graphs. First of all, a MultiCrawler [78] is employed to extract RDF triples from non-semantic (HTML or XML) documents. Next, the generated RDF datasets are integrated by considering the same URIs, shared values of inverse functional properties, and seeAlso properties. Once a query has been sent to SWSE, the matched entities from the integrated RDF graphs are classified by their types. Users can then select the types to browse the results. The results are ranked by considering both their tf-idf weights and link weights.

Their approach may have the following challenges:

1. The large-scale semantic networks may make it difficult to improve the efficiency of this approach.

2. The performance of the search approach relies on the quality of queries.

3. Incompleteness of search result is another issue because the search could miss some results when searching in a huge semantic tree.

4. Halting problem – all relations in semantic have the feather of uncertainty, which means that some statements returned from semantic search cannot be proven by logic in practice.

5. The search engines and the crawlers could be challenged by distributed Web environments.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

In 2006, Völkel et al. [198] created Semantic Wikipedia, the aim of which is to enhance Wikipedia with Semantic Web technologies. Auer et al. [5] proposed a DBpedia project, which converts Wikipedia content into RDF dataset. Moreover, DBpedia are interlinked with other open datasets by RDF links. A Search DBpedia.org was designed by the authors, enabling users to use keyword-based search to query DBpedia dataset and interlinked dataset. The retrieved result can be ranked by combining the number of incoming links, relevance of linked resources and relation depth. Similar projects also include SweetWiki [22], Semantic MultimediaWiki [104] and IkeWiki [164].

Their approaches may be challenged by the following issues:

1. The large-scale semantic networks could lead to difficulty in improving the efficiency of the approaches.

2. The performance of the search approach relies on the quality of queries.

3. Incompleteness of search results is another issue because the search could miss some result when searching in a huge semantic tree.

4. Halting problem – all relations in semantic have the feather of uncertainty, which means that some statements returned from semantic search cannot be proven by logic in practice.

5. They do not propose any means for semantic search in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.9 Faceted Search Engines and Technologies

Faceted search engines and technologies refer to the semantic search engines and technologies that provide a set of predefined, high-level categories called *facets* for result categorization [65].

Dichev and Dicheva [40] produced a view-based semantic search engine in the context of a topic-centred learning repository, by means of an extension of the topic maps (TM) model which is a lightweight ontology model constructed by topics and relationships between topics. The TM is implemented in a TM4L environment, which is "*an environment for building and using ontology-aware learning repositories represented by topics*" [41]. A view in TM is defined as a collection of related topics, occurrences of topics, associations between topics and scopes of topics. The view-based semantic search in a TM4L environment includes two phases: transforming a view-based query to a traversal expression and then locating some corresponding resources; and using the retrieved resources to locate other relevant resources. Similarly, Hyvönen et al. [89, 90, 135] proposed a semantic view-based search engine – Ontogator, which uses ontological view-based search. The authors used ontologies for image file classification and visualized the ontologies for view-based search. Users can browse images based on ontological properties. Based on Ontogator, Hyvönen et al. [91] proposed another project – MuseumFinland, which enables users to browse and search over 4000 cultural artefacts from 260 historical sites in Finland. Oren et al. [146] proposed a faceted interface for RDF browsing. Apart from defining ontology-based facets, the authors also proposed a set of "*navigation quality*" metrics for facets ranking.

Their approach has the following shortcomings:

1. The performance of the search engines relies on the quality of ontologies or facets and queries.

2. They do not propose any means for evolving the ontologies or facets to adapt the dynamic service domain knowledge.

3. They do not propose any means to deal with the perceptual differences between designers and users toward the ontologies or facets.

4. Their approach may be challenged by the increase of resources in the repositories.

5. The search engines may be challenged by heterogeneous and distributed repositories.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Hildebrand et al. [83] proposed a /facet (or called Slashfacet) browser for facet search over distributed and heterogeneous Semantic Web repositories. Since resources in difference repositories have the features of heterogeneousness and diversity, it is difficult to use one type of facet to classify all resources. Consequently, it is necessary to create multi-type facets for resource browsing. The authors thus proposed a multi-facet-based search approach, which enables users to search facet types before a facet search by means of a keyword-based search.

The facet approach has the following limitations:

1. The performance of the search engines relies on the quality of facets and queries.
2. They do not propose any means for evolving the facets to adapt the dynamic service domain knowledge.
3. They do not propose any means for dealing with the perceptual differences between designers and users toward the facets.
4. Their approach may be challenged by the increase of resources in repositories.
5. They do not propose any means for semantic search in service domains.
6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Can et al. [25] proposed a semantic search engine – MedicoPort for the users with limited knowledge of health, a domain in which knowledge is formulated with Unified Medical Language System (UMLS). UMLS contains three parts – a specialist lexicon, a UMLS semantic network and a metathesaurus. This system has three main components. The first components are a query formulator and a concept generator. The query formulator extracts medical terms from user queries by means of the specialist lexicon. Based on the terms, the UMLS semantic network and metathesaurus are utilized to seek their synonyms, words with partial relations and contextual relations. Thus user queries are conceptualized and expanded. By means of UMLS, a focused crawler, Lokman, is adopted in order to continuously harvest and index the URLs of medical literature from the internet to the knowledge base, by calculating the semantic relevance between UMLS topics and queried documents. Finally, the search and query module is used to search and rank relevant literature according to the formulated query terms in the knowledge base.

Can et al.'s approach may meet the following challenges:

1. The performance of the search engine relies on concept generation (a 3.3% decrease in precision when the latter is not used).
2. In the search and query phase, in order to calculate the relevance between query terms and document terms, the authors empirically assigned different weights to the words with different relations to query terms, which lacks theoretical supports.
3. The performance of the search approach relies on the quality of queries.
4. They do not propose any means for dealing with the update of the UMLS semantic network.

5. Their approach may be challenged by the increase in the number of new documents crawled by the Lokman Crawler.

6. The search engine and crawler may be challenged by distributed knowledge bases.
7. They do not propose any means for semantic search in service domains.

8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Khan et al. [99] proposed an ontology-based audio retrieval method. Firstly, metadata is generated from audio files by extracting their contents. A domain ontology consisting of concepts, individual and their synonyms is employed to classify the metadata. Users can choose keyword-based query or SQL query to retrieve the audio files. For the former, tokens are generated from queries and matched with concepts from the ontology with depth-first search and breadth-first search. The similarity value between concepts and queries is computed by referring to the correlations between concept-to-concept, and concept-to-query, with the purpose of concept pruning. For the latter, if the "*where*" restrictions of a SQL query occur in parent concepts, all their children concepts are added into the "*where*" restrictions; otherwise not.

Khan et al.'s approach may have the following shortcomings:

1. The keyword-based query-concept matching may result in retrieved concepts being incomplete.

2. The performance of the approach relies on the quality of ontologies and queries.

3. They do not propose any means for updating the ontology to make it to adapt the dynamic domain knowledge.

4. They do not propose any means for making agreements between ontology designers and users regarding ontology design.

5. This approach may be challenged by large-scale knowledge bases.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Bonino et al. [16, 17] proposed an Holistic Distributed Open Semantic Elaboration (H-DOSE) platform. Based on the platform, they designed a framework for multimedia resource classification and search. First of all, they defined multiple perspectives of multimedia resources in the form of "*anchors*" – RDF-annotated views. Secondly, the domain ontologies that classify multimedia resources are associated with the anchors.

Therefore, multimedia resources are associated with the anchors. When a user query is sent to H-DOSE, the query is converted into a ranked list of anchors. The anchors are ranked according to their relevance to the query.

The H-DOSE-based multimedia resource classification and search approach could have the following limitations:

1. They do not reveal the technical details regarding how to rank anchors by their relevance.
2. The performance of the search approach relies on the quality of queries.
3. Their approach could lead to query incompleteness or query flooding.
4. The scalability of their approach could be challenged by large-scale knowledge bases.
5. Their approach could be challenged by distributed Web environments.
6. They do not propose any means for semantic search in service domains.
7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Petel et al. [151] developed a Semantic Web portal – OntoKhoj, for the purpose of ontology search, ranking and classification. The utilized ontology classification model is derived from DMOZ, which has a great number of manually created ontological concept hierarchies. When a new ontology is retrieved, the ontology is matched with the model to decide the discipline to which it belongs. An OntoRank algorithm is designed based on the PageRank algorithm, to rank the retrieved ontologies for a given topic. The difference between PageRank and OntoRank is that the latter takes into account not only the number of references between ontologies, but also includes the weight of the references between ontologies.

Petel et al.'s approach could meet the following challenges:

1. The large-scale semantic networks could lead to difficulty in improving the efficiency of this approach.
2. The performance of the search approach relies on the quality of queries.
3. Incompleteness of search result is another issue which the search could miss some result when searching in a huge semantic tree.

4. Halting problem – all relations in semantic have the feather of uncertainty, which means that some statements returned from semantic search cannot be proven by logic in practice.

5. This approach could be challenged by distributed Web environments.

6. They do not propose any means for semantic search in service domains.

7. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.10 Clustered Search Engines and Technologies

Clustered search engines and technologies refer to the semantic search engines and technologies that cluster retrieved objects without predefined categories [65].

Hoi and Lyu [87] proposed a semantic image search schema, by means of learning concepts from image descriptions. The overall structure of the proposed system contains two major steps:

- Searching and clustering Web images. When users enter query terms into the search engine, it starts to collect a pool of images whose descriptions are associated with the terms; and it then clusters them according to k-means algorithm. Finally, the top-ranked clusters are chosen as a training set.
- Learning semantic concepts by support vector machine (SVM) technique. One-class SVM is utilized to learn concepts by generalizing and classifying the training set, which produces the preliminary search results. The search results are then processed by two-class VSM that uses a feedback relevance technique, in order to improve the retrieval performance.

Hoi and Lyu's approach could have the following limitations:

1. This approach could be challenged by vast image information.

2. The experiment shows that the precision decreases when the numbers of results in each cluster increases.

3. The learned concepts might not have any meaning to users as a result of the lack of semantic supports.

4. The performance of this approach relies on the quality of image descriptions and queries.

5. They do not propose any means for semantic search in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Wang et al. [200] proposed an image search engine – IGroup, by means of semantic clustering technique. With IGroup, users can refine query results by a navigation panel which contains a list of clusters and each cluster is represented with a thumbnail and a cluster name. The overview of Web image clustering process has two major steps:

- Learning candidate image cluster names. When users enter queries into a text search engine, the text search engine returns a list of ranked results. IGroup analyses the titles and descriptions of the results, by parsing them into several phrases and then calculating the properties for each phrase. The properties are integrated as a salience score, and the phrases are ranked based on the salience score. The top-ranked phrases are chosen as salient phrases.
- Merging and pruning cluster names. This process is designed to determine the final cluster name based on given candidate cluster names. First of all, the cluster names are merged according to their similarity values. Secondly, unimportant words are removed from these merged names. The remaining names are then tested with a search engine, and the names with too many results or too few results are removed. Finally, the remaining names are regarded as the final cluster names.

Wang et al.'s approach has the following shortcomings:

1. IGroup is a keyword-based search engine, which may miss semantically similar images with incomplete queries.

2. This approach could be challenged by vast image information.

3. The cluster names might not have any meaning to users as a result of the lack of semantic supports.

4. The performance of this approach relies on the quality of image descriptions and queries.

5. They do not propose any means for semantic search in service domains.

6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Schreiber et al. [166] proposed a MultimediaN E-Culture project, using Semantic Web technologies to annotate and index virtual collections of cultural heritage resources. The aim of the project is to enable users to use simple keywords to search the annotated resources. The search algorithm involves several steps: 1) the keywords are matched with all RDF triples in the repository; 2) the search engine traverses the matched RDF graphs to find target resources; and 3) the target resources are clustered based on the path between the matched RDF triples and target resources.

Several issues emerging from this research are:

1. They do not propose any experiments for evaluating their search algorithm.
2. The approach could be challenged by large-scale knowledge bases.
3. The keyword-based search could cause query incompleteness
4. The performance of this approach relies on the quality of resource descriptions and queries.
5. They do not propose any means for semantic search in service domains.
6. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

2.4.11 Natural Language Search Engines and Technologies

Natural language search engines and technologies refer to the semantic search engines and technologies designed to seek answers for natural language questions by means of semantic technologies [65].

Shin et al. [171] proposed a cooperative query-answering approach based on a semantic knowledge representation framework – Metricized Knowledge Abstraction Hierarchy (MKAH). Cooperative query answering is able to provide the function of query relaxation and approximate answering, which can extend the query scope. In MKAH, two sorts of knowledge representation hierarchies exist: value abstraction hierarchies and domain abstraction hierarchies. The former represents is-a relationships between objects, and the latter represents instance-of relationships between objects. Based on the distance between nodes/objects in the hierarchies, the similarity relevance between objects can be measured. By means of the former hierarchies, basic SQL queries are expanded based on the queries toward semantically relevant objects at the same level. By means of the latter hierarchies, SQL queries are expanded based on the queries towards instances of objects.

Shin et al.'s approach has the following drawbacks:

1. The hierarchical knowledge structure cannot represent more complicated relationships between objects.
2. Performance relies on the quality of MKAH and queries.
3. They do not propose any means for updating MKAH.

4. The approach could be challenged by large-scale databases, as the possibly retrieved large number of results from the expanded SQL queries.

5. They do not propose any experiments for evaluating their approach.

6. SQL queries may cause query incompleteness.

7. They do not propose any means for semantic search in service domains.

8. They do not provide any means for QoS-based indexing of retrieved results (especially for service domains).

Lopez et al. [62, 126, 127] proposed an ontology-driven, question-answering system. By means of a GATE parser [35], natural language queries are translated into linguistic triples in the form of term-relation and classified into categories. A Related Similarity Service (RSS) is then employed to further translate the triples into ontology-compatible queries. Domain ontologies are employed to make sense of the query triples, collaborated with string similarity matching, general lexical resources and a domain-independent lexicon generated by a learning mechanism. The ontology-compatible queries are then matched with RDF triples for retrieving answers.

Lopez et al.'s approach has the following defects:

1. The keyword-based query-concept matching may cause query incompleteness.

2. They do not propose any means to rank results based on their similarity values with queries.

3. This approach may be challenged by large-scale knowledge bases.

4. The performance of this approach relies on the quality of ontologies and queries.

5. They do not propose any means for semantic search in service domains.

Bernstein et al. [11, 12, 95] proposed a framework of a Ginseng search engine. By using ontologies as Thesauruses, Ginseng enables users to use natural languages to query OWL knowledge bases. When user entered queries, Ginseng uses query-completion popup/choice windows to recommend users with query terms that originate from ontologies. The architecture of Ginseng comprises three parts: 1) a multi-level grammar which specifies the generally possible query sentence structure and gets dynamic grammar rules from ontologies; 2) an incremental parser which uses the multi-level grammar to provide users with query choices and generate SPARQL queries based in user queries; and 3) an ontology access layer which is used to execute the SPARQL queries.

Bernstein et al.'s approach could meet the following challenges:

1. Ginseng cannot process all natural language questions based on its limited knowledge bases.

2. The current SPARQL does not provide aggregated search functions (search based on multiple concepts).

3. This approach could be challenged by large-scale knowledge bases.

4. This approach could lead to the problem of query incompleteness.

5. The search performance relies on the quality of ontologies and queries.

6. They do not propose any means for ranking retrieved results.

7. They do not propose any means for semantic search in service domains.

2.5 Semantic Similarity Models

Semantic relatedness refers to human judgment about the extent to which a given pair of concepts are related to each other [152]. Studies have shown that most people agree on the relative semantic relatedness of most of pairs of concepts [139, 161]. Therefore, many technologies have been developed to date in order to precisely measure the extent of similarity relatedness and similarity between concepts in multiple disciplines, such as information retrieval (IR) [85, 108, 155, 173, 176, 185], natural language processing (NLP) [112, 116, 156, 160], linguistics [189], health informatics [177], bioinformatics [29, 34, 147, 152, 168], Web services [122] and other fields. In the field of IR and NLP, the researches primarily focus on word sense disambiguation [156, 185], multimodal document retrieval [162], text segmentation [116, 173] and query preciseness enhancement [85, 108]. In the linguistics area, the researches emphasize computing semantic similarity between uncertain or imprecise concept labels [189]. In the health domain, the researchers are mainly concerned with seeking similar health science terms. In the field of bioinformatics, the focus is on the measurement of the similarity between concepts from the gene ontology [29, 34, 147, 168]. In the field of Web services, the researchers concentrate on semantic service discovery [122]. Moreover, the semantic similarity models can also be used to estimate the similarity between land use/land cover classification system [57].

Current similarity measures between concepts are manifold and can be classified into three main categories according to the utilized information: edge (distance)-based models [84, 107, 108, 155, 157, 185, 201], node (information content)-based models [115, 153, 156], and hybrid models [93, 112, 134, 212]. In the rest of the section, we will briefly introduce the three categories and the typical models in each category, and analyse their limitations when applying these to the ontology environment.

2.5.1 Edge (Distance)-based Models

Edge-based models are based on the shortest path between two nodes in a definitional network. Definitional networks are a type of hierarchical/taxonomic semantic networks, in which all nodes are linked by *is-a* relations [174]. The models are based on the assumption that all nodes are evenly distributed and are of similar densities and the distance between any two nodes is equal. They can also be applied to a network structure.

One typical edge-based model is provided by Rada [155], and is described as:

For two node C_1 and C_2 in a semantic network,

$$\text{Distance } (C_1, C_2) = \text{minimum number of edges seperating } C_1 \text{ and } C_2 \tag{2.1}$$

and the similarity between C_1 and C_2 is given by

$$sim_{Rada}(C_1, C_2) \ = \ 2 \times Max - \text{Distance}(C_1, C_2) \tag{2.2}$$

where Max is the maximum depth of a definitional network.

In order to ensure that the interval of sim_{Rada} is between 0 and 1, Equation 2 can also be expressed as

$$sim_{Rada}(C_1, C_2) = 1 - \frac{\text{Distance}(C_1, C_2)}{2 \times Max} \tag{2.3}$$

Leacock et al. [107] consider that the number of edges on the shortest path between two nodes should be normalized by the depth of a taxonomic structure, which is expressed mathematically as

$$\text{Distance } (C_1, C_2) = \frac{\text{minimum number of edges separating } C_1 \text{ and } C_2}{2 \times Max} \tag{2.4}$$

and the similarity between C_1 and C_2 is given by

$$sim_{Leacock}(C_1, C_2) = \text{-}\log(\text{Distance } (C_1, C_2)) \tag{2.5}$$

Wu and Palmer[201] are concerned with the node that subsumes two nodes when computing the similarity between the two nodes, which can be expressed mathematically as follows:

$$sim_{Wu\&Palmer}(C_1, C_2) = \frac{2 \times N_3}{N_1 + N_2 + 2 \times N_3} \tag{2.6}$$

where C_3 is the most informative node that subsumes C_1 and C_2, N_1 is the minimum number of edges from C_1 to C_3, N_2 is the minimum number of edges from C_2 to C_3, N_3 is the depth of C_3.

Sussna [185] considered the factors of network density, node depth and type of link in the node distance computation. The weights between node A and B can be obtained as follows:

$$wt(A, B) = \frac{wt(A \rightarrow_r B) + wt(B \rightarrow_{r'} A)}{2Max(d_A, d_B)} \tag{2.7}$$

$$\text{with } wt(x \rightarrow_r y) = Max_r - \frac{Max_r - Min_r}{n_r(x)} \tag{2.8}$$

where $\rightarrow_r$ is a relation of type r, $\rightarrow_{r'}$ is its reverse, d_A is the depth of the node, Max_r and Min_r are the maximum and the minimum possible weights for the relation type r, and $n_r(x)$ is the number of relations type r leaving node x.

Zuber and Faltings [212] proposed an Ontology Structure based Similarity (OSS) model, which measure the semantic distance between two concepts in a hierarchy by three steps as follows:

Step 1. The a-priori score (APS) of each concept in the hierarchy is inferred by

$$APS(C) = \frac{1}{n+2} \tag{2.9}$$

where n is number of decedents of concept C.

Step 2. How much is transferred from C_2 to C_1 is obtained based on APS, which is calculated by observing trends of inferences (paths) from C_2 to C_1, which can be presented as follows:

$$\begin{cases} T(x, y) = \hat{\alpha}(x, y) & \nearrow \\ T(y, z) = 1 + \dfrac{\hat{\beta}(z, y)}{1 + 2\hat{\beta}(z, y)} & \searrow \end{cases} \tag{2.10}$$

$$\text{with } \begin{cases} \hat{\alpha}(x, y) = APS(x) / APS(y) \\ \hat{\beta}(z, y) = APS(z) - APS(y) \end{cases} \tag{2.11}$$

where x, y, z are APS of nodes on the path from C_2 to C_1, T is the transfer value between any two nodes on the path, $\nearrow$ and $\searrow$ respectively represent the upward (child concept to parent concept) and downward path (parent concept to child concept) between two nodes.

Step 3. The transfer of the score is transformed to the distance between C_2 and C_1 by

$$T(C_1, C_2) = -\frac{\log T(C_1, C_2)}{\max D} \tag{2.12}$$

where maxD is the longest distance between any two concepts in the ontology.

The limitations of the edge-based models are as follows:

1. The distances between any two adjacent nodes are not necessarily equal.
2. They do not consider the fact that the difference between nodes decreases along with the increase of depth of hierarchy, as a result of the increasingly reduced definitional differences.
3. Most of the edge-based models consider only the *is-a* relationship and ignore other types of relationships.
4. They ignore content of nodes when estimating similarity values.
5. They do not propose any means for the similarity measure in the ontology environment owing to the fact that they are all designed for the semantic network environment.

2.5.2 Node (Information Content)-based Models

Information content-based models are used to judge the semantic similarity between concepts in a definitional network or in a corpus, based on measuring the similarity by taking into account information content, namely the term occurrence in corpora or the subsumed nodes in taxonomies. These models can avoid the defect of the edge counting approaches which cannot control variable distances in a dense definitional network [156].

Resnik [156] developed such a model whereby the information shared by two concepts can be indicated by the concept which subsumes the two concepts in a taxonomy. Then, the similarity between the two concepts C_1 and C_2 can be mathematically expressed as follows:

$$sim_{\text{Resnik}}(C_1, C_2) = \max_{C \in S(C_1, C_2)}[-\log(\mathrm{P}(C))] \tag{2.13}$$

where S(C$_1$, C$_2$) is the set of concepts that subsume both C$_1$ and C$_2$, and P(C) is the possibility of encountering an instance of concept C.

Lin [115]'s semantic similarity model is the extension of Resnik's model, which measures the similarity between two nodes as the ratio between the amount of commonly shared information of the two nodes and the amount of information of the two nodes, which can be mathematically expressed as follows:

$$sim_{Lin} = \frac{2 \times sim_{\mathrm{Re}snilk}(C_1, C_2)}{IC(C_1) + IC(C_2)} \tag{2.14}$$

Pirro [153] proposed a feature-based similarity model, which is based on Tversky's theory that the similarity between two concepts is the function of common features between the two concepts minus those in each concept but not in another concept [194]. By integrating Resnik's model, the similarity model can be mathematically expressed as follows:

$$sim_{P\&S}(C_1, C_2) = \begin{cases} 3 \times sim_{\mathrm{Re}snik}(C_1, C_2) - IC(C_1) - IC(C_2) & if\ C_1 \neq C_2 \\ 1 & if\ C_1 = C_2 \end{cases} \tag{2.15}$$

The node-based models have the following drawbacks:

1. Most of the models are designed only for definitional networks and thus ignore other types of relationships between nodes.
2. They do not differentiate the similarity values of any pair of concepts in a sub-hierarchy as long as their lowest super-ordinate concept is the same.
3. For the information-content-based corpus similarity measure, polysemous words will have an exaggerated content value if only word frequency data are used.
4. They do not consider the semantic-rich contents of ontology concepts when calculating similarity values.
5. They are all designed for the semantic network environment but not for the ontology environment.

2.5.3 Hybrid Models

Hybrid models use multiple factors for similarity measurement. Jiang and Conath [93] developed a hybrid model that uses the node-based theory to enhance the edge-based model. Their method takes into account the factors of local density, node depth and link types. The weight between a child concept C and its parent concept P can be measured as:

$$wt(C,P) = (\beta + (1-\beta)\frac{\bar{E}}{E(P)})(\frac{d(P)+1}{d(P)})^{\alpha}(IC(C) - IC(P))T(C,P) \qquad (2.16)$$

where d(P) is the depth of node P, E(P) is the number of edges in the child links, $\bar{E}$ is the average density of the whole hierarchy, T(C, P) represents the link type, and α and β (α ≥ 0, 0 ≤ β ≤ 1) are the control parameters of the effect of node density and node depth on the weight.

The distance between two concepts is defined as follows:

$$\text{Distance}(C_1,C_2) = \sum_{C \in \{path(C_1,C_2) - LS(C_1,C_2)\}} wt(C, p(C)) \qquad (2.17)$$

where path(C_1, C_2) is the set that contains all the nodes in the shortest path from C_1 to C_2, and LS(C_1, C_2) is the most informative concept that subsumes both C_1 and C_2.

In some special cases, such as when only the link type is considered as the factor of weight computing (α=0, β=1 and T (C, P) =1), the distance algorithm can be simplified as follows:

$$\text{Distance}(C_1,C_2) = IC(C_1) + IC(C_2) - 2 \times sim_{\mathrm{Re}snik}(C_1,C_2) \qquad (2.18)$$

where IC(C) = -logP(C).

Finally, the similarity value between two concepts C_1 and C_2 is measured by converting the semantic distance as follows:

$$sim_{Jiang\&Conath}(C_1,C_2) = 1 - \text{Distance}(C_1,C_2) \qquad (2.19)$$

In addition, Seco [167]'s research showed that the similarity equation can also be expressed as

$$sim_{Jiang\&Conath}(C_1,C_2) = 1 - \frac{\text{Distance}(C_1,C_2)}{2} \qquad (2.20)$$

The test results show that the parameters α and β do not significantly influence the similarity computation [93].

Li et al. [112] proposed a hybrid semantic similarity model combining structural semantic information in a nonlinear model. The factors of path length, depth and density are considered in the assessment, which can be mathematically expressed as

$$sim_{Li}(C_1,C_2)=\begin{cases} e^{-\alpha l}\cdot\dfrac{e^{\beta h}-e^{-\beta h}}{e^{\beta h}+e^{-\beta h}} & if\ C_1\neq C_2 \\ 1 & if\ C_1=C_2 \end{cases} \tag{2.21}$$

where l is the shortest path length between C_1 and C_2, h is the depth of the subsumer of C_1 and C_2, α and β are the effect of l and h to the similarity measure.

The hybrid models could meet the following challenges:

1. The hybrid models need the factors of local density, node depth and link types to be weighted before measuring concept similarity. However, they do not provide any means for weighting these factors and thus it is difficult to realize in the real environment.

2. They do not consider the rich-semantic contents of ontology concepts when calculating similarity values.

3. They are all designed for the semantic network environment but not for the ontology environment.

2.6 Critical Evaluation of Existing Technologies: An Integrative View

This section extends the critical evaluation of existing approaches carried out in the previous sections. It attempts an integrative view to discover the main issues contained in the literature that need to be addressed. As a general note, basically no attempt has been made to devise a complete methodology for service retrieval in the Digital Ecosystems environment that takes into account the heterogeneous, context-dependent, and dynamic nature of Digital Ecosystem services, and the heterogeneous and dynamic nature of service providers and service requesters. The main contingencies of the above approach for service retrieval in the Digital Ecosystems environment are summarized into five areas, as discussed below:

1. There is no methodology for discovering service information (including service advertisements and service provider profiles) in Digital Ecosystems that takes into account the heterogeneous and context-dependent nature of services and the heterogeneous nature of service providers in Digital Ecosystems.

2. There is no methodology for retrieving service information in Digital Ecosystems that takes into account the heterogeneous nature of Digital Ecosystem service requesters.

3. There is no methodology for recommending relevant service knowledge to service requesters who do not have domain knowledge as to their service requests in Digital Ecosystems that takes into account the heterogeneous nature of Digital Ecosystem service requesters.

4. There is no methodology for QoS-based service ranking in Digital Ecosystems that takes into account the context-dependent nature of Digital Ecosystem services.
5. There is no methodology for service domain knowledge updating that takes into account the dynamic nature of Digital Ecosystem services and service providers.

2.6.1 Service Information Discovery in Digital Ecosystems

As described in Chapter 1, information regarding service entities is heterogeneous and context-dependent, and information regarding service providers is heterogeneous in Digital Ecosystems. Therefore, we need a methodology to discover service entities and relevant service providers from the Digital Ecosystems environment. First of all, since information regarding service entities and service provider is nonstandard and intermingled with other Web information, we need a methodology to exploit and standardize the information. Second, since service entity information (or service advertisements) is heterogeneous and context-dependent, we need a methodology to define specific knowledge in Digital Ecosystem service contexts and use the knowledge to clarify and semanticize service entity information.

As can be inferred from the discussion in Section 2.2 and 2.3, a significant majority of the existing approaches in the literature cannot fulfil the above two requirements for service information discovery in the Digital Ecosystems environment, which needs to be addressed as follows:

As can be inferred from the discussion in Section 2.2.1, none of the metadata abstraction crawlers works on service domains. Due to the heterogeneous nature of Digital Ecosystem services and service providers, the domain-specific service and service provider metadata schema need to be regulated, which can be employed to standardize the information regarding service entities and service providers. In addition, none of the crawlers uses semantic technologies to annotate and clarify service entity information. In Section 2.2.2, we show that none of the semantic focused crawlers is designed for service domains, except for Toch et al.'s approach [191]. However, their approach focuses only on the Web service field rather than the Digital Ecosystem service fields which are broader. Moreover, semantic focused crawlers cannot generate metadata and they can use only existing metadata. In Section 2.2.3, we explain how none of the metadata harvesting crawlers is designed for service domains. Furthermore, they cannot generate metadata or semanticize metadata by means of specific service domain knowledge. From our discussion in Section 2.2.4, it appears that none of the metadata abstraction focused crawlers is designed for service domains. Lastly, in regards to metadata harvesting and abstraction crawlers discussed in Section 2.2.5, none of them is designed for service fields. Furthermore, they do not have the function of semanticizing metadata by means of specific service domain knowledge. Therefore, it can be concluded that the existing approaches in the field of semantic crawlers cannot fulfil the requirements of service discovery in the Digital Ecosystems environment.

As can be inferred from the discussion in Section 2.3, two common limitations of the existing semantic service discovery/matchmaking approaches for service discovery in the Digital Ecosystems environment that need to be addressed are as follows:

1. All of the approaches focus only on the Web service field. As introduced in Chapter 1, compared with Web services, Digital Ecosystem services have broader contents. However, none of the existing semantic service discovery/matchmaking approaches considers this field.

2. None of the approaches is able to discover non-standard service information in the Web. On the one hand, Web services can be described by WSDL, which has a unified standard for service description; on the other hand, Digital Ecosystem services can be described by any languages (English, French, etc.) and in any formats (plain texts, tables, etc.), which cannot be directly observed without picking and standardizing from Web documents. Current semantic discovery approaches are designed for WSDL-described Web services other than natural languages-described Digital Ecosystem services.

As a consequence, current semantic service discovery/matchmaking approaches cannot fulfil the technical requirements of service discovery in Digital Ecosystems.

In Chapter 3, we will formally define the problems associated with service discovery in Digital Ecosystems in the existing literature. In Chapter 4, we present the solution overview for the abovementioned problems within the existing literature.

2.6.2 Service Retrieval in Digital Ecosystems

As introduced in Chapter 1, service requesters in the Digital Ecosystems environment are heterogeneous. In Digital Ecosystems, service requesters could be human, organizations or agents. If service requesters are either human or organizations, from the perspective of service requests, they can be divided into two basic groups – service requesters without domain knowledge regarding their service requests and service requesters with domain knowledge regarding their service requests. This feature leads to two requirements for service retrieval in Digital Ecosystems, which are: 1) service retrieval needs to provide the first group of service requesters with specific Digital Ecosystem service domain knowledge in order to help them to denote their service requests and retrieve requested service entities; and 2) service retrieval must enable the second group of service requesters to quickly and precisely locate service entities according to their service requests by means of domain knowledge-based service annotation.

As can be inferred from the discussion in Sections 2.2, 2.3 and 2.4, a significant majority of the existing approaches in the literature cannot fulfil the above two requirements for service retrieval in the Digital Ecosystems environment, which needs to be addressed as follows:

It is apparent from the discussion in Section 2.2, that most of the existing literature in the field of semantic crawlers focuses on non-service information crawling. The only exception is Toch et al.'s approach [191]. However, their approach focuses only on the Web service field and never takes into account the particular nature of Digital Ecosystem services. Thus, it can be concluded that the existing semantic crawling methodologies cannot fulfil the requirements of service retrieval in the Digital Ecosystems environment.

The discussion in Section 2.3 indicates that, owing to the lack of specific Digital Ecosystem service domain knowledge, the semantic service discovery/matchmaking approaches in the existing literature cannot help service requesters to denote their service requests in terms of domain knowledge. Moreover, the lack of domain knowledge also makes it impossible to semantically annotate Digital Ecosystem service information. Consequently, the existing semantic service discovery/matchmaking approaches cannot fulfil the requirements of service retrieval in the Digital Ecosystems environment.

The information and discussion presented in Section 2.4 implies that none of the semantic search engines and technologies in existing literature is devoted to service fields, except Linckels et al. [117]'s e-librarian service search approach. Nevertheless, Digital Ecosystem services have contents broader than e-librarian services, and these have not been addressed in this approach. Hence, the existing semantic service retrieval approaches cannot fulfil the requirements of service retrieval in the Digital Ecosystems environment.

In Chapter 3, we will formally define the problems associated with service retrieval in Digital Ecosystems in the existing literature. In Chapter 4, we present the solution overview for the abovementioned problems within the existing literature.

2.6.3 Service Recommendation in Digital Ecosystems

For service requesters who are humans or organizations and do not have sufficient knowledge about their service queries, their service queries could be incorrect or incomplete, which may lead to incorrect or incomplete domain-specific knowledge-based query annotation. Therefore, in Digital Ecosystems, when a service requester retrieves a service, when s/he finds that the service domain knowledge matched to his/her incorrect or incomplete queries cannot represent his/her real service request, we need a methodology to help the service requester to find the specific service domain knowledge (ontological concepts) that can represent his/her real service request. Moreover, this methodology should work in the ontology environment, as Digital Ecosystem service domain knowledge is represented by ontologies [66].

As can be inferred from the discussion in Section 2.3, 2.4 and 2.5, a significant majority of the existing approaches in the literature cannot fulfil the above two requirements for service recommendation in the Digital Ecosystems environment, which needs to be addressed as follows:

As can be inferred from the discussion in Section 2.3 and 2.4, the semantic service discovery approaches and the semantic search approaches in the existing literature are not concerned with providing a solution for incorrect or incomplete queries.

Our discussion in Section 2.5 reveals that the semantic similarity models in the existing literature all focus on the semantic network environment and ignore the semantically-rich contents of ontology concepts when measuring concept similarity.

In Chapter 3, we will formally define the problems associated with service recommendation in Digital Ecosystems in the existing literature. In Chapter 4, we present the solution overview for the abovementioned problems within the existing literature.

2.6.4 QoS Evaluation and Service Ranking in Digital Ecosystems

As depicted in Chapter 1, service entities in the Digital Ecosystems environment have the nature of contextual dependence. Namely, when putting a service entity in different contexts, its content is different. Similarly, the QoS evaluation criteria are different in different contexts. Analogously, as providers of the contextual dependent service entities, service providers may have different service reputation values in different contexts, owing to their different QoS values for each provided service in different contexts. Therefore, we need a methodology to realize the context-dependent QoS evaluation and service ranking in Digital Ecosystems, once many services have been retrieved based on a service request. This methodology has two requirements: 1) the QoS evaluation criterion should be context-dependent; and 2) the service contexts should be consistent with specific service domain knowledge in the Digital Ecosystems environment.

The discussion in Section 2.2, 2.3 and 2.4 indicates that a significant majority of the existing approaches in the literature cannot fulfil the above two requirements for service recommendation in the Digital Ecosystems environment, which needs to be addressed as follows:

As can be inferred from the discussion in Section 2.2, none of the research in the field of semantic crawlers is concerned with qualitatively evaluating and ranking the crawled information.

We can infer from the discussion in Section 2.3, that most of the semantic service discovery approaches in the existing literature do not provide the function of QoS evaluation with the exception of Bianchini et al.'s approach [15], Zhang et al.'s approach [208] and Mokhtar et al.'s approach [140]. However, their QoS evaluation criteria are built on Web service domain knowledge but not Digital Ecosystem service domain knowledge, and therefore cannot be employed to evaluate Digital Ecosystem services. Hence, the existing semantic service discovery approaches cannot fulfil the technical requirements of QoS evaluation and service ranking in Digital Ecosystems.

As can be inferred from the discussion in Section 2.4, none of the semantic search approaches in the existing literature is concerned with QoS evaluation and service ranking in Digital Ecosystems.

In Chapter 3, we will formally define the problems associated with QoS evaluation and service ranking in Digital Ecosystems in the existing literature. In Chapter 4, we present the solution overview for the abovementioned problems within the existing literature.

2.6.5 Service Domain Knowledge Updating in Digital Ecosystems

As explained in Chapter 1, service entity information, service knowledge and service provider information are dynamic in the Digital Ecosystems environment. First of all, service entity information in the Web is dynamic, which is altered by service providers in order to correspond to changes in advertised service activities in the real environment. Secondly, service knowledge in the Web is dynamic for two reasons: 1) service knowledge needs to change to align with the changes in humans' understandings toward services in the real environment; and 2) service knowledge needs to change so that there is agreement between knowledge creators (normally ontology designers) and knowledge receivers (normally ontology users). In addition, as a result of dynamic service knowledge, QoS evaluation criteria that depend on service knowledge also need to be changed. Third, service provider information in the Web is dynamic, and is altered by service providers in order to indicate changes of service providers' status in the real environment. In order to cater for the dynamic nature of services and service providers in Digital Ecosystems, we need a service domain knowledge (here service domain knowledge represents the combination of service entity information, service knowledge and service provider information) updating methodology: 1) to allow service providers to maintain (add, remove or update) their advertised service entity information; 2) to allow the update of Digital Ecosystem service domain knowledge and context-dependent QoS evaluation criteria; and 3) to allow service providers to update their service profiles.

From the discussion presented in Section 2.2, 2.3 and 2.4, it can be inferred that a significant majority of the existing approaches in the literature cannot fulfil the above three requirements for service domain knowledge updating in the Digital Ecosystems environment, which needs to be addressed as follows:

Information given in Section 2.2 indicates that only the ontology-based focused crawlers (Section 2.2.2) and metadata abstraction focused crawlers (Section 2.2.4) employ ontologies. Nevertheless, none of the crawlers proposes any means for updating the employed ontologies. Thus, the existing semantic crawlers cannot fulfil the requirements for service domain knowledge updating in Digital Ecosystems.

As can be inferred from the discussion in Section 2.3, most of the semantic service discovery approaches in the existing literature use ontologies, with the exception of Liu et al.'s approach [121] which proposed a semantic link network for Web service clustering and discovery. However, none of the approaches is concerned with the issue of ontology evolution. Therefore, the existing semantic service discovery/matchmaking approaches cannot fulfil the requirements for service domain knowledge updating in Digital Ecosystems.

The discussion in Section 2.4 demonstrates that many approaches employ domain ontologies or facets for semantic search (with the exception of Kandogan et al. [96], Zhu and Hu [210], Zhang et al. [207], Ning et al. [145], Jin et al. [94], Lee and Tsai [109], Kwon et al. [105], Guha et al. [67, 68], Bhagwat and Polyzotis [13], Anyanwu et al. [3], Rocha et al. [159], Maedche et al. [133], Ding et al. [47], Hogan et al. [80, 86], Völkel et al. [198], Hoi and Lyu [87], Wang et al. [200], and Schreiber et al. [166]'s approaches, SweetWiki [22], Semantic MultimediaWiki [104], and IkeWiki [164]). However, none of these semantic search approaches is concerned with the issue of ontology evolution. Therefore, the existing semantic search approaches cannot fulfil the requirements for service domain knowledge updating in Digital Ecosystems.

In Chapter 3, we will formally define the problems associated with service knowledge updating in Digital Ecosystems in the existing literature. In Chapter 4, we present the solution overview for the abovementioned problems within the existing literature.

2.7 Conclusion

In this chapter we carried out an extensive survey of the existing literature on four application areas of semantic search. We evaluated the existing literature critically and found that no approach in the literature proposes a means for service retrieval in the Digital Ecosystems environment that takes into account the heterogeneous, context-dependent and dynamic nature of Digital Ecosystem services, and the heterogeneous and dynamic nature of service providers and service requesters.

In the next chapter, we define the problem that we intend to address in this thesis.

2.8 References

1. Introduction to UDDI: Important features and functional concepts. Organization for the Advancement of Structured Information Standards (OASIS) (2004), http://www.uddi.org

2. Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.-T., Sheth, A., Verma, K.: Web Service Semantics - WSDL-S. University of Georgia Research Foundation, Inc, Athens, GA, USA (2005), http://www.w3.org/Submission/WSDL-S/

3. Anyanwu, K., Maduko, A., Sheth, A.: SemRank: Ranking complex relationship search results on the Semantic Web. Proceedings of the 14th International World Wide Web Conference (WWW '05). ACM, Chiba, Japan (2005) 117-127

4. Ardö, A.: Focused crawling in the ALVIS semantic search engine. Proceedings of the 2nd Annual European Semantic Web Conference (ESWC 2005). Springer-Verlag, Heraklion, Greece (2005) 19-20

5. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: A nucleus for a Web of open data. Proceedings of the 6th International Semantic Web Conference (ISWC/ASWC 2007). Springer-Verlag, Busan, Korea (2007) 722-735

6. Barfourosh, A.A., Anderson, M.L., Nezhad, H.R.M., Perlis, D.: Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition. Department of Computer Science, University of Maryland, Maryland, USA (2002)

7. Batsakis, S., Petrakis, E.G.M., Milios, E.: Improving the performance of focused Web crawlers. Data & Knowledge Engineering **68** (2009) 1001-1013

8. Batzios, A., Dimou, C., Symeonidis, A.L., Mitkas, P.A.: BioCrawler: An intelligent crawler for the Semantic Web. Expert Systems with Applications **35** (2008) 524-530

9. Bellavista, P., Corradi, A., Montanari, R., Stefanelli, C.: Contextaware middleware for resource management in the wireless Internet. IEEE Transactions on Software Engineering **29** (2003) 1086-1099

10. Berman, F., Fox, G., Hey, A.J.G.: Grid Computing: Making the Global Infrastructure a Reality. Wiley, West Sussex, England (2003)

11. Bernstein, A., Kaufmann, E., Kaiser, C.: Querying the Semantic Web with ginseng: A guided input natural language search engine. Proceedings of the 15th Workshop

on Information Technologies and Systems (WITS 2005), Las Vegas, NV, USA (2005) 1-10

12. Bernstein, A., Kaufmann, E., Kaiser, C., Kiefer, C.: Ginseng: A guided input natural language search engine for querying ontologies. Proceedings of the 2006 Jena User Conference, Bristol, UK (2006) 1-3

13. Bhagwat, D., Polyzotis, N.: Searching a file system using inferred semantic links. Proceedings of the 16th ACM Conference on Hypertext and Hypermedia (HT '05). ACM, Salzburg, USA (2005) 85-87

14. Bianchini, D., Antonellis, V.D., Melchiori, M.: Flexible semantic-based service matchmaking and discovery. World Wide Web **11** (2008) 227-251

15. Bianchini, D., De Antonellis, V., Pernici, B., Plebani, P.: Ontology-based methodology for e-service discovery. Information Systems **31** (2006) 361-380

16. Bonino, D., Corno, F., Pellegrino, P.: Versatile RDF representation for multimedia semantic search. Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007), Vol. 2. IEEE Computer Society, Patras, Greece (2007) 32-38

17. Bonino, D., Bosca, A., Corno, F., Farinetti, L., Pescarmona, F.: H-DOSE: An Holistic Distributed Open Semantic Elaboration platform. Proceedings of the 1st Italian Semantic Web Workshop (SWAP 2004), Ancona, Italy (2004) 1-10

18. Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.: Web Services Architecture. W3C (2004), http://www.w3.org/TR/ws-arch/

19. Bratsas, C., Koutkias, V., Kaimakamis, E., Bamidis, P.D., Pangalos, G., Maglaveras, N.: KnowBaSICS-M: An ontology-based system for semantic management of medical problems and computerised algorithmic solutions. Computer Methods and Programs in Biomedicine **88** (2007) 39-51

20. Broens, T.: Context-aware, ontology based, semantic service discovery. Master's thesis. University of Twente, Ensched, Netherland (2004)

21. Bruijn, J.d., Bussler, C., Domingue, J., Fensel, D., Hepp, M., Keller, U., Kifer, M., Knig-Ries, B., Kopecky, J., Lara, R., Lausen, H., Oren, E., Polleres, A., Roman, D., Scicluna, J., Stollberg, M.: Web Service Modeling Ontology (WSMO). (2005), http://www.w3.org/Submission/WSMO/

22. Buffa, M., Gandon, F., Ereteo, G., Sander, P., Faron, C.: SweetWiki: A semantic wiki. Web Semantics: Science, Services and Agents on the World Wide Web **6** (2008) 84-97

23. Bullot, H., Gupta, S.K., Mohania, M.K.: A Data-mining approach for optimizing performance of an incremental crawler. Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence (WI '03). IEEE Computer Society, Halifax, Canada (2003) 610

24. Burton-Jones, A., Storey, V.C., Sugumaran, V., Purao, S.: A heuristic-based methodology for semantic augmentation of user queries on the Web. Proceedings of the 22nd International Conference on Conceptual Modeling (ER 2003). ACM, Chicago, IL, USA (2003) 476-489

25. Can, A.B., Baykal, N.: MedicoPort: A medical search engine for all. Computer Methods and Programs in Biomedicine **86** (2007) 73-86

26. Celino, I., Valle, E.D., Cerizza, D., Turati, A.: Squiggle: A semantic search engine for indexing and retrieval of multimedia content. Proceedings of the 1st International Workshop on Semantic-enhanced Multimedia Presentation Systems (SEMPS 2006), Athens, Greece (2006) 1-15

27. Celino, I., Valle, E.D., Cerizza, D., Turati, A.: Squiggle: An experience in model-driven development of real-world semantic search engines. Proceedings of the 7th International Conference on Web Engineering (ICWE 2007). Springer-Verlag, Como, Italy (2007) 485–490

28. Cesarano, C., d'Acierno, A., Picariello, A.: An intelligent search agent system for semantic information retrieval on the internet. Proceedings of the Fifth International Workshop on Web Information and Data Management (WIDM '03). ACM, New Orleans, USA (2003) 111-117

29. Chiang, J.-H., Ho, S.-H., Wang, W.-H.: Similar genes discovery system (SGDS): Application for predicting possible pathways by using GO semantic similarity measure. Expert Systems with Applications **35** (2008) 1115-1121

30. Chiang, R.H.L., Chua, C.E.H., Storey, V.C.: A smart Web query method for semantic retrieval of Web data. Data & Knowledge Engineering **38** (2001) 63-84

31. Chiat, L.C., Huang, L., Xie, J.: Matchmaking for Semantic Web Services. Proceedings of the 2004 IEEE International Conference on Services Computing (SCC'04). IEEE, Shanghai (2004) 455-458

32. Cho, J., Garcia-Molina, H.: Parallel crawlers. Proceedings of the Eleventh International World Wide Web Conference (WWW '02). ACM, Honolulu, USA (2002) 124-135

33. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language (WSDL) 1.1. W3C (2001), http://www.w3.org/TR/wsdl

34. Couto, F.M., Silva, M.J., Coutinho, P.M.: Measuring semantic similarity between Gene Ontology terms. Data & Knowledge Engineering **61** (2007) 137-152

35. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL '02). Association for Computational Linguistics, Philadelphia, USA (2002) 1-8

36. Davies, J., Weeks, R.: QuizRDF: Search technology for the Semantic Web. Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS 2004). IEEE Computer Society, Big Island, HI, USA (2004) 40112

37. Davulcu, H., Vadrevu, S., Nagarajan, S.: OntoMiner: Bootstrapping ontologies from overlapping domain specific websites. Proceedings of the Thirteenth International World Wide Web Conference (WWW '04). ACM, New York, USA (2004) 500-501

38. Decker, K., Sycara, K., Williamson, M.: Middle-agents for the Internet. Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI '97). IJCAI, Nagoya, Japan (1997) 578-583

39. Decker, S., Erdmann, M., Fensel, D., Studer, R.: Ontobroker: Ontology based access to distributed and semi-structured Information. In: Meersman, R. (ed.): Database Semantics: Semantic Issues in Multimedia Systems. Kluwer Academic Publisher (1999) 351–369

40. Dichev, C., Dicheva, D.: View-based semantic search and browsing. Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI '06). IEEE Computer Society, Hong Kong (2006) 919-925

41. Dicheva, D., Dichev, C.: TM4L: Creating and browsing educational topic maps. British Journal of Educational Technology **37** (2006) 391-404

42. Ding, D., Yang, J., Li, Q., Wang, L., Liu, W.: Towards a flash search engine based on expressive semantics. Proceedings of the 13th International World Wide Web Conference (WWW '04). ACM, New York, USA (2004) 472-473

43. Ding, L., Finin, T.: Characterizing the Semantic Web on the Web. Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, USA (2006) 242-257

44. Ding, L., Zhou, L., Finin, T., Joshi, A.: How the Semantic Web is being used: An analysis of FOAF documents. Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS '05). IEEE Computer Society, Hawaii, USA (2005) 113c

45. Ding, L., Finin, T., Joshi, A., Peng, Y., Pan, R., Reddivari, P.: Search on the Semantic Web. Computer **38** (2005) 62-69

46. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari1, P.: Finding and ranking knowledge on the Semantic Web Proceedings of the 4th International Semantic Web Conference (ISWC 2005). Springer-Verlag, Garlway, Ireland (2005) 156-170

47. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., Sachs, J.: Swoogle: A search and metadata engine for the Semantic Web. Proccedings of the Thirteenth ACM Conference on Information and Knowledge Management (CIKM '04). ACM, Washington D.C., USA (2004) 652 - 659

48. Dodds, L.: Slug: a Semantic Web crawler. (2006), http://jena.hpl.hp.com/juc2006/proceedings/dodds/paper.pdf

49. Dong, H., Hussain, F.K., Chang, E.: State of the art in metadata abstraction crawlers. Proceedings of the 2008 IEEE International Conference on Industrial Technology (ICIT 2008). IEEE, Chengdu, China (2008) 1-6

50. Dong, H., Hussain, F.K., Chang, E.: A survey in Semantic Web technologies-inspired focused crawlers. Proceedings of the Third International Conference on Digital Information Management (ICDIM 2008). IEEE Computer Society, London, UK (2008) 934-936

51. Dong, H., Hussain, F.K., Chang, E.: A survey in semantic search technologies. Proceedings of the 2008 2nd IEEE International Conference on Digital Ecosystem and Technologies (DEST 2008). IEEE, Phitsanulok, Thailand (2008) 403-408

52. Dong, H., Hussain, F.K., Chang, E.: State of the art in semantic focused crawlers In: Gervasi, O., Taniar, D., Murgante, B., Laganà, A., Mun, Y., Gavrilova, M. (eds.): Computational Science and Its Applications – ICCSA 2009. Springer-Verlag, Seoul, Korea (2009) 910-924

53. Dong, H., Hussain, F.K., Chang, E.: A framework for discovering and classifying ubiquitous services in digital health ecosystems. Journal of Computer and System Sciences **In Press**

54. Duke, A., Glover, T., Davies, J.: Squirrel: An advanced semantic search and browse facility. In: Franconi, E., Kifer, M., May, W. (eds.): Proceedings of the 4th European Semantic Web Conference (ESWC 2007). Springer-Verlag, Innsbruck, Austria (2007) 341–355

55. Ehrig, M., Maedche, A.: Ontology-focused crawling of Web documents. Proceedings of the Eighteenth Annual ACM Symposium on Applied Computing (SAC 2003). ACM, Melbourne, USA (2003) 9-12

56. Fang, W., Cui, Z., Zhao, P.: Ontology-based focused crawling of deep Web sources. Proceedings of the 1st International Conference on Knowledge Science, Engineering and management (KSEM 2007). Springer-Verlag, Melbourne, Australia (2007) 514-519

57. Feng, C.C., Flewelling, D.M.: Assessment of semantic similarity between land use/land cover classification systems. Computers, Environment and Urban Systems **28** (2004) 229-246

58. Fenza, G., Loia, V., Senatore, S.: A hybrid approach to Semantic Web Services matchmaking. International Journal of Approximate Reasoning **48** (2008) 808-828

59. Francesconi, E., Peruginelli, G.: Searching and retrieving legal literature through automated semantic indexing. Proceedings of the Eleventh International Conference on. Artificial Intelligence and Law (ICAIL '07). ACM, Stanford, USA (2007) 131-138

60. Ganesh, S., Jayaraj, M., Kalyan, V., Aghila, G.: Ontology-based Web crawler. Proceedings of the 2004 International Conference on Information Technology: Coding and Computing (ITCC '04). IEEE Computer Society, Las Vegas, USA (2004) 337

61. García, E., Sicilia, M.-Á.: Designing ontology-based interactive information retrieval interfaces. Proceedings of the Workshop on Human Computer Interface for Semantic Web and Web Applications (HCI-SWWA). Springer-Verlag, New York, USA (2003) 152-165

62. Garcia, V.L., Motta, E., Uren, V.: AquaLog: an ontology-driven question answering system to interface the Semantic Web. Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language. Association for Computational Linguistics, New York, USA (2006) 269 - 272

63. Garofalakis, J., Panagis, Y., Sakkopoulos, E., Tsakalidis, A.: Contemporary Web service discovery mechanisms. Journal of Web Engineering **5** (2006) 265-290

64. Giles, C.L., Petinot, Y., Teregowda, P.B., Han, H., Lawrence, S., Rangaswamy, A., Pal, N.: eBizSearch: A niche search engine for e-business. Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '03). ACM, Toronto, Canada (2003) 213-214

65. Grimes, S.: Breakthrough analysis: Two + nine types of semantic search. InformationWeek. United Business Media, Manhasset, NY, USA (2010).

66. Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition **5** (1993) 199-220

67. Guha, R., McCool, R.: TAP: a Semantic Web platform. Computer Networks **42** (2003) 557-577

68. Guha, R., McCool, R., Miller, E.: Semantic search. Proceedings of the Twelfth International World Wide Web Conference (WWW '03). ACM, Budapest, Hungary (2003) 700-709

69. Haarslev, V., R. M¨oller: RACER user's guide and reference manual. (2004), http://www.sts.tu-harburg.de/˜r.f.moeller/racer/racer-manual-1-7-19.pdf

70. Halkidi, M., Nguyen, B., Varlamis, I., Vazirgiannis, M.: THESUS: Organizing Web document collections based on link semantics. The VLDB Journal **12** (2003) 320–332

71. Han, L., Chen, G.: The HWS hybrid Web search. Information and Software Technology **48** (2006) 687-695

72. Han, L., Berry, D.: Semantic-supported and agent-based decentralized grid resource discovery. Future Generation Computer Systems **24** (2008) 806-812

73. Handschuh, S., Staab, S.: Authoring and annotation of webpages in CREAM. Proceedings of the 12th International World Wide Web Conference (WWW '03). ACM, Honolulu, USA (2002) 462-473

74. Handschuh, S., Staab, S.: Annotation of the shallow and the deep Web. In: S Handschuh, S.S. (ed.): Annotation for the Semantic Web. IOS Press, Amsterdam, Netherland (2003) 25-45

75. Handschuh, S., Staab, S.: CREAM: CREAting metadata for the Semantic Web. Computer Networks **42** (2003) 579-598

76. Handschuh, S., Staab, S., Maedche, A.: CREAM — creating relational metadata with a component-based, ontology-driven annotation framework. Proceedings of the 1st International Conference on Knowledge Capture (K-CAP '01). ACM, Victoria, Canada (2001) 76-83

77. Hansmann, U., Merk, L., Nicklous, M.S., Stober, T.: Pervasive computing: The mobile world. Springer-Verlag, New York, USA (2003)

78. Harth, A., Umbrich, J., Decker, S.: MultiCrawler: A pipelined architecture for crawling and indexing Semantic Web data. Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2006) 258-271

79. Harth, A., Hey, Y., Tangmunarunkity, H., Deckeryz, S.: A semantic matchmaker service on the grid. Proceedings of the Thirteenth International World Wide Web Conference (WWW '04). ACM, New York (2004) 326-327

80. Harth, A., Hogan, A., Delbru, R., Umbrich, J., O'Riain, S.: Swse: Answers before links. Proceedings of the Semantic Web Challenge 2007. Springer-Verlag, Busan, Korea (2007) 1-8

81. Heflin, J., Hendler, J.: Searching the Web with SHOE. Proceedings of the 2000 AAAI Workshop. AAAI, Menlo Park, CA, USA (2000) 35–40

82. Hendler, J.: Agents and the Semantic Web. IEEE Intelligent System **16** (2001) 30-37

83. Hildebrand, M., Ossenbruggen, J.v., Hardman, L.: /facet: A browser for heterogeneous Semantic Web repositories. Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2006) 272-285

84. Hirst, G., St-Onge, D.: Lexical chains as representations of context for the detection and correction of malapropisms. In: Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. The MIT Press, Cambridge, USA (1998)

85. Hliaoutakis, A., Varelas, G., Voutsakis, E., Petrakis, E.G.M., Milios, E.E.: Information retrieval by semantic similarity. International Journal on Semantic Web and Information Systems **2** (2006) 55-73

86. Hogan, A., Harth, A., Umbrich, J.u., Decker, S.: Towards a scalable search and query engine for the Web. Proceedings of the 16th International Conference on World Wide Web (WWW '07). ACM, Banff, Alberta, Canada (2007) 1301 - 1302

87. Hoi, C.-H., Lyu, M.R.: Web image learning for searching semantic concepts in image databases. Proceedings of the 13th International World Wide Web Conference (WWW '04). ACM, New York, USA (2004) 406-407

88. Huang, W., Zhang, L., Zhang, J., Zhu, M.: Semantic focused crawling for retrieving ecommerce information. Journal of Software **5** (2009) 436-443

89. Hyvönen, E., Saarela, S., Viljanen, K.: Ontogator — A semantic view-based search engine service for Web applications. Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2003) 847-860

90. Hyvönen, E., Saarela, S., Viljanen, K.: Application of ontology techniques to view-based semantic search and browsing. In: Davies, J. (ed.): Proceedings of the 1st European Semantic Web Symposium (ESWS 2004). Springer-Verlag, Heraklion, Greece (2004) 92-106

91. Hyvönen, E., Eetu Makel¨a, Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.: MuseumFinland - Finnish museums on the Semantic Web. Web Semantics: Science, Services and Agents on the World Wide Web **3** (2005) 224-241

92. Jansen, B.J., Mullen, T., Spink, A., Pedersen, J.: Automated gathering of Web information: an in-depth examination of agents interacting with search engines. ACM Transactions on Internet Technology **6** (2006) 442-464

93. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. Proceedings of the International Conference on Research in Computational Linguistics (ROCLING X), Taiwan (1997) 19-33

94. Jin, H., Ning, X., Chen, H.: Efficient search for peer-to-peer information retrieval using semantic small world. Proceedings of the 15th International World Wide Web Conference (WWW '06). ACM, Edinburgh, UK (2006) 1003-1004

95. Kaiser, C.: Ginseng - A Natural Language User Interface for Semantic Web Search. Diploma Thesis. University of Zurich, Gams, Switzerland (2004)

96. Kandogan, E., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S., Zhu, H.: Avatar semantic search: A database approach to information retrieval. Proceedings of the 25th ACM SIGMOD International Conference on Management of Data (SIGMOD '06). ACM, Chicago, USA (2006) 790-792

97. Kawamura, T., Blasio, J.-A.D., Hasegawa, T., Paolucci, M., Sycara, K.: Preliminary report of public experiment of semantic service matchmaker with UDDI business registry Proceedings of the International Conference on Service Oriented Computing (ICSOC 2003). Springer-Verlag, Trento, Italy (2003) 208-224

98. Keller, U., Lara, R., Lausen, H., Polleres, A., Predoiu, L., Toma, I.: Semantic Web Service discovery. (2005), http://www.wsmo.org/TR/d10/v0.2/20051003/

99. Khan, L., McLeod, D., Hovy, E.H.: Retrieval effectiveness of an ontology-based model for information selection. The International Journal on Very Large Data Bases **13** (2004) 71-85

100. Klusch, M., Fries, B., Sycara, K.: Automated Semantic Web Service discovery with OWLS-MX. Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006). ACM, Hakodate, Japan (2006) 915-922

101. Klusch, M., Fries, B., Sycara, K.: OWLS-MX: A hybrid Semantic Web Service matchmaker for OWL-S services. Web Semantics: Science, Services and Agents on the World Wide Web **7** (2009) 121-133

102. Koschmann, T.: Paradigm shift and instructional technology. In: (Ed.), T.K. (ed.): CSCL: Theory and practice of an emerging paradigm. Lawrence Erlbaum, Mahwah, NJ, USA (1996) 1-23

103. Kozanidis, L.: An ontology-based focused crawler. Proceedings of the 13th International Conference on Natural Language and Information Systems: Applications of Natural Language to Information Systems (NLDB 2008). Springer-Verlag, London, UK (2008) 376-379

104. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: Cruz, I. (ed.): Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2006) 935-942

105. Kwon, I.-H., Kim, C.O., Kim, K.P., Kwak, C.: Recommendation of e-commerce sites by matching category-based buyer query and product e-catalogs. Computers in Industry **59** (2008) 380-394

106. Lausen, H., Polleres, A., Roman, D.: Web Service Modeling Ontology (WSMO). DERI, Innsbruck (2005), http://www.w3.org/Submission/WSMO/

107. Leacock, C., Chodorow, M.: Combining local context and WordNet similarity for word sense identification. WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998) 265-283

108. Lee, J., Kim, M., Lee, Y.: Information retrieval based on conceptual distance in IS-A hierarchies. Journal of Documentation **49** (1993) 188-207

109. Lee, W.-P., Tsai, T.-C.: An interactive agent-based system for concept-based Web search. Expert Systems with Applications **24** (2003) 365-373

110. Li, L., Horrocks, I.: A software framework for matchmaking based on Semantic Web technology. Proceedings of the Twelfth International World Wide Web Conference (WWW '03). ACM, Budapest, Hungary (2003) 331-339

111. Li, L., Yang, Y., Wu, B.: Ontology-based matchmaking in e-marketplace with Web Services. Proceedings of the Asia-Pacific Web Conference 2005 (APWeb 2005). Springer-Verlag, Shanghai, China (2005) 620-631

112. Li, Y., Bandar, Z.A., McLean, D.: An approach for measuring semantic similarity between words using multiple information sources. IEEE Transactions on Knowledge and Data Engineering **15** (2003) 871-882

113. Liang, B., Tang, J., Li, J.: Association search in Semantic Web: search + inference. Proceedings of the 14th International World Wide Web Conference (WWW '05). ACM, Chiba, Japan (2005) 992-993

114. Liang, B., Tang, J., Li, J., Wang, K.: Using DAML+OIL to enhance search semantic. Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI '04). IEEE Computer Society, Washington D.C., USA (2004) 465 - 468

115. Lin, D.: An Information-theoretic definition of similarity. Proceedings of the 15th International Conference on Machine Learning (ICML '98). Morgan Kaufmann Publishers, Madison, USA (1998) 296-304

116. Lin, D.: Automatic retrieval and clustering of similar words. Proceedings of the 17th International Conference on Computational Linguistics (COLING-ACL '98). Association for Computational Linguistics, Montreal, Quebec, Canada (1998) 768-774

117. Linckels, S., Repp, S., Karam, N., Meinel, C.: The virtual tele-tASK professor: Semantic search in recorded lectures. Proceedings of the 38th ACM Technical Symposium on Computer Science Education (SIGCSE '07). ACM, Covington, Kentucky, USA (2007) 50-54

118. Liu, D.-R., Shen, M., Liao, C.-T.: Designing a composite e-service platform with recommendation function. Computer Standards & Interfaces **25** (2003) 103-117

119. Liu, H., Milios, E., Janssen, J.: Focused crawling by learning HMM from user's topic-specific browsing. Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI '04). IEEE Computer Society, Beijing, China (2004) 732-732

120. Liu, H., Milios, E., Janssen, J.: Probabilistic models for focused Web crawling. Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management (WIDM '04). ACM, Washington D.C., USA (2004) 16-22

121. Liu, J., Zhuge, H.: A semantic-link-based infrastructure for Web service discovery in P2P networks. Proceedings of the 14th International World Wide Web Conference (WWW '05). ACM, Chiba, Japan (2005) 940-941

122. Liu, M., Shen, W., Hao, Q., Yan, J.: A weighted ontology-based semantic similarity algorithm for Web service. Expert Systems with Applications **36** (2009) 12480-12490

123. Liu, W., Chen, Z., Lin, F., Yang, R., Li, M., Zhang, H.: Ubiquitous media agents for managing personal multimedia files. Proceedings of the Ninth ACM International Conference on Multimedia (MM '01). ACM, Ottawa, Canada (2001) 519-521

124. Liu, Y., Bai, K., Mitra, P., Giles, C.L.: Automatic searching of tables in digital libraries. Proceedings of the 16th International Conference on World Wide Web (WWW '07). ACM, Banff, Canada (2007) 1135-1136

125. Liu, Y., Bai, K., Mitra, P., Giles, C.L.: TableSeer: Automatic table metadata extraction and searching in digital libraries. Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital libraries (JCDL '07). ACM, Vancouver, Canada (2007) 91-100

126. Lopez, V., Pasin, M., Motta, E.: AquaLog: An ontology-portable question answering system for the Semantic Web. In: Gómez-Pérez, A., Euzenat, J. (eds.): Proceedings of the 2nd Annual European Semantic Web Conference (ESWC 2005). Springer-Verlag, Heraklion, Greece (2005) 546-562

127. Lopez, V., Uren, V., Motta, E., Pasin, M.: AquaLog: An ontology-driven question answering system for organizational semantic intranets. Web Semantics: Science, Services and Agents on the World Wide Web **5** (2007) 72-105

128. Lord, P., Alper, P., Wroe, C., Goble, C.: Feta: A light-weight architecture for user oriented semantic service discovery. In: G´omez-P´erez, A., Euzenat, J. (eds.): Proceedings of the 2nd European Semantic Web Conference (ESWC 2005). Springer-Verlag, Heraklion, Crete (2005) 17-31

129. Ludwig, S.A., Reyhani, S.M.S.: Introduction of semantic matchmaking to Grid computing. Journal of Parallel and Distributed Computing **65** (2005) 1533-1541

130. Ludwig, S.A., Reyhani, S.M.S.: Semantic approach to service discovery in a Grid environment. Web Semantics: Science, Services and Agents on the World Wide Web **4** (2006) 1-13

131. Luong, H.P., Gauch, S., Wang, Q.: Ontology-based focused crawling. Proceedings of the 2009 International Conference on Information, Process, and Knowledge Management (eKNOW '09). IEEE Computer Society, Cancun, Mexico (2009) 123 - 128

132. Maedche, A., Ehrig, M., Stojanovic, L., Handschuh, S., Volz, R.: Ontology-focused crawling of documents and relational metadata. Proceedings of the Eleventh International World Wide Web Conference (WWW '02). ACM, Hawaii, USA (2002) 1-1

133. Maedche, A., Motik, B., Stojanovic, L., Studer, R., Volz, R.: An infrastructure for searching, reusing and evolving distributed ontologies. Proceedings of the 12th International World Wide Web Conference (WWW '03). ACM, Budapest, Hungary (2003) 439-448

134. Maguitman, A., Menczer, F., Roinestad, H., Vespignani, A.: Algorithmic detection of semantic similarity. Proceedings of the 14th International World Wide Web Conference (WWW '05). ACM, Chiba, Japan (2005) 107-116

135. Mäkelä, E., Hyvönen, E., Saarela1, S.: Ontogator — A semantic view-based search engine service for Web applications. Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2006) 847-860

136. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K.: OWL-S: Semantic mark-up for Web services W3C (2004), http://www.w3.org/Submission/OWL-S/

137. McIlraith, S., Martin, D.: Bringing semantics to Web services. IEEE Intelligent Systems **18** (2003) 90–93

138. McIlraith, S.A., Son, T.C., Zeng, H.: Semantic Web Services. IEEE Intelligent Systems **16** (2003) 46-53

139. Miller, G., Charles, W.: Contextual correlates of semantic similarity. Language and Cognitive Processes **6** (1991) 1-28

140. Mokhtar, S.B., Preuveneers, D., Georgantas, N., Issarny, V., Berbers, Y.: EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support. Journal of Systems and Software **81** (2008) 785-808

141. Navas-Delgado, I., Roldan-Garcia, M.d.M., Aldana-Montes, J.F.: Deep crawling in the Semantic Web: In search of deep knowledge In: al., X.Z.e. (ed.): Proceedings of the 5th International Conference on Web Information Systems Engineering (WISE 2004). Springer-Verlag, Brisbane, Australia (2004) 541-546

142. Navas-Delgado, I., Moreno-Vergara, N., G´omez-Lora, A.C., Rold´an-Garc´ıa, M.M., Ruiz-Mostazo, I., Aldana-Montes, J.F.: Embedding semantic annotations into dynamic Web contents. Proceedings of the 15th International Workshop on Database and Expert Systems Applications (DEXA '05). IEEE Computer Society, Zaragoza, Spain (2004) 221-235

143. Nelson, M.L., a, J.A.S., Campo, I.G.d., Sompel, H.V.d., Liu, X.: Efficient, automatic Web resource harvesting. Proceedings of the 8th Annual ACM International Workshop on Web Information and Data Management (WIDM '06). ACM, Arlington, USA (2006) 43-50

144. Nikravesh, M.: Beyond the Semantic Web: fuzzy logic-based Web intelligence. In: Ma, Z. (ed.): Soft Computing in Ontlogies and Semantic Web. Springer-Verlag, New York (2006) 149-209

145. Ning, X., Jin, H., Wu, H.: RSS: A framework enabling ranked search on the Semantic Web. Information Processing & Management **44** (2008) 893-909

146. Oren, E., Delbru, R., Decker, S.: Extending faceted navigation for RDF data Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2006) 559-572

147. Othman, R.M., Deris, S., Illias, R.M.: A genetic similarity algorithm for searching the Gene Ontology terms and annotating anonymous protein sequences. Journal of Biomedical Informatics **41** (2008) 65-81

148. Panayiotou, C., Samaras, G.: mPERSONA: Personalized portals for the wireless user: an agent approach. Mobile Networks and Applications **9** (2004) 663–677

149. Paolucci, M., Sycara, K.: Autonomous Semantic Web Services. IEEE Internet Computing **7** (2003) 34-41

150. Patel, C., Supekar, K., Lee, Y., Park, E.K.: OntoKhoj: A Semantic Web portal for ontology searching, ranking and classification. Proceedings of the 5th Annual ACM International Workshop on Web Information and Data Management (WIDM '03). ACM New Orleans, USA (2003) 58-61

151. Patel, C., Supekar, K., Lee, Y., Park, E.K.: OntoKhoj: A Semantic Web portal for ontology searching, ranking and classification. Proceedings of the Fifth International Workshop on Web Information and Data Management (WIDM '03). ACM, New Orleans, USA (2003) 58-61

152. Pedersen, T., Pakhomov, S.V.S., Patwardhan, S., Chute, C.G.: Measures of semantic similarity and relatedness in the biomedical domain. Journal of Biomedical Informatics **40** (2006) 288-299

153. Pirro, G.: A semantic similarity metric combining features and intrinsic information content. Data & Knowledge Engineering **68** (2009) 1289-1308

154. Popov, B., Kiryakov, A., Ognyanoff, D., Manov, D., Kirilov, A.: KIM – a semantic platform for information extraction and retrieval. Natural Language Engineering **10** (2004) 375 - 392

155. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. IEEE Transactions on Systems, Man and Cybernetics **19** (1989) 17-30

156. Resnik, P.: Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. Journal of Artificial Intelligence Research **11** (1999) 95-130

157. Richardson, R., Smeaton, A.F.: Using WordNet in a knowledge-based approach to information retrieval. Dublin City University, Dublin, Ireland (1995)

158. Roberson, S., Dicheva, D.: Semi-automatic ontology extraction to create draft topic maps. Proceedings of the 45th Annual ACM Southeast Regional Conference (ACMSE '07). ACM, Winston-Salem, USA (2007) 100-105

159. Rocha, C., Schwabe, D., Aragao, M.P.: A hybrid approach for searching in the Semantic Web. Proceedings of the 13th International World Wide Web Conference (WWW '04). ACM, New York, USA (2004) 374-383

160. Rosenfield, R.: A maximum entropy approach to adaptive statistical modelling. Computer Speech and Language **10** (1996) 187-228

161. Rubenstein, H., Goodenough, J.B.: Contextual correlates of synonymy. Communications of the ACM **8** (1965) 627-633

162. Sahami, M., Heilman, T.: A web-based kernel function for measuring the similarity of short text snippets. Proceedings of the15th International World Wide Web Conference (WWW '06). ACM, Edinburgh, UK (2006) 377-386

163. Satyanarayanan, M.: Pervasive computing: Vision and challenges. IEEE Personal Communications **8** (2001) 10-17

164. Schaffert, S.: IkeWiki: A semantic Wiki for collaborative knowledge management. Proceedings of the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2006). IEEE Computer Society, Manchester, UK (2006) 388-396

165. Schollmeier, R.: A definition of Peer-to-Peer networking for the classification of Peer-to-Peer architectures and applications. Proceedings of the First International Conference on Peer-to-Peer Computing (P2P '01). IEEE Computer Society, Linköpings universitet, Sweden (2001) 0101

166. Schreiber, G., Amin, A., Assem, M.v., Boer, V.d., Hardman, L., Hildebrand, M., Hollink, L., Huang, Z., Kersen, J.v., Niet, M.d., Omelayenko, B., Ossenbruggen, J.v., Siebes, R., Taekema, J., Wielemaker, J., Wielinga, B.: MultimediaN E-Culture demonstrator. Proceedings of the 5th International Semantic Web Conference (ISWC 2006). Springer-Verlag, Athens, GA, USA (2006) 951-958

167. Seco, N.: Computational models of similarity in lexical ontologies. Master's thesis. University College Dublin, Dublin, Ireland (2005)

168. Sevilla, J.L., Segura, V.c., Podhorski, A., Guruceaga, E., Jose´ M. Mato, Martı´nez-Cruz, L.A., Corrales, F.J., Rubio, A.: Correlation between gene expression and GO semantic similarity. IEEE/ACM Transaction on Computational Biology and Bioinformatics **2** (2005) 330-338

169. Sheth, A., Bertram, C., Avant, D., Hammond, B., Kochut, K., Warke, Y.: Managing semantic content for the Web. IEEE Internet Computing **6** (2002) 80-87

170. Shimazu, K., Arisawa, T., Saito, I.: Interdisciplinary contents management using 5W1H interface for metadata. Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI '06). IEEE Computer Society, Hong Kong (2006) 909-912

171. Shin, M.K., Huh, S.Y., Lee, W.: Providing ranked cooperative query answers using the metricized knowledge abstraction hierarchy. Expert Systems with Applications **32** (2007) 469-484

172. Smith, J.A., Nelson, M.L.: Generating best-effort preservation metadata for Web resources at time of dissemination. Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries (JCDL '07). ACM, Vancouver, Canada (2007) 51-52

173. Song, W., Li, C.H., Park, S.C.: Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures. Expert Systems with Applications **36** (2009) 9095-9104

174. Sowa, J.F.: Semantic Networks. In: Shapiro, S.C. (ed.): Encyclopaedia of Artificial Intelligence. Wiley (1992)

175. Sriharee, N., Senivongse, T.: Matchmaking and ranking of Semantic Web Services using integrated service profile. International Journal of Metadata, Semantics and Ontologies **1** (2006) 110-118

176. Srihari, R.K., Zhang, Z.F., Rao, A.B.: Intelligent indexing and semantic retrieval of multimodal documents. Information Retrieval **2** (2000) 245-275

177. Steichen, O., Bozec, C.D., Thieu, M., Zapletal, E., Jaulent, M.C.: Computation of semantic similarity within an ontology of breast pathology to assist inter-observer consensus. Computers in Biology and Medicine **36** (2006) 768–788

178. Stevens, R., Tipney, H., Wroe, C., Oinn, T., Senger, M., Lord, P., Goble, C., A. Brass, Tassabehji, M.: Exploring Williams Beuren Syndrome using myGrid. Bioinformatics **20** (2004) 303–310

179. Stoilova, L., Holloway, T., Markines, B., Maguitman, A.G., Menczer, F.: GiveALink: Mining a semantic network of bookmarks for Web search and recommendation. Proceedings of the Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD '05). ACM, Chicago, USA (2005) 66-73

180. Stojanovic, L., Stojanovic, N., Volz, R.: Migrating data-intensive websites into the Semantic Web. Proceedings of the 2002 ACM symposium on Applied computing (SAC 2002). ACM, Madrid, Spain (2002) 1100-1107

181. Stojanovic, N.: On analysing query ambiguity for query refinement: the librarian agent approach. Proceedings of the 22nd International Conference on Conceptual Modeling (ER 2003). ACM, Chicago, IL, USA (2003) 490-505

182. Stojanovic, N., Maedche, A., Staab, S., Studer, R., Sure, Y.: SEAL — A framework for developing SEmantic PortALs. Proceedings of the 1st International Conference on Knowledge Capture (K-CAP '01). ACM, Victoria, Canada (2001) 155-162

183. Stroulia, E., Wang, Y.Q.: Structural and semantic matching for assessing web-service similarity. International Journal of Cooperative Information Systems **14** (2005) 407-437

184. Su, C., Gao, Y., Yang, J., Luo, B.: An efficient adaptive focused crawler based on ontology learning. Proceedings of the Fifth International Conference on Hybrid Intelligent Systems (HIS' 05). IEEE Computer Society, Rio de Janeiro, Brazil (2005) 1-1

185. Sussna, M.: Word sense disambiguation for free-text indexing using a massive semantic network. Proceedings of the Second International Conference on Information and Knowledge Management (CIKM '93). ACM, Washington D.C., USA (1993) 67-74

186. Sycara, K., Paolucci, M., Ankolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of Semantic Web Services. Web Semantics: Science, Services and Agents on the World Wide Web **1** (2003) 27-46

187. Talantikite, H.N., Aissani, D., Boudjlida, N.: Semantic annotations for Web services discovery and composition. Computer Standards & Interfaces **31** (2009) 1108–1117

188. Tane, J., Schmitz, C., Stumme, G.: Semantic resource management for the Web: an elearning application. Proceedings of the Thirteenth International World Wide Web Conference (WWW '04). ACM, New York, USA (2004) 1-10

189. Tang, Y., Zheng, J.: Linguistic modelling based on semantic similarity relation among linguistic labels. Fuzzy Sets and Systems **157** (2006) 1662-1673

190. Tangmunarunkit, H., Decker, S., Kesselman, C.: Ontology-based resource matching in the Grid - the Grid meets the Semantic Web Proceedings of the 2nd International Semantic Web Conference (ISWC 2003). Springer-Verlag, Berlin, Germany (2003)

191. Toch, E., Gal, A., Reinhartz-Berger, I., Dori, D.: A semantic approach to approximate service retrieval. ACM Transactions on Internet Technology **8** (2007) 2-31

192. Toninelli, A., Corradi, A., Montanari, R.: Semantic-based discovery to support mobile context-aware service access. Computer Communications **31** (2008) 935-949

193. Tran, D.T., Bloehdorn, S., Cimiano, P., Haase, P.: Expressive resource descriptions for ontology-based information retrieval. Proceedings of the 1st International Conference on the Theory of Information Retrieval (ICTIR '07). Alma Mater, Budapest, Hungary (2007) 55-68

194. Tversky, A.: Features of similarity. Psychological Review **84** (1977) 327–352

195. Vazquez, J.I., Lo´pez-de-Ipin˜a, D.: mRDP: An HTTP-based lightweight semantic discovery protocol. Computer Networks **51** (2007) 4529-4542

196. Vega-Gorgojo, G., Bote-Lorenzo, M.L., G´omez-S´anchez, E., Dimitriadis, Y.A., Asensio-P´erez, J.I.: A semantic approach to discovering learning services in grid-based collaborative systems. Future Generation Computer Systems **22** (2006) 709-719

197. Verma, K., Sivashanmugam, K., Sheth, A., Patil, A., Oundhakar, S., Miller, J.: METEOR-SWSDI: A scalable P2P infrastructure of registries for semantic publication and discovery of Web services. Information Technology and Management **6** (2005) 17-39

198. Völkel, M., Krötzsch, M., Vrandecic, D., Haller, H., Studer, R.: Semantic Wikipedia. Proceedings of the 15th International World Wide Web Conference (WWW '06). ACM, Edinburgh, Scotland (2006)

199. Wang, H.L., Wu, S.H., Wang, I.C., Sung, C.L., Hsu, W.L., Shih, W.K.: Semantic search on Internet tabular information extraction for answering queries. Proceedings of the Seventh International Conference on Information and Knowledge Management (CIKM '00). ACM, McLean, VA, USA (2000) 243-249

200. Wang, S., Jing, F., He, J., Du, Q., Zhang, L.: IGroup: Presenting Web image search results in semantic clusters. Proceedings of the Computer/Human Interaction 2007 Conference (CHI '07). ACM, San Jose, USA (2007) 587-596

201. Wu, Z., Palmer, M.: Verb semantics and lexical selection. Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics (ACL 1994). The Association for Computational Linguistics, Las Cruces, USA (1994) 133-138

202. Yang, S.-Y.: An ontological website models-supported search agent for Web services. Expert Systems with Applications **35** (2008) 2056-2073

203. Yang, S.-Y.: OntoPortal: An ontology-supported portal architecture with linguistically enhanced and focused crawler technologies. Expert Systems with Applications **36** (2009) 10148-10157

204. Yang, S.-Y., Hsu, C.-L.: Ontology-supported focused-crawler for specified scholar's webpages. Proceedings of the Eighth International Conference on Intelligent

Systems Design and Applications (ISDA '08). IEEE Computer Society, Taiwan (2008) 409-414

205. Yang, S.: Developing of an ontological focused-crawler for ubiquitous services. Proceedings of the 22nd International Conference on Advanced Information Networking and Applications (AINA 2008). IEEE Computer Society, Washington D.C., USA (2008) 1486-1491

206. Yuvarani, M., Iyengar, N.C.S.N., Kannan, A.: LSCrawler: a framework for an enhanced focused Web crawler based on link semantics. Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI '06). IEEE Computer Society, Hong Kong (2006) 794-800

207. Zhang, L., Yu, Y., Zhou, J., Lin, C., Yang, Y.: An enhanced model for searching in semantic portals. Proceedings of the 14th International World Wide Web Conference (WWW '05). ACM, Chiba, Japan (2005) 453-462

208. Zhang, Y., Qu, Y., Huang, H., Yang, D., Zhang, H.: An ontology and peer-to-peer based data and service unified discovery system. Expert Systems with Applications **36** (2009) 5436-5444

209. Zheng, H.-T., Kang, B.-Y., Kim, H.-G.: An ontology-based approach to learnable focused crawling. Information Sciences **178** (2008) 4512-4522

210. Zhu, Y., Hu, Y.: Efficient semantic search on DHT overlays. Journal of Parallel and Distributed Computing **67** (2007) 604-616

211. Zhuang, Z., Wagle, R., Giles, C.L.: What's there and what's not? : Focused crawling for missing documents in digital libraries. Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '05). ACM, Denver, USA (2005) 301-310

212. Zuber, V.S., Faltings, B.: OSS: A semantic similarity function based on hierarchical ontologies. Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007). IJCAI, Hyderabad, India (2007) 551-556

Chapter 3 - Problem Definition

3.1 Introduction

The first chapter highlighted the importance of service retrieval in the Digital Ecosystems environment. In the previous chapter, we presented a comprehensive overview of the existing literature in the field of semantic search. It was noted that significant advances have been made by various researchers in the area of semantic crawlers, semantic service discovery, semantic search engines and technologies, and semantic similarity. However, in Chapter 2, it was noted that none of the existing proposals offers a comprehensive methodology for service retrieval in the Digital Ecosystems environment, that takes into account the heterogeneous, context dependent and dynamic nature of services, and the heterogeneous and dynamic nature service providers and service requesters in the Digital Ecosystems environment. Additionally, in the previous chapter, we identified five main shortcomings within the existing literature that need to be addressed in order to devise a complete methodology for service retrieval in the Digital Ecosystems environment. In this chapter, we formally define and present the problem that we intend to address in this thesis in Section 3.3. In Section 3.2, we propose a set of definitions of those terminologies that will be used when defining the problem in Section 3.3. We break the problem down into six cohesive research issues and define each of these research issues formally in Section 3.4. In Section 3.5, we then outline the solution proposal and choice of research method to address the identified research issues. Finally, Section 3.6 concludes the chapter.

3.2 Key Concepts

In this section, we present a formal definition of terminologies and concepts which will be used to introduce, elucidate and formally define the problem addressed in this thesis.

3.2.1 Digital Ecosystems/Digital Ecosystems environment

A Digital Ecosystem is defined as "*an open, loosely coupled, domain clustered, demand-driven, self-organizing and agent-based environment, in which each species is proactive and responsive for its own benefit and profit*" [5], where

"*Open*" refers to a transparent environment where all interactions are visible [5].

"*Loosely coupled*" refers to a freely bound and open relationship between species in Digital Ecosystems [5].

"*Domain clustered*" refers to an environment which consists of the field where some species have common interest [5].

"*Demand-driven*" means that species actively join in a community based on their own interests [5].

"*Self-organizing*" means that species are capable of acting autonomously, making decisions and carrying out tasks in Digital Ecosystems [5].

"*Agent-based*" refers to an environment which contains human individuals, information technologies and tools that facilitate interaction and knowledge sharing along with the resources that help maintain synergy among human beings and organizations [5].

3.2.2 Species in Digital Ecosystems

We define *species* as the participants in Digital Ecosystems who interact with other species, share commonly agreed vocabularies such as domain ontologies, come from certain domains, follow the rules of Digital Ecosystems, are proactive and responsive for their own profit or benefit, and carry out tasks that relate to their own profit or benefit. There are three subclasses of species – biological species, economic species and digital species that inherit all properties from the species concept. The major differences between them and their living counterparts in the virtual world are as follows [8, 9]:

- The instance of biological species is human. They live in the environment of a natural ecosystem and interact with other humans. As members of the society, they must follow the common rules in the human society, such as laws and regulations.
- An organization is one example of an economic species. Organizations exist in the business environment and their communication scope is limited to themselves. An organization is the aggregation of individuals. Since organizations belong to certain industries, they must follow some domain-specific rules, such as industrial agreements.
- Digital species mainly include hardware, software and agents. Digital species are connected by networks, and they communicate with each other via networks; thus, their living environment is a digital network. The communication rules in the network world are protocols and so on.

3.2.3 Services in the Digital Ecosystems Environment

We define the services (or service entities) in the Digital Ecosystems environment as the finite number of non-intersecting and possible cohesive set of activities performed by a service provider to achieve desired results for a service requester.

According to the types of species, from the perspective of served objects, we classify the services in the Digital Ecosystems environment into three primary types as follows [11]:

- personal services such as eating, drinking and entertaining, and the served objects are biological species;
- economic/organizational services such as meeting, planning and organizational consulting, and the served objects are economic species; and
- digital services such as hardware services, software services and Web services, and the served objects are digital species.

3.2.4 Generic Services in the Digital Ecosystems Environment

We define generic services in the Digital Ecosystems environment as the collection of personal servcies and economic/organizational services, which are provided by humans and are used to distinguish them from digital services.

3.2.5 Service Requester and Service Provider in Digital Ecosystems

Species can play dual roles simultaneously in Digital Ecosystems [5]. We define the roles of species in Digital Ecosystems as a service requester (or called a client in [5]) that requests a service and a service provider (or called a server in [5]) that provides a service [10].

3.2.6 Digital Ecosystem Service Information

Digital Ecosystem service information refers to Digital Ecosystem service advertisement information and Digital Ecosystem service provider profile information over the Internet.

3.2.7 Service Description Entity (SDE)

We define an SDE as the entity used to describe a specific service entity provided by a service provider in Digital Ecosystems [7].

3.2.8 Ontology

An ontology is defined as “*a formal, explicit specification of a shared conceptualization*”, which comprises objects (concepts) and relationships among objects (concepts) [15].

3.2.9 Ontology Mark-up Language

Ontology mark-up language refers to the formal languages that use schema to encode ontologies, such as DAML+OIL, RDF, RDFS and OWL.

3.2.10 Metadata

Metadata is structured data about data. In the context of Semantic Web, metadata is machine understandable information for the Web, which aptly describes the semantics and structures of Web information [26]. Metadata can be encoded by ontology mark-up languages.

3.2.11 SDE Metadata

We define an SDE metadata as the metadata used to describe a specific service provided by a service provider in Digital Ecosystems [7].

3.2.12 Service Factory

Service factory is a group of functional components within the Digital Ecosystems environment, which allows a service provider to create and test a service entity [6, 18].

3.2.13 Service Discovery in the Digital Ecosystems Environment

We define service discovery in the Digital Ecosystems environment as the process of discovering a piece of service information in the Digital Ecosystems environment.

3.2.14 Service Annotation in the Digital Ecosystems Environment

We define service annotation in the Digital Ecosystems environment as the process by which a piece of service information is described, summarized and stored into one or more SDE metadata in terms of ontology mark-up languages.

3.2.15 Service Classification in the Digital Ecosystems Environment

We define service classification as the process of associating one annotated SDE metadata to one or more predefined ontology concepts.

3.2.16 Focused Crawlers

Focused (topical) crawlers are a group of distributed crawlers that specialize in certain specific topics [2].

3.2.17 Semantic Crawlers

We define semantic crawlers as a subset of focused crawlers enhanced by various Semantic Web technologies [12].

3.2.18 Service Search/Retrieval in the Digital Ecosystems environment

We define service search/retrieval in the Digital Ecosystems environment as the process by which a service requester finds the service information, including service advertisements and service provider profiles published by a service provider in the Digital Ecosystems environment based on a service request of the service requester.

3.2.19 Semantic Search/Retrieval

Semantic search is defined as the process of assisting a user to denote an object about which the user is trying to gather/research information, given that the user has little knowledge about the information that s/he is trying to find [16] .

3.2.20 Semantic Similarity

Semantic similarity is a concept whereby a set of documents or terms within term lists are assigned a metric based on the likeness (similarity) of their meanings/semantic content [1].

3.2.21 Quality of Service (QoS)

The term "*QoS*" originated from the teletraffic engineering field, which is defined in the ITU standard X. 902 as "*a set of quality requirements on the collective behaviour of one or more objects*" in the telephony field [21]. In the network field, QoS is concerned with the level of quality of services.

3.2.22 QoS in the Digital Ecosystems Environment

We define QoS in the Digital Ecosystems environment as service requesters' perceptions of the QoS provided by service providers in the Digital Ecosystems environment.

3.2.23 QoS Evaluation Criterion

We define a QoS evaluation criterion as a decisive factor in the mutually agreed service performance between the service provider and service requester for quality assessment purposes [11].

3.2.24 QoS-based Service Ranking in the Digital Ecosystems Environment

We define QoS-based service ranking in the Digital Ecosystems environment as the process of ranking all SDE metadata related to a service ontology concept based on the values of the SDE metadata against the domain-specific QoS evaluation criteria assigned to the service ontology concept.

3.2.25 Human-Centered Computing (HCC)

HCC can be defined as "*the development, evaluation, and dissemination of technology that is intended to amplify and extend the human capabilities to: 1) perceive, understand, reason, decide, and collaborate; 2) conduct cognitive work; and 3) achieve, maintain, and exercise expertise*" [20].

3.2.26 HCC in the Digital Ecosystems Environment

In the Digital Ecosystems environment, HCC concerns the perceptions of service requesters during the process of service retrieval, QoS evaluation and service ranking, subjective perceptions of service providers in the process of SDE metadata classification, and perceptions of both in the process of service domain knowledge evolution [13]. The applications of the HCC in the Digital Ecosystems environment primarily appear in four aspects as follows:

- human-centered semantic service retrieval, which involves service requesters undertaking the process of semantic service retrieval, and enhances service search performance by interacting between service search engines and service requesters.

- human-centered QoS evaluation and service ranking, which considers the perceptions of service requesters toward a service when estimating the quality of the service.

- human-centered ontology updating, which considers both service providers and service requesters' perceptions towards service domain knowledge, when evolving the ontologies being used to represent the service domain knowledge.
- human-centered SDE metadata classification, which considers the requirements of service providers when linking their published SDE metadata to ontology concepts in order to achieve precise and service-providers-satisfied classification results.

3.2.27 Ontology Evolution

Ontology evolution is defined as a process of "*timely adaptation of an ontology to the arisen changes and the consistent management of these changes*" [17].

3.3 Problem Overview and Problem Definition

As we discussed in Chapter 2, semantic search refers to the use of semantic technologies for information retrieval. We carried out an in-depth survey of the state-of-the-art approaches in the four areas of semantic search – semantic crawlers, semantic service discovery, semantic search engines and technologies, and semantic similarity models.

As was discussed in Chapter 2, much research attention has been given to the field of semantic crawlers for harvesting, creating and annotating Semantic Web information. However, as was pointed out in Chapter 2, there are no proposed means or frameworks for discovering, annotating and classifying service information from the Internet, which take into account the heterogeneous and context-dependent nature of services and service providers in Digital Ecosystems. As discussed in Chapter 1, the heterogeneous Digital Ecosystem service information is mingled with other Web information (e.g. product information) on the Internet. Therefore, we require a methodology to discover the embedded Digital Ecosystem service information from the Web. Moreover, due to the nature of contextual dependence, we need a methodology to define the contexts of the Digital Ecosystem service information. The existing literature in semantic crawlers mostly focuses on non-service domains and ignores the exploration of generic service information from the Web, which cannot provide such a methodology. Furthermore, most of the crawlers are in the conceptual phase and lack sufficient experiments to validate the approaches.

As was discussed in Chapter 2, several researchers have made significant advances and impacts on the semantic service discovery in multiple environments, including centralized environments, P2P environments, grid computing environments and ubiquitous computing environments, aiming to describe and explain the capabilities of a Web service so as to match with the functionalities specified in a service request. However, as was mentioned and discussed in the previous chapter, none of the researchers proposes any means for service discovery in the Digital Ecosystems environment that takes into account the heterogeneous, context-dependent, and dynamic nature of services and the heterogeneous nature of service providers in Digital Ecosystems. As discussed in Chapter 1, Digital Ecosystem services cover almost all available service entities in the human society, and a large proportion of Digital Ecosystem services are generic services. In the Web environment, there is vast heterogeneous and context-dependent generic service information available without sufficient technical supports. However, the current semantic service discovery approaches ignore this issue. Furthermore, the existing literature in semantic service discovery focuses on semantically annotating WSDL-encoded service profiles. Owing to the fact that Digital Ecosystem service information is mostly embedded in the Web information and is described by natural languages, the literature does not provide any means for exploiting this service information. Next, because most semantic service discovery approaches use keyword-based approaches to match service requests with service ontology concepts, the performance of the semantic service discovery approaches relies on the quality of queries, and more particularly on service requesters' precise descriptions of desired services. However, there are no proposals for assisting service requesters without relevant knowledge about their queries in order to amend incorrect or incomplete user queries. Subsequently, the semantic service discovery approaches do not propose any means for updating employed service domain ontologies, in order to enable them to adapt to the dynamic Digital Ecosystem service knowledge and to achieve agreements between ontology designers and ontology users.

As was discussed in Chapter 2, some research has been done in the area of semantic search engines and technologies, in order to disambiguate the search process and enhance search quality. However, none of the semantic search engines and technologies focuses on generic service retrieval in the Digital Ecosystems environment. Furthermore, as most of the approaches employing ontologies or facts directly match the keywords between queries and ontological concepts or facets, the performance of these approaches relies on the quality of ontologies. Nevertheless, these search approaches are not concerned with the issue of incomplete and incorrect queries, which could result in an incomplete or inaccurate query. Finally, although the performance of the approaches employing ontologies or facets relies on the quality of employed ontologies or facets, none of them proposes any means for updating the ontologies or facets to make them consistent with the dynamic domain knowledge and to establish agreements between ontology/facet designers and ontology/facet users.

As was discussed in Chapter 2, many semantic similarity models have been developed so far. Nevertheless, most of the semantic similarity models are designed for the semantic network environment. A semantic network is a graph comprised of nodes and arcs [25]. WordNet is a typical example of the semantic network, which is a network composed of words and relationships of words. One feature of semantic networks is that the content of the nodes is simple and the nodes normally consist of single words or phrases. Owing to this feature, most of the semantic similarity models are more concerned with the relative position between nodes than with the content of nodes. Ontology consists of concepts and relationships between concepts [15], in which concepts can be defined with semantic-rich content. For example, in RDF and OWL, concepts can be defined by datatype properties, object properties, restrictions etc. Hence, although the graphs of ontologies and semantic networks look similar, the concepts in the ontology environment have more semantic content than do the nodes in the semantic networks. Because of the importance of the concepts in the ontology environments, the content of the concepts cannot be ignored when measuring similarity between concepts. Another issue regarding the semantic similarity models is that most of them are designed for definitional semantic networks. Definitional semantic networks are one subtype of semantic networks, in which the only type of relationships (arcs) that link between nodes is *is-a*, in which the concept is the superclass of another concept. Thus, many semantic similarity models treat weights of arcs as equal. In ontologies, the relationships between concepts are more complicated and can be self-defined with arbitrary restrictions. It is obvious that the weights of relationships between concepts cannot be regarded as equal when measuring semantic similarity between concepts. Whilst the hybrid models take into account the factor of link types before measuring concept similarity, they do not provide any means for weighting the factors and thus it is difficult to realize in the real environment. Hence, the existing semantic similarity models will be challenged by the semantic-rich and multi-relational ontology environment.

Finally, to the best of our knowledge, there is no proposed methodology in the literature that can be applied to QoS-based service ranking in the Digital Ecosystems environment that takes into account the context-dependent nature of services in Digital Ecosystems. This is a key component of the complete methodology for service retrieval in the Digital Ecosystems environment.

This thesis presents a complete methodology for service retrieval in the Digital Ecosystems environment and involves five facets, namely: service discovery, service retrieval, service recommendation, QoS-based service ranking, and service knowledge updating.

The above description leads to the proposal of a complete methodology for service retrieval in the Digital Ecosystems environment that takes into account the abovementioned facets. Based on the above overview and description of the problem, we formally define the problem that we intend to address in this thesis as follows:

How can a service requester precisely retrieve a high quality service advertised by a reliable and trustworthy service provider, that takes into account the heterogeneous, context dependent and dynamic nature of services, and the heterogeneous and dynamic nature service providers and service requesters in the Digital Ecosystems environment?

The next section describes the research issues that need to be addressed in order to solve the above problem.

3.4 Research Issues

In this section, we present and discuss in detail the research issues that need to be addressed in order to solve the aforesaid problem of service retrieval in the Digital Ecosystems environment. The research issues that need to be addressed are as follows:

1. Propose a methodology for service information discovery, annotation and classification in the Digital Ecosystems environment that takes into account the heterogeneous and context-dependent nature of Digital Ecosystem services and the heterogeneous nature of service providers. We need a methodology by which the ubiquitous service information in the Web can be automatically discovered, annotated with ontology mark-up languages, and classified based on specific service domain knowledge.

2. Propose a methodology for service retrieval in the Digital Ecosystems environment that takes into account the heterogeneous nature of Digital Ecosystem service requesters. We need a methodology by which service requesters can precisely search service advertisements and service providers in terms of given service requests.

3. Propose a methodology for service concept recommendation that takes into account the heterogeneous nature of Digital Ecosystem service requesters. We need a methodology whereby service requesters can be given recommendations regarding related concepts if the retrieved service concepts cannot match service requesters' query intentions.

4. Propose a methodology for QoS evaluation and service ranking in the Digital Ecosystems environment that takes into account the context-dependent nature of Digital Ecosystem services. We need a methodology enabling service requesters to evaluate QoS based on specific service domain knowledge after service transactions, and the context-specific QoS can be employed for service ranking.

5. Propose a methodology for service domain knowledge updating, and service-provider-based SDE metadata publishing, maintenance and classification in the Digital Ecosystems environment that takes into account the dynamic nature of Digital Ecosystem services and service providers. This methodology includes: (a) a methodology by which service domain knowledge can adapt to the dynamic change of most people's perceptions toward domain knowledge; and (b) a methodology by which service providers can publish and modify their published SDE metadata, as well as determine the classification of the SDE metadata.

6. Validate the methodologies proposed in order to address Research Issues 1 to 5 by simulation experiments in order to evaluate the proposed approaches.

In this section, we define clearly each of the aforesaid research issues that needs to be addressed in order to solve the problems stated in Section 3.3

3.4.1 Research Issue 1: Propose a methodology for service information discovery, annotation and classification in the Digital Ecosystems environment

As mentioned previously, a service provider enters Digital Ecosystems by publishing an SDE, which will be stored into the distributed service knowledge bases [5]. Here the SDEs are stored in the form of SDE metadata [3]. When a service provider publishes an SDE, by means of a service factory, the SDE can be annotated by alternative Semantic Web mark-up languages, such as RDF and OWL, and classified by domain-specific ontologies provided within Digital Ecosystems, by referencing the Uniform Resource Identifier (URI) of the SDE metadata to ontological concepts [3, 22]. However, the service factory ignores the issue that, before the emergence of the service factory, service information has already been ubiquitous in Digital Ecosystems, and heterogeneous without sufficient ontological support. For instance, in the online Yellowpages®, Yahoo! or Google™ local search or other local business directories, a vast amount of service information is available. However, the service information is intermingled with other information, such as product information, which cannot be easily distinguished. Therefore, discovering the ubiquitous service information within the Digital Ecosystems environment is an issue for Digital Ecosystems.

As we discussed in Chapter 2, although there are many semantic crawlers available in the existing literature, none of them specializes in extracting service information from the Digital Ecosystems environment. This leads to a troublesome situation where Digital Ecosystems can classify only the limited registered service information in terms of the service factory but cannot deal with the ubiquitous unregistered service information.

As we discussed in Chapter 2, whereas there are many semantic service discovery approaches in the existing literature, none of them can be utilized in Digital Ecosystems, owing to the ignorance of discovering generic service information from the Web. Furthermore, the Digital Ecosystem service information is normally described by natural languages other than unified languages such as WSDL. Since the current semantic service discovery approaches mostly focuses on unified language-encoded services, they cannot be employed for Digital Ecosystem service discovery.

As we discussed in Chapter 1, in the local business repositories, there is vast service information available which cannot be easily retrieved. One reason for this phenomenon is that the service information is not annotated with Semantic Web mark-up languages. In most local business repositories, the service information is annotated by non-semantic mark-up languages, such as HTML or XHTML. In the case of service retrieval, the non-semantic service information may not be used to match the service requesters' ambiguous service queries. As a result, the lack of semantic annotation for the service information leads to the low precision of the service retrieval.

Another reason for the low precision of the service information retrieval is the lack of domain knowledge-oriented service information classification in the local business repositories. Here the domain knowledge can be represented by domain ontologies. Without the domain knowledge-oriented classification, the service information is mingled with other information with no means of classifying it. Whereas some local business directories provide non-semanticized taxonomies, as a result of the semantic-free supports, these taxonomies cannot help service requesters to denote their service intentions to a great extent. Hence, the lack of ontology-based classification can be considered as an issue of Digital Ecosystems.

As we discussed in Chapter 2, whereas there are many ontology-based classification approaches employed in the existing semantic crawlers, there is no technology specially designed for classifying the service information in the Digital Ecosystems environment. The main reason is that no specific service ontology has been designed that aims at modelling service domain knowledge. Hence, lacking service domain ontologies, current ontology-based classification technologies cannot be directly applied to the Digital Ecosystems environment.

In Chapter 4, we present the overview of the solution for the service discovery, service annotation and service classification methodologies for the Digital Ecosystems environment. In Chapter 5, we present a detailed conceptual framework of a semantic crawler, by means of which the service information in the Digital Ecosystems environment can be discovered, annotated and classified in an automatic manner. In Chapter 9, we present a methodology by which service providers can customize the classification of the SDE metadata to complement the automatic service classification methodology described in Chapter 6.

3.4.2 Research Issue 2: Propose a methodology for service retrieval in the Digital Ecosystems environment

In Digital Ecosystems, species are geographically dispersed and heterogeneous [5]. Nevertheless, from the perspective of services, the species that play the role of service requester need to locate the required species that play the role of service provider, in order to retrieve the services demanded. Therefore, the provision of a search engine enabling service requesters to retrieve service providers is a basic requirement for Digital Ecosystems. However, no such an infrastructure has been provided for Digital Ecosystems. Consequently, the lack of a service search engine to retrieve services and service providers can be regarded as an issue of Digital Ecosystems that needs urgent research attention.

Many local search engines can allow service requesters to retrieve service providers. As the scenario described in Chapter 1 demonstrates, due to the fact that these search engines are not facilitated with Semantic Web technologies, the search engines cannot return accurate results when ambiguous queries are entered. Thus, to enable a precise service search, a semantic service search engine is needed for Digital Ecosystems.

As was discussed in Chapter 2, there are many semantic service discovery approaches and semantic search engines mentioned in the existing literature. However, few of them can be applied to the Digital Ecosystem field. This is because none of the approaches is specifically designed for searching services in the Digital Ecosystems environment. Since the services in the Digital Ecosystems environment have the specification of heterogeneity, and the existing semantic service discovery approaches mostly focus on the Web service field, these approaches cannot cover all Digital Ecosystem services. Accordingly, a semantic service search engine specifically designed for retrieving service providers and services in the Digital Ecosystems environment is required.

In Chapter 4, we present the overview of the solution for service retrieval in the Digital Ecosystems environment. In Chapter 6, we present in detail the conceptual framework of a semantic service retrieval methodology for the Digital Ecosystems environment, by means of which a service requester can precisely find a service concept and all related SDEs.

3.4.3 Research Issue 3: Propose a methodology for service concept recommendation in the Digital Ecosystems environment

As was discussed in Chapter 2, most of the emerging semantic search engines use keyword-based search modules to match user queries with ontological concepts/facets, in order to disambiguate users' queries. Such a disambiguation process is realized by referencing connotations (matched ontological concepts) of queries from domain ontologies, and by interacting with users in order to indicate the extracted sense of users' queries. By means of this process, users' query intentions can be modelled. However, this methodology can be applied only to the situation where users have some knowledge about queries. For the users who do not have such knowledge, the keyword-based semantic search engines may not work appropriately. In other words, if users' initial query words are incomplete or incorrect, the final query results may not reflect their query intentions. In this situation, we may suppose that, even though the final query results may not indicate users' query intentions, if we use Human Centered Computing-based approaches to allow users to interact with search engines in the search process, the final query results may be relevant to the users' query intentions to some extent. Therefore, in order to avoid the failure of the semantic search engines, we may need a concept recommendation methodology in order to recommend relevant concepts to the users. In the Digital Ecosystems environment, we need to design a service concept recommendation methodology in order to enhance the dependability of the service retrieval methodology proposed for Research Issue 2.

As was discussed in Chapter 2, most of the semantic similarity models are designed for semantic networks. A semantic network is defined as "*a graphic notation for representing knowledge in patterns of interconnected nodes and arcs*" [25]. WordNet is a typical example of a semantic network, in which words or phrases are represented as nodes and are linked by multiple relations. Current semantic similarity models focus on estimating similarities between nodes. Since these nodes normally consist of single words or simple phrases, these models ignore the content of the nodes and deduce similarities based on the relative distance between the nodes or the position of the nodes in the whole semantic network. Nevertheless, in the ontology environment, each ontology concept is defined with semantic-rich content. For example, in OWL, each class is defined by its data type properties, object properties, restrictions, etc. Hence, it is obvious that the content of ontology concepts cannot be ignored when computing concept similarities within the ontology environment.

As was discussed in Chapter 2, most of the semantic similarity models are designed for definitional networks. A definitional network is a subset of semantic networks, in which the only type of relation is *class/subclass*, or called *is-a* [25]. Therefore, it is easy to visualize that each definitional network is a hierarchy of nodes linked by the *is-a* relations. In contrast, an ontology is more complicated than a definitional network. Although most ontologies have a hierarchical structure, the types of relations among concepts are more complicated and customizable. Obviously, the existing semantic similarity models may meet challenges when dealing with the multi-relational ontologies.

In Chapter 4, we present the solution overview for the service concept recommendation methodology. In Chapter 7, we present a methodology for recommending related service concepts to service requesters when service requesters find that the eventually retrieved service concepts do not match their query intentions in the Digital Ecosystems environment.

3.4.4 Research Issue 4: Propose a methodology for QoS evaluation and service ranking in the Digital Ecosystems environment

As was discussed in Chapter 1, since service information is ubiquitous, heterogeneous and context-dependent in the Digital Ecosystems environment, service requesters have to determine the reliability and trustworthiness of the service information after retrieving it. Therefore, it is crucial that the service information in the Digital Ecosystems environment be supplemented with QoS information. However, the current Digital Ecosystems do not have this function. Due to the inability to determine the QoS information, the retrieved service results may not necessarily satisfy service requesters. This is another research issue that needs urgent research attention, in order to ensure that the retrieved services match the QoS expectation of service requesters.

As a result, we should regard the lack of QoS information as a problem for Digital Ecosystems.

As was discussed in Chapter 2, in the existing semantic service discovery approaches, the employed QoS methods are designed for the field of Web services. However, in the Digital Ecosystems environment, the content of services is broader than that of the current research, in that it covers almost all the available service domains. Obviously, the current research has been unable to resolve the problem. In addition, since each service domain has its own special features, it is unfeasible to use a set of unified criteria to evaluate and compare the QoS for services in different domains. Last but not least, most of the current methods ignore the importance of service requesters' perceptions of the QoS. Nevertheless, as service requesters are direct recipient of services, their perceptions should be considered as the most important factor in the QoS evaluation process.

In Chapter 4, we present the solution overview for the QoS evaluation and service ranking methodology. In Chapter 8, we present the QoS evaluation and service ranking methodology by which service requesters can context-specifically evaluate the quality of services of service providers after service transactions, and the QoS can be used for ranking services under domain-specific QoS evaluation criteria.

3.4.5 Research Issue 5: Propose a methodology for service domain knowledge updating, and service-provider-based SDE metadata publishing, maintenance and classification in the Digital Ecosystems environment

As was mentioned in Sections 3.4.1 to 3.4.4, we propose to design domain ontologies for SDE metadata classification, semantic service retrieval, service concept recommendation, and domain knowledge-based QoS evaluation and service ranking. It is well known that ontologies are representation of domain knowledge. There are two research issues that need to be addressed in the maintenance of domain ontologies. The first issue is that, as domain knowledge is not static but evolves over time, we need to determine the most appropriate mechanism for maintaining consistency between dynamic domain knowledge and domain ontologies. The second issue is that, as end users are not involved in the ontology design, we need to find a means of ensuring that the ontologies are consistent with end users' perceptions of domain knowledge. In other words, if end users do not agree with the designed ontologies, we need to find a means of establishing agreements between end users and these designed ontologies.

As was discussed in Chapter 2, the performance of the existing semantic service discovery approaches, and semantic search engines and technologies that employ ontologies or facets, rely on the quality of ontologies or facets. They ignore the issue of ontology evolution, which takes account the dynamic nature of knowledge.

As was mentioned in Section 3.4.1, we propose a methodology for automatic service information classification. As the service information is published by service providers, service providers should have the absolute right to decide the classification of their service information. Therefore, if the service providers are not satisfied with the results of the automatic classification, they should be given the rights to customize the classification results. Hence, the design of a methodology that allows service providers to customize the service information classification results becomes an issue for Digital Ecosystems.

In Chapter 4, we present a solution overview for the service domain knowledge updating, and service-provider-based SDE metadata publishing, maintenance and classification methodology. In Chapter 9, we present in detail a methodology for enabling ontology evolution, QoS evaluation criteria updating, and service-provider-based SDE metadata publishing, maintenance and classification.

3.4.6 Research Issue 6: Validate the methodology proposed by simulation and functional testing experiments

We need to validate the solutions proposed for Research Issues 1 to 3. By validation, in the context of service discovery, annotation and classification, we intend to build an approximation or representation of a prototyping system that is based on the proposed customized semantic service retrieval methodology, thereby allowing us to verify the soundness of the proposed methodology. A customized semantic service retrieval methodology will help to establish confidence. In order to validate the methodology, we use a prototype approach. We present the solution overview for this research issue in Chapter 4, and in Chapter 10, we present the prototype used for validation of the proposed semantic service search engine.

3.5 Research Approach to Problem Solving

In addressing the stated problem, this thesis focuses on the development and subsequent testing and validation of a methodology for semantic service retrieval in the Digital Ecosystems environment. In order to propose a solution for the Research Issues 1 to 6 in the previous section, we need to follow a systematic scientific approach to ensure that the methodology development is scientifically based. Therefore, in this section, we provide an overview of the existing scientifically based research methods and give reasons for our choice of a particular research method.

3.5.1 Research Methods

There are two broad categories of research in information systems, namely: (a) the science and engineering approach, and (b) the social science approach. Science and engineering based research is concerned with confirming theoretical predictions. Gallier et al. [14] state that in the engineering field, the spirit of ‘making something work’ is essential and has three levels: conceptual level, perceptual level and the practical level, as explained below:

1. Conceptual level (level one): creating new ideas and new concepts through analysis

2. Perceptual level (level two): formulating a new method and a new approach through design and building the tools or environment or system through implementation

3. Practical level (level three): carrying out testing and validation through experimentation with real-world examples, using laboratory or field testing.

Science and engineering research may lead to new techniques, new architectures, new methodologies, new devices or a set of new concepts which together form a new theoretical framework. Frequently, it not only addresses the issue of what problems need to be addressed, but also proposes a solution.

Social science research can be either quantitative or qualitative research. It is often carried out through survey or interview processes. Quantitative research involves extensive data gathering usually using methods such as survey, and statistical analysis of the gathered data in order to prove or disprove various hypotheses that have been formulated. Qualitative research frequently involves in-depth structured or semi-structured interviews that allow one to pursue particular issues of interest that may arise during the interview. It does not normally involve a large sample of data and the information gathered may not be in a form that readily allows statistical analysis. A typical social science research approach, the use of survey forms, is used to identify problems which are subsequently formulated as hypotheses. The goal of social science research is to obtain evidence to support or refute a formulated hypothesis [4, 23, 24]. The research assists the researcher to understand people and social issues, such as culture, within the area of research. Kaplan et al. [8] argue that the ability to understand a phenomenon within its social and cultural context is forfeited when textual data results are quantified. This kind of research can indicate the extent to which the methodology is or is not accepted and sometimes may be able to give the reason for this. However, unlike engineering based research, this type of research does not explain what a methodology should be and how to produce a new methodology for problem solving. This research tests or evaluates only a method that has already been produced from science and engineering research.

This thesis deals with the development of a new methodology for customized semantic service retrieval in the Digital Ecosystems environment. Therefore, our research clearly falls into the science and engineering research domain.

3.5.2 Choice of Science and Engineering based Research Method

In this thesis, a science and engineering based approach is chosen as the research method for the development of the proposed solution.

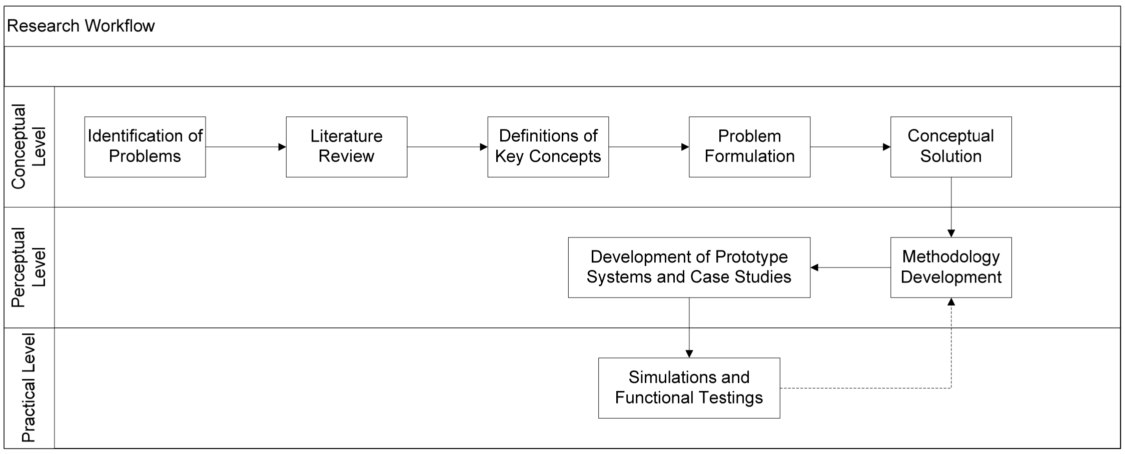


Figure 3.1: An engineering-based research approach

We began by identifying the research problems. We collected and analysed extensive literature on topics related to the study. Based on an extensive review of the existing literature, we formulated the problem that needs to be addressed. Subsequently, we defined some key concepts (for example, service, service providers, service requesters and so on) in the Digital Ecosystems environment. These definitions were used when developing the conceptual solution. Subsequently, we formulated the conceptual solution for the problem being addressed in this thesis. All processes ranging from the literature review to the conceptual solution fall under the umbrella of the conceptual level. At the perceptual level, we developed the methodology for Customized semantic service retrieval in the Digital Ecosystems environment. Subsequently, we engineered prototype systems and developed some case studies to be used later for testing of our proposed methodology. The processes of methodology development and development of prototype systems and case studies fall under the umbrella of the perceptual level. Once the prototype systems had been engineered, then, using them and the developed case studies, we validated our proposed methodology. At the practical level, based on the results obtained, we then evaluated and validated our proposed methodology. Based on the evaluation and validation, we then fine-tuned our proposed methodology.

With regard to aspects of research output evaluation and validation (Practical), Henver et al. argued that the evaluation of a design science research must be rigorously demonstrated via a well-executed evaluation method [19]. The selected evaluation methods must be appropriate for the designs and the selected evaluation metrics. Designs can be evaluated in term of functionality, completeness, consistency, performance, reliability, usability or other relevant quality attributes. By referring to our design in this thesis, we therefore intend to adopt the design methods proposed by Henver et al., which are simulation and functional testing. Simulation is used to execute a prototype with artificial data, and functional testing is used to execute a prototype to discover failures and identity defects.

3.6 Conclusion

In this chapter, we presented a formal definition of the problem that we intend to address in this thesis. The identified problem was subsequently decomposed as a set of six key cohesive research issues, which need to be solved in order to address the problem presented in this thesis. Each of the identified six research issues were explained in depth in relation to the existing literature and were subsequently defined formally. Furthermore, we outlined the different approaches to research and pointed out that we intend to implement a science and engineering research methodology in conjunction with the research methodology that uses system development as an information system research methodology.

In the next chapter, we present an overview of the solution to the problem being addressed in this thesis. Additionally, we present an overview of the solutions for each of the six research issues that comprise the problem being addressed in this thesis. The detailed framework of the service retrieval methodology is then described in Chapters 5-11.

3.7 References

1. Semantic similarity. Wikipedia (2009), http://en.wikipedia.org/wiki/Semantic_similarity

2. Barfourosh, A.A., Anderson, M.L., Nezhad, H.R.M., Perlis, D.: Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition. Department of Computer Science, University of Maryland, Maryland, USA (2002)

3. Boley, H., Chang, E.: Digital Ecosystems: Principles and semantics. Proceedings of the 2007 Inaugural IEEE International Conference on Digital Ecosystems and Technologies (DEST 2007). IEEE, Cairns, Australia (2007) 398-403

4. Burstein, F., Gregor, S.: The systems development or engineering approach to research in information systems: An action research perspective. Proceedings of the 10th Australasian Conference on Information Systems (ACIS 1999), Wellington, New Zealand (1999) 1222-1234

5. Chang, E., West, M.: Digital Ecosystem - A next generation of the collaborative environment. Proceedings of the 8th International Conference on Information Integration and Web-based Applications & Services (iiWAS 2006). ACM, Yogyakarta, Indonesia (2006) 3-24

6. Dini, P., Rathbone, N., Vidal, M., Hernandez, P., Ferronato, P., Briscoe, G., Hendryx, S.: The digital ecosystems research vision: 2010 and beyond. Creative Commons, Stanford, California, USA (2005), http://www.digital-ecosystems.org/events/2005.05/de_position_paper_vf.pdf

7. Dong, H., Hussain, F.K., Chang, E.: A service search engine for the industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

8. Dong, H., Hussain, F.K., Chang, E.: Ontology-based digital ecosystem conceptual representation. Proceedings of the Third International Conference on Autonomic and Autonomous Systems (ICAS '07). IEEE Computer Society, Athens, Greek (2007) 42-42

9. Dong, H., Hussain, F.K., Chang, E.: An integrative view of the concept of Digital Ecosystem. Proceedings of the Third International Conference on Networking and Services (ICNS '07). IEEE Computer Society, Athens, Greek (2007) 42-42

10. Dong, H., Hussain, F.K., Chang, E.: Developing a conceptual framework of semantic search engine to promote the collaborations between SMEs in the Digital Ecosystems environment. Proceedings of the 2008 2nd IEEE International Conference on Digital Ecosystem and Technologies (DEST 2008). IEEE, Phitsanulok, Thailand (2008) 409-412

11. Dong, H., Hussain, F.K., Chang, E.: A QoS-based service retrieval methodology for digital ecosystems. International Journal of Web and Grid Services **5** (2009) 261-283

12. Dong, H., Hussain, F.K., Chang, E.: State of the art in semantic focused crawlers In: Gervasi, O., Taniar, D., Murgante, B., Laganà, A., Mun, Y., Gavrilova, M. (eds.): Computational Science and Its Applications – ICCSA 2009. Springer, Korea (2009) 910-924

13. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

14. Galliers, R.D.: Information Systems Research: Issues, Methods and Practical Guidelines. Blackwell Scientific Publications (1992)

15. Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition **5** (1995) 199-220

16. Guha, R., McCool, R., Miller, E.: Semantic search. Proceedings of the 12th International World Wide Web Conference (WWW '03). ACM, Budapest, Hungary (2003) 700 - 709

17. Haase, P., Stojanovic, L.: Consistent evolution of owl ontologies. Proceedings of the 2nd European Semantic Web Conference (ESWC 2005). Springer-Verlag, Heraklion, Greece (2005) 182-197

18. Heistracher, T., Kurz, T., Masuch, C., Ferronato, P., Vidal, M., Corallo, A., Briscoe, G., Dini, P.: Pervasive service architecture for a digital business ecosystem. Proceedings of the18th European Conference on Object-Oriented Programming (ECOOP 2004). Springer-Verlag, Oslo, Norway (2004) 1-10

19. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly **28** (2004) 75-105

20. Hoffman, R.: Human-centered computing principles for advanced decision architectures. Army Research Laboratory (2004)

21. ITU-T Study Group: Teletraffic Engineering Handbook International Telecommunication Union (2006)

22. Malone, P.: DE services in ecosystem oriented architectures. In: Nachira, F., Nicolai, A., Dini, P., Lèon, L.R., Louarn, M.L. (eds.): Digital Business Ecosystems. European Commission (2007)

23. McTavish, D.G., Loether, H.J.: Social Research. Addison-Wesley Educational Publishers (1999)

24. Nunamaker, J.F., Chen, M., Purdin, T.D.M.: Systems development in information systems research. Journal of Management Information Systems **7** (1999)

25. Sowa, J.F.: Semantic Networks. In: Shapiro, S.C. (ed.): Encyclopaedia of Artificial Intelligence. John Wiley & Sons, New York, NY, USA (1992)

26. Swick, R.: Metadata and Resource Description. W3C (2003), http://www.w3.org/Metadata/

Chapter 4 – Solution Overview

4.1 Introduction

As explained in Chapter 2, several scholarly research works have made attempts and advances to solve the problem of service retrieval in the virtual environment (e.g. Web service environment). However, as evident from the discussions in Chapter 2 and 3, there is still the issue of how a service requester can precisely retrieve a high quality service provided by a reliable and trustworthy service provider in a Digital Ecosystems environment, taking into account both the heterogeneous, context-specific and dynamic nature of services and the heterogeneous and dynamic nature of service providers and service requesters in the Digital Ecosystems environment. In Chapter 3, we articulated the six research issues that will be addressed in our attempt to resolve this pivotal problem.

In this chapter, we present an overview of the solutions to each of the six research issues identified in the previous chapter. In Section 4.2, we propose the solution for the problem being addressed in this thesis. Subsequently, the solutions for the each of the research issues identified in the last chapter are presented in Sections 4.3 to 4.8. We then conclude the chapter in Section 4.9.

4.2 Overview of the Solution for Service Retrieval in the Digital Ecosystems Environment

In this section, we present an overview of the overall solution for service retrieval in the Digital Ecosystems environment – a customized semantic service retrieval methodology. In Section 4.3 to Section 4.9, we present an overview of the individual solutions for each of six research issues discussed in Chapter 3.

The diagrammatic representation of the solution overview is shown in Fig. 4.1. The core of the overview is a Service Knowledge Base, which is a repository used to store service domain knowledge, and information regarding services and service providers. All of the other solutions are designed based on the Service Knowledge Base, and their functions are realized by accessing the data from the knowledge base. In Chapter 5, we introduce the framework of the Service Knowledge Base for storing service domain knowledge and service information. In Section 4.3, we present the overview of the solution.

We divide the rest of the solutions into four clusters from the perspective of users, which are solutions for semantic crawlers (agents), service requesters and service providers.

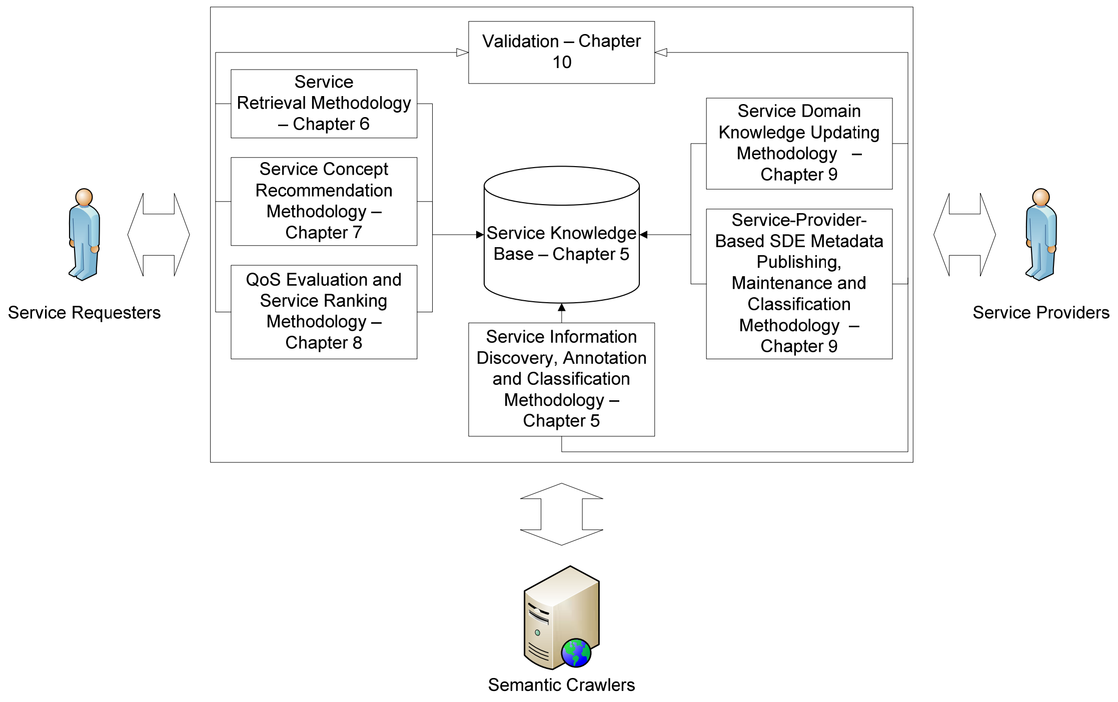

Validation – Chapter 10
Service Retrieval Methodology – Chapter 6
Service Concept Recommendation Methodology – Chapter 7
QoS Evaluation and Service Ranking Methodology – Chapter 8
Service Knowledge Base – Chapter 5
Service Domain Knowledge Updating Methodology – Chapter 9
Service-Provider-Based SDE Metadata Publishing, Maintenance and Classification Methodology – Chapter 9
Service Information Discovery, Annotation and Classification Methodology – Chapter 5
Service Requesters
Service Providers
Semantic Crawlers

Figure 4.1: Overview of the solution for customized semantic service retrieval in the Digital Ecosystems environment

First of all, the semantic crawlers can use the solution for service information discovery, annotation and classification in the Digital Ecosystems environment, which are comprised of the following three sub-solutions:

1. Solution for automatically discovering service information in the Web, given specific service domain knowledge. In Chapter 5, we propose a generic SDE metadata schema-based service information discovery methodology. In Section 4.3, we present the overview of the solution.

2. Solution for automatically annotating the discovered service information in the Web. In Chapter 5, we propose a generic SDE metadata schema-based service information annotation methodology, which converts a service snippet into SDE metadata. In Section 4.3, we present the overview of the solution.

3. Solution for automatically classifying the annotated SDE metadata. In Chapter 5, we propose a methodology for classifying the relevant SDE metadata and filter the non-relevant SDE metadata based on domain-specific service knowledge from the Service Knowledge Base. In Section 4.3, we present the overview of the solution.

Secondly, the solution for service requesters is comprised of:

1. Solution for assisting a service requester to denote a service concept which can precisely represent a service request of the service requester, by means of a series of human-centred interactions between the service requester and a semantic service search engine. In Chapter 6, we propose a methodology for service retrieval in the Digital Ecosystems environment. In Section 4.4, we present the overview of the solution.

2. Solution for recommending relevant service concepts in order to precisely match a service requester's query intention when the service requester cannot find a service concept from the Service Knowledge Base which can be used to express the service requester's service request. In Chapter 7, we propose a methodology for service concept recommendation. In Section 4.5, we present the overview of the solution.

3. Solution for a service requester to evaluate the quality of a service provided by a service provider, after a service transaction between the service requester and the service provider is completed, and solution for a service requester to rank the available services associated with a service concept based on the past QoS information. In Chapter 8, we propose a methodology for the QoS evaluation and service ranking in the Digital Ecosystems environment. In Section 4.6, we present the overview of the solution.

Thirdly, the solution for service providers comprises:

1. Solution for service providers to update service domain knowledge stored in the Service Knowledge Base. In Chapter 9, we propose a methodology for service domain knowledge updating. In Section 4.7, we present the overview of the solution.

2. Solution for service providers to publish new SDE metadata, maintaining the existing SDE metadata, and classifying the metadata according to own perceptions of service domain knowledge. In Chapter 10, we propose a methodology for the service provider-based SDE metadata publishing, maintenance and classification. In Section 4.7, we present the overview of the solution.

In addition, we also present a means for validating all the solutions previously stated. In Chapter 4.8, we present the overview of this solution.

4.3 Overview of the Solution for Service Information Discovery, Annotation and Classification in the Digital Ecosystems Environment

In the previous chapter, it was pointed out in Research Issue 1 that, in the existing literature, there is no methodology for discovering the heterogeneous and context-dependent service information existing in the Digital Ecosystems environment. Additionally, it was noted that none of the existing technologies can semanticize the service information in the current Digital Ecosystems. This results in the low precision of service retrieval. Finally, it was pointed out that the existing research cannot provide a solution for semantically classifying Digital Ecosystem service information, which is another reason for the poor performance of the current service search engines as shown in Chapter 1. Furthermore, there is a lack of domain knowledge to assist the semantic classification.

In order to address this research issue:

1. In Chapter 5, we propose a framework of a Service Knowledge Base, which is used to store the conceptualized service domain knowledge and to store the semanticized service information [2, 3]. With the two different objectives stated above, we divide a Service Knowledge Base into a Service Ontology Base and a SDE Metadata Base. Within them, the former is used to store service ontologies, and the latter is designed to store SDE metadata. A service ontology is the representation of the knowledge with regard to the taxonomy in a specific service domain, which is concerned with the abstraction of the generic service concepts and their relationships between concepts from that domain, e.g., truck transport and road transport can be viewed as two concepts in the transport domain, and the former is the subclass of the latter. A SDE metadata is the conceptualization of the information with regard to an actual SDE, which is concerned with the structuralized descriptions of a SDE provided by a service provider, e.g., a truck transport service provided by a transport company can be represented by a SDE metadata.

2. In order to standardize the information within the Service Knowledge Base for its further use, in Chapter 5, we propose a generic service concept schema [8, 9] for conceptualizing the service domain knowledge in the Digital Ecosystems environment, and we propose a generic SDE metadata schema [2, 3] for conceptualizing the service information in the Digital Ecosystems environment .

3. In Chapter 5, we propose a framework of a Semantic Crawler. By cooperating with the Service Knowledge Base introduced above, the Semantic Crawler is able

to implement the solution of the service information discovery, annotation and classification in the Digital Ecosystems environment. The whole working mechanism is [2, 3]:

a) The Semantic Crawler extracts service information by referring to the predefined service information extraction heuristics and the generic SDE metadata schema.

b) The Semantic Crawler generates SDE metadata by annotating the extracted service information with ontology mark-up languages.

c) The Semantic Crawler classifies the SDE metadata based on service ontologies. Two mathematical models – an Extended Case-Based Reasoning (ECBR) model [2, 10] and an Index term-based Case-Based Reasoning (IECBR) model [3] are used respectively to compute the similarity values between each SDE metadata and each service ontology concept. Based on the similarity value, the Semantic Crawler can determine whether or not each pair-wise SDE metadata and concept should be associated. As a result of this process, the SDE metadata are able to be classified by service ontologies, and irrelevant SDE metadata can be filtered based on specific service domain knowledge. The use of the two mathematical models in the metadata classification process is explained in Chapter 5.

4. In Chapter 5, we present in detail each of the steps involved in the service information discovery, annotation, and classification process, including two mathematical models in order to classify annotated SDE metadata and filter irrelevant SDE metadata based on specific service domain knowledge.

4.4 Overview of the Solution for Service Retrieval in the Digital Ecosystems Environment

In the previous chapter, it was pointed out in Research Issue 2 that, in the existing literature, there is no methodology available for service requesters to precisely retrieve a service provided by a service provider in the Digital Ecosystems environment. Furthermore, as stated in Chapter 1, current service search engines lack semantic supports, which leads to the low precision of search performance. Finally, the current research with regard to semantic search does not concentrate on semantic service retrieval.

In order to address the issue, in Chapter 6, we provide the framework of a Service Search Engine. Taking into account the heterogeneity of the service requesters in Digital Ecosystems, we propose two search modules respectively for the service requesters with relevant domain knowledge and the service requesters without relevant domain knowledge. For the former, we design a SPARQL-based search module, which allows a service requester to directly retrieve a SDE metadata by querying its attributes. For the latter, by means of a series of human-centred interactions between a service requester and the search engine interface, an interactive search module is designed in order to assist the service requester to precisely retrieve a service concept, which can be used to denote the query intention of the service requester to a maximum degree. The working process of the proposed interactive search module is [14]:

1. A service requester enters a service request to the interface of the semantic service search engine. The service request is normally comprised of plain texts and Boolean operations.

2. The service search engine returns to the service requester a list of service ontology concepts retrieved from the Service Ontology Base. Two mathematical models – an ECBR model [4] and an IECBR model [5] are respectively employed to compute the similarity values between the service request and all ontology concepts form the Service Ontology Base. The ontology concepts can then be ranked based on the similarity values. The use of the two models in the query-concept matching process is explained in Chapter 6.

3. By means of the returned ontology concept list and the structure of the service ontologies, the service requester can interact with the search engine in order to denote a concept which can best represent the service requester's query intention. This step is a recursive process until the service requester finally chooses a bottom-level concept, which is the concept associated with SDE metadata.

4. If a bottom-level concept of a service ontology is finally determined by the service requester, the SDE metadata associated with the concept are returned to the service requester.

In Chapter 6, we present in detail each of the steps involved in the service retrieval methodology, and two mathematical models for the classification of SDE metadata and filtering of irrelevant SDE metadata.

4.5 Overview of the Solution for Service Concept Recommendation in the Digital Ecosystems Environment

As indicated by Research Issue 3, the current semantic search engines do not have solutions for the service requesters who lack sufficient knowledge about their service requests. Moreover, most of the existing semantic similarity models focus on assessing similarity within the semantic network environment, which estimate the similarity based on absolute and relative locations between nodes. Nevertheless, when applying the semantic similarity models in the ontology environment, they cannot deal with customized relations, semantic-rich contents of ontology concepts, more restrictions and more complicated axioms.

In order to address this issue, in Chapter 7, we propose a service concept recommendation methodology. The mechanism of this methodology is [6]:

1. A service requester enters a service request to a search engine and eventually chooses a bottom-level concept of a service ontology. However, the service requester finds out that the service concept cannot be used to denote the service request. Based on the premise that the service concept should be relevant to the service concept that can be used to represent the service request to a maximum degree, two semantic similarity models are respectively utilized to compute the similarity values between the concept and the other concepts from the service ontology by considering both the factor of contents of concepts and relations, and the factor of locations of concepts [7, 12]. The similar concepts are then returned and ranked based on the similarity values. The two semantic similarity models are presented and discussed in Chapter 7.

2. The search engine can recommend a list of similar concepts to the service requester. The service requester can determine which concept can be used to denote the service request to a maximum degree from the concept list. Step 2 is a recursive process until the service requester finds the right service concept.

In Chapter 7, we present in detail each of the steps involved in the service concept recommendation methodology, and two semantic similarity models for the similar concept computation, by which the service requesters who do not have enough knowledge about own service requests can be assisted to denote their query intentions.

4.6 Overview of the Solution for QoS Evaluation and Service Ranking in the Digital Ecosystems Environment

In the previous chapter, as pointed out in Research Issue 4, the existing literature in semantic service discovery focuses mostly on the QoS in the field of Web services, and thus few of them can be applied to the field of Digital Ecosystem services, as the latter has more contents, including both of virtual services and concrete services in the world. In addition, owing to the diversity of Digital Ecosystem services, we cannot evaluate or rank the services according to a set of common QoS criteria. Moreover, many current methodologies ignore the factor of service requesters' subjective perceptions towards QoS. However, as only recipients of services, the service requesters' subjective perceptions should be considered as the most important factor in QoS evaluation.

In order to address this issue, in Chapter 8, we deliver a QoS evaluation and service ranking methodology. This methodology is extended from the theory of Chang et al.'s CCCI (correlation, commitment, clarity and importance) metrics which is used for trust modelling in a Service-Oriented Environment [1]. Here we extend the theory of CCCI metrics for QoS evaluation and ranking in the Digital Ecosystems environment, by cooperating with service ontologies that are used to represent service domain knowledge and SDE metadata that are used to represent actual services. The working process of the QoS evaluation and service ranking methodology is [11, 13]:

1. For each service ontology, each bottom-level concept of the ontology is regarded as the conceptualization of an independent service domain. Therefore, each bottom-level concept should have domain-specific QoS evaluation criteria. Furthermore, all SDE metadata associated with a bottom-level concept should follow the same QoS evaluation criteria assigned to this concept.

2. When a service requester completes a service transaction with a service provider, the service requester can obtain an authorization. This authorization would enable the service requester to evaluate the quality of the service provided by the service provider.

3. The QoS evaluation criteria of the service (SDE metadata) can be obtained by referring to the service ontology concept associated by the service (SDE metadata).

4. The service requester can evaluate the performance of the service provider under each criterion.

5. The trustworthiness value of the service (SDE metadata) is obtained by using the extended CCCI metrics to aggregate the service requester's evaluation value towards each criterion. The extended CCCI metrics are presented and discussed in Chapter 8.

6. The reputation value of the service provider on this service (SDE metadata) can be obtained by leveraging all available trustworthiness values of the service assigned by all service requesters.

7. The performance value of the service provider under each criterion of the service (SDE metadata) can be computed by leveraging all available evaluation values on this service (SDE metadata) under each criterion provided by all service requesters.

8. When a service requester chooses a bottom-level concept, all its associated SDE metadata are ranked in a multi-linear manner – either based on the reputation values of service providers on these SDE metadata or single performance values of service providers under each criterion which the bottom-level concept corresponds to.

In Chapter 7, we present in detail each of the steps in the QoS evaluation and service ranking methodology, and we present the mathematical models of the extended CCCI metrics. A case study is provided to illustrate the use of the methodology for QoS evaluation and service ranking.

4.7 Overview of the Solution for Service Domain Knowledge Updating and Service-Provider-based SDE Metadata Publishing, Maintenance and Classification in the Digital Ecosystems Environment

First of all, in the previous chapter, as pointed out in Research Issue 5, service ontologies are not static, since the domain knowledge that the ontologies represent is dynamic along with changes of objective environment and changes of subjective perceptions. Moreover, ontologies are designers' subjective understandings of the objective world; consequently, it needs to establish agreements between designers and end users, and between end users (both service providers and service requesters). Analogously, the domain-specific QoS evaluation criteria are unilateral to some extent, as initial criteria are designed by system designers. Therefore, it needs to make agreements between end users and designers, and between end users.

In order to address the issue above, in Chapter 9, we propose a service domain knowledge updating methodology, which is a community-based voting mechanism. If a change request (either in ontologies or in QoS evaluation criteria) is initiated by either a normal user (service requester or service provider who does not have any SDE metadata associated with the concept that the service provider wants to change) or a domain expert (service provider who has a SDE metadata associated with the concept that the service provider wants to change), the voting mechanism invites both normal users and domain experts to vote for the change request. Finally, the voting mechanism combines the votes from both sides to determine whether or not the change request is accepted [14].

Secondly, as pointed out in Research Issue 5, in the existing literature, the research mostly separate virtual service (e.g. Web services) and concrete service (e.g. business services) publishing/registry, and thus there is no specifically designed methodology to integrate the services from both aspects. Nevertheless, since Digital Ecosystem services cover both virtual services and concrete services, a methodology is required for the Digital Ecosystem service publishing. In addition, although the Semantic Crawler proposed in Chapter 5 enables the automatic classification of SDE metadata published by service providers, owing to the differences in individual understandings of domain knowledge, service providers may have their own understanding with respect to the classification of own SDE metadata. As a result, a methodology is required to enable the service provider-based SDE metadata classification [3, 14].

In order to address the issue above, in Chapter 9, we present a human-centred methodology for service providers to publish and maintain their service, and classify their SDE metadata according to personal preferences.

4.8 Overview of the Solution for the Validation of the Proposed Methodology

In this thesis, we make use of the evaluation method of simulation and functional testing, in order to validate and evaluate the customized semantic service retrieval methodology in the Digital Ecosystems environment. Specifically, with respect to the five solutions above (Section 4.3 to Section 4.7), we propose the following evaluation methods:

1. Semantic Crawler Functional Testing and Simulation. The objectives of the Semantic Crawler include three perspectives – service information discovery, annotation and classification. For the former two objectives, we build a prototype for the Semantic Crawler and run the prototype in different service domains and in several websites. Subsequently, we validate the functions of service information discovery and annotation of the crawler by observing the annotated service metadata. For the third objective, as several mathematical models are involved in the methodology, we need to compare their outputs with an existing mathematical model used in this similar field, based on several information retrieval metrics. The objective of the Semantic Crawler functional testing and simulation is to validate the functionality, completeness, usability, performance and reliability of the service information discovery, annotation and classification methodology. The Semantic Crawler functional testing and simulation is presented in Chapter 10, along with the results obtained.

2. Service Retrieval Simulation and Functional Testing. The validation of the service retrieval methodology can be divided into two steps, as the methodology incorporates the algorithm-based service retrieval and HCIs. Firstly, we simulate the service retrieval methodology by building a prototype – a semantic service search engine, and using artificial service requests from different service domains. Based on the same input, we compare the performance of the search engine with an existing service search engine, in terms of several information retrieval metrics, with the purpose of validating the whole methodology. We compare the performance of mathematical models used in the methodology with several existing mathematical models used in this domain, based on several information retrieval metrics. Secondly, we run the prototype of the service retrieval methodology under several premised scenarios, in order to validate the proposed HCI functions of the methodology. The objective of the service retrieval simulation and functional testing is to validate the usability, performance, reliability, functionality and completeness of a service retrieval methodology. The service retrieval simulation and functional testing is presented in Chapter 10, along with the results obtained.

3. Service Concept Recommendation Simulation. For the semantic similarity models deployed in the service concept recommendation methodology, we simulate the methodology by using artificial service ontologies created for

different service domains. Based on these ontologies, we compare the output of the proposed semantic similarity models with several existing semantic similarity models, based on several information retrieval metrics. The objective of the service retrieval simulation is to validate the usability, performance and reliability of service concept recommendation methodology. The service concept recommendation simulation is presented in Chapter 10, along with the results obtained.

4. QoS Evaluation and Service Ranking Functional Testing. We run the prototype of the QoS evaluation and service ranking methodology by means of simulating several use scenarios from different service domains, in order to validate the feasibility of the proposed functions. The objective of the functional testing is to validate the usability, functionality, completeness of the QoS evaluation and service ranking methodology. The functional testing is presented in Chapter 10, along with the results obtained.

5. Service Domain Knowledge Updating and Service Provider-Based SDE Metadata Publishing, Maintenance and Classification Functional Testing. First of all, we test the functions of the service domain knowledge updating methodology, by means of running its prototype in different use scenarios. Secondly, we test the functions of the SDE metadata publishing, maintenance and classification interface in different use scenarios. The objective of the functional testing is to validate the usability, functionality, completeness of the service domain knowledge updating and service provider-based SDE metadata publishing, maintenance and classification approach. The functional testing is presented in Chapter 10, along with the results obtained.

4.9 Conclusion

In this chapter, we presented the overview of our solution to the problem that is being addressed in this thesis, and to the six cohesive research issues that were identified in Chapter 3.

In the next chapter, we propose the service information discovery, annotation and classification methodology, which was identified in this chapter as being an important part of the customized semantic service retrieval methodology.

4.10 References

1. Chang, E., Dillon, T.S., Hussain, F.: Trust and Reputation for Service Oriented Environments-Technologies for Building Business Intelligence and Consumer Confidence. John Wiley & Sons (2005)

2. Dong, H., Hussain, F.K., Chang, E.: Focused crawling for automatic service discovery, annotation and classification in industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

3. Dong, H., Hussain, F.K., Chang, E.: A framework for discovering and classifying ubiquitous services in digital health ecosystems. Journal of Computer and System Sciences **In Press**

4. Dong, H., Hussain, F.K., Chang, E.: A service search engine for the industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

5. Dong, H., Hussain, F.K., Chang, E.: Semantic service matchmaking for digital health ecosystems. Knowledge-Based Systems **Submitted**

6. Dong, H., Hussain, F.K., Chang, E.: A service concept recommendation system for enhancing the dependability of semantic service matchmakers in the service ecosystem environment. Journal of Network and Computer Applications **Submitted**

7. Dong, H., Hussain, F.K., Chang, E.: A context-aware semantic similarity model for ontology environments. Concurrency and Computation: Practice and Experience **In Press**

8. Dong, H., Hussain, F.K., Chang, E.: Transport service ontology and its application in the field of semantic search. Proceedings of the 2008 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI 2008). IEEE, Beijing, China (2008) 820-824

9. Dong, H., Hussain, F.K., Chang, E.: A transport service ontology-based focused crawler. In: Hai, Z. (ed.): Proceedings of the fourth International Conference on Semantics, Knowledge and Grid (SKG '08). IEEE Computer Society, Beijing, China (2008) 49-56

10. Dong, H., Hussain, F.K., Chang, E.: A semantic crawler based on an extended CBR algorithm. In: Meersman, R., Tari, Z., Herrero, P. (eds.): On the Move to Meaningful Internet Systems: OTM 2008 Workshops. Springer-Verlag, Monterrey, Mexico (2008) 1084-1093

11. Dong, H., Hussain, F.K., Chang, E.: Quality of service (QoS) based service retrieval engine. In: Gabriele Kotsis, D.T., Eric Pardede, Ismail Khalil (ed.): Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM '08). ACM, Linz, Austria (2008) 405-408

12. Dong, H., Hussain, F.K., Chang, E.: A hybrid concept similarity measure model for the ontology environment. In: Meersman, R., Tari, Z., Herrero, P. (eds.): On the Move to Meaningful Internet Systems: OTM 2009 Workshops. Springer-Verlag, Vilamoura, Portugal (2009) 848-857

13. Dong, H., Hussain, F.K., Chang, E.: A QoS-based service retrieval methodology for digital ecosystems. International Journal of Web and Grid Services **5** (2009) 261-283

14. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

Chapter 5 – Service Information Discovery, Annotation and Classification Methodology

5.1 Introduction

In this chapter, we present a methodology for service information discovery, annotation and classification. The objectives of this methodology include:

- Automatically discovering heterogeneous service information from the Digital Ecosystems environment in terms of domain-specific service knowledge and crawling technology.

- Automatically semanticizing the discovered service information by means of service domain knowledge.

- Automatically organizing the annotated service information based on specific service domain knowledge.

The methodology is presented by a prototype comprised of two main parts as:

- A Service Knowledge Base. The Service Knowledge Base is designed for storing domain-specific service knowledge and annotated service information. In addition, the annotated service information is classified by the service domain knowledge.
- A Semantic Crawler. The Semantic Crawler implements the task of automatic service discovery, annotation and classification, by making use of the Service Knowledge Base.

In the next section, we reveal the proposed methodology by introducing the system architecture of its prototype.

5.2 System Architecture

In this section, we propose the system architecture of the prototype of the service information discovery, annotation and classification methodology, which can be found in

Fig. 5.1. As mentioned in Section 5.1, the whole system architecture consists of two main parts – a Service Knowledge Base and a Semantic Crawler.

The rest of this chapter is organized as follows: in Section 5.3, we introduce the components of the Semantic Crawler and their functions; in Section 5.4, we describe the components of the Service Knowledge Base; in Section 5.5, we explain the whole system workflow to show how these components collaborate to realize the objective of automatic service information discovery, annotation and classification; in Section 5.6, we introduce two algorithms designed with the purpose of SDE metadata classification; and in Section 5.7 we make a conclusion toward this chapter.

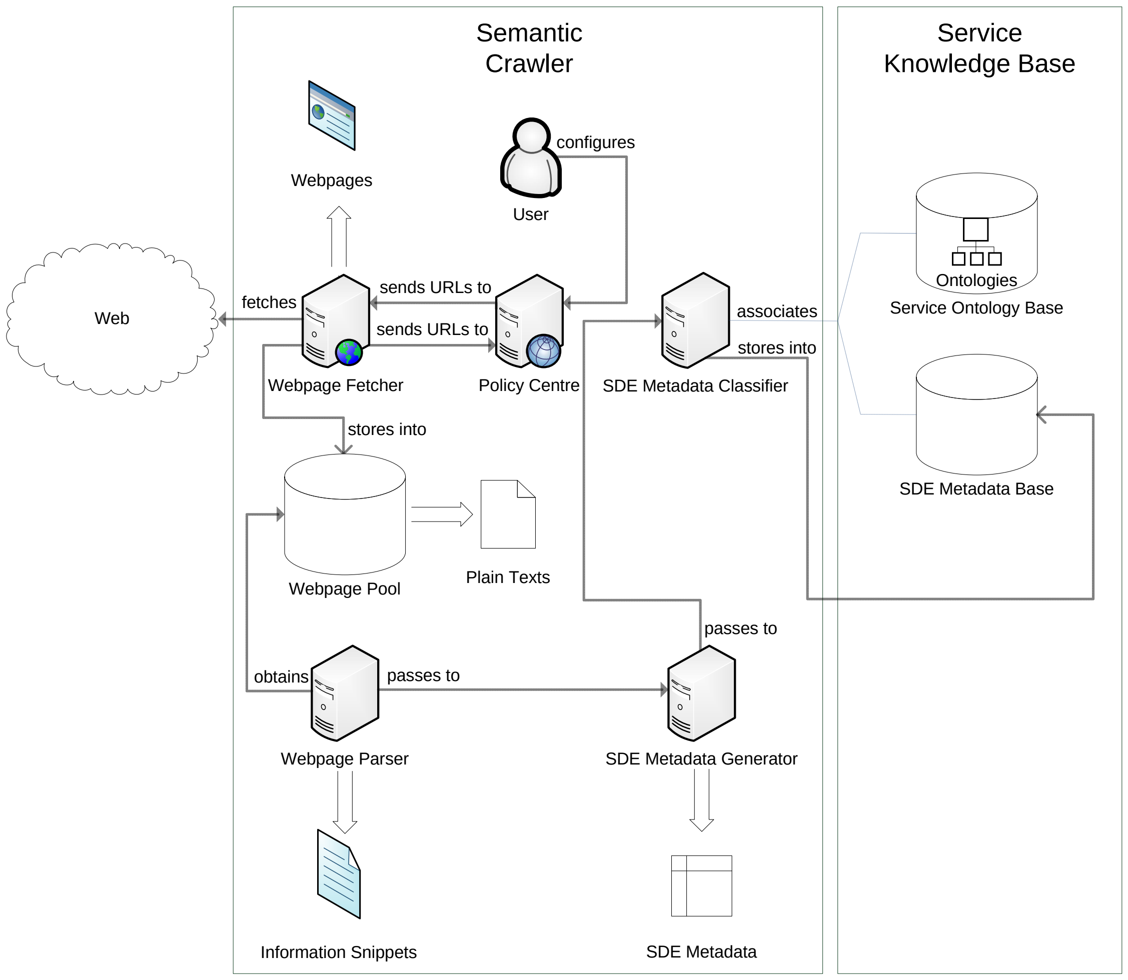


Figure 5.1: System architecture of the prototype of the service information discovery, annotation and classification methodology

5.3 Service Knowledge Base

As introduced earlier, the Service Knowledge Base is employed to provide specific service domain knowledge and to store the annotated and classified service information. As shown in Fig. 5.1, the Service Knowledge Base has two main components:

- A Service Ontology Base, which is designed to store service ontologies. A service ontology is the representation of the service knowledge within a specific service domain. In the Digital Ecosystems environment, a service ontology is a hierarchy of service concepts, in which each concept is an abstraction of the service entities that share some common features. These service concepts are related by the concept-subconcept relations (which can be represented by <rdfs:subClassOf> in RDFS and OWL), in which a subconcept is the specification of its parent concept, and a concept is the generalization of its subconcept(s). With the purpose of classification and disambiguation of SDE metadata, only the most specific concepts in an ontology, namely the concepts without subconcepts, can be associated with SDE metadata [5]. In order to better represent specific service domain knowledge, and to make service ontologies work better in the process of service ontology-based SDE metadata classification (introduced in Section 5.6) and the process of service ontology-based service retrieval (introduced in Chapter 6), we design an extendible general service concept schema for all service domains in the Digital Ecosystems environment.

- A SDE Metadata Base, which is designed to store the annotated and classified SDE metadata. Here we design an extendible general SDE metadata schema, with the purpose of describing the features of service entities from all service domains in the Digital Ecosystems environment [5]. The general SDE schema can be employed in the process of service information discovery, annotation and classification (introduced in Section 5.5 and 5.6), the process of service retrieval (introduced in Chapter 6), and the process of service publishing and maintenance (introduced in Chapter 9).

5.3.1 General Service Concept Schema

In this section, we define a general service concept schema, in order to represent the service knowledge in all service domains of the Digital Ecosystems environment. As described previously, there are two primary types of general service concepts in a service ontology, which are defined as [2, 3] :

- Abstract general service concept, which makes possible further specialization, namely subconcepts. Abstract general service concepts cannot be associated with SDE metadata, since these concepts are too general to specify the service domains of SDE metadata, which may generate ambiguities in the process of SDE metadata classification and service retrieval.

- Concrete general service concept, which does not have the possibility of further specialization. Concrete general service concepts can be associated with SDE metadata, owing to the feature that they are specialized so as to define specific service domains, which are able to disambiguate the classification of SDE metadata.

The abstract general service concept consists of at least one property as:

- **conceptDescription,** which is a datatype property of a general service concept. It refers to a body of text that is able to define and describe the concept. Owing to the differences between service domain knowledge, the number of this property can be extended to more than one according to actual definitions of service concepts. This property can be used in the computing process for the SDE metadata classification and the forthcoming service retrieval [5].

The abstract general service concept schema in OWL is shown in Fig. 5.2.

```
<owl:Class rdf:ID="Abstract_General_Service_Concept"/>
   <owl:DatatypeProperty rdf:ID="conceptDescription_1">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Abstract_General_Service_Concept"/>
   </owl:DatatypeProperty>
……
   <owl:DatatypeProperty rdf:ID="conceptDescription_M">
……
</owl:Class>
```

Figure 5.2: Abstract general service concept schema in OWL

Apart from the *conceptDescription* property, the concrete general service concept has a property as:

- **linkedMetadata,** which is an object property of a general service concept. It is used to store the URIs of the SDE metadata that associate to the concept. This property can be used for the forthcoming SDE metadata classification process [5].

The concrete general service concept schema in OWL is shown in Fig. 5.3.

```
<owl:Class rdf:ID="Concrete_General_Service_Concept"/>
   <owl:DatatypeProperty rdf:ID="conceptDescription_1">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="# Concrete_General_Service_Concept"/>
   </owl:DatatypeProperty>
……
   <owl:DatatypeProperty rdf:ID="conceptDescription_M">
……
   <owl:ObjectProperty rdf:ID="linkedMetadata">
      <owl:inverseOf rdf:resource="#linkedConcepts"/>
      <rdfs:domain rdf:resource="#Concrete_General_Service_Concept"/>
      <rdfs:range rdf:resource="#SDE_Metadata"/>
   </owl:ObjectProperty>
</owl:Class>
```

Figure 5.3: Concrete general service concept schema in OWL

5.3.2 General SDE Metadata Schema

In this section, we define a general SDE metadata schema, which refers to the detailed description of a service entity provided by a service provider [6]. The schema can benefit the semantic crawler, service providers and service requesters from three perspectives, which are:

1. The semantic crawler is able to discover service information, annotate the discovered service information, and classify the annotated SDE metadata based on the general SDE metadata schema.

2. The service providers are able to publish, maintain and classify their services based on the schema.

3. The service requesters are able to directly retrieve requested services by SPARQL based on the schema.

The general SDE metadata schema consist of two-double parts, which is a general SDE metadata schema that describes a service entity and a general service provider schema that describes a service provider. The former is linked to the latter by a many-to-one relationship. This design has three advantages:

1. It accords with the fact that a service provider can provide multiple services.

2. It can compress the volume of metadata and thus can save the storage of metadata.
3. It can facilitate the maintenance of SDE metadata of service providers.

Fig. 5.4 shows a fragment of the general SDE metadata schema in OWL, which defines the properties of a general SDE metadata as [6]:

- **SDEName.** This property provides the name of a service entity provided by a service provider.

- **serviceDescription**. In contrast to the *conceptDescription* of the general service concepts, the *serviceDescription* stores the detailed description of a service entity. Similarly, this number of properties can be an arbitrary amount, which depends on the number of information snippets describing a service entity. Analogously, this property is used in the computing process for the forthcoming SDE metadata classification.

- **linkedConcepts.** This property is the inverse property of the *linkedMetadata*, which is used to store URIs of the associated service concepts in order to realize the association process.

- **isProvidedBy.** This property is used to reference a service provider metadata by storing its URI, which reflects the fact that the service provider provides the service entity represented by the SDE metadata.

```
<owl:Class rdf:ID="SDE_Metadata"/>
  <owl:DatatypeProperty rdf:ID="SDEName">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#SDE_Metadata"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="serviceDescription_1">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#SDE_Metadata"/>
</owl:DatatypeProperty>
……
  <owl:DatatypeProperty rdf:ID="serviceDescription_N">
……
  <owl:ObjectProperty rdf:ID="linkedConcepts">
    <rdfs:range rdf:resource="#Concrete_Gerneral_Service_Concept"/>
    <rdfs:domain rdf:resource="#SDE_Metadata"/>
    <owl:inverseOf rdf:resource="#linkedMetadata"/>
  </owl:ObjectProperty>
  <owl:FunctionalProperty rdf:ID="isProvidedBy">
    <rdfs:domain rdf:resource="#SDE_Metadata"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <owl:inverseOf>
      <owl:InverseFunctionalProperty rdf:ID="provides"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="#Service_Provider"/>
  </owl:FunctionalProperty>
</owl:Class>
```

Figure 5.4: General SDE metadata schema in OWL

Fig. 5.5 depicts an instance of the general service provider metadata schema in OWL, which defines the properties of a general service provider metadata as [6]:

- **providerName.** This property is used to store the identity of a service provider, e.g. the company name.

- **providerProfile.** This property is used to store the descriptive information about the profile of a service provider.

- **address.** This property is used to store the address information of a service provider.

- **contactDetails.** This property is used to store the contact information of a service provider, including phone number, fax number, URL of website, email etc.

- **provides.** *provides* is the inverse property of the *isProvidedBy* property of the general SDE metadata, which is used to store the URI(s) of the SDE metadata that are linked to a service provider metadata.

```
<owl:Class rdf:ID="Service_Provider"/>
  <owl:DatatypeProperty rdf:ID="providerName">
    <rdfs:domain rdf:resource="#Service_Provider"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="providerProfile">
    <rdfs:domain rdf:resource="#Service_Provider"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="address">
    <rdfs:domain rdf:resource="#Service_Provider"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="contactDetails">
    <rdfs:domain rdf:resource="#Service_Provider"/>
  </owl:DatatypeProperty>
  <owl:InverseFunctionalProperty rdf:about="#provides">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#Service_Provider"/>
    <rdfs:range rdf:resource="#SDE_Metadata"/>
    <owl:inverseOf rdf:resource="#isProvidedBy"/>
  </owl:InverseFunctionalProperty>
</owl:Class>
```

Figure 5.5: General service provider metadata schema in OWL

It is important to note that the general service concept schema and general SDE metadata schema are both defined for the general domains in the Digital Ecosystems environment. Since diversity is a characteristic of the services in Digital Ecosystems, the knowledge in each service domain varies significantly. Therefore, we allow certain properties of the schemas extendible and adapt to the variation of service domain knowledge. Especially for the property *conceptDescription*, we can define multi-domain service concepts by adding different contents to it and changing the number of this property.

By defining the three schemas above, we construct an association map between service concepts, SDE metadata and service provider metadata in the Service Knowledge Base. Within them, the service concept and SDE metadata follow a many-to-many relationship and SDE metadata and service provider metadata follow a many-to-one relationship. These associations enable further efficient search applications within the Service Knowledge Base from any of the three perspectives [6]. An example of the service concept-SDE metadata-service provider metadata association map is presented in Fig. 5.6.

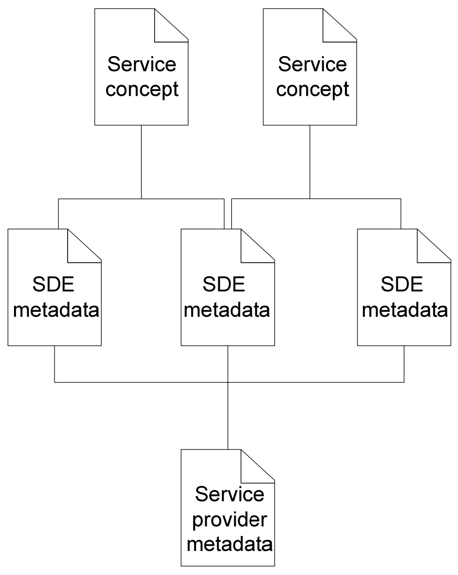


Figure 5.6: Example of service concept-SDE metadata-service provider association map

5.4 Semantic Crawler

In this section, we explain the functions of each component involved in the system architecture of the Semantic Crawler (Fig. 5.1).

The semantic crawler consists primarily of five agents – Webpage Fetcher, Policy Centre, Webpage Parser, SDE Metadata Generator and SDE Metadata Classifier, and a repository – Webpage Pool, which are described as follows [5]:

- **Webpage Fetcher**. Its function is to selectively download webpages by using the multi-threading technology which allows several webpage fetching processes to work concurrently. Given a list of Uniform Resource Locaters (URLs), the Webpage Fetcher can download the webpages linked by the URLs. In addition, the Webpage Fetcher can extract the URLs from the downloaded webpages, and send them to the Policy Centre for further analysis.

- **Policy Centre**. The function of the Policy Centre is to control the behaviour of the Webpage Fetcher by configuring several policies introduced as follows:

    - **Selection Policy** is defined in order to regulate the initial URLs that a Webpage Fetcher needs to visit, and to define the fetching boundary in order to ensure that a Webpage Fetcher does not escape from a configured website. Additionally, the policy can also be used to rank the priorities of visiting URLs, by means of setting up a set of heuristic rules for analysing

URL annotations (e.g., URLs annotated with “next page” and page numbers etc. may have higher visiting priorities).

- **Maximum Visiting Policy** is designed to regulate the maximum depth to which a website can be explored, in order to avoid overloading a Webpage Fetcher.
- **Parallelization Policy** is used to coordinate multi-threading fetching processes. The main task is to distribute the initial visiting URLs to each process and, by means of tracking the visited URLs, to avoid repetitively visiting the same URLs.

- **Webpage Pool.** Webpage Pool is designed to store the Web documents downloaded by the Webpage Fetcher. Here, all the embedded webpage mark-up language tags, such as Hypertext Mark-up Language (HTML) tags and Extensible Mark-up Language (XML) tags, and scripting language tags, such as JavaScript tags, are removed from the webpages. As a result, webpages are stored in the form of plain texts.

- **Webpage Parser.** The task of the Webpage Parser is to extract meaningful information snippets from the Web documents stored in the Webpage Pool. This is realized by following a set of heuristic rules for the processing of text. In order to deal with the information heterogeneity in the large number of webpages, we define heuristic rules by referring to actual webpage layouts in websites and the general SDE metadata schema (introduced in Section 5.3.2), since service information in a website usually maintains a consistent style.

- **SDE Metadata Generator.** SDE Metadata Generator is employed to produce metadata by annotating the information snippets obtained by the Webpage Parser with the ontology mark-up languages. The annotation process follows the general SDE metadata schema (introduced in Section 5.3.2).

- **SDE Metadata Classifier.** SDE Metadata Classifier has the mission of employing structured domain knowledge to classify the generated SDE metadata, by means of associating the metadata with predefined service domain ontology concepts. Whether or not a couple of metadata and concept should be associated is based on the extent of the similarity between the metadata and each service concept. There are specially designed mathematical models for the similarity computation, which are discussed in Section 5.6.

5.5 System Workflow

As introduced previously, the Service Knowledge Base and Semantic Crawler collaborate together to realize the objective of automatic service information discovery, annotation and classification. In this section, we introduce the workflow of the process in detail as follows [5]:

Step 1. Before the semantic crawler starts to work, users need to configure the initial URLs of visiting websites (usually the websites' domain names), and the depth of exploring the websites in the Policy Centre. Once the configuration is completed, the Policy Centre sends the commands to the Webpage Fetcher for the webpage crawling.

Step 2. The Webpage Fetcher will start to obtain webpages after it receives the URL list. Once the Webpage Fetcher downloads a webpage, it will extract the URLs in the webpage and send them to the Policy Centre for further analysis. The webpage will be sent to the Webpage Pool for storage purposes.

Step 3. When the Policy Centre receives the URLs from the Webpage Fetcher, it will determine whether or not they are within the crawling boundary, by analysing their domain names. After that, the Policy Centre will rank the URLs by their visiting priorities and send them back to the Webpage Fetcher. Steps 2 and 3 are a recursive process until the user-defined website exploration depth has been reached.

Step 4. Once a webpage has been passed to the Webpage Pool, all its embedded tags will be removed and the webpage will be stored in the form of plain texts.

Step 5. The Webpage Parser will obtain the processed webpage information from the Webpage Pool, extract the meaningful information snippets from each webpage, and pass them to the SDE Metadata Generator.

Step 6. The SDE Metadata Generator will annotate the delivered information snippets with the ontology mark-up languages, in order to create SDE metadata. The SDE metadata will then be passed to the SDE Metadata Classifier.

Step 7. On receiving the SDE metadata from the SDE Metadata Generator, the SDE Metadata Classifier will either classify the metadata based on the ontologies from the Service Ontology Base and store the metadata into the SDE Metadata Base, or filter the metadata, in terms of a SDE metadata classification process. The technical details regarding the SDE metadata classification process are discussed in the next section.

5.6 SDE Metadata Classification Process

5.6.1 SDE Metadata Classification Process

In this section, we introduce the technical details regarding the SDE metadata classification process by the algorithmic style [5], which can be mathematically expressed in Fig. 5.7.

Input: $C = (c_1, c_2 \ldots c_m)$ is a sequence of concrete service concepts of the service ontologies from a Service ontology Base, $M = (m_1, m_2 \ldots m_n)$ is a sequence of SDE metadata generated by a SDE Generator.
Output: The relevant SDE metadata are associated with corresponding concrete service concepts and stored into a SDE Metadata Base, and non-relevant SDE metadata are filtered.
Procedure:

```
1. For i = 1 to n
2.    Fetch the serviceDescription values of $m_i$ and store it into $sd_i$
3.    For j = 1 to m
4.       Fetch conceptDescription property values of $c_j$ and store it into $cd_j$
5.       Compute the similarity value between $sd_i$ and $cd_j$ by an algorithm and store the
value into $s_{ij}$
6.       If $s_{ij}$ > a threshold value then
7.          Put the URI of $c_j$ into the linkedConcepts property of $m_i$
8.          Put the URI of $m_i$ into the linkedMetadata property of $c_j$
9.          Store $m_i$ into the SDE Metadata Base.
10.      End if
11.   End for
12. End for
```

Figure 5.7: SDE metadata classification process

5.6.2 SDE Metadata Classification Algorithms

As observed from Fig. 5.7, the SDE metadata classification process uses an algorithm to compute the similarity value between a SDE metadata and a service concept. Based on the similarity value and a threshold value, the SDE Metadata Classifier can determine whether or not the metadata should be associated. To realize the objective of metadata-concept similarity computation, we design two alternative algorithms, which are an Extended Case-Based Reasoning (ECBR) algorithm [4, 5] and an Index term-based Case-Based Reasoning (IECBR) algorithm [6], both of which are based upon the Case-Based Reasoning (CBR) algorithm [1].

The CBR algorithm is used to retrieve and reuse the existing problem solutions for emerging problems, which has four sub-processes as follows [1]:

1. **Retrieve**: a new problem is matched with cases in the database.
2. **Reuse**: if there are cases matched, the solutions to the retrieved cases are reused as the solutions to the emerging problem.
3. **Revise**: if the retrieved cases cannot completely match the problems, the solutions to the problem need to be revised.
4. **Retain**: the new case, incorporating both problems and solutions, is stored in the database.

Every feature extracted from incident reports is awarded an equal weight. Every feature in a new incident is compared with the corresponding feature in each of the other incidents. If the features match, a score of 1 is awarded. If the features do not match, a score of 0 is awarded. A similarity score is calculated by:

1. finding the sum of the matching features; and
2. dividing this sum by the number of features contained in the incident, as in the formula below:

$$sim(T,S) = \frac{\sum_{i=1}^{n} f_i(T_i, S_i)}{n} \tag{5.1}$$

where T is a new incident, S is an existing incident, T_i is a feature of incident T, S_i is a feature of incident S, n is the number of the features in the incident T and S, $f_i \in \{0, 1\}$ is a function to match between T_i and S_i .

Then a threshold value is set up to determine whether or not the two incidents are matched.

In order to compute the similarity value between SDE metadata and service concepts, we extend the theory of the CBR algorithm to the context of metadata-concept matching. Here we design an ECBR algorithm to realize the objective of the metadata-concept similarity computation, which can be mathematically expressed as [4]:

$$sim(M,C) = \max_{SD_i \in M, CD_j \in C} \left(\sum_{t_{jh} \in CD_j} \frac{f(SD_i, t_{jh})}{l_{CD_j}} \right) \tag{5.2}$$

$$f(SD_i, t_{jh}) = \begin{cases} 1 \ if \ \ t_{jh} \in SD_i \\ 0 \ otherwise \end{cases} \tag{5.3}$$

where M is a SDE metadata, C is a service concept, SD_i is the value of a *serviceDescription* property of the metadata M, CD_j is the value of a *conceptDescription* property of the concept C, t_{jh} is a term that appears in the *conceptDescription* CD_j and l_{CDj} is the total number of terms that appears in the *conceptDescription* CD_j.

The principle objective of the ECBR model is to match the features of objects in order to find the maximum similarity between two objects [5]. In this research, the aim of the ECBR model aims is to compare the terms in the *serviceDescription* property(s) of a SDE metadata and the terms in the *conceptDescription* property(s) of a service concept, in order to find the maximum similarity between the two groups of properties, which is represented by Fig. 5.8.

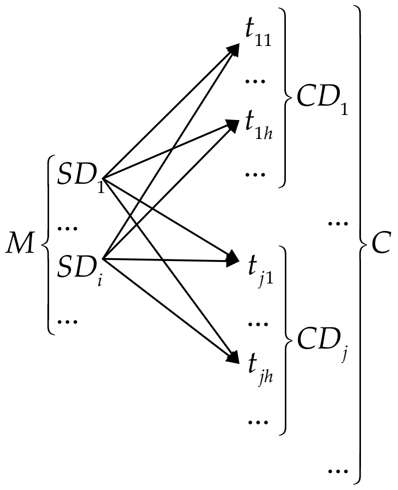


Figure 5.8: The matching process between SDE metadata and a service concept in the ECBR algorithm.

The advantages of the ECBR algorithm include: 1) it is simple to implement; 2) it does not need to generate index terms before matching, which saves pre-processing time; 3) it can also adapt to the frequent update of the ontologies, which often need the regenerating of index terms in most of index term-based algorithms; and 4) since the model is independent of index terms, it does not have the issue of index term dependency [5].

However, the ECBR algorithm also has a limitation, which emerged from the subsequent experiments (introduced in Chapter 10). The limitation is that the computing cost of the algorithm is relatively high, as a result of the term-term matching between the features of metadata and concepts. Since computing cost is a key factor in assessing the quality of a crawler and a search system, we seek to enhance the efficiency of the ECBR model. As a result, we design an IECBR algorithm. By means of introducing the theory of index terms into the ECBR algorithm, it is expected that the IECBR algorithm will be more efficient than the ECBR algorithm [6].

The principle of the IECBR algorithm is similar to that of the ECBR algorithm, which finds the highest coupling value between the *serviceDescription* property(s) of a SDE metadata and the *conceptDescription* property(s) of a service concept. The theoretical foundation is the belief that a metadata can be defined by its *serviceDescription* property(s), and in parallel, a concept can be defined by its *conceptDescription* property(s) [6]. Accordingly, the similarity value between a metadata and a concept can be determined by considering the maximum similarity value between the belonging *serviceDescription* property(s) and *conceptDescription* property(s). The major difference is that the IECBR algorithm has a pre-processing step which generates a list of index terms from the concrete concepts of ontologies from the Service Ontology Base, and it uses these index terms as an intermedium to match the metadata with the concepts [6].

The mathematical representation of the IECBR model is shown as:

First of all, a list of index terms k_t (t = 1, 2…w) are generated from all *conceptDescription* properties CD in the service concept. Each CD_j is associated with an array ($w_{1,j}$, $w_{2,j}$…$w_{w,j}$) where $w_{t,j}$ Є {0, 1} is the weight between k_t and CD_j, 1 indicates that k_t appears in CD_j and 0 indicates no. For each SDE metadata M, each of its *serviceDescription* property SD_i is associated with an array ($w_{1,t}$, $w_{2,t}$…$w_{w,t}$), where $w_{t,i}$ Є {0, 1} is the weight between k_t and SD_i, 1 indicates that k_t appears in CD_j and 0 indicates not.

The similarity value between a SDE metadata M and a service concept C and is obtained by Equation (4) and (5) as:

$$sim(M,C) = \max_{SD_i \in M, CD_j \in C} \left(\sum_{t_{jh} \in CD_j} \frac{f(SD_i, t_{jh})}{\sum CD_j} \right) \tag{5.4}$$

$$f(SD_i, t_{jh}) = \begin{cases} 1 \; if \; \exists \Delta \,|\, (\forall k_t, (g_t(t_{jh}) = g_t(\Delta)) \wedge (\Delta \in SD_i)) \\ 0 \; otherwise \end{cases} \tag{5.5}$$

where t_{jh} is a word involved in CD_j, $\sum CD_j$ is the sum of the array ($w_{1,j}$, $w_{2,j}$…$w_{t,j}$) associated with CD_j, Δ is the term involved in SD_i, and g_t is a function that returns a weight associated with k_t.

The primary advantage of the IECBR algorithm is that it divides the matching process into two steps: 1) generating a list of index terms; and 2) matching metadata and concept based on those index terms. The first step can be pre-processed before a SDE metadata is passed to the SDE Metadata Classifier, which partly saves the actual metadata-concept matching time.

The empirical evaluation of the ECBR and IECBR algorithm can be found in Chapter 10. Moreover, in the SDE metadata classification process, we need to find a proper threshold value for the computational results of the two algorithms, in order to determine the boundary whether or not each pairwise metadata and concept should be associated. In order to find the threshold value, we implement a series of experiments in Chapter 10.

5.7 Conclusion

In this chapter, we proposed a methodology for service information discovery, annotation and classification, which is represented by a Service Knowledge Base and a Semantic Crawler.

The Service Knowledge Base consists of a Service Ontology Base and an SDE Metadata Base. The former is used to store the service ontologies which are hierarchies of service concepts linked by concept-subconcept relations and each concept presents domain-specific service knowledge. The latter is used to store the SDE metadata that describes the features of service entities provided by service providers in the real world. Following that, we defined an extendable general service concept schema, in order to define the service concepts from all service domains in the Digital Ecosystems environment. Analogously, we defined an extendible general SDE schema to enable the possibility of describing the features of service entities form all Digital Ecosystem service domains. Moreover, by defining the object properties between the two schemas, we make it possible to classify the service ontology-based SDE metadata.

The Semantic Crawler consists of five agents and a repository. By means of a series of collaborations between the components of the Semantic Crawler and the components of the Service Knowledge Base, the objective of automatic service information discovery, classification and annotation is realized. Especially for service ontology-based SDE metadata classification, this is realized by exchanging the URIs between the relevant pairwise metadata and concepts. The relevance is determined by comparing the similarity value of each pairwise metadata and concept with a proper threshold value. If the similarity value is higher than the threshold value, the couple can be determined as relevant; otherwise not. The similarity computation is realized by designing two alternative algorithms – an ECBR algorithm and an IECBR algorithm, which are both based upon the thought of a CBR algorithm, which is to match the features between two objects. The empirical evaluation of the two algorithms and the discovery process of the proper threshold value can be found in Chapter 10.

The service information discovery, annotation and classification methodology realizes the objective of automatically discovering, semanticizing and structuring the service information in the Digital Ecosystems environment, which provides a semanticized and well-organized Service Knowledge Base for the further service retrieval. In the next chapter, based upon the Service Knowledge Base, we design a service retrieval methodology in order to resolve the ambiguous problems in service retrieval, and to allow service requesters to retrieve services with the assistance of service domain knowledge.

5.8 References

1. Cassidy, D., Carthy, J., Drummond, A., Dunnion, J., Sheppard, J.: The use of data mining in the design and implementation of an incident report retrieval system. Proceedings of the IEEE 2003 Systems and Information Engineering Design Symposium (SIEDS 2003). IEEE, Charlottesville, USA (2003) 13-18

2. Dong, H., Hussain, F.K., Chang, E.: Transport service ontology and its application in the field of semantic search. Proceedings of the 2008 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI 2008). IEEE, Beijing, China (2008) 820-824

3. Dong, H., Hussain, F.K., Chang, E.: A transport service ontology-based focused crawler. In: Hai, Z. (ed.): Proceedings of the fourth International Conference on Semantics, Knowledge and Grid (SKG '08). IEEE Computer Society, Beijing, China (2008) 49-56

4. Dong, H., Hussain, F.K., Chang, E.: A semantic crawler based on an extended CBR algorithm. In: Meersman, R., Tari, Z., Herrero, P. (eds.): On the Move to Meaningful Internet Systems: OTM 2008 Workshops. Springer-Verlag, Monterrey, Mexico (2008) 1084-1093

5. Dong, H., Hussain, F.K., Chang, E.: Focused crawling for automatic service discovery, annotation and classification in industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

6. Dong, H., Hussain, F.K., Chang, E.: A framework for discovering and classifying ubiquitous services in digital health ecosystems. Journal of Computer and System Sciences **In Press**

Chapter 6 – Service Retrieval Methodology

6.1 Introduction

In Chapter 5, we propose a service information discovery, annotation and classification methodology. A key element of this methodology is a Service Knowledge Base, which is comprised of a Service Ontology Base and a SDE Metadata Base. The former stores the knowledge regarding all Digital Ecosystem service domains represented by domain-specific service ontologies, and the latter stores descriptive information regarding service entities represented by SDE metadata. Moreover, the SDE metadata are organized by the service ontologies, in order to denote their domains. By means of this methodology, the service entities in the Digital Ecosystems environment can be discovered, annotated and semantically clarified.

As introduced previously, in the Digital Ecosystems environment, service requesters need to find the service providers who can provide the requested services. Since services in Digital Ecosystems are described and represented by entities in the Service Knowledge Base, the task of service retrieval from the Digital Ecosystems environment can be pre-digested into the task of service retrieval from the Service Knowledge Base, which reduces the labour cost of service requesters on seeking demanded services from the huge amount of information available on the Web.

In this chapter, we present a service retrieval methodology, with the purpose of assisting service requesters to precisely retrieve their requested service entities (SDE metadata) from the Service Knowledge Base.

Before this methodology is presented, we need to first study the user requirements in the scenario of service retrieval. We premise that there are two primary groups of service requesters as the potential users of service retrieval as follows [1]:

- The first group comprises those service requesters who do not have sufficient knowledge about the service domain to which their service requests belong. In this instance, their service queries may be ambiguous and may not necessarily reflect their service intentions accurately. This phenomenon often occurs when a service requester does not have previous experience with a service.

- The second group comprises those service requesters who have adequate knowledge about their service requests. In this instance, their service queries are able to precisely match the corresponding service domain knowledge. These service requesters usually have some experience in interacting with the requested services.

Based on the two aforementioned groups of users, we premise that each group of users has special requirements of the proposed service retrieval methodology as [1]:

- Service requesters without corresponding service domain knowledge with regard to their service queries require that the proposed methodology be able to assist them to denote and disambiguate their service queries with corresponding service domain knowledge, in order to precisely retrieve the requested services. Here, preciseness is considered as the most important factor over other factors regarding these users' requirements.

- Service requesters with service domain knowledge require that the proposed methodology be able to assist them to quickly find demanded services based on their clear queries. In this instance, retrieval time is the most important factor for these users.

In terms of the two different user requirements, we propose a service retrieval methodology which contains two different retrieval modules that cater for these two groups of service requesters.

6.2 System Architecture

The service retrieval methodology for the Digital Ecosystems environment is realized by cooperation between a Service Search Engine and the Service Knowledge Base introduced in Chapter 5. The service retrieval methodology is presented by its prototype – a Service Search Engine. The system architecture and system workflow of the Service Search Engine is displayed in Fig. 6.1.

The Service Search Engine contains two service retrieval modules for the two groups of service requesters introduced in Section 6.1, which are [1]:

- Generic Service Retrieval Module, which is designed for the service requesters who do not have domain knowledge with regard to their service queries; and

- Specific Service Retrieval Module, which is designed for the service requesters who have domain knowledge with regard to their service queries.

In the next two sections, we introduce the technical details regarding the two modules respectively. Conclusions are drawn in the final section.

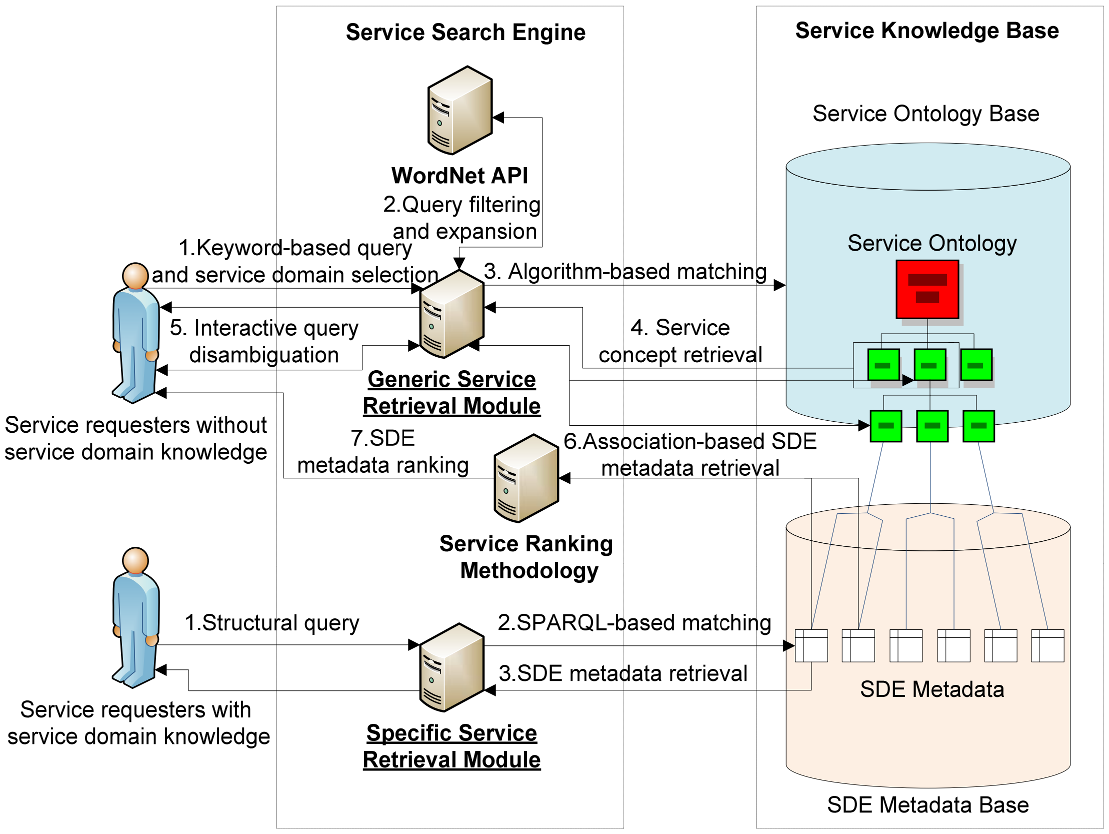

Service Search Engine
Service Knowledge Base
Service Ontology Base
WordNet API
2.Query filtering and expansion
1.Keyword-based query and service domain selection
3. Algorithm-based matching
Service Ontology
5. Interactive query disambiguation
4. Service concept retrieval
Generic Service Retrieval Module
Service requesters without service domain knowledge
7.SDE metadata ranking
6.Association-based SDE metadata retrieval
Service Ranking Methodology
1.Structural query
2.SPARQL-based matching
3.SDE metadata retrieval
SDE Metadata
Service requesters with service domain knowledge
Specific Service Retrieval Module
SDE Metadata Base

Figure 6.1: System architecture of the Service Search Engine

6.3 Generic Service Retrieval Module

In this section, we introduce the Generic Service Retrieval Module in detail. Since the service requesters lack sufficient knowledge regarding their service queries, these queries could contain ambiguities. Consequently, the task of this module is to help the users to disambiguate those queries and denote their query intentions with service domain knowledge, in order to precisely find the service entities (SDE metadata) which can best fit their query intentions. To this purpose, we propose the use of Human-Centered Computing (HCC) in the service retrieval methodology, by designing a series of interactions between service requesters and search interfaces. The next two sections introduce the workflow of the interactions and the mathematical models employed within the system.

6.3.1 System Workflow of the Generic Service Retrieval Module

The workflow of the Generic Service Retrieval Module is illustrated in Fig. 6.1 and consists of seven steps as follows [2]:

Step 1. Keyword-based query and service domain selection. A service requester enters a plain-text-based service query into the interface of the Service Search Engine. We premise that the service query consists of keywords rather than long sentences as in natural languages. This premise is based on a survey which indicates that the average query length for most of the Web search engines is around 2.3 words [4]. As a result, we do not need to specially employ natural language processing (NLP) programs to parse and analyse service queries. In addition, before the service requester sends the query, he/she needs to select the general service domain where the query belongs to. For example, "transport service domain" or "health service domain". This action will result in the selection of a corresponding ontology from the Service Ontology Base for the forthcoming query-concept matching.

Step 2. Query filtering and expansion. Instead of the NLP, we employ a WordNet API to implement the task of query processing and explanation. By sending all words of the service query to the WordNet API, for the valid words, the API will return the synonyms of the words; and, for the invalid words, the API will return error messages. Hence, within the query, the valid words will be expanded with their synonyms and the invalid words will be filtered.

Step 3. Algorithm-based matching. After receiving the processed query from the WordNet API, the search engine will match the query with each concept within the selected ontology in the Service Ontology Base. This query-concept matching is realized by an algorithm which is introduced in Section 6.3.2. The algorithm is able to compute the similarity value between the query and each concept.

Step 4. Service concept retrieval. The retrieved concepts for the query will be returned to the service requester and ranked based on their similarity values.

Step 5. Interactive query disambiguation. The query disambiguation process is realized by the Human-Computer Interaction (HCI) approach, which consists of the subsequent interactions:

Step 5.1. The service requester needs to choose a service concept from the returned list in order to find a concept that can best denote his/her query intention.

Step 5.2. If the selected concept is an abstract concept in the ontology, it will unfold all its subconcepts from the ontology, for the further denotation (concept selection). Step 5.1 and 5.2 are an iterative process that continues until the service requester selects a concrete concept.

Step 6. Association-based SDE metadata retrieval. If the service requester chooses a concrete service concept, the search engine will return all its associated SDE metadata from the SDE Metadata Base by retrieving the *linkedMetadata* property of the concept.

Step 7. SDE Metadata ranking. The retrieved SDE metadata will be ranked by a service ranking methodology and displayed to the service requester. The service ranking methodology is introduced in Chapter 8.

6.3.2 Query-Concept Matching Algorithm

In order to realize the function of computing the similarity value between a query and each concept from a service ontology, we design two alternative query-concept matching algorithms, which are based on the ECBR algorithm and the IECBR algorithm respectively, introduced in Chapter 5. In this section, we introduce the two algorithms in detail.

First of all, the principle of the ECBR-based query-concept matching algorithm is to compare 1) the remaining keywords in a query after the processing of the WordNet API and the synonyms of these keywords with 2) the *conceptDescription* property(s) of a service concept. For the value of each comparison between a query and a *conceptDescription* property, if a keyword of the query is contained in it, a value 1 will be awarded; and, if a synonym of a keyword of the query is contained in it, a value 0.5 will be awarded. After the comparison process, the sum of the values for the comparison of all terms in the *conceptDescription* property will be normalized by the total number of the terms in the property; thus, its value should be between 0 and 1. Since a concept may have more than one *conceptDescription* properties, the maximum value among them is the similarity value between the query and the concept. The ECBR-based query-concept matching algorithm for computing the similarity value between a processed query Q and a concept C is shown mathematically below [3]:

$$sim(Q,C) = \max_{CD_j \in C} \left(\sum_{t_{jh} \in CD_j} \frac{f(Q,t_{jh}) + f_s(S,t_{jh})}{l_{CD_j}} \right) \tag{6.1}$$

with

$$f(Q,t_{jh}) = \begin{cases} 1 & if\ t_{jh} \in Q \\ 0 & otherwise \end{cases} \tag{6.2}$$

$$f_s(S,t_{jh}) = \begin{cases} 0.5 & if \ t_{jh} \in S \\ 0 & otherwise \end{cases} \tag{6.3}$$

where CD_j is a *conceptDescription* property of the concept C, t_{jh} is a term that occurs within CD_j, S is a group of synonyms of the terms that occur in query Q, and l_{CDj} is the number of terms appearing in CD_j.

Afterwards, the IECBR-based query-concept matching algorithm divides the matching process into two steps as follows [2]:

Step 1. Generating an index term list before receiving a query, by obtaining all terms from *conceptDescription* properties of all concepts from the Service Ontology Base; and then each *conceptDescription* property is assigned with an array, in which each element corresponds to the each element of the index term list, and the weight of each element is 1 if the corresponding index term occurs in the *conceptDescription* property, or 0 otherwise.

Step 2. Once it receives a processed query from the WordNet API, the search engine will generate an array for the query, based on the index term list. Each element of the array is assigned a weight {0, 0.5, 1} where 1 indicates the corresponding index term occurs in the query, 0.5 indicates that the corresponding index term occurs in the synonyms of the query words and 0 indicates not. The array of the query is then compared with the arrays of the *conceptDescription* properties of each concept from a selected ontology. The maximum value between the query and any *conceptDescription* properties of a concept is considered as the similarity value between the query and the concept.

The IECBR-based query-concept matching algorithm can be mathematically expressed as [2]:

$$sim(Q,C) = \max_{CD_j \in C} \left(\sum_{t_{jh} \in CD_j} \frac{f(Q,t_{jh}) + f_s(S,t_{jh})}{\sum CD_j} \right) \tag{6.4}$$

with

$$f(Q,t_{jh}) = \begin{cases} 1 & if \ \exists \Delta \mid (\forall k_t, (g_t(t_{jh}) = g_t(\Delta)) \wedge (\Delta \in Q)) \\ 0 & otherwise \end{cases} \tag{6.5}$$

$$f_s(S,t_{jh}) = \begin{cases} 0.5 & if \ \exists \Delta \mid (\forall k_t, (g_t(t_{jh}) = g_t(\Delta)) \wedge (\Delta \in S)) \\ 0 & otherwise \end{cases} \tag{6.6}$$

where Q is a processed query, C is a concept, CD_j is a *conceptDescription* property of concept C, t_{jh} is a term involved in CD_j, $\sum CD_j$ is the sum of the weights within the array associated with CD_j, Δ is the term involved in CD_j, k_t is an index term, and g_t is a function that returns a weight associated with k_t.

Since a service ontology normally consists of hundreds of service concepts, if all the concepts from an ontology are retrieved and displayed to a service requester, regardless of whether or not they are relevant to his/her query intention, the service requester will easily become confused by receiving too many results. Therefore, analogous to the semantic crawler, we need to configure a threshold value to filter irrelevant service concepts for a service query based on the similarity values between queries and concepts. Moreover, at the threshold value, the Service Search Engine should be at its optimal performance. In Chapter 10, we reveal the details on how to discover the optimal threshold values for the two query-concept matching algorithms, and the details regarding the evaluation of the two algorithms.

6.4 Specific Service Retrieval Module

For the service requesters who have domain knowledge regarding their service queries, we design a Specific Service Retrieval Module, in order to help them to quickly retrieve SDE metadata from the Service Knowledge Base. Since the SDE metadata are annotated with ontology mark-up languages, it is possible to make use of SPARQL-based queries, which retrieve items by querying their attributes, in order to assist the service requesters to omit the query denotation/disambiguation process and directly search SDE metadata from the Service Metadata Base. This search style is based on the premise that service requesters have some knowledge about a SDE metadata, such as its service provider's name, its service descriptions and so on.

The workflow of the Specific Service Retrieval Module is as follows [1]:

Step 1. Structural query. A service requester enters a structural query into the search interface. A structural query is realized by describing restrictions for some attributes of a desired SDE metadata.

Step 2. SPARQL-based matching. The search engine annotates the structural query with SPARQL code, and directly retrieve metadata by the SPARQL-based query. Fig. 6.2 shows an example of the SPARQL-based query.

Step 3. SDE metadata retrieval. Having obtained the metadata that can meet the restrictions, the search engine then returns the metadata to the service requester.

```
Prefix meta: <http://www.owl-ontologies.com/Ontology1204779700.owl#>
SELECT ?MetadataName ?ServDesc ?Name ?Profile ?Address ?Phone
?Fax ?Email ?Website
WHERE
{
?SDEMetadata meta:SDEName ?MetadataName.
?SDEMetadata meta:serviceDescription ?ServDesc.
?SDEMetadata meta:isProvidedby ?ServiceProvider.
?ServiceProvider meta: providerName ?Name.
?ServiceProvider meta: providerProfile ?Profile.
?ServiceProvider meta:address ?Address.
?ServiceProvider meta:contactDetails ?Phone.
?ServiceProvider meta:contactDetails ?Fax.
?ServiceProvider meta:contactDetails ?Email.
?ServiceProvider meta:contactDetails ?Website.
FILTER regex (?MetadataName, var1, "i")
FILTER regex (?ServDesc, var2, "i")
FILTER regex (?Name, var3, "i")
FILTER regex (?Profile, var4, "i")
FILTER regex (?Address, var5, "i")
FILTER regex (?Phone, var6, "i")
FILTER regex (?Fax, var7, "i")
FILTER regex (?Email, var8, "i")
FILTER regex (?Website, var9, "i")
}
```

Figure 6.2: Example of the SPARQL-based query for the Specific Service Retrieval Module

We use the simulation method to evaluate the functions of the Specific Service Retrieval Module. The details of the evaluation can be found in Chapter 10.

6.5 Conclusion

In this chapter, we proposed a service retrieval methodology which is realized by a Service Search Engine and the Service Knowledge Base. The methodology contains two retrieval modules for two groups of service requesters as follows:

- For the service requesters who do not have domain knowledge regarding their service queries, a General Service Retrieval Module enables the use of service ontologies from the Service Ontology Base to assist these service requesters to disambiguate or precisely denote their service queries. By means of a series of HCI-based activities between service requesters and service ontologies, service requesters can eventually find the service concepts which can best describe their query intentions. Subsequently, the SDE metadata associated with these service concepts are retrieved from the Service Knowledge Base and ranked based on a service ranking methodology (introduce in Chapter 8). In addition, two alternative algorithms are employed for the initial matching of user queries with ontology concepts.

- For the service requesters who have knowledge regarding their service queries, a Specific Service Retrieval Module is designed to allow service requesters to quickly and directly search SDE metadata from the Service Metadata Base, by means of the SPARQL query language.

However, the two service retrieval modules omit a use scenario, which can be described as follows:

> For some service requesters who know nothing about their service queries, their initial service queries could be wrong (worse than ambiguous), which may lead to initial mismatching between the service intentions and service concepts. Moreover, owing to the lack of service domain knowledge, the interactions between these service requesters and service ontologies cannot actually resolve the issue of query ambiguities.

Consequently, in order to deal with the use scenario above, we propose a service concept recommendation methodology, which is introduced in Chapter 7.

6.6 References

1. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

2. Dong, H., Hussain, F.K., Chang, E.: Semantic service matchmaking for digital health ecosystems. Knowledge-Based Systems **Submitted**

3. Dong, H., Hussain, F.K., Chang, E.: A service search engine for the industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

4. Hawking, D.: Web search engines: Part 2. Computer **39** (2006) 88-90

Chapter 7 – Service Concept Recommendation Methodology

7.1 Introduction

In Chapter 6, we proposed a service retrieval methodology in order to disambiguate service queries. However, for the service requesters who do not have enough domain knowledge about their service queries, their initial query terms could be wrong, which may lead to a mismatch between service queries and service concepts. Although the service requesters can make use of human-centred selection to select other concepts from the Service Ontology Base in order to rectify the mismatching, as there are normally hundreds of concepts in an ontology, this adjustment could be difficult. Hence, in this chapter, we propose an automatic service concept recommendation methodology, in order to alleviate this problem.

The solution is based on the scenario that, by means of a series of interactions with the Service Search Engine, a service requester selects a concrete service concept; however, by observing the associated SDE metadata with the selected concept, the service requester finds that the concept does not match his/her query intention. In order to solve this problem, we premise that there is a certain amount of overlap between the selected concept and the concept which can represent the service requester's actual service request, owing to the impact of their understandings of service requests on the concept selection by a series of HCIs. Therefore, we design two semantic similarity models to automatically seek those concepts which are semantically similar to the selected concept, and to recommend these concepts to the service requester for further query disambiguation.

7.2 System Architecture

Fig. 7.1 depicts the system architecture and workflow of the prototype of the proposed service concept recommendation methodology – a Service Concept Recommendation System. The workflow of the methodology is comprised of the following four steps [1]:

Step 1. Concrete service concept selection. A service requester selects a concrete service concept by means of a series of interactions with the Service Search Engine.

Step 2. Service concept obtainment. The Service Concept Recommendation System obtains the selected concept from the Service Search Engine.

Step 3. Semantically similar concept retrieval. The Service Concept Recommendation System makes use of the semantic similarity models explained in this chapter to find the semantically similar concepts from the service ontology to which the selected concept belongs, based on the concept.

Step 4. Service concept recommendation. The semantically similar service concepts are displayed to the service requester and ranked according to their similarly values to the selected concept. When the service requester selects a concept from the ranking list, Step 1 to Step 4 will be repeated until he/she eventually finds a concept that best fits his/her query intention.

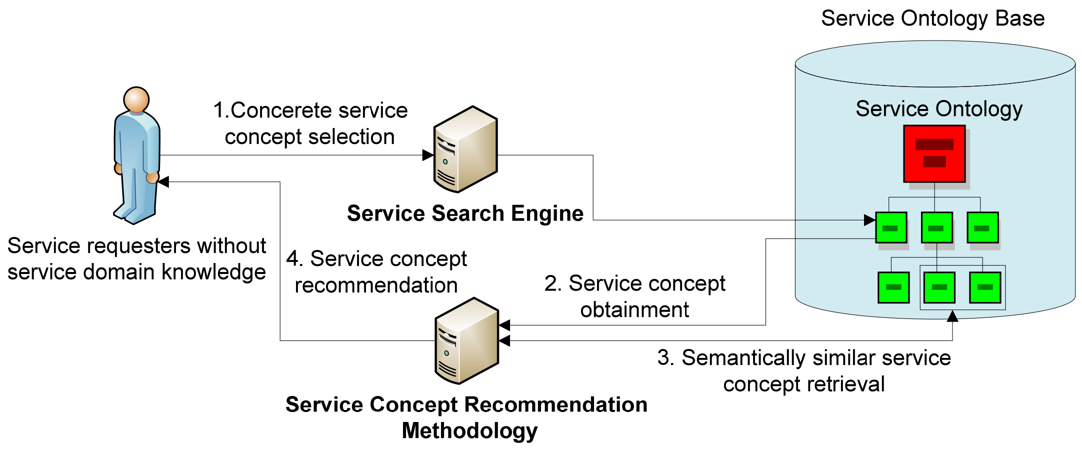

Service Ontology Base
Service Ontology
1.Concerete service concept selection
Service Search Engine
Service requesters without service domain knowledge
4. Service concept recommendation
2. Service concept obtainment
3. Semantically similar service concept retrieval
Service Concept Recommendation Methodology

Figure 7.1: System architecture and workflow of the Service Concept Recommendation System

Before we introduce the semantic similarity models, in order to address the issue of similarity computation based on content of relations and ontology concepts (introduced in Chapter 3), we design a conversion process which transforms an ontology to a lightweight ontology space. In Section 7.3 we introduce the ontology conversion process in detail. In Section 7.4 we present two alternative semantic similarity models. The conclusion to this chapter is drawn in the final section.

7.3 Converting Ontology to Lightweight Ontology Space

7.3.1 Lightweight ontology space

Before we introduce the ontology conversion process, first of all we provide a concept of lightweight ontology space, which includes two basic definitions as follows [2, 3]:

Definition 1. Pseudo-concept

We define a pseudo-concept ς for an ontology concept C, which can be represented as a tuple as follows:

$$\varsigma = \left\{ C, \left[\delta_i, \gamma_{\delta_i} \right], \left[o_j, \gamma_{o_j} \right], C_{o_j}^{x}, \lambda_{o_j}^{y} \right\} \tag{7.1}$$

where in OWL-annotated Semantic Web documents, C is the name (or Uniform Resource Identifier (URI)) of the concept C, each [] is a property tuple including a property and its restriction (if available), δ_i (i = 1…n) is a datatype property(s) of the concept C, $\gamma_{\delta i}$ is a restriction (s) for the datatype property δ_i, o_j (j = 1…m) is an object property(s) of the concept C, γ_{oj} is a restriction(s) for the object property o_j, C_{oj}^{x} (x = 1…k) is a concept(s) related by the object property o_j, and λ_{oj}^{y} (y = 1…k-1) is a Boolean operation(s) between concepts C_{oj}^{x}.

The aim of defining the pseudo-concept is to encapsulate all properties, and restrictions and characteristics of the properties of a concept into a corpus for the concept, which makes it possible to assess the similarity between concepts based on the content of their pseudo-concepts.

Definition 2. Lightweight ontology space

Based on the definition of pseudo-concept, we define a lightweight ontology space as a space of pseudo-concepts, in which pseudo-concepts are linked only by *is-a* relations. An *is-a* relation is a generalization/specification relationship between an upper generic pseudo-concept and a lower specific pseudo-concept. In OWL documents, the *is-a* relation is represented by *subClassOf*. The aim of constructing a pseudo-concept space is to simplify the complicated ontology structure and hence to construct a definitional network-like taxonomy. This taxonomy makes it possible to measure concept similarity based on the existing semantic similarity models.

7.3.2 Theorems for Ontology Conversion Process

In order to convert an ontology to a lightweight ontology space, we need a conversion process. It needs to be noted that the proposed ontology conversion process takes place only in OWL Lite or OWL DL-annotated Semantic Web documents. Additionally, from the definitions above, it can be observed that the conversion process concerns only the schema (concept) level but not the instance level, because the information of instances is specialized to some degree and cannot completely represent belonged concepts. Considering the complexity and flexibility in defining restrictions and characteristics of object properties and datatype properties, a set of theorems, aligned with the conversion process, needs to be defined. The theorems can be divided into six categories in accordance with the components of a pseudo-concept, which are the theorems regarding the conversion of concepts, datatype properties, object properties, property restrictions, property characteristics and Boolean operations. In the rest of this section, we introduce and illustrate these theorems based on the six divisions [2].

Theorem 7.1. If C is the name (URI) of a concept, then C is a component of its pseudo-concept [2].

For example, for the concept C_1 shown in Fig. 7.2, its pseudo-concept $ς_1 = \{C_1\}$

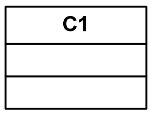


Figure 7.2: Example of an ontology concept

Theorem 7.2.1. If C is the name (URI) of a concept, and δ is a datatype property of C, then δ is a component of its pseudo-concept [2].

For example, the concept C_1 shown in Fig. 7.3, has a datatype property δ. According to Theorem 2.1, its pseudo-concept $ς_1 = \{C_1, δ\}$.

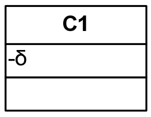


Figure 7.3: Example of an ontology concept with a datatype property

Theorem 7.2.2. If C_1 is the name (URI) of a concept, δ is a datatype property of C_1, and C_2 is the name (URI) of a subclass of C_1, then δ is a component of the pseudo-concept of C_2 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.4, C_1 has a datatype property δ, and C_2 is a subclass of C_1. According to Theorem 2.2, the pseudo-concept $ς_2$ for C_2 is a tuple that can be expressed as $\{C_2, δ\}$

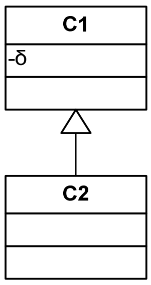


Figure 7.4: Example of an inherited ontology concept with a datatype property

Theorem 7.3.1. If C_1 is the name (URI) of a concept, o is an object property of C_1, and C_2 is the name (URI) of a concept that relates to C_1 through o, then o and C_2 are the components of the pseudo-concept of C_1 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.5, C_1 has an object property o which connects C_1 to C_2. According to Theorem 3.1, the pseudo-concept ς_1 for C_1 is a tuple that can be expressed as $\{C_1, o, C_2\}$.

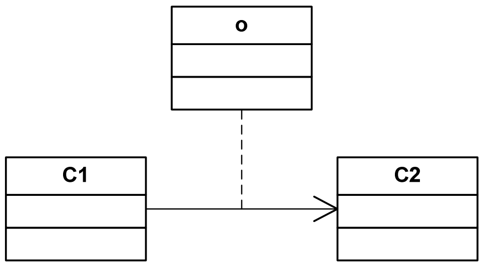


Figure 7.5: Example of an ontology concept with an object property

Theorem 7.3.2. If C_1 is the name (URI) of a concept, o is an object property of C_1, C_2 is the name (URI) of a concept that relates to C_1 through o, and C_3 is the name (URI) of a subclass of C_1, then o and C_2 are the components of the pseudo-concept of C_3 [2].

For example, for the concept C_1, C_2 and C_3 shown in Fig. 7.6, C_1 has an object property o which connects C_1 to C_2, and C_3 is a subclass of C_1. According to Theorem 3.2, the pseudo-concept ς_3 for C_3 is a tuple that can be expressed as $\{C_3, o, C_2\}$

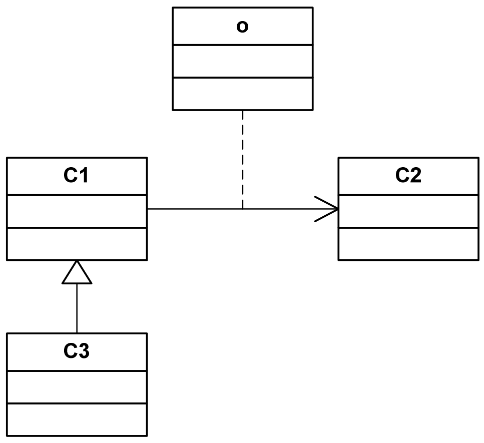


Figure 7.6: Example of an inherited ontology concept with an object property

Theorem 7.4.1. If C is the name (URI) of a concept, δ is a datatype property of C, and γ is a restriction for δ, then the tuple $[\delta, \gamma]$ is a component of the pseudo-concept of C [2].

For example, for the concept C_1 shown in Fig. 7.7, it has a datatype property δ, which has a value restriction *hasValue* and a cardinality restriction *minCardinality* 5. According to Theorem 4.1, its pseudo-concept ς_1 = {C_1, [δ, *hasValue minCardinality* 5]}.

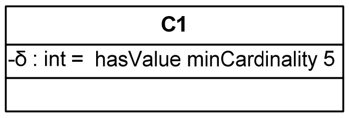


Figure 7.7: Example of an ontology concept with a restricted datatype property

Theorem 7.4.2. If C_1 is the name (URI) of a concept, o is an object property of C_1, C_2 is the name (URI) of a concept that relates to C_1 through o, and γ is a restriction for the datatype property o, then the tuple $[o, \gamma]$ is a component of the pseudo-concept of C_1 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.8, C_1 has an object property o which connects C_1 to C_2, and o has a property restriction *someValuesFrom* and a cardinality restriction *cardinality* 1. According to Theorem 3.2, the pseudo-concept ς_1 for C_1 is a tuple that can be expressed as {C_1, [o, *someValuesFrom*], C_2, [o, *cardinality* 1], C_2}.

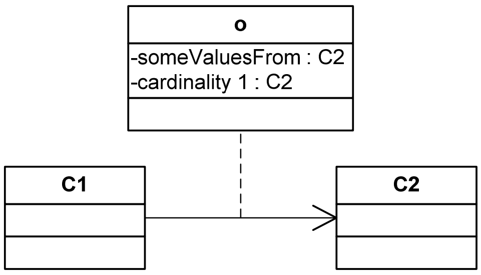


Figure 7.8: Example of an ontology concept with a restricted object property

Theorem 7.5.1. If C is the name (URI) of a concept, and δ is a functional datatype property of C, then the tuple [δ, *cardinality* 1] is a component of the pseudo-concept of C [2].

For example, for the concept C_1 shown in Fig. 7.9, it has a functional datatype property δ. According to Theorem 2.1, its pseudo-concept $ς_1$ = {C_1, [δ, *cardinality* 1]}.

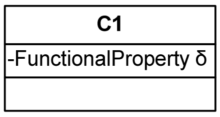


Figure 7.9: Example of an ontology concept with a functional datatype property

Theorem 7.5.2. If C_1 is the name (URI) of a concept, o is a functional object property of C_1, and C_2 is the name (URI) of a concept that relates to C_1 through o, then the tuple [o, *cardinality* 1] is the component of the pseudo-concept of C_1 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.10, C_1 has a functional object property o which connects C_1 to C_2. According to Theorem 5.2, the pseudo-concept $ς_1$ for C_1 is a tuple that can be expressed as {C_1, [o, *cardinality* 1], C_2}.

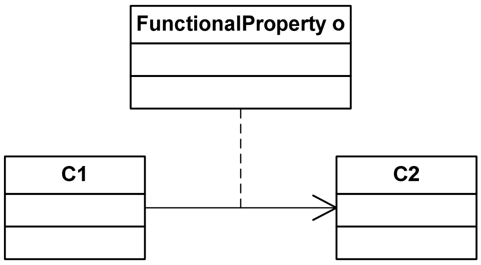


Figure 7.10: Example of an ontology concept with a functional object property

Theorem 7.5.3. If C_1 is the name (URI) of a concept, o is a transitive object property of C_1, C_2 is the name (URI) of a concept that relates to C_1 through o, and C_3 is the name

(URI) of a concept that relates to C_2 through o, then o, C_2 and C_3 are the components of the pseudo-concept of C_1 [2].

For example, for the concept C_1, C_2 and C_3 shown in Fig. 7.11, C_1 has a transitive object property o which connects C_1 to C_2, and C_2 has o which connects C_2 to C_3. According to Theorem 5.3, the pseudo-concept ς_1 for C_1 is a tuple that can be expressed as {C_1, o, C_2, o, C_3}.

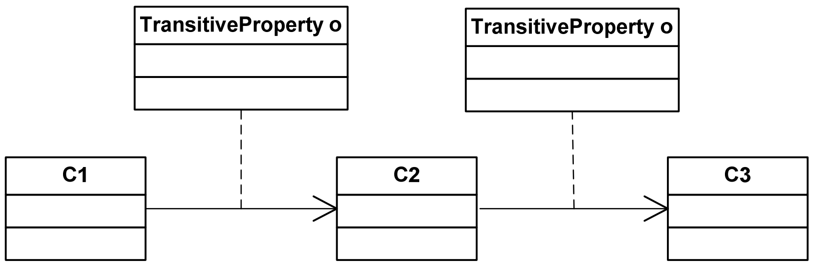


Figure 7.11: Example of ontology concepts with a transitive object property

Theorem 7.5.4. If C_1 is the name (URI) of a concept, o is a symmetric object property of C_1, and C_2 is the name (URI) of a concept that relates to C_1 through o, then o and C_2 are the components of the pseudo-concept of C_1, and o and C_1 are the components of the pseudo-concept of C_2 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.12, C_1 has a symmetric object property o which connects C_1 to C_2. According to Theorem 5.4, the pseudo-concept ς_1 for C_1 is a tuple that can be expressed as {C_1, o, C_2}, and the pseudo-concept ς_2 for C_2 is a tuple that can be expressed as {C_2, o, C_1}.

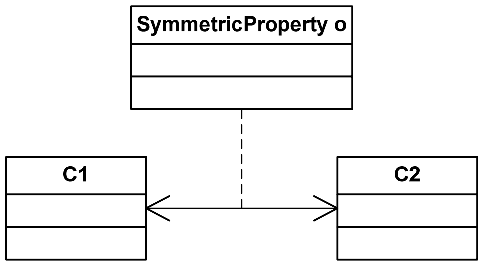


Figure 7.12: Example of ontology concepts with a symmetric object property

Theorem 7.5.5. If C_1 is the name (URI) of a concept, o1 is an inverse functional object property of C_1, C_2 is the name (URI) of a concept that relates to C_1 through o_1, and o_2 is the inverse property of o_1, then the tuple [o_2, *cardinality* 1] is the component of the pseudo-concept of C_2 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.13, C_1 has an inverse functional object property o_1 which connects C_1 to C_2, and o_2 is the inverse property of o_1. According to Theorem 5.5, the pseudo-concept ς_2 for C_2 is a tuple that can be expressed as {C_2, [o_2, *cardinality* 1], C_1}.

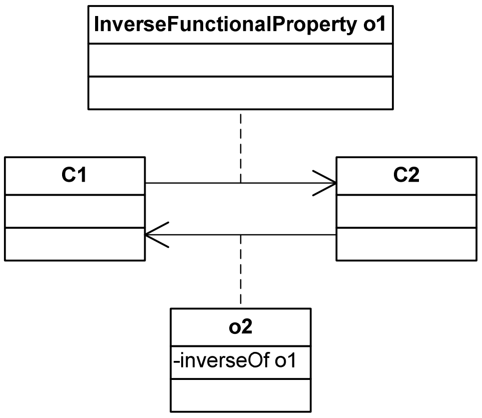


Figure 7.13: Example of ontology concepts with an inverse functional object property

Theorem 7.6.1. If C_1 is the name (URI) of a concept, o is an object property of C_1, C_2 and C_3 are the names (URI) of concepts that relate to C_1 through o, and λ is a Boolean operation (*unionOf* or *intersectionOf*) between C_2 and C_3 for o, then o, C_2, λ and C_3 are the components of the pseudo-concept of C_1 [2].

For example, for the concept C_1, C_2 and C_3 shown in Fig. 7.14, C_1 has an object property o which connects C_1 to C_2 and C_3, and C_2 and C_3 are connected with *intersectionOf*. According to Theorem 6.1, the pseudo-concept ς_1 for C_1 is a tuple that can be expressed as {C_1, o, C_2, *intersectionOf*, C_3}.

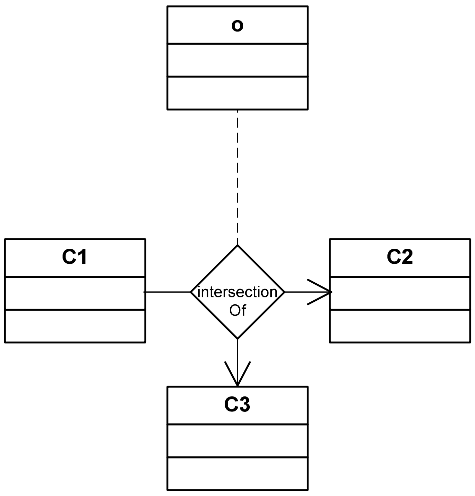


Figure 7.14: Example of ontology concepts connected with a Boolean operation (*unionOf* or *intersectionOf*)

Theorem 7.6.2. If C_1 is the name (URI) of a concept, o is an object property of C_1, and C_2 is the name (URI) of a concept that relates to C_1 through the complement of o, then *complementOf*C_2 is a component of the pseudo-concept of C_1 [2].

For example, for the concept C_1 and C_2 shown in Fig. 7.15, C_1 has an object property o which connects C_1 to the complement of C_2. According to Theorem 6.2, the pseudo-concept ς_1 for C_1 is a tuple that can be expressed as {C1, o, *complementOf*C_2}.

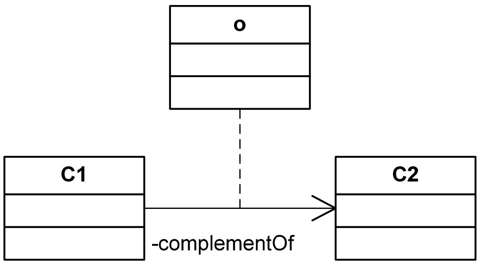


Figure 7.15: Example of ontology concepts connected with a *complementOf* operation

In order to illustrate the ontology conversion theorems in detail, we create a sample ontology – a pizza ontology shown in Fig. 7.16, and convert it to a lightweight ontology space in terms of the theorems.

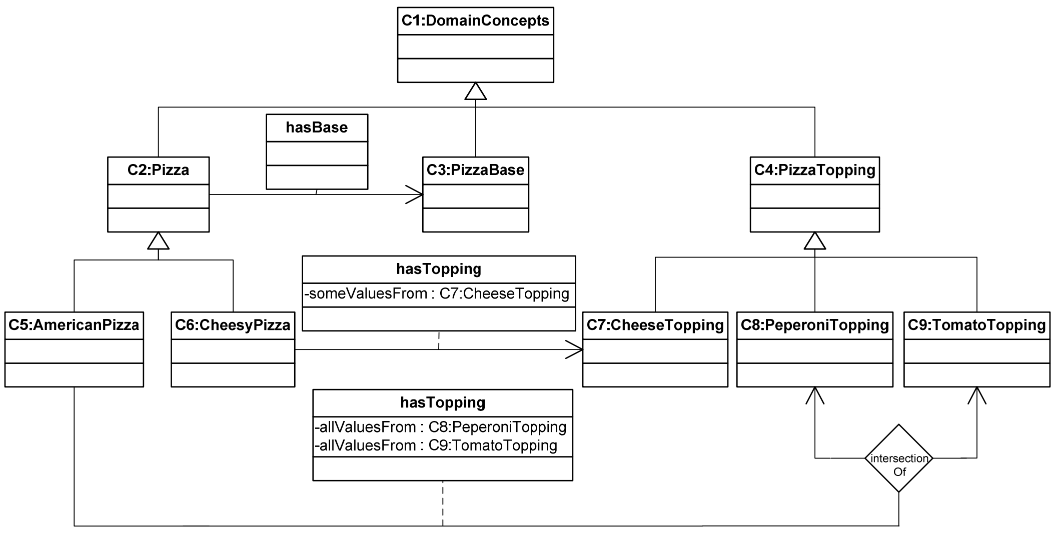


Figure 7.16: Ontology example – pizza ontology in UML

According to the theorems above, the lightweight ontology space for the pizza ontology can be represented as follows:

ς_1= {DomainConcepts}
ς_2= {Pizza, hasBase, PizzaBase, hasTopping, PizzaTopping}
ς_3= {PizzaBase}
ς_4= {PizzaTopping}
ς_5= {AmericanPizza, hasBase, PizzaBase, [hasTopping, allValuesFrom], PeproniTopping, *intersectionOf*, TomatoTopping }
ς_6= {CheesyPizza, hasBase, PizzaBase, [hasTopping, someValuesFrom], CheesyTopping}
ς_7= {CheesyTopping}
ς_8= {PeproniTopping}
ς_9= {TomatoTopping}

7.4 Two Hybrid Semantic Similarity Models

As described in the previous section, there are two advantages of the ontology conversion process as follows [1, 3]:

- Each ontology concept is converted to a pseudo-concept, which is a tuple of plain texts. Since the pseudo-concepts include almost all features of ontology concepts, it is possible to measure the similarity between concepts based on the content of pseudo-concepts.

- An ontology with a complicated structure can be simplified to a lightweight ontology by means of the conversion process. The taxonomic lightweight ontology enables the adoption of the existing semantic similarity models to measure the similarity between concepts.

In this section, we propose two alternative hybrid semantic similarity models, by assessing the concept similarity from the two perspectives above, which are described as follows:

- The first proposed hybrid model involves two sub-models. The first sub-model is used to measure concept similarity based on content of pseudo-concepts, by means of the cosine correlation approach. The second sub-model is used to measure concept similarity based on structure of lightweight ontology graphs, by means of an approach originating from the enhanced topic-based vector space model (eTVSM) [4]. The products of the two sub-models are two concept-concept matrixes. Then we integrate the two matrixes to obtain a new concept-concept matrix that indicates the extent of similarity between concepts.

- This second hybrid model integrates the pseudo-concept-based semantic similarity sub-model which is the same as its counterpart in the first hybrid model, and a lightweight-ontology-structure-based semantic similarity sub-model which is evolved from Resnik's semantic similarity model, which takes into account the information content of concepts [1, 3]. Analogously, the products of the two models are two concept-concept matrixes, and the eventual product is the combination of the two matrixes.

Therefore, it can be observed that there are three sub-models involved in the two hybrid semantic similarity models, which are:

- a pseudo-concept-based semantic similarity sub-model, which is introduced in Section 7.4.1,

- a lightweight-ontology-structure-based semantic similarity sub-model that originates from the eTVSM, which is depicted in Section 7.4.2, and

- a lightweight-ontology-structure-based semantic similarity sub-model that originates from Resnik's semantic similarity model, which is described in Section 7.4.3.

The two hybrid models are a combination of the first and the second sub-models and the first and the third sub-models respectively, which are discussed in Section 7.4.4.

7.4.1 Pseudo-Concept-based Semantic Similarity Sub-Model

In the information retrieval field, in order to measure the similarity between two corpora, the usual method is to use cosine correlation, which can be mathematically expressed as follows:

$$sim_{\cos\theta}(x,y)=\frac{\vec{x}\cdot\vec{y}}{\|\vec{x}\|\ \|\vec{y}\|} \tag{7.2}$$

where each corpus can be represented by a vector in which each dimension corresponds to a separate term, and the weight of each term in the vector can be obtained by the TF-IDF scheme.

In this research, in order to measure the similarity between two pseudo-concepts, we adopt the cosine correlation aligned with the pseudo-concept model displayed in Equation (7.1). The pseudo-concept model has several special features as follows [2]:

- Each component is separated by a comma and is viewed as a basic unit for the measure. For example, in the property tuple [o, cardinality 1], cardinality 1 is seen as a whole for the measure.

- The property tuple have some features as follows:
    - Each property tuple contains no more than two items, which are a property and a restriction (if necessary).
    - The weights of the terms occurring in each property tuple should be averaged, as a property tuple should be treated as same as other single items in a pseudo-concept tuple in the measure. For example, in the tuple [o, *someValuesFrom*], if the TF-IDF weight of o is 0.56 and it of *someValuesFrom* is 0.44, then their actual weights should be 0.28 and 0.22, as the average weight of the tuple is 0.5.
    - In each tuple, a property takes priority over its affiliated restriction in the measure, since the restriction is a modifier of the property. In other words, if there are two property tuples whose properties are different and their restrictions are the same, then there is no similarity between the two property tuples. For example, a pseudo-concept ς_1 has a tuple [o_1, *someValuesFrom*] and another pseudo-concept ς_2 has a tuple [o_2, *someValuesFrom*], the similarity value between the two tuples is 0 as $o_1 \neq o_2$.

In accordance with the features of the pseudo-concept model, we design an enhanced cosine correlation model to implement the similarity measure based on content of pseudo-concepts [2], which is displayed in Fig. 7.17.

Input: A list of pseudo-concepts $\varsigma = (\varsigma_1, \varsigma_2 \ldots \varsigma_m)$.
Output: A matrix P where each element P_{ij} is the similarity value between pseudo-concept ς_i and ς_j.
Algorithm

```
1. for i = 1 to m
2.    Read $\varsigma_i$;
3.    Generate an array of index term T = $(t_1, t_2 \ldots t_n)$;
4.    Put all the items in the tuples of $\varsigma_i$ into an array $\Theta_i$;
5. end for
6. for i = 1 to m
7.    Set l to the number of items in $\Theta_i$;
8.    for k = 1 to l
9.       for j = 1 to n
10.          if $\Theta_{i,k} = t_j$ then
11.             Put j into an array $\Delta_i$;
12.          end if
13.       end for
14.    end for
15. end for
```

```
16. for i = 1 to m
17.    for j = 1 to n
18.       Set $w_{ij}$ to the TF-IDF weight of $t_j$ in $\varsigma_i$;
19.       Set l to the number of items in $\Theta_i$;
20.       for k = 1 to l
21.          if j = $\Theta_{i,k}$ then
22.             $w_{ij} = 0.5 \times w_{ij}$;
23.          end if
24.       end for
25.       put $w_{ij}$ into $\vec{\varsigma}_i$;
26.    end for
27.    Normalize $\vec{\varsigma}_i$ by $|\vec{\varsigma}_i| = 1$;
28. end for
29. for i = 1 to m
30.    for j = 1 to m
31.       for k = 1 to n
32.          Set a to the number of items in $\Delta_i$;
33.          Set b to the number of items in $\Delta_j$;
34.          for u = 1 to a
35.             for v = 1 to b
36.                if k = $\Delta_{i,u}$ and u%2 = 0 then
37.                   if k = $\Delta_{j,v}$ and v%2 = 0 then
38.                      if $w_{i,\Delta_{i,u-1}} \times w_{j,\Delta_{j,v-1}} = 0$ then
39.                         $w_{ik} \times w_{jk} = 0$;
40.                      end if
41.                   end if
42.                end if
43.             end for
44.          end for
45.          $P_{ij} = P_{ij} + w_{ik} \times w_{jk}$;
46.       end for
47.    end for
48. end for
```

Figure 7.17: Pseudo-code of the pseudo-concept-based semantic similarity model

The result of the pseudo-concept-based semantic similarity model for the pizza ontology in Fig. 7.16 can be found in Table 7.1.

Table 7.1: Result of the pseudo-concept-based semantic similarity model for the pizza ontology.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
c_1	1	0	0	0	0	0	0	0	0
c_2	0	1	0.3135	0	0.1707	0.2201	0	0	0
c_3	0	0.3135	1	0	0.1921	0.2476	0	0	0
c_4	0	0	0	1	0	0	0	0	0
c_5	0	0.1707	0.1921	0	1	0.1758	0	0.3563	0.3563
c_6	0	0.2201	0.2476	0	0.1758	1	0.4592	0	0
c_7	0	0	0	0	0	0.4592	1	0	0
c_8	0	0	0	0	0.3563	0	0	1	0
c_9	0	0	0	0	0.3563	0	0	0	1

7.4.2 Lightweight-Ontology-Structure-based Semantic Similarity Model (eTVSM)

As mentioned earlier, the structure-based approach originates from the topic similarity measure model for the topic map environment [4]. In our model, we employ this method in the environment of lightweight ontologies. As lightweight ontologies have only one type of relationship, the weights of relations can be viewed as equal and the issue of relation weights can be ignored in the measurement process. The process of computing the extent of concept similarity can be divided into two stages: 1) determining the pseudo-concept vectors based on a lightweight ontology structure; and 2) obtaining a concept similarity matrix by means of the scalar product of the pseudo-concept vectors. The operational vector space dimensionality is specified by the number of pseudo-concepts in a lightweight ontology [3].

Let m be the number of pseudo-concepts in a lightweight ontology, and a set of pseudo-concepts can be represented as $\Theta = \{\varsigma_1 \ldots \varsigma_m\}$. In order to represent the lightweight ontology structure, we can use $G(\varsigma_i)$ to represent the generic concept of a pseudo-concept ς_i in an *is-a* relation. Returning to the ontology example from Fig. 7.15, the structure of the lightweight ontology can be represented as follows [3]:

$G(\varsigma_1) = \{\}$
$G(\varsigma_2) = \{\varsigma_1\}$
$G(\varsigma_3) = \{\varsigma_1\}$
$G(\varsigma_4) = \{\varsigma_1\}$
$G(\varsigma_5) = \{\varsigma_2\}$
$G(\varsigma_6) = \{\varsigma_2\}$
$G(\varsigma_7) = \{\varsigma_4\}$
$G(\varsigma_8) = \{\varsigma_4\}$

$G(\varsigma_9) = \{\varsigma_4\}$

Since it is well-known that the *is-a* relation is transitive in ontologies, here we use $G^*(\varsigma_i)$ to represent all the generic concepts of a pseudo-concept ς_i. Again, returning to the ontology example from Fig. 7.15, the structure of lightweight ontology can be represented as follows [3]:

$G^*(\varsigma_1) = \{\}$
$G^*(\varsigma_2) = \{\varsigma_1\}$
$G^*(\varsigma_3) = \{\varsigma_1\}$
$G^*(\varsigma_4) = \{\varsigma_1\}$
$G^*(\varsigma_5) = \{\varsigma_1, \varsigma_2\}$
$G^*(\varsigma_6) = \{\varsigma_1, \varsigma_2\}$
$G^*(\varsigma_7) = \{\varsigma_1, \varsigma_3\}$
$G^*(\varsigma_8) = \{\varsigma_1, \varsigma_3\}$
$G^*(\varsigma_9) = \{\varsigma_1, \varsigma_3\}$

We use Θ_s to represent the set of specific concepts that are not generic concepts of any pseudo-concepts in a lightweight ontology. In our ontology example, the specific concept set comprises [3]:

$\Theta_s = \{\varsigma_5, \varsigma_6, \varsigma_7, \varsigma_8, \varsigma_9\}$

On the other hand, the complement of Θ_s is the set of all generic concepts, which can be represented as Θ_g. The generic concept set in our ontology example consists of [3]:

$\Theta_g = \{\varsigma_1, \varsigma_2, \varsigma_3, \varsigma_4\}$

As mentioned previously, each pseudo-concept is assigned a vector with the dimensions relating to all pseudo-concepts in a lightweight ontology. The approach of obtaining vectors can be divided into two steps: 1) obtaining vectors for the specific pseudo-concept set; and 2) obtaining vectors for the generic pseudo-concept set [3].

First, we employ Equation (7.3) and Equation (7.4) to obtain vectors for the specific pseudo-concept set. We assign the same weight to the dimensions of a vector $\vec{\varsigma}_i$ that have counterparts in its generic concept set $G^*(\varsigma_i)$ and itself. The heuristics behind this can be found in [4].

$$\forall \varsigma_i \in \Theta_s : \vec{\varsigma}_i = (\varsigma_{i,1} \cdots \varsigma_{i,m}) \tag{7.3}$$

with

$$\varsigma_{i,k} = \begin{cases} 1 & \text{if } \varsigma_k \in G^*(\varsigma_i) \vee i = k \\ 0 & \text{else} \end{cases} \tag{7.4}$$

Once a specific pseudo-concept vector has been obtained, we normalize the vector length to 1 by making use of the Equation (7.5), in order to make the weight of each pseudo-concept in the vector dependent on the number of generic concepts. The normalization also benefits the angle measure between two vectors [3].

$$\forall \varsigma_i \in \Theta_s : |\vec{\varsigma}_i| = 1 \qquad (7.5)$$

Second, the generic pseudo-concept vector can be obtained by the sum of all its specific concepts related by the direct *is-a* relations as shown in Equation (7.6). Similar to Equation (7.5), the length of generic pseudo-concept vector needs to be normalized to 1 as shown in Equation (7.7). This is a recursive process whereby the lower level generic pseudo-concept vectors are obtained first by gaining all their specific concepts from Θ_s and then by normalizing the sum. Subsequently, the upper level generic pseudo-concept vectors are obtained by gaining all its specific concepts from the lower level generic pseudo-concept set and Θ_s and by normalizing the sum [3].

$$\forall \varsigma_i \in \Theta_g : \vec{\varsigma}_i = \sum_{\varsigma_k \in \Theta : \varsigma_i \in G(\varsigma_k)} \vec{\varsigma}_k \qquad (7.6)$$

with

$$\forall \varsigma_i \in \Theta_g : |\vec{\varsigma}_i| = 1 \qquad (7.7)$$

Further, we provide pseudo-concept vectors for concepts from our pizza ontology from Fig. 7.15.

$\vec{\varsigma}_1 = (0.6579, 0.3253, 0.5111, 0.3346, 0.1616, 0.1616, 0.1115, 0.1115, 0.1115)$

$\vec{\varsigma}_2 = (0.6325, 0.6325, 0, 0, 0.3162, 0.3162, 0, 0, 0)$

$\vec{\varsigma}_3 = (0, 0, 1, 0, 0, 0, 0, 0, 0)$

$\vec{\varsigma}_4 = (0.6547, 0, 0, 0.6547, 0, 0, 0.2182, 0.2182, 0.2182)$

$\vec{\varsigma}_5 = (0.5774, 0.5774, 0, 0, 0.5774, 0, 0, 0, 0)$

$\vec{\varsigma}_6 = (0.5774, 0.5774, 0, 0, 0, 0.5774, 0, 0, 0)$

$\vec{\varsigma}_7 = (0.5774, 0, 0, 0.5774, 0, 0, 0.5774, 0, 0)$

$\vec{\varsigma}_8 = (0.5774, 0, 0, 0.5774, 0, 0, 0, 0.5774, 0)$

$\vec{\varsigma}_9 = (0.5774, 0, 0, 0.5774, 0, 0, 0, 0, 0.5774)$

Once we have all the pseudo-concept vectors, we can obtain the pseudo-concept similarity matrix L by the scalar product of arbitrary pairs of vectors. Since one pseudo-concept corresponds to one concept, matrix L is also the assembly of similarity values between all corresponding pairs of concepts from an ontology. Meanwhile, the similarity value between two concepts can also be viewed as the cosine of the angle between the two corresponding pseudo-concept vectors. The extent of similarity between two concepts can be obtained by using Equation (7.8) shown below [3]:

$$sim_{eTVSM}(C_i, C_j) = L_{i,j} = \cos(\vec{\varsigma}_i, \vec{\varsigma}_j) = \vec{\varsigma}_i \vec{\varsigma}_j = \sum_{k=1}^{m} \varsigma_{i,k} \varsigma_{j,k} \tag{7.8}$$

Finally, the pair-wise concept similarity values from the pizza ontology in Fig. 7.15 are given in Table 7.2.

Table 7.2: Result of the lightweight-ontology-structure-based semantic similarity model (eTVSM) on the pizza ontology.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
c_1	1	0.7241	0.5111	0.7228	0.6610	0.6610	0.6374	0.6374	0.6374
c_2	0.7241	1	0	0.4141	0.9130	0.9130	0.3652	0.3652	0.3652
c_3	0.5111	0	1	0	0	0	0	0	0
c_4	0.7228	0.4141	0	1	0.3780	0.3780	0.8820	0.8820	0.8820
c_5	0.6610	0.913	0	0.3780	1	0.6668	0.3333	0.3333	0.3333
c_6	0.6610	0.913	0	0.3780	0.6667	1	0.3333	0.3333	0.3333
c_7	0.6375	0.3652	0	0.8820	0.3333	0.3333	1	0.6667	0.6667
c_8	0.6374	0.3652	0	0.8820	0.3333	0.3333	0.6667	1	0.6667
c_9	0.6374	0.3652	0	0.8820	0.3333	0.3333	0.6667	0.6667	1

7.4.3 Lightweight-Ontology-Structure-based Semantic Similarity Model (Normalized Resnik's Model)

Resnik [5] developed a semantic similarity model whereby the information shared by two concepts can be indicated by the concept which subsumes the two concepts in a taxonomy. Then, the similarity between the two concepts C_i and C_j can be mathematically expressed as follows:

$$sim_{\text{Resnik}}(C_i, C_j) = \max_{C \in S(C_i, C_j)}[-\log(\text{P}(C))] \tag{7.9}$$

where $S(C_i, C_j)$ is the set of concepts that subsumes both C_i and C_j, and $P(C)$ is the possibility of encountering an instance of concept C.

As mentioned previously, the lightweight ontology structure enables the use of existing semantic similarity models in the ontology environment. Here we use Resnik's model (Equation (7.9) for the lightweight ontology-based semantic similarity measure. Nevertheless, one limitation of Resnik's model is that its interval is $[0,\infty]$. With the purpose of according with the interval of the cosine correlation, we normalize Resnik's model by giving

$$|sim_{\text{Resnik}}(C_i, C_j)| = \begin{cases} \dfrac{\max_{\varsigma \in S(\varsigma_i, \varsigma_j)}[-\log(\mathrm{P}(\varsigma))]}{\max_{\varsigma \in \Theta}[-\log(\mathrm{P}(\varsigma))]} & \text{if } \varsigma_i \neq \varsigma_j \\ 1 & \text{if } \varsigma_i = \varsigma_j \end{cases} \quad (7.10)$$

where Θ is the collection of pseudo-concepts in a lightweight ontology [2].

The result of the lightweight-ontology-structure-based semantic similarity model on the pizza ontology in Fig. 7.15 is shown in Table 7.3.

Table 7.3: Result of the lightweight-ontology-structure-based semantic similarity model (Normalized Resnik's model) on the pizza ontology.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
c_1	1	0	0	0	0	0	0	0	0
c_2	0	1	0	0	0.5	0.5	0	0	0
c_3	0	0	1	0	0	0	0	0	0
c_4	0	0	0	1	0	0	0.3690	0.3690	0.3690
c_5	0	0.5	0	0	1	0.5	0	0	0
c_6	0	0.5	0	0	0.5	1	0	0	0
c_7	0	0	0	0.3690	0	0	1	0.3690	0.3690
c_8	0	0	0	0.3690	0	0	0.3690	1	0.3690
c_9	0	0	0	0.3690	0	0	0.3690	0.3690	1

7.4.4 Two Hybrid Semantic Similarity Models

Based on the three sub-models, here we present our two alternative hybrid semantic similarity models, which are:

Semantic similarity model-1. This hybrid model is a weighted arithmetic mean between the pseudo-concept-based semantic similarity sub-model and the lightweight-ontology-structure-based semantic similarity model that originates from the eTVSM, which can be expressed mathematically as [1, 3]:

$$sim(C_i, C_j) = (1-\beta) \times sim_{\cos\theta}(c_i, c_j) + \beta \times sim_{eTVSM}(c_i, c_j) \ \ (0 \le \beta \le 1) \quad (7.11)$$

Semantic similarity model-2. This model is a weighted arithmetic mean between the pseudo-concept-based semantic similarity sub-model and the lightweight-ontology-structure-based semantic similarity model that originates from Resnik's semantic similarity model, which can be expressed mathematically as [2]:

$$sim(C_i, C_j) = (1-\beta) \times sim_{\cos\theta}(C_i, C_j) + \beta \times |sim_{\text{Resnik}}(C_i, C_j)| \ \ (0 \le \beta \le 1) \quad (7.12)$$

The value of β for each model is determined upon the optimal performance of each model in the following experiments, which is introduced in Chapter 10.

In addition, analogous to the service retrieval methodology, owing to the huge number of concepts in an ontology, there could be a great number of semantically similar service concepts returned by the two semantic similarity models, based on the similarity values between those concepts and a service-requester-selected concept, which may make it difficult for a service requester to make a choice. Hence, we need to choose the optimal threshold value for each model in order to reduce the number of semantically similar concepts retrieved in addition to retaining the optimal performance of each model. The experiments with regard to the evaluation of the two semantic similarity models and the selection of optimal threshold values are presented in Chapter 10.

7.5 Conclusion

In this chapter, we proposed a service concept recommendation methodology in order to avoid the failure of the Service Search Engine, when service requesters find that the eventually selected concrete service concepts cannot match their service intentions. In this instance, the service concept recommendation methodology can be used to recommend semantically similar service concepts to the selected concept, which can help service requesters in the next round of service query disambiguation.

The service concept recommendation methodology uses a semantic similarity model to compute the similarity values between the selected concept and other concepts in an ontology. In order to deal with the semantic-rich content of ontology relations and concepts as well as the complicated structure of ontologies, we define a pseudo-concept and a lightweight ontology space. The former encapsulates all properties, restrictions and characteristics of properties of an ontology concept into a space, which enables the feasibility of assessing similarity between concepts based on content of pseudo-concepts. The latter simplifies an ontology to a taxonomic structure, in which pseudo-concepts are linked by *is-a* relations. Based on the two definitions, we define a series of theorems for converting ontologies to lightweight ontology spaces. Furthermore, we provide two hybrid semantic similarity models. The first model combines a pseudo-concept-based semantic similarity sub-model and a lightweight-ontology-structure-based semantic similarity sub-model which originates from the eTVSM, by means of a weighted arithmetic mean. The second model combines the pseudo-concept-based semantic similarity sub-model and a lightweight-ontology-structure-based semantic similarity sub-model which originates from Resnik's semantic similarity model, in terms of a weighted arithmetic mean. The details regarding the evaluation of the two models are provided in Chapter 10.

As we mentioned in Chapter 6, as a concrete service concept is selected from the Service Ontology Base by a service requester, all its associated SDE metadata are retrieved and ranked from the SDE metadata Base. In Chapter 8, we propose a QoS-base SDE metadata methodology.

7.6 References

1. Dong, H., Hussain, F.K., Chang, E.: A service concept recommendation system for enhancing the dependability of semantic service matchmakers in the service ecosystem environment. Journal of Network and Computer Applications **Submitted**

2. Dong, H., Hussain, F.K., Chang, E.: A context-aware semantic similarity model for ontology environments. Concurrency and Computation: Practice and Experience **In Press**

3. Dong, H., Hussain, F.K., Chang, E.: A hybrid concept similarity measure model for the ontology environment. In: Meersman, R., Tari, Z., Herrero, P. (eds.): On the Move to Meaningful Internet Systems: OTM 2009 Workshops. Springer-Verlag, Vilamoura, Portugal (2009) 848-857

4. Kuropka, D.: Modelle zur repräsentation natürlichsprachlicher dokumente. ontologie-basiertes information-filtering und -retrieval mit relationalen datenbanken. In: Becker, J., Grob, H.L., Klein, S., Kuchen, H., Müller-Funk, U., Vossen, G. (eds.): Advances in Information Systems and Management Science. Logos Verlag Berlin, Berlin (2004)

5. Resnik, P.: Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. Journal of Artificial Intelligence Research **11** (1999) 95-130

Chapter 8 – QoS Evaluation and Service Ranking Methodology

8.1 Introduction

In Chapter 6, by means of a series of HCI activities with the Service Search Engine, a service requester eventually selects a concrete service concept that can be used to denote his/her service query intention. All the SDE metadata associated with the selected service concept are then retrieved from the SDE Metadata Base. However, as in normal instances where there are numerous service entities in a service domain, there could be a huge number of metadata associated with a concept. A critical question that needs to be addressed is – how can a service requester make an appropriate selection from amongst the retrieved services represented by the service metadata based on Quality of Services (QoS)? One possible solution is QoS-based service selection. In this section, by means of the premised scenario below, we discuss the requirements associated with the QoS-based service selection in the Digital Ecosystems environment.

Hai lives in City A and one day he needs an air cargo service. There are many companies that provide air cargo services within City A. As a result, Hai intends to find out the overall QoS ranking for the air cargo companies in order to make an objective decision. In addition, Hai may intend to make a service selection by assigning greater preference to certain criteria. Moreover, after selecting an air cargo company and conducting a service transaction with the company, Hai may intend to contribute to the ranking of air cargo companies by evaluating this company's performance in this transaction [3].

Here we formalize Hai's requirements in the Digital Ecosystems as follows [3]:

- A QoS-based service ranking methodology is required to rank service providers under a service concept in multiple user-desired ways (or according to multiple criteria) according to their performance in the relevant service activities or against context-specific criteria. e.g., ranking the companies who can provide an air cargo service based on their past performance under the criteria of quality, speed, price or a combination of these.

- A QoS-based service evaluation methodology is required to allow a given service requester to evaluate a service provider's performance in a service transaction comprising of multiple context-specific criteria, thereby allowing Hai to evaluate an air cargo company's performance under the criteria of quality, speed, price, after he conducts an air cargo service transaction with the company.

- The service evaluation methodology can be utilized for the benefit of the service ranking methodology; that is, Hai's evaluation of the air cargo company can be employed as one of the foundations that the company relies on for its ranking within the air cargo service.

With the purpose of fulfilling the above requirements, in this chapter, we propose a QoS evaluation and service ranking methodology in order to assist a service requester to [2]:

- evaluate the performance of a service provider in a service transaction with the service requester under each domain-specific QoS evaluation criteria,
- evaluate the trustworthiness of a service provider in a service transaction with the service requester,
- discover the performance of the service providers under each domain-specific QoS evaluation criteria, and
- ascertain the reputation of the service providers that providers a requested service.

The rest of paper is organized as follows: in Section 8.2 we introduce the system architecture and workflow of the proposed conceptual framework; in Section 8.3 we introduce a QoS evaluation methodology employed in the system framework; and conclusions are drawn in Section 8.4.

8.2 System Architecture

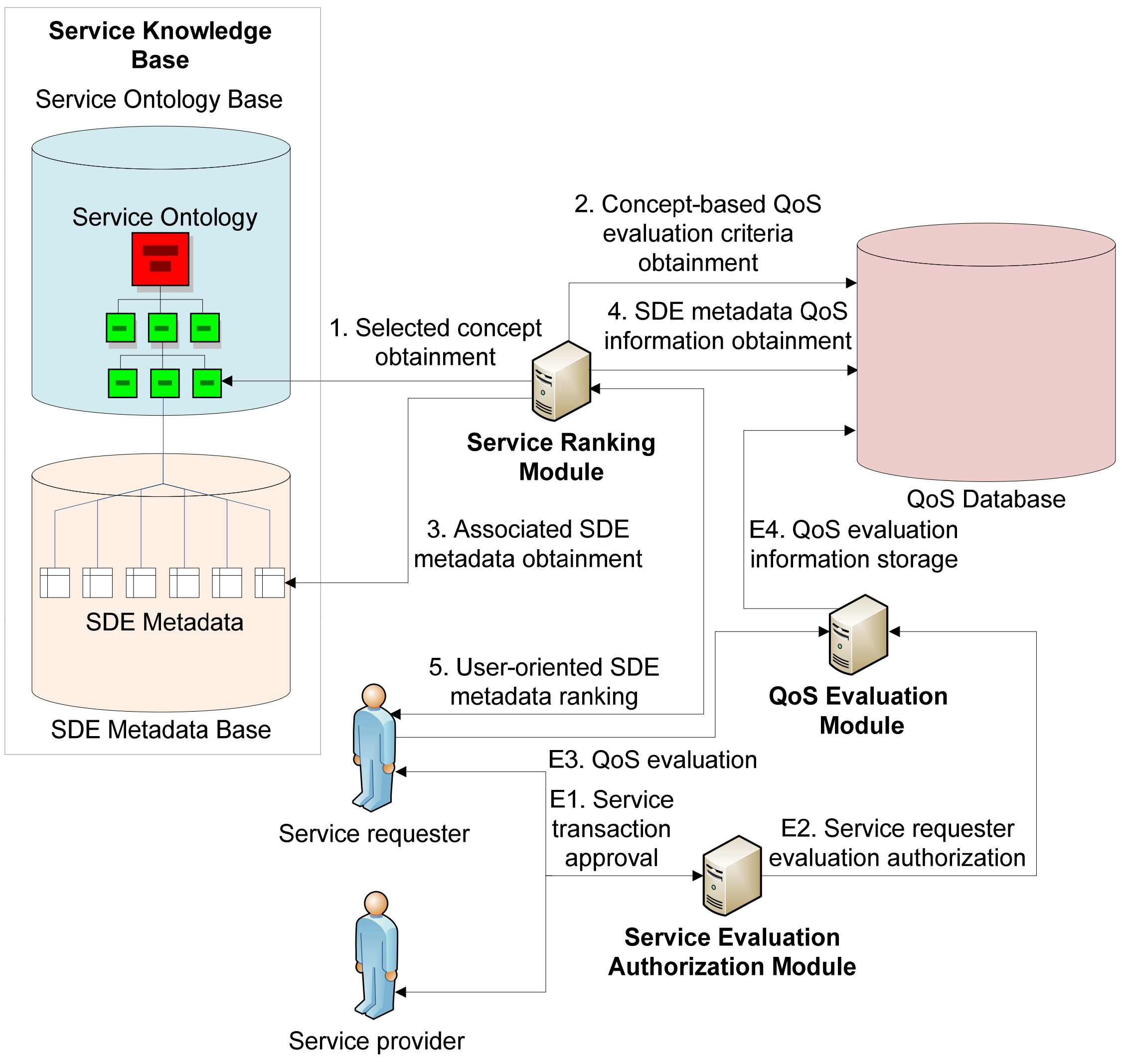
In this section, in order to present the QoS evaluation and service ranking methodology, we explain the system architecture, including the system workflow of its prototype – a QoS Evaluation and Service Ranking System, which is shown in Fig. 8.1. The whole system consists of four basic components, which are:

- QoS Database. This database is designed to store service-concept (service-domain)-based QoS evaluation criteria, and QoS information regarding SDE metadata, which includes: 1) performance values of service providers on each belonged SDE metadata under each QoS evaluation criterion, 2) reputation values of service providers on each belonged SDE metadata, and 3) service requesters' evaluation values on each SDE metadata. Therefore, each concrete service concept in the Service Ontology Base corresponds to several particular QoS evaluation criteria stored in the QoS Database. Furthermore, the SDE metadata associated with a service concept must follow the particular criteria to which the concept corresponds. Here the criteria must be domain-specific; and, in other words, even though two criteria may have the same contents since they are associated with two concepts, these two criteria cannot be regarded as being the same.

- Service Ranking Module. The aim of the module is to rank SDE metadata in a multi-linear manner, based on either each domain-specific QoS criterion or the overall reputation of the corresponding service providers, according to preferences or requirements of service requesters.

- Service Evaluation Authorization Module. This module authorizes a service requester to evaluate a service provider's performance after completing a service transaction with the service provider.

- QoS Evaluation Module. This module enables a service requester to evaluate a service provider's performance by means of a proposed QoS evaluation methodology, which is introduced in the next section.



Service Knowledge Base
Service Ontology Base
Service Ontology
SDE Metadata
SDE Metadata Base
2. Concept-based QoS evaluation criteria obtainment
1. Selected concept obtainment
4. SDE metadata QoS information obtainment
Service Ranking Module
QoS Database
3. Associated SDE metadata obtainment
E4. QoS evaluation information storage
5. User-oriented SDE metadata ranking
QoS Evaluation Module
E3. QoS evaluation
E1. Service transaction approval
E2. Service requester evaluation authorization
Service requester
Service Evaluation Authorization Module
Service provider

Figure 8.1: System architecture of the QoS Evaluation and Service Ranking System

The whole system is designed to realize two basic functions – service ranking and QoS evaluation. In the rest of this section, we introduce the workflow of the whole system from the perspective of the function realization.

In order to realize the function of service ranking, the components within the system follow a five-step workflow as follows:

Step 1. Selected concept obtainment. When a service requester selects a concrete service concept that can best denote his/her query intention from a Service Search Engine, the Service Ranking Module will retrieve the selected service concept from the engine.

Step 2. Concept-based QoS evaluation criteria obtainment. Based on the retrieved service concept, the Service Ranking Module will retrieve the corresponding QoS evaluation criteria from the QoS Database.

Step 3. Associated SDE metadata obtainment. The Service Ranking Module retrieves the URIs of all SDE metadata associated with the selected concept from the Service Search Engine.

Step 4. SDE metadata QoS information obtainment. Based on the retrieved URIs, the Service Ranking Module is able to retrieve all QoS information of the SDE metadata from the QoS Database.

Step 5. User-oriented SDE metadata ranking. The Service Ranking Module then combines the concept-based QoS evaluation criteria information and the QoS information of the SDE metadata together, and displays the information to the service requester, in order to enable the service requester to rank the SDE metadata according to personal preference.

The function of QoS evaluation is implemented according to three steps as follows:

Step 1. Service transaction approval. After a service requester completes a service transaction with a service provider, we propose that the service requester needs to send a request to the Service Evaluation Authorization Module for the QoS evaluation authorization. Upon receiving the request, the Service Evaluation Authorization Module sends a request to the service provider in order to ask for service transaction approval.

Step 2. Service requester evaluation authorization. We propose that the service provider will send an approval message after receiving the service transaction approval request. Once receiving the approval from the service provider, the Service Evaluation Authorization Module will send an authorization approval message to the QoS Evaluation Module.

Step 3. QoS evaluation. On receiving the authorization approval message, the QoS Evaluation Module sends a message to the service requester, and the service requester can evaluate the performance of the service provider in the service transaction by means of a QoS evaluation methodology introduced in the next section.

Step 4. QoS evaluation information storage. Once the service requester submits the evaluation information, the information will then be stored in the QoS Database for the service ranking.

8.3 QoS Evaluation Methodology

In this section, we introduce a QoS evaluation methodology employed in our proposed framework of the QoS Evaluation and Service Ranking System. The methodology is extended from the CCCI (Correlation of Interaction, Correlation of Criterion, Clarity of Criterion, and Importance of Criterion) metrics proposed by Chang et al. [1] and Hussain et al. [4], which measures the reputation extent of a service provider on a service entity by assessing the extent of his/her trustworthiness which past service requesters have assigned for the same service transactions. Here we propose two metrics which would assist in the process of (a) ranking of the service providers based on their past behaviour; and (b) ranking of service providers according to domain specific criteria. However, in order to understand the metrics that we are proposing in this paper, the users need to understand the definitions of trust, reputation and each of the metrics proposed in the CCCI metrics. In Section 8.3.1 and Section 8.3.2 we present a very brief overview of the CCCI metrics and the concepts of trust and reputation. In Section 8.3.3, we propose and define our proposed metrics.

8.3.1 Definition of trustworthiness and reputation

As proposed by Chang et al. [1] and Hussain et al. [4], trustworthiness is defined as a numeric value that estimates the level of trust in a service interaction. Two roles are involved in such a service interaction, namely service provider and service requester. The service provider is an agent that provides a given service to another agent who is requesting it. The service requester is an agent who is requesting a particular service [1]. The service requester has a given level of trust in the service provider, which is quantified and expressed as the trustworthiness value.

Based on the definitions proposed by Chang et al. [1] and Hussain et al. [4], below we explain the terms "*a numeric value*", "*estimate*", "*level of trust*", for elucidation purposes:

> "*A numeric value*" refers to the quantification of the level of the trust in the service interaction.
>
> "*Estimate*" refers to the measurement of the level of trust, which is a predictive and tentative measure.
>
> "*Level of trust*" refers to the degree or the scale of trust that the service requester has in the service provider.

As proposed by Chang et al. [1] and Hussain et al. [4], reputation is defined as the aggregation of the recommendations from the entire third party recommendation agents, in response to a service requester's reputation query with respect to the quality of a service provided by a given service provider, for a given service and within a given time slot.

Based on the definitions proposed by Chang et al. [1] and Hussain et al. [4], we explain the terms "*reputation query*", "*recommendation third party*", "*recommendation*" below for elucidation purposes:

> "*Reputation query*" refers to the query made by the service requester with respect to the reputation of the service provider. It normally consists of the queried service provider, the context and the time slot of the query request.
>
> "*Third party recommendation agent*" refers to an agent that has previously interacted with the service provider within the context and time slot specified in the reputation query. The third party recommendation agent can provide recommendations to the reputation query.
>
> "*Recommendation*" refers to the opinion of the third party recommendation agent about the reputation query.

8.3.2 CCCI metrics

CCCI metrics is a group of metrics developed by Chang et al. [1] and Hussain et al. [4], with the purpose of measuring the trustworthiness and reputation of services. These metrics are based on the premise that a service interaction between a given service provider and a service requestor takes place in a given context (or scenario) which in turn could be de-composed into a finite number of criteria. A criterion is considered to be a decisive factor of the mutually agreed service performance between the service provider and service requester for quality assessment purposes. It is important to note that the service requester and the service provider engage in a negotiation phase before the interaction and agree on the mutually agreed service or the service level agreement (SLA).
By making use of the CCCI metrics, the service requester can evaluate the performance of the service provider according to the decisive factor(s) after the service interaction. The criteria are often a series of activities. Thus, the measurement of the QoS for the service provider in the interaction becomes the measurement of each criterion involved in the service interaction.

To enhance the application of CCCI metrics in the evaluation and ranking of the service, we extend the theory of CCCI metrics by proposing two metrics in addition to the existing ones. They are (a) the reputation of the service provider; and (b) the actual behaviour of the service provider against a given criterion. First of all, we provide the metrics for the third party recommendation agents (namely the service requesters who have previously completed the specified service interactions with the service providers in the Digital Ecosystems) to provide recommendations (quantitative evaluations) as their trustworthiness value about the service provider. Secondly, based on the trustworthiness values from all third party recommendation agents involved in the specified service interaction, the corresponding service provider's reputation value in the service can be computed and utilized for the service ranking. Finally, an additional function is provided from the extended CCCI metrics, by allowing service requesters to rank service providers based on given specified criteria.

In order to understand these metrics and their workings, it is very important to understand the workings of the proposed CCCI metrics. For elucidation purposes, we explain the proposed CCCI metrics by Chang et al. [1] and Hussain et al. [4] in this section.

The CCCI metrics is a suite of four metrics as shown below:

1. Correlation of an interaction ($Corr_{Interaction}$)

2. Correlation of a criterion ($Corr_{Criterion}$)

3. Clarity of a criterion ($Clear_{Criterion}$)

4. Importance of a criterion ($Imp_{Criterion}$)

In the following sections, we will describe them in detail for elucidation purposes only.

Definition 8.1. Correlation of an interaction ($Corr_{Interaction}$)

As defined by Chang et al. [1] and Hussain et al. [4], the $Corr_{Interaction}$ is a metric that expresses *"the degree of parallelism between the actual behaviour of a service provider in the service interaction ($ActualBehaviour_{Interaction}$) and the mutually agreed behaviour of the service provider in the service interaction ($MutuallyAgreedBehaviour_{Interaction}$)"*. The $ActualBehaviour_{Interaction}$ can be determined by aggregating the actual behaviour of each criterion involved in the service interaction. Similarly, the $MutuallyAgreedBehaviour_{Interaction}$ can be determined by aggregating the mutually agreed behaviour of each criterion involved in the service interaction.

Mathematically, $Corr_{Interaction}$ is computed as shown below:

$$Corr_{Interaction} = \frac{ActualBehaviour_{Interaction}}{MatuallyAgreedBehaviour_{Interaction}} \tag{8.1}$$

Definition 8.2. Correlation of a criterion ($Corr_{Criterion}$)

As defined by Chang et al. [1] and Hussain et al. [4], the $Corr_{Criterion}$ is a metric that expresses *"the degree of parallelism between the actual behaviour and the mutually agreed behaviour of the service provider in a given criterion"*.

Definition 8.3. Actual Behaviour Criterion Correlation ($ABCorr_{Criterion}$)

As defined by Chang et al. [1] and Hussain et al. [4], the $AB_{CorrCriterion}$ is a metric that *"qualifies and expresses the actual behaviour of the service provider in the given* *criterion"* for the third party recommendation agent". The $AB_{CorrCriterion}$ can have seven levels as shown below:

0 – Ignorance

1 – Very Dissatisfied

2 – Dissatisfied

3 – Neutral

4 – Partially satisfied

5 – Satisfied

6 – Very satisfied

Definition 8.4. Mutually Agreed Behaviour Criterion Correlation ($MAB_{CorrCriterion}$)

As defined by Chang et al. [1] and Hussain et al. [4], the $MAB_{CorrCriterion}$ is a metric that *"qualifies and expresses the mutually agreed behaviour of the service provider in the given criterion"*. $MAB_{CorrCriterion}$ for each criterion that has been mutually agreed upon by both the interacting parties and documented in the service level agreement (SLA) would be '1'.

Definition 8.5. Clarity of a criterion ($Clear_{Criterion}$)

As proposed by Chang et al. [1] and Hussain et al. [4], the $Clear_{Criterion}$ is a metric that *"qualifies the extent of weather a criterion is mutually agreed between the service provider and the service requestor*".

For a given criterion, $Clear_{Criterion}$ can have two levels as shown below:

0 – This criterion or its output or both have not been mutually agreed upon by the involved parties

1 – This criterion along with its output has been mutually agreed upon by the involved parties

Definition 8.6. Importance of a criterion ($Imp_{Criterion}$)

As proposed by Chang et al. [1] and Hussain et al. [4], the $Imp_{Criterion}$ is a metric that qualifies the extent of importance of a criterion for the service requestor in the interaction.
For a given criterion, $Imp_{Criterion}$ can have three levels as shown below:

1 – Not important

2 – Important

3 – Very important

Definition 8.7. Trustworthiness (*Trustworthiness*)

As proposed by Chang et al. [1] and Hussain et al. [4], the *trustworthiness* value of a service interaction is expressed as *"the degree of consonance or parallelism between the actual behaviour of a service provider and the mutually agreed behaviour of the service provider in the service interaction, as perceived by a service requestor"*. As a result of this, the trustworthiness value of a service interaction would be the same as the correlation of the interaction ($Corr_{Criterion}$).

Mathematically, the formula for computing *Trustworthiness* is shown below:

$$\begin{aligned} Trustworthiness &= Corr_{Interaction} \\ &= \frac{ActualBehaviour_{Interaction}}{MutuallyAgreedBehaviour_{Interaction}} \\ &= \frac{\sum_{i=1}^{n} ABorr_{Criterion\ i} \times Clear_{Criterion\ i} \times \mathrm{Imp}_{Criterion\ i}}{\sum_{i=1}^{n} MABorr_{Criterion\ i} \times Clear_{Criterion\ i} \times \mathrm{Imp}_{Criterion\ i}} \end{aligned} \tag{8.2}$$

where n is the number of criteria involved in a given service interaction.

Trustworthiness can have seven levels as shown below:

0 – Ignorance or unknown

1 – Completely untrustworthy or extremely untrustworthy

2 – Untrustworthy

3 – Minimal trustworthy

4 – Partially trustworthy

5 – Trustworthy

6 – Extremely trustworthy

By observing the formula for trustworthiness, it is found that the service requesters can evaluate the trustworthiness of the service providers in the service interaction by assigning values to the components of the formula, namely the components of CCCI metrics – $ABCorr_{Criterion}$, $Clear_{Criterion}$, and $Imp_{Criterion}$. Once these values have been assigned, the trustworthiness value can be computed. The service requester can make use of these assigned values for recommendation purposes in the future.

8.3.3 Extension of the CCCI Metrics for Reputation-based Service Ranking and Domain-Specific-Criteria-based Service Ranking

In this section, in order to apply CCCI metrics to QoS-based service ranking, we propose two new metrics in addition to the existing suite of CCCI metrics. The definitions of these metrics are given in this section. These metrics enable domain specific (or service requestor specific) service ranking.

Definition 8.8. Reputation (*Reputation*)

We define the *reputation* value of a service provider in a given context as the average of all involved third party recommendation agents' *trustworthiness* values for this service provider in the same service context [3].

$$\text{Reputation} = \frac{\sum_{i=1}^{m} Trustworthiness_i}{m} \tag{8.3}$$

where m is the number of third party recommendation agents for this service provider in the service interaction.

Reputation can have seven levels as shown below [1, 4]:

0 – Cannot determine reputation

1 – Extremely bad reputation

2 – Bad reputation

3 – Minimally good reputation

4 – Partially good reputation

5 – Good reputation

6 – Extremely good reputation

This metric enables the ranking of a set of service providers according to their reputation value. The reputation values computed by using the above formulae could be used for context-based service ranking.

Definition 8.9. Actual behaviour of a service provider in a single criterion (*ActualBehaviour*$_{Criterion}$)

In addition, in order to allow service requesters to rank services according to a given criterion, we propose the metric of Actual behaviour of a service provider against a single criterion (*ActualBehaviour*$_{Criterion}$).

We define the *ActualBehaviour*$_{Criterion}$ as a metric that qualifies and expresses the actual behaviour of a service provider against a given criterion, as perceived by all involved third party recommendation agents [3].

In consonance with the levels of trustworthiness, *ActualBehaviour*$_{Criterion}$ can have seven levels as shown below:

0 – Ignorance

1 – Very Dissatisfied

2 – Dissatisfied

3 – Neutral

4 – Partially satisfied

5 – Satisfied

6 – Very satisfied

ActualBehaviour$_{Criterion}$ is determined by the average of all the previous service requesters' evaluation values for this given criterion, namely the average value of Actual Behaviour Criterion Correlation (*ABCorr*$_{Criterion}$). The formula of *ActualBehaviour*$_{Criterion}$ is given as [3]:

$$ActualBehaviour_{Criterion} = \frac{\sum_{i=1}^{m} ABCorr_{Criterion\ i}}{m} \tag{8.4}$$

where m is the number of evaluators for this criterion.

With regard to the use scenario in Section 8.1, the functions of the extended CCCI metrics that can be utilised to satisfy Hai's requirements, by providing a series of quantitative methodologies for him, are as follows:

- Evaluating its performance in each air cargo industry standard (criterion) – *ABCorr*$_{Criterion}$;

- Evaluating Hai's trustworthiness toward a given air cargo service provider – *Trustworthiness*;

- Finding out the reputation of the companies that provide air cargo services in City A from the perspective of air cargo services – *Reputation*; and

- Finding out the performance of the companies in each industry standard (criterion) of air cargo services – $ABCorr_{Criterion}$.

Therefore, the extended CCCI metrics can be applied to allow a service requester to quantitatively evaluate and rank service providers against multiple user-desired criteria and the overall reputation of service providers.

8.4 Conclusion

After a service requester selects a concrete service concept that can be used to represent his/her service request to a maximum degree, s/he also needs to make a selection from the retrieved SDE metadata associated with the concept. Usually, the service requester wants to choose the SDE metadata with the highest quality. In order to realize this function, in this chapter, we proposed a QoS evaluation and service ranking methodology. The QoS methodology is extended from Chang and Hussain's CCCI metrics, which measures the extent of a service provider's reputation regarding a service entity by assessing the trustworthiness extent of the service provider which the service requesters have assigned in the past for the same service transactions. In order to realize this methodology, first of all, we regard each concrete service concept as a particular service domain, and each domain is associated with several domain-specific QoS evaluation criteria. As a result, the SDE metadata related to these concepts are also associated with these criteria. Therefore, after completing a service transaction with a service provider, a service requester can evaluate the performance of the service provider, based on each criterion associated with the SDE metadata that is used to represent the service entity, from three perspectives: 1) the performance of the service provider on this criterion ($ABCorr_{Criterion}$), 2) the clarity of this criterion ($Clear_{Criterion}$), and 3) the importance of this criterion ($Imp_{Criterion}$). By aggregating the scores on all criteria for the SDE metadata, the trustworthiness extent (*Trustworthiness*) of the service provider towards the service requester can be measured. By leveraging the trustworthiness values of the service provider on the SDE metadata assigned by all the service requesters, the reputation value (*Reputation*) of the service provider on the SDE metadata can be obtained. Hence, service requesters can rank SDE metadata under a concrete service concept (service domain) by obtaining the reputation values of their relevant service providers on these metadata. In addition, the SDE metadata can also be ranked according to their particular service providers' performances according to each criterion ($ActualBehaviour_{Criterion}$).

In the next chapter, in order to allow our proposed service retrieval methodology to adapt to the dynamic nature of services and service providers in the Digital Ecosystems environment, we design a methodology enabling community-based ontology updating and service-provider-based service publishing, maintenance and classification.

8.5 References

1. Chang, E., Dillon, T.S., Hussain, F.: Trust and Reputation for Service Oriented Environments-Technologies for Building Business Intelligence and Consumer Confidence. John Wiley & Sons (2005)

2. Dong, H., Hussain, F.K., Chang, E.: Quality of service (QoS) based service retrieval engine. Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM '08). ACM, Linz, Austria (2008) 405-408

3. Dong, H., Hussain, F.K., Chang, E.: A QoS-based service retrieval methodology for digital ecosystems. International Journal of Web and Grid Services **5** (2009) 261-283

4. Hussain, F.K., Chang, E., Dillon, T.S.: Trustworthiness and CCCI metrics in P2P communication. International Journal of Computer Systems Science and Engineering **19** (2004) 173-190

Chapter 9 – Service Domain Knowledge Updating and Service-Provider-based SDE Metadata Publishing, Maintenance and Classification Methodology

9.1 Introduction

One notable feature of the Digital Ecosystems environment is the dynamic nature of service domain knowledge and the subjective perceptions of service providers and service requesters. First of all, service domain knowledge is not static and changes constantly, along with evolving technologies and dynamic service environments. In contrast, the subjective perceptions of service providers and service requesters are impacted upon by the dynamic service domain knowledge and vice versa. For example, a new technology may give birth to a new service available to service requesters, and service requesters may have new requirements for the new service, which may evolve the new service. In our design, the service domain knowledge is represented by service ontologies within the Service Ontology Base. Therefore, these ontologies need to evolve to adapt to the dynamic nature of service domain knowledge. Additionally, service domain knowledge also contains service-concept-based QoS evaluation criteria which are stored in the QoS Database. These QoS evaluation criteria are also influenced by the dynamic nature of services and the subjective perceptions of service providers and service requesters, and thus keep evolving to adapt to the dynamic nature of knowledge. Therefore, we need to provide a methodology enabling the service ontologies in our Service Ontology Base and the QoS evaluation criteria in the QoS Database to keep pace with the changing service domain knowledge.

Although in Chapter 5 we designed a service information discovery, annotation and classification methodology enabling automatic SDE metadata classification, since the SDE metadata in the Service Metadata Base are descriptions of actual service entities provided by service providers, service providers should be provided with the flexibility to decide the classification and descriptions of their SDE metadata. In addition, one objective of our research is to construct a community in which service providers and service requesters can collaborate to enhance the service knowledge in Digital Ecosystems. In the Web service architecture, a Universal Description Discovery and

Integration (UDDI) feature enables service providers to register and maintain their published Web services [5]. Analogous to the Web service architecture, we also need a similar platform to help service providers in the Digital Ecosystems environment to advertise and edit their services. This motivates us to design a service-provider-based SDE metadata publishing, maintenance and classification methodology.

Therefore, the two methodologies can be employed together to evolve information within the Service Knowledge Base and the QoS Database, in order to adapt to the dynamic nature of services and subjective perception of service providers and service requesters in the Digital Ecosystems environment. The rest of the chapter is organized as follows: in Section 9.2 we introduce the technical details regarding the service domain knowledge updating methodology; in Section 9.3 we propose the SDE metadata publishing, maintenance and classification methodology; and we conclude this chapter in Section 9.4.

9.2 Service Domain Knowledge Updating Methodology

While most existing ontologies are designed by individuals or small groups of experts, actual ontology users are not involved in the development process. Such an ivory-towered ontology creating approach may lead to a weak community grounding. According to Tim Berners-Lee's vision, online communities play an essential role in the mission of knowledge contribution [2]. Web 2.0, which provides various forms of platform for online communities [4], could be utilized to enhance the community grounding of ontologies, in order to improve the feasibility of ontologies in the community environment.

In order to maintain the service ontologies in the Service Ontology Base and the QoS evaluation criteria in the QoS Database, we adopt the notion of Web 2.0 to design a voting-based service domain knowledge updating methodology. Below is the definition of the voting-based service domain knowledge updating methodology [1].

$Value_{voting}$ is the voting result for a change request, which is the weighted average value of the average values of

1. The voting result for a change request from a normal user ($Vote_{user}$), varying among

    -1 – Disagree

    0 – Neutral

    1 – Agree

2. The voting result for a change request from a domain expert ($Vote_{Expert}$), varying among

-1 – Disagree

0 – Neutral

1 – Agree

The score of $Value_{voting}$ can be obtained by

$$Value_{voting} = \alpha \times \frac{\sum_{i=1}^{n} Vote_{User_i}}{n} + \beta \times \frac{\sum_{j=1}^{m} Vote_{Expert_j}}{m} \quad (9.1)$$

where n is the number of voting normal users for a given voting, m is the number of voting domain experts for a voting, α is the weight of the votes from normal users, β is the weight of the votes from domain experts, $\alpha < \beta$ and $\alpha + \beta = 1$. A threshold value is given to determine the overall result of a vote for change or otherwise [1].

In terms of the types and scopes of changes, we distinguish change requests for service domain knowledge according to three main categories:

- Concept-based change requests. The requested changes occur only within the scope of an ontology concept. For example, the change of the URI (name) or datatype properties of a concept.

- Ontology-based change requests. The requested changes occur beyond the scope of a concept but within the scope of an ontology. For example, the change of objective properties of a concepts.

- Criterion-based change requests. The requests for changing QoS evaluation criteria for concepts.

The identification of a domain expert and a normal user is determined by whether or not a service provider has a SDE metadata associated with the concept(s) in which a requested change occurs. Therefore, the service providers who have any SDE metadata associated with the concepts for which a change has been requested can be deemed as domain experts for the update of the concepts.

Fig. 9.1 shows the sequence diagram of a service domain knowledge updating process, which illustrates the components and workflow of the service domain knowledge updating system. When a user desires to make a change request, s/he may submit the request through a Service Domain Knowledge Update Request Interface. Once the request has been submitted, the request will be sent to a central Voting for Update Module. This module will generate a voting procedure with a time limit, and send messages to all registered users to ask for participation. When users log into the voting interface, according to their user rights (normal user or domain expert), they are able to access different voting interfaces and submit a vote. When the time has ended, the Voting for Update Module will obtain the voting results from the two user groups, and compute the final result based on the service domain knowledge updating methodology. The result will then be sent to all users' voting interfaces and the platform administrator for the final decision [1].

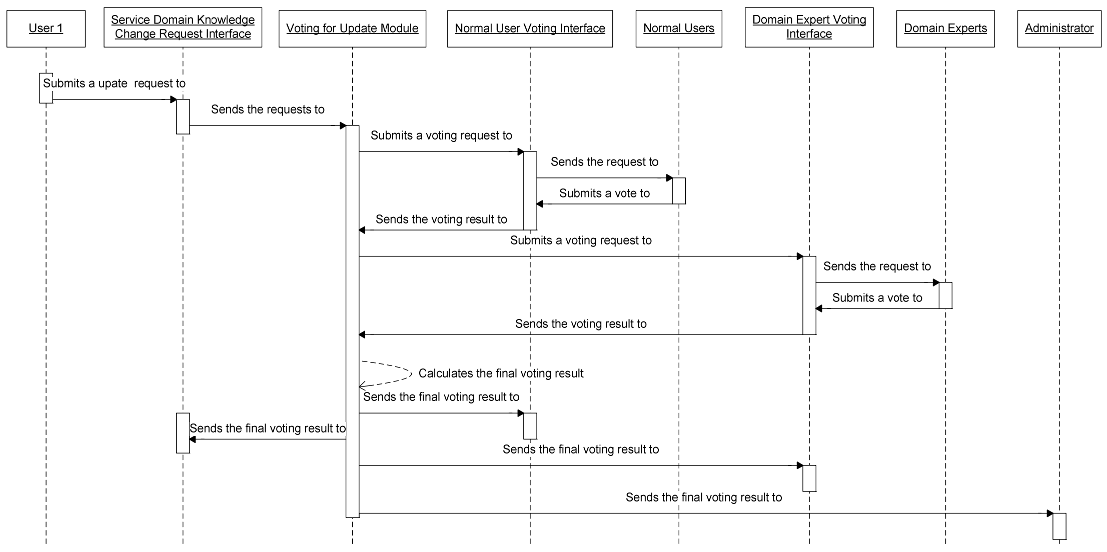

User 1
Service Domain Knowledge Change Request Interface
Voting for Update Module
Normal User Voting Interface
Normal Users
Domain Expert Voting Interface
Domain Experts
Administrator
Submits a upate request to
Sends the requests to
Submits a voting request to
Sends the request to
Submits a vote to
Sends the voting result to
Submits a voting request to
Sends the request to
Submits a vote to
Sends the voting result to
Calculates the final voting result
Sends the final voting result to
Sends the final voting result to
Sends the final voting result to
Sends the final voting result to

Figure 9.1: Sequence diagram of the community-driven ontology updating methodology

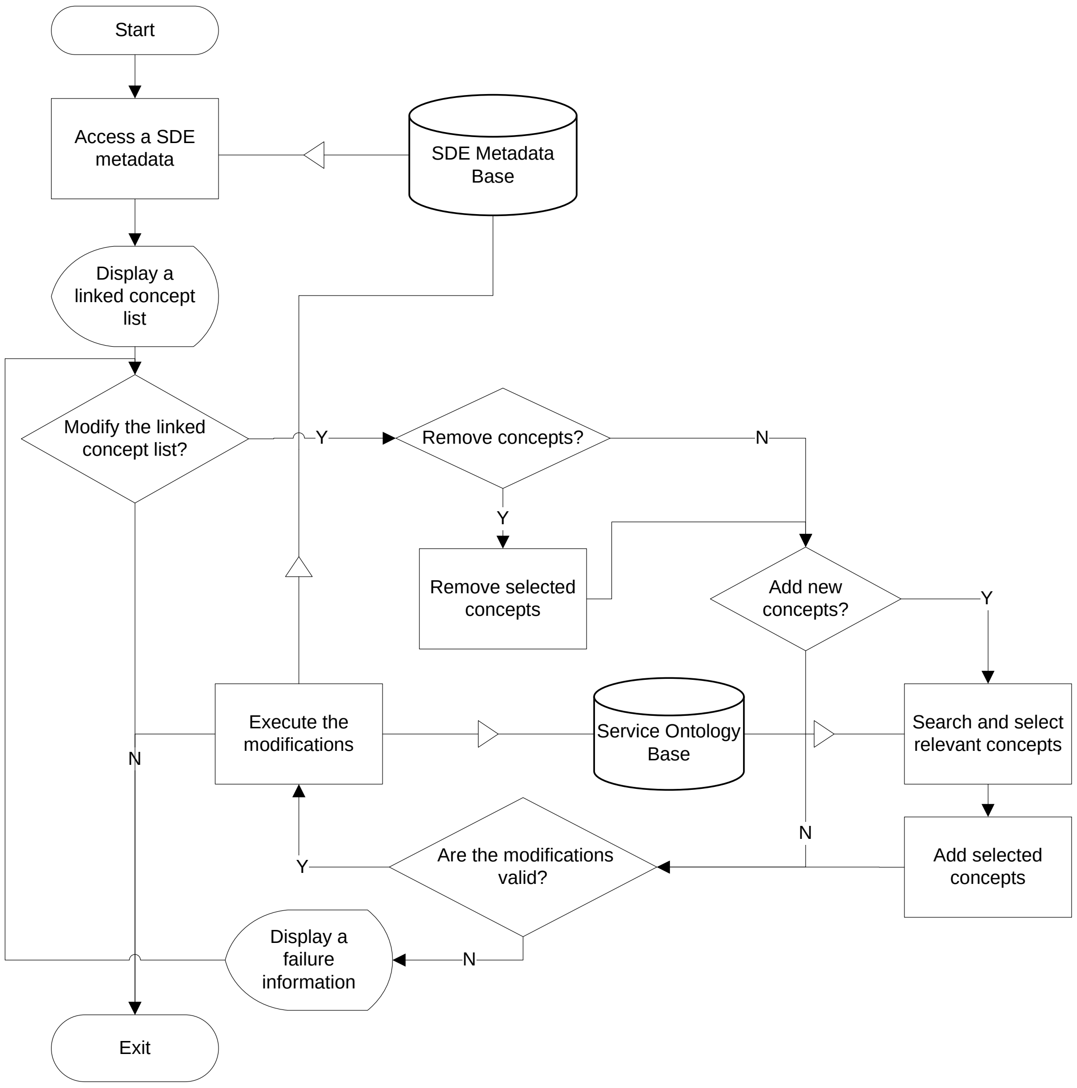
In Chapter 10, we present the prototype of the methodology for service domain knowledge updating.

9.3 Service-Provider-based SDE Metadata Publishing, Maintenance and Classification Methodology

Social classification is the process by which a community of users categorizes the resources in that community for their own use [3]. One objective of our research is to construct an online service-based community in which 1) service providers publish and edit their services, and 2) service requesters retrieve and evaluate the services. Chapter 6 and Chapter 8 provide the solution for the latter part of our objective. In this section, we propose a solution for the former part of our objective.

The core of the methodology is a service-provider-based SDE metadata classification methodology. Although Chapter 5 presents an automatic classification method to solve this issue, due to personal differences whereby everyone has his/her own opinion about the classification, the classification method still lacks the capability of resolving differences in understanding. Therefore, we propose a service-provider-based SDE metadata classification methodology, which adopts the notion of social classification in the metadata-concept association process [1].

The model performs a step-to-step process for the manual association between SDE metadata and service concepts. The workflow of the association process is shown in Fig. 9.2.



Start
Access a SDE metadata
SDE Metadata Base
Display a linked concept list
Modify the linked concept list?
Y
Remove concepts?
N
Y
Remove selected concepts
Add new concepts?
Y
Search and select relevant concepts
Execute the modifications
Service Ontology Base
N
N
Add selected concepts
Are the modifications valid?
Y
Display a failure information
N
Exit

Figure 9.2: Workflow of the service provider-based SDE metadata classification methodology

After a user (user right: service provider) logs in to the platform, s/he obtains the right to access his/her own SDE metadata. Once the service provider gains access to a SDE metadata, the concept list stored in the *linkedConcepts* property of the metadata is obtained from the SDE Metadata Base and displayed to the user. The user can then decide whether or not s/he needs to modify the concept list. If the answer is "*yes*", the user needs to determine whether or not s/he needs to remove concepts from the concept list. If the answer is "*yes*", the user may then delete the desired concepts. Following this, the user is required to decide whether or not s/he needs to add concepts to the concept If the answer is "*yes*", the user can employ the Service Search Engine (introduced in Chapter 6) to retrieve the desired concepts from the Service Ontology Base, and add these concepts to the concept list. After all modifications have been completed and submitted by the user, a validation module will run to validate the URIs of concepts in the concept list. Once the modifications have passed the validation, the *linkedConcepts* property values of the metadata and the *linkedMetadata* property values of the corresponding concepts are updated [1].

In Chapter 10, we exhibit the prototype for the service-provider-based SDE metadata publishing, maintenance and classification methodology.

9.4 Conclusion

In this chapter, we propose two methodologies: 1) a service domain knowledge updating methodology, and 2) a service-provider-based SDE metadata publishing, maintenance, and classification methodology. The two methodologies collaborate to update the data within the Service Knowledge Base and the QoS Database, in order to allow our design to adapt to the dynamic nature of services and subjective perceptions of service providers and service requesters in the Digital Ecosystems environment.

The first methodology uses the notation of Web 2.0 to update service domain knowledge to adapt to the dynamic nature of services and service providers as well as service requesters. Domain knowledge updating involves changes of ontologies and changes of concept-based QoS evaluation criteria. The updating is done by achieving agreements between normal users and domain experts via a voting mechanism.

The second methodology creates an interface for service providers to publish, maintain and classify SDE metadata that can be used to represent their provided service entities. Of them, the service-provider-based SDE metadata classification methodology can be viewed as the solution for making agreements between the automatic SDE metadata classification methodology proposed in Chapter 5 and the subjective perceptions of service providers.

Therefore, from Chapter 5 to Chapter 9, we presented the technical details regarding the each portion of the proposed customized semantic service retrieval methodology. In Chapter 10, we present the evaluation details of the whole methodology.

9.5 References

1. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

2. Gendarmi, D., Lanubile, F.: Community-driven ontology evolution based on folksonomies. In: R. Meersman, Z.T., P. Herrero et al. (ed.): On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops. Springer-Verlag Berlin Heidelberg (2006) 181-188

3. Mathes, A.: Folksonomies: Cooperative classification and communication through shared metadata. University of Illinois Urbana-Champaign, Champaign, USA (2004)
4. O'Reilly, T.: What is Web 2.0? O'Reilly (2005), http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-Web-20.html

5. Sabbouh, M., Jolly, S., Allen, D., Silvey, P., Denning, P.: Workshop on Web services. World Wide Web Consortium, San Jose, USA (2001), http://www.w3.org/2001/03/WSWS-popa/paper08

Chapter 10 – Validation of the Proposed Methodology

10.1 Introduction

In this chapter, we validate the customized semantic service retrieval methodology by means of simulation and functional testing. As stated in Chapter 3, simulation is defined by Hevner et al. [13] as an evaluation approach by executing a prototype with artificial data, and functional testing is used to execute a prototype to discover shortcomings and identity defects. Therefore, for the mathematical algorithms proposed in this thesis, we validate them by using simulation; and we validate the other proposed functions of this methodology by using the functional testing approach.

In order to validate our methodology, we built a prototype which we term the Customized Semantic Service Search Engine (CSSSE) (Fig. 10.1). In accordance with the components of the customized semantic service retrieval methodology, the CSSSE prototype encompasses seven basic parts as follows:

1. Service Knowledge Base. It is the prototype of the Service Knowledge Base introduced in Chapter 5. Its technical details are introduced in Section 10.2.

2. Semantic Crawler. It is the prototype of the service information discovery, annotation and classification methodology introduced in Chapter 5. Its technical details are introduced in Section 10.3.

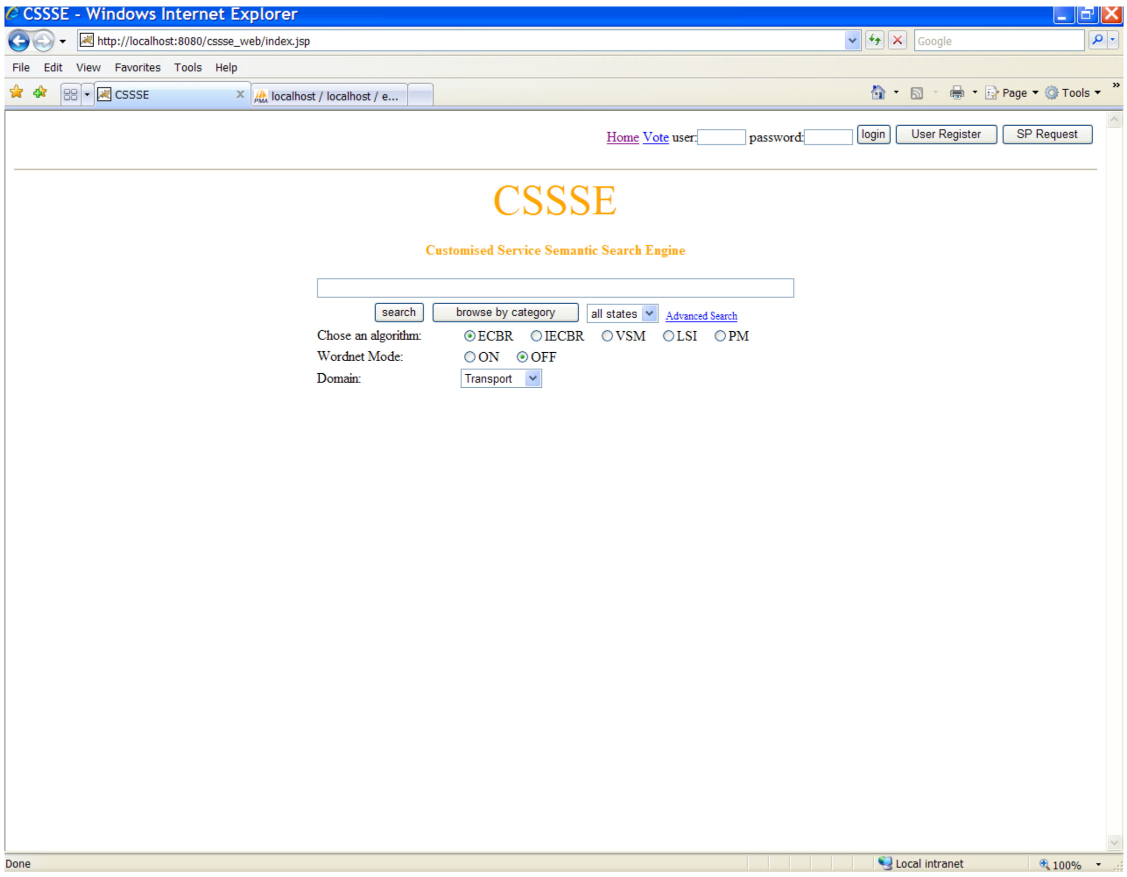
3. Service Search Engine. It is the prototype of the service retrieval methodology introduced in Chapter 6. Its technical details are introduced in Section 10.4.

4. Service Concept Recommendation System. It is the prototype of the service concept recommendation methodology introduced in Chapter 7. Its technical details are introduced in Section10.5.

5. QoS Evaluation and Service Ranking System. It is the prototype of the QoS evaluation and service ranking methodology introduced in Chapter 8. Its technical details are introduced in Section 10.6.

6. Service Domain Knowledge Updating System. It is the prototype of the service domain knowledge updating methodology introduced in Chapter 9. Its technical details are introduced in Section 10.7.

7. Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System. It is the prototype of the service-provider-based SDE metadata publishing, maintenance and classification methodology introduced in Chapter 9. Its technical details are introduced in Section 10.7.


CSSSE - Windows Internet Explorer
http://localhost:8080/cssse_web/index.jsp
Google
File Edit View Favorites Tools Help
CSSSE
localhost / localhost / e...
Page Tools
Home Vote user: password:
login
User Register
SP Request
CSSSE
Customised Service Semantic Search Engine
search
browse by category
all states
Advanced Search
Done
Local intranet
100%

Chose an algorithm: [✓ ECBR] IECBR VSM LSI PM
Wordnet Mode: ON [✓ OFF]
Domain: Transport

Figure 10.1: Screenshot of the CSSSE system

The prototype is built primarily using the following tools:

1. Eclipse 3.3. Eclipse is an Integrated Development Environment for developing applications in Java and, by means of plug-ins, in other languages as well. In this experiment, Eclipse is used as the platform to develop the whole CSSSE prototype in Java.

2. Java Development Kit (JDK) 1.6. JDK is used as a Java Software Development Kit (SDK) for the implementation of the whole CSSSE prototype.

3. Protégé 3.2.1. Protégé is an open source ontology editor, which is used as the platform for developing service ontologies in the Service Knowledge Base.

4. Apache Tomcat 6.0. Apache Tomcat is an open source software implementation of Java Servlet and JaveServer Pages, which provides a HTTP Web server environment for Java code to run. In this experiment, Apache Tomcat enables the Web server functions of the Service Search Engine, the Service Concept Recommendation System, the QoS Evaluation and Service Ranking System, the Service Domain Knowledge Updating System, and the Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System.

5. MySQL 5.4. MySQL is an open source database software, which provides supports for the data storage in the QoS Evaluation and Service Ranking System and Service Domain Knowledge Updating System.

6. Asynchronous JavaScript and XML (AJAX). AJAX is a group of interrelated Web development techniques used on the client-side to create interactive Web applications. In this experiment, AJAX is used to realize the service concept retrieval function of the Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System.

7. Protégé-OWL API. Protégé-OWL API is an open-source Java library for the Web Ontology Language and RDF(S). In this experiment Protégé-OWL API is used to load an OWL-coded service ontology.

8. Jena. Jena is a Java framework for building Semantic Web applications. In this experiment Jena is used to load an RDF(S)-coded service ontology.

9. Java WordNet Library (JWNL) 14-rc2. JWNL is a Java API for accessing WordNet relational dictionary. In this experiment, JWNL is used to find synonyms for query words in the Service Search Engine and in the service concept retrieval function of the Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System.

In this chapter, we present the whole validation process for the customized semantic service retrieval methodology. In Section 10.2, we select two Digital Ecosystem service domains as the basis for constructing the Service Knowledge Base. In addition, these two domains are regarded as the domains for empirically validating the whole customized semantic service retrieval methodology. In Section 10.3, we validate the service information discovery, classification and annotation methodology (the core of this methodology is a Semantic Crawler so the methodology can be abbreviated as "Semantic Crawler") by means of functional testing and simulation. In Section 10.4, we validate the service retrieval methodology by means of simulation and functional testing. In Section 10.5, we validate the concept recommendation methodology, by means of simulation. In Section 10.6, we validate the QoS evaluation and service ranking methodology by means of functional testing. In Section 10.7, we validate the service domain knowledge updating methodology and the service-provider-based SDE metadata publishing, maintenance and classification methodology, by means of functional testing. Finally, in Section 10.8 we summarize the whole validation process and draw our conclusions.

10.2 Domain Selection for Service Knowledge Base Construction and Methodology Validation

As stated in Section 10.1, before we validate the whole customized semantic service retrieval methodology, we need to choose a specific service domain from the Digital Ecosystem service domains, in order to create service domain ontologies for the Service Ontology Base. Furthermore, the selected service domains are treated as the domains in which we validate our methodology.

Here we choose two independent service domains: the transport service domain and health service domain. The reason for our choice is that these two domains are both comprehensive service domains about which we have knowledge and thus it is convenient for us to construct service ontologies. Services from these two service domains play important roles in daily life, and we frequently need to search for these services. Moreover, each service domain is very important for its own specific reasons

First, transport is a crucial component of logistics, which connects each activity in the logistics chain. It is believed that more than one third of cost in logistics is spent on transport [25]. In addition, the performance of logistics relies highly on the efficiency of transport services. However, to date there has not been a clear definition about transport services. Moreover, due to the rapid rise of fuel price, the input-output ratio of transport has been trapped in a troublesome situation, which causes the high cost of logistics and presents barriers to the global economic development. Therefore, research on transport services and transport service efficiency is urgently required. Therefore, we propose to use Semantic Web technologies to construct the conceptual model of transport services and thus present a clear structure of the transport service hierarchy. Furthermore, we propose a semantic search engine based on this transport service ontology, in order to provide an efficient tool for users to query transport service providers. This design can also be seen as a potentially alternative approach for solving the issue of transport service inefficiency and enhancing the global economic development [6].

Second, as one of its subdomains, the Digital Health Ecosystems inherit similar features from the Digital Ecosystems [12]. Because of the diversity and the geographical distribution of the health services, a species needing a service may not have the knowledge regarding the service request and thus might not timely find the species who can provide the service. Considering that time is crucial to health services, the Digital Health Ecosystems require a reliable means of establishing an efficient and time-conscious link between the diverse and distributed species, in order to facilitate the species' survival. Hence, there is an urgent need for a semantic service search engine which is able to provide a reliable and quick health service retrieval service within the Digital Health Ecosystems.

Based on the reasons above, we chose these two service domains and created initial ontologies for them.

First, in order to create the transport service ontology, we surveyed more than 1000 Australian transport companies' websites, and referred the classification systems in the Wikipedia website (http://en.wikepedia.org), as well as the Open Directory Project (ODP) website (http://www.dmoz.org). We then created the ontology by means of OWL.

The OWL-coded transport service ontology is a four-tier hierarchical structure which consists of 304 transport service concepts. Each concept is defined by a certain number of *conceptDescription* properties and these concepts are related by a concept-subconcept relationship. The first tier is the root of hierarchy, which is the abstraction of all services in the transport service domain. The second tier defines four basic subdomains of transport services – air transport services, rail transport services, road transport services and shipping services. The third tier provides abstract or concrete service concepts for each subdomain. The fourth tier is the specification of the abstract service concepts in the third tier. As discussed in Chapter 5, only the concrete concepts have the property of *linkedMetadata*, which are allowed to associate with SDE metadata [8]. The abbreviated view of the transport service ontology is depicted in Fig.10.2.

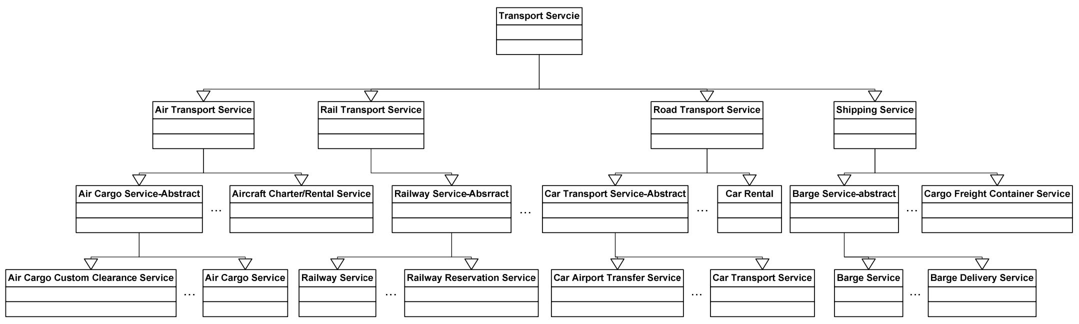


Figure 10.2: Abbreviated view of the transport service ontology

Analogously, in order to obtain the health service domain knowledge for designing a health service ontology, we have conducted a survey of over 1000 Australian health service companies' and organizations' websites, and referred the relevant knowledge from the Wikipedia and ODP website. We then construct the ontology in terms of RDF(S). We use RDF(S) because we want to test the feasibility and adaptability of our methodology in multiple ontology mark-up language environments.

The RDF(S)-coded health service ontology is a four-tier structure with 218 abstract or concrete concepts. The root concept defines the boundary of the general health service domain and the properties of a generic health service. The second-tier concepts define the six health service subdomains, which are allied health service, dentistry service, medicine service, nursing service, pathological and clinical laboratory service, and hospital and clinic service. The third-tier and fourth-tier concepts are the abstract or concrete service concepts in each subdomain, which inherit the properties from its parent concepts and also have own domain-specific properties [9]. An abbreviated view of the health service ontology is given in Fig. 10.3.

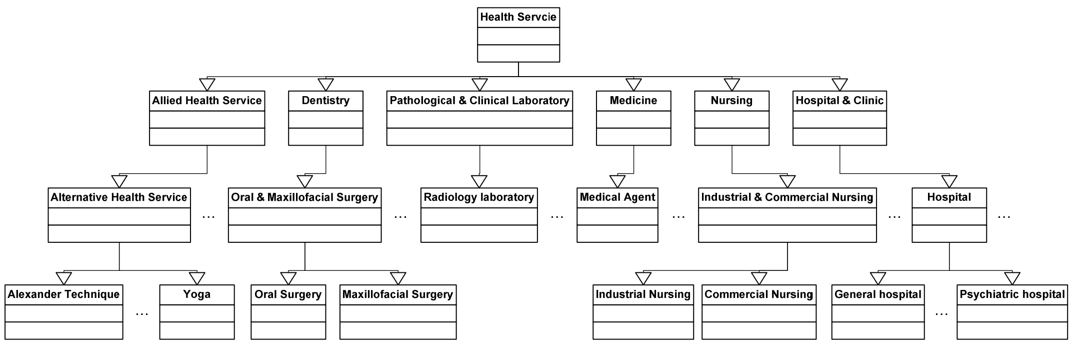


Figure 10.3: Abbreviated view of the health service ontology

It is noted that these two ontologies are only initial knowledge models that can be used to represent service domain knowledge in the selected service domains. It is difficult to guarantee that the designed ontologies can precisely and comprehensively represent all the knowledge in the service domains to which they belong. However, our proposed service domain knowledge updating methodology allows domain experts and normal users to be involved in future changes to the ontologies, in order to minimize the gaps between the ontologies and the domain knowledge.

10.3 Semantic Crawler Functional Testing and Simulation

It is known that the Semantic Crawler is the prototype of the service information discovery, annotation and classification methodology. Since the service information discovery and annotation methodology is realized by designing a system (Semantic Crawler) and defining a series of mechanisms and rules for guiding the interactions among the parts within the system, we need to run the prototype to test its functions in order to validate the methodology. This can be achieved by observing the quality of the output of this methodology. For the service information classification methodology which are realized by two algorithms (ECBR and IECBR), we use the approach of simulation to validate this methodology. In order to realize the simulation, we compare the performance of these two algorithms with a classical classification algorithm, based on several information retrieval performance indicators.

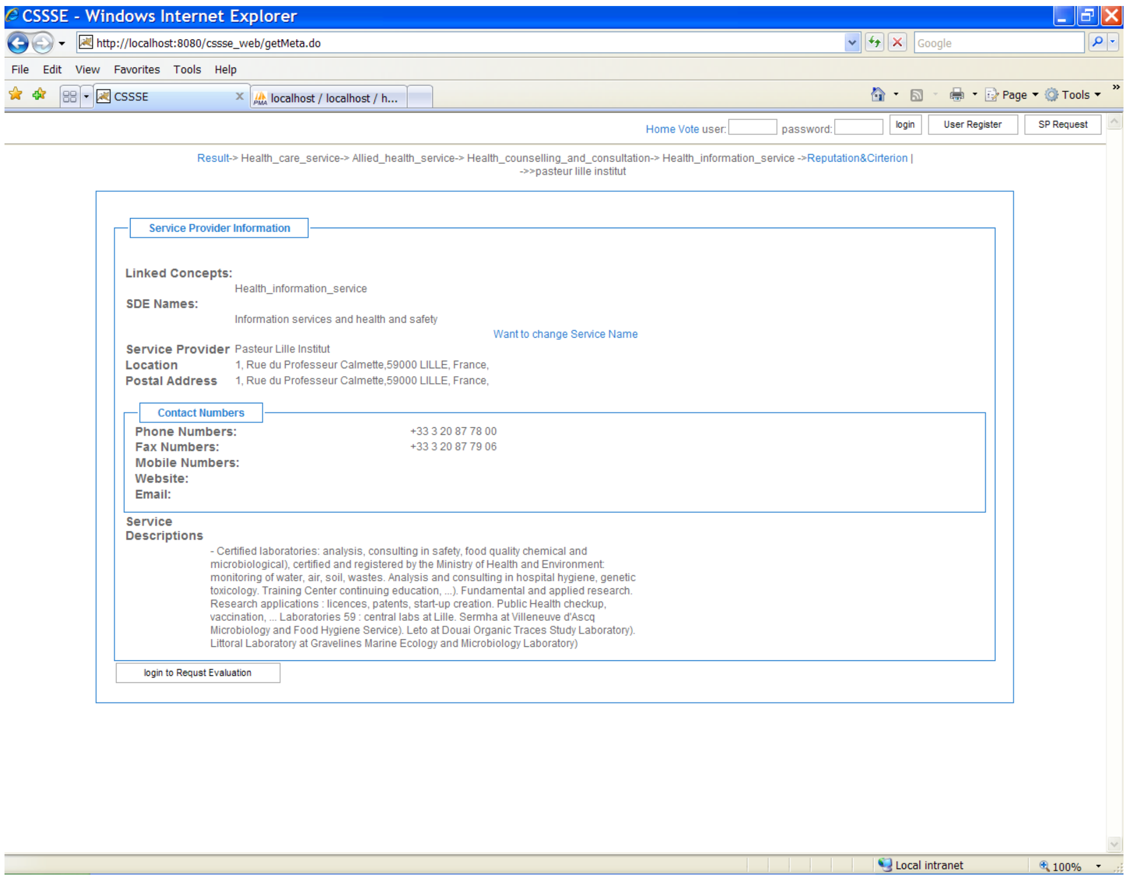
10.3.1 Experimental Environment Setup

One objective of the Semantic Crawler is to discover the service information from the Web, and then annotate the service information to SDE metadata. We choose the Australian Yellowpages® website (http://www.yellowpages.com.au) as the data source for service information. First of all, we run the Semantic Crawler facilitated with the transport service ontology on the website. We allow the Semantic Crawler to download 4000 business webpages under the category of transport from this website, and it generates 8114 SDE metadata for the transport service domains. Second, we facilitate the Semantic Crawler with the health service ontology and run it on the website. We allow the crawler to download 4000 business webpages from the category of health, and it generates 7639 SDE metadata for the health service domain.

10.3.2 Semantic Crawler Functional Testing

In order to validate the functions of the Semantic Crawler in service information discovery and annotation, we run the Semantic Crawler in our experimental data source as stated in Section 10.3.1, followed by observing the quality of the automatically generated SDE metadata from the data source.

The result shows that the generated SDE metadata with the defined properties are able to represent the features of an actual service advertisement from the Web. Fig. 10.4 displays an example of a generated SDE metadata. From the figure, it is found that every property of the SDE metadata shows correct information as it is in the website. Therefore, it can be deduced that our service information discovery and annotation methodology is valid in this experiment.


CSSSE - Windows Internet Explorer
http://localhost:8080/cssse_web/getMeta.do
Google
File Edit View Favorites Tools Help
CSSSE
localhost / localhost / h...
Page
Tools
Home Vote user:
password:
login
User Register
SP Request
Local intranet
100%

Result-> Health_care_service-> Allied_health_service-> Health_counselling_and_consultation-> Health_information_service ->Reputation&Cirterion |
->>pasteur lille institut

Service Provider Information

Linked Concepts:
Health_information_service

SDE Names:
Information services and health and safety

Want to change Service Name

Service Provider Pasteur Lille Institut
Location 1, Rue du Professeur Calmette,59000 LILLE, France,
Postal Address 1, Rue du Professeur Calmette,59000 LILLE, France,

Contact Numbers

Phone Numbers:	+33 3 20 87 78 00
Fax Numbers:	+33 3 20 87 79 06
Mobile Numbers:	
Website:	
Email:	

Service Descriptions
- Certified laboratories: analysis, consulting in safety, food quality chemical and microbiological), certified and registered by the Ministry of Health and Environment: monitoring of water, air, soil, wastes. Analysis and consulting in hospital hygiene, genetic toxicology. Training Center continuing education, ...). Fundamental and applied research. Research applications : licences, patents, start-up creation. Public Health checkup, vaccination, ... Laboratories 59 : central labs at Lille. Sermha at Villeneuve d'Ascq Microbiology and Food Hygiene Service). Leto at Douai Organic Traces Study Laboratory). Littoral Laboratory at Gravelines Marine Ecology and Microbiology Laboratory)

login to Requst Evaluation

Figure 10.4: Example of an automatically generated SDE metadata by the Semantic Crawler

10.3.3 Performance Indicators for Semantic Crawler Simulation

In order to thoroughly evaluate the performance of our ECBR and IECBR algorithms, we employ eight indicators from the field of information retrieval, which are: harvest rate, precision, mean average precision, recall, F-measure, F-measure$_\beta$, fallout and crawling time. Here we provide their definitions for the forthcoming experiments.

Harvest rate in the information retrieval is used to measure the crawling ability of a crawler. In this experiment, we define harvest rate as the proportion of associated SDE metadata in the whole collection of generated SDE metadata [9], which can be mathematically represented as:

$$\text{Harvest Rate} = \frac{\text{Number of associated SDE metadata}}{\text{Number of generated SDE metadata}} \tag{10.1}$$

Precision in the information retrieval is used to measure the precision of a retrieval system [1]. In this experiment, we define precision for a single service concept as the proportion of the relevant SDE metadata associated by this concept in all the SDE metadata associated by this concept [9], which can be mathematically represented as:

$$\text{Precision(S)} = \frac{\text{Number of associated and relevant SDE metadata}}{\text{Number of associated SDE metadata}} \tag{10.2}$$

With regard to the whole collection of service concepts in a service ontology, we define that the precision is the sum of the precision value for each concept normalized by the number of concepts in the collection [9], which can be represented as:

$$\text{Precision(W)} = \frac{\sum_{i=1}^{n} \text{Precision}(\text{S}_i)}{n} \tag{10.3}$$

Before we introduce the definition of Mean Average Precision, the concept of Average Precision should be defined. Average Precision for a single service concept is defined by us as the average of precision values after truncating a ranked SDE metadata list associated by this concept after each of the relevant metadata for this concept [9]. This indicator emphasizes the return of more relevant results earlier [1], which can be represented as:

$$\text{Average Precision(S)} = \frac{\text{Sum(Precision @ Each relevant SDE metadata in the list)}}{\text{Number of associated and relevant SDE metadata in the list}} \tag{10.4}$$

We define Mean Average Precision as the average of the average precision values for the collection of service concepts in a service ontology [9], which can be represented as:

$$\text{Mean Average Precision} = \frac{\sum_{i=1}^{n} \text{Average Precision}(\text{S}_i)}{n} \tag{10.5}$$

Recall that the information retrieval refers to the measure of effectiveness of a query system [1]. In this experiment, we define that recall for a single service concept is the proportion of the relevant SDE metadata associated by this concept in all the relevant metadata of this concept in the collection of generated SDE metadata [9], which can be represented as:

$$\text{Recall(S)} = \frac{\text{Number of associated and relevant SDE metadata}}{\text{Number of relevant SDE metadata}} \tag{10.6}$$

With regard to the whole collection of service concepts in a service ontology, we define that the recall value is the sum of the recall value for each concept normalized by the number of concepts in the collection [9], which can be represented as:

$$\text{Recall(W)} = \frac{\sum_{i=1}^{n} \text{Recall}(\text{S}_i)}{n} \tag{10.7}$$

It is important to note that the number of relevant SDE metadata can be determined only by a peer-reviewed method, as the estimation of relevance between metadata and concept requires detailed knowledge of all concepts and metadata in the knowledge base, which can only be manually implemented in the current situation [9].

The F-measure (or F-measure(β=1)) in the information retrieval is used as an aggregated performance scale for a search system [1]. In this experiment, the F-measure value is the mean of precision value and recall value, which can be represented as:

$$\text{F-measure}(\beta{=}1) = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \tag{10.8}$$

When the F-measure reaches the highest point, it means the integrated value between precision and recall reaches the highest at the same time.

F-measure$_\beta$ is another measure that combines precision and recall, and the difference is that users can specify the preference on recall or precision by configuring different weights [18]. In this experiment, we employ F-measure(β=2) that weights recall twice as much as precision, which is close to the fact that most search engines concern recall more than precision, as a result of most users' purposes in obtaining information [24]. The F-measure(β=2) can be represented below as:

$$\text{F-measure}(\beta{=}2) = \frac{(1+\beta^2)\cdot \text{Precision} \times \text{Recall}}{\beta^2 \cdot \text{Precision+Recall}} = \frac{5 \times \text{Precision} \times \text{Recall}}{4 \times \text{Precision+Recall}} \tag{10.9}$$

All of the above indicators have the same limitation – they do not consider the amount of non-relevant SDE metadata in an associated SDE metadata collection. Furthermore, if there is no relevant metadata in the associated collection, Recall cannot be defined. To resolve this issue, we need another performance indicator – fallout [1]. In this experiment, fallout for a single service concept is the proportion of non-relevant SDE metadata associated by this concept in the whole collection of non-relevant metadata for this concept in the generated metadata [9], which can be represented as:

$$\text{Fallout(S)} = \frac{\text{Number of associated and non-relevant SDE metadata}}{\text{Number of non-relevant SDE metadata}} \tag{10.10}$$

With regard to the whole collection of service concepts, the fallout rate is the sum of the fallout rate for each concept normalized by the number of concepts in a service ontology [9], which can be represented as:

$$\text{Fallout(W)}=\frac{\sum_{i=1}^{n}\text{Fallout}(\text{S}_i)}{n} \qquad (10.11)$$

In contrast to other performance indicators, the lower the fallout value, the better is the crawler's performance [9].

Crawling time is an important metrics for evaluating the efficiency of a crawler, which is defined by us as the interval between the time of reading a Web document and the time of classifying all the metadata generated from the Web document [9].

10.3.4 Latent Semantic Indexing Algorithm for SDE Metadata Classification

In order to horizontally evaluate the performance of the ECBR and the IECBR algorithm, we adopt a Latent Semantic Indexing (LSI) algorithm, which is a classical algorithm used for information retrieval and document classification [2], in our proposed Semantic Crawler, with the purpose of SDE metadata classification. The implementation details are introduced as follows:

In the Service Ontology Base, first of all, each service concept C is regarded as a body of plain texts comprised of *conceptDescription* property(s). Following that, an index term list is obtained from all the concepts in a service ontology. Based on the index term list, each concept C is formed as an array in which each element corresponds to an index term from the index term list and the weight of the element is obtained by the term frequency-inverse document frequency (tf-idf) [23], and all the concepts in the service ontology are formed as a term-concept matrix A. The term-concept matrix is then decomposed by the Singular Value Decomposition (SVD) approach, which can be mathematically represented by Equation 10.12 [8].

$$A = U\Sigma V^T \qquad (10.12)$$

where U is the matrix derived from the term-to-term matrix given by AA^T, V^T is the matrix derived from the transpose of the concept-to-concept matrix given by A^TA, and Σ is a r×r diagonal matrix of singular values where r = min(t, N) is the rank of A [8].

Considering now that only k largest singular values of Σ are kept along with their corresponding columns in U and V^T, the resultant A_k matrix is the matrix of rank k which is closest to the original matrix A in the least square sense. This matrix is given by Equation 10.13.

$$A_k = U_k \Sigma_k V_k^T \tag{10.13}$$

where k (k < r) is the dimensionality of a reduced concept space.

Analogous to the concept, a SDE metadata M can be regarded as a body of plain texts comprising its *serviceDescription* property(s). The metadata can thus be represented as an index term-based array in which each element is the tf-idf weight between the metadata and a term from the index term list. The array can then be translated into the concept space by Equation 10.14, and then compared with A_k by the cosine algorithm to calculate the similarity values with each concept, which can be represented by Equation 10.15 [8].

$$M' = \Sigma_k^{-1} U_k^T M \tag{10.14}$$

$$sim(C, M) = \frac{A_k \cap M'}{|A_k| \times |M'|} \tag{10.15}$$

10.3.5 Semantic Crawler Simulation in Transport Service Domain

In this section, we compare the performance of the ECBR and IECBR algorithms with the LSI algorithm introduced in Section 10.3.4 in the transport service domain, based on the eight performance indicators introduced in Section 10.3.3, including harvest rate, precision, mean average precision, recall, F-measure(β=1), F-measure(β=2), fallout and crawling time. In addition, as stated in Chapter 5, both of the ECBR and IECBR algorithms need an optimal threshold value in order for these two algorithms to obtain their best performance. Hence, in this experiment, we have another task – choosing optimal threshold values for the ECBR and IECBR algorithms based on their best performance on these indicators. Therefore, we set the initial threshold value at 0.5, and increase it by 0.05 each time. Then we measure the performance scores of the ECBR, IECBR and LSI algorithms on the performance indicators at each time of the increment of the threshold value. Finally, this measurement is based on the experiment in the Australian Yellowpages® website introduced in Section 10.3.1.

Fig. 10.5 depicts the comparison of the ECBR, IECBR and LSI algorithms on harvest rate. Since the ECBR and the IECBR follow a similar principle with the only difference being their efficiency, these two algorithms have the same performance on harvest rate. It can be observed that there is a significant gap between the group of the ECBR and IECBR and the LSI. Furthermore, the gap enlarges (from 25.39% to 95.93%) along with the increase of the threshold value. Additionally, the curve of the ECBR and IECBR keeps relatively stable (from 100% to 99.22%), in contrast to the rapid fall of the LSI (from 74.61% to 3.29%), when the threshold value increases. This phenomenon results from the fact that the higher threshold values may filter more associated SDE metadata for the LSI. This experiment indicates that the Semantic Crawler shows better crawling ability, when employing ECBR and IECBR other than the LSI. Moreover, the performance of the ECBR and IECBR on harvest rate is not heavily influenced by the variation of the threshold value, compared with the LSI.

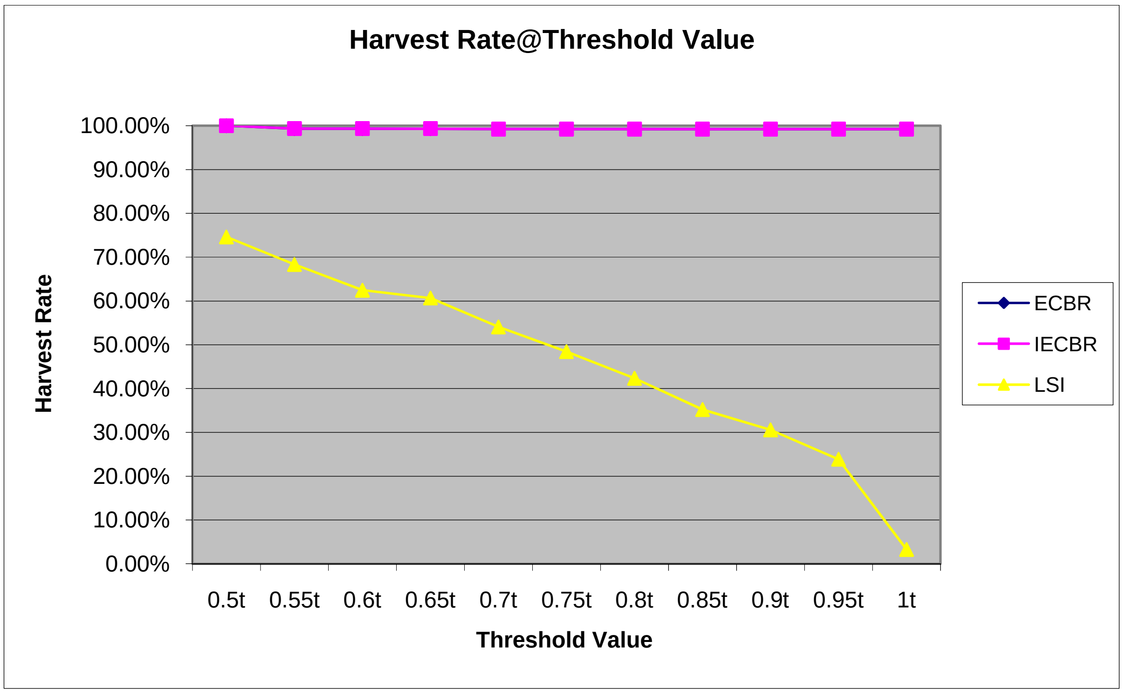


Figure 10.5: Comparison of the ECBR, IECBR and LSI algorithms on harvest rate in the semantic crawler simulation in the transport service domain

Fig. 10.6 displays the comparison of the ECBR, IECBR and LSI algorithms on precision. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have the same performance on precision. First of all, the curves of the ECBR, IECBR and LSI all ascend along with the increase of the threshold value, because the higher threshold values may filter more non-relevant SDE metadata. The only fall occurred in the LSI curve because there is no associated SDE metadata for the LSI when the threshold value is 1. Second, the variation interval of the LSI (63.27%) is larger than that of the ECBR and IECBR (38.60%), which indicates that the former is more easily influenced by the variation of the threshold value than is the latter. Third, the IECBR and ECBR perform better than the LSI when the threshold value is 0.7 or 0.75, which shows that the Semantic Crawler that employs the former has a higher precision than does the latter at the relatively moderate threshold values.

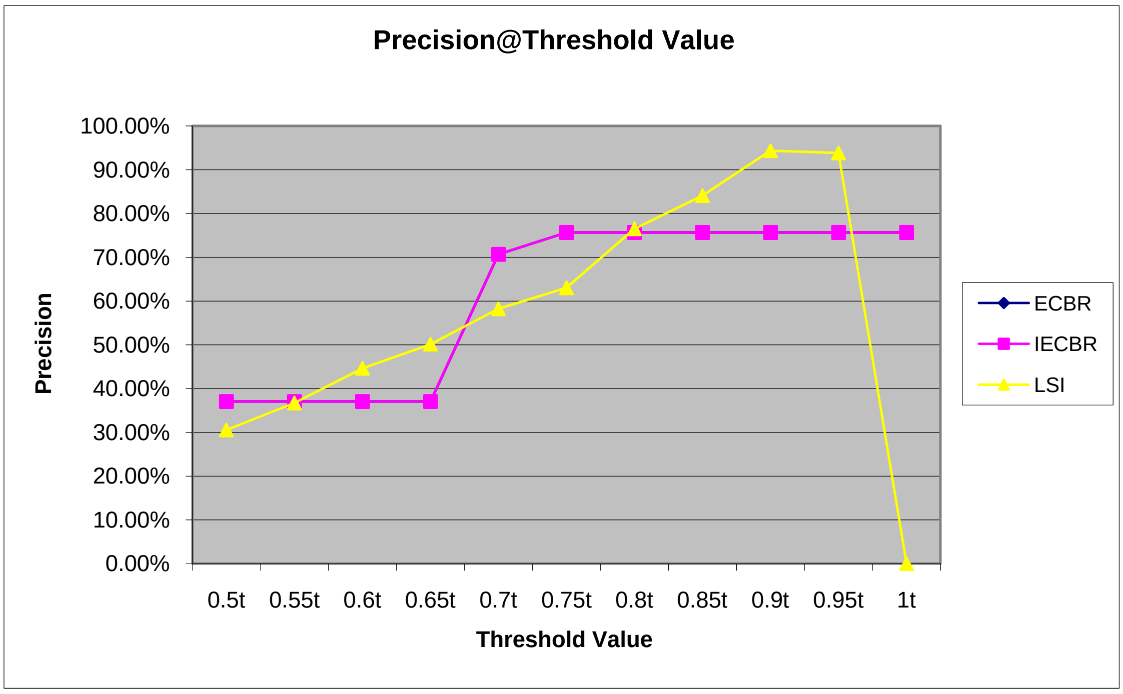


Figure 10.6: Comparison of the ECBR, IECBR and LSI algorithms on precision in the semantic crawler simulation in the transport service domain

Fig. 10.7 shows the comparison of the ECBR, IECBR and LSI algorithms on mean average precision. This figure shows curves similar to those in Fig.10.6. The difference is that scores of the ECBR, IECBR and LSI on means average precision are relatively higher than on precision. Similarly, three conclusions are deduced from the figure, which are: 1) the performance of the three algorithms are all positively affected by the increase of the threshold value; 2) the influence of the variation of the threshold value on the mean average precision of the LSI is more obvious than on the ECBR and IECBR; and 3) the performance of the ECBR and IECBR on mean average precision is better than the LSI at the relatively moderate threshold values (0.7 and 0.75).

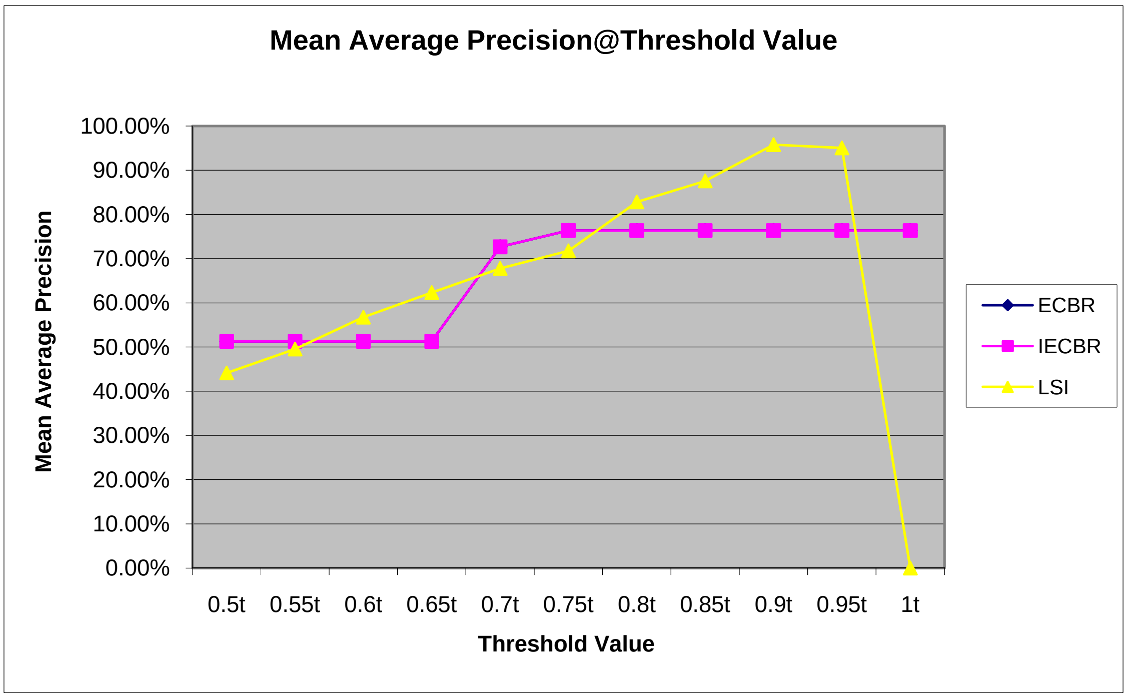


Figure 10.7: Comparison of the ECBR, IECBR and LSI algorithms on mean average precision in the semantic crawler simulation in the transport service domain

Fig. 10.8 reveals the comparison of the ECBR, IECBR and LSI algorithms on recall. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have the same performance on recall. It is found that the ECBR and IECBR perform far better than does the LSI, as the former retain more than 90% and the latter remains at less than 35% during the process of increasing the threshold value. This is because the Semantic Crawler filters more relevant SDE metadata when employing the LSI other than the ECBR and IECBR. This indicates the outstanding effectiveness of the ECBR and IECBR compared with that of the LSI. The relatively smaller variation interval of the ECBR and IECBR (1.47%) proves their stronger anti-jamming ability against the variation of the threshold value, in contrast to the bigger variation interval of the LSI (28.16%).

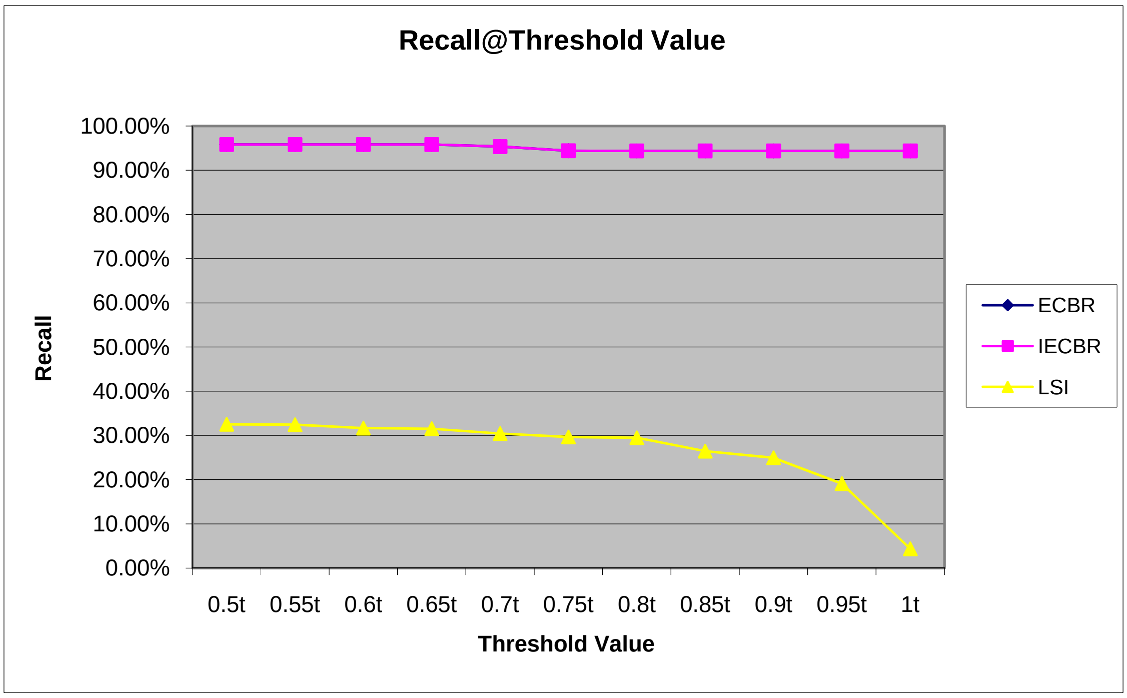


Figure 10.8: Comparison of the ECBR, IECBR and LSI algorithms on recall in the semantic crawler simulation in the transport service domain

Fig. 10.9 illustrates the comparison of the ECBR, IECBR and LSI algorithms on F-measure(β=1). It can be seen that the trend of the curves of the ECBR and IECBR is different from the LSI, where the former shows an up trend and the latter shows an up-down trend, along with the increase of the threshold value. This is because F-measure(β=1) is the aggregation of precision and recall and thus its score is affected by the scores of both precision and recall. For the ECBR and IECBR, since their precision values vary obviously and their recall values remain relatively stable, the trend of their F-measure(β=1) curves are primarily affected by the trend of the precision curves. Because the precision value and recall value of the LSI both vary visibly, its F-measure(β=1) curve displays a different trend from its precision and recall curve. Since its precision curve maintains an ascending trend and its recall curve maintains a descending trend, in the former period of the increase of the threshold value (from 0.5 to 0.8), the F-measure(β=1) curve keeps an ascending trend, which indicates that its precision curve has a heavier effect on the trend of the F-measure(β=1) curve than the recall curve, in the latter period of the increase of the threshold value (from 0.8 to 1.0), the F-measure(β=1) curve keeps an descending trend, which indicates that its recall curve has a heavier effect on the trend of the F-measure(β=1) curve than the precision curve. In addition, when we compare the F-measure(β=1) scores between the ECBR, IECBR and LSI, it can be found that the ECBR and IECBR have an overall advantage than the LSI, which illuminates that advantage of the ECBR and IECBR on recall dominates the comparison. Therefore, we can deduce that the ECBR and IECBR have better aggregated performance than does the LSI.

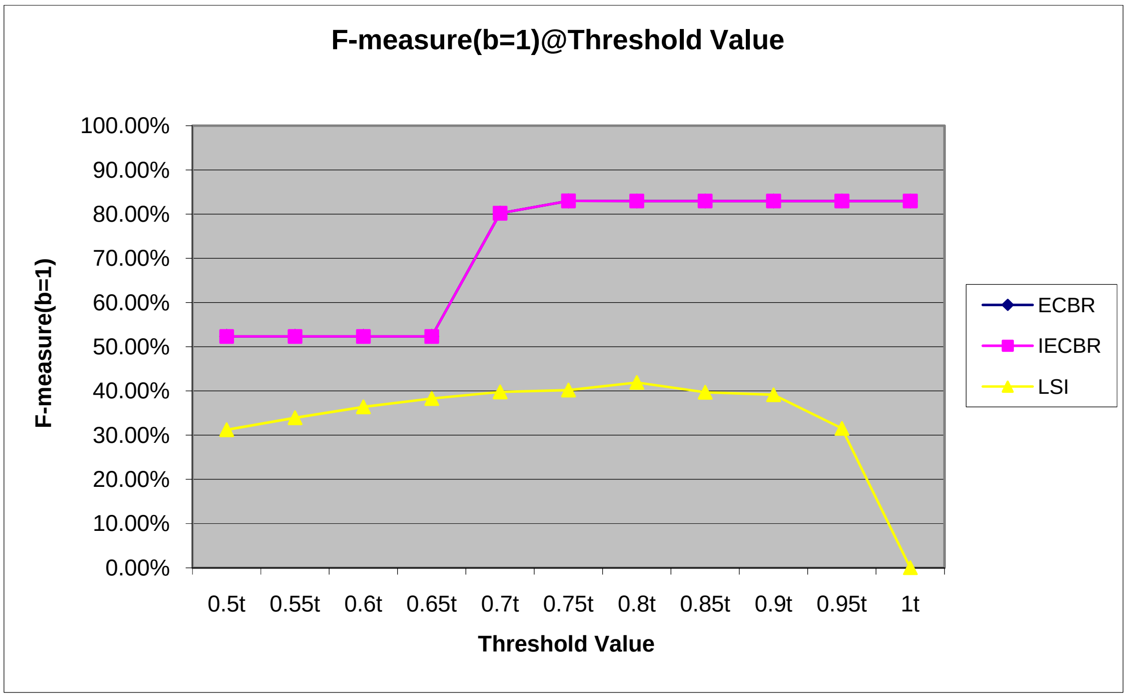


Figure 10.9: Comparison of the ECBR, IECBR and LSI algorithms on F-measure(β=1) in the semantic crawler simulation in the transport service domain

Fig. 10.10 shows the comparison of the ECBR, IECBR and LSI algorithm on F-measure(β=2). Here the F-measure(β=2) curves shows the similar trend as the F-measure(β=1) curves. The difference is that the F-measure(β=2) scores of the ECBR and IECBR are higher than their F-measure(β=1) scores and, in contrast, the F-measure(β=2) scores of the LSI are lower than its F-measure(β=1) scores. It is because F-measure(β=2) more emphasizes on recall, the performance of the ECBR and IECBR on recall is better than on precision, and the performance of the LSI on recall is worse than on precision. Consequently, the advantage of the ECBR and IECBR on F-measure(β=2) is more obvious than on the F-measure(β=1), compared with the LSI. By means of this comparison, it can be concluded that the ECBR and IECBR are more likely to fulfil the preference of most search engines on recall.

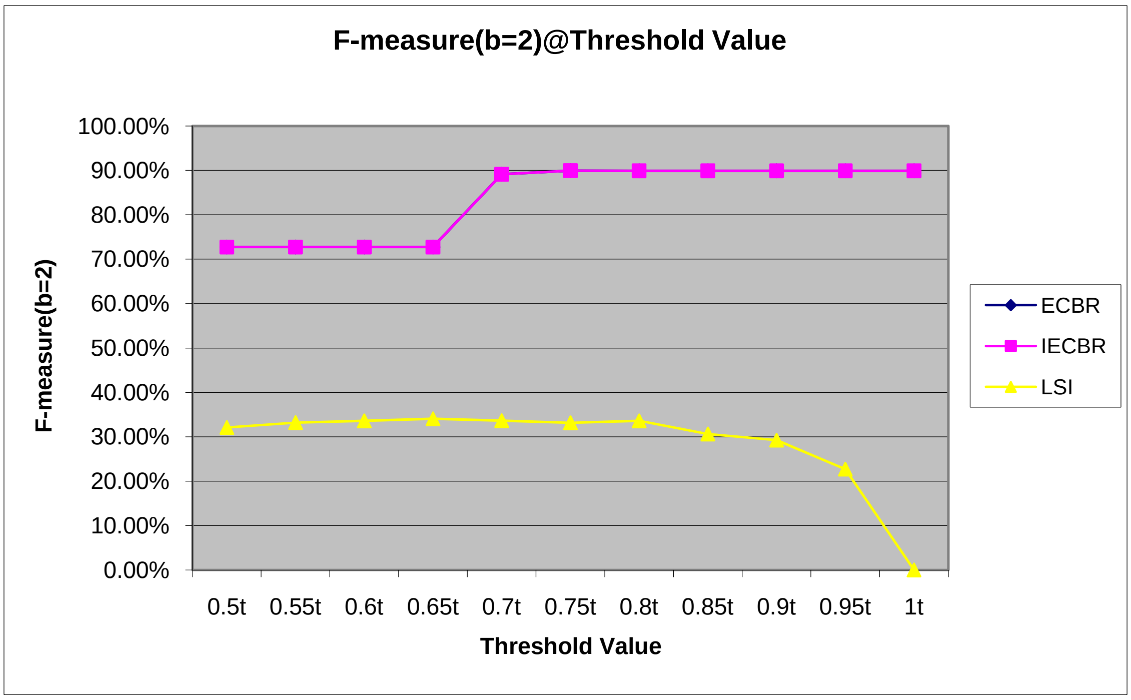


Figure 10.10: Comparison of the ECBR, IECBR and LSI algorithms on F-measure(β=2) in the semantic crawler simulation in the transport service domain

Fig. 10.11 presents the comparison of the ECBR, IECBR and LSI algorithms on fallout. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have the same performance on fallout. Three conclusions are obtained from the comparison. First, the three curves all show a descending trend, since the higher threshold value may filter more associated and non-relevant SDE metadata. Second, the variation intervals of the ECBR and IECBR (1.02%) are bigger than the LSI (0.35%). Third, the ECBR and IECBR performs better than does the LSI on the relatively moderate threshold values (from 0.7 to 0.85); otherwise not. Therefore, the selection of the threshold value directly determines which model performs better on this indicator.

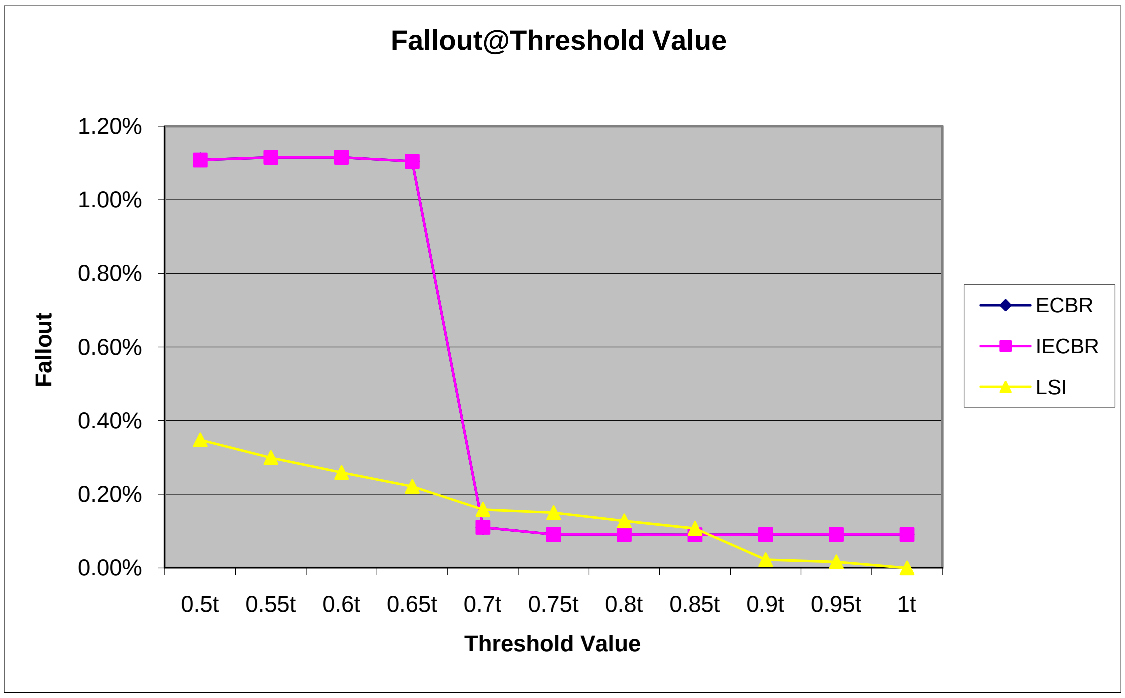


Figure 10.11: Comparison of the ECBR, IECBR and LSI algorithms on fallout in the semantic crawler simulation in the transport service domain

Fig. 10.12 shows the comparison of the ECBR, IECBR and LSI algorithms on crawling time. It can be clearly observed that the IECBR uses the least time when crawling the same number of webpages, followed by the ECBR and LSI. From this comparison, it can be deduced that the Semantic Crawler that employs the IECBR is faster than the ECBR, in addition to keeping the same performance on the other indicators, which preliminarily proves the success of the IECBR on efficiency improvement.

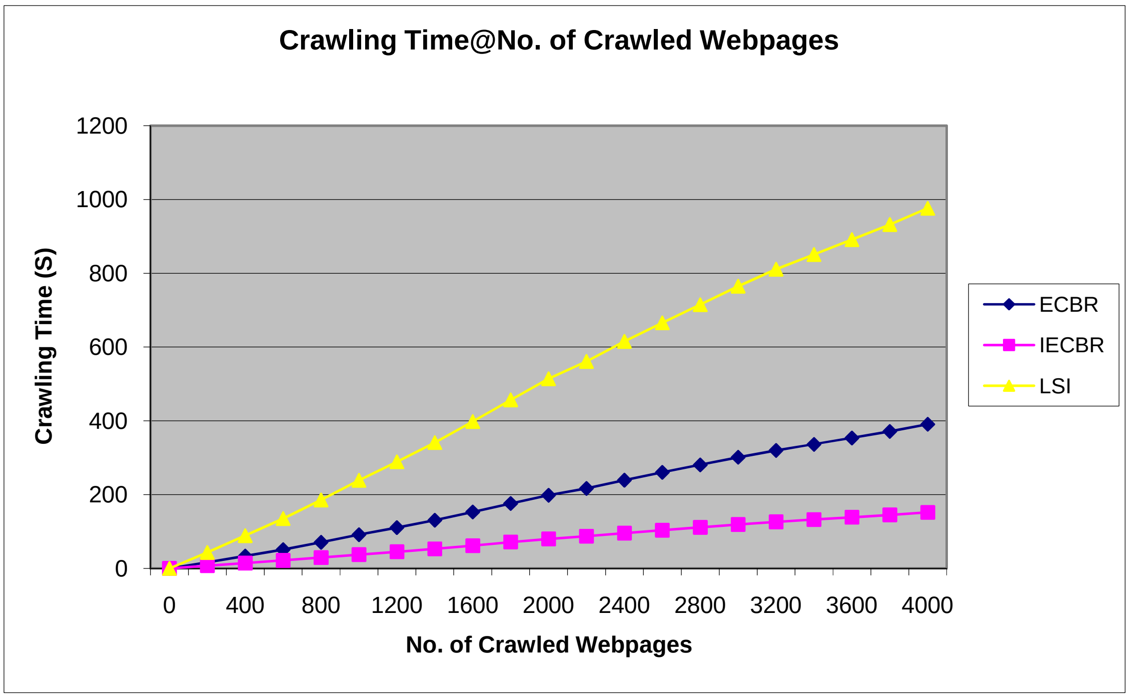


Figure 10.12: Comparison of the ECBR, IECBR and LSI algorithms on crawling time in the semantic crawler simulation in the transport service domain

The comparison results of the ECBR, IECBR and LSI algorithms in the semantic crawler simulation in the transport service domain can be summarized as follows:

1. The Semantic Crawler has better crawling ability when employing the ECBR and IECBR algorithms.

2. The Semantic Crawler has better precision when employing the ECBR and IECBR algorithms and the relatively moderate threshold values (0.7 and 0.75).

3. The Semantic Crawler is able to retrieve more relevant SDE metadata earlier when employing the ECBR and IECBR algorithm and the relatively moderate threshold values (0.7 and 0.75).

4. The Semantic Crawler has higher effectiveness when the ECBR and IECBR algorithms are used.

5. The Semantic Crawler has better aggregated performance when the ECBR and IECBR algorithms are used.

6. The Semantic Crawler is more likely to fulfil the preference of most search engines on recall when the ECBR and IECBR algorithms are used.

7. The Semantic Crawler has better performance on fallout when employing the ECBR and IECBR algorithms and the relatively moderate threshold values (from 0.7 to 0.85).

8. The Semantic Crawler has higher efficiency when the IECBR algorithm is used.

Therefore, we can conclude from this experiment that: 1) the ECBR and IECBR algorithm performs better than the LSI on the eight indicators when choosing the relatively moderate threshold values (0.7 and 0.75); and, 2) the IECBR algorithm is more efficient than the ECBR algorithm.

For the task of optimal threshold selection, since F-measure(β=1) and F-measure(β=2) are the aggregations of precision and recall, we can put more weight on these two performance indicators and thus reduce the weights of precision and recall, when determining the optimal threshold values. At the point of 0.75, the F-measure(β=1) and F-measure(β=2) scores of the ECBR and IECBR both reach to the maximum, and their fallout rates both reach to the minimum. Therefore, we choose 0.75 as the optimal threshold value for both the ECBR and IECBR algorithm.

10.3.6 Semantic Crawler Simulation in Health Service Domain

In this section, we compare the performance of the ECBR and IECBR algorithms with the LSI algorithm introduced in Section 10.3.4 in the health service domain, based on the eight performance indicators introduced in Section 10.3.3, namely harvest rate, precision, mean average precision, recall, F-measure(β=1), F-measure(β=2), fallout and crawling time. In addition, as stated in Chapter 5, both of the ECBR and IECBR algorithms need an optimal threshold value by which these two algorithms can obtain their best performance. Hence, in this experiment, we have another task – choosing optimal threshold values for the ECBR and IECBR algorithm based on their best performance on these indicators. Therefore, we set the initial threshold value at 0.5, and increase it by 0.05 each time. Then we measure the performance scores of the ECBR, IECBR and LSI algorithms using the performance indicators at each time of the increment of the threshold value. Finally, this measurement is based on the experiment in the Australian Yellowpages® website introduced in Section 10.3.1.

Fig. 10.13 depicts the comparison of the ECBR, IECBR and LSI algorithms on harvest rate. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have the same performance on harvest rate. It can be observed that there is a remarkable gap between the group of the ECBR and IECBR and the LSI. Furthermore, the gap enlarges (from 24.57% to 77.49%) along with the increase of the threshold value. Additionally, the curve of the ECBR and IECBR keeps relatively stable (from 99.93% to 91.56%), in contrast to the rapid fall of the LSI (from 70.49% to 14.07%), when the threshold value increases. This phenomenon is due to the fact that the higher threshold values may filter more associated SDE metadata for the This experiment indicates that the Semantic Crawler shows better crawling ability, when employing ECBR and IECBR compared with the LSI. Moreover, the performance of the ECBR and IECBR on harvest rate is not heavily influenced by the variation of the threshold value, compared with the LSI.

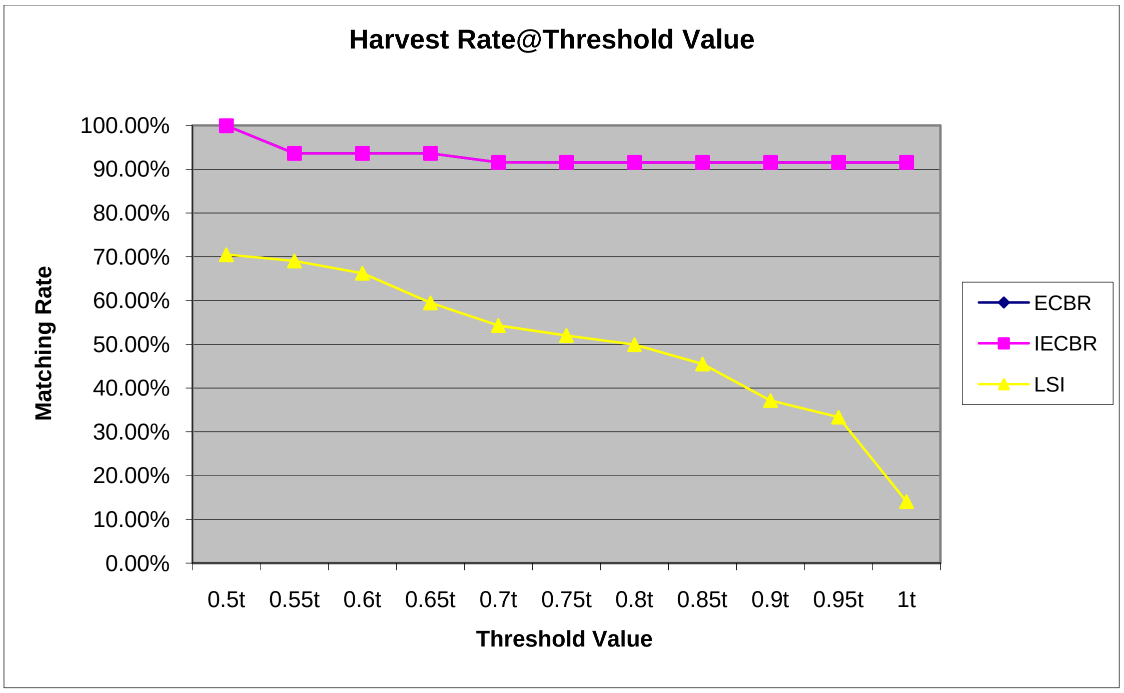


Figure 10.13: Comparison of the ECBR, IECBR and LSI algorithms on harvest rate in the semantic crawler simulation in the health service domain

Fig. 10.14 displays the comparison of the ECBR, IECBR and LSI algorithms on precision. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have same performance on precision. First of all, as a whole, the curves of the ECBR, IECBR and LSI all ascend along with the increase of the threshold value, owing to the reason that higher threshold values may filter more non-relevant SDE metadata. Second, except for the initial threshold value, the variation interval of the LSI (47.65%) is larger than the ECBR and IECBR (5.93%) on the other threshold values, which indicates that the former is more easily influenced by the variation of the threshold value than the latter when the threshold value is more than 0.5. Third, the performance of the ECBR and IECBR on precision are better than the LSI on most threshold values except for these two boundaries (0.5, 0.95 and 1), which shows that the Semantic Crawler that employs the former has a higher preciseness than the latter on the most threshold values (from 0.55 to 0.9).

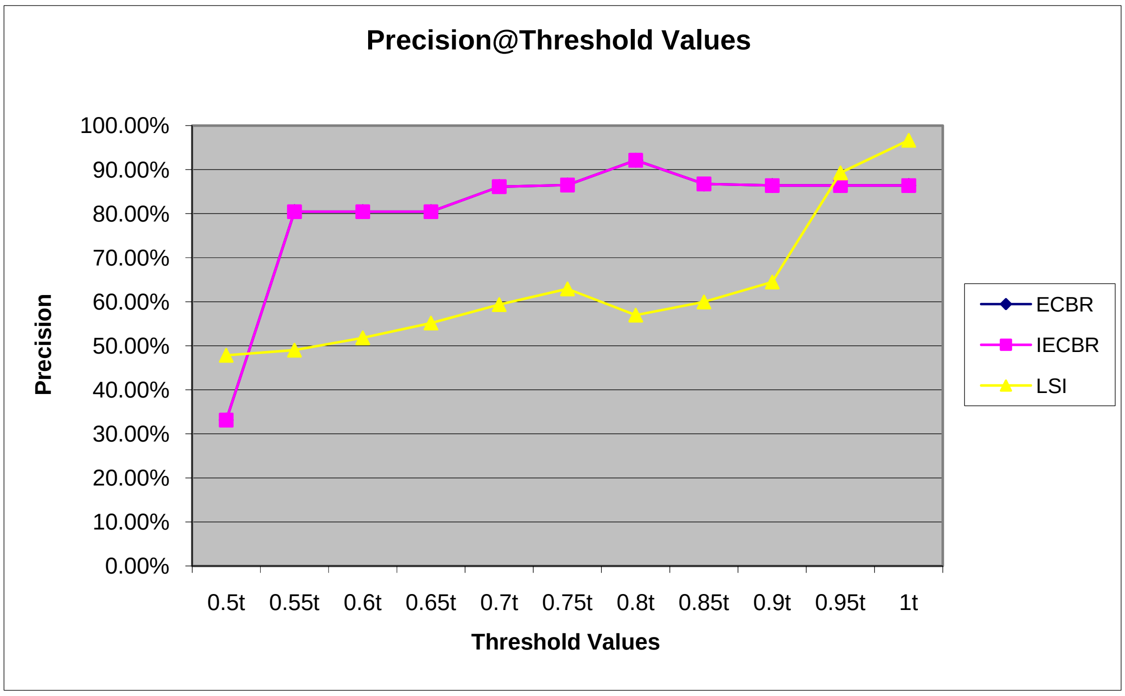


Figure 10.14: Comparison of the ECBR, IECBR and LSI algorithms on precision in the semantic crawler simulation in the health service domain

Fig. 10.15 shows the comparison of the ECBR, IECBR and LSI algorithms on mean average precision. This figure shows curves similar to those in Fig. 10.14. The difference is that scores of the ECBR, IECBR and LSI on means average precision are relatively higher than on precision. Similarly, three conclusions are deduced from the figure: 1) the performance of the three algorithms are all positively affected by the increase of the threshold value; 2) the influence of the variation of the threshold value on the mean average precision of the LSI is more obvious than on the ECBR and IECBR; and 3) the performance of the ECBR and IECBR on mean average precision is better than the LSI at most of the threshold values except 0.95 and 1.0.

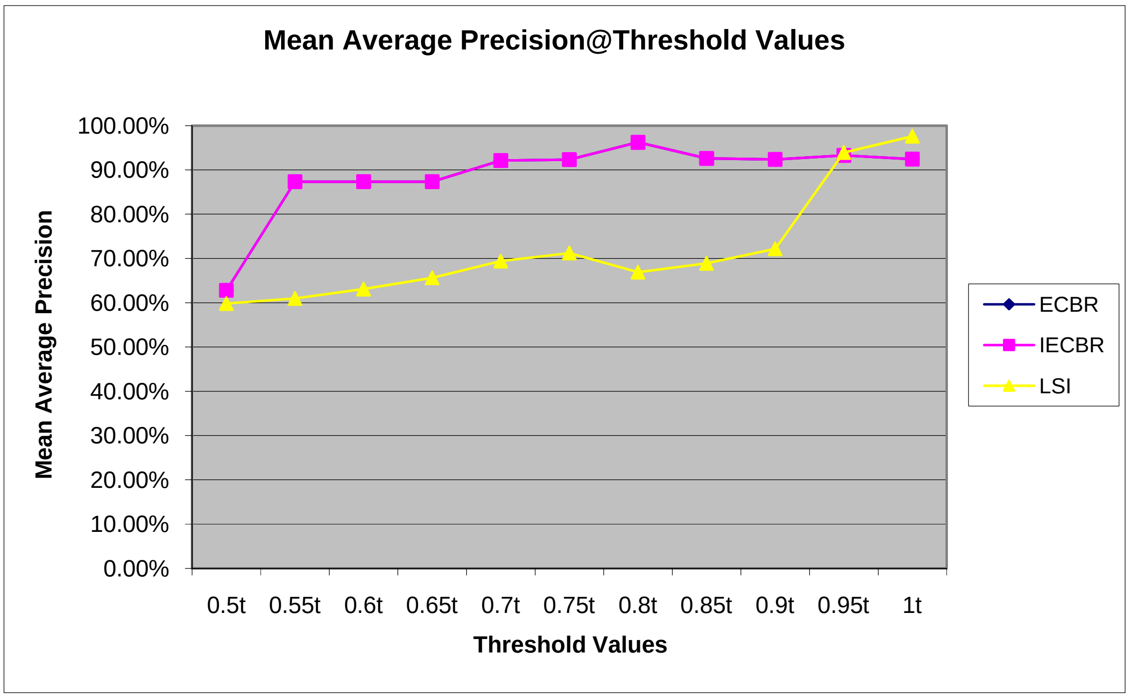


Figure 10.15: Comparison of the ECBR, IECBR and LSI algorithms on mean average precision in the semantic crawler simulation in the health service domain

Fig. 10.16 reveals the comparison of the ECBR, IECBR and LSI algorithms on recall. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have the same performance on recall. It is found that the ECBR and IECBR perform far better than does the LSI, as the former maintain more than 90% and the latter remains at less than 51% during the process of increasing the threshold value. This is because the Semantic Crawler filters more relevant SDE metadata when employing the LSI other than do the ECBR and IECBR. This indicates the outstanding effectiveness of the ECBR and IECBR compared with the LSI. The relatively smaller variation interval of the ECBR and IECBR (9.51%) proves their stronger anti-jamming ability against the variation of the threshold value, in contrast to the bigger variation interval of the LSI (30.59%).

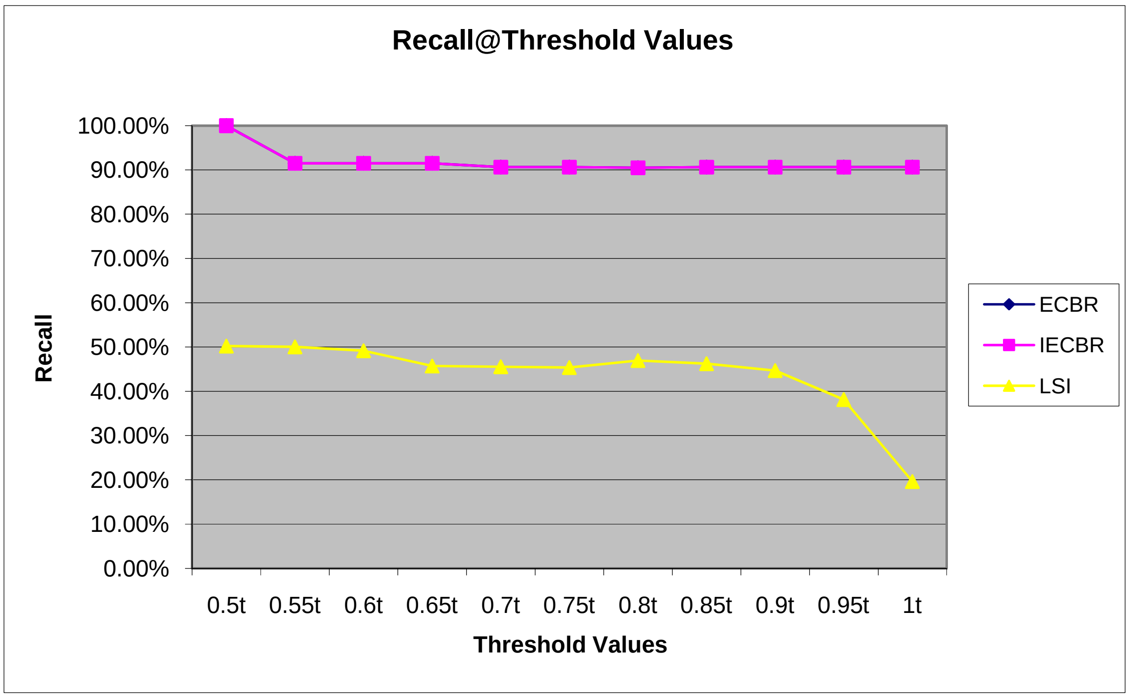


Figure 10.16: Comparison of the ECBR, IECBR and LSI algorithms on recall in the semantic crawler simulation in the health service domain

Fig. 10.17 illustrates the comparison of the ECBR, IECBR and LSI algorithms on F-measure(β=1). It can be seen that the trend of the curves of the ECBR and IECBR is different from the LSI, where the former shows an overall up trend and the latter shows an overall up-down trend, along with the increase of the threshold value. This is because the F-measure(β=1) is the aggregation of precision and recall and thus its score is affected by the scores of both precision and recall. For the ECBR and IECBR, since their precision values vary obviously and their recall values keeps relatively stable, the trend of their F-measure(β=1) curves are primarily affected by the trend of the precision curves. For the LSI, because its precision value and recall value both vary visibly, its F-measure(β=1) curve displays a trend different from its precision and recall curve. Since its precision curve maintains an ascending trend and its recall curve maintains a descending trend, in the former period of the increase of the threshold value (from 0.5 to 0.95), the F-measure(β=1) curve keeps an overall ascending trend, which indicates that its precision curve has a heavier effect on the shape of the F-measure(β=1) curve than the recall curve, in the latter period of the increase of the threshold value (1.0), the F-measure(β=1) curve falls, which indicates that its recall curve has a stronger effect on the shape of the F-measure(β=1) curve than the precision curve. In addition, when we compare the F-measure(β=1) scores between the ECBR, IECBR and LSI, it can be found that the ECBR and IECBR have an overall advantage than the LSI, which illuminates that advantage of the ECBR and IECBR on recall dominates the comparison. Therefore, we can deduce that the ECBR and the IECBR have better aggregated performance than the LSI.

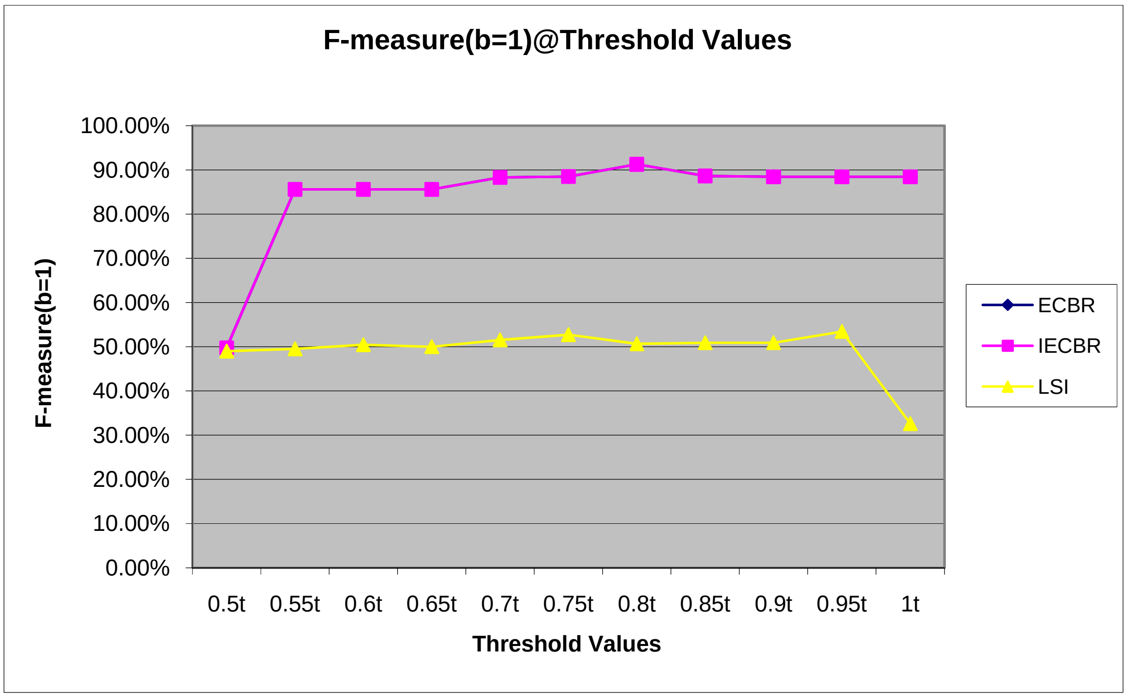


Figure 10.17: Comparison of the ECBR, IECBR and LSI algorithms on F-measure(β=1) in the semantic crawler simulation in the health service domain

Fig. 10.18 shows the comparison of the ECBR, IECBR and LSI algorithms on the F-measure(β=2). Here, the F-measure(β=2) curves shows the similar trend as the F-measure(β=1) curves. The difference is that the F-measure(β=2) scores of the ECBR and IECBR are higher than their F-measure(β=1) scores and, in contrast, the F-measure(β=2) scores of the LSI are lower than its F-measure(β=1) scores. This is because the F-measure(β=2) more emphasizes on recall, the performance of the ECBR and IECBR on recall is better than on precision, and the performance of the LSI on recall is worse than on precision. Consequently, the advantage of the ECBR and IECBR on the F-measure(β=2) is more obvious than on the F-measure(β=1), compared with the LSI. By means of this comparison, it can be concluded that the ECBR and IECBR are more likely to fulfil the preference of most search engines on recall.

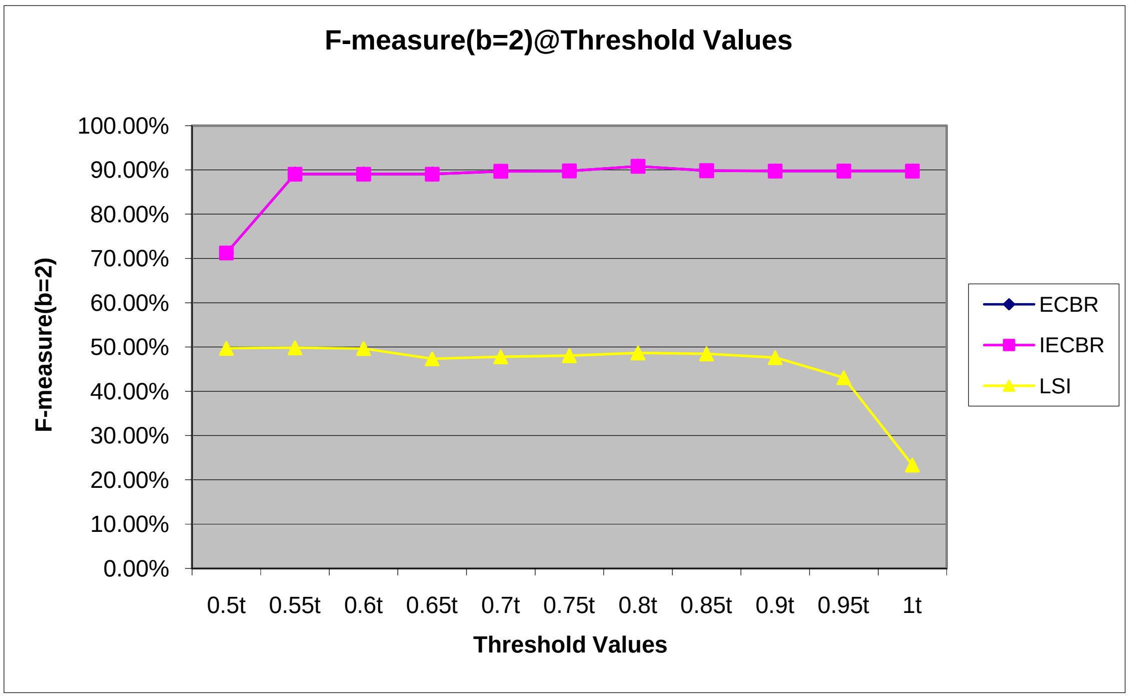


Figure 10.18: Comparison of the ECBR, IECBR and LSI algorithm on F-measure(β=2) in the semantic crawler simulation in the health service domain

Fig. 10.19 presents the comparison of the ECBR, IECBR and LSI algorithms on fallout. Since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have same performance on fallout. Three conclusions are obtained from the comparison. First, the three curves all show an overall descending trend, since the higher threshold value may filter more associated and non-relevant SDE metadata. Second, the variation intervals of the ECBR and IECBR (2.16%) are greater than the LSI (0.64%). Third, the ECBR and IECBR perform better than the LSI on most threshold values (from 0.55 to 0.9), otherwise not. Therefore, the selection of the threshold value directly determines the model which performs better on this indicator.

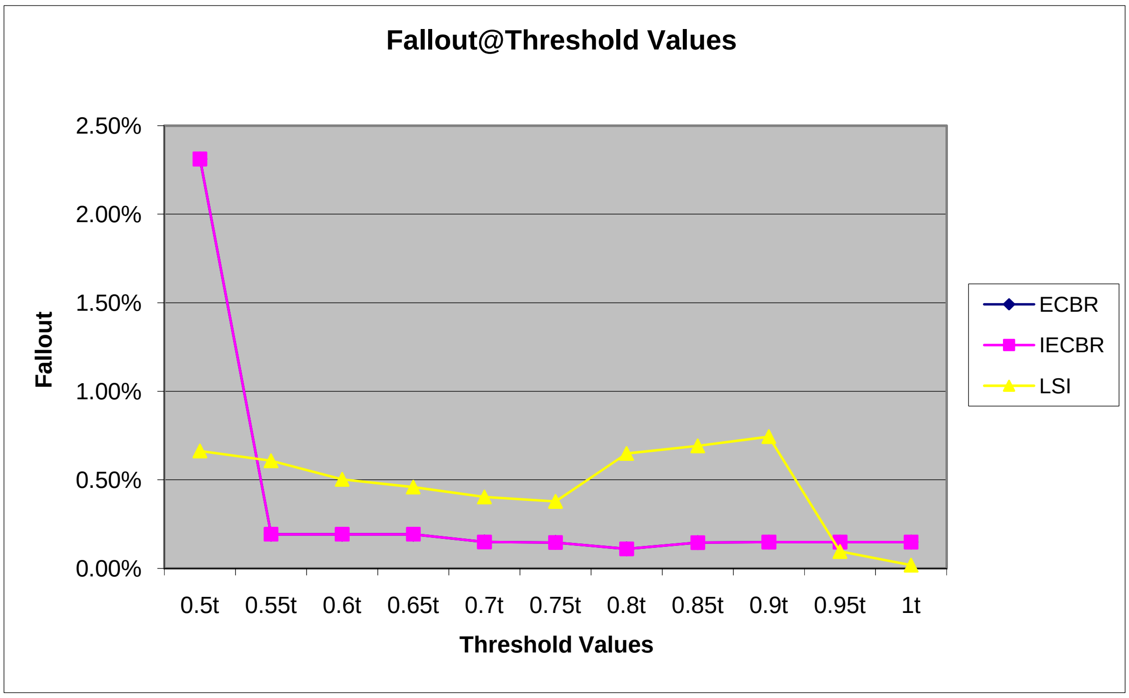


Figure 10.19: Comparison of the ECBR, IECBR and LSI algorithm on fallout in the semantic crawler simulation in the health service domain

Fig 10.20 presents the comparison of the ECBR, IECBR and LSI algorithms on crawling time. It can be clearly observed that the IECBR uses the least time when crawling the same number of webpages, followed by the ECBR and LSI. From this comparison, it can be deduced that the Semantic Crawler that employs the IECBR is faster than the ECBR, in addition to maintaining the same performance on the other indicators, which preliminarily proves the success of the IECBR on efficiency improvement.

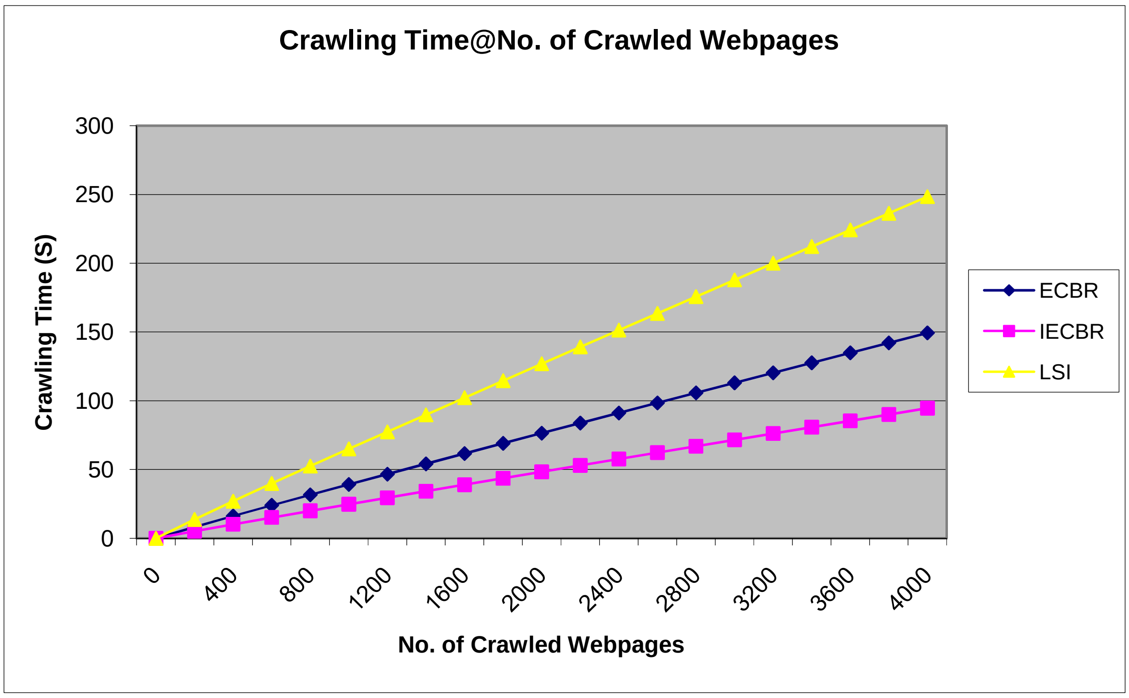


Figure 10.20: Comparison of the ECBR, IECBR and LSI algorithms on crawling time in the semantic crawler simulation in the health service domain

The comparison results of the ECBR, IECBR and LSI algorithms in the semantic crawler simulation in the health service domain can be summarized as follows:

1. The Semantic Crawler has better crawling ability when employing the ECBR and IECBR algorithms.

2. The Semantic Crawler has higher preciseness when employing the ECBR and IECBR algorithms and the relatively moderate threshold values (from 0.55 to 0.9).
3. The Semantic Crawler is able to retrieve more relevant SDE metadata earlier when employing the ECBR and IECBR algorithms and the relatively lower threshold values (from 0.5 to 0.9).

4. The Semantic Crawler has higher effectiveness when employing the ECBR and IECBR algorithms.

5. The Semantic Crawler has better aggregated performance when the ECBR and IECBR algorithms are utilised.

6. The Semantic Crawler is more likely to fulfil the preference of most search engines on recall when employing the ECBR and IECBR algorithms.

7. The Semantic Crawler has better performance on fallout when employing the ECBR and IECBR algorithms and the relatively moderate threshold values (from 0.55 to 0.9).

8. The Semantic Crawler has higher efficiency when the IECBR algorithm is used.

Therefore, from this experiment we can conclude that: 1) the ECBR and IECBR algorithms perform better than does the LSI on the eight indicators when choosing the relatively moderate threshold values (0.55 and 0.9); and 2) the IECBR is more efficient than the ECBR algorithm. This conclusion is consistent with the conclusion that we have drawn in the transport service domain.

For the task of optimal threshold selection, since F-measure(β=1) and F-measure(β=2) are the aggregations of precision and recall, we can assign more weight to these two performance indicators and thus reduce the weight of precision and recall, when determining the optimal threshold values. At the point of 0.8, the F-measure(β=1) and F-measure(β=2) scores of the ECBR and IECBR both reach to the maximum, and their fallout rates both reach to the minimum. Therefore, we choose 0.8 as the optimal threshold value for both the ECBR and IECBR algorithms.

10.3.7 Conclusion

In Section 10.3, we completed two validation tasks.

Firstly, in order to validate the service information and annotation methodology, we ran the Semantic Crawler over the experimental data source and checked the quality of the generate SDE metadata. The result validates the functions of the Semantic Crawler on service information discovery and annotation.

Secondly, we simulated the service information classification methodology, by simulating the ECBR and IECBR algorithm respectively in the transport service domain and the health service domain and comparing their performance with a LSI algorithm on eight performance indicators. The comparison result shows that the ECBR and IECBR algorithms have better performance than the LSI algorithm when choosing the relatively moderate threshold values and the IECBR algorithm has higher efficiency than the ECBR algorithm. In addition, we choose optimal threshold values for the ECBR and IECBR algorithms in order to achieve their best performance. The experiment in the transport service domain shows that 0.75 is the optimal threshold value for both the ECBR and IECBR algorithms, and the experiment in the health service domain shows that 0.8 is the optimal threshold value for them.

<u>10.4 Service Retrieval Simulation and Functional Testing</u>

As described in Chapter 6, the service retrieval methodology includes two modules – a generic service retrieval module and a specific service retrieval module. The former includes an algorithm-based matching and a series of HCIs between users and the search engine. The latter uses SPARQL for the SDE metadata retrieval. Therefore, in this section, we need to implement two validation tasks.

Firstly, we use the simulation approach to validate the algorithm-based matching, by comparing the employed ECBR and IECBR algorithm with three classical information retrieval algorithms – Vector Space Model (VSM), Latent Semantic Indexing (LSI) algorithm and Probabilistic Model (PM), based on sever performance indicators.

Secondly, we use the functional testing approach to validate the rest parts in this methodology.

10.4.1 Tasks of Service Retrieval Simulation

The validation of the algorithm-based matching includes three tasks as follows:

1. Validating the ECBR and IECBR algorithms in the algorithm-based matching. With this objective, we simulate the ECBR and IECBR algorithms respectively in the transport service domain and the health service domain. First of all, we respectively install the transport service ontology and the health service ontology in the Service Knowledge Base. Second, we simulate the ECBR, IECBR, VSM, LSI and PM algorithms in the Service Search Engine, which is the prototype of the service retrieval methodology. Third, we instantiate 100 service queries for each service domain, which cover nearly all the service requests in each domain. Based on the 100 service queries in each service domain, we simulate the use of the ECBR, IECBR, VSM, LSI and PM algorithms for the algorithm-based query-concept matching in the Service Search Engine. By means of the simulation, we compare the performance of these algorithms based on the indicators introduced in the next section.

2. Validating the impact of the WordNet API on the performance of the algorithm-based matching. As described in Chapter 6, we use the WordNet API for query filtering and expansion. In the prototype of the Service Search Engine, we allow users to switch on or switch off the function of the WordNet-based query filtering and expansion in order to measure the impact of the WordNet API on the search performance of the algorithms. Therefore, in the following experiments, we measure the performance of the algorithms with and without the WordNet API based on the indicators, so as to analyse the role of the WordNet API in the scenario of service retrieval.

3. Validating the conceptual framework of the algorithm-based matching. This can be achieved by comparing the outputs between the prototype of the service retrieval methodology – the Service Search Engine and another service search engine based on the same input. Since the Australian Yellowpages® website also has the function of service search (searching for service categories), and our Semantic Crawler uses the same data source as the Yellowpages® website in the transport service domain and the health service domain, we can compare the performance between the Service Search Engine facilitated by the ECBR and IECBR algorithm and the Australian Yellowpages® search engine based on the 100 queries that we instantiate for each service domain in Task 1, in terms of the performance indicator introduced in the next section.

10.4.2 Performance Indicators

In order to quantitatively evaluate the performance of the service retrieval methodology, seven performance indicators are employed from the information retrieval field in this test; and these are: Precision, Mean Average Precision, Recall, F-measure(β=1), F-measure(β=2), Fallout and Response Time. We define the seven indicators in this experiment as follows:

Precision is used to measure the preciseness of a retrieval system [1]. In this experiment, precision is defined as the proportion of retrieved and relevant service concepts in all retrieved service concepts for a query [10], which can be represented as:

$$\text{Precision} = \frac{\text{Number of retrieved and relevant concepts}}{\text{Number of retrieved concepts}} \qquad (10.16)$$

Note that the human judgment approach is used to determine the set of relevant service concepts for a query.

Before we introduce the definition of Mean Average Precision, the concept of Average Precision should be defined. Average Precision is the average of precision values at each retrieved and relevant service concept for a query, given that these concepts are ranked according to their computed similarity values [10]. This indicator is used to measure how quickly and precisely a search engine works [1], and can be represented as:

$$\text{Average Precision(S)} = \frac{\text{Sum (Precision @ Each retrieved and relevant concept)}}{\text{Number of retrieved and relevant concepts}} \qquad (10.17)$$

Mean Average Precision refers to the average of average precision values for a set of queries [10], and can be represented as:

$$\text{Mean Average Precision}=\frac{\sum_{i=1}^{n}\text{Average Precision}(\text{S}_i)}{n} \tag{10.18}$$

Recall refers to the measure of effectiveness of a query system [1]. In this experiment, recall is the proportion of retrieved and relevant service concepts in all relevant service concepts for a query [10], and can be represented as:

$$\text{Recall} = \frac{\text{Number of retrieved and relevant concepts}}{\text{Number of relevant concepts}} \tag{10.19}$$

The F-measure is an aggregated performance scale for the search engine and users can specify the preference on recall or precision by configuring different weights [18]. F-measure(β=1) measures the balance between precision and recall by assigning the same weights to them, which can be represented as:

$$\text{F-measure } (\beta\text{=1}) = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision+Recall}} \tag{10.20}$$

In fact, most search engines are more concerned with recall than precision, as a result of users' purposes in obtaining information [24]. Hence, we employ the F-measure(β=2) that weights recall twice as much as precision, which can be represented as:

$$\text{F-measure } (\beta\text{=2}) = \frac{5\times\text{Precision}\times\text{Recall}}{4\times\text{Precision+Recall}} \tag{10.21}$$

All of the above indicators have the same limitation – they do not consider the number of non-relevant concepts in a retrieved collection. In addition, if there is no relevant concept in the retrieved collection, recall cannot be defined. To solve this issue, we need another performance indicator – Fallout. Fallout is defined as the proportion of retrieved and non-relevant service concepts in the whole collection of non-relevant service concepts in a service ontology [10], and is represented as:

$$\text{Fallout} = \frac{\text{Number of retrieved and non-relevant concepts}}{\text{Number of non-relevant concepts}} \tag{10.22}$$

Unlike the prior performance indicators, the lower the fallout value, the better is the search engine's performance.

Response Time is the interval between the time that an input is received by a system and the time that an output is generated by the system, which is used to test the efficiency of a search system. In this experiment, we define the Response Time as the interval between a user submitting a query to the search engine and the search engine returning ranked concepts for the query.

10.4.3 Vector Space Model for Service Retrieval

In order to horizontally compare the performance of the ECBR and IECBR model in the service retrieval methodology, we make use of the Vector Space Model (VSM). The VSM is a classical model for information retrieval and indexing [20]. Here we employ the VSM in the prototype of our methodology with the purpose of service retrieval.

The implementation details are introduced as follows [3]:

In the Service Ontology Base, first of all, each service concept C is regarded as a body of plain texts comprised of *conceptDescription* property(s). Then, an index term list is obtained from all the concepts in a service ontology. Based on the index term list, each concept C is formed as a vector in which each element corresponds to an index term in the index term list, and the weight of each element is computed by tf-idf. Similarly, a query q can also be seen as a concept and is corresponded with a vector. Thus, the relevance between a concept c_j and a query q can be calculated as the cosine of the angle between these two vectors.

$$sim(c_j, q) = \frac{|\vec{c}_j \cap \vec{q}|}{|\vec{c}_j| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \qquad (10.23)$$

where $\vec{c}_j$ and $\vec{q}$ are two vectors corresponding to c_j and q respectively, $|\vec{c}_j|$ and $|\vec{q}|$ are the norms of $\vec{c}_j$ and $\vec{q}$, t is the number of index terms in an index terms list of a service ontology, $w_{i,j}$ and $w_{i,q}$ are weights of each element of $\vec{c}_j$ and $\vec{q}$ corresponding to each index term. Since $w_{i,j} >= 0$, $w_{i,q} >= 0$, $sim(c_j, q) \rightarrow [0, 1]$.

A threshold value needs to be configured in order to determine the similarity value between a service concept and a query, which enables a concept to be partially similar to a query.

10.4.4 Latent Semantic Indexing Algorithm for Service Retrieval

In order to horizontally compare the performance of the ECBR and IECBR models in the service retrieval methodology, we make use of the LSI algorithm in the prototype of the methodology, for the purpose of service retrieval [10].

The implementation details are introduced as follows [3]:

In the Service Ontology Base, first of all, each service concept C is regarded as a body of plain texts comprised of *conceptDescription* property(s). Following that, an index term list is obtained from all the concepts in a service ontology. Based on the index term list, each concept C is formed as an array in which each element is obtained by tf-idf, and all the concepts in the service ontology are formed as a term-concept matrix A. The term-concept matrix is then decomposed by the Singular Value Decomposition (SVD) approach, which can be mathematically represented by Equation 10.24.

$$A = U\Sigma V^T \tag{10.24}$$

where U is the matrix derived from the term-to-term matrix given by AA^T, V^T is the matrix derived from the transpose of the concept-to-concept matrix given by A^TA, and Σ is a r×r diagonal matrix of singular values where r = min(t, N) is the rank of A.

Considering that now only k largest singular values of Σ are kept along with their corresponding columns in U and V^T, the resultant A_k matrix is the matrix of rank k which is closest to the original matrix A in the least square sense. This matrix is given by Equation 10.25.

$$A_k = U_k \Sigma_k V_k^T \tag{10.25}$$

where k (k < r) is the dimensionality of a reduced concept space.

Analogous to the concept, a query q can be formed as an index term-based array in which each element is the tf-idf weight between the query and a term from the index term list. The array can then be translated into the concept space by Equation 10.26, and then compared with A_k by the cosine algorithm to calculate the similarity values with each concept, which can be represented by Equation 10.27.

$$q' = \Sigma_k^{-1} U_k^T q \tag{10.26}$$

$$sim(c,\ q') = \frac{|A_k \cap q'|}{|A_k| \times |q'|} \tag{10.27}$$

A threshold value needs to be configured in order to determine the similarity value between a service concept and a query, which enables a concept to be partially similar to a query.

10.4.5 Probabilistic Model for Service Retrieval

In order to horizontally compare the performance of the ECBR and IECBR models in the service retrieval methodology, we make use of the probabilistic model (also known as the Binary Independence Retrieval (BIR) model [19]) in the prototype of the methodology, for the purpose of service retrieval.

The fundamental of the probabilistic model, is as follows [3]:

In the Service Ontology Base, first of all, each service concept C is regarded as a body of plain texts comprised of *conceptDescription* property(s). Following that, an index term list is obtained from all the concepts in a service ontology. Based on the index term list, each concept C is formed as a vector in which each element corresponds to an index term in the index term list, and the weight of each element is 1 or 0, in order to indicate whether or not an index term appears in the concept. Similarly, a query can be formed as a similarity vector.

Given a user query q and a service concept c_j in a service ontology, the probabilistic model tries to estimate the probability that the user will find the concept of interest. The model assumes that this probability of relevance only depends on the query and the concept representations only. Furthermore, the model assumes that there is a subset of all concepts which the user prefers as the answer set for the query. Such an ideal answer set should maximize the overall probability of relevance to the user. Concepts in the ideal set are predicted to be relevant to the query, and concepts outside this set are predicted to be non-relevant. The similarity value between a concept and a query is the ratio between the possibility that the concept is relevant to the query and the possibility that the concept is non-relevant to the query. The probabilistic model is represented as follows:

$$sim(c_j, q) = \frac{P(R \mid c_j)}{P(\bar{R} \mid c_j)} \tag{10.28}$$

where R is the set of concepts known to be relevant to q, $\bar{R}$ is the complement of R, P(R|c_j) is the possibility that c_j is relevant to q, and P($\bar{R}$|c_j) is the possibility that that c_j is non-relevant to q.

According to Bayes' theorem, Equation 10.28 can be transformed to Equation 10.29.

$$sim(c_j, q) = \frac{P(c_j \mid R) \times P(R)}{P(c_j \mid \bar{R}) \times P(\bar{R})} \tag{10.29}$$

where P(c_j|R) is the possibility of selecting c_j from R, and P(c_j|$\bar{R}$) is the possibility of selecting c_j from $\bar{R}$, P(R) is the possibility of selecting a concept from R and P($\bar{R}$) is the possibility of selecting a concept from $\bar{R}$. Since P(R) and P($\bar{R}$) are both approximate to the number of service concepts in a service ontology, the similarity between q and c_j is:

$$sim(c_j, q) \sim \frac{P(c_j \mid R)}{P(c_j \mid \bar{R})} \tag{10.30}$$

Given the prerequisite of index terms independence, Equation 10.30 can be represented by

$$sim(c_j,q) \sim \frac{(\prod_{g_i(\vec{c}_j)=1} P(k_i \mid R)) \times (\prod_{g_i(\vec{c}_j)=0} P(\bar{k}_i \mid R))}{(\prod_{g_i(\vec{c}_j)=1} P(k_i \mid \bar{R})) \times (\prod_{g_i(\vec{c}_j)=0} P(\bar{k}_i \mid \bar{R}))} \tag{10.31}$$

where P(k_i|R) and P(k_i|$\bar{R}$) are the possibility of selecting an index term k_i from R and $\bar{R}$ respectively, P($\bar{k}_i$|R) and P($\bar{k}_i$|$\bar{R}$) are the complement of p(k_i|R) and P(k_i|$\bar{R}$), $\vec{c}_j$ is the vector corresponding to c_j, and g_i is a function that returns the weight of $\vec{c}_j$ on the index term k_i.

Since P(k_i|R) + P($\bar{k}_i$|R) = 1, and P(k_i|$\bar{R}$) + P($\bar{k}_i$|$\bar{R}$) = 1, Equation 10.31 can be represented by

$$\text{sim}(c_j,\ q) \ \sim\ \sum_{i=1}^{t} w_{i,q} \times w_{i,j} \times \left(\log \frac{P(k_i \mid R)}{1-P(k_i \mid R)} + \log \frac{1-P(k_i \mid \bar{R})}{P(k_i \mid \bar{R})} \right) \tag{10.32}$$

where t is the number of index terms in a service ontology, $w_{i,q}$ is the weight of vector $\vec{q}$ on the index term k_i, and $w_{i,j}$ is the weight of vector $\vec{c}_j$ on the index term k_i

Two assumptions are made for the initial value of P(k_i|R) and P(k_i|$\bar{R}$), which are that P(k_i|R) is constant for all k_i (equal to 0.5), and that P(k_i|$\bar{R}$) is approximate to the distribution of k_i in a service ontology.

Thus, P(k_i|R) = 0.5 and P(k_i|$\bar{R}$) ~ n_i/N, where N is the number of service concepts in a service ontology and $\mathbf{n_i}$ is the number of concepts containing k_i. Hence, Equation 10.32 can be converted into Equation 10.33.

$$\text{sim}(c_j,\ q) \ \sim\ \sum_{i=1}^{t} w_{i,q} \times w_{i,j} \times \log\left(\frac{N}{n_i} - 1 \right) \tag{10.33}$$

Then, the performance of retrieval can be improved by further two assumptions, which are that P(k_i|R) is approximate to the distribution of k_i in the retrieved concepts, and P(k_i|$\bar{R}$) is approximate to the distribution of k_i in the non-relevant and non-retrieved concepts.
Let V be a subset of concepts retrieved and ranked by the probabilistic model, V_i be the subset of V composed of the concepts in V, which contain k_i, so

$$P(k_i \mid R) = \frac{V_i}{V} \tag{10.34}$$

$$P(k_i \mid \bar{R}) = \frac{n_i - V_i}{N - V} \tag{10.35}$$

Hence, Equation 10.32 can be represented by

$$\mathrm{sim}(c_j,\ q)\ \sim\ \sum_{i=1}^{t} w_{i,q} \times w_{i,j} \times \left(\log \frac{V_i}{V - V_i} + \log \left(\frac{N - V}{n_i - V_i} - 1 \right) \right) \qquad (10.36)$$

A threshold value needs to be configured in order to determine the similarity value between a service concept and a query, which enables a concept to be partially similar to a query.

10.4.6 Service Retrieval Simulation in Transport Service Domain

In this section, we compare the performance of the ECBR, IECBR, VSM, LSI and PM algorithms with/without the WordNet API for query filtering expansion in the transport service domain, based on the seven performance indicators, which are: precision, mean average precision, recall, F-measure(β=1), F-measure(β=2), fallout and response time. In addition, as stated in Chapter 6, the performance of these algorithms partly depends on their threshold values, which are used to filter non-relevant service concepts from the retrieved results based on the similarity values between the concepts and a user query. Hence, in this experiment, we have another task – choosing the optimal threshold values for these algorithms, and comparing their best performance when facilitating the algorithms with their optimal threshold values. Therefore, we set the initial threshold value at 0.5, and increase it by 0.05 each time until it reaches 0.8. Then we measure the performance scores of the ECBR, IECBR and LSI algorithms on the performance indicators at the time of each increment of the threshold value. The performance scores of the ECBR, IECBR, VSM, LSI and PM algorithms on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown from Table 10.1 to Table 10.6.

Table 10.1: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on precision in the service retrieval simulation in the transport service domain

Threshold value	ECBR (OFF*)	ECBR (ON*)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	17.73%	12.38%	17.73%	12.38%	23.13%	21.43%	4.13%	4.09%	24.01%	22.15%
>0.05	17.73%	12.38%	17.73%	12.38%	23.35%	21.81%	14.29%	13.50%	24.17%	29.11%
>0.1	17.73%	12.38%	17.73%	12.38%	23.90%	23.26%	21.08%	20.23%	25.08%	40.96%
>0.15	22.59%	14.86%	22.59%	14.86%	25.32%	25.57%	25.04%	24.52%	27.14%	44.37%
>0.2	23.63%	17.21%	23.63%	17.21%	28.33%	28.93%	27.96%	27.17%	27.17%	49.76%
>0.25	26.92%	24.75%	26.92%	24.75%	32.36%	34.67%	31.35%	30.85%	32.08%	55.35%
>0.3	26.92%	24.75%	26.92%	24.75%	37.36%	40.19%	32.82%	32.12%	35.08%	61.97%
>0.35	29.79%	28.15%	29.79%	28.15%	41.57%	44.38%	34.72%	34.95%	38.00%	62.35%
>0.4	29.65%	28.01%	29.65%	28.01%	48.35%	51.45%	36.15%	37.01%	43.31%	63.39%
>0.45	29.57%	27.95%	29.57%	27.95%	53.08%	60.03%	37.79%	39.59%	46.33%	73.33%
>0.5	29.57%	27.95%	29.57%	27.95%	63.93%	68.55%	40.09%	42.38%	47.91%	73.28%

>0.55	71.53%	66.46%	71.53%	66.46%	67.09%	72.52%	43.74%	46.36%	53.05%	73.21%
>0.6	71.53%	66.46%	71.53%	66.46%	76.65%	79.75%	46.76%	51.60%	56.66%	75.62%
>0.65	71.66%	66.51%	71.66%	66.51%	79.02%	79.25%	49.48%	52.85%	56.07%	67.33%
>0.7	80.03%	74.41%	80.03%	74.41%	76.32%	76.96%	55.19%	60.80%	59.17%	57.94%
>0.75	80.98%	79.43%	80.98%	79.43%	84.00%	85.63%	59.95%	66.83%	61.28%	60.83%
>0.8	80.98%	79.43%	80.98%	79.43%	85.71%	87.93%	73.06%	76.90%	59.64%	53.92%

*ON – WordNet API switched on
*OFF – WordNet API switched off

Table 10.2: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on mean average precision in the service retrieval simulation in the transport service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	72.48%	70.97%	72.48%	70.97%	67.65%	69.38%	58.56%	59.34%	64.23%	64.63%
>0.05	72.48%	70.97%	72.48%	70.97%	67.65%	70.28%	62.86%	64.26%	64.51%	68.15%
>0.1	72.48%	70.97%	72.48%	70.97%	68.44%	70.65%	65.06%	65.93%	65.27%	71.80%
>0.15	72.48%	70.97%	72.48%	70.97%	69.68%	72.66%	66.91%	67.48%	66.87%	69.92%
>0.2	72.66%	71.22%	72.66%	71.22%	71.41%	73.93%	68.80%	69.89%	67.99%	71.10%
>0.25	72.95%	71.82%	72.95%	71.82%	73.43%	75.17%	71.35%	72.88%	70.32%	73.24%
>0.3	72.95%	71.82%	72.95%	71.82%	74.67%	76.78%	72.34%	74.04%	70.76%	77.44%
>0.35	80.33%	74.91%	80.33%	74.91%	76.40%	79.69%	73.93%	75.77%	71.62%	73.56%
>0.4	81.53%	78.18%	81.53%	78.18%	78.78%	81.83%	75.23%	77.57%	72.78%	75.05%
>0.45	81.69%	78.39%	81.69%	78.39%	81.75%	84.01%	77.97%	79.99%	73.21%	73.71%
>0.5	81.69%	78.39%	81.69%	78.39%	86.17%	86.99%	79.33%	82.09%	74.48%	70.74%
>0.55	90.33%	90.65%	90.33%	90.65%	85.88%	88.13%	80.23%	82.23%	73.37%	71.04%
>0.6	90.33%	90.65%	90.33%	90.65%	87.07%	87.95%	80.44%	83.25%	74.89%	73.74%
>0.65	90.33%	90.79%	90.33%	90.79%	84.44%	83.81%	80.65%	83.45%	74.60%	68.64%
>0.7	90.65%	91.15%	90.65%	91.15%	81.90%	79.25%	81.82%	85.11%	75.09%	61.10%
>0.75	90.65%	90.65%	90.65%	90.65%	89.34%	86.05%	82.08%	85.17%	76.30%	64.15%
>0.8	90.65%	90.65%	90.65%	90.65%	87.14%	87.93%	83.97%	87.24%	74.99%	57.82%

Table 10.3: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on recall in the service retrieval simulation in the transport service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	70.85%	75.26%	70.85%	75.26%	66.31%	67.31%	80.68%	81.76%	64.86%	65.86%
>0.05	70.85%	75.26%	70.85%	75.26%	66.30%	65.89%	73.29%	73.12%	63.92%	57.83%
>0.1	70.85%	75.26%	70.85%	75.26%	65.21%	64.74%	69.49%	70.09%	63.13%	48.03%
>0.15	70.85%	75.25%	70.85%	75.25%	61.92%	61.34%	64.78%	67.08%	61.06%	36.76%
>0.2	70.51%	74.59%	70.51%	74.59%	58.54%	58.29%	62.11%	63.30%	63.30%	28.80%
>0.25	70.15%	73.80%	70.15%	73.80%	54.35%	55.59%	59.13%	59.83%	55.48%	23.17%
>0.3	70.15%	73.80%	70.15%	73.80%	51.02%	49.99%	56.71%	56.61%	52.33%	19.32%
>0.35	62.19%	67.71%	62.19%	67.71%	46.17%	45.50%	53.95%	53.79%	46.82%	14.59%
>0.4	60.27%	65.80%	60.27%	65.80%	42.59%	41.98%	51.50%	50.50%	45.30%	12.41%
>0.45	60.02%	65.55%	60.02%	65.55%	37.56%	37.25%	47.91%	47.55%	44.44%	10.32%
>0.5	60.02%	65.55%	60.02%	65.55%	32.74%	33.04%	45.38%	44.00%	39.01%	9.40%

>0.55	40.76%	41.43%	40.76%	41.43%	29.51%	29.30%	42.31%	42.37%	36.12%	6.90%
>0.6	40.76%	41.43%	40.76%	41.43%	27.00%	26.48%	39.58%	38.22%	35.39%	6.74%
>0.65	40.76%	41.09%	40.76%	41.09%	22.90%	19.84%	35.80%	34.03%	33.71%	6.38%
>0.7	37.12%	37.46%	37.12%	37.46%	17.62%	13.40%	31.87%	31.55%	28.14%	4.38%
>0.75	36.62%	36.62%	36.62%	36.62%	15.31%	11.52%	28.36%	27.27%	25.17%	4.38%
>0.8	36.62%	36.62%	36.62%	36.62%	9.83%	8.19%	24.97%	23.98%	23.95%	3.46%

Table 10.4: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on F-measure(β=1) in the service retrieval simulation in the transport service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	28.36%	21.27%	28.36%	21.27%	34.30%	32.51%	7.86%	7.79%	35.05%	33.15%
>0.05	28.36%	21.27%	28.36%	21.27%	34.54%	32.78%	23.91%	22.79%	35.07%	38.73%
>0.1	28.36%	21.27%	28.36%	21.27%	34.98%	34.22%	32.35%	31.39%	35.89%	44.21%
>0.15	34.25%	24.83%	34.25%	24.83%	35.94%	36.10%	36.12%	35.91%	37.57%	40.21%
>0.2	35.40%	27.97%	35.40%	27.97%	38.18%	38.67%	38.57%	38.02%	38.02%	36.49%
>0.25	38.90%	37.07%	38.90%	37.07%	40.57%	42.71%	40.97%	40.71%	40.65%	32.67%
>0.3	38.90%	37.07%	38.90%	37.07%	43.13%	44.56%	41.57%	40.99%	42.00%	29.45%
>0.35	40.28%	39.77%	40.28%	39.77%	43.75%	44.93%	42.25%	42.37%	41.95%	23.65%
>0.4	39.75%	39.29%	39.75%	39.29%	45.29%	46.23%	42.48%	42.72%	44.28%	20.75%
>0.45	39.62%	39.19%	39.62%	39.19%	43.99%	45.97%	42.25%	43.21%	45.37%	18.09%
>0.5	39.62%	39.19%	39.62%	39.19%	43.30%	44.59%	42.57%	43.18%	43.00%	16.67%
>0.55	51.93%	51.04%	51.93%	51.04%	40.99%	41.74%	43.01%	44.27%	42.98%	12.62%
>0.6	51.93%	51.04%	51.93%	51.04%	39.93%	39.75%	42.87%	43.92%	43.57%	12.37%
>0.65	51.96%	50.80%	51.96%	50.80%	35.50%	31.73%	41.54%	41.40%	42.11%	11.65%
>0.7	50.72%	49.83%	50.72%	49.83%	28.64%	22.83%	40.40%	41.54%	38.14%	8.14%
>0.75	50.43%	50.13%	50.43%	50.13%	25.90%	20.31%	38.50%	38.74%	35.69%	8.16%
>0.8	50.43%	50.13%	50.43%	50.13%	17.64%	14.98%	37.22%	36.56%	34.17%	6.50%

Table 10.5: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on F-measure(β=2) in the service retrieval simulation in the transport service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	44.30%	37.34%	44.30%	37.34%	48.29%	47.13%	17.15%	17.03%	48.39%	47.22%
>0.05	44.30%	37.34%	44.30%	37.34%	48.47%	46.93%	40.14%	38.82%	48.10%	48.30%
>0.1	44.30%	37.34%	44.30%	37.34%	48.46%	47.72%	47.62%	46.95%	48.43%	46.43%
>0.15	49.64%	41.52%	49.64%	41.52%	48.03%	47.93%	49.17%	49.79%	48.85%	38.07%
>0.2	50.48%	44.75%	50.48%	44.75%	48.25%	48.46%	49.92%	50.00%	50.00%	31.45%
>0.25	53.09%	52.85%	53.09%	52.85%	47.85%	49.60%	50.22%	50.37%	48.42%	26.22%
>0.3	53.09%	52.85%	53.09%	52.85%	47.54%	47.67%	49.50%	49.12%	47.65%	22.40%
>0.35	51.08%	52.85%	51.08%	52.85%	45.17%	45.27%	48.57%	48.56%	44.75%	17.24%
>0.4	49.96%	51.81%	49.96%	51.81%	43.63%	43.58%	47.47%	47.07%	44.89%	14.79%
>0.45	49.77%	51.65%	49.77%	51.65%	39.89%	40.31%	45.47%	45.71%	44.81%	12.46%
>0.5	49.77%	51.65%	49.77%	51.65%	36.28%	36.86%	44.22%	43.67%	40.51%	11.39%
>0.55	44.60%	44.80%	44.60%	44.80%	33.24%	33.27%	42.59%	43.11%	38.59%	8.43%
>0.6	44.60%	44.80%	44.60%	44.80%	31.02%	30.56%	40.84%	40.31%	38.27%	8.24%

>0.65	44.61%	44.49%	44.61%	44.49%	26.69%	23.34%	37.89%	36.64%	36.64%	7.78%
>0.7	41.58%	41.59%	41.58%	41.59%	20.83%	16.06%	34.81%	34.91%	31.44%	5.37%
>0.75	41.13%	41.05%	41.13%	41.05%	18.31%	13.93%	31.70%	30.94%	28.54%	5.37%
>0.8	41.13%	41.05%	41.13%	41.05%	11.95%	10.00%	28.76%	27.80%	27.20%	4.25%

Table 10.6: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on fallout in the service retrieval simulation in the transport service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)
>0	7.37%	12.74%	7.37%	12.74%	5.41%	6.08%	45.70%	46.11%	5.01%
>0.05	7.37%	12.74%	7.37%	12.74%	5.11%	5.57%	9.60%	10.01%	4.71%
>0.1	7.37%	12.74%	7.37%	12.74%	4.66%	4.82%	5.55%	6.02%	4.11%
>0.15	6.69%	11.85%	6.69%	11.85%	3.93%	3.99%	4.19%	4.51%	3.47%
>0.2	6.04%	9.28%	6.04%	9.28%	3.17%	3.12%	3.40%	3.64%	3.64%
>0.25	5.03%	5.77%	5.03%	5.77%	2.52%	2.40%	2.84%	2.97%	2.38%
>0.3	5.03%	5.77%	5.03%	5.77%	1.90%	1.66%	2.43%	2.46%	1.92%
>0.35	3.36%	3.95%	3.36%	3.95%	1.39%	1.19%	2.06%	2.00%	1.36%
>0.4	3.31%	3.88%	3.31%	3.88%	1.00%	0.82%	1.73%	1.64%	1.15%
>0.45	3.31%	3.88%	3.31%	3.88%	0.71%	0.52%	1.42%	1.32%	0.94%
>0.5	3.31%	3.88%	3.31%	3.88%	0.44%	0.34%	1.16%	1.06%	0.74%
>0.55	0.41%	0.48%	0.41%	0.48%	0.30%	0.22%	0.91%	0.83%	0.52%
>0.6	0.41%	0.48%	0.41%	0.48%	0.18%	0.14%	0.71%	0.63%	0.47%
>0.65	0.40%	0.48%	0.40%	0.48%	0.13%	0.09%	0.56%	0.50%	0.43%
>0.7	0.15%	0.23%	0.15%	0.23%	0.10%	0.07%	0.42%	0.37%	0.33%
>0.75	0.14%	0.16%	0.14%	0.16%	0.06%	0.03%	0.31%	0.24%	0.28%
>0.8	0.14%	0.16%	0.14%	0.16%	0.03%	0.01%	0.17%	0.13%	0.28%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we place more weight on them when we determine the optimal threshold values for the algorithms. Therefore, we choose two groups of optimal threshold values respectively based on F-measure(β=1) and F-measure(β=2).

Table 10.7 shows the comparison results of the ECBR, IECBR, VSM, LSI and PM algorithm on their optimal threshold values for F-measure(β=1) as well as the Yellowpages® search engine. First of all, since the ECBR and the IECBR follow the similar principle with the only difference being their efficiency, these two algorithms have the same optimal threshold value and the best performance. It can be seen that the ECBR and IECBR algorithms have the highest scores on the performance indicators of precision, mean average precision, F-measure(β=1) and fallout. Second, whereas the WordNet API enhances the performance of the ECBR and the IECBR on mean average precision (0.32%) and recall (0.67%), it reduces the performance on precision (-5.2%), F-measure(β=1) (-0.92%) and fallout (-0.08%) at the same time. Third, since we cannot obtain the exact number of non-relevant items in the database of the Yellowpages® search engine for a query, the fallout rate for the Yellowpages® cannot be computed. Similarly, there is no threshold value that can be found from the Yellowpages® search engine. It can be observed that the performance of all the algorithms employed by the

Service Search Engine surpasses the Yellowpages® search engine for all the performance indicators (excluding fallout).

Table 10.7: Comparison of the ECBR, IECBR, VSM, LSI and PM algorithms on their optimal threshold values as well as the Yellowpages® search engine for F-measure(β=1) in the service retrieval simulation in the transport service domain

Algorithm	Optimal threshold value	Precision	Mean Average Precision	Recall	F-measure(β=1)	Fallout
ECBR(OFF)	>0.65	71.66%	90.33%	40.76%	51.96%	0.40%
ECBR(ON)	>0.55	66.46%	90.65%	41.43%	51.04%	0.48%
IECBR(OFF)	>0.65	71.66%	90.33%	40.76%	51.96%	0.40%
IECBR(ON)	>0.55	66.46%	90.65%	41.43%	51.04%	0.48%
VSM(OFF)	>0.4	48.35%	78.78%	42.59%	45.29%	1.00%
VSM(ON)	>0.4	51.45%	81.83%	41.98%	46.23%	0.82%
LSI(OFF)	>0.55	43.74%	80.23%	42.31%	43.01%	0.91%
LSI(ON)	>0.55	46.36%	82.23%	42.37%	44.27%	0.83%
PM(OFF)	>0.45	46.33%	73.21%	44.44%	45.37%	0.94%
PM(ON)	>0.1	40.96%	71.80%	48.03%	44.21%	1.72%
YP*	N/A	14.63%	17.85%	25.00%	18.46%	N/A

*YP stands for the Australian Yellowpages® Website search engine

Table 10.8 shows the comparison results of the ECBR, IECBR, VSM, LSI and PM algorithms on their optimal threshold values for F-measure(β=2) as well as the Yellowpages® search engine. First, due to the preference of F-measure(β=2) on recall, it can be seen that the ECBR and IECBR algorithms have the highest scores on recall and F-measure(β=2). Second, the WordNet API enhances the performance of the ECBR and IECBR on recall but reduces the performance on the other indicators. Third, it can be observed that all the algorithms employed in the Service Search Engine double the scores of the Yellowpages® search engine on recall and F-measure(β=2).

Table 10.8: Comparison of the ECBR, IECBR, VSM, LSI and PM algorithms on their optimal threshold values for F-measure(β=2) as well as the Yellowpages® search engine in the service retrieval simulation in the transport service domain

Algorithms	Proper Threshold Value	Precision	Mean Average Precision	Recall	F-measure(β=2)	Fallout
ECBR(OFF)	>0.25	26.92%	72.95%	70.15%	53.09%	5.03%
ECBR(ON)	>0.25	24.75%	71.82%	73.80%	52.85%	5.77%
IECBR(OFF)	>0.25	26.92%	72.95%	70.15%	53.09%	5.03%
IECBR(ON)	>0.25	24.75%	71.82%	73.80%	52.85%	5.77%
VSM(OFF)	>0.05	23.35%	67.65%	66.30%	48.47%	5.11%
VSM(ON)	>0.25	34.67%	75.17%	55.59%	49.60%	2.40%
LSI(OFF)	>0.25	31.35%	71.35%	59.13%	50.22%	2.84%
LSI(ON)	>0.25	30.85%	72.88%	59.83%	50.37%	2.97%
PM(OFF)	>0.2	27.17%	67.99%	63.30%	50.00%	3.64%

PM(ON)	>0.05	29.11%	68.15%	57.83%	48.30%	3.32%
YP	N/A	14.63%	17.85%	25.00%	21.90%	N/A

Fig. 10.21 presents the comparison of the ECBR, IECBR, VSM, LSI and PM algorithms on response time. As the Yellowpages® search engine cannot record the response time, we exclude it from this comparison. First, it can be seen that the PM is the fastest algorithm, followed by the VSM and the IECBR which have a difference of 0.003s, and the last is the ECBR. Second, the WordNet API increases the time cost of each algorithm, which is nearly 5 times for the ECBR and nearly 7 times for IECBR, etc. This is because the WordNet API increases the number of the query words by finding their synonyms and thus affects the efficiency of the algorithms in the Service Search Engine.

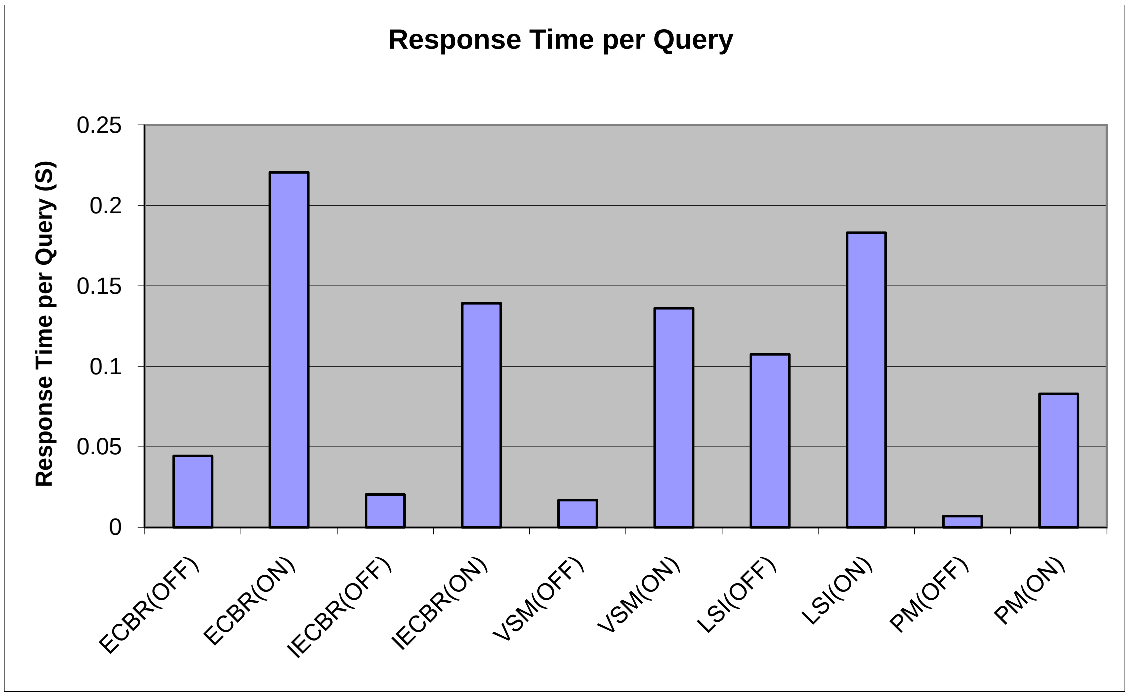


Figure 10.21: Comparison of the ECBR, IECBR, VSM, LSI and PM algorithms on response time in the service retrieval simulation in the transport service domain

The comparison results in the service retrieval simulation in the transport service domain can be summarised according to three tasks as follows:

1. In this experiment, the ECBR and IECBR algorithms show superior performance on precision, mean average precision, F-measure(β=1), F-measure(β=2) and fallout when comparing with the traditional information retrieval algorithms. The efficiency of the IECBR is higher than the ECBR in terms of response time.

2. In this experiment, the WordNet API does not remarkably enhance the performance of the Service Search Engine and, in contrast, the WordNet API obviously reduces the efficiency of the Service Search Engine.

3. In this experiment, the Service Search Engine shows an overall advantage over the ordinary search engines.

10.4.7 Service Retrieval Simulation in Health Service Domain

In this section, we compare the performance of the ECBR, IECBR, VSM, LSI and PM algorithms with/without the WordNet API for query filtering expansion in the health service domain, based on the seven performance indicators, which are precision, mean average precision, recall, F-measure(β=1), F-measure(β=2), fallout and response time. In addition, as stated in Chapter 6, the performance of these algorithms partly depends on their threshold values, which are used to filter non-relevant service concepts from the retrieved results based on the similarity values between the concepts and a user query. Hence, in this experiment, we have another task – to choose the optimal threshold values for these algorithms, and compare their best performance when facilitating the algorithms with their optimal threshold values. Therefore, we set the initial threshold value at 0.5, and increase it by 0.05 each time until 0.95 is reached. Then we measure the performance scores of the ECBR, IECBR and LSI algorithm on the performance indicators at each time of the increment of the threshold value. The performance scores of the ECBR, IECBR, VSM, LSI and PM algorithms on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2), and fallout are shown from Table 10.9 to Table 10.14.

Table 10.9: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on precision in the service retrieval simulation in the health service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	14.39%	14.14%	14.39%	14.14%	14.42%	14.18%	1.36%	1.32%	53.34%	51.86%
>0.05	14.39%	14.14%	14.39%	14.14%	14.44%	14.21%	1.36%	45.41%	53.34%	57.11%
>0.1	14.39%	14.14%	14.39%	14.14%	15.23%	15.28%	1.36%	52.66%	53.76%	69.29%
>0.15	14.39%	14.14%	14.39%	14.14%	17.63%	18.51%	64.02%	61.05%	55.43%	76.57%
>0.2	14.39%	14.14%	14.39%	14.14%	23.04%	26.15%	71.90%	67.59%	58.23%	79.89%
>0.25	14.58%	14.34%	14.58%	14.34%	29.28%	33.84%	77.09%	72.06%	61.44%	80.06%
>0.3	14.58%	14.34%	14.58%	14.34%	41.36%	47.48%	81.72%	75.60%	69.52%	82.23%
>0.35	17.27%	16.96%	17.27%	16.96%	61.47%	63.85%	85.91%	82.22%	70.63%	82.98%
>0.4	17.27%	16.96%	17.27%	16.96%	87.69%	85.10%	89.58%	85.16%	73.41%	85.02%
>0.45	17.27%	16.96%	17.27%	16.96%	89.46%	86.32%	95.11%	90.43%	75.83%	87.02%
>0.5	92.08%	90.50%	92.08%	90.50%	91.20%	87.59%	95.65%	92.47%	76.54%	87.52%
>0.55	92.08%	90.50%	92.08%	90.50%	94.94%	92.73%	96.59%	96.51%	77.86%	88.61%
>0.6	92.08%	90.50%	92.08%	90.50%	96.55%	95.83%	97.65%	98.75%	78.32%	89.93%
>0.65	92.08%	90.50%	92.08%	90.50%	100.00%	99.09%	98.78%	98.73%	78.99%	89.79%
>0.7	95.67%	94.33%	95.67%	94.33%	100.00%	98.98%	98.68%	98.65%	82.06%	92.40%

>0.75	95.67%	94.50%	95.67%	94.50%	100.00%	98.72%	98.57%	98.51%	85.87%	95.13%
>0.8	95.67%	94.50%	95.67%	94.50%	100.00%	100.00%	98.51%	98.41%	85.87%	94.97%
>0.85	95.67%	94.50%	95.67%	94.50%	100.00%	100.00%	98.41%	98.28%	85.72%	96.45%
>0.9	95.67%	94.50%	95.67%	94.50%	N/A	N/A	98.33%	98.15%	87.28%	98.28%
>0.95	95.67%	94.50%	95.67%	94.50%	N/A	N/A	98.11%	97.78%	87.28%	98.28%

Table 10.10: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on mean average precision in the service retrieval simulation in the health service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	96.28%	81.17%	96.28%	81.17%	87.80%	70.46%	83.76%	68.92%	93.49%	75.37%
>0.05	96.28%	81.17%	96.28%	81.17%	87.80%	70.46%	83.76%	86.36%	93.49%	77.38%
>0.1	96.28%	81.17%	96.28%	81.17%	88.00%	70.46%	83.76%	88.47%	93.49%	82.21%
>0.15	96.28%	81.17%	96.28%	81.17%	88.40%	71.47%	95.99%	93.80%	93.86%	86.05%
>0.2	96.28%	81.17%	96.28%	81.17%	89.10%	72.67%	95.99%	93.80%	94.01%	90.50%
>0.25	96.28%	81.17%	96.28%	81.17%	92.49%	81.98%	97.40%	93.80%	94.95%	90.50%
>0.3	96.28%	81.17%	96.28%	81.17%	96.02%	84.63%	97.40%	93.80%	95.29%	94.46%
>0.35	96.98%	81.17%	96.98%	81.17%	97.48%	89.23%	98.90%	95.59%	96.93%	98.57%
>0.4	96.98%	81.17%	96.98%	81.17%	98.47%	92.32%	99.44%	98.44%	97.18%	98.57%
>0.45	96.98%	81.17%	96.98%	81.17%	98.84%	94.74%	100.00%	98.44%	97.35%	98.57%
>0.5	99.33%	81.17%	99.33%	81.17%	99.35%	98.44%	100.00%	100.00%	97.45%	98.57%
>0.55	99.33%	81.17%	99.33%	81.17%	99.31%	100.00%	100.00%	100.00%	97.58%	98.57%
>0.6	99.33%	81.17%	99.33%	81.17%	100.00%	100.00%	100.00%	100.00%	97.58%	98.57%
>0.65	99.33%	81.17%	99.33%	81.17%	100.00%	100.00%	100.00%	100.00%	97.58%	98.53%
>0.7	99.33%	81.17%	99.33%	81.17%	100.00%	100.00%	100.00%	100.00%	97.58%	100.00%
>0.75	99.33%	81.17%	99.33%	81.17%	100.00%	100.00%	100.00%	100.00%	97.75%	100.00%
>0.8	99.33%	81.17%	99.33%	81.17%	100.00%	100.00%	100.00%	100.00%	97.75%	100.00%
>0.85	99.33%	81.17%	99.33%	81.17%	100.00%	100.00%	100.00%	100.00%	98.21%	100.00%
>0.9	99.33%	81.17%	99.33%	81.17%	N/A	N/A	100.00%	100.00%	98.89%	100.00%
>0.95	99.33%	81.17%	99.33%	81.17%	N/A	N/A	100.00%	100.00%	99.40%	100.00%

Table 10.11: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on recall in the service retrieval simulation in the health service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	93.33%	91.50%	100.00%	100.00%
>0.05	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	93.33%	82.67%	100.00%	96.50%
>0.1	100.00%	100.00%	100.00%	100.00%	99.67%	99.67%	93.33%	79.83%	100.00%	92.67%
>0.15	100.00%	100.00%	100.00%	100.00%	99.17%	98.33%	78.50%	78.67%	99.50%	85.83%
>0.2	100.00%	100.00%	100.00%	100.00%	98.17%	97.17%	78.50%	78.17%	99.17%	82.00%
>0.25	100.00%	100.00%	100.00%	100.00%	92.33%	91.83%	76.33%	76.33%	97.33%	77.83%
>0.3	100.00%	100.00%	100.00%	100.00%	86.00%	87.33%	75.83%	75.50%	94.83%	75.83%
>0.35	99.17%	99.17%	99.17%	99.17%	82.50%	85.17%	73.00%	72.67%	92.17%	73.17%
>0.4	99.17%	99.17%	99.17%	99.17%	78.67%	81.00%	71.67%	71.00%	89.67%	70.50%

>0.45	99.17%	99.17%	99.17%	99.17%	75.83%	77.33%	70.67%	70.67%	89.17%	66.50%
>0.5	94.33%	94.33%	94.33%	94.33%	71.17%	71.33%	69.83%	68.83%	88.00%	66.00%
>0.55	94.33%	94.33%	94.33%	94.33%	67.83%	67.83%	65.33%	64.67%	87.67%	65.50%
>0.6	94.33%	94.33%	94.33%	94.33%	41.83%	44.67%	63.33%	62.33%	87.67%	63.83%
>0.65	94.33%	94.33%	94.33%	94.33%	36.33%	40.33%	61.33%	61.33%	87.67%	61.83%
>0.7	92.83%	92.83%	92.83%	92.83%	32.17%	36.17%	56.50%	56.83%	87.67%	60.17%
>0.75	92.83%	92.83%	92.83%	92.83%	22.00%	28.67%	53.00%	52.83%	87.17%	58.17%
>0.8	92.83%	92.83%	92.83%	92.83%	10.00%	16.17%	50.67%	49.50%	87.17%	56.67%
>0.85	92.83%	92.83%	92.83%	92.83%	2.83%	9.00%	46.67%	45.00%	86.17%	53.17%
>0.9	92.83%	92.83%	92.83%	92.83%	0.00%	0.00%	44.67%	42.50%	85.83%	52.50%
>0.95	92.83%	92.83%	92.83%	92.83%	0.00%	0.00%	40.17%	36.17%	84.33%	52.50%

Table 10.12: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on F-measure(β=1) in the service retrieval simulation in the health service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	25.16%	24.78%	25.16%	24.78%	25.21%	24.83%	2.68%	2.60%	69.57%	68.30%
>0.05	25.16%	24.78%	25.16%	24.78%	25.24%	24.88%	2.68%	58.62%	69.57%	71.75%
>0.1	25.16%	24.78%	25.16%	24.78%	26.43%	26.50%	2.68%	63.46%	69.93%	79.29%
>0.15	25.16%	24.78%	25.16%	24.78%	29.94%	31.15%	70.52%	68.75%	71.20%	80.94%
>0.2	25.16%	24.78%	25.16%	24.78%	37.32%	41.21%	75.06%	72.49%	73.37%	80.93%
>0.25	25.45%	25.09%	25.45%	25.09%	44.47%	49.46%	76.71%	74.13%	75.33%	78.93%
>0.3	25.45%	25.09%	25.45%	25.09%	55.86%	61.51%	78.67%	75.55%	80.23%	78.90%
>0.35	29.42%	28.97%	29.42%	28.97%	70.45%	72.98%	78.93%	77.15%	79.98%	77.77%
>0.4	29.42%	28.97%	29.42%	28.97%	82.93%	83.00%	79.63%	77.44%	80.73%	77.08%
>0.45	29.42%	28.97%	29.42%	28.97%	82.09%	81.58%	81.09%	79.33%	81.96%	75.39%
>0.5	93.19%	92.38%	93.19%	92.38%	79.95%	78.63%	80.73%	78.92%	81.87%	75.25%
>0.55	93.19%	92.38%	93.19%	92.38%	79.13%	78.35%	77.95%	77.44%	82.47%	75.32%
>0.6	93.19%	92.38%	93.19%	92.38%	58.37%	60.93%	76.83%	76.43%	82.73%	74.67%
>0.65	93.19%	92.38%	93.19%	92.38%	53.30%	57.33%	75.68%	75.66%	83.10%	73.23%
>0.7	94.23%	93.58%	94.23%	93.58%	48.68%	52.98%	71.86%	72.12%	84.77%	72.88%
>0.75	94.23%	93.66%	94.23%	93.66%	36.07%	44.43%	68.93%	68.78%	86.51%	72.19%
>0.8	94.23%	93.66%	94.23%	93.66%	18.18%	27.83%	66.92%	65.87%	86.51%	70.98%
>0.85	94.23%	93.66%	94.23%	93.66%	5.51%	16.51%	63.31%	61.73%	85.94%	68.55%
>0.9	94.23%	93.66%	94.23%	93.66%	N/A	N/A	61.43%	59.32%	86.55%	68.44%
>0.95	94.23%	93.66%	94.23%	93.66%	N/A	N/A	57.00%	52.80%	85.78%	68.44%

Table 10.13: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on F-measure(β=2) in the service retrieval simulation in the health service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	45.66%	45.17%	45.66%	45.17%	45.73%	45.23%	6.42%	6.23%	85.11%	84.34%
>0.05	45.66%	45.17%	45.66%	45.17%	45.77%	45.30%	6.42%	71.02%	85.11%	84.80%
>0.1	45.66%	45.17%	45.66%	45.17%	47.27%	47.36%	6.42%	72.37%	85.32%	86.81%

>0.15	45.66%	45.17%	45.66%	45.17%	51.52%	52.79%	75.10%	74.37%	85.85%	83.80%
>0.2	45.66%	45.17%	45.66%	45.17%	59.41%	62.97%	77.09%	75.79%	86.94%	81.57%
>0.25	46.05%	45.57%	46.05%	45.57%	64.54%	68.40%	76.48%	75.44%	87.15%	78.27%
>0.3	46.05%	45.57%	46.05%	45.57%	70.73%	74.78%	76.94%	75.52%	88.39%	77.03%
>0.35	50.90%	50.36%	50.90%	50.36%	77.22%	79.84%	75.26%	74.39%	86.87%	74.94%
>0.4	50.90%	50.36%	50.90%	50.36%	80.32%	81.79%	74.65%	73.44%	85.86%	72.99%
>0.45	50.90%	50.36%	50.90%	50.36%	78.22%	78.98%	74.50%	73.90%	86.14%	69.79%
>0.5	93.87%	93.54%	93.87%	93.54%	74.44%	74.08%	73.82%	72.54%	85.44%	69.41%
>0.55	93.87%	93.54%	93.87%	93.54%	71.94%	71.68%	69.85%	69.24%	85.51%	69.10%
>0.6	93.87%	93.54%	93.87%	93.54%	47.18%	50.01%	68.12%	67.30%	85.62%	67.77%
>0.65	93.87%	93.54%	93.87%	93.54%	41.63%	45.76%	66.37%	66.36%	85.78%	65.94%
>0.7	93.39%	93.13%	93.39%	93.13%	37.22%	41.42%	61.78%	62.10%	86.48%	64.68%
>0.75	93.39%	93.16%	93.39%	93.16%	26.07%	33.41%	58.40%	58.23%	86.90%	63.07%
>0.8	93.39%	93.16%	93.39%	93.16%	12.20%	19.42%	56.12%	54.96%	86.90%	61.64%
>0.85	93.39%	93.16%	93.39%	93.16%	3.52%	11.00%	52.15%	50.47%	86.08%	58.41%
>0.9	93.39%	93.16%	93.39%	93.16%	N/A	N/A	50.14%	47.94%	86.12%	57.89%
>0.95	93.39%	93.16%	93.39%	93.16%	N/A	N/A	45.55%	41.38%	84.91%	57.89%

Table 10.14: Performance of the ECBR, IECBR, VSM, LSI and PM algorithms on fallout in the service retrieval simulation in the health service domain

Threshold value	ECBR (OFF)	ECBR (ON)	IECBR (OFF)	IECBR (ON)	VSM (OFF)	VSM (ON)	LSI (OFF)	LSI (ON)	PM (OFF)	PM (ON)
>0	2.29%	2.38%	2.29%	2.38%	2.28%	2.37%	21.43%	21.45%	1.22%	1.32%
>0.05	2.29%	2.38%	2.29%	2.38%	2.27%	2.35%	21.43%	1.02%	1.22%	0.76%
>0.1	2.29%	2.38%	2.29%	2.38%	2.00%	1.99%	21.43%	0.52%	1.08%	0.39%
>0.15	2.29%	2.38%	2.29%	2.38%	1.54%	1.46%	0.30%	0.32%	0.87%	0.20%
>0.2	2.29%	2.38%	2.29%	2.38%	1.06%	0.98%	0.18%	0.22%	0.69%	0.14%
>0.25	2.23%	2.31%	2.23%	2.31%	0.70%	0.64%	0.12%	0.16%	0.53%	0.12%
>0.3	2.23%	2.31%	2.23%	2.31%	0.43%	0.39%	0.08%	0.12%	0.32%	0.10%
>0.35	1.82%	1.90%	1.82%	1.90%	0.21%	0.21%	0.05%	0.07%	0.29%	0.08%
>0.4	1.82%	1.90%	1.82%	1.90%	0.07%	0.08%	0.03%	0.05%	0.24%	0.07%
>0.45	1.82%	1.90%	1.82%	1.90%	0.06%	0.06%	0.01%	0.03%	0.21%	0.05%
>0.5	0.05%	0.06%	0.05%	0.06%	0.04%	0.04%	0.01%	0.02%	0.18%	0.05%
>0.55	0.05%	0.06%	0.05%	0.06%	0.02%	0.02%	0.01%	0.01%	0.17%	0.04%
>0.6	0.05%	0.06%	0.05%	0.06%	0.01%	0.01%	0.00%	0.00%	0.16%	0.04%
>0.65	0.05%	0.06%	0.05%	0.06%	0.00%	0.00%	0.00%	0.00%	0.16%	0.04%
>0.7	0.02%	0.03%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%	0.13%	0.02%
>0.75	0.02%	0.03%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%	0.08%	0.01%
>0.8	0.02%	0.03%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%	0.08%	0.01%
>0.85	0.02%	0.03%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%	0.08%	0.01%
>0.9	0.02%	0.03%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%	0.07%	0.00%
>0.95	0.02%	0.03%	0.02%	0.03%	0.00%	0.00%	0.00%	0.00%	0.07%	0.00%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we put more weights on them when we determine the optimal threshold values for the algorithms. Therefore, we choose two groups of optimal threshold values respectively based on F-measure(β=1) and F-measure(β=2).

Table 10.15 shows the comparison results of the ECBR, IECBR, VSM, LSI and PM algorithms on their optimal threshold values for F-measure(β=1) as well as the Yellowpages® search engine. First of all, since the ECBR and the IECBR follow the similar principle and the only difference is their efficiency, these two algorithms have the same optimal threshold value and the best performance. It can be seen that the ECBR and IECBR algorithms have the highest scores on the performance indicators of precision, recall, and F-measure(β=1). Second, the WordNet API reduces the performance of the ECBR and the IECBR on nearly all the performance indictors except for being equal on recall. Third, since we cannot obtain the exact number of non-relevant items in the database of the Yellowpages® search engine for a query, the fallout rate for the Yellowpages® search engine cannot be computed. Similarly, there is no threshold value that can be found from the Yellowpages® search engine. It can be observed that the performance of all the algorithms employed by the Service Search Engine surpass the Yellowpages® search engine from all the performance indicators (excluding fallout)

Table 10.15: Comparison of the ECBR, IECBR, VSM, LSI and PM algorithm on their optimal threshold values for F-measure(β=1) as well as the Yellowpages® search engine in the service retrieval simulation in the health service domain

Algorithm	Optimal threshold value	Precision	Mean Average Precision	Recall	F-measure(β=1)	Fallout
ECBR(OFF)	>0.7	95.67%	99.33%	92.83%	94.23%	0.02%
ECBR(ON)	>0.75	94.50%	81.17%	92.83%	93.66%	0.03%
IECBR(OFF)	>0.7	95.67%	99.33%	92.83%	94.23%	0.02%
IECBR(ON)	>0.75	94.50%	81.17%	92.83%	93.66%	0.03%
VSM(OFF)	>0.4	87.69%	98.47%	78.67%	82.93%	0.07%
VSM(ON)	>0.4	85.10%	92.32%	81.00%	83.00%	0.08%
LSI(OFF)	>0.45	95.11%	100.00%	70.67%	81.09%	0.01%
LSI(ON)	>0.45	90.43%	98.44%	70.67%	79.33%	0.03%
PM(OFF)	>0.9	87.28%	98.89%	85.83%	86.55%	0.07%
PM(ON)	>0.15	76.57%	86.05%	85.83%	80.94%	0.20%
YP	N/A	22.41%	77.33%	42.27%	29.29%	N/A

Table 10.16 shows the comparison results of the ECBR, IECBR, VSM, LSI and PM algorithm on their optimal threshold values for F-measure(β=2) as well as the Yellowpages® search engine. First, due to the preference of F-measure(β=2) on recall, it can be seen that the ECBR and IECBR algorithm has the highest scores on recall and F-measure(β=2). Second, the WordNet API reduces the performance of the ECBR and the IECBR on nearly all the performance indictors except an equal on recall. Third, it can be observed that all the algorithms employed in the Service Search Engine doubles the scores of the Yellowpages® search engine on recall and F-measure(β=2).

Table 10.16: Comparison of the ECBR, IECBR, VSM, LSI and PM algorithm on their optimal threshold values for F-measure(β=2) as well as the Yellowpages® search engine in the service retrieval simulation in the health service domain

Algorithm	Optimal Threshold Value	Precision	Mean Average Precision	Recall	F-measure(β=2)	Fallout
ECBR(OFF)	>0.5	92.08%	99.33%	94.33%	93.87%	0.05%
ECBR(ON)	>0.5	90.50%	81.17%	94.33%	93.54%	0.06%
IECBR(OFF)	>0.5	92.08%	99.33%	94.33%	93.87%	0.05%
IECBR(ON)	>0.5	90.50%	81.17%	94.33%	93.54%	0.06%
VSM(OFF)	>0.4	87.69%	98.47%	78.67%	80.32%	0.07%
VSM(ON)	>0.4	85.10%	92.32%	81.00%	81.79%	0.08%
LSI(OFF)	>0.2	71.90%	95.99%	78.50%	77.09%	0.18%
LSI(ON)	>0.2	67.59%	93.80%	78.17%	75.79%	0.22%
PM(OFF)	>0.3	69.52%	95.29%	94.83%	88.39%	0.32%
PM(ON)	>0.1	69.29%	82.21%	92.67%	86.81%	0.39%
YP	N/A	22.41%	77.33%	42.27%	35.90%	N/A

Fig. 10.22 presents the comparison of the ECBR, IECBR, VSM, LSI and PM algorithms on response time. As the Yellowpages® search engine cannot record the response time, we exclude it from this comparison. First, it can be seen that the PM is the fastest algorithm, followed by the IECBR and the VSM which have a difference of 0.001s, and the last is the ECBR. Second, the WordNet API increase the time cost of each algorithm, which are nearly 6 times for the ECBR and more than 15 times for the IECBR, etc. This is because the WordNet API increases the number of the query words by finding their synonyms, thereby affecting the efficiency of the algorithms in the Service Search Engine.

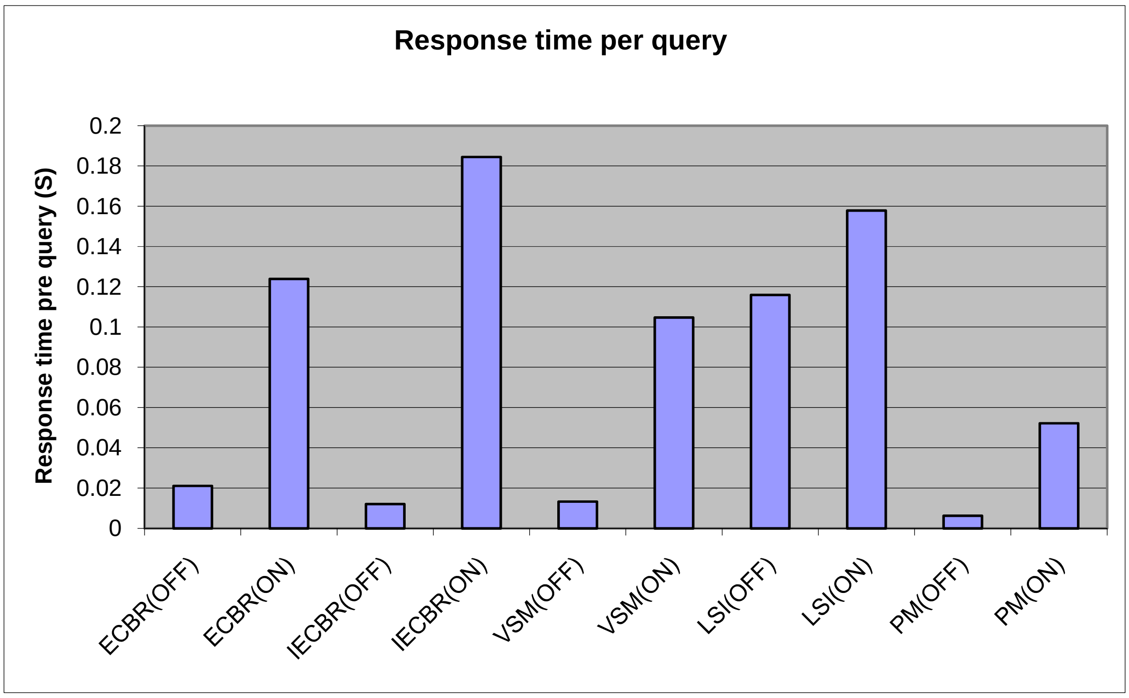


Figure 10.22 Comparison of the ECBR, IECBR, VSM, LSI and PM algorithms on response time in the service retrieval simulation in the health service domain

The comparison results can be summarised according to three tasks as follows:

1. In this experiment, the ECBR and IECBR algorithms show their advantage on precision, recall, F-measure(β=1), and F-measure(β=2) when compared with the traditional information retrieval algorithms. The efficiency of the IECBR is higher than the ECBR in terms of response time.

2. In this experiment, the WordNet API does not remarkably enhance the performance of the Service Search Engine and, in contrast, the WordNet API obviously reduces the efficiency of the Service Search Engine.

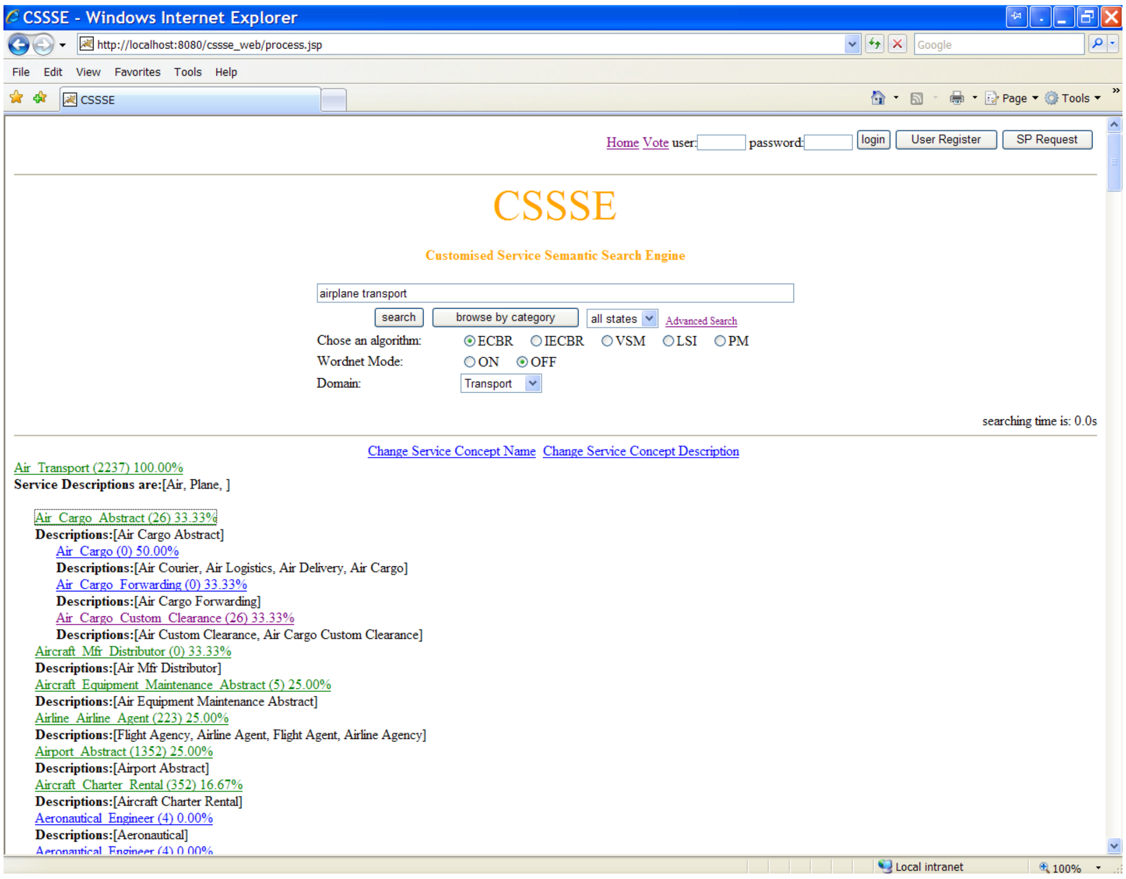
3. In this experiment, the Service Search Engine shows an overall advantage over the ordinary search engines.

10.4.8 Service Retrieval Functional Testing

In the Generic Service Retrieval Module of the service retrieval methodology, after a list of ordered service concepts is returned from the algorithm-based matching for a user query, the Service Search Engine still needs to interact with the user by means of a series of HCIs, in order to assist the user to denote his/her service request. In order to validate the mechanism for the HCIs, we run the prototype of the Generic Service Retrieval Module and test its functions.

Fig. 10.23 is a screenshot of the Generic Service Retrieval Module in the Service Search Engine. As can be seen, a user searches for a service with the terms "*airplane transport*". The service search engine returns a list of ranked service concepts from the transport service ontology, and each service concept is identified by its URIs and described by its *conceptDescription* property values under the URI of the concept. If the user selects a service concept that is a parent concept in an ontology ("*Air_Transport*" in this example), the search interface displays all its subconcepts described and ranked in the same manner as their parent concept, which can help the user to denote his/her query intention. Similar to the prior part, the user then needs to choose a concept among these subconcepts. In this case, the user chooses an "*Air_Cargo_Abstract*" concept and this concept still has subconcepts. Therefore, all its subconcepts are displayed to the user. These subconcepts are bottom-level concepts in the transport service ontology as the number of URIs of SDE metadata stored in their *linkedMetadata* property are displayed after the URIs of the concepts, e.g., "*Air_Cargo_Custom_Clearance (26)*".

Therefore, from the above descriptions, it can be seen that the Generic Service Retrieval Module uses the service ontologies to interact with service requesters, in order to help them to disambiguate their service queries and denote their query intentions.


CSSSE - Windows Internet Explorer
http://localhost:8080/cssse_web/process.jsp
File Edit View Favorites Tools Help
CSSSE
Home Vote user: password: login User Register SP Request

CSSSE

Customised Service Semantic Search Engine

airplane transport

search browse by category all states Advanced Search

Chose an algorithm: [✓ ECBR] IECBR VSM LSI PM
Wordnet Mode: ON [✓ OFF]
Domain: Transport

searching time is: 0.0s

Change Service Concept Name Change Service Concept Description

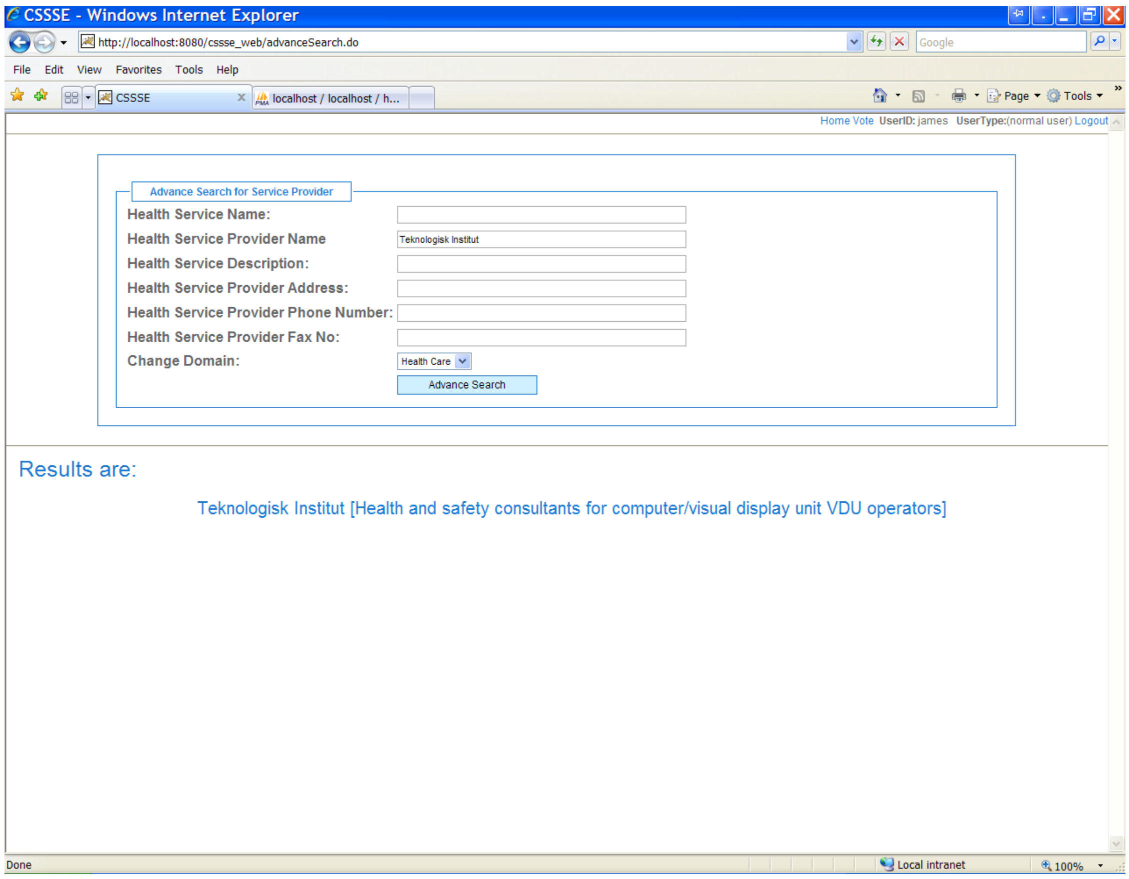
Air_Transport (2237) 100.00%
Service Descriptions are:[Air, Plane,]

Air_Cargo_Abstract (26) 33.33%
Descriptions:[Air Cargo Abstract]
Air_Cargo (0) 50.00%
Descriptions:[Air Courier, Air Logistics, Air Delivery, Air Cargo]
Air_Cargo_Forwarding (0) 33.33%
Descriptions:[Air Cargo Forwarding]
Air_Cargo_Custom_Clearance (26) 33.33%
Descriptions:[Air Custom Clearance, Air Cargo Custom Clearance]
Aircraft_Mfr_Distributor (0) 33.33%
Descriptions:[Air Mfr Distributor]
Aircraft_Equipment_Maintenance_Abstract (5) 25.00%
Descriptions:[Air Equipment Maintenance Abstract]
Airline_Airline_Agent (223) 25.00%
Descriptions:[Flight Agency, Airline Agent, Flight Agent, Airline Agency]
Airport_Abstract (1352) 25.00%
Descriptions:[Airport Abstract]
Aircraft_Charter_Rental (352) 16.67%
Descriptions:[Aircraft Charter Rental]
Aeronautical_Engineer (4) 0.00%
Descriptions:[Aeronautical]
Aeronautical_Engineer (4) 0.00%

Figure 10.23: Screenshot of the Generic Service Retrieval Module in the Service Search Engine

Compared with the users of the Generic Service Retrieval Module, the users of the Specific Service Retrieval Module have relevant domain knowledge regarding their service queries and thus they do not need the function of query disambiguation. Therefore, they use the SPARQL-based service retrieval approach to directly retrieve SDE metadata by searching for their property values. Fig. 10.24 displays a screenshot of the Specific Service Retrieval Module in the Service Search Engine. In this use case, the user (ID: James) requests a health and safety consultant service for computer visual display unit (VDU) operators and he knows the extract name of a service provider ("*Teknologisk Institut*") who can provide this service. Hence, he searches a SDE metadata by configuring the restriction on its *providerName* property. The SPARQL-based retrieval module then returns a correct SDE Metadata.

As a consequence, based on the screenshot and the descriptions of the prototype of the Specific Service Retrieval Module, we can deduce that the functions of this module are basically realized and validated by the functional testing.


CSSSE - Windows Internet Explorer
http://localhost:8080/cssse_web/advanceSearch.do
File Edit View Favorites Tools Help
CSSSE
localhost / localhost / h...
Page Tools
Home Vote

UserID: james UserType:(normal user) Logout

Advance Search for Service Provider

Health Service Name:	
Health Service Provider Name	Teknologisk Institut
Health Service Description:	
Health Service Provider Address:	
Health Service Provider Phone Number:	
Health Service Provider Fax No:	
Change Domain:	Health Care

Advance Search

Results are:

Teknologisk Institut [Health and safety consultants for computer/visual display unit VDU operators]

Figure 10.24: Screenshot of the Specific Service Retrieval Module in the Service Search Engine

10.4.9 Conclusion

In this section, we have completed three tasks. First of all, we validated the use of the ECBR and IECBR algorithms in the service retrieval methodology by comparing them with three classical information retrieval algorithms. The experimental result shows the advantages of the ECBR and IECBR in the aggregated performance indicators and the IECBR is more efficient than the ECBR. Second, we found that the WordNet API does not show obvious improvement to the performance of the Service Search Engine and, in contrast, it visibly reduces the search efficiency. Third, by comparing with a commercial search engine – Australian Yellowpages® search engine, our prototype reveals its outstanding technical advantages from all the performance indicators. In addition, by horizontally comparing the experimental results between the transport service domain and the health service domain, it is found that the algorithms used in the latter all perform better in the former. This is because of the differences between terms within domains: in the health service domain, the terms used to describe service concepts and service queries are more specific, compared with more generic terms used in the transport service domain. These specific terms increase search precision and thus enhance the performance of the algorithms to some extent.

Next, we ran the prototype of the Generic Service Retrieval Module in order to test its interactive function for assisting a service requester to denote his/her query intention, and the prototype of the Specific Service Retrieval Module for testing its SPARQL-based search functions. The functional testing proves the feasibility of the service retrieval methodology by realizing these two functions.

10.5 Service Concept Recommendation Simulation

In this section, we validate the service concept recommendation methodology by means of the simulation approach: first of all, we simulate these two proposed semantic similarity models in a service concept recommendation system; secondly, we compare these two models with four existing semantic similarity models based on several performance indicators adopted from the information retrieval field.

10.5.1 Steps of Service Concept Recommendation Simulation

The simulation of the service concept recommendation methodology can be divided into three steps as follows:

Step 1. We test the performance of these two semantic similarity models respectively on the transport service ontology and the health service ontology in terms of the performance indicators introduced in the next section. As stated in Chapter 7, each of our model is the weighted arithmetic mean of two sub-models, where the weight is determined by a β value (0 <= β <= 1). Hence, based on the performance scores, we need to choose an optimal β value and an optimal threshold value for each semantic similarity model in each ontology.

Step 2. We test the performance of the four existing semantic similarity models respectively on the transport service ontology and the health service ontology in terms of the performance indicators introduced in the next section. Based on the performance scores, we choose the optimal threshold value for each semantic similarity model in each ontology.

Step 3. We make a comparison between the best performances of the six semantic similarity models in each ontology.

10.5.2 Performance Indicators

In order to empirically compare our proposed model with the existing models, we utilize the six most widely used performance indicators from the information retrieval field as the evaluation metrics. The performance indicators in this experiment are defined as follows:

Precision in the information retrieval field is used to measure the preciseness of a search system [1]. In this experiment, Precision for a single service concept refers to the proportion of matched and logically similar service concept in all service concepts matched to this concept [4, 5], which can be represented by Equation 10.37 below:

$$\text{Precision(S)} = \frac{\text{Number of matched and logically similar concepts}}{\text{Number of matched concepts}} \tag{10.37}$$

With regard to the whole collection of service concepts in a service ontology, the total precision is the sum of the precision value for each concept normalized by the number of concepts in the collection [4, 5], which can be represented by Equation 10.38 below:

$$\text{Precision}=\frac{\sum_{i=1}^{n}\text{Precision}(\text{S}_i)}{n} \tag{10.38}$$

Before we introduce the definition of Mean Average Precision, the concept of Average Precision should be defined. In this experiment, Average Precision for a single service concept is the average of precision values after truncating a ranked service concept list matched by this concept after each of the logically similar concept for this concept [4, 5]. This indicator emphasizes the return of more logically similar concepts earlier, which can be represented as:

$$\begin{aligned}&\text{Average Precision(S)}\\&=\frac{\text{Sum(Precision @ Each logically similar concept in a list)}}{\text{Number of matched and logically similar concepts in a list}}\end{aligned} \tag{10.39}$$

Mean Average Precision refers to the average of the average precision values for the collection of service concepts in a service ontology [4, 5], which can be represented as:

$$\text{Mean average precision} = \frac{\sum_{i=1}^{n}\text{Average precision}(\text{S}_i)}{n} \tag{10.40}$$

Recall in the information retrieval field is used to measure the effectiveness of a search system [1]. In this experiment, Recall for a single concept is the proportion of matched and logically similar concepts in all concepts that are logically similar to this concept [4, 5], which can be represented by Equation 10.41 below:

$$\text{Recall(S)}=\frac{\text{Number of matched and logically similar concepts}}{\text{Number of logically similar concepts}} \tag{10.41}$$

With regard to the whole collection of service concepts in a service ontology, the total recall is the sum of the recall value for each concept normalized by the number of concepts in the collection [4, 5], which can be represented by Equation 10.42 below:

$$\text{Recall}=\frac{\sum_{i=1}^{n}\text{Recall}(\text{S}_i)}{n} \tag{10.42}$$

F-measure (or F-measure(β=1)) in the information retrieval field is used as an aggregated performance scale for a search system [1]. In this experiment, F-measure is the mean of precision and recall [4, 5], which can be represented below as:

$$\text{F-measure}(\beta\text{=1}) = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}} \tag{10.43}$$

When the F-measure value reaches the highest, it means the aggregated value between precision and recall reaches the highest point at the same time.

F-measure$^{\beta}$ is another measure that combines precision and recall, the difference being that users can specify the preference on recall or precision by configuring different weights [18]. In this experiment, we employ F-measure(β=2) that weights recall twice as much as precision, which is close to the fact that most search engines concern recall more than precision, as a result of most users' purposes in obtaining information [24]. F-measure(β=2) can be represented below as:

$$\text{F-measure}(\beta\text{=2})=\frac{(1+\beta^2)\cdot\text{Precision}\times\text{Recall}}{\beta^2\cdot\text{Precision+Recall}}=\frac{5\times\text{Precision}\times\text{Recall}}{4\times\text{Precision+Recall}} \tag{10.44}$$

All of the above indicators have the same limitation – they do not consider the amount of non-logically similar concepts in a matched concept collection of a concept. Furthermore, if there is no logically similar concept in the matched collection, recall cannot be defined. To resolve this issue, we need another performance indicator – Fallout. In this experiment, Fallout for a single service concept is the proportion of non-logically similar concept matched by this concept in the whole collection of non-logically similar concepts for this concept [4, 5], which can be represented as:

$$\text{Fallout(S)} = \frac{\text{Number of matched and non-logically similar concept}}{\text{Number of non-logically similar concept}} \tag{10.45}$$

With regard to the whole collection of service concepts in a service ontology, the total fallout value is the sum of the fallout value for each concept normalized by the number of concepts in the ontology [4, 5], which can be represented as:

$$\text{Fallout}=\frac{\sum_{i=1}^{n}\text{Fallout}(S_i)}{n} \tag{10.46}$$

In contrast to other performance indicators, the lower the fallout value, the better is the search performance.

10.5.3 Existing Semantic Similarity Models

Edge (Distance)-based Models. Edge-based models are based on the shortest path between two nodes in a definitional network. Definitional networks are a type of hierarchical/taxonomic semantic networks, in which all nodes are linked by *is-a* relations [22]. The models are based on the assumption that all nodes are evenly distributed and are of similar densities and the distance between any two nodes is equal. They can also be applied to a network structure.

One typical edge-based model is provided by Rada [16], and is described as:

For the two nodes C_1 and C_2 in a semantic network,

$$\text{Distance } (C_1,C_2) = \text{minimum number of edges seperating } C_1 \text{ and } C_2 \tag{10.47}$$

and the similarity between C_1 and C_2 is given by

$$sim_{Rada}(C_1,C_2) \;=\; 2\times Max-\text{Distance}(C_1,C_2) \tag{10.48}$$

where Max is the maximum depth of a definitional network.

In order to ensure that the interval of sim_{Rada} is between 0 and 1, Equation 10.48 can also be expressed as

$$sim_{Rada}(C_1,C_2)=1-\frac{\text{Distance}(C_1,C_2)}{2\times Max} \tag{10.49}$$

Node (Information Content)-based Models. Information content-based models are used to judge the semantic similarity between concepts in a definitional network or in a corpus, based on measuring the similarity by taking into account information content, namely the term occurrence in corpora or the subsumed nodes in taxonomies. These models can avoid the defect of the edge counting approaches which cannot control variable distances in a dense definitional network [17].

Resnik [17] developed such a model whereby the information shared by two concepts can be indicated by the concept which subsumes these two concepts in a taxonomy. Then, the similarity between these two concepts C_1 and C_2 can be mathematically expressed as follows:

$$sim_{\mathrm{Resnik}}(C_1, C_2) = \max_{C \in S(C_1, C_2)}[-\log(\mathrm{P}(C))] \tag{10.50}$$

where $S(C_1, C_2)$ is the set of concepts that subsume both C_1 and C_2, and P(C) is the possibility of encountering an instance of concept C.

Nevertheless, one limitation of Resnik's model is that its interval is [0, ∞], compared with the interval of [0, 1] of other models. With the purpose of according with the interval of other semantic similarity models, we normalize Resnik's model by given

$$|sim_{\mathrm{Resnik}}(\varsigma_1, \varsigma_2)| = \begin{cases} \dfrac{\max_{\varsigma \in S(\varsigma_1, \varsigma_2)}[-\log(\mathrm{P}(\varsigma))]}{\max_{\varsigma \in \Theta}[-\log(\mathrm{P}(\varsigma))]} & if\ \ \varsigma_1 \neq \varsigma_2 \\ 1 & if\ \ \varsigma_1 = \varsigma_2 \end{cases} \tag{10.51}$$

where Θ is the collection of pseudo-concepts in a lightweight ontology [4].

Lin [15]'s semantic similarity model is the extension of Resnik's model, which measures the similarity between two nodes as the ratio between the amount of commonly shared information of these two nodes and the amount of information of these two nodes, which can be mathematically expressed as follows:

$$sim_{Lin} = \frac{2 \times sim_{\mathrm{Re}snilk}(C_1, C_2)}{IC(C_1) + IC(C_2)} \tag{10.52}$$

Hybrid Models. Hybrid models consider multiple factors in their similarity measure. Jiang and Conath [14] developed a hybrid model that uses the node-based theory to enhance the edge-based model. Their method takes into account the factors of local density, node depth and link types. The weight between a child concept C and its parent concept P can be measured as:

$$wt(C,P) = (\beta + (1-\beta)\frac{\bar{E}}{E(P)})(\frac{d(P)+1}{d(P)})^{\alpha}(IC(C) - IC(P))T(C,P) \tag{10.53}$$

where d(P) is the depth of node P, E(P) is the number of edges in the child links, $\bar{E}$ is the average density of the whole hierarchy, T(C, P) represents the link type, and α and β ($\alpha \geq 0$, $0 \leq \beta \leq 1$) are the control parameters of the effect of node density and node depth on the weight.

The distance between two concepts is defined as follows:

$$\text{Distance}(C_1,C_2) = \sum_{C\in\{path(C_1,C_2)-LS(C_1,C_2)\}} wt(C, p(C)) \tag{10.54}$$

where path(C_1, C_2) is the set that contains all the nodes in the shortest path from C_1 to C_2, and LS(C_1, C_2) is the most informative concept that subsumes both C_1 and C_2.

In some special cases such as when only the link type is considered as the factor of weight computing (α=0, β=1, and T(C, P) =1), the distance algorithm can be simplified as follows:

$$\text{Distance}(C_1,C_2) = IC(C_1) + IC(C_2) - 2\times sim_{\text{Resnik}}(C_1,C_2) \tag{10.55}$$

where IC(C) = -log[P(C)].

Finally, the similarity value between two concepts C_1 and C_2 is measured by converting the semantic distance as follows:

$$sim_{Jiang\&Conath}(C_1,C_2) = 1-\text{Distance}(C_1,C_2) \tag{10.56}$$

In addition, Seco [21]'s research showed that the similarity equation can also be expressed as

$$sim_{Jiang\&Conath}(C_1,C_2) = 1-\frac{\text{Distance}(C_1,C_2)}{2} \tag{10.57}$$

The testing results show that the parameters α and β do not heavily influence the similarity computation [14].

10.5.4 Service Concept Recommendation Simulation in Transport Service Domain

In this section, we simulate our Semantic Similarity Model 1 (SSM-1) and Semantic Similarity Model 2 (SSM-2), Rada's model, Resnik's model, Lin's model, and Jiang and Conath's model in service concept recommendation in the transport service domain. in order to obtain the optimal β values for the SSM-1 and the SSM-2, we set the initial β value at 0.0, and increase it by 0.1 each time until 1.0. We then measure the performance scores of these models based on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout at each time of the increment of the β value. In addition, each of the six candidate semantic similarity models needs a threshold value to filter the non-similar concepts and the selection of the threshold values can impact the performance of the models. In order to obtain the optimal performance for the models, we set the initial threshold value at 0.0, and increase it by 0.05 each time until it reaches We then measure the performance scores of the six models based on the performance indicators at each time of the increment of the threshold value. Finally, we compare the performance of these models at their optimal threshold values.

The performance scores of the SSM-1 in the service concept recommendation simulation in the transport service domain on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown in Table 10.17 to Table 10.22.

Table 10.17: Precision of SSM-1 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	8.45%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%
>0.05	8.68%	3.73%	1.01%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%
>0.1	9.22%	8.82%	3.47%	1.59%	0.95%	0.50%	0.50%	0.50%	0.50%	0.50%	0.50%
>0.15	10.51%	9.66%	8.74%	3.51%	1.64%	1.51%	0.95%	0.50%	0.50%	0.50%	0.50%
>0.2	12.02%	10.95%	10.19%	8.25%	3.58%	1.66%	1.56%	1.45%	0.95%	0.50%	0.50%
>0.25	14.85%	13.36%	11.74%	10.87%	8.58%	3.69%	1.68%	1.65%	1.55%	1.15%	0.97%
>0.3	16.95%	16.81%	14.59%	12.69%	11.29%	8.79%	3.81%	1.72%	1.72%	1.63%	1.53%
>0.35	19.78%	18.69%	18.31%	15.99%	13.86%	11.20%	9.24%	3.91%	1.74%	1.74%	1.71%
>0.4	22.82%	21.88%	21.86%	19.91%	17.47%	14.79%	16.02%	9.90%	3.95%	1.74%	1.74%
>0.45	23.34%	24.60%	22.96%	24.18%	21.45%	18.62%	16.02%	13.34%	6.30%	4.01%	1.74%
>0.5	23.07%	23.83%	26.73%	25.90%	26.72%	22.65%	19.21%	16.81%	13.84%	6.44%	4.62%
>0.55	24.52%	26.38%	27.28%	29.34%	29.29%	29.10%	24.05%	18.77%	16.23%	14.40%	5.82%
>0.6	27.16%	28.17%	31.17%	30.85%	33.14%	32.93%	31.52%	23.07%	18.32%	16.29%	14.07%
>0.65	35.37%	36.97%	36.56%	37.16%	40.43%	39.13%	35.47%	29.21%	21.82%	18.25%	16.35%
>0.7	39.30%	42.58%	48.62%	48.33%	45.63%	47.44%	43.22%	39.88%	27.32%	22.22%	16.53%
>0.75	41.59%	44.71%	50.33%	51.09%	56.79%	50.33%	48.29%	46.47%	38.44%	26.57%	25.68%
>0.8	38.11%	35.67%	43.91%	53.90%	56.88%	56.44%	68.91%	56.71%	51.80%	32.50%	26.51%
>0.85	33.78%	48.21%	59.52%	61.36%	76.19%	80.56%	77.08%	75.86%	68.80%	59.92%	28.72%
>0.9	47.22%	62.50%	68.75%	85.71%	85.71%	68.75%	68.75%	75.00%	80.77%	80.56%	60.85%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Table 10.18: Mean average precision of SSM-1 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	53.38%	53.85%	56.66%	59.59%	61.21%	61.58%	62.48%	58.09%	56.00%	55.16%	48.94%
>0.05	53.37%	55.31%	56.66%	59.59%	61.21%	61.58%	62.48%	58.09%	56.00%	55.16%	48.94%
>0.1	54.72%	56.00%	58.14%	59.59%	61.21%	61.58%	62.48%	58.09%	56.00%	55.16%	48.94%
>0.15	56.02%	56.20%	58.77%	61.12%	61.21%	61.58%	62.48%	58.09%	56.00%	55.16%	48.94%
>0.2	57.90%	59.16%	59.06%	61.53%	62.62%	61.58%	62.48%	58.09%	56.00%	55.16%	48.94%
>0.25	58.49%	60.65%	61.65%	61.92%	62.75%	62.92%	62.48%	58.09%	56.00%	55.16%	48.94%
>0.3	62.47%	61.43%	63.41%	64.67%	63.34%	63.10%	63.78%	58.09%	56.00%	55.16%	48.94%
>0.35	66.66%	66.00%	65.11%	66.59%	65.25%	63.63%	64.00%	59.33%	56.00%	55.16%	48.94%

>0.4	67.50%	69.07%	68.58%	67.74%	67.89%	65.43%	66.03%	59.57%	57.19%	55.16%	48.94%
>0.45	68.57%	69.92%	72.12%	71.89%	70.31%	68.04%	66.03%	60.72%	57.32%	56.27%	48.94%
>0.5	73.29%	72.09%	74.14%	74.66%	74.31%	72.12%	69.40%	61.70%	59.05%	57.19%	52.75%
>0.55	79.14%	77.02%	78.78%	77.22%	77.21%	75.75%	72.70%	64.23%	59.98%	60.90%	53.16%
>0.6	82.68%	82.88%	80.46%	81.17%	82.48%	80.16%	76.83%	68.85%	62.87%	61.43%	54.12%
>0.65	84.86%	85.75%	86.15%	86.33%	82.92%	84.45%	83.33%	73.76%	67.83%	61.73%	54.44%
>0.7	89.36%	90.09%	91.03%	92.33%	92.12%	87.59%	86.27%	81.42%	78.78%	67.47%	54.98%
>0.75	93.62%	93.29%	94.30%	94.58%	94.52%	95.42%	95.97%	91.82%	89.60%	92.76%	91.24%
>0.8	94.61%	95.32%	98.15%	96.55%	98.39%	97.77%	97.95%	98.21%	96.19%	95.21%	92.66%
>0.85	93.75%	96.88%	100%	100%	100%	100%	100%	100%	100%	98.80%	93.39%
>0.9	90.00%	100%	100%	100%	100%	100%	100%	100%	100%	100%	97.77%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 10.19: Recall of SSM-1 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	66.89%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%
>0.05	66.55%	69.13%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%
>0.1	64.54%	66.89%	69.13%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%
>0.15	62.64%	65.88%	67.34%	69.13%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%
>0.2	58.72%	62.08%	65.88%	68.01%	69.13%	71.14%	71.14%	71.14%	71.14%	71.14%	71.14%
>0.25	57.20%	59.28%	62.75%	66.33%	68.68%	69.13%	71.14%	71.14%	71.14%	71.14%	71.14%
>0.3	51.59%	56.98%	59.40%	62.86%	67.00%	68.68%	69.13%	71.14%	71.14%	71.14%	71.14%
>0.35	46.57%	50.36%	56.64%	59.51%	63.98%	67.67%	68.68%	69.13%	71.14%	71.14%	71.14%
>0.4	45.29%	46.17%	51.34%	56.73%	60.63%	64.43%	65.21%	68.68%	69.13%	71.14%	71.14%
>0.45	39.72%	43.34%	45.20%	50.00%	55.73%	61.07%	65.21%	66.89%	68.90%	69.13%	71.14%
>0.5	33.74%	37.38%	41.36%	44.36%	49.59%	55.34%	61.02%	65.04%	66.05%	67.67%	64.99%
>0.55	27.98%	31.38%	34.37%	38.49%	43.26%	49.47%	55.50%	61.41%	64.65%	62.64%	64.77%
>0.6	25.62%	26.11%	30.35%	31.58%	35.16%	40.19%	49.23%	53.38%	60.40%	61.74%	62.64%
>0.65	23.91%	24.92%	25.15%	25.95%	30.21%	32.29%	35.29%	44.49%	51.70%	61.07%	62.42%
>0.7	19.28%	19.96%	21.45%	21.79%	21.12%	24.44%	27.73%	33.26%	37.84%	51.25%	61.07%
>0.75	14.14%	14.47%	15.48%	14.25%	15.15%	14.47%	15.93%	19.85%	25.69%	23.34%	23.55%
>0.8	7.85%	6.40%	7.07%	7.40%	8.08%	8.19%	10.20%	11.21%	15.67%	22.09%	23.21%
>0.85	4.50%	4.50%	3.83%	4.16%	4.50%	4.36%	5.37%	6.71%	7.83%	13.69%	22.87%
>0.9	2.48%	1.81%	1.68%	1.68%	1.68%	1.68%	1.68%	2.35%	3.02%	4.36%	7.94%
>0.95	0.34%	0.34%	0.34%	0.34%	0.34%	0.34%	0.34%	0.34%	0.34%	0.34%	0.34%

Table 10.20: F-measure(β=1) of SSM-1 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	15.01%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%
>0.05	15.36%	7.08%	2.00%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%

>0.1	16.13%	15.58%	6.60%	3.10%	1.87%	0.99%	0.99%	0.99%	0.99%	0.99%	0.99%
>0.15	18.00%	16.85%	15.47%	6.68%	3.20%	2.96%	1.87%	0.99%	0.99%	0.99%	0.99%
>0.2	19.95%	18.61%	17.65%	14.71%	6.81%	3.24%	3.05%	2.84%	1.87%	0.99%	0.99%
>0.25	23.58%	21.81%	19.77%	18.68%	15.26%	7.00%	3.28%	3.22%	3.03%	2.26%	1.92%
>0.3	25.52%	25.96%	23.42%	21.12%	19.33%	15.58%	7.22%	3.36%	3.35%	3.18%	3.00%
>0.35	27.77%	27.26%	27.67%	25.21%	22.78%	19.22%	16.29%	7.40%	3.40%	3.40%	3.35%
>0.4	30.35%	29.69%	30.67%	29.48%	27.13%	24.05%	25.72%	17.31%	7.46%	3.40%	3.40%
>0.45	29.41%	31.38%	30.45%	32.59%	30.97%	28.54%	25.72%	22.25%	11.55%	7.59%	3.40%
>0.5	27.40%	29.11%	32.47%	32.71%	34.73%	32.15%	29.23%	26.72%	22.89%	11.77%	8.63%
>0.55	26.13%	28.66%	30.42%	33.30%	34.93%	36.64%	33.56%	28.76%	25.94%	23.42%	10.67%
>0.6	26.37%	27.10%	30.75%	31.21%	34.12%	36.20%	38.43%	32.21%	28.11%	25.78%	22.98%
>0.65	28.54%	29.77%	29.80%	30.56%	34.58%	35.39%	35.38%	35.26%	30.69%	28.10%	25.92%
>0.7	25.87%	27.17%	29.77%	30.04%	28.87%	32.26%	33.78%	36.27%	31.73%	31.00%	26.02%
>0.75	21.10%	21.87%	23.68%	22.28%	23.91%	22.48%	23.95%	27.82%	30.80%	24.85%	24.57%
>0.8	13.02%	10.85%	12.18%	13.02%	14.14%	14.30%	17.77%	18.72%	24.06%	26.30%	24.75%
>0.85	7.94%	8.23%	7.19%	7.79%	8.49%	8.28%	10.04%	12.33%	14.06%	22.29%	25.47%
>0.9	4.72%	3.52%	3.28%	3.29%	3.29%	3.28%	3.28%	4.56%	5.82%	8.28%	14.05%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Table 10.21: F-measure(β=2) of SSM-1 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	28.07%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%
>0.05	28.52%	15.35%	4.80%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%
>0.1	29.34%	28.86%	14.44%	7.28%	4.50%	2.43%	2.43%	2.43%	2.43%	2.43%	2.43%
>0.15	31.45%	30.44%	28.76%	14.58%	7.51%	6.97%	4.50%	2.43%	2.43%	2.43%	2.43%
>0.2	33.04%	32.10%	31.48%	27.77%	14.84%	7.59%	7.16%	6.70%	4.50%	2.43%	2.43%
>0.25	36.43%	35.13%	33.57%	32.83%	28.61%	15.19%	7.68%	7.55%	7.13%	5.40%	4.62%
>0.3	36.62%	38.55%	36.79%	35.11%	33.73%	29.06%	15.60%	7.85%	7.83%	7.45%	7.06%
>0.35	36.64%	37.61%	39.92%	38.54%	37.12%	33.69%	30.04%	15.93%	7.94%	7.92%	7.82%
>0.4	37.84%	37.78%	40.44%	41.42%	40.58%	38.55%	40.40%	31.40%	16.06%	7.94%	7.94%
>0.45	34.83%	37.61%	37.86%	41.20%	42.23%	41.95%	40.40%	37.11%	23.08%	16.28%	7.94%
>0.5	30.88%	33.57%	37.28%	38.83%	42.34%	42.94%	42.52%	41.33%	37.65%	23.33%	18.00%
>0.55	27.21%	30.23%	32.67%	36.23%	39.49%	43.40%	43.99%	42.23%	40.49%	37.51%	21.39%
>0.6	25.91%	26.50%	30.51%	31.43%	34.73%	38.49%	44.26%	42.27%	41.39%	39.63%	37.06%
>0.65	25.57%	26.66%	26.82%	27.62%	31.82%	33.46%	35.33%	40.27%	40.59%	41.57%	39.92%
>0.7	21.47%	22.33%	24.15%	24.48%	23.66%	27.07%	29.87%	34.40%	35.13%	40.63%	39.69%
>0.75	16.29%	16.74%	17.97%	16.65%	17.75%	16.88%	18.39%	22.42%	27.52%	23.93%	23.94%
>0.8	9.33%	7.65%	8.49%	8.95%	9.75%	9.88%	12.30%	13.35%	18.21%	23.60%	23.80%
>0.85	5.44%	5.49%	4.71%	5.11%	5.54%	5.38%	6.60%	8.21%	9.52%	16.19%	23.84%
>0.9	3.06%	2.25%	2.08%	2.09%	2.09%	2.08%	2.08%	2.91%	3.74%	5.38%	9.61%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Table 10.22: Fallout of SSM-1 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	7.56%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%
>0.05	7.04%	16.24%	53.71%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%
>0.1	6.23%	6.92%	17.41%	36.11%	56.80%	97.80%	97.80%	97.80%	97.80%	97.80%	97.80%
>0.15	5.29%	6.03%	6.79%	17.24%	35.60%	37.92%	56.80%	97.80%	97.80%	97.80%	97.80%
>0.2	4.23%	5.00%	5.73%	7.14%	17.01%	35.18%	36.64%	38.53%	56.79%	97.80%	97.80%
>0.25	3.27%	3.87%	4.65%	5.33%	7.09%	16.64%	34.82%	35.09%	37.00%	47.82%	55.64%
>0.3	2.48%	2.86%	3.46%	4.21%	4.91%	6.79%	16.12%	34.17%	34.25%	35.34%	36.89%
>0.35	1.94%	2.18%	2.53%	3.14%	3.86%	5.06%	6.41%	15.73%	33.78%	33.84%	33.99%
>0.4	1.47%	1.63%	1.84%	2.22%	2.88%	3.55%	3.23%	6.05%	15.61%	33.75%	33.76%
>0.45	1.11%	1.20%	1.36%	1.53%	1.94%	2.62%	3.23%	4.39%	9.98%	15.39%	33.75%
>0.5	0.85%	0.90%	0.96%	1.08%	1.28%	1.74%	2.52%	3.16%	4.28%	9.75%	13.60%
>0.55	0.59%	0.61%	0.66%	0.72%	0.88%	1.12%	1.60%	2.54%	3.37%	4.02%	10.91%
>0.6	0.41%	0.41%	0.44%	0.47%	0.56%	0.70%	0.99%	1.56%	2.58%	3.29%	4.06%
>0.65	0.26%	0.25%	0.27%	0.29%	0.33%	0.41%	0.56%	0.85%	1.54%	2.79%	3.43%
>0.7	0.17%	0.16%	0.15%	0.15%	0.17%	0.21%	0.28%	0.40%	0.70%	1.46%	3.21%
>0.75	0.11%	0.10%	0.09%	0.08%	0.07%	0.09%	0.11%	0.17%	0.29%	0.43%	0.50%
>0.8	0.07%	0.05%	0.04%	0.03%	0.03%	0.03%	0.03%	0.05%	0.09%	0.31%	0.46%
>0.85	0.04%	0.02%	0.01%	0.01%	0.01%	0.00%	0.01%	0.01%	0.02%	0.05%	0.39%
>0.9	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.02%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we assign more weight to them when determining the optimal threshold values for the SSM-1. Therefore, we choose two groups of optimal β values and optimal threshold values respectively based on the highest F-measure(β=1) and F-measure(β=2) scores.

The performance scores of the SSM-2 in the service concept recommendation simulation in the transport service domain on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown from Table 10.23 to Table 10.28.

Table 10.23: Precision of SSM-2 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	8.45%	1.71%	1.71%	1.71%	1.71%	1.71%	1.71%	1.71%	1.71%	1.71%	1.79%
>0.05	8.68%	8.25%	7.77%	4.13%	1.73%	1.73%	1.74%	1.75%	1.79%	1.78%	1.79%
>0.1	9.22%	9.22%	8.90%	8.54%	8.41%	4.81%	4.36%	4.33%	1.78%	1.79%	1.79%
>0.15	10.51%	9.99%	10.26%	10.10%	10.05%	9.86%	9.93%	5.70%	5.90%	5.97%	7.20%
>0.2	12.02%	12.02%	11.87%	12.21%	12.11%	12.17%	12.37%	12.54%	12.79%	8.54%	8.48%
>0.25	14.85%	15.17%	14.58%	14.54%	15.20%	15.56%	15.89%	16.19%	16.57%	16.54%	17.85%

>0.3	16.95%	17.82%	18.23%	18.09%	17.69%	18.39%	19.10%	19.09%	18.81%	18.91%	18.42%
>0.35	19.78%	19.86%	21.06%	20.81%	20.97%	20.31%	20.55%	20.21%	20.06%	19.83%	19.54%
>0.4	22.82%	21.95%	23.75%	24.90%	23.92%	23.26%	21.90%	21.24%	20.95%	20.41%	20.43%
>0.45	23.34%	25.04%	26.94%	29.08%	28.42%	29.33%	25.47%	23.49%	22.90%	23.65%	21.48%
>0.5	23.07%	25.65%	28.36%	31.23%	33.40%	32.23%	31.79%	32.27%	29.46%	22.47%	22.54%
>0.55	24.52%	27.40%	31.06%	35.51%	40.16%	37.21%	35.97%	33.15%	31.13%	27.87%	26.15%
>0.6	27.16%	30.54%	37.04%	43.45%	40.81%	39.78%	35.50%	35.08%	35.06%	34.04%	30.47%
>0.65	35.37%	37.17%	41.02%	44.06%	42.45%	36.79%	36.06%	36.59%	35.11%	35.14%	35.67%
>0.7	39.30%	47.27%	47.14%	40.28%	43.39%	40.42%	44.87%	39.40%	40.85%	38.97%	35.73%
>0.75	41.59%	44.41%	45.10%	52.59%	48.50%	50.83%	50.30%	48.12%	44.46%	45.50%	41.24%
>0.8	38.11%	47.67%	48.68%	57.58%	61.14%	54.61%	54.63%	47.45%	41.98%	41.57%	47.05%
>0.85	33.78%	50.00%	58.82%	71.43%	72.22%	76.19%	62.87%	57.41%	49.76%	36.97%	36.97%
>0.9	47.22%	59.09%	66.67%	61.11%	81.25%	75.00%	75.00%	77.00%	56.60%	37.47%	36.97%
>0.95	0.00%	50.00%	50.00%	66.67%	66.67%	75.00%	75.00%	81.25%	75.00%	56.60%	36.97%

Table 10.24: Mean average precision of SSM-2 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	53.38%	54.52%	57.15%	58.89%	59.61%	59.42%	57.89%	56.66%	56.65%	56.65%	45.61%
>0.05	53.37%	56.57%	59.26%	59.19%	59.61%	59.42%	57.89%	56.66%	56.65%	56.86%	45.61%
>0.1	54.72%	56.87%	59.26%	61.03%	61.79%	59.68%	58.15%	57.13%	56.86%	56.86%	45.61%
>0.15	56.02%	58.41%	60.04%	61.61%	62.26%	62.29%	60.72%	57.55%	57.53%	57.53%	48.23%
>0.2	57.90%	60.21%	62.20%	62.89%	63.57%	62.99%	60.95%	59.67%	59.92%	58.33%	48.23%
>0.25	58.49%	61.52%	64.62%	64.50%	63.86%	63.49%	62.41%	61.42%	61.68%	63.37%	52.09%
>0.3	62.47%	63.16%	65.45%	67.13%	65.65%	64.96%	63.41%	62.06%	63.89%	64.50%	52.09%
>0.35	66.66%	67.38%	67.19%	67.86%	68.87%	67.18%	64.24%	62.94%	63.92%	64.53%	52.38%
>0.4	67.50%	69.30%	70.37%	70.02%	70.41%	69.39%	66.38%	64.84%	65.06%	64.53%	52.38%
>0.45	68.57%	70.69%	72.65%	73.39%	74.09%	71.00%	69.37%	68.06%	66.03%	65.70%	57.32%
>0.5	73.29%	72.53%	74.98%	74.59%	74.95%	73.83%	72.99%	70.79%	69.52%	64.06%	58.84%
>0.55	79.14%	76.83%	79.38%	80.69%	78.96%	77.94%	77.00%	73.42%	70.56%	68.11%	62.45%
>0.6	82.68%	82.42%	82.88%	84.38%	82.98%	80.78%	79.34%	78.07%	75.64%	74.50%	70.35%
>0.65	84.86%	87.20%	89.55%	87.94%	86.86%	86.71%	84.33%	82.27%	82.87%	82.39%	81.35%
>0.7	89.36%	89.19%	90.91%	91.08%	88.92%	88.19%	82.23%	83.20%	83.22%	81.70%	82.16%
>0.75	93.62%	91.47%	92.08%	93.80%	92.91%	88.95%	88.65%	82.68%	81.92%	82.32%	80.93%
>0.8	94.61%	94.04%	97.73%	96.95%	97.32%	95.21%	89.94%	86.95%	80.20%	80.22%	81.19%
>0.85	93.75%	96.43%	95.45%	95.45%	96.67%	96.28%	94.91%	89.93%	80.41%	79.21%	75.08%
>0.9	90.00%	92.86%	92.86%	100%	100%	100%	100%	99.06%	89.57%	79.18%	75.08%
>0.95	N/A	100%	100%	100%	100%	100%	100%	100%	100%	89.57%	75.08%

Table 10.25: Recall of SSM-2 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1

>0	66.89%	70.81%	70.81%	70.81%	70.81%	70.81%	70.81%	70.81%	70.81%	70.81%	70.47%
>0.05	66.55%	67.67%	67.67%	70.13%	70.81%	70.81%	70.81%	70.81%	70.81%	70.47%	70.47%
>0.1	64.54%	66.55%	67.67%	67.67%	67.67%	70.13%	70.13%	69.80%	70.47%	70.47%	70.47%
>0.15	62.64%	64.21%	65.94%	66.67%	67.00%	66.67%	66.67%	69.13%	69.13%	69.13%	65.94%
>0.2	58.72%	61.63%	63.20%	64.93%	65.32%	66.00%	66.33%	66.33%	66.00%	67.95%	65.94%
>0.25	57.20%	59.17%	60.01%	62.92%	64.60%	65.32%	64.65%	64.32%	63.76%	61.63%	60.29%
>0.3	51.59%	55.62%	58.02%	59.51%	61.91%	62.92%	63.09%	63.09%	60.96%	60.29%	60.29%
>0.35	46.57%	48.83%	54.59%	56.88%	57.40%	59.19%	62.02%	62.19%	60.74%	60.07%	60.07%
>0.4	45.29%	45.84%	48.49%	53.13%	54.31%	56.71%	58.50%	59.23%	59.36%	60.07%	60.07%
>0.45	39.72%	42.13%	44.14%	46.79%	49.61%	53.41%	53.58%	54.34%	57.28%	57.91%	51.85%
>0.5	33.74%	36.51%	38.19%	41.39%	43.94%	45.10%	45.62%	49.56%	51.28%	49.50%	49.83%
>0.55	27.98%	30.67%	32.00%	32.95%	37.24%	35.73%	39.17%	36.88%	40.27%	43.29%	44.97%
>0.6	25.62%	26.06%	28.03%	28.72%	28.04%	27.49%	26.54%	28.47%	30.65%	31.82%	35.57%
>0.65	23.91%	22.63%	22.27%	22.28%	19.54%	17.82%	18.86%	19.72%	20.93%	22.48%	24.83%
>0.7	19.28%	19.90%	17.20%	10.88%	12.68%	12.91%	15.55%	17.23%	17.67%	19.30%	19.80%
>0.75	14.14%	12.66%	8.91%	8.59%	8.68%	10.87%	12.19%	15.44%	15.94%	17.84%	18.46%
>0.8	7.85%	7.40%	5.84%	6.38%	6.77%	7.34%	9.19%	10.68%	13.81%	13.87%	17.00%
>0.85	4.50%	3.66%	3.36%	3.19%	4.36%	5.09%	6.11%	8.02%	11.63%	12.53%	12.53%
>0.9	2.48%	2.01%	2.18%	1.85%	2.18%	2.52%	3.69%	5.09%	7.85%	12.47%	12.53%
>0.95	0.34%	0.50%	0.50%	0.84%	0.84%	1.01%	1.68%	2.18%	3.69%	7.85%	12.53%

Table 10.26: F-measure(β=1) of SSM-2 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	15.01%	3.35%	3.35%	3.35%	3.35%	3.35%	3.35%	3.35%	3.35%	3.35%	3.49%
>0.05	15.36%	14.71%	13.94%	7.80%	3.38%	3.38%	3.40%	3.41%	3.49%	3.47%	3.49%
>0.1	16.13%	16.19%	15.73%	15.16%	14.97%	9.00%	8.20%	8.16%	3.47%	3.49%	3.49%
>0.15	18.00%	17.30%	17.76%	17.54%	17.48%	17.18%	17.29%	10.54%	10.87%	10.99%	12.98%
>0.2	19.95%	20.12%	19.98%	20.56%	20.43%	20.55%	20.85%	21.10%	21.43%	15.18%	15.03%
>0.25	23.58%	24.15%	23.46%	23.63%	24.61%	25.13%	25.51%	25.86%	26.31%	26.09%	27.55%
>0.3	25.52%	27.00%	27.74%	27.75%	27.52%	28.46%	29.32%	29.31%	28.74%	28.78%	28.22%
>0.35	27.77%	28.23%	30.39%	30.47%	30.72%	30.24%	30.87%	30.51%	30.16%	29.81%	29.49%
>0.4	30.35%	29.68%	31.88%	33.91%	33.22%	32.99%	31.87%	31.26%	30.97%	30.47%	30.49%
>0.45	29.41%	31.41%	33.46%	35.87%	36.14%	37.86%	34.53%	32.80%	32.72%	33.58%	30.37%
>0.5	27.40%	30.13%	32.55%	35.60%	37.95%	37.60%	37.47%	39.09%	37.42%	30.91%	31.04%
>0.55	26.13%	28.94%	31.52%	34.19%	38.64%	36.45%	37.50%	34.92%	35.11%	33.91%	33.07%
>0.6	26.37%	28.12%	31.91%	34.59%	33.24%	32.51%	30.37%	31.43%	32.71%	32.89%	32.82%
>0.65	28.54%	28.13%	28.87%	29.60%	26.76%	24.01%	24.77%	25.63%	26.22%	27.42%	29.28%
>0.7	25.87%	28.01%	25.21%	17.14%	19.63%	19.57%	23.09%	23.97%	24.67%	25.81%	25.48%
>0.75	21.10%	19.71%	14.89%	14.77%	14.72%	17.91%	19.63%	23.37%	23.47%	25.63%	25.50%
>0.8	13.02%	12.82%	10.43%	11.48%	12.19%	12.94%	15.74%	17.44%	20.79%	20.80%	24.98%
>0.85	7.94%	6.82%	6.35%	6.10%	8.23%	9.54%	11.13%	14.07%	18.86%	18.71%	18.71%
>0.9	4.72%	3.89%	4.22%	3.58%	4.25%	4.87%	7.04%	9.55%	13.79%	18.71%	18.71%
>0.95	0.00%	1.00%	1.00%	1.66%	1.66%	1.99%	3.28%	4.25%	7.04%	13.79%	18.71%

Table 10.27: F-measure(β=2) of SSM-2 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	28.07%	7.82%	7.82%	7.82%	7.82%	7.82%	7.82%	7.82%	7.82%	7.82%	8.12%
>0.05	28.52%	27.73%	26.63%	16.71%	7.88%	7.90%	7.92%	7.96%	8.13%	8.07%	8.12%
>0.1	29.34%	29.65%	29.16%	28.37%	28.09%	18.87%	17.45%	17.35%	8.07%	8.12%	8.12%
>0.15	31.45%	30.79%	31.62%	31.45%	31.41%	30.97%	31.12%	21.44%	22.00%	22.18%	25.06%
>0.2	33.04%	33.77%	33.89%	34.85%	34.77%	35.02%	35.42%	35.71%	36.02%	28.42%	28.00%
>0.25	36.43%	37.45%	36.97%	37.78%	39.15%	39.84%	40.06%	40.33%	40.63%	39.89%	40.86%
>0.3	36.62%	39.05%	40.39%	40.82%	41.28%	42.39%	43.19%	43.18%	42.09%	41.93%	41.45%
>0.35	36.64%	37.80%	41.40%	42.24%	42.60%	42.81%	44.19%	43.94%	43.21%	42.73%	42.46%
>0.4	37.84%	37.64%	40.13%	43.31%	43.31%	44.04%	43.84%	43.62%	43.43%	43.26%	43.27%
>0.45	34.83%	37.07%	39.14%	41.71%	43.17%	45.88%	43.89%	43.03%	44.05%	44.90%	40.41%
>0.5	30.88%	33.66%	35.71%	38.86%	41.33%	41.77%	41.96%	44.77%	44.66%	39.90%	40.12%
>0.55	27.21%	29.95%	31.81%	33.44%	37.79%	36.01%	38.49%	36.07%	38.04%	38.97%	39.31%
>0.6	25.91%	26.85%	29.47%	30.81%	29.91%	29.30%	27.95%	29.58%	31.44%	32.24%	34.42%
>0.65	25.57%	24.55%	24.51%	24.73%	21.91%	19.87%	20.85%	21.72%	22.77%	24.23%	26.44%
>0.7	21.47%	22.51%	19.71%	12.74%	14.78%	14.94%	17.89%	19.41%	19.94%	21.46%	21.74%
>0.75	16.29%	14.77%	10.62%	10.32%	10.39%	12.90%	14.37%	17.86%	18.29%	20.31%	20.75%
>0.8	9.33%	8.91%	7.09%	7.76%	8.23%	8.87%	11.03%	12.64%	15.96%	16.00%	19.49%
>0.85	5.44%	4.49%	4.14%	3.94%	5.37%	6.26%	7.45%	9.69%	13.74%	14.44%	14.44%
>0.9	3.06%	2.50%	2.70%	2.29%	2.71%	3.12%	4.56%	6.26%	9.49%	14.39%	14.44%
>0.95	0.00%	0.63%	0.63%	1.05%	1.05%	1.25%	2.09%	2.71%	4.56%	9.49%	14.44%

Table 10.28: Fallout of SSM-2 in the service concept recommendation simulation in the transport service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	7.56%	34.99%	34.99%	34.99%	34.99%	34.99%	34.99%	34.99%	34.99%	34.99%	33.10%
>0.05	7.04%	7.64%	9.28%	19.82%	34.83%	34.77%	34.65%	34.47%	33.98%	33.32%	33.10%
>0.1	6.23%	6.43%	6.85%	7.79%	8.43%	17.79%	18.73%	18.58%	33.32%	33.10%	33.10%
>0.15	5.29%	5.56%	5.86%	5.85%	6.33%	6.84%	6.93%	16.01%	16.22%	16.60%	16.12%
>0.2	4.23%	4.46%	4.55%	4.74%	4.62%	4.81%	5.12%	5.48%	5.59%	14.70%	14.94%
>0.25	3.27%	3.37%	3.44%	3.85%	3.82%	3.71%	3.79%	4.15%	4.39%	4.60%	4.53%
>0.3	2.48%	2.55%	2.59%	2.66%	3.04%	3.12%	3.10%	3.34%	3.62%	3.82%	4.23%
>0.35	1.94%	1.92%	1.94%	1.97%	2.30%	2.48%	2.68%	2.87%	3.19%	3.44%	3.72%
>0.4	1.47%	1.44%	1.43%	1.44%	1.50%	1.91%	2.17%	2.44%	2.59%	3.09%	3.20%
>0.45	1.11%	1.07%	1.01%	0.99%	1.08%	1.38%	1.57%	1.83%	2.21%	2.29%	2.49%
>0.5	0.85%	0.78%	0.71%	0.70%	0.74%	0.82%	1.14%	1.23%	1.49%	1.89%	2.07%
>0.55	0.59%	0.53%	0.48%	0.47%	0.48%	0.55%	0.87%	0.93%	1.02%	1.26%	1.51%
>0.6	0.41%	0.35%	0.31%	0.29%	0.31%	0.36%	0.43%	0.72%	0.78%	0.85%	1.02%
>0.65	0.26%	0.23%	0.20%	0.18%	0.19%	0.22%	0.30%	0.59%	0.61%	0.65%	0.70%
>0.7	0.17%	0.14%	0.11%	0.10%	0.11%	0.13%	0.17%	0.28%	0.55%	0.56%	0.62%
>0.75	0.11%	0.08%	0.06%	0.04%	0.05%	0.07%	0.10%	0.20%	0.52%	0.53%	0.56%
>0.8	0.07%	0.04%	0.03%	0.02%	0.02%	0.04%	0.06%	0.11%	0.24%	0.51%	0.52%

>0.85	0.04%	0.02%	0.01%	0.01%	0.01%	0.01%	0.02%	0.05%	0.13%	0.50%	0.51%
>0.9	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.01%	0.05%	0.24%	0.50%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.05%	0.50%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we put more weights on them when determining the optimal threshold values for the SSM-2. Therefore, we choose two groups of optimal β values and optimal threshold values respectively based on the highest F-measure(β=1) and F-measure(β=2) scores.

The performance scores of the Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model in the service concept recommendation simulation in the transport service domain on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown from Table 10.29 to Table 10.34.

Table 10.29: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on precision in the service concept recommendation simulation in the transport service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	0.97%	1.79%	1.79%	18.46%
>0.05	0.97%	1.79%	1.79%	18.20%
>0.1	0.97%	1.79%	1.79%	23.31%
>0.15	0.97%	7.20%	3.23%	23.49%
>0.2	1.51%	8.48%	6.92%	24.47%
>0.25	1.51%	17.85%	11.12%	25.06%
>0.3	1.51%	18.42%	13.07%	27.96%
>0.35	4.59%	19.54%	16.80%	28.31%
>0.4	4.59%	20.43%	19.81%	31.25%
>0.45	4.59%	21.48%	19.80%	35.31%
>0.5	13.80%	22.54%	20.32%	40.04%
>0.55	13.80%	26.15%	21.63%	47.23%
>0.6	13.80%	30.47%	23.59%	50.28%
>0.65	13.80%	35.67%	26.58%	55.44%
>0.7	25.99%	35.73%	30.76%	66.67%
>0.75	25.99%	41.24%	39.56%	70.56%
>0.8	25.99%	47.05%	52.12%	50.00%
>0.85	0.00%	36.97%	58.71%	0.00%
>0.9	0.00%	36.97%	0.00%	0.00%
>0.95	0.00%	36.97%	0.00%	0.00%

Table 10.30: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on mean average precision in the service concept recommendation simulation in the transport service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	44.37%	45.61%	48.66%	73.62%

>0.05	44.37%	45.61%	48.66%	79.41%
>0.1	44.37%	45.61%	48.66%	86.53%
>0.15	44.37%	48.23%	51.20%	86.53%
>0.2	44.37%	48.23%	51.20%	89.13%
>0.25	44.37%	52.09%	55.08%	89.13%
>0.3	44.37%	52.09%	55.88%	89.56%
>0.35	47.82%	52.38%	55.88%	89.56%
>0.4	47.82%	52.38%	56.69%	90.41%
>0.45	47.82%	57.32%	62.53%	91.64%
>0.5	49.14%	58.84%	63.94%	93.22%
>0.55	49.14%	62.45%	68.30%	96.41%
>0.6	49.14%	70.35%	77.91%	96.48%
>0.65	49.14%	81.35%	89.68%	96.59%
>0.7	86.38%	82.16%	93.51%	98.17%
>0.75	86.38%	80.93%	96.03%	98.17%
>0.8	86.38%	81.19%	96.52%	100.00%
>0.85	N/A	75.08%	97.77%	N/A
>0.9	N/A	75.08%	N/A	N/A
>0.95	N/A	75.08%	N/A	N/A

Table 10.31: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on recall in the service concept recommendation simulation in the transport service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	71.14%	70.47%	70.47%	39.43%
>0.05	71.14%	70.47%	70.47%	34.40%
>0.1	71.14%	70.47%	70.47%	29.70%
>0.15	71.14%	65.94%	66.11%	29.70%
>0.2	71.14%	65.94%	66.11%	27.68%
>0.25	71.14%	60.29%	60.85%	26.01%
>0.3	71.14%	60.29%	60.40%	25.00%
>0.35	64.99%	60.07%	60.40%	25.00%
>0.4	64.99%	60.07%	59.40%	24.11%
>0.45	64.99%	51.85%	51.68%	22.43%
>0.5	62.64%	49.83%	50.00%	19.24%
>0.55	62.64%	44.97%	45.97%	13.31%
>0.6	62.64%	35.57%	38.09%	11.97%
>0.65	62.64%	24.83%	28.36%	8.28%
>0.7	23.55%	19.80%	23.32%	5.82%
>0.75	23.55%	18.46%	18.79%	5.82%
>0.8	23.55%	17.00%	13.20%	0.67%
>0.85	0.34%	12.53%	7.94%	0.34%
>0.9	0.34%	12.53%	0.34%	0.34%
>0.95	0.34%	12.53%	0.34%	0.34%

Table 10.32: Comparison of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on F-measure(β=1) in the service concept recommendation simulation in the transport service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	1.92%	3.49%	3.49%	25.15%
>0.05	1.92%	3.49%	3.49%	23.80%
>0.1	1.92%	3.49%	3.49%	26.12%
>0.15	1.92%	12.98%	6.17%	26.23%
>0.2	2.95%	15.03%	12.53%	25.98%
>0.25	2.95%	27.55%	18.81%	25.53%
>0.3	2.95%	28.22%	21.49%	26.40%
>0.35	8.57%	29.49%	26.29%	26.55%
>0.4	8.57%	30.49%	29.71%	27.22%
>0.45	8.57%	30.37%	28.63%	27.43%
>0.5	22.62%	31.04%	28.90%	25.99%
>0.55	22.62%	33.07%	29.42%	20.77%
>0.6	22.62%	32.82%	29.13%	19.34%
>0.65	22.62%	29.28%	27.44%	14.40%
>0.7	24.71%	25.48%	26.53%	10.70%
>0.75	24.71%	25.50%	25.48%	10.75%
>0.8	24.71%	24.98%	21.06%	1.32%
>0.85	0.00%	18.71%	13.99%	0.00%
>0.9	0.00%	18.71%	0.00%	0.00%
>0.95	0.00%	18.71%	0.00%	0.00%

Table 10.33: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on F-measure(β=2) in the service concept recommendation simulation in the transport service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	4.62%	8.12%	8.12%	32.13%
>0.05	4.62%	8.12%	8.12%	29.20%
>0.1	4.62%	8.12%	8.12%	28.15%
>0.15	4.62%	25.06%	13.52%	28.21%
>0.2	6.94%	28.00%	24.39%	26.98%
>0.25	6.94%	40.86%	32.13%	25.81%
>0.3	6.94%	41.45%	35.03%	25.54%
>0.35	17.90%	42.46%	39.77%	25.60%
>0.4	17.90%	43.27%	42.43%	25.26%
>0.45	17.90%	40.41%	39.09%	24.19%
>0.5	36.68%	40.12%	38.70%	21.47%
>0.55	36.68%	39.31%	37.53%	15.54%
>0.6	36.68%	34.42%	33.92%	14.12%
>0.65	36.68%	26.44%	27.98%	9.97%
>0.7	24.00%	21.74%	24.51%	7.12%

>0.75	24.00%	20.75%	21.00%	7.12%
>0.8	24.00%	19.49%	15.52%	0.84%
>0.85	0.00%	14.44%	9.60%	0.00%
>0.9	0.00%	14.44%	0.00%	0.00%
>0.95	0.00%	14.44%	0.00%	0.00%

Table 10.34: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on fallout in the service concept recommendation simulation in the transport service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	55.64%	33.10%	33.10%	1.36%
>0.05	55.64%	33.10%	33.10%	1.19%
>0.1	55.64%	33.10%	33.10%	0.74%
>0.15	55.64%	16.12%	18.25%	0.71%
>0.2	37.35%	14.94%	14.13%	0.63%
>0.25	37.35%	4.53%	5.54%	0.58%
>0.3	37.35%	4.23%	4.09%	0.46%
>0.35	13.90%	3.72%	3.05%	0.44%
>0.4	13.90%	3.20%	2.68%	0.37%
>0.45	13.90%	2.49%	2.11%	0.28%
>0.5	4.24%	2.07%	1.82%	0.19%
>0.55	4.24%	1.51%	1.30%	0.09%
>0.6	4.24%	1.02%	0.83%	0.07%
>0.65	4.24%	0.70%	0.51%	0.03%
>0.7	0.50%	0.62%	0.37%	0.01%
>0.75	0.50%	0.56%	0.18%	0.01%
>0.8	0.50%	0.52%	0.07%	0.00%
>0.85	0.00%	0.51%	0.03%	0.00%
>0.9	0.00%	0.50%	0.00%	0.00%
>0.95	0.00%	0.50%	0.00%	0.00%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we put more weights on them when determining the optimal threshold values for the four existing models. Therefore, we choose four groups of optimal threshold values respectively based on the highest F-measure(β=1) and F-measure(β=2) scores.

Table 10.35 shows the comparison of the SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=1) in the service concept recommendation simulation in the transport service domain. It can be seen that the SSM-2 and the SSM-1 stand in the first and second position in F-measure(β=1). Additionally, the SSM-2 and SSM-1 respectively stand in the second and third positions for precision and recall.

Table 10.35: Comparison of SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=1) in the service concept recommendation simulation in the transport service domain

	Optimal threshold value	Precision	Mean Average Precision	Recall	F-measure(β=1)	Fallout
SSM-1(β=0.6)	>0.6	31.52%	76.83%	49.23%	38.43%	0.99%
SSM-2(β=0.7)	>0.5	32.27%	70.79%	49.56%	39.09%	1.23%
Rada	>0.7	25.99%	86.38%	23.55%	24.71%	0.50%
Normalized Resnik	>0.55	26.15%	62.45%	44.97%	33.07%	1.51%
Lin	>0.4	19.81%	56.69%	59.40%	29.71%	2.68%
Jiang and Conath	>0.45	35.31%	91.64%	22.43%	27.43%	0.28%

Table 10.36 shows the comparison of the SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=2) in the service concept recommendation simulation in the health service domain. It can be seen that the SSM-1 stands in the first position for precision, mean average precision and fallout, as well as the second position in F-measure(β=2); and the SSM-2 stand in the first position in F-measure(β=2) and the second position in precision.

Table 10.36: Comparison of SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=2) in the service concept recommendation simulation in the transport service domain

	Optimal threshold value	Precision	Mean Average Precision	Recall	F-measure(β=2)	Fallout
SSM-1(β=0.6)	>0.6	31.52%	76.83%	49.23%	44.26%	0.99%
SSM-2(β=0.5)	>0.45	29.33%	71.00%	53.41%	45.88%	1.38%
Rada	>0.5	13.80%	49.14%	62.64%	36.68%	4.24%
Normalized Resnik	>0.4	20.43%	52.38%	60.07%	43.27%	3.20%
Lin	>0.4	19.81%	56.69%	59.40%	42.43%	2.68%
Jiang and Conath	>0	18.46%	73.62%	39.43%	32.13%	1.36%

10.5.5 Service Concept Recommendation Simulation in Health Service Domain

In this section, we simulate our SSM-1 and SSM-2 model, Rada's model, Resnik's Lin's model, and Jiang and Conath's model in service concept recommendation in the health service domain. Next, in order to obtain the optimal β values for the SSM-1 and the SSM-2, we set the initial β value at 0.0, and increase it by 0.1 each time until 1.0. We then measure the performance scores of these models based on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout at each time of the increment of the β value. In addition, each of the six candidate semantic similarity models needs a threshold value to filter the non-similar concepts and the selection of the threshold values can impact the performance of the models. In order to obtain the optimal performance for the models, we set the initial threshold value at 0.0, and increase it by 0.05 each time until 0.95. Then we measure the performance scores of the six models based on the performance indicators at each time the threshold value is incremented. Finally, we compare the performance of these models at their optimal threshold values.

The performance scores of the SSM-1 in the service concept recommendation simulation in the health service domain on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown from Table 10.37 to Table 10.42.

Table 10.37: Precision of SSM-1 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	26.59%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%
>0.05	26.89%	9.23%	1.45%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%
>0.1	26.93%	27.14%	9.04%	5.02%	1.42%	1.22%	1.22%	1.22%	1.22%	1.22%	1.22%
>0.15	28.51%	28.45%	20.57%	8.98%	5.87%	3.20%	1.42%	1.22%	1.22%	1.22%	1.22%
>0.2	31.72%	30.90%	30.26%	18.51%	9.09%	6.39%	4.92%	3.01%	1.42%	1.22%	1.22%
>0.25	37.40%	35.33%	35.71%	28.48%	13.73%	9.27%	6.53%	5.36%	3.21%	1.44%	1.36%
>0.3	44.74%	39.36%	36.68%	36.36%	24.84%	13.07%	9.35%	6.63%	5.69%	4.69%	2.59%
>0.35	48.66%	45.18%	42.63%	37.98%	35.12%	20.39%	12.67%	8.98%	6.45%	5.60%	4.84%
>0.4	53.45%	50.42%	46.63%	41.63%	36.48%	24.21%	20.25%	12.93%	8.86%	6.48%	6.40%
>0.45	54.90%	54.77%	51.03%	47.50%	42.59%	35.61%	24.13%	14.88%	10.38%	8.90%	6.48%
>0.5	51.38%	53.88%	54.81%	50.31%	48.06%	43.67%	30.18%	21.08%	14.65%	10.47%	9.08%
>0.55	58.85%	47.59%	51.01%	53.43%	51.02%	46.00%	39.85%	25.64%	20.95%	14.20%	10.22%
>0.6	58.82%	62.50%	54.82%	54.77%	52.71%	46.59%	40.73%	30.06%	25.19%	21.17%	13.99%
>0.65	69.44%	70.51%	69.77%	63.82%	48.41%	52.17%	46.05%	46.29%	27.77%	20.07%	15.61%
>0.7	56.52%	64.29%	69.44%	78.95%	69.05%	52.16%	51.53%	47.92%	39.35%	22.23%	19.78%
>0.75	50.00%	50.00%	50.00%	66.67%	77.78%	73.91%	66.67%	51.67%	48.24%	29.02%	21.37%
>0.8	50.00%	50.00%	50.00%	50.00%	100%	100%	87.50%	73.91%	58.27%	45.49%	22.80%
>0.85	N/A	N/A	N/A	N/A	N/A	N/A	N/A	100%	100%	69.57%	25.43%
>0.9	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	100%	70.00%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 10.38: Mean average precision of SSM-1 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	65.93%	54.27%	53.39%	51.98%	49.01%	44.88%	40.49%	38.34%	36.72%	35.33%	29.16%

>0.05	65.97%	61.35%	53.51%	51.98%	49.01%	44.88%	40.49%	38.34%	36.72%	35.33%	29.16%
>0.1	66.67%	66.82%	60.36%	58.00%	49.13%	44.88%	40.49%	38.34%	36.72%	35.33%	29.16%
>0.15	73.74%	67.73%	66.09%	59.03%	54.85%	49.85%	40.66%	38.34%	36.72%	35.33%	29.16%
>0.2	78.66%	72.44%	68.49%	63.70%	56.65%	50.74%	45.63%	42.71%	36.89%	35.33%	29.16%
>0.25	81.64%	78.72%	71.52%	66.28%	59.05%	52.44%	46.73%	44.30%	41.83%	35.80%	31.19%
>0.3	82.74%	82.23%	78.78%	69.33%	64.03%	55.64%	49.08%	46.63%	45.50%	46.10%	40.31%
>0.35	84.61%	84.18%	81.89%	77.59%	68.19%	62.89%	56.52%	53.98%	52.64%	53.27%	42.88%
>0.4	85.16%	87.05%	86.94%	87.01%	78.96%	70.28%	62.44%	59.07%	57.47%	54.30%	43.23%
>0.45	88.72%	88.88%	87.89%	87.02%	85.30%	79.97%	69.26%	63.08%	58.60%	55.54%	43.23%
>0.5	93.75%	89.15%	88.39%	91.75%	87.55%	86.14%	77.99%	66.62%	60.32%	56.95%	46.64%
>0.55	94.57%	95.12%	92.22%	91.33%	89.03%	89.45%	88.25%	75.45%	67.07%	61.23%	47.48%
>0.6	95.59%	97.22%	94.59%	94.59%	93.42%	92.34%	89.85%	89.06%	75.40%	64.43%	48.89%
>0.65	94.44%	96.55%	96.88%	98.08%	97.92%	96.88%	95.79%	91.39%	88.67%	70.54%	49.25%
>0.7	96.43%	96.43%	96.15%	100%	100%	97.73%	94.44%	95.42%	93.78%	85.34%	65.98%
>0.75	91.67%	100%	100%	100%	100%	100%	97.50%	94.00%	94.97%	90.82%	90.19%
>0.8	100%	100%	100%	100%	100%	100%	100%	100%	93.91%	95.00%	90.19%
>0.85	N/A	N/A	N/A	N/A	N/A	N/A	N/A	100%	100%	100%	90.51%
>0.9	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	100%	100%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 10.39: Recall of SSM-1 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	75.39%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%
>0.05	75.10%	85.67%	98.05%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%
>0.1	70.31%	74.78%	85.53%	87.72%	98.05%	99.42%	99.42%	99.42%	99.42%	99.42%	99.42%
>0.15	57.46%	68.78%	73.17%	84.40%	87.55%	88.30%	98.05%	99.42%	99.42%	99.42%	99.42%
>0.2	50.01%	58.93%	68.44%	78.38%	81.89%	85.30%	87.22%	88.30%	98.05%	99.42%	99.42%
>0.25	44.12%	49.45%	60.17%	69.86%	79.21%	81.11%	83.16%	84.13%	86.13%	97.35%	89.53%
>0.3	41.34%	42.37%	47.17%	59.26%	68.47%	77.68%	79.24%	78.51%	76.53%	71.78%	64.73%
>0.35	38.09%	38.50%	41.20%	45.66%	54.39%	67.06%	67.47%	66.45%	65.54%	61.59%	61.35%
>0.4	32.49%	32.83%	31.48%	32.31%	38.62%	48.04%	60.41%	60.76%	59.16%	61.21%	61.21%
>0.45	24.63%	24.97%	25.81%	28.05%	26.69%	33.72%	44.21%	55.85%	57.89%	59.16%	61.21%
>0.5	16.59%	18.55%	20.84%	20.43%	23.61%	23.98%	32.16%	48.93%	54.79%	56.92%	55.56%
>0.55	14.25%	12.61%	12.68%	14.09%	17.41%	18.21%	20.83%	33.09%	48.25%	52.41%	54.78%
>0.6	10.08%	10.86%	10.28%	10.28%	8.43%	12.07%	14.70%	19.77%	31.80%	49.71%	52.41%
>0.65	7.55%	7.94%	9.11%	6.08%	5.50%	6.58%	8.13%	14.07%	17.62%	35.14%	51.62%
>0.7	4.68%	4.39%	2.67%	3.06%	4.82%	5.11%	5.67%	7.50%	11.53%	17.04%	34.75%
>0.75	2.73%	0.39%	0.97%	1.46%	2.77%	3.84%	4.82%	5.50%	8.28%	11.93%	12.55%
>0.8	0.39%	0.39%	0.39%	0.39%	0.39%	1.17%	2.77%	3.84%	4.92%	8.54%	12.22%
>0.85	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.58%	2.38%	3.65%	8.63%
>0.9	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.58%	3.26%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Table 10.40: F-measure(β=1) of SSM-1 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	39.32%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%
>0.05	39.60%	16.67%	2.86%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%
>0.1	38.94%	39.83%	16.35%	9.50%	2.80%	2.41%	2.41%	2.41%	2.41%	2.41%	2.41%
>0.15	38.11%	40.25%	32.12%	16.23%	11.00%	6.18%	2.80%	2.41%	2.41%	2.41%	2.41%
>0.2	38.82%	40.54%	41.96%	29.94%	16.36%	11.89%	9.31%	5.82%	2.80%	2.41%	2.41%
>0.25	40.48%	41.21%	44.82%	40.46%	23.40%	16.65%	12.11%	10.07%	6.20%	2.84%	2.68%
>0.3	42.97%	40.81%	41.27%	45.07%	36.45%	22.37%	16.73%	12.23%	10.60%	8.80%	4.99%
>0.35	42.73%	41.57%	41.90%	41.47%	42.68%	31.27%	21.34%	15.82%	11.74%	10.27%	8.98%
>0.4	40.42%	39.77%	37.59%	36.39%	37.52%	32.19%	30.33%	21.33%	15.42%	11.72%	11.59%
>0.45	34.01%	34.30%	34.28%	35.27%	32.81%	34.64%	31.22%	23.50%	17.61%	15.47%	11.72%
>0.5	25.08%	27.60%	30.19%	29.06%	31.66%	30.96%	31.14%	29.46%	23.12%	17.69%	15.62%
>0.55	22.95%	19.94%	20.32%	22.30%	25.96%	26.09%	27.36%	28.90%	29.21%	22.35%	17.22%
>0.6	17.21%	18.51%	17.31%	17.30%	14.54%	19.17%	21.60%	23.85%	28.11%	29.69%	22.09%
>0.65	13.61%	14.27%	16.11%	11.11%	9.88%	11.69%	13.81%	21.58%	21.56%	25.54%	23.97%
>0.7	8.64%	8.21%	5.15%	5.90%	9.01%	9.31%	10.21%	12.97%	17.84%	19.29%	25.21%
>0.75	5.18%	0.77%	1.91%	2.86%	5.35%	7.31%	8.99%	9.94%	14.14%	16.91%	15.82%
>0.8	0.77%	0.77%	0.77%	0.77%	0.78%	2.31%	5.37%	7.31%	9.07%	14.37%	15.91%
>0.85	N/A	N/A	N/A	N/A	N/A	N/A	N/A	1.16%	4.65%	6.93%	12.89%
>0.9	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	1.16%	6.23%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 10.41: F-measure(β=2) of SSM-1 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	55.15%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%
>0.05	55.28%	32.25%	6.86%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%
>0.1	53.18%	55.35%	31.76%	20.43%	6.71%	5.82%	5.82%	5.82%	5.82%	5.82%	5.82%
>0.15	47.76%	53.59%	48.41%	31.49%	23.15%	13.98%	6.71%	5.82%	5.82%	5.82%	5.82%
>0.2	44.84%	49.88%	54.65%	47.59%	31.47%	24.58%	20.06%	13.25%	6.71%	5.82%	5.82%
>0.25	42.59%	45.79%	52.92%	54.12%	40.54%	31.82%	24.85%	21.35%	13.99%	6.81%	6.41%
>0.3	41.97%	41.73%	44.62%	52.63%	50.67%	39.05%	31.76%	24.78%	21.94%	18.59%	11.18%
>0.35	39.82%	39.67%	41.48%	43.88%	49.01%	46.00%	36.18%	29.15%	23.13%	20.53%	18.40%
>0.4	35.26%	35.29%	33.67%	33.83%	38.17%	40.14%	43.25%	34.93%	27.71%	22.76%	22.56%
>0.45	27.69%	28.02%	28.64%	30.55%	28.84%	34.08%	37.90%	36.02%	30.23%	27.78%	22.76%
>0.5	19.19%	21.35%	23.78%	23.18%	26.28%	26.35%	31.74%	38.70%	35.40%	30.16%	27.46%
>0.55	16.80%	14.79%	14.93%	16.52%	20.05%	20.71%	23.03%	31.28%	38.27%	34.08%	29.26%
>0.6	12.08%	13.01%	12.27%	12.27%	10.14%	14.17%	16.85%	21.22%	30.21%	39.15%	33.83%
>0.65	9.18%	9.65%	11.02%	7.43%	6.68%	7.98%	9.73%	16.34%	19.01%	30.55%	35.32%
>0.7	5.73%	5.39%	3.31%	3.79%	5.92%	6.23%	6.89%	9.02%	13.43%	17.87%	30.18%
>0.75	3.37%	0.49%	1.21%	1.82%	3.43%	4.74%	5.92%	6.70%	9.93%	13.52%	13.68%

>0.8	0.49%	0.49%	0.49%	0.49%	0.49%	1.46%	3.44%	4.74%	6.02%	10.19%	13.47%
>0.85	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.73%	2.96%	4.50%	9.95%
>0.9	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.73%	4.03%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 10.42: Fallout of SSM-1 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	4.38%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%
>0.05	4.26%	17.43%	85.78%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%
>0.1	3.62%	4.09%	17.96%	33.94%	86.91%	99.54%	99.54%	99.54%	99.54%	99.54%	99.54%
>0.15	2.33%	3.20%	4.21%	17.70%	31.74%	44.62%	86.91%	99.54%	99.54%	99.54%	99.54%
>0.2	1.54%	2.11%	2.72%	6.55%	17.16%	30.52%	33.84%	45.52%	86.91%	99.54%	99.54%
>0.25	0.97%	1.40%	1.98%	2.75%	12.43%	16.80%	29.97%	31.65%	42.68%	85.08%	86.62%
>0.3	0.59%	0.87%	1.28%	1.87%	2.78%	12.69%	16.64%	29.73%	30.85%	32.82%	44.29%
>0.35	0.39%	0.52%	0.76%	1.25%	1.83%	5.45%	12.52%	16.52%	29.56%	30.34%	31.44%
>0.4	0.26%	0.33%	0.44%	0.72%	1.28%	2.17%	5.31%	12.33%	16.45%	29.51%	29.53%
>0.45	0.18%	0.20%	0.27%	0.37%	0.66%	1.33%	2.16%	11.36%	14.10%	16.44%	29.51%
>0.5	0.14%	0.16%	0.17%	0.23%	0.32%	0.61%	1.53%	4.95%	11.39%	14.01%	15.85%
>0.55	0.09%	0.12%	0.12%	0.14%	0.20%	0.29%	0.64%	1.81%	4.98%	11.70%	14.07%
>0.6	0.06%	0.05%	0.09%	0.09%	0.11%	0.18%	0.28%	0.89%	1.84%	5.06%	11.76%
>0.65	0.03%	0.03%	0.03%	0.04%	0.08%	0.10%	0.15%	0.25%	0.91%	4.47%	11.20%
>0.7	0.03%	0.02%	0.01%	0.01%	0.02%	0.06%	0.09%	0.14%	0.29%	1.00%	4.65%
>0.75	0.01%	0.00%	0.01%	0.01%	0.01%	0.01%	0.03%	0.08%	0.15%	0.49%	0.85%
>0.8	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.01%	0.05%	0.17%	0.71%
>0.85	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.02%	0.41%
>0.9	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.02%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we assign more weights to them when determining the optimal threshold values for the SSM-1. Therefore, we choose two groups of optimal β values and optimal threshold values respectively based on the highest F-measure(β=1) and F-measure(β=2) scores.

The performance scores of the SSM-2 in the service concept recommendation simulation in the health service domain on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown from Table 10.43 to Table 10.48.

Table 10.43: Precision of SSM-2 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	26.59%	7.16%	7.16%	7.16%	7.16%	7.16%	7.16%	7.16%	7.16%	7.16%	7.44%

>0.05	26.89%	25.10%	13.20%	12.23%	7.18%	7.18%	7.23%	7.32%	7.61%	8.02%	7.44%
>0.1	26.93%	27.75%	26.00%	19.74%	13.50%	12.59%	12.78%	7.74%	8.02%	7.44%	7.44%
>0.15	28.51%	30.58%	31.05%	30.64%	22.79%	22.05%	15.14%	14.97%	13.45%	13.53%	14.71%
>0.2	31.72%	33.15%	36.29%	36.23%	33.18%	27.67%	23.22%	22.27%	15.83%	15.84%	14.71%
>0.25	37.40%	38.02%	38.50%	38.90%	40.23%	34.47%	27.21%	23.08%	23.07%	23.11%	16.06%
>0.3	44.74%	43.78%	43.49%	41.68%	39.66%	36.80%	34.70%	28.11%	22.47%	20.90%	20.90%
>0.35	48.66%	49.14%	46.53%	46.47%	42.50%	42.21%	36.27%	32.65%	24.78%	25.08%	20.92%
>0.4	53.45%	50.66%	52.32%	48.42%	44.62%	42.07%	42.89%	34.97%	28.67%	26.18%	24.87%
>0.45	54.90%	56.04%	49.33%	49.37%	48.46%	43.71%	41.93%	41.35%	35.06%	32.77%	25.64%
>0.5	51.38%	54.08%	53.81%	57.45%	54.51%	52.90%	47.80%	48.38%	47.37%	41.15%	34.87%
>0.55	58.85%	56.42%	63.36%	62.86%	62.37%	57.38%	56.43%	54.60%	60.98%	56.35%	47.72%
>0.6	58.82%	63.61%	68.40%	68.44%	63.04%	63.70%	64.26%	64.20%	61.61%	62.13%	61.66%
>0.65	69.44%	74.24%	66.67%	61.71%	73.09%	69.47%	70.19%	65.11%	64.20%	63.32%	58.78%
>0.7	56.52%	60.53%	76.92%	73.53%	73.94%	73.34%	69.72%	66.12%	60.20%	57.54%	62.30%
>0.75	50.00%	40.00%	62.50%	87.50%	75.00%	60.91%	62.90%	62.68%	51.55%	51.55%	47.22%
>0.8	50.00%	100%	100%	50.00%	80.00%	77.78%	57.36%	55.44%	48.29%	41.35%	41.35%
>0.85	N/A	N/A	100%	100%	100%	75.00%	85.71%	57.36%	54.58%	41.35%	41.35%
>0.9	N/A	N/A	N/A	N/A	N/A	100%	100%	80.00%	57.36%	48.29%	41.35%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	100%	57.36%	41.35%

Table 10.44: Mean average precision of SSM-2 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	65.93%	61.17%	60.74%	60.26%	58.58%	55.19%	48.70%	45.54%	44.78%	44.75%	42.38%
>0.05	65.97%	65.89%	62.08%	61.55%	58.73%	55.44%	49.17%	47.33%	46.96%	53.94%	42.38%
>0.1	66.67%	68.24%	66.01%	64.01%	62.62%	59.08%	52.10%	48.98%	54.01%	60.89%	42.38%
>0.15	73.74%	70.79%	69.99%	68.17%	65.63%	61.31%	54.63%	56.85%	61.14%	63.11%	45.70%
>0.2	78.66%	76.24%	72.74%	70.50%	69.40%	64.64%	63.05%	63.70%	64.39%	64.65%	45.70%
>0.25	81.64%	80.25%	78.63%	76.10%	73.44%	71.98%	72.17%	71.22%	69.64%	71.55%	45.92%
>0.3	82.74%	82.15%	82.07%	83.33%	78.38%	78.85%	74.19%	72.62%	70.23%	67.34%	55.69%
>0.35	84.61%	83.63%	87.87%	86.27%	85.74%	84.69%	76.81%	73.45%	71.35%	71.23%	55.72%
>0.4	85.16%	88.96%	88.78%	88.77%	88.60%	86.02%	84.00%	79.97%	74.33%	73.69%	64.64%
>0.45	88.72%	88.54%	89.35%	92.50%	89.78%	85.46%	85.70%	81.80%	81.71%	82.07%	64.95%
>0.5	93.75%	91.34%	93.80%	91.89%	90.45%	87.30%	85.41%	85.93%	83.15%	81.19%	72.85%
>0.55	94.57%	96.15%	94.87%	95.15%	90.57%	89.38%	84.79%	85.24%	86.13%	84.79%	74.22%
>0.6	95.59%	95.71%	94.64%	95.83%	95.46%	91.52%	87.21%	84.41%	84.83%	87.00%	81.57%
>0.65	94.44%	96.15%	94.12%	97.06%	95.45%	91.78%	87.13%	83.86%	84.41%	84.83%	79.84%
>0.7	96.43%	95.83%	100%	100%	96.67%	94.44%	89.35%	85.08%	81.17%	81.92%	80.31%
>0.75	91.67%	100%	100%	100%	100%	94.44%	90.17%	86.75%	76.21%	76.21%	69.95%
>0.8	100%	100%	100%	100%	100%	100%	93.75%	90.09%	74.43%	69.87%	62.13%
>0.85	N/A	N/A	100%	100%	100%	100%	100%	93.75%	82.25%	69.87%	62.13%
>0.9	N/A	N/A	N/A	N/A	N/A	100%	100%	100%	93.75%	74.43%	62.13%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	100%	93.75%	62.13%

Table 10.45: Recall of SSM-2 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	75.39%	87.72%	87.72%	87.72%	87.72%	87.72%	87.72%	87.72%	87.72%	87.72%	60.04%
>0.05	75.10%	78.51%	85.77%	85.70%	87.11%	86.34%	84.33%	82.53%	81.58%	69.57%	60.04%
>0.1	70.31%	72.02%	74.53%	80.38%	80.87%	80.58%	80.02%	79.43%	69.57%	60.04%	60.04%
>0.15	57.46%	64.13%	66.08%	71.12%	75.97%	77.29%	76.07%	66.99%	59.89%	57.55%	54.92%
>0.2	50.01%	55.06%	58.53%	63.06%	64.30%	66.59%	63.04%	58.09%	56.19%	55.80%	54.92%
>0.25	44.12%	47.31%	50.31%	53.92%	54.26%	51.47%	49.44%	46.71%	48.03%	47.64%	54.82%
>0.3	41.34%	41.83%	42.75%	40.12%	41.38%	38.96%	41.89%	44.42%	43.40%	43.35%	43.35%
>0.35	38.09%	37.23%	32.04%	32.32%	32.03%	32.04%	33.41%	34.38%	38.08%	40.13%	43.06%
>0.4	32.49%	28.54%	27.64%	24.61%	22.74%	23.68%	25.38%	28.14%	29.31%	32.38%	35.75%
>0.45	24.63%	23.75%	19.15%	17.98%	16.61%	15.86%	17.98%	21.71%	23.67%	25.57%	30.19%
>0.5	16.59%	16.26%	13.93%	13.86%	12.12%	13.09%	14.58%	17.52%	20.67%	22.62%	23.49%
>0.55	14.25%	11.45%	10.59%	10.53%	8.61%	10.32%	12.26%	13.26%	16.66%	18.72%	22.32%
>0.6	10.08%	9.84%	8.18%	5.43%	6.53%	7.81%	9.84%	11.97%	12.65%	14.92%	17.45%
>0.65	7.55%	7.64%	4.09%	4.09%	5.04%	6.35%	7.74%	9.06%	11.82%	12.89%	15.28%
>0.7	4.68%	3.80%	2.26%	3.31%	3.75%	4.48%	6.42%	7.96%	8.23%	10.50%	13.52%
>0.75	2.73%	0.39%	0.93%	1.73%	2.60%	2.85%	3.99%	5.93%	7.32%	7.41%	8.55%
>0.8	0.39%	0.39%	0.19%	0.19%	0.73%	2.02%	2.55%	4.12%	6.18%	6.57%	6.90%
>0.85	0.00%	0.00%	0.19%	0.19%	0.19%	0.58%	1.53%	2.55%	5.03%	6.57%	6.57%
>0.9	0.00%	0.00%	0.00%	0.00%	0.00%	0.19%	0.19%	0.73%	2.55%	6.18%	6.57%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.19%	2.55%	6.57%

Table 10.46: F-measure(β=1) of SSM-2 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	39.32%	13.24%	13.24%	13.24%	13.24%	13.24%	13.24%	13.24%	13.24%	13.24%	13.23%
>0.05	39.60%	38.04%	22.87%	21.41%	13.27%	13.26%	13.31%	13.45%	13.92%	14.37%	13.23%
>0.1	38.94%	40.07%	38.55%	31.70%	23.14%	21.77%	22.04%	14.11%	14.37%	13.23%	13.23%
>0.15	38.11%	41.41%	42.25%	42.83%	35.06%	34.32%	25.25%	24.47%	21.96%	21.91%	23.21%
>0.2	38.82%	41.39%	44.80%	46.02%	43.77%	39.10%	33.93%	32.20%	24.70%	24.67%	23.21%
>0.25	40.48%	42.16%	43.62%	45.20%	46.20%	41.29%	35.10%	30.89%	31.17%	31.13%	24.84%
>0.3	42.97%	42.79%	43.12%	40.89%	40.50%	37.85%	37.96%	34.43%	29.61%	28.20%	28.20%
>0.35	42.73%	42.37%	37.95%	38.12%	36.53%	36.43%	34.78%	33.49%	30.03%	30.87%	28.16%
>0.4	40.42%	36.51%	36.17%	32.63%	30.13%	30.31%	31.89%	31.19%	28.99%	28.95%	29.33%
>0.45	34.01%	33.36%	27.59%	26.36%	24.74%	23.28%	25.17%	28.47%	28.26%	28.73%	27.73%
>0.5	25.08%	25.01%	22.14%	22.33%	19.82%	20.98%	22.34%	25.72%	28.78%	29.19%	28.07%
>0.55	22.95%	19.03%	18.15%	18.04%	15.14%	17.49%	20.14%	21.34%	26.17%	28.10%	30.42%
>0.6	17.21%	17.04%	14.61%	10.06%	11.84%	13.91%	17.07%	20.17%	21.00%	24.06%	27.20%
>0.65	13.61%	13.86%	7.70%	7.67%	9.43%	11.63%	13.94%	15.91%	19.96%	21.42%	24.25%
>0.7	8.64%	7.15%	4.40%	6.33%	7.13%	8.44%	11.76%	14.20%	14.49%	17.76%	22.22%
>0.75	5.18%	0.77%	1.82%	3.39%	5.03%	5.44%	7.50%	10.84%	12.82%	12.95%	14.48%
>0.8	0.77%	0.78%	0.39%	0.39%	1.45%	3.94%	4.89%	7.67%	10.96%	11.34%	11.83%

>0.85	N/A	N/A	0.39%	0.39%	0.39%	1.16%	3.01%	4.89%	9.22%	11.34%	11.34%
>0.9	N/A	N/A	N/A	N/A	N/A	0.39%	0.39%	1.45%	4.89%	10.96%	11.34%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.39%	4.89%	11.34%

Table 10.47: F-measure(β=2) of SSM-2 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	55.15%	26.98%	26.98%	26.98%	26.98%	26.98%	26.98%	26.98%	26.98%	26.98%	24.86%
>0.05	55.28%	55.07%	40.85%	38.93%	27.01%	26.95%	26.91%	27.02%	27.71%	27.43%	24.86%
>0.1	53.18%	54.60%	54.27%	49.79%	40.48%	38.73%	38.99%	27.85%	27.43%	24.86%	24.86%
>0.15	47.76%	52.59%	53.91%	56.26%	51.79%	51.50%	42.15%	39.52%	35.42%	34.86%	35.51%
>0.2	44.84%	48.63%	52.14%	54.93%	54.14%	51.97%	46.94%	43.95%	37.22%	37.09%	35.51%
>0.25	42.59%	45.10%	47.40%	50.06%	50.72%	46.85%	42.49%	38.77%	39.49%	39.30%	36.97%
>0.3	41.97%	42.21%	42.90%	40.43%	41.02%	38.51%	40.22%	39.80%	36.58%	35.68%	35.68%
>0.35	39.82%	39.13%	34.17%	34.41%	33.69%	33.66%	33.94%	34.02%	34.39%	35.83%	35.54%
>0.4	35.26%	31.27%	30.52%	27.30%	25.21%	25.95%	27.63%	29.29%	29.18%	30.92%	32.87%
>0.45	27.69%	26.84%	21.82%	20.60%	19.13%	18.18%	20.30%	23.99%	25.31%	26.75%	29.16%
>0.5	19.19%	18.91%	16.36%	16.34%	14.35%	15.41%	16.93%	20.08%	23.29%	24.86%	25.13%
>0.55	16.80%	13.62%	12.71%	12.63%	10.41%	12.34%	14.53%	15.63%	19.49%	21.60%	24.98%
>0.6	12.08%	11.84%	9.93%	6.65%	7.96%	9.47%	11.85%	14.29%	15.05%	17.59%	20.37%
>0.65	9.18%	9.32%	5.03%	5.03%	6.19%	7.76%	9.41%	10.95%	14.12%	15.33%	17.93%
>0.7	5.73%	4.68%	2.81%	4.09%	4.62%	5.51%	7.84%	9.65%	9.95%	12.56%	16.04%
>0.75	3.37%	0.49%	1.15%	2.15%	3.23%	3.52%	4.91%	7.24%	8.84%	8.94%	10.23%
>0.8	0.49%	0.49%	0.24%	0.24%	0.91%	2.51%	3.16%	5.06%	7.49%	7.90%	8.28%
>0.85	N/A	N/A	0.24%	0.24%	0.24%	0.73%	1.91%	3.16%	6.15%	7.90%	7.90%
>0.9	N/A	N/A	N/A	N/A	N/A	0.24%	0.24%	0.91%	3.16%	7.49%	7.90%
>0.95	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	0.24%	3.16%	7.90%

Table 10.48: Fallout of SSM-2 in the service concept recommendation simulation in the health service domain

Threshold value	β=0	β=0.1	β=0.2	β=0.3	β=0.4	β=0.5	β=0.6	β=0.7	β=0.8	β=0.9	β=1
>0	4.38%	30.25%	30.25%	30.25%	30.25%	30.25%	30.25%	30.25%	30.25%	30.25%	28.56%
>0.05	4.26%	5.38%	15.39%	16.09%	30.01%	29.93%	29.67%	29.13%	28.82%	28.60%	28.56%
>0.1	3.62%	3.63%	4.67%	8.16%	14.31%	14.78%	14.81%	28.70%	28.60%	28.56%	28.56%
>0.15	2.33%	2.50%	2.89%	3.51%	6.11%	7.15%	13.45%	13.40%	14.09%	14.07%	13.86%
>0.2	1.54%	1.76%	1.87%	2.38%	2.99%	3.98%	6.64%	6.71%	13.08%	13.09%	13.66%
>0.25	0.97%	1.09%	1.31%	1.88%	2.02%	2.68%	3.71%	5.28%	6.34%	6.39%	13.06%
>0.3	0.59%	0.71%	0.81%	0.90%	1.52%	1.76%	2.43%	3.53%	5.19%	6.24%	6.24%
>0.35	0.39%	0.43%	0.47%	0.57%	1.08%	1.20%	1.66%	2.30%	3.35%	3.54%	6.07%
>0.4	0.26%	0.29%	0.30%	0.39%	0.43%	0.89%	1.02%	1.45%	2.22%	2.79%	3.32%
>0.45	0.18%	0.19%	0.21%	0.25%	0.31%	0.72%	0.77%	0.87%	1.20%	1.68%	2.70%
>0.5	0.14%	0.14%	0.14%	0.17%	0.22%	0.27%	0.68%	0.72%	0.79%	1.03%	1.59%
>0.55	0.09%	0.09%	0.09%	0.10%	0.14%	0.23%	0.64%	0.65%	0.73%	0.75%	1.01%
>0.6	0.06%	0.05%	0.05%	0.06%	0.09%	0.16%	0.23%	0.62%	0.65%	0.72%	0.72%

>0.65	0.03%	0.02%	0.02%	0.04%	0.05%	0.10%	0.19%	0.62%	0.62%	0.65%	0.72%
>0.7	0.03%	0.02%	0.01%	0.01%	0.03%	0.06%	0.12%	0.22%	0.61%	0.62%	0.65%
>0.75	0.01%	0.01%	0.01%	0.00%	0.01%	0.03%	0.06%	0.16%	0.61%	0.61%	0.65%
>0.8	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.03%	0.08%	0.22%	0.61%	0.61%
>0.85	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.03%	0.12%	0.61%	0.61%
>0.9	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.03%	0.22%	0.61%
>0.95	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.03%	0.61%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we assign more weight to them when determining the optimal threshold values for the SSM-2. Therefore, we choose two groups of optimal β values and optimal threshold values respectively based on the highest F-measure(β=1) and F-measure(β=2) scores.

The performance scores of the Rada model, normalized Resnik model, Lin model and the Jiang and Conath model in the service concept recommendation simulation in the health service domain on precision, mean average precision, recall, F-measure(β=1), F-measure(β=2) and fallout are shown from Table 10.49 to Table 10.54.

Table 10.49: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on precision in the service concept recommendation simulation in the health service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	1.36%	7.44%	7.44%	17.81%
>0.05	1.36%	7.44%	7.44%	19.78%
>0.1	1.36%	7.44%	7.44%	20.25%
>0.15	1.36%	14.71%	10.08%	22.97%
>0.2	2.59%	14.71%	12.59%	23.09%
>0.25	2.59%	16.06%	14.31%	25.60%
>0.3	2.59%	20.90%	18.51%	30.48%
>0.35	7.21%	20.92%	18.79%	35.14%
>0.4	7.21%	24.87%	19.16%	40.82%
>0.45	7.21%	25.64%	19.85%	45.41%
>0.5	13.57%	34.87%	21.53%	54.04%
>0.55	13.57%	47.72%	30.62%	69.35%
>0.6	13.57%	61.66%	30.33%	66.93%
>0.65	13.57%	58.78%	40.06%	66.03%
>0.7	22.50%	62.30%	55.84%	55.88%
>0.75	22.50%	47.22%	76.00%	60.00%
>0.8	22.50%	41.35%	76.92%	57.14%
>0.85	N/A	41.35%	75.00%	0.00%
>0.9	N/A	41.35%	N/A	N/A
>0.95	N/A	41.35%	N/A	N/A

Table 10.50: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on mean average precision in the service concept recommendation simulation in the health service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	27.70%	42.38%	43.14%	82.67%
>0.05	27.70%	42.38%	43.14%	88.02%
>0.1	27.70%	42.38%	43.14%	89.41%
>0.15	27.70%	45.70%	45.79%	90.68%
>0.2	35.82%	45.70%	46.87%	90.68%
>0.25	35.82%	45.92%	47.30%	91.38%
>0.3	35.82%	55.69%	61.55%	91.80%
>0.35	41.15%	55.72%	61.55%	93.45%
>0.4	41.15%	64.64%	69.52%	95.37%
>0.45	41.15%	64.95%	68.89%	97.34%
>0.5	44.00%	72.85%	81.41%	98.53%
>0.55	44.00%	74.22%	83.22%	98.53%
>0.6	44.00%	81.57%	91.96%	99.30%
>0.65	44.00%	79.84%	95.31%	100.00%
>0.7	87.59%	80.31%	98.56%	100.00%
>0.75	87.59%	69.95%	99.30%	100.00%
>0.8	87.59%	62.13%	100.00%	100.00%
>0.85	N/A	62.13%	100.00%	N/A
>0.9	N/A	62.13%	N/A	N/A
>0.95	N/A	62.13%	N/A	N/A

Table 10.51: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on recall in the service concept recommendation simulation in the health service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	89.53%	60.04%	60.04%	24.52%
>0.05	89.53%	60.04%	60.04%	21.01%
>0.1	89.53%	60.04%	60.04%	20.87%
>0.15	89.53%	54.92%	55.85%	19.55%
>0.2	64.73%	54.92%	55.46%	18.05%
>0.25	64.73%	54.82%	54.58%	16.88%
>0.3	64.73%	43.35%	43.13%	14.83%
>0.35	55.70%	43.06%	43.13%	14.25%
>0.4	55.70%	35.75%	36.40%	10.18%
>0.45	55.70%	30.19%	30.12%	9.01%
>0.5	52.41%	23.49%	24.04%	8.23%
>0.55	52.41%	22.32%	23.16%	7.48%
>0.6	52.41%	17.45%	18.38%	6.35%
>0.65	52.41%	15.28%	14.48%	4.01%
>0.7	12.51%	13.52%	12.14%	2.26%

>0.75	12.51%	8.55%	7.35%	1.96%
>0.8	12.51%	6.90%	4.59%	0.46%
>0.85	0.00%	6.57%	3.43%	0.00%
>0.9	0.00%	6.57%	0.00%	0.00%
>0.95	0.00%	6.57%	0.00%	0.00%

Table 10.52: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on F-measure(β=1) in the service concept recommendation simulation in the health service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	2.68%	13.23%	13.23%	20.63%
>0.05	2.68%	13.23%	13.23%	20.38%
>0.1	2.68%	13.23%	13.23%	20.56%
>0.15	2.68%	23.21%	17.08%	21.12%
>0.2	4.98%	23.21%	20.52%	20.26%
>0.25	4.98%	24.84%	22.68%	20.34%
>0.3	4.98%	28.20%	25.90%	19.95%
>0.35	12.76%	28.16%	26.17%	20.27%
>0.4	12.76%	29.33%	25.10%	16.29%
>0.45	12.76%	27.73%	23.93%	15.03%
>0.5	21.55%	28.07%	22.71%	14.28%
>0.55	21.55%	30.42%	26.37%	13.50%
>0.6	21.55%	27.20%	22.89%	11.60%
>0.65	21.55%	24.25%	21.27%	7.56%
>0.7	16.08%	22.22%	19.95%	4.34%
>0.75	16.08%	14.48%	13.41%	3.80%
>0.8	16.08%	11.83%	8.67%	0.91%
>0.85	N/A	11.34%	6.55%	N/A
>0.9	N/A	11.34%	N/A	N/A
>0.95	N/A	11.34%	N/A	N/A

Table 10.53: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on F-measure(β=2) in the service concept recommendation simulation in the health service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	6.41%	24.86%	24.86%	22.80%
>0.05	6.41%	24.86%	24.86%	20.75%
>0.1	6.41%	24.86%	24.86%	20.74%
>0.15	6.41%	35.51%	29.27%	20.15%
>0.2	11.16%	35.51%	32.99%	18.87%
>0.25	11.16%	36.97%	34.92%	18.11%
>0.3	11.16%	35.68%	34.07%	16.53%
>0.35	23.74%	35.54%	34.25%	16.17%
>0.4	23.74%	32.87%	30.85%	11.98%

>0.45	23.74%	29.16%	27.29%	10.73%
>0.5	33.33%	25.13%	23.49%	9.91%
>0.55	33.33%	24.98%	24.34%	9.10%
>0.6	33.33%	20.37%	19.95%	7.75%
>0.65	33.33%	17.93%	16.60%	4.94%
>0.7	13.73%	16.04%	14.39%	2.79%
>0.75	13.73%	10.23%	8.97%	2.43%
>0.8	13.73%	8.28%	5.66%	0.57%
>0.85	N/A	7.90%	4.23%	N/A
>0.9	N/A	7.90%	N/A	N/A
>0.95	N/A	7.90%	N/A	N/A

Table 10.54: Performance of Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on fallout in the service concept recommendation simulation in the health service domain

Threshold value	Rada	Normalized Resnik	Lin	Jiang and Conath
>0	86.62%	28.56%	28.56%	1.69%
>0.05	86.62%	28.56%	28.56%	1.17%
>0.1	86.62%	28.56%	28.56%	1.08%
>0.15	86.62%	13.86%	15.50%	0.80%
>0.2	44.40%	13.66%	13.06%	0.77%
>0.25	44.40%	13.06%	12.46%	0.59%
>0.3	44.40%	6.24%	5.89%	0.45%
>0.35	17.86%	6.07%	5.86%	0.36%
>0.4	17.86%	3.32%	3.41%	0.24%
>0.45	17.86%	2.70%	2.62%	0.18%
>0.5	11.89%	1.59%	1.47%	0.10%
>0.55	11.89%	1.01%	0.63%	0.05%
>0.6	11.89%	0.72%	0.40%	0.04%
>0.65	11.89%	0.72%	0.22%	0.03%
>0.7	0.74%	0.65%	0.09%	0.02%
>0.75	0.74%	0.65%	0.03%	0.02%
>0.8	0.74%	0.61%	0.02%	0.01%
>0.85	0.00%	0.61%	0.01%	0.00%
>0.9	0.00%	0.61%	0.00%	0.00%
>0.95	0.00%	0.61%	0.00%	0.00%
>0.95	0.00%	0.50%	0.00%	0.00%

Since F-measure(β=1) and F-measure(β=2) are two aggregated performance indicators, we assign more weight to them when determining the optimal threshold values for the four existing models. Therefore, we choose four groups of optimal threshold values respectively based on the highest F-measure(β=1) and F-measure(β=2) scores.

Table 10.55 shows the comparison of the SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=1) in the service concept recommendation simulation in the health service domain. It can be seen that the SSM-2 has the highest score on F-measure(β=1), followed by the SSM-1 which is more than 15% higher than the existing semantic similarity models. This is because the SSM-2 and SSM-1 have the highest scores on recall.

Table 10.55: Comparison of SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=1) in the service concept recommendation simulation in the health service domain

Algorithm	Optimal threshold value	Precision	Mean Average Precision	Recall	Fallout	F-measure(β=1)
SSM-1(β=0.3)	>0.3	36.36%	69.33%	59.26%	1.87%	45.07%
SSM-2(β=0.4)	>0.25	40.23%	73.44%	54.26%	2.02%	46.20%
Rada	>0.5	13.57%	44.00%	52.41%	11.89%	21.55%
Normalized Resnik	>0.55	47.72%	74.22%	22.32%	1.01%	30.42%
Lin	>0.55	30.62%	83.22%	23.16%	0.63%	26.37%
Jiang and Conath	>0.15	22.97%	90.68%	19.55%	0.80%	21.12%

Table 10.56 shows the comparison of the SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=2) in the service concept recommendation simulation in the health service domain. Because of the distinct advantages of the SSM-1 and SSM-2 on recall, the SSM-1 and SSM-2 are nearly 20% higher than the existing semantic similarity models on F-measure(β=2).

Table 10.56: Comparison of SSM-1, SSM-2, Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model on their optimal threshold values for F-measure(β=2) in the service concept recommendation simulation in the health service domain

Algorithm	Optimal threshold value	Precision	Mean Average Precision	Recall	Fallout	F-measure (β =2)
SSM-1(β=0.1)	>0.1	27.14%	66.82%	74.78%	4.09%	55.35%
SSM-2(β=0.3)	>0.15	30.64%	68.17%	71.12%	3.51%	56.26%
Rada	>0.5	13.57%	44.00%	52.41%	11.89%	33.33%
Normalized Resnik	>0.25	16.06%	45.92%	54.82%	13.06%	36.97%
Lin	>0.25	14.31%	47.30%	54.58%	12.46%	34.92%
Jiang and Conath	>0	17.81%	82.67%	24.52%	1.69%	22.80%

10.5.6 Conclusion

As a conclusion, by comparing the optimal performance between our proposed semantic similarity models – SSM-1 and SSM-2, and the existing semantic similarity models, our proposed models show huge improvements in terms of those two aggregated performance indicators – F-measure(β=1) and F-measure(β=2) in the service concept recommendation simulation in both the transport and health service domains. Hence, this experiment preliminarily proves the higher adaptability of the SSM-1 and SSM-2 model in the ontology environment.

10.6 QoS Evaluation and Service Ranking Functional Testing

In this section, in order to validate the QoS evaluation and service ranking methodology proposed in Chapter 8, we build a QoS Evaluation and Service Ranking System, and test its functions by means of a use case.

10.6.1 QoS Database

As stated in Chapter 8, the core of the QoS Evaluation and Service Ranking System is a QoS Database, which is displayed in Fig. 10.25. The tables in the QoS Database can be classified into three primary categories as follows [7]:

- Evaluation Criterion Table is designed to store the evaluation criteria for each service concept in the Service Ontology Base. Therefore, each service concept has its own particular QoS evaluation criteria. There is only one evaluation criterion table in which each row corresponds to a QoS evaluation criterion of a service concept.

- Service Provider Reputation Table is designed to store the service provider's performance against each QoS evaluation criterion of a service concept (*ActualBehaviourCriterion*) and service provider's *Reputation* values for all SDE metadata associated with a service concept. Apart from the SDE metadata QoS evaluation, this table also functions as the data source for the QoS-based SDE metadata ranking. Each service concept corresponds to a table in which each row corresponds to SDE metadata that are associated with the concept. In other words, if a service provider's service entities belong to a number of service concepts, there should be the same number of Service Provider Reputation Tables corresponding to the service provider. Furthermore, the contents of the QoS evaluation criteria (column title, e.g., "Criterion_1_1" in Fig. 10.25) are obtained from the Evaluation Criterion Table.

- Service Requester Evaluation Table is designed to store a service requester's evaluation values to the SDE metadata linked by a service concept. Each Service Requester Evaluation Table corresponds to a Service Provider Evaluation Table. Each row of this table contains a service requester's evaluation value for each QoS evaluation criterion of a service (*ABCorrCriterion*) and the service requester's *Trustworthiness* value for this service.

In addition, there are some other available tables such as a Service Requester Table that is used to store the basic information about service requesters, and a Service Provider Table that is used to store the basic information about service providers [7].

By means of the QoS Database (Fig. 10.25), each service requester's evaluation values can be stored as a new row in the corresponding service requester evaluation table by obtaining the $ABCorr_{Criterion}$ values and calculating the *Trustworthiness* value. Subsequently the service providers' *Reputation* value and $ActualBehaviour_{Criterion}$ values in the corresponding service provider reputation table can be recomputed and updated immediately, to complete the whole QoS evaluation and service ranking process [7].

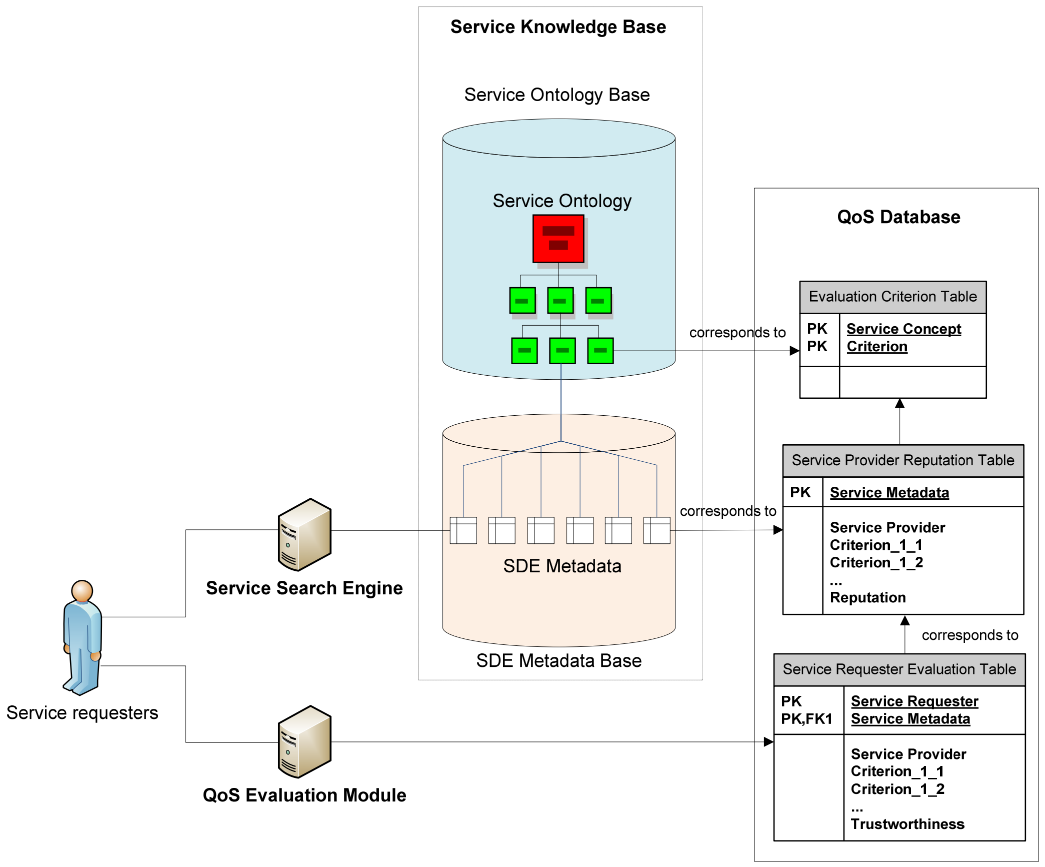
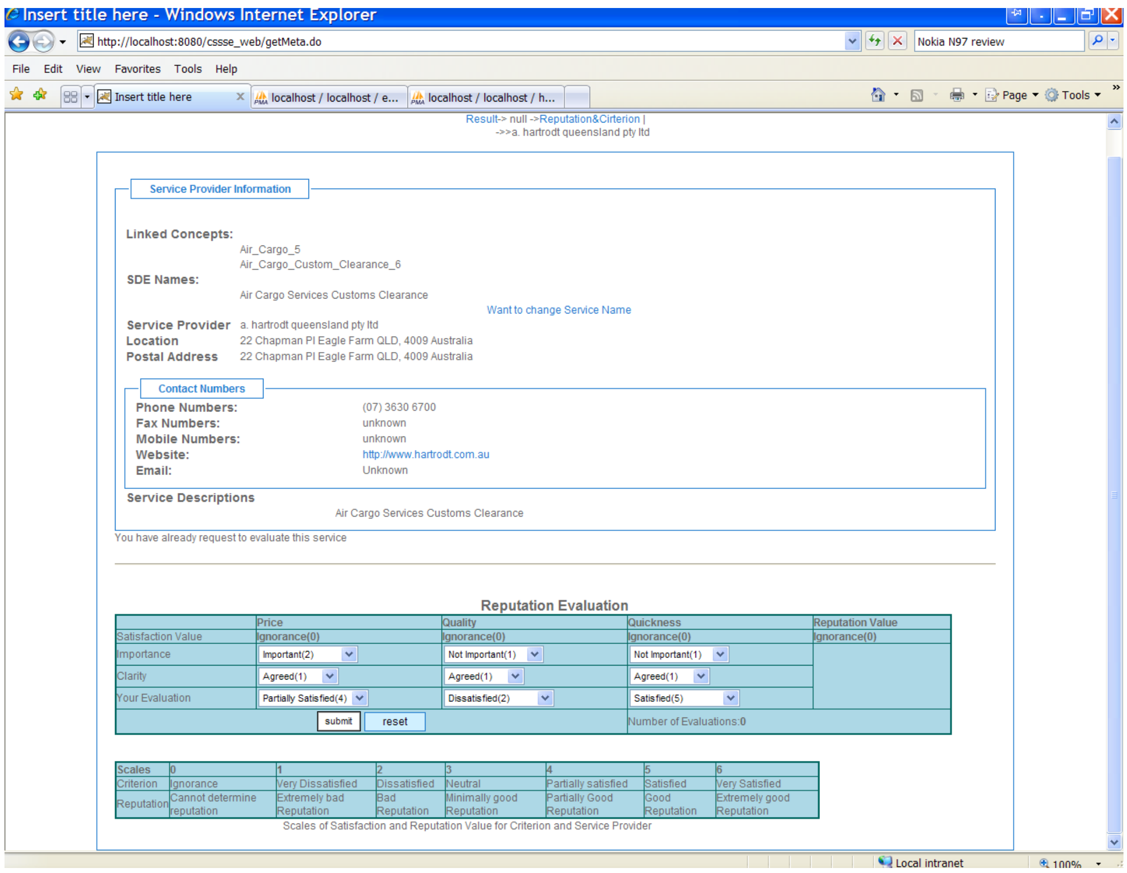


Figure 10.25: System architecture of the QoS Database

10.6.2 A Use Case for Functional Testing

In this section, we utilise a use case to examine the functions of the QoS evaluation and service ranking methodology.

A service requester retrieves a service provider ("*a. hartrodt Queensland pty ltd*" in Fig. 10.26) who can provide an air cargo service in South Australia by means of the CSSSE search engine. After the service requester completes an air cargo service transaction with the service provider, s\he wants to evaluate the quality of this service. Therefore, the service requester needs to obtain the evaluation permission from the system administrator by sending the administrator an inquiry email. The administrator will forward the email to the service provider to ask for confirmation. If the provider confirms the transaction, the service requester will obtain the permission immediately. Once the requester logs into the system and find the metadata under the corresponding service concept, s/he will find an available reputation evaluation form under the metadata information. The service requester needs to assign values to the metrics of $ABCorr_{Criterion}$ (labelled as "*Your evaluation*" in Fig. 10.26), $Clear_{Criterion}$ (labelled as "*Clarity*"), and $Imp_{Criterion}$ (labelled as "*Importance*") for each criterion (price, quality, quickness in this case) involved in the service interaction. Once the form has been submitted, the *Trustworthiness* value that the service requester has given to the service provided by the service provider will be shown in a dialog box. Then the $ActualBehaviour_{Criterion}$ value of each criterion and the *Reputation* value of the service provider will be updated in the QoS Database. It is important to note that a service requester will have only one opportunity to evaluate the service provided by a service provider after the service transaction [7].



Result-> null ->Reputation&Cirterion |
->>a. hartrodt queensland pty ltd

Service Provider Information

Linked Concepts:
Air_Cargo_5
Air_Cargo_Custom_Clearance_6

SDE Names:
Air Cargo Services Customs Clearance
Want to change Service Name

Service Provider	a. hartrodt queensland pty ltd
Location	22 Chapman Pl Eagle Farm QLD, 4009 Australia
Postal Address	22 Chapman Pl Eagle Farm QLD, 4009 Australia

Contact Numbers

Phone Numbers:	(07) 3630 6700
Fax Numbers:	unknown
Mobile Numbers:	unknown
Website:	http://www.hartrodt.com.au
Email:	Unknown

Service Descriptions
Air Cargo Services Customs Clearance

You have already request to evaluate this service

Reputation Evaluation

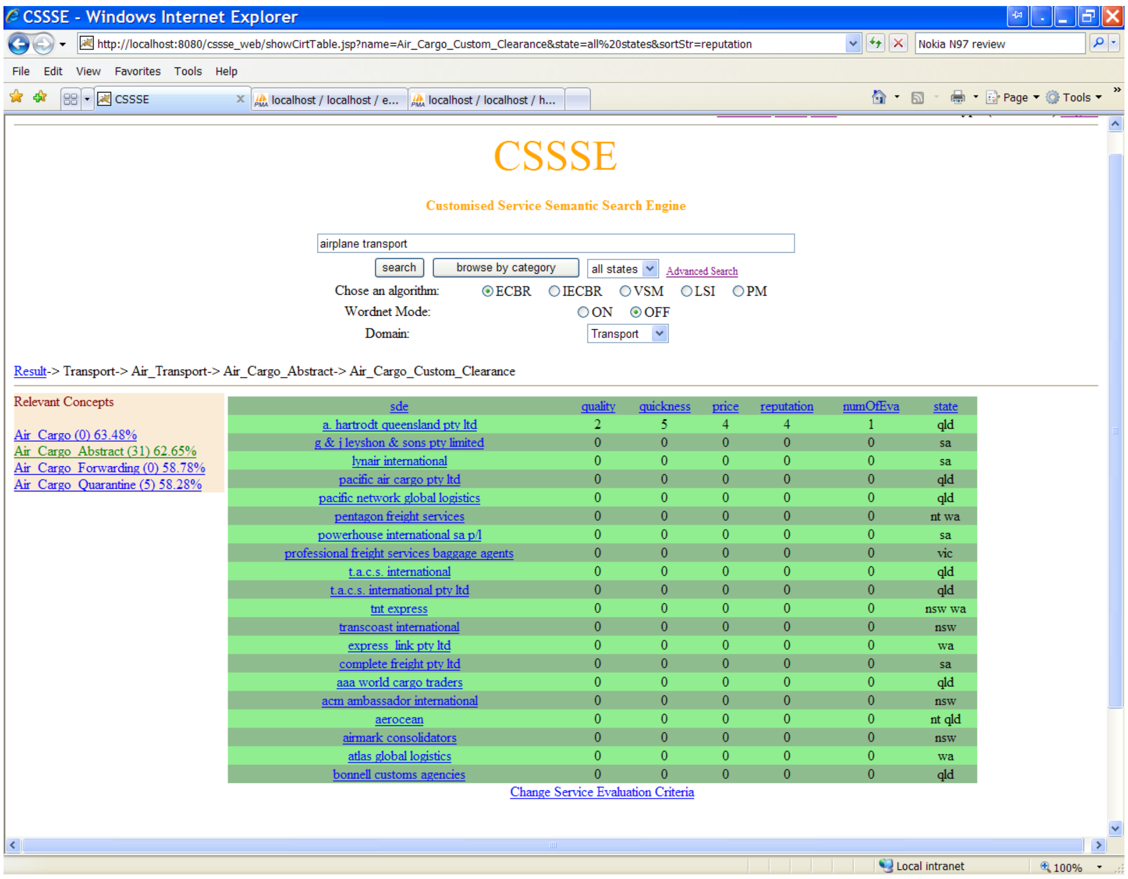
	Price	Quality	Quickness	Reputation Value
Satisfaction Value	Ignorance(0)	Ignorance(0)	Ignorance(0)	Ignorance(0)
Importance	Important(2)	Not Important(1)	Not Important(1)	
Clarity	Agreed(1)	Agreed(1)	Agreed(1)	
Your Evaluation	Partially Satisfied(4)	Dissatisfied(2)	Satisfied(5)	
	submit reset		Number of Evaluations:0	

Scales	0	1	2	3	4	5	6
Criterion	Ignorance	Very Dissatisfied	Dissatisfied	Neutral	Partially satisfied	Satisfied	Very Satisfied
Reputation	Cannot determine reputation	Extremely bad Reputation	Bad Reputation	Minimally good Reputation	Partially Good Reputation	Good Reputation	Extremely good Reputation

Scales of Satisfaction and Reputation Value for Criterion and Service Provider

Figure 10.26: Screenshot of a QoS evaluation window in the QoS Evaluation and Service Ranking System

Accordingly, the next time that any service requester searches for service providers who can provide an air cargo service, all the SDE metadata associated with the "*air cargo*" concept are retrieved from the SDE Metadata Base. These SDE metadata are denoted by their service providers' names (usually a company's name, e.g. "*lynair international*"), in order to enable the service requester to easily distinguish between these service metadata. These SDE metadata can be ranked in a multi-linear manner, including their $ActualBehaviour_{Criterion}$ values on each criterion (all the values under the label "*quality*", "*quickness*", and "*price*" in Fig. 10.27), their *Reputation* values (all the values under the label "*reputation*"), number of evaluations (all the values under the label "*NumOfEva*"), and states where the service providers are located (e.g. "*SA*") [7].


CSSSE - Windows Internet Explorer
http://localhost:8080/cssse_web/showCirtTable.jsp?name=Air_Cargo_Custom_Clearance&state=all%20states&sortStr=reputation
File Edit View Favorites Tools Help
CSSSE
localhost / localhost / e...
localhost / localhost / h...
CSSSE
Customised Service Semantic Search Engine
airplane transport
search
browse by category
all states
Advanced Search
Local intranet
100%

Chose an algorithm: [✓ ECBR] IECBR VSM LSI PM

Wordnet Mode: ON [✓ OFF]

Domain: Transport

Result-> Transport-> Air_Transport-> Air_Cargo_Abstract-> Air_Cargo_Custom_Clearance

Relevant Concepts

Air_Cargo (0) 63.48%
Air_Cargo_Abstract (31) 62.65%
Air_Cargo_Forwarding (0) 58.78%
Air_Cargo_Quarantine (5) 58.28%

sde	quality	quickness	price	reputation	numOfEva	state
a. hartrodt queensland pty ltd	2	5	4	4	1	qld
g & j leyshon & sons pty limited	0	0	0	0	0	sa
lynair international	0	0	0	0	0	sa
pacific air cargo pty ltd	0	0	0	0	0	qld
pacific network global logistics	0	0	0	0	0	qld
pentagon freight services	0	0	0	0	0	nt wa
powerhouse international sa p/l	0	0	0	0	0	sa
professional freight services baggage agents	0	0	0	0	0	vic
t.a.c.s. international	0	0	0	0	0	qld
t.a.c.s. international pty ltd	0	0	0	0	0	qld
tnt express	0	0	0	0	0	nsw wa
transcoast international	0	0	0	0	0	nsw
express link pty ltd	0	0	0	0	0	wa
complete freight pty ltd	0	0	0	0	0	sa
aaa world cargo traders	0	0	0	0	0	qld
acm ambassador international	0	0	0	0	0	nsw
aerocean	0	0	0	0	0	nt qld
airmark consolidators	0	0	0	0	0	nsw
atlas global logistics	0	0	0	0	0	wa
bonnell customs agencies	0	0	0	0	0	qld

Change Service Evaluation Criteria

Figure 10.27: Screenshot of the updated SDE metadata ranking after evaluation

Therefore, by means of this use case, we examined the functions of the QoS evaluation and service ranking methodology in evaluating and ranking SDE metadata based on concept-based QoS evaluation criteria. The experimental result preliminarily proves the feasibility of the methodology in the real world.

10.7 Service Domain Knowledge Updating and Service-Provider-based SDE Metadata Publishing, Maintenance and Classification Functional Testing

In this section, we have the following two validation tasks:

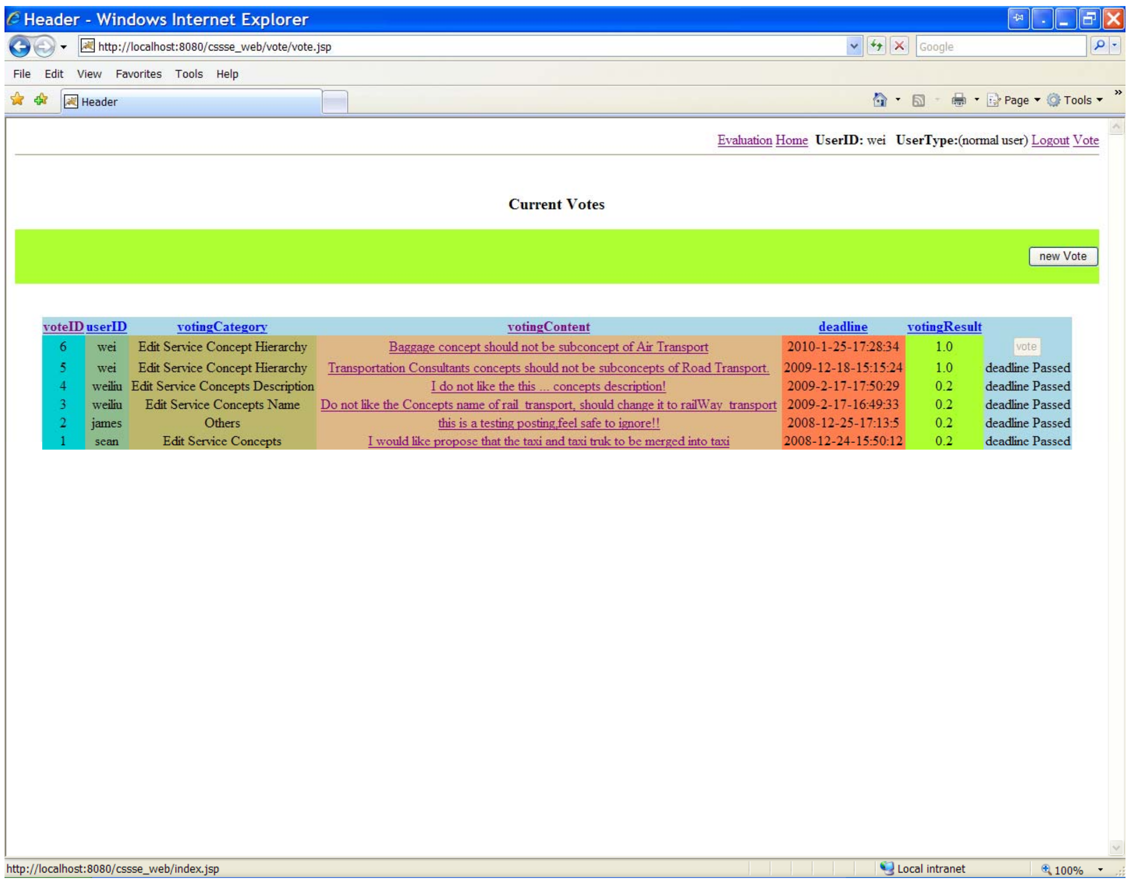
1. We need to validate the service domain knowledge updating methodology by means of the functional testing approach, which is introduced in Section 10.7.1.

2. We need to validate the service-provider-based SDE metadata publishing, maintenance and classification methodology by means of the functional testing approach, which is introduced in Section 10.7.2.

10.7.1 Service Domain Knowledge Updating Functional Testing

In order to validate the methodology for updating service domain knowledge in terms of the functional testing approach, we built a prototype of the methodology – the Service Domain Knowledge Updating System, which can be regarded as a platform for gaining the ideas about knowledge evolution from service communities.

As stated in Chapter 9, the essence of this updating methodology is a voting platform, and all users of the CSSSE system have the right to initiate a voting process or to vote. The outcome of a voting can be: 1) change of the hierarchy of service concepts; 2) change of the name of a service concept; 3) change of the *conceptDescription* property(s) of a service concept; 4) change of a QoS evaluation criterion of a service concept; and 5) other changes in the CSSSE system. The screenshot of the Service Domain Knowledge Updating System can be seen in Fig. 10.28 [11].

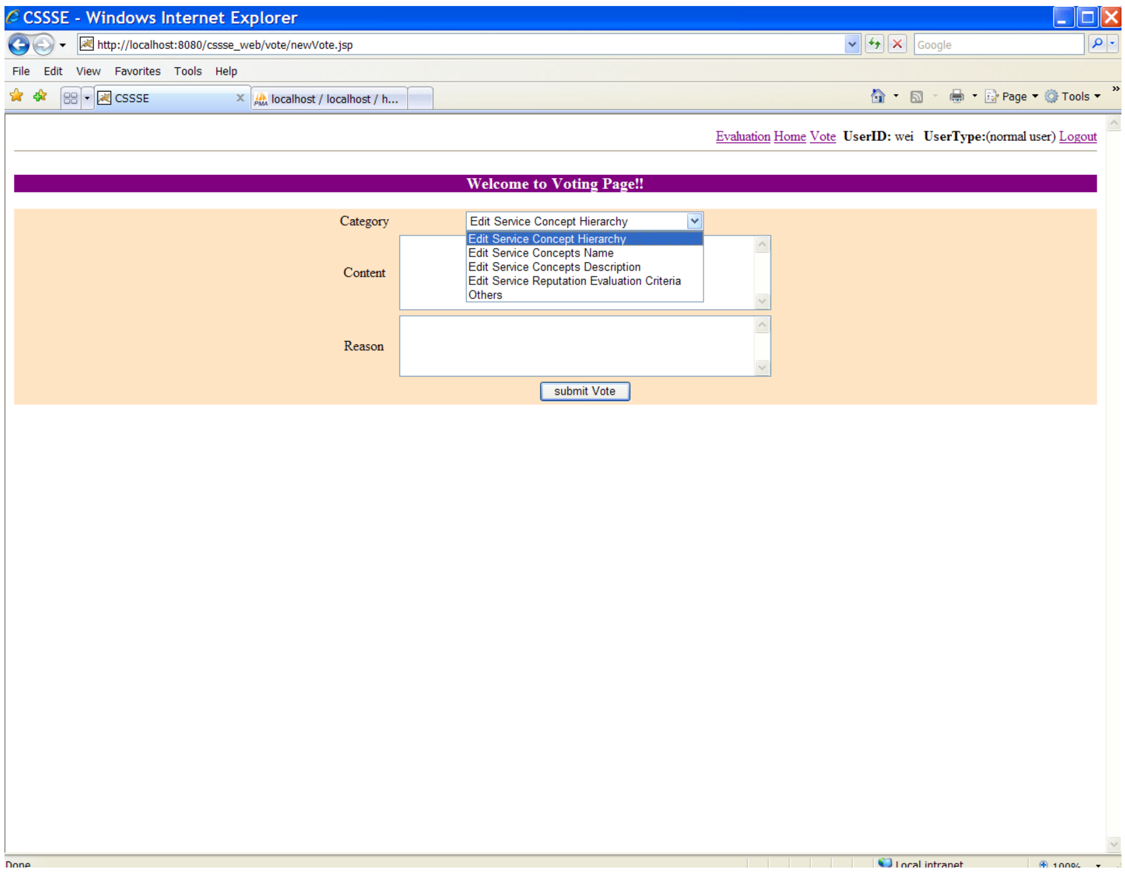

Header - Windows Internet Explorer
http://localhost:8080/cssse_web/vote/vote.jsp
File Edit View Favorites Tools Help
Header
Page Tools
Evaluation Home
UserID: wei UserType:(normal user) Logout Vote
new Vote
vote
http://localhost:8080/cssse_web/index.jsp
Local intranet
100%

Current Votes

voteID	userID	votingCategory	votingContent	deadline	votingResult	
6	wei	Edit Service Concept Hierarchy	Baggage concept should not be subconcept of Air Transport	2010-1-25-17:28:34	1.0	vote
5	wei	Edit Service Concept Hierarchy	Transportation Consultants concepts should not be subconcepts of Road Transport.	2009-12-18-15:15:24	1.0	deadline Passed
4	weiliu	Edit Service Concepts Description	I do not like the this ... concepts description!	2009-2-17-17:50:29	0.2	deadline Passed
3	weiliu	Edit Service Concepts Name	Do not like the Concepts name of rail transport, should change it to railWay transport	2009-2-17-16:49:33	0.2	deadline Passed
2	james	Others	this is a testing posting,feel safe to ignore!!	2008-12-25-17:13:5	0.2	deadline Passed
1	sean	Edit Service Concepts	I would like propose that the taxi and taxi truk to be merged into taxi	2008-12-24-15:50:12	0.2	deadline Passed

Figure 10.28: Screenshot of the Service Domain Knowledge Updating System

If a user desires to request a change to the service knowledge, s/he needs to fill an online knowledge updating request form (Fig. 10.29), in order to describe the contents and reasons for the change. Once the form has been submitted to the system, it will be verified by the system administrators. Once the change request has been approved by the system administrators, the voting system will generate a new voting process, and automatically assign a deadline (normally one month) to the voting [11].


CSSSE - Windows Internet Explorer

http://localhost:8080/cssse_web/vote/newVote.jsp

Evaluation Home Vote **UserID:** wei **UserType:**(normal user) Logout

Welcome to Voting Page!!

Category	Edit Service Concept Hierarchy
Content	
Reason	

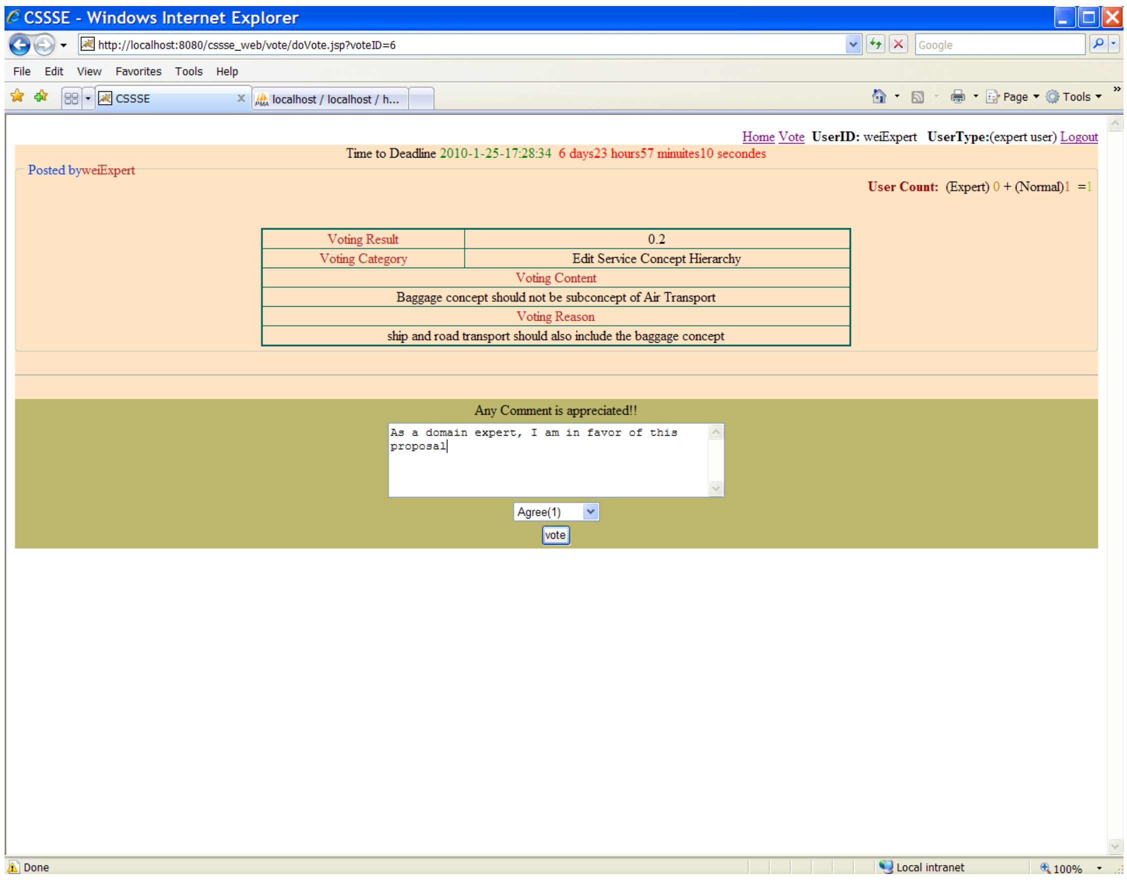
Edit Service Concept Hierarchy
Edit Service Concepts Name
Edit Service Concepts Description
Edit Service Reputation Evaluation Criteria
Others

File Edit View Favorites Tools Help
CSSSE
localhost / localhost / h...
Page Tools
submit Vote
Done
Local intranet
100%

Figure 10.29: Screenshot of the online knowledge updating request form

When other users log in to the voting interface and want to submit a vote, they can press the "*vote*" button behind the new voting, and the system will display a voting form to them. Therefore, they can choose from the "*agree*", "*neutral*" or "*disagree*" options and present their reasons for their choices. Once the voting form has been submitted, the voting system will make use of the algorithm introduced in Chapter 9 to calculate the score of the votes – $Value_{voting}$ based on the classes of the users. It is important to note that each user can vote only once. Users can review the voting history, including the voting scores and comments of the voted users for a voting, by clicking the hyperlink under the content of each voting (see hyperlinks under the "*votingContent*" in Fig. 10.28). Once the deadline has been reached, the system administrator will determine whether or not the change will be implemented based on the final voting score. The screenshot of the voting form can be found in Fig. 10.30 [11].

In conclusion, by means of the screenshots and the functions introduced above, the Service Domain Knowledge Updating System realizes the function of community-based service domain knowledge evolution. Therefore, it can be concluded that the service domain knowledge updating methodology is preliminarily validated by the functional testing process.


CSSSE - Windows Internet Explorer
http://localhost:8080/cssse_web/vote/doVote.jsp?voteID=6
File Edit View Favorites Tools Help
CSSSE
localhost / localhost / h...
Page
Tools
Done
Local intranet
100%

Home Vote **UserID:** weiExpert **UserType:**(expert user) Logout

Time to Deadline 2010-1-25-17:28:34 6 days23 hours57 minuites10 secondes

Posted byweiExpert

User Count: (Expert) 0 + (Normal)1 =1

Voting Result	0.2
Voting Category	Edit Service Concept Hierarchy
Voting Content	
Baggage concept should not be subconcept of Air Transport	
Voting Reason	
ship and road transport should also include the baggage concept	

Any Comment is appreciated!!

```
As a domain expert, I am in favor of this
proposal
```

Agree(1)

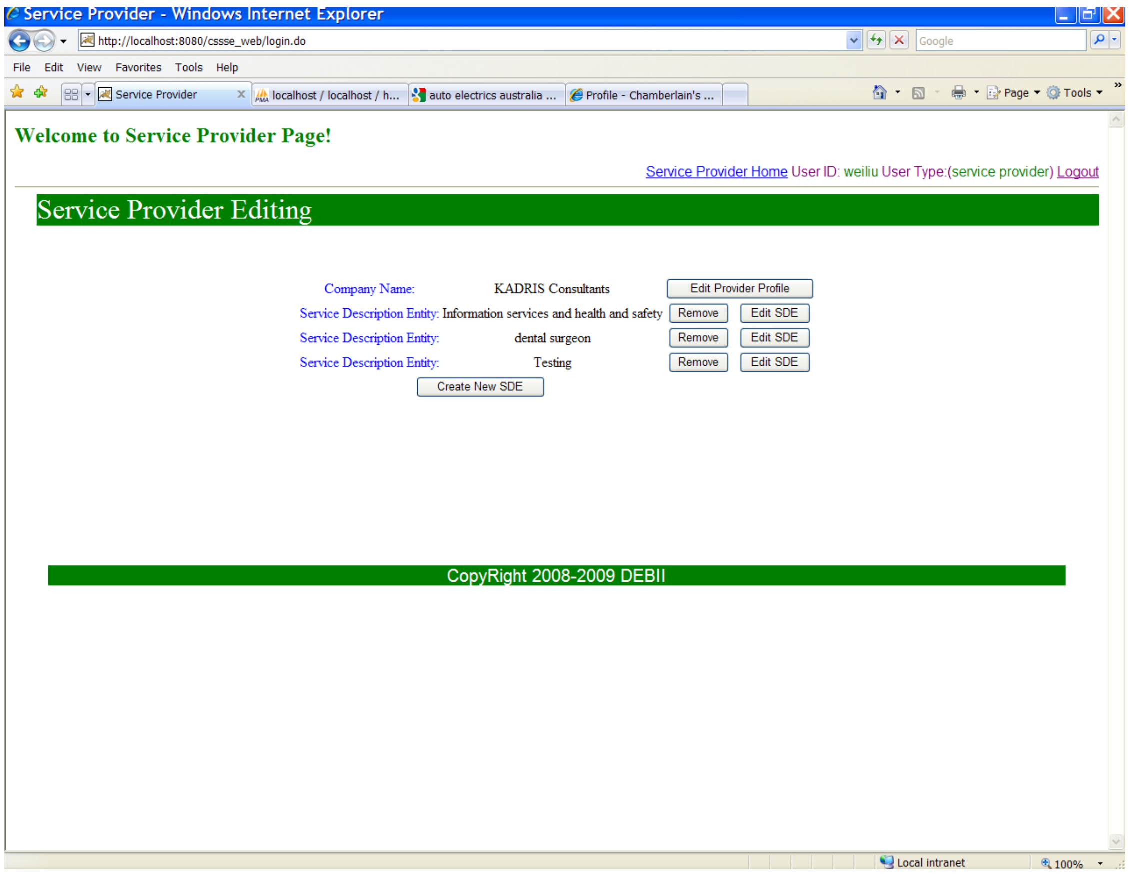
vote

Figure 10.30: Screenshot of a voting form

10.7.2 Service-Provider-based SDE Metadata Publishing, Maintenance and Classification Functional Testing

We built a prototype of the service-provider-based SDE metadata publishing, maintenance and classification methodology – the Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System, with the purpose of validating the methodology in terms of the functional testing approach.

Fig. 10.31 provides a screenshot of the system. Once a service provider logs into the system, first of all, the system will display the information about the service provider's name (usually a company name) and all SDE metadata published by the service provider. According to the designed functions of the system, a service provider can: 1) publish a new SDE metadata by clicking the "*Create New SDE button*" in Fig. 10.31; 2) edit or remove an existing SDE metadata by clicking the "*Edit SDE*" or "*Remove*" button; and 3) edit the service provider's profile (properties of service provide metadata) by clicking the "*Edit Provider Profile*" button.


Welcome to Service Provider Page!

Service Provider Home User ID: weiliu User Type:(service provider) Logout

Service Provider Editing

Company Name:	KADRIS Consultants	Edit Provider Profile	
Service Description Entity:	Information services and health and safety	Remove	Edit SDE
Service Description Entity:	dental surgeon	Remove	Edit SDE
Service Description Entity:	Testing	Remove	Edit SDE

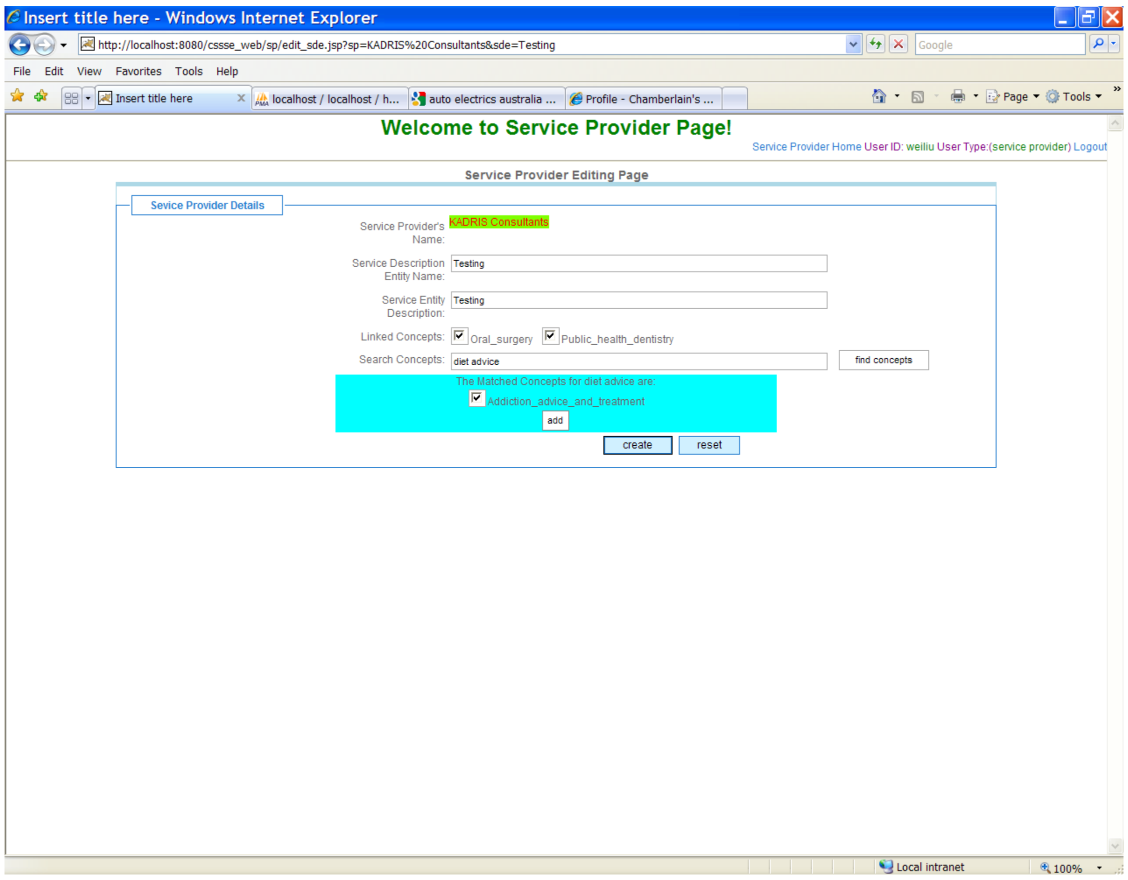
Create New SDE

CopyRight 2008-2009 DEBII

Figure 10.31: Screenshot of the Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System

Fig. 10.32 displays a screenshot of the SDE metadata editing interface. This interface allows a service provider to edit the properties of one of his/her SDE metadata, including the *linkedConcepts* property. In order to facilitate the change of the associated concepts to the SDE metadata, we employ an IECBR-based service search engine to assist the service provider to look for the new concept(s) that s/he wants to associate to the SDE metadata from the service ontologies. By selecting the retrieved concept(s) from the search engine, the function of service-provider-based SDE metadata classification is realized.

In conclusion, in this section, we built a Service-Provider-based SDE Metadata Publishing, Maintenance and Classification System. It is found that the prototype is able to assist service providers to publish new SDE metadata, to edit existing SDE metadata, to edit service provider metadata and to decide the classification of SDE metadata on their own wills. Consequently, by means of the functional testing approach, we preliminarily validate the feasibility of the service-provider-based SDE metadata publishing, maintenance and classification methodology in the real environment.


Insert title here - Windows Internet Explorer
File Edit View Favorites Tools Help
Insert title here
localhost / localhost / h...
auto electrics australia ...
Profile - Chamberlain's ...
Page
Tools
Service Provider Home
Logout
Sevice Provider Details
find concepts
add
create
reset
Local intranet
100%

http://localhost:8080/cssse_web/sp/edit_sde.jsp?sp=KADRIS%20Consultants&sde=Testing

Welcome to Service Provider Page!

User ID: weiliu User Type:(service provider)

Service Provider Editing Page

Service Provider's Name: KADRIS Consultants

Service Description Entity Name: Testing

Service Entity Description: Testing

Linked Concepts: ✓ Oral_surgery ✓ Public_health_dentistry

Search Concepts: diet advice

The Matched Concepts for diet advice are:

✓ Addiction_advice_and_treatment

Figure 10.32: Screenshot of the SDE metadata editing interface

10.8 Conclusion

In this chapter, we validated all parts of the customized semantic service retrieval methodology in terms of the simulation or functional testing approach. The validation steps were implemented as follows:

First of all, we built a prototype of the methodology – a CSSSE system in the transport service domain and the health service domain.

For the service information discovery annotation and classification methodology, first of all, we validated the service information discovery and annotation methodology using the functional testing approach. By checking the quality of generated SDE metadata from the prototype of the methodology – a Semantic Crawler, we validated the feasibility of this methodology. Next, we simulated the service information classification methodology, by simulating the ECBR and IECBR algorithms respectively in the transport service domain and the health service domain and comparing their performance with a LSI algorithm on eight performance indicators. The test results show the advancement of the methodology compared with the existing classification algorithm.

For the service retrieval methodology, first of all, we simulated the ECBR and IECBR algorithms by comparing these with three existing information retrieval algorithms – VSM, LSI and PM, and an existing search engine – the Australian Yellowpages® search engine, The comparison result shows the advantages of the ECBR and IECBR in the aggregated performance indicators and the IECBR is more efficient than the ECBR. Secondly, we ran the prototypes of the Generic Service Retrieval Module and the Specific Service Retrieval Module to test their functions. This functional testing proves their validity.

For the service concept recommendation methodology, we compared these two proposed models with four existing semantic similarity models – Rada's model, normalized Resnik's model, Lin's model and Jiang and Conath's model. This comparison preliminarily proves the higher adaptability of the SSM-1 and SSM-2 model in the ontology environment.

Next, by means of the functional testing approach, we examined the functions of the QoS evaluation and service ranking methodology in evaluating and ranking SDE metadata based on concept-based QoS evaluation criteria. The experimental result preliminarily proves the feasibility of the methodology in the real world.

Last but not least, we validated the service domain knowledge updating methodology and the service-provider-based SDE metadata publishing, maintenance and classification methodology by means of the functional testing approach.

In the next chapter, we provide a conclusion to the work undertaken in the thesis and present our vision and direction for future work.

10.9 References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley, New York (1999)

2. Deerwester, S.C., Dumais, S.T., Furnas, G.W., Harshman, R.A., Landauer, T.K., Lochbaum, K.E., Streeter, L.A.: Computer information retrieval using latent semantic structure. In: Patent, U.S. (ed.), Vol. 4839853. Bell Communications Research, Inc., USA (1988)

3. Dong, H., Hussain, F.K., Chang, E.: Semantic service matchmaking for digital health ecosystems. Knowledge-Based Systems **Submitted**

4. Dong, H., Hussain, F.K., Chang, E.: A context-aware semantic similarity model for ontology environments. Concurrency and Computation: Practice and Experience **In Press**

5. Dong, H., Hussain, F.K., Chang, E.: A service concept recommendation system for enhancing the dependability of semantic service matchmakers in the service ecosystem environment. Journal of Network and Computer Applications **Submitted**

6. Dong, H., Hussain, F.K., Chang, E.: Transport service ontology and its application in the field of semantic search. Proceedings of the 2008 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI 2008). IEEE, Beijing, China (2008) 820-824

7. Dong, H., Hussain, F.K., Chang, E.: A QoS-based service retrieval methodology for digital ecosystems. International Journal of Web and Grid Services **5** (2009) 261-283

8. Dong, H., Hussain, F.K., Chang, E.: Focused crawling for automatic service discovery, annotation and classification in industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

9. Dong, H., Hussain, F.K., Chang, E.: A framework for discovering and classifying ubiquitous services in digital health ecosystems. Journal of Computer and System Sciences **In Press**

10. Dong, H., Hussain, F.K., Chang, E.: A service search engine for the industrial digital ecosystems. IEEE Transactions on Industrial Electronics **In Press**

11. Dong, H., Hussain, F.K., Chang, E.: A human-centered semantic service platform for the digital ecosystems environment. World Wide Web **13** (2010) 75-103

12. Hadzic, M., Sidhu, A.: Digital Health Ecosystems. Proceedings of the 2nd IEEE International Conference on Digital Ecosystems and Technologies (DEST 2008). IEEE, Thailand (2008) cv-cvii

13. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly **28** (2004) 75-105

14. Jiang, J.J., Conrath, D.W.: Semantic similarity based on corpus statistics and lexical taxonomy. International Conference on Research in Computational Linguistics (ROCLING X), Taiwan (1997) 19-33

15. Lin, D.: An Information-theoretic definition of similarity. 15th International Conference on Machine Learning (ICML-98), Madison (1998) 296-304

16. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and Application of a Metric on Semantic Nets. IEEE Transactions on Systems, Man and Cybernetics 19 (1989) 17-30

17. Resnik, P.: Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. Journal of Artificial Intelligence Research 11 (1999) 95-130

18. Rijsbergan, C.J.V.: Information Retrieval. Butterworths (1979)

19. Robertson, S.E., Jones, K.S.: Relevance weighting for search terms. Journal of the American Society for Information Science 27 (1976) 129-146

20. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of the ACM **18** (1975) 613-620

21. Seco, N.: Computational models of similarity in lexical ontologies. University College Dublin, Dublin (2005)

22. Sowa, J.F.: Semantic Networks. In: Shapiro, S.C. (ed.): Encyclopedia of Artificial Intelligence. Wiley (1992)

23. Spärck Jones, K.: A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation **28** (1972) 11-21

24. Su, L.T.: The relevance of recall and precision in user evaluation. Journal of the American Soceity for Information Science and Technology **45** (1999) 207-217

25. Tseng, Y.-y., Yue, W.L., Taylor, M.A.P.: The role of transportation in logistics chain. Proceedings of the 2005 Eastern Asia Society for Transportation Studies (EASTS 2005). Association for Planning Transportation Studies, Bangkok, Thailand (2005) 1657-1672

Chapter 11 – Recapitulation and Future Work

11.1 Introduction

Digital Ecosystems are characterised by ubiquitous, heterogeneous, context-dependent and dynamic services, which have increasingly attracted the attention of researchers in recent years. In addition, the field of semantic search has recently been given much attention by researchers. At the time of writing this thesis, semantic search and its various applications is being investigated and researched by various researchers. As is evident from the state-of-the-art survey carried out in Chapter 2, several proposals have been made by various researchers to address the application of semantic retrieval in the area of Web service discovery and other areas. At the same time however, the major shortcoming of these proposals is that none of them provides a complete solution for service retrieval in the Digital Ecosystems environment. The reason for this is that none of these proposals presents an integrated methodology for the Digital Ecosystem environment that provides: 1) service discovery; 2) service retrieval; and 3) service recommendation so that the service requesters without relevant domain knowledge about their service requests can be recommended relevant service domain knowledge to denote their service requests; 4) a QoS-based service ranking so that service requesters can rank retrieved service advertisements based on their context-specific QoS information; and, 5) service domain knowledge update so that the knowledge employed for semantic search can be duly updated.

In order to propose a complete solution for service retrieval in the Digital Ecosystems environment, in this thesis we identified five research issues and addressed them.

In the next section, we discuss the research issues related to service retrieval in the Digital Ecosystems environment that were addressed in this thesis. In Section 11.3 we outline and discuss the contributions of this thesis to the existing body of literature. Section 11.4 concludes the thesis and sets the stage for future work.

11.2 Research Issues Addressed in this Thesis

In this thesis, we addressed six major issues with service retrieval in a Digital Ecosystems environment and undertook to:

1. Propose a methodology by which service information on the Internet can automatically be discovered, annotated and classified, taking into account the heterogeneous and context-dependent nature of services and the heterogeneous nature of service providers in Digital Ecosystems.

2. Propose a methodology by which a service requester can precisely retrieve service information, taking into account the heterogeneous nature of service requesters in Digital Ecosystems.

3. Propose a methodology by which a service requester without relevant service domain knowledge about his/her service request can be given relevant knowledge recommendation, taking into account the heterogeneous nature of service requesters in Digital Ecosystems.

4. Propose a methodology by which a service requester can evaluate the trustworthiness of a service advertisement, and rank the retrieved service advertisements based on context-specific QoS evaluation criteria, taking into account the context-dependent nature of services in Digital Ecosystems.

5. Propose a methodology by which service domain knowledge can be duly updated in order to correctly reveal people's understanding of domain knowledge, taking into account the dynamic nature of services in Digital Ecosystems.

6. Validate the proposals for service retrieval in the Digital Ecosystems environment by simulation and functional testing.

11.3 Contribution of the Thesis to the Existing body of Literature

The major contribution of this thesis to the existing body of literature is that it proposes a complete methodology for service retrieval in the Digital Ecosystems environment by taking into account the heterogeneous, context dependent and dynamic nature of services, and the heterogeneous and dynamic nature service providers and service requesters in the Digital Ecosystems environment. The complete solution encompasses five methodologies which comprise the contribution of this thesis to the existing body of literature. The five methodologies encompassed by the thesis are as follows:

1. Methodology for automatically discovering, annotating and classifying Digital Ecosystem service information over the Internet

2. Methodology for service retrieval in the Digital Ecosystems Environment

3. Methodology for service concept recommendation in the Digital Ecosystems Environment

4. Methodology for QoS evaluation and service ranking in the Digital Ecosystems Environment

5. Methodology for service domain knowledge updating and service-provider-based service metadata publishing, maintenance and classification in the Digital Ecosystems Environment

Prior to developing a complete solution for service retrieval in the Digital Ecosystems environment, this thesis provides a comprehensive state-of-the-art survey of the various proposals in the existing body of literature for semantic search, which is an additional contribution of this thesis to the existing body of literature.

In this section, we provide a brief overview of the six contributions made by this thesis to the existing body of literature.

11.3.1 Contribution 1: State-of-the-art Survey of Present Literature

In Chapter 2, we carried out an extensive state-of-the-art survey in the four areas of the existing research in semantic search – semantic crawlers, semantic service discovery, semantic search engines and technologies, and semantic similarity models. To the best of our knowledge, other researchers who have presented a state-of-the-art survey on semantic search research are Hildebrand et al. [2] and Mangold [3]. These surveys are widely cited by various researchers. However, the state-of-the-art survey carried out and presented by us in Chapter 2, is more comprehensive, more extensive and more current than that carried out by Hildebrand et al. [2] and Mangold [3].

For the purpose of discussion and evaluation, we divided the existing body of literature on semantic crawlers into five classes based on functionality, which are as follows:

1. Metadata abstraction crawlers

2. Semantic focused crawlers

3. Metadata harvesting crawlers

4. Metadata abstraction focused crawlers

5. Metadata harvesting and abstraction crawlers

For the purpose of discussion and evaluation, we divided the existing body of literature on semantic service discovery into four classes based on application environments, which are as follows:

1. Semantic service discovery in centralized environments

2. Semantic service discovery in P2P environments
3. Semantic service discovery in grid computing environments
4. Semantic service discovery in ubiquitous computing environments

For the purpose of discussion and evaluation, we divided the existing body of literature on semantic search engines and technologies into eleven classes based on Grimes [1]'s classification schema, which are as follows:

1. Related searches engines and technologies
2. Semantic search engines and technologies for reference results
3. Search engines and technologies for semantically annotated results
4. Full-text similarity search engines and technologies
5. Search engines and technologies on semantic/syntactic annotations
6. Concept search engines and technologies
7. Ontology-based search engines and technologies
8. Semantic Web search engines and technologies
9. Faceted search engines and technologies
10. Clustered search engines and technologies
11. Natural language search engines and technologies

For the purpose of discussion and evaluation, we divided the existing body of literature on semantic similarity models into three classes based on utilized information, which are as follows:

1. Edge (distance)-based models
2. Node (information content)-based models
3. Hybrid models

11.3.2 Contribution 2: Methodology for Service Information Discovery, Annotation and Classification in the Digital Ecosystems Environment

The second major contribution of this thesis is that it proposes a methodology for automatic service information discovery, annotation and classification in the Digital Ecosystems environment. The methodology was presented in Chapter 5. To the best of our knowledge, the existing body of literature on semantic service discovery does not propose any means for service information discovery in Digital Ecosystems. The salient features of this methodology are as follows:

1. The methodology takes into account the heterogeneous and context-dependent nature of services and the heterogeneous nature of service providers in Digital Ecosystems.

2. It enables the automated discovery of heterogeneous generic service information from the Digital Ecosystems environment in terms of domain-specific service knowledge and crawling technology, in contrast to the existing body of literature on semantic service discovery which focuses only on the Web service domain which has been discussed in Chapters 1 to 4.

3. It enables the automatic exploration of natural language-described service information, in contrast to the existing body of literature on semantic service discovery which works only on WSDL-described service information, as discussed previously.

4. It enables the automatic annotation of the discovered service information by means of ontology mark-up languages.

5. It enables the automatic organization of the annotated service information based on specific Digital Ecosystem service domain knowledge.

6. It proposes two alternative algorithms for Service Description Entity (SDE) metadata-service concept matching, which have shown outstanding advantages in the ontology environment in two experiments, compared with the traditional information retrieval algorithms by the simulation displayed in Chapter 10.

11.3.3 Contribution 3: Methodology for Service Retrieval in the Digital Ecosystems Environment

The third major contribution of this thesis is that it proposes a methodology for retrieving service metadata in Digital Ecosystems based on user queries. The methodology was presented in Chapter 6. To the best of our knowledge, the existing body of literature on semantic service discovery and semantic search engines and technologies does not propose any means for semantically retrieving generic service advertisements over the Internet. The salient features of this methodology are as follows:

1. The methodology takes into account the heterogeneous nature of service requesters in Digital Ecosystems.
2. It proposes a method for semantically retrieving generic services, in contrast to the omission of semantic generic service retrieval in the existing body of literature, which has been discussed previously.
3. It proposes a method by which the service requesters who do not have domain knowledge with regard to their service queries can precisely retrieve a service advertisement based on their service queries and the HCIs between service requesters and the search engine.
4. The methodology proposes two alternative algorithms for user query-service concept matching, which have shown outstanding advantages in the ontology environment in two experiments, compared with the traditional information retrieval algorithms demonstrated by the simulation presented in Chapter 10.

11.3.4 Contribution 4: Methodology for Service Concept Recommendation in the Digital Ecosystems Environment

The fourth major contribution of this thesis is that, in case a service requester enters incomplete or incorrect queries into the search engine as a result of a lack of relevant domain knowledge, a methodology is proposed for recommending relevant service concepts to the service requester for query disambiguation. This methodology was proposed in Chapter 7. To the best of our knowledge, the existing body of literature on semantic service discovery and semantic search engines and technologies does not propose any means for solving the issue of query incompleteness or incorrectness. The salient features of this methodology are as follows:

1. The methodology takes into account the heterogeneous nature of service requesters in Digital Ecosystems.
2. It proposes a methodology for solving the issue of query incompleteness or incorrectness, in contrast to the omission of this issue in the existing body of literature, which has been discussed previously.
3. It proposes two alternative semantic similarity models for the ontology environment, by taking into account the content of ontological concepts in a concept similarity measure. This solves the limitation of the current semantic similarity models that focus only on measuring the relative positions between nodes within semantic networks, especially within definitional networks, and

ignore the semantic-rich content of ontological concepts, which have been discussed previously.

4. It proposes two semantic similarity models for a concept similarity measure, which show superior performance in the ontology environment in two experiments, compared with the existing semantic similarity models by means of the simulation introduced in Chapter 10.

11.3.5 Contribution 5: Methodology for QoS Evaluation and Service Ranking in the Digital Ecosystems Environment

The fifth major contribution of this thesis is that it proposes a methodology for: 1) a service requester to evaluate the trustworthiness of a service advertisement published by a service provider, by evaluating the performance of the service provider in the service entity, after completing a service transaction with the service provider; and, 2) service requesters to rank service advertisements based on service providers' past performance in services under context-specific QoS evaluation criteria. The methodology was introduced in Chapter 8. To the best of our knowledge, the existing body of literature on semantic service discovery and semantic search engines and technologies does not propose any means for generic service ranking and quality of generic service evaluation. The salient features of this methodology are as follows:

1. The methodology takes into account the context-dependent nature of services in Digital Ecosystems.

2. It proposes a method for evaluating the trustworthiness of a generic service advertisement, in contrast to the omission of quality of generic service evaluation in the existing body of literature on semantic service discovery, which has been discussed previously.

3. It takes into account all the criteria involved in a service entity based on a specific context.

4. It takes into account whether or not a given criterion has been mutually agreed by a service requester and a service provider.

5. It takes into account service requesters' perceptions of the importance of each criterion involved in a service entity.

11.3.6 Contribution 6: Methodology for Service Domain Knowledge Updating and Service-provider-based Service Metadata Publishing, Maintenance and Classification

The sixth contribution of this thesis is that it proposes a methodology for updating service domain knowledge based on agreements among knowledge users, and for service providers to advertise their service entities and manually annotate their service entities with specific service domain knowledge. The methodology was described in Chapter 9. To the best of our knowledge, the existing body of literature on semantic crawlers, semantic service discovery, and semantic search engines and technologies does not propose any means for updating the employed ontologies/facets. The salient features of this methodology are as follows:

1. This methodology takes into account the dynamic nature of services in Digital Ecosystems.

2. It proposes a method for updating domain knowledge in semantic search engines, in contrast to the existing body of literature on semantic crawlers, semantic service discovery, and semantic search engines and technologies, which has tended to ignore the issue, as discussed previously.

3. It takes into account the different weights of knowledge users in different service domains when determining a change in domain knowledge.

4. It takes into account service domain ontologies and QoS evaluation criteria as the object for service domain knowledge updating.

5. It takes into account the role of service providers in the service metadata classification process, as a complement of the automatic service metadata classification process implemented by the Semantic Crawlers.

11.4 Conclusion and Future Work

The work that we have undertaken has been published extensively as refereed journal papers, book chapters and for international conference proceedings. Since the start of this research and to date, we have published six journal papers, five book chapters and 15 conference papers on this topic. We have attached some selected publications in the appendix. Additionally, the list of the publications arising as a result of the work documented in this thesis is attached in the front pages.

Although we have undertaken extensive research on the topic of this study, there is scope for future work. It is our intention to continue working on this topic, primarily along, but not limited to the following lines:

1. Text mining for ambiguous generic service information in the Web. As introduced previously, unlike the WSDL-described Web service information, generic service information is embedded in vast Web information, is described by natural languages, and is thus difficult to explore. Moreover, generic service information does not retain a consistent format and standard, and varies from webpage to webpage. Therefore, it is necessary to use text mining technologies to mine the ambiguous generic service information from Web documents. Whereas, we utilized a rule-based text mining approach in our semantic crawler in a website and the experimental results are convincing. However, owing to the difference between websites, we cannot ensure the success of this approach in universal websites. Therefore, as a part of our future work, we intend to design a universal approach to mining service information from multiple websites by means of text mining approaches and specific service domain knowledge.

2. Ontology representation of the contexts involved in a service entity. As introduced previously, Digital Ecosystem service information is context-dependent, and therefore has different content and different QoS evaluation criteria in different contexts. Although we use service ontologies to annotate the contexts of service entities, the contextual differences are shown only in QoS evaluation criteria and do not obviously impact on the content of service entities, which could result in the confusion of users regarding the contexts of service entities. Our future research will involve the development of an ontological representation of the contexts involved in a service entity, so that the contexts of service entities can be clearly observed by users in the service retrieval process.

3. Advanced algorithms for service metadata-ontological concept matching and user query-ontological concept matching in the ontology environment. In this thesis, we proposed two algorithms – ECBR and IECBR for service metadata-ontological concept matching in the metadata classification process, and their variations for user query-ontological concept matching in the service retrieval process. In our experiments, the two algorithms show better performance in terms of precision and recall, compared with the traditional information retrieval algorithms. However, the experiments revealed two shortcomings of both algorithms. The first is that the response times of the two algorithms are slower than for the PM algorithm in the service retrieval experiments. The second drawback is that the performance of the two algorithms relies on the specificity of terms that define ontological concepts. Hence, as a part of our future work, we intend to design advanced algorithms in order to resolve these issues, by considering the characteristics of ontological concepts and service metadata in the matching process.

4. Advanced semantic similarity models. In this thesis, we propose two alternative semantic similarity models for measuring concept similarity in the ontology environment. Despite the two models showing better performance on the aggregated scales (F-measure (β=1) and F-measure (β=1)) in two experiments,

compared with the existing semantic similarity models, we are not satisfied with the current testing results in terms of application. Hence, as a part of our future work, we intend to design semantic similarity models with better performance in experiments, by considering the characteristics of ontological concepts and relationships in the measuring process.

5. An advanced methodology for updating community-based service domain knowledge. In this thesis, we proposed a methodology for updating community-based service domain knowledge, which takes into account the different weights of domain experts and normal users' votes in the voting for change process. However, we do not propose any means for computing the weights for different users' weights. Thus, as a part of our future work, we intend to associate the domain-specific reputation values of users with the weights of their votes in the domain-specific voting process.

11.5 References

1. Grimes, S.: Breakthrough analysis: Two + nine types of semantic search. InformationWeek. United Business Media, Manhasset, NY, USA (2010).

2. Hildebrand, M., Ossenbruggen, J.v., Hardman, L.: An analysis of search-based user interaction on the Semantic Web. Stichting Centrum voor Wiskunde en Informatica, Amsterdam, Netherland (2007), http://www.narcis.info/publication/RecordID/oai:cwi.nl:12302

3. Mangold, C.: A survey and classification of semantic search approaches. International Journal of Metadata, Semantics and Ontologies **2** (2007) 23-34

Appendix – Selected Publications