การค้นหาเว็บเซอร์วิสเชิงความหมายโดยใช้เซอร์วิสโปรไฟล์เชิงโครงสร้างและเชิงพฤติกรรมแบบมีกฎ

นางสาวเนตรนภา สีหารี

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรดุษฎีบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2548

ISBN 974-53-2692-5

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

SEMANTIC WEB SERVICES DISCOVERY USING STRUCTURAL AND RULE-BASED BEHAVIOURAL SERVICE PROFILES

Miss Natenapa Sriharee

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

A Dissertation Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy Program in Computer Engineering
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic year 2005
ISBN 974-53-2692-5

Thesis Title	Semantic Web Services Discovery Using Structural And Rule-Based Behavioural Service Profiles
By	Miss Natenapa Sriharee
Filed of study	Computer Engineering
Thesis Advisor	Assistant Professor Twittie Senivongse, Ph.D.

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Doctor's Degree

[illegible] .... Dean of Faculty of Engineering

(Professor Direk Lavansiri, Ph.D.)

THESIS COMMITTEE

[illegible] .... Chairman

(Yunyong Teng-amnuay, Ph.D.)

[illegible] .... Thesis Advisor

(Assistant Professor Twittie Senivongse, Ph.D.)

[illegible] .... Member

(Proadpran Punyabukkana, Ph.D.)

[illegible] .... Member

(Vishnu Kotrajaras, Ph.D.)

[illegible] .... Member

(Benchaphon Limthanmaphon, Ph.D.)

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

เนตรนภา สีหารี: การค้นหาเว็บเซอร์วิสเชิงความหมายโดยใช้เซอร์วิสโปรไฟล์เชิงโครงสร้างและเชิงพฤติกรรมแบบมีกฎ. (SEMANTIC WEB SERVICES DISCOVERY USING STRUCTURAL AND RULE-BASED BEHAVIOURAL SERVICE PROFILES)
อ. ที่ปรึกษา : ผศ. ดร. ทวิติย์ เสนีวงศ์ ณ อยุธยา จำนวนหน้า 100 หน้า ISBN 974-53-2692-5

การค้นหาเซอร์วิสเป็นการทำงานซึ่งเป็นหัวใจหลักอย่างหนึ่งของเทคโนโลยีการพัฒนาระบบเชิงเซอร์วิส เช่น เทคโนโลยีเว็บเซอร์วิส ในปัจจุบันการกำหนดเนื้อหาสาระหรือรายละเอียดของข้อมูลที่ใช้อธิบายรายละเอียดของเซอร์วิสเป็นหัวข้อที่นักวิจัยได้ให้ความสนใจเป็นอย่างมาก เนื่องจากรายละเอียดของคำอธิบายเซอร์วิสจะส่งผลต่อความยืดหยุ่นในการค้นหาและความเที่ยงตรงของผลลัพธ์ที่ได้เมื่อเทียบกับความต้องการของผู้ค้นหา

วิทยานิพนธ์ฉบับนี้นำเสนอโมเดลคำอธิบายเซอร์วิสเชิงความหมาย ซึ่งใช้ออนโทโลยีในการแสดงรายละเอียดของเซอร์วิสในโดเมนงานหนึ่งๆ และนำเสนอคลังคำอธิบายเซอร์วิสเชิงความหมาย ซึ่งเป็นส่วนขยายของยูดีดีไอในมาตรฐานเว็บเซอร์วิสให้สามารถเก็บข้อมูลคำอธิบายเซอร์วิสเชิงความหมายตามโมเดลดังกล่าวและทำการค้นหาเซอร์วิสได้อย่างยืดหยุ่น โมเดลคำอธิบายเซอร์วิสเชิงความหมายจะอธิบายความสามารถในการให้บริการของเซอร์วิสในรูปของโครงสร้างการให้บริการ พฤติกรรมของเซอร์วิสแบบมีกฎ รวมไปถึงคุณลักษณะอย่างง่ายและคุณลักษณะเชิงความหมายของเซอร์วิส การค้นหาเซอร์วิสจะกลายเป็นการค้นหาเชิงความหมาย เนื่องจากผู้ค้นหาสามารถสืบค้นเซอร์วิสที่มีการทำงานหรือพฤติกรรม รวมไปถึงคุณลักษณะต่างๆ ตามที่ต้องการได้ วิทยานิพนธ์นี้นำเสนออัลกอริทึมในการจับคู่เซอร์วิสกับความต้องการของผู้ค้นหา และจัดอันดับเซอร์วิสตามระดับความใกล้เคียงกับความต้องการ โดยผู้ค้นหาจะสามารถระบุเกณฑ์ที่ใช้ช่วยในการจับคู่และจัดอันดับเซอร์วิสได้ตามความชอบของตน นอกจากนี้วิทยานิพนธ์ยังได้นำเสนอการประเมินอัลกอริทึมที่ใช้ในการจับคู่และจัดอันดับเซอร์วิสด้วย

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต [illegible]
สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่ออาจารย์ที่ปรึกษา ทวิติย์ เสนีวงศ์ ณ อยุธยา
ปีการศึกษา 2548

4471814721 : MAJOR COMPUTER ENGINEERING

KEY WORD: SERVICE DISCOVERY / SEMANTIC WEB SERVICES / ONTOLOGY / MATCHMAKING

NATENAPA SRIHAREE: SEMANTIC WEB SERVICES DISCOVERY USING STRUCTURAL AND RULE-BASED BEHAVIOURAL SERVICE PROFILES. THESIS ADVISOR: ASST.PROF. TWITTIE SENIVONGSE, Ph.D., 100 pp. ISBN 974-53-2692-5.

Service discovery is one key aspect in the enabling technologies for service-oriented systems including Web Services. Growing attention has been paid to the content of business and service descriptions to allow services to be discovered more flexibly and accurately.

This thesis presents a semantic service description model which adopts ontology as a shared knowledge representation for describing service descriptions in a particular domain. A semantic registry is proposed to complement the standard UDDI registry to maintain semantic service descriptions and enable flexible service discovery. The semantic service description model describes service capabilities by structural and rule-based behavioural profiles as well as simple attribute and semantic attribute profiles. Service discovery will be semantics-based since query can be defined in terms of functional capabilities, behaviour, or attributes. The thesis proposes the algorithms for matchmaking and ranking services against service consumers' requirements. Service consumers can specify preference criteria for matching and ranking. Also, an evaluation of these algorithms is discussed.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Department Computer Engineering	Student's signature........Natenapa Sriharee........
Field of study Computer Engineering	Advisor's signature.......[illegible].......
Academic year 2005	

ACKNOWLEDGEMENTS

I am especially deeply grateful to my advisor, Assistant Professor Dr. Twittie Senivongse, who has provided me with a great deal of guidance and has supported me with great words that motivate me towards this success.

I would like to thank the Commission on Higher Education, Ministry of Education, who awarded me the scholarship, and also to Supaphorn and Tawa, who are always beside me through the ups and downs during the whole course of doctoral study.

I will not be able to come this far without everlasting love and supports from my beloved parents. Special thanks go to my father, who is my tremendous inspiration towards this successful step in life and also to my sister and brother who constantly give me encouragement and goodwill.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CONTENTS

Page

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

LIST OF TABLES

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

LIST OF FIGURES

Figure Page

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER I

INTRODUCTION

1.1 Problems and Motivation

Web Services are networked applications that are able to interact using standard application-to-application Web protocols over well-defined interfaces (W3C, 2002). Web Services have been targeted for service-oriented architecture in which service providers can offer their services on the Web, and service consumers can dynamically discover and invoke the services. Service consumers can interact with service matchmakers, for example at http://www.xMethods.com, http://www.capescience.com, and http://www.salcentral.com. These Web sites act as an intermediary for service providers to advertise their Web Services and for service consumers to query for those services. The matchmakers categorise the Web Services into various groups depending on the domains or the functions of the services. Therefore the service consumers usually have to browse those categories and the service providers' homepages in order to select a Web Service to interact with.

Web Services architecture itself provides such a matchmaker functionality through the standard registry called the Universal Description, Discovery and Integration (UDDI) (uddi.org, 2002). The information model in UDDI attempts to standardise business and service metadata. Its information model defines attributes for Web Services, consisting of Business Entity, Business Service, Binding Template, Publisher Assertion, and Operational Info (Figure 1.1). A Business Entity refers to a service provider that publishes its Web Services, and is detailed by business name, description, contact, a list of business categories to which this provider belongs, and a list of Business Services that it provides. A Business Service describes a Web Service that is offered by the provider, and is described by service name, description, a list of service categories to which this service belongs, and a list of Binding Templates. A Binding Template defines where a Web Service is located and the technical details about that service. This is described by a description, access point (e.g. a link to access the Web Service or even a link to the homepage of the service provider), and a tModel. A tModel describes the technical specifications implemented by the service such as the service interface and access protocol written in the Web Services Description Language (WSDL) (W3C, 2001). A Business Entity may also be associated with a Publisher Assertion that specifies a relationship with other Business Entity. An Operational Info keeps tracks of changes that occur to other kinds of UDDI information entries.

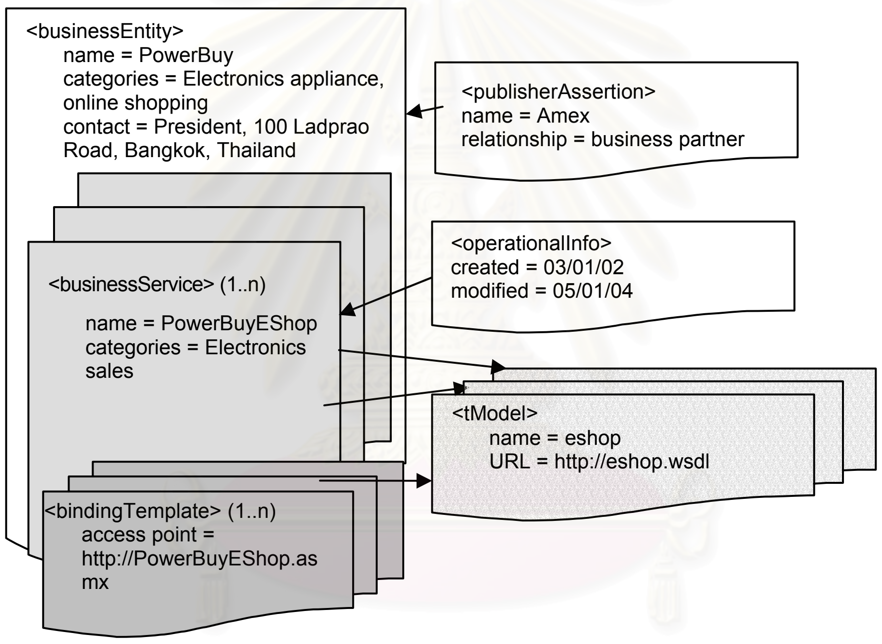


Figure 1.1 UDDI information model.

With the set of attributes above, UDDI offers a limited support for Web Services discovery. The attribute set is coarse and gives only preliminary information about the service providers and the offered Web Services. Generally a search is by matching of the name or category of Business Entities, Business Services, or tModels against the values specified in the query. For example, a query could be “find information of a business called

PowerBuy" or "find Web Services in the electronics sales category", and the rest is left to the service consumers to browse the Web pages of those companies to make a selection. This is difficult when there are a lot of Web pages to browse or when the service consumers want to query for businesses or services based on semantic or behavioural information. In Figure 1.2, a service consumer wants to buy a desktop computer by using an Amex credit card and the service should be able to deliver the computer to the consumer's address after payment has been made. Such a query is closer to a query by semantic knowledge about the Web Service and its behaviour. To accommodate this kind of query, Web Services are required to advertise their semantics and behaviour, but it would be difficult and inconvenient to express such information and support such a query under the capability the standard UDDI currently provides.

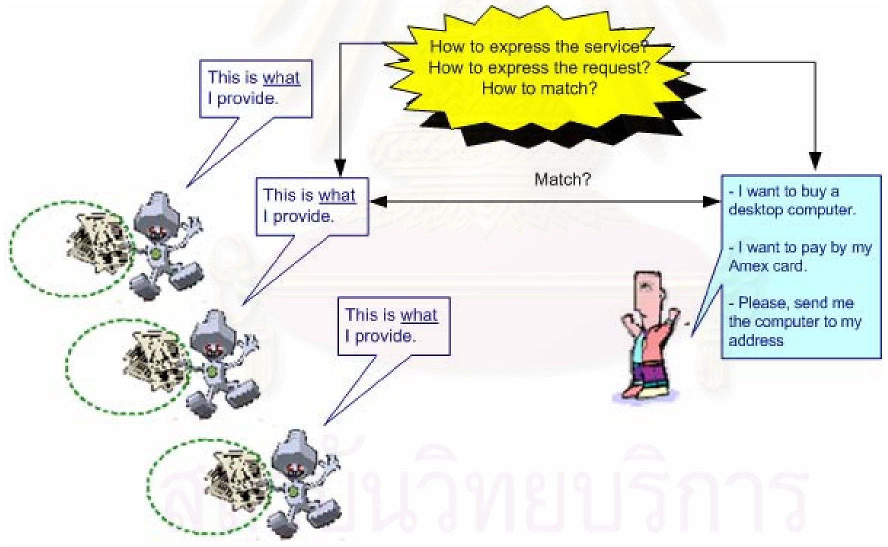

How to express the service?
How to express the request?
How to match?
This is what I provide.
This is what I provide.
This is what I provide.
Match?
- I want to buy a desktop computer.
- I want to pay by my Amex card.
- Please, send me the computer to my address

Figure 1.2 Semantics-based query.

With the emergence of Semantic Web (Semantic Web, 2001) by which well-defined semantics is given to the information on the Web, Web information model becomes semantic descriptions which are described by ontology (Gruber, 1993). Ontology is an agreement for representing conceptualisation or knowledge about a particular domain of interest in order to enable common understanding among the participants within such a domain. When information on the Web has some ontology or knowledge associated with it, computer agents or programs can analyse such knowledge, enabling semantics-based discovery of Web information.

Since Web Services are seen as one kind of Web resources, there are a number of efforts to also add semantics to Web Services to make Semantic Web Services. A Semantic Web Service will be described by a semantics-enriching markup language. The semantic description will facilitate the external agents to understand both the internal structure and functionality of the Semantic Web Service and to be able to discover, compose, and invoke the Semantic Web Service. The current approach is towards describing various aspects of the semantics and behaviour of a Web Service and integrating the semantic information with the standard Web Services architecture. For example, an ordinary Web Service will be enhanced by an ontology-based behavioural model or has its WSDL annotated with semantics.

This research falls in the area of Semantic Web Services discovery. The focus is on the enhancement to the discovery mechanism that is currently supported by UDDI. Web Services will be advertised by their *integrated service profiles*. An integrated service profile is referred to as a collection of profiles that represent service metadata in terms of attributes, structure, behaviour, and operational constraints of the service. Each profile is created by adopting service domain ontologies. The brief details of the profiles are as follows.

(i) Attribute profile models the static characteristics of a Web Service and represents the characteristics by a set of attribute names and attribute values. Attribute profile consists of two subprofiles: simple attribute profile and semantic attribute profile, where the former refers to the service attributes whose values are simple lexical values while the latter refers to an ontology-based service attributes whose values are ontological terms.

(ii) Structural profile models the knowledge structure of a Web Service, i.e. product detail, sales detail, and product delivery methods.

(iii) Behavioural profile models the functional behaviour of a Web Service in terms of operation, input, output, precondition, and effect. In the case that the service exhibits particular behaviour under a certain condition, such a condition can be defined by using the following operational rule-based profile.

(iv) Operational rule-based profile models the rules that constrain the structure or behaviour of a Web Service.

The main technology in this research is the use of Web ontology language and ontology inference engine to express and query Web Services by considering their semantic descriptions. This research will look at modelling semantic descriptions, creating and publishing integrated service profiles, specifying the query, matching and ranking the services, and deploying semantics onto UDDI registry.

1.2. Objectives

The objectives of this research are as follows:

1.2.1 Propose an integrated service profile for Web Services. The profile consists of attribute profile, structural profile, behavioural profile, and operational rule-based profile.

1.2.2 Develop a mechanism that enables semantic discovery of Web Services based on the proposed integrated service profile. The mechanism will support discovery of individual Web Services.

1.3. Scope of Work

The scope of this work is as follows.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

1.3.1 Propose the upper ontology for attribute profile, structural profile, behavioural profile, and operational rule-based profile.

1.3.2 Propose a mechanism for semantic service discovery that supports publishing and query of individual Web Services.

1.3.3 Propose similarity criteria for determining matched results.

1.3.4 Develop a prototype of the architecture for the proposed discovery mechanism.

1.3.5 Not consider ontology change.

1.3.6 Use OWL for ontology language, Jena for ontology reasoning, XML and RDF for request.

1.4 Research Methodology

1.4.1 Review and study research papers that are related to service discovery.

1.4.2 Design all information models including all ontologies and profiles.

1.4.3 Develop a prototype.

1.4.4 Analyse the result of the matchmaking process.

1.4.5 Write thesis.

1.5 Organisation of Thesis

The remainder of the thesis is organised into six chapters. In Chapter II, theoretical background and related literature are reviewed. Semantic Web technology and Semantic Web Services discovery are introduced. Also, semantic approaches to enhance UDDI are discussed. In Chapter III, metadata for semantic Web Services and details about modelling ontologies for creating semantic service profiles are proposed. Examples of semantic attribute profile, structural profile, behavioural profile, and operational rule-based profile for publishing and querying services are illustrated. In Chapter IV, matchmaking and ranking methodologies are proposed and their evaluation can be found in Chapter V. Chapter VI presents the semantic Web Services discovery architecture and finally, the summary of this thesis is in Chapter VII.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER II

THEORETICAL BACKGROUND AND LITERATURE REVIEW

This chapter focuses on background study in the area of semantic Web Services discovery. Sections 2.1-2.3 present semantic technology and general approaches to apply semantics to Web Services. Section 2.4 discusses other research papers related specially to this research.

2.1 Semantic Web and Ontology

Semantic Web is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation (Semantic Web, 2001). Figure 2.1 shows a stack of Semantic Web technology which comprises the standards and tools of XML, XML Schema, RDF, RDF Schema, and ontology language. XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents. XML Schema is a language for restricting the structure of XML documents. RDF is a data model for objects (resources) and relations between them, providing simple semantics for this data model and representing it in an XML syntax. RDF Schema is a vocabulary for describing properties and classes of RDF resources, with the semantics for generalisation hierarchies of such properties and classes. Ontology language provides a vocabulary, that is richer than RDF Schema, for describing properties and classes and relations between classes.

Ontology is a formal explicit specification of a shared conceptualisation (Gruber, 1993). In other words, ontology is an agreement for representing conceptualisation or knowledge within a particular domain of interest in order to enable common understanding among the participants within that domain. Ontology has become a flexible tool that allows the participants to describe their information by using their own vocabularies, and ontology reasoning can reason the meaning and mediate diverse sources of information from different domains. Semantic markup languages provide schema language model to represent semantic of Web resources in ontology and are usually in terms of XML-based, RDF-based, and RDFS-based languages (Costello and Jacobs, 2003). There are several semantic markup languages for describing ontology, for example DAML (released in October, 2000 by daml.org), OIL (Fensel et al., 2000), DAML+OIL (a combination of DAML-ONT and OIL), SHOE (Heflin et al., 1999) (pure XML-based markup), and OWL (W3C, 2004) (a recently becoming standard language for Semantic Web).

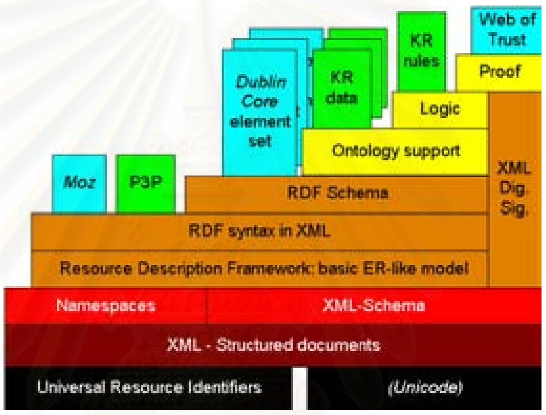


Figure 2.1. Semantic Web stack.

In this research, we select OWL as the ontology language for describing service metadata because of its expressive capability that can represent semantic expressions in various aspects with clear meanings. Figure 2.2 shows an example of wine ontology from http://ontolingua.stanford.edu/doc/chimaera/ontologies/wines.daml. It shows common vocabularies or concepts (i.e. Class and Property) in such a knowledge domain and also the relationships between those concepts. A particular RDF instance or resource can refer to its domain ontology and describe its own semantics. An inference engine can infer or derive more facts or knowledge for a particular domain from the facts that are already present in the domain ontology. For example, SELAKS-ICE-WINE is a kind of ice wine that is a subclass of LATE-HARVEST and DESSERT-WINE and the colour is white. The prototype of this research uses Jena (Jena, 2003) for interpreting and inferring the semantics defined in the ontologies.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<rdf:RDF xmlns="http://ontolingua.stanford.edu/doc/chimaera/ontologies/wines.daml#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:daml="http://www.daml.org/2001/03/daml+oil#">

   <rdf:Property rdf:ID="REGION">
      <rdf:type rdf:resource="http://www.daml.org/2001/03/daml+oil#UniqueProperty"/>
      <rdfs:range rdf:resource="#WINE-REGION"/>
      <rdfs:domain rdf:resource="#WINE"/>
   </rdf:Property>
   <rdf:Property rdf:ID="FLAVOR">
      <rdf:type rdf:resource="http://www.daml.org/2001/03/daml+oil#UniqueProperty"/>
      <rdfs:range rdf:resource="#WINE-FLAVOR"/>
      <rdfs:domain rdf:resource="#WINE"/>
   </rdf:Property>
   <rdf:Property rdf:ID="MAKER">
      <rdf:type rdf:resource="http://www.daml.org/2001/03/daml+oil#UniqueProperty"/>
      <rdfs:range rdf:resource="#WINERY"/>
      <rdfs:domain rdf:resource="#WINE"/>
   </rdf:Property>
   <rdf:Property rdf:ID="BODY">
      <rdf:type rdf:resource="http://www.daml.org/2001/03/daml+oil#UniqueProperty"/>
      <rdfs:domain rdf:resource="#WINE"/>
      <rdfs:range rdf:resource="#WINE-BODY"/>
   </rdf:Property>
   <rdf:Property rdf:ID="COLOR">
      <rdf:type rdf:resource="http://www.daml.org/2001/03/daml+oil#UniqueProperty"/>
      <rdfs:range rdf:resource="#WINE-COLOR"/>
      <rdfs:domain rdf:resource="#WINE"/>
   </rdf:Property>

   <daml:Restriction rdf:ID="MEDIUM-OR-FULL-BODY-TO-CLASS-RESTRICTION">
      <daml:onProperty rdf:resource="#BODY"/>
      <daml:toClass>
         <rdfs:Class>
            <daml:oneOf rdf:parseType="daml:collection">
               <rdf:Description rdf:about="#FULL"/>
               <rdf:Description rdf:about="#MEDIUM"/>
            </daml:oneOf>
         </rdfs:Class>
      </daml:toClass>
   </daml:Restriction>
   <daml:Restriction rdf:ID="MODERATE-OR-STRONG-FLAVOR-TO-CLASS-RESTRICTION">
      <daml:onProperty rdf:resource="#FLAVOR"/>
      <daml:toClass>
         <rdfs:Class>
            <daml:oneOf rdf:parseType="daml:collection">
               <rdf:Description rdf:about="#MODERATE"/>
               <rdf:Description rdf:about="#STRONG"/>
            </daml:oneOf>
         </rdfs:Class>
      </daml:toClass>
   </daml:Restriction>


<rdfs:Class rdf:ID="ICE-WINE">
      <rdfs:subClassOf rdf:resource="#MEDIUM-OR-FULL-BODY-TO-CLASS-RESTRICTION"/>
      <rdfs:subClassOf rdf:resource="#MODERATE-OR-STRONG-FLAVOR-TO-CLASS-
RESTRICTION"/>
      <daml:intersectionOf rdf:parseType="daml:collection">
         <rdfs:Class rdf:about="#LATE-HARVEST"/>
         <rdfs:Class rdf:about="#DESSERT-WINE"/>
         <daml:Restriction rdf:about="#WHITE-COLOR-RESTRICTION"/>
      </daml:intersectionOf>
   </rdfs:Class>

<rdf:Description rdf:ID="SELAKS-ICE-WINE">
      <rdf:type rdf:resource="#ICE-WINE"/>
      <REGION rdf:resource="#NEW-ZEALAND"/>
      <MAKER rdf:resource="#SELAKS"/>
      <FLAVOR rdf:resource="#MODERATE"/>
      <BODY rdf:resource="#MEDIUM"/>
      <COLOR rdf:resource="#WHITE"/>
</rdf:Description>
```

Figure 2.2 Fragment of wine ontology.

2.2 Semantic Web Services Discovery

Web Services technology realises the basic service-oriented architecture (SOA) which comprises three kinds of participants: service provider, service consumer, and service discovery agent or matchmaker (c.f. UDDI registry in Web Services). Enhancing service discovery with semantics can be achieved by introducing a semantic matchmaker as an engine to complement the standard matchmaker (Figure 2.3). This semantic matchmaker will be aware of the semantic knowledge that is added to enrich service metadata. Such semantics can be represented by a semantic markup language such as OWL and used for discovery. Therefore, descriptions of a service can be seen as comprising descriptions by standard description model and additional semantic descriptions. For Web Services technology, the association between a semantic description and the information entry of a Web Service in the standard UDDI can be maintained via a tModel specification to which the service refers (Sriharee et al., 2004). A service consumer can query for a Web Service by interacting with the semantic matchmaker and specifying a semantic-based request.

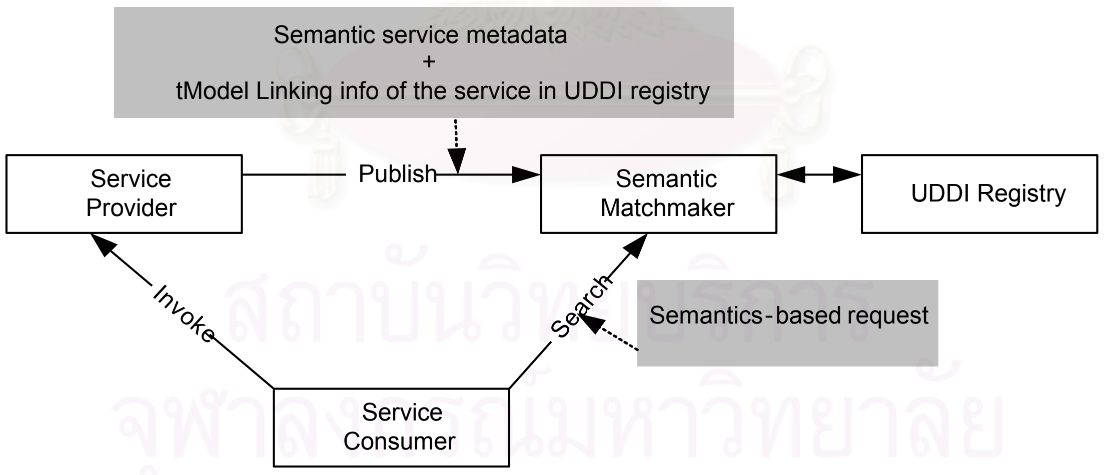


Figure 2.3 Enhancing service model with semantic approach.

The general picture of the framework for semantic Web Services discovery is depicted in Figure 2.4 and the scenario is as follows:

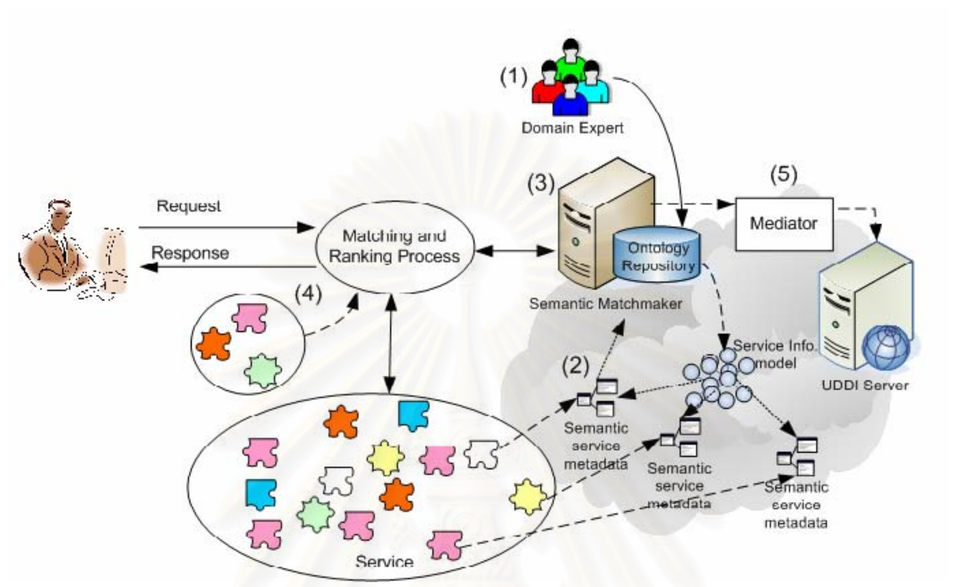


Figure 2.4 Overview of semantic Web Services discovery.

(1) Domain experts define ontologies for a particular service domain. The ontologies will describe vocabularies for various aspects of the service domain such as general knowledge or capability of services in the domain. The ontologies are stored in an ontology repository which is maintained by the framework.

(2) Service providers can create semantic service metadata by deriving from domain-specific ontologies and by using an ontology editor such as Protégé (Protégé, 2001). Semantic service metadata will be stored at service providers' locations but the references to the semantic descriptions will be registered with the semantic matchmaker in order to maintain the association with the corresponding information entries of the services within the standard matchmaker (Dogac et al., 2002). Once this is done, service consumers can submit semantic requests to query with the semantic matchmaker.

(3) Semantic matchmaker is a software module responsible for semantic service discovery. Its main functions are the matching and ranking processes that consider semantics of the Web Services against semantic query.

(4) The matching process will try to match the requests from the service consumers to the descriptions of the published Web Services. The result will be a list of individual Web Services that can realise the requests.

(5) Once appropriate Web Services are discovered, their corresponding information entries in the standard matchmaker can be retrieved by the help of a mediator which is a software module that maintains the links between the semantic service metadata and the standard UDDI information entries of the Web Services.

This research follows this general architecture of the semantic discovery framework but will focus on the model of semantic service metadata and the algorithms for matching and ranking Web Services.

2.3 Enriching Service Metadata by Semantics

Model for Web Services metadata according to Web Services standard includes UDDI information model (uddi.org, 2002) as in Figure 1.1 and WSDL interface specification (Christensen et al., 2001). There are two approaches to enrich semantics into the service metadata model – by annotating semantics onto the standard service metadata and by proposing separate semantic specifications.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

2.3.1 Annotating Semantics onto Standard Service Metadata

This is the approach taken by Sriharee and Senivongse (2003) and Sivashanmugan et al. (2003) in which WSDL document of a Web Service is tagged with the behavioural concepts that have been defined in relevant ontologies. Figure 2.5 depicts an example of an annotated WSDL from (Sriharee and Senivongse, 2003). Additional XML schemas such as WSDLext:semantic-operation, WSDLext:semantic-data, WSDLext:semantic-condition, WSDLext:conditionalEffect, WSDLext:conditionalOutput, WSDLext:effect, and WSDLext:servicecommunity are defined for specifying semantic information.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE uridef [
<!ENTITY do "http://www.wssemantic.com/flightbooking.daml#">
<!ENTITY lo "http://www.AmexTravel.com/amextravel.daml#"> ]>
<definitions
name="AmexTravel"
targetNamespace="http://www.AmexTravel.com/amextravel.wsdl"
xmlns:tns="http://www.AmexTravel.com/amextravel.wsdl"
xmlns:WSDLext="http://www.wssemantic.com/WSDLext.dtd"
xmlns:do="&do;"
xmlns:lo="&lo;"  ...>
 <message name="BookingAirTravelResponse">
<part name="AirTicket" type="xsd1:AirTicketDetail"/>
 </message>
 <message name="BookingAirTravelRequest">
<part name="BookingFlightDetail" type="xsd1:BookingFlightDetail"/>
<part name="creditCardDetail" type="xsd1:creditCardDetail"/>
 </message>
 <portType name="AmexTravelPortType">
   <operation name="BookingTicket"

    WSDLext:semantic-operation="&do;BookingFlightTicket">

 <documentation>Provide service for booking air ticket
     </documentation>
 <input message="tns:BookingAirTravelRequest"
    WSDLext:semantic-data="&do;BookingFlightInfo">
 </input>
 <WSDLext:precondition
     WSDLext:semantic-condition="&do;CreditcardHolder"/>
 <output message="tns:BookingAirTravelResponse" />
 <WSDLext:conditionalOutput WSDLext:semantic-data="&do;AirTicket">
  <WSDLext:condition WSDLext:semantic-condition="&lo;AmexHolder">
      <WSDLext:conditionalEffect
           WSDLext:semantic-effect="&lo;CollectMileReward"/>
  </WSDLext:condition>
 </WSDLext:conditionalOutput>
 <WSDLext:conditionalOutput
     WSDLext:data="&lo;AirTicketWithFeeCharged">
  <WSDLext:condition
      WSDLext:semantic-condition="&lo;OthersCardHolder"/>
 </WSDLext:conditionalOutput>
 <WSDLext:effect WSDLext:semantic-effect="&do;DeliverTicket"/>
 <WSDLext:effect WSDLext:semantic-effect="&do;CreditcardCharged"/>
 </operation>
</portType>
  <binding name="AmexTravelBinding"
            type="tns:AmexTravelPortType">...

  <service name="AmexTravel" WSDLext:service="AmexTravelBooking">
    <WSDLext:servicecommunity
      WSDLext:semantic-community="&do;FlightBooking"/>
    <port binding="tns:AmexTravelBinding" name="AmexTravelPort">
     <soap:address
         location="http://www.AmexHolder.com/BookingAmexTicket"/>
    </port>
  </service>
</definitions>
```

Figure 2.5 Annotated WSDL.

In this approach, annotated semantics has to be extracted before reasoning can be performed. This may be inconvenient as parsing WSDLs can be time-consuming when a large number of Web Services are available and mixing semantics with standard information can be troublesome for maintenance.

2.3.2 Proposing Separate Semantic Service Specification

This approach is dominated by the DAML services coalition who proposed an ontology-based language for semantic Web Services called DAML-S (The DAML Services Coalition, 2002). DAML-S and its successor called OWL-S (Martin et al., 2004) model service metadata with three profiles, i.e. Service Profile, Process Model, and Service Grounding. Figure 2.6 depicts the upper ontology for describing services in OWL-S.

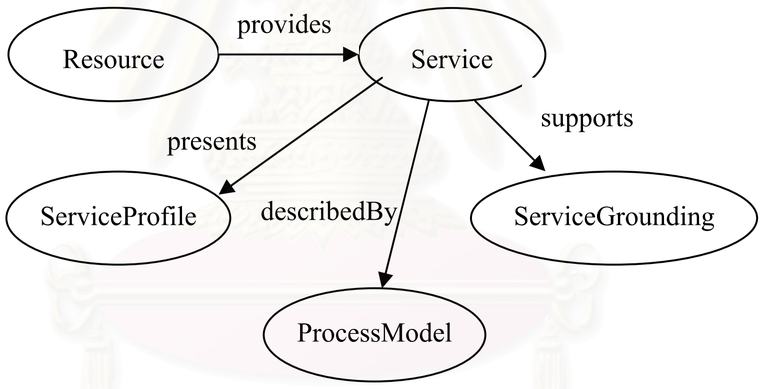


Figure 2.6 Upper ontology for describing services in OWL-S.

OWL-S Service Profile is defined in terms of general attributes of the service such as service name, category, quality rating, and provider information, as well as functional attributes such as input, output, precondition, and effect. These functional attributes are taken from the Process Model. The Process Model describes dynamic view of the service in terms of functional behaviour, internal processes of the service, and its workflow. Figure 2.7 shows part of the Service Profile and Process Model that describes the functional behaviour of a BookingFlightTicket service. This is the counterpart of the annotated semantics in Figure 2.5. OWL-S Service Grounding describes a mapping between abstract specification of OWL-S and concrete specification of Web Services such as WSDL.

```
--- Service Profile ---
<profile:Profile rdf:ID="BookingFlightTicketProfile">
    <profile:hasInput rdf:resource="#BookingFlightInfo"/>
    <profile:hasOutput rdf:resource="#AirTicketWithFeeCharge"/>
    <profile:has_process rdf:resource="#BookingFlightTicket"/>
    <profile:hasOutput rdf:resource="#AirTicket"/>
    <profile:hasResult rdf:resource="#BookedWithOthersCard"/>
    <profile:hasResult rdf:resource="#BookedWithAmex"/>
    <profile:hasPrecondition rdf:resource="#CreditcardHolder"/>
</profile:Profile>

--- Process Model ---
  <process:AtomicProcess rdf:ID="BookingFlightTicket">
    <process:hasPrecondition>
      <expr:Condition rdf:ID="CreditcardHolder">
        <expr:expressionLanguage rdf:resource="http://www.daml.org/services/
               owl-s/1.1/generic/Expression.owl#SWRL"/>
      </expr:Condition>
    </process:hasPrecondition>
    <process:hasResult>
      <process:Result rdf:ID="BookedWithOthersCard">
        <process:hasEffect rdf:resource="#DeliverTicket"/>
        <process:hasEffect rdf:resource="#CreditcardCharged"/>
        <process:inCondition>
          <expr:Condition rdf:ID="OthersCardHolder">
            <expr:expressionLanguage rdf:resource="http://www.daml.org/
                services/owl-s/1.1/generic/Expression.owl#SWRL"/>
          </expr:Condition>
        </process:inCondition>
      </process:Result>
    </process:hasResult>
    <process:Result rdf:ID="BookedWithAmex">
        <process:hasEffect rdf:resource="#DeliverTicket"/>
        <process:hasEffect rdf:resource="#CreditcardCharged"/>
        <process:hasEffect rdf:resource="#CollectMileReward"/>
    <process:withOutput>
      <process:OutputBinding rdf:ID="AirTicketWithAmex">
        <process:toParam>
          <process:Output rdf:ID="AirTicket"/>
        </process:toParam>
      </process:OutputBinding>
    </process:withOutput>
    <process:inCondition>
      <expr:Condition rdf:ID="AmexHolder">
        <expr:expressionLanguage rdf:resource="http://www.daml.org/services/
              owl-s/1.1/generic/Expression.owl#SWRL"/>
      </expr:Condition>
    </process:inCondition>
  </process:Result>
    <process:hasOutput rdf:resource="#AirTicket"/>
    <process:hasOutput>
      <process:Output rdf:ID="AirTicketWithFeeCharge"/>
    </process:hasOutput>
    <process:hasInput>
      <process:Input rdf:ID="BookingFlightInfo"/>
    </process:hasInput>
  </process:AtomicProcess>
```

Figure 2.7 BookingFlightTicket service in OWL-S Service Profile and Process Model.

In general, Service Profile is aimed for service discovery while other profiles are for other purposes, such as Process Model are generally for service composition and coordination.

This thesis will follow this approach by providing additional integrated service profiles which will be associated with the Web Services for easy reasoning. Also, the association between these profiles and the information entries of the Web Services within the UDDI will still be maintained.

2.4 Literature Review

As introduced in Section 1.1, this research will propose an integrated service profile which consists of attribute profile (i.e. simple attribute profile and semantic attribute profile), structural profile, behavioural profile, and operational rule-based profile. This section will discuss related papers that also take semantic approaches to service metadata and service matchmaking.

As mentioned in Section 2.3.2, OWL-S (Martin et al., 2004) or its predecessor DAML-S define a collection of ontology-based profiles, i.e. Service Profile, Process Model, and Service Grounding. OWL-S Service Profile overlaps by function with our attribute profile and behavioural profile. However, our attribute profile will accommodate more useful attributes. Paolucci et al., (2002) describes Web Services with DAML-S and discovers Web Services based on operation, input, and output only. Scale of matching by ontological subsumption is also proposed but without ranking. In our approach, all aspects of the behavioural profile can be used in the query including precondition and effect. Our behavioural profile is also enhanced by the operational rule-based profile and service constraint evaluation. As suggested by (Trastour et al., 2002; Li and Horrocks, 2003), additional profiles such as those proposed in our approach can be supplementary to OWL-S Service Profile.

Some other work that tries to extend matchmaking on DAML-S such as (Bansal and Vidal, 2003) presents an algorithm that considers details in DAML-S Process Model for service matching based on input and output. Jaeger and Tang (2004) takes properties into account for matching of two profiles. Ranking is proposed based on subsumption of input, output, and property.

Another effort to define semantic specifications for Web Services is Web Services Modeling Ontology (WSMO) (WSMO, 2004). WSMO provides the conceptual framework for semantically describing Web Services. Based on WSMO, the Web Services Modeling Language (WSML) (Bruijn et al., 2005) implements this conceptual framework in a formal language for annotating Web Services with semantic information. WSML defines semantics in terms of four elements: ontologies, goals, Web Service descriptions, and mediators. Ontologies provide vocabularies, concepts, instances, and axioms that will be used by other elements. Goals are similar to queries. Web Service descriptions describe capability in terms of assumption, precondition, postcondition, effect, and allow for interface and orchestration specifications. Mediators connect different WSMO elements and resolve heterogeneity between them. Although not the same, our integrated service profile has capabilities that correspond to many of WSML elements. Our semantic attribute profile and structural profile allows specifications of concepts and instances. The simple attribute profile allows for references to interface or orchestration specifications. The behavioural profile corresponds to the WSML capability. WSML provides syntax for conceptual structure of Web Services but does not say exactly what capability or vocabulary should be defined. Our integrated service profile gives a clearer picture of service metadata since the profile defines more concrete details, e.g. what attributes should be defined in the attribute profile, what basic information should be provided in the structural profile, and the provision of shared domain ontologies which are concrete template for descriptions. WSML capability is defined by formal logical expressions, so it is powerful but would require complex reasoning for query and powerful tools. Matchmaking in WSMO defines types of matching based on the number of elements that are matched (i.e. exact, subsumption, plug-in, intersection). Ranking is simple and based on such types of matching. Our matchmaking and ranking will be more complex and refined as there are several profiles involved and each profile will be considered for matching first before combining the results for ranking. Types of matching in our approach will vary depending on the capabilities that are considered.

UDDI version 4 is trying to incorporate an ontology-based taxonomy for the standard categories of Business Entity and Business Service (Paolucci and Sycara, 2004). This effort will allow UDDI to also return businesses or services of a specialised or generalised category. This corresponds to having business or service categories defined as semantic attributes in our approach. However, the semantic attribute profile is open to any attributes.

Other work presents semantic-based frameworks based on description logics formalisation and description logic reasoning. Trastour et al. (2002) and Li and Horrocks (2003) propose matchmaking in the e-commerce scenario in a multi-agent system when an advertisement represents the product description. This corresponds to our structural profile. They consider matching by subsumption relationship between the request and each advertisement in the repository. Ranking is not addressed in their work. Sycara et al. (2002) propose the agent-based framework in which the service capability is described by a description language LARKS. They propose a matchmaker which consists of a number of filters, each of which performs partial matching on the descriptions. Advertisements and request specifications will be compared whether they are sufficiently similar by using TF-IDF method and word distance values. Signature and constraints of input and output are also checked. Our work focuses on the specification of the profiles and mainly uses approximate match based on similarity through subsumption to determine the degree of similarity, as also adopted by (Paolucci et al., 2002; Di Noia et al., 2003; Andreasen et al., 2003).

Other work integrates semantic information into UDDI architecture such as METEOR-S (Verma et al., 2004) that proposes a framework for enhancing WSDL description by annotation of semantics into WSDL and also extending UDDI registry with semantics.

Sivashanmugan et al. (2003) enhances service metadata by using ontological information to annotate service functions in the WSDL document. They use ontological subsumption for matching of operation, input, and output. Searching for services allows for both the standard Web Service search and the extended search by semantics that is proposed as ontological taxonomy in UDDI. Matching is also based on ontological input, output, and operation, but ontological reasoning is not available in their framework.

Bernstein and Klein (2002) takes a different approach to discover a service based on the process model of its behaviour. The process model is ontology-based and consists of subactivities. Such discovery is for planning a new business logic view. However, discovery on the process model is not in the scope of our research.

This thesis will propose a mechanism that accommodates discovery based on the semantics-based integrated service profile. Unlike most related work that addresses only the behaviour aspect of a Web Service, this thesis will consider attribute-based, semantic structure, and behavioural aspects. For the behavioural aspect, a Web Service will take the approach similar to the related work and model a Web Service by its operation, input, output, precondition, and effect, but all of these behavioural aspects will be used to determine a match. Moreover, rule-based constraints in an operational rule-based profile can be put onto the behavioural model, and therefore a behavioural match will additionally require rule-based evaluation. Similarity criteria for determining matched results will be defined as well.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER III

METADATA FOR SEMANTIC WEB SERVICES

3.1 Metadata Model : Integrated Service Profile

Metadata for Web Services give information primarily for service consumers to get to know any potential services without having to really deploy them, and therefore are useful sources of information for discovering Web Services. With regards to the purpose of the interaction between service providers and service consumers, information about attributes, characteristics, operational aspects, and deployment aspects of Web Services are typical Web Services metadata (Booth et al., 2004). Simple metadata for Web Services are modelled as attribute-based service descriptions (uddi.org, 2002; Dumas et al., 2001), meaning that the characteristics and other information about the services are described through a set of concrete attributes with corresponding attribute values. To enable more flexible and accurate discovery, semantic annotation is added to Web Services metadata (Martin et al., 2004; WSMO, 2004). Whether they are attribute-based or semantics-based, metadata will influence individual and organisational use of the services (Lynne and Soh, 2002), and therefore should reflect both functional and psychological needs of the individuals and organisations.

The model for Web Services metadata should attract service providers to publish useful information and at the same time facilitate service consumers to discover the right services. Web Services discovery will be enhanced if it can allow semantic query such as *"Find an online electronics shop that sells desktop computers and is rewarded Thailand Electronics Association's Vendor award from the Ministry of Commerce. The store should accept Amex credit card and deliver the computer that I have bought to my place (in Bangkok) within 3 days".* Such a query involves several aspects of service semantics which should be published (Oaks et al., 2003).

It is assumed here that service providers will do their best to please service consumers, and will advertise rich information regarding their profiles and service capabilities in order to get themselves discovered easily. This research works around the questions "what should be in a service description to allow service consumers to query more conveniently and flexibly" and "how such information in the service description can help the consumers make a service selection".

To answer the question concerning how to model the metadata for Web Services, we conducted an empirical survey, as reported previously in (Tapabut et al., 2003), to find what information should be included in Web Services metadata model. Information was gathered from Web Services brokerage sites (such as http://www.salcentral.com, http://www.capescience.com, http://www.xmethods.com, and http://www.webserviceoftheday.com), a survey on commercial software components on the Internet market (since Web Services can be seen as service components), and a survey on relevant research papers. Our empirical survey resulted in an attribute-based model for Web Services metadata; some part of it is shown in Table 3.1.

Table 3.1 Part of the survey result on Web Services descriptions

Service	Operational Info		ServiceName, Version, TimeOfRelease, …
	Functional Info		Domain, Description, DevelopmentEnv, QoS, Security, …
	Commercial Info	Purchasing	Pricing, Licensing, …
		Incentive	Award, ReferenceCustomer, Promotion, Testing, …
	Technical Support		Contact, FAQ, …
	Specification		Interface, Structure, Behaviour, Component, …
Provider	ProviderName, About, Domain, Certificate, …		

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

It is seen from Table 3.1 that some information can be easily modelled as attributes, meaning that simple attribute values can be assigned (e.g. ServiceName, Description, Award), while some refers to more complex values (e.g. interface, structure, or behaviour information). We hence see Web Services metadata as a combination of attribute-based information and more complex specifications on which complex analysis of the service characteristics can be conducted. As semantic annotation is a major vehicle to more flexible service discovery, this research uses ontology as a shared formal representation (Gruber, 1993) to represent semantics of Web Services in those specifications.

In this research, a combination of information models of semantic service profiles will be referred to as an *integrated service profile* (Sriharee and Senivongse, 2005). An integrated service profile is a metadata model which is a combination of various information models and it is a shared metadata model for semantic Web Services description in a specific domain (Figure 3.1). The integrated service profile comprises a number of subprofiles which maintain either the attribute-based information or the more complex capability-based information as follows:

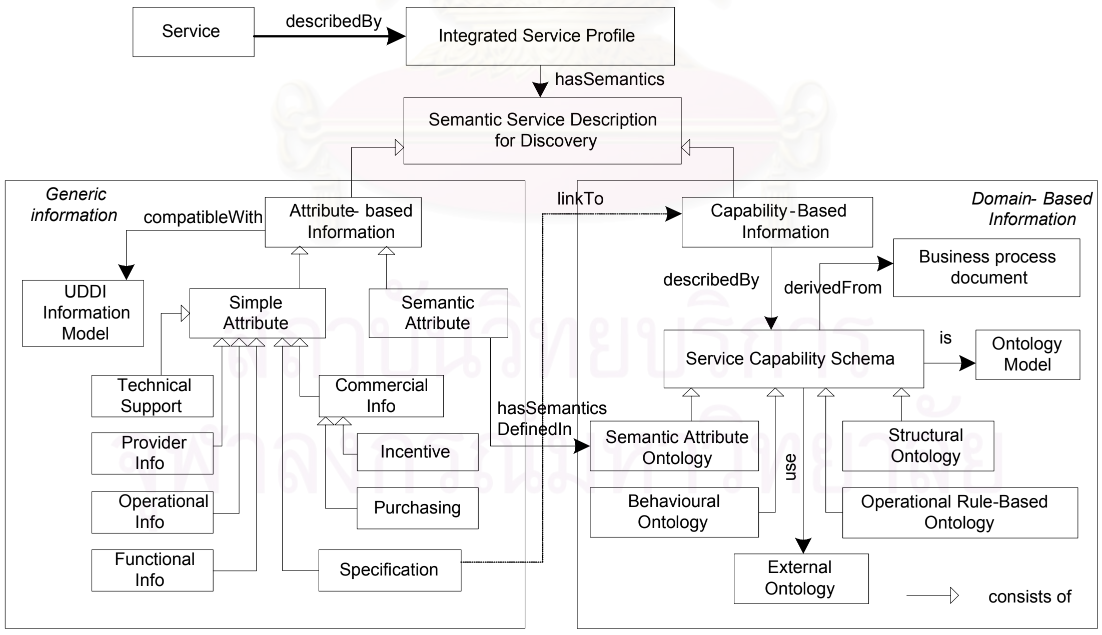


Figure 3.1 Conceptual model of the Integrated Service Profile.

(i) *Attribute-based information* refers to those attributes in Table 3.1. This set of attributes is applicable to model Web Services of any application domains. It is also compatible with the attributes define in the standard UDDI information model; some attributes in the set can be mapped directly to those in the UDDI registry while some can be accommodated by an extended registry. Most of the attributes are *simple attributes* as they can be characterised by simple attribute values. Nevertheless, ontology can be useful to turn a simple attribute into a *semantic attribute* by assigning an ontological term, defined in a *semantic attribute ontology*, as its value (Sriharee et al., 2004a). For example, the value 'ThailandBestBrand' of the attribute Award may be a term in an *external ontology* (i.e. an award-related ontology), not just a simple string value. This will enable the matchmaking process to perform ontological matching, rather than string matching, when comparing the consumer's query against the service's capability. Simple attributes of a Web Services will be maintained by an extended UDDI whereas semantic attributes will be maintained by a *semantic attribute profile.*

(ii) *Capability-based information* refers to the more complex aspects of Web Services, i.e. the capabilities, which will be represented by ontology-based *service capability schema*. The capabilities here relate to the specification-oriented attributes in Table 1 as follows:

- *Service structure* captures fundamental knowledge structure of Web Services of a particular domain. This is static information that service consumers would generally expect to know such as the product of the service, sales detail, and means of service delivery (Trastour et al., 2001; Li and Horrocks, 2003). Service structure is represented by a *structural ontology* in a *structural profile*.

- *Service behaviour* captures more dynamic behavioural information by which a Web Service is modelled as providing a function which may require some inputs in order to produce some outputs and effects under certain conditions. This behavioural model is aligned with WSDL, OWL-S, and WSMO. Service behaviour is represented by a *behavioural ontology* in a *behavioural profile*.

- *Service constraints* capture constraints on service provision in terms of rules. Rules may state conditions or policies concerning the activity of Web Services (c.f. structural assertion rules (Hay and Healy, 2000)), and add the dynamicity to the semantics of Web Services. To give a better view of what business rules may be enforced within a particular service domain, service constraints within the domain can be represented by an *operational rule-based ontology* in an *operational rule- based profile*. These constraints can then be associated with either the service structure or service behaviour.

This research uses ontology as a shared formal representation for semantics-based metadata, the ontology is modelled using a top-down approach in which the development process starts with the definition of the most general concepts followed by subsequent specialisation of the concepts (Gómez-Pérez, 1999). In this research, the general concepts for all capability-based metadata will be introduced as upper ontologies (see Section 3.2). As the name implies, capability-based metadata should in fact vary according to different capabilities of Web Services in different domains. Experts in a particular application domain who are familiar with the nature and business processes of the domain will therefore subsequently derive, from the upper ontologies, the service structure, service behaviour, and operational constraints for the domain. Service providers in this domain can then use such shared domain ontologies as templates for publishing their own capability-based profiles. On defining ontology-based metadata, auxiliary *external ontologies* can also be used to define some data elements which make the ontologies more complete. For example, the structural profile of an electronics appliance vendor may use an electronics appliance manufacturer ontology for the concept that represents the product model that are available at the vendor's shop.

Combining attribute-based and all capability-based profiles, the integrated service profile will be able to accommodate both conventional service discovery via attribute values matching and semantic discovery via ontological analysis on ontology-based metadata.

3.2 Service Capability Schema

This section focuses on modelling all capability-based profiles with ontology. As mentioned in the previous section, the conceptual model for semantic Web Services comprises two layers of ontology, namely the upper ontology layer and the service domain ontology layer. Service domain ontology is derived from base concepts in the upper ontology and defines new concepts that are specific to the domain. Subsequently, service providers can describe their capability-based profiles based on the domain ontologies. Figure 3.2 depicts roles of service providers and domain experts in preparing capability-based profiles with ontology, while Figure 3.3 shows graphically the two-layer ontology model – upper ontologies for service capability and domain ontologies derived from the upper ontologies. Details are as follows:

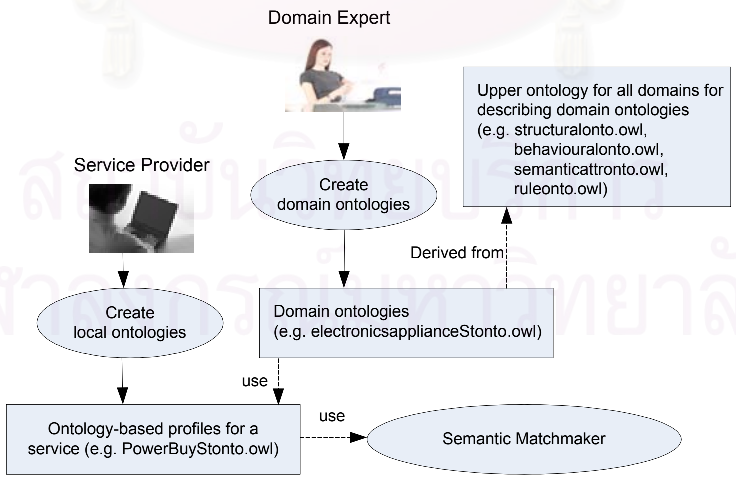

Domain Expert
Upper ontology for all domains for describing domain ontologies (e.g. structuralonto.owl, behaviouralonto.owl, semanticattronto.owl, ruleonto.owl)
Service Provider
Create domain ontologies
Derived from
Create local ontologies
Domain ontologies (e.g. electronicsapplianceStonto.owl)
use
use
Ontology-based profiles for a service (e.g. PowerBuyStonto.owl)
Semantic Matchmaker

Figure 3.2 Role of service providers and domain experts in preparing capability-based profiles.

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

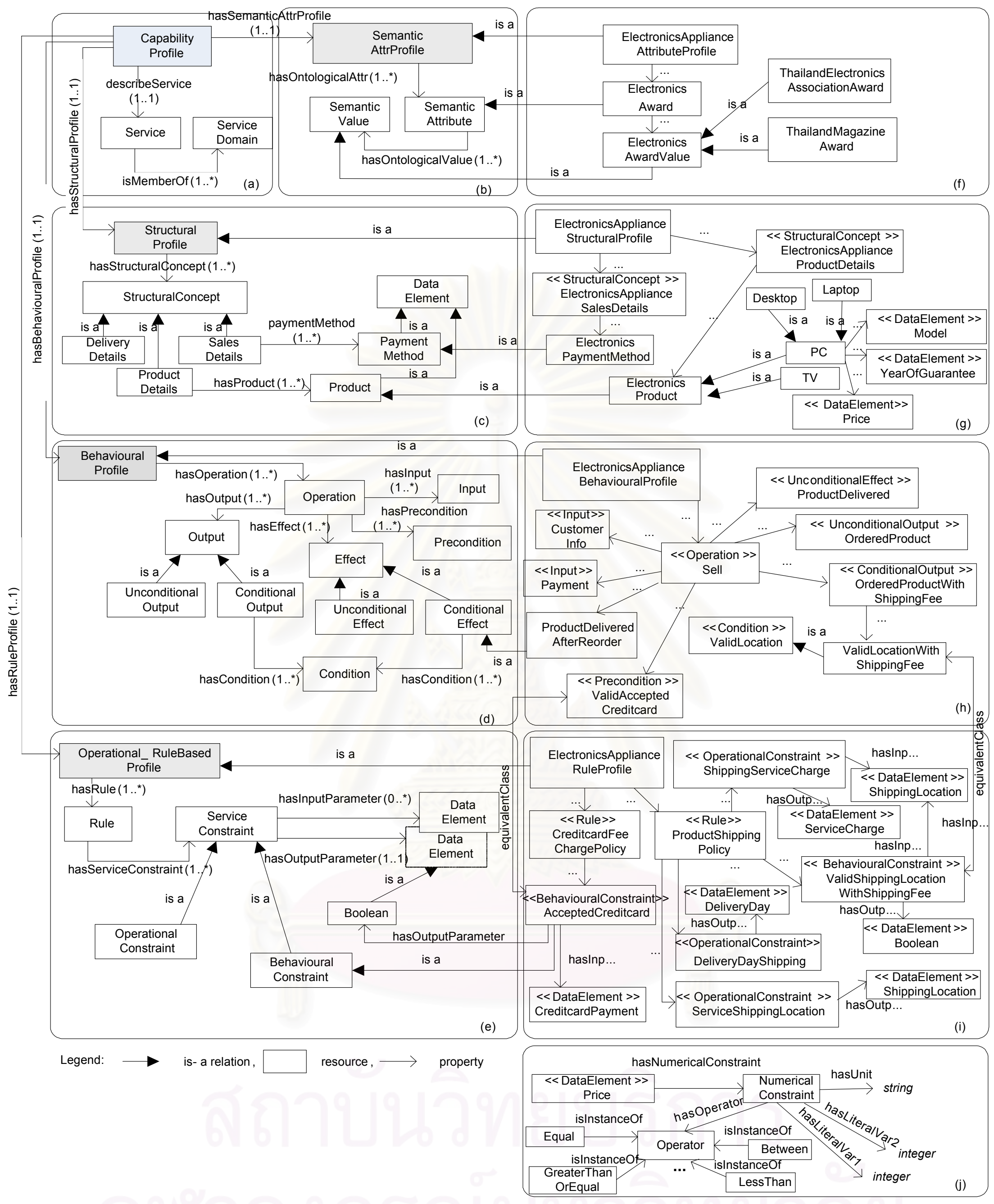


Figure 3.3 Ontology model for service metadata (a) Capability ontology (b)-(e) Upper ontologies for semantic attribute, structural, behavioural, and operational-rule based profiles respectively (f)-(i) Domain ontologies for electronics appliance domain: semantic attribute, structural, behavioural, and operational-rule based ontologies respectively (j) Ontology for numerical constraints.

(i) *Capability ontology* (Figure 3.3 (a)) models a collection of Web Service capabilities within a domain. It is used to derive a *capability profile* which refers to the semantic attribute profile, structural profile, behavioural profile, and operational rule-based profile of a Web Service. The capability profile itself does not actually represent any of the service capabilities (so it will not be considered further in the matching and ranking process).

(ii) *Upper semantic attribute ontology* (Figure 3.3 (b)) models a number of semantic attributes whose values are ontological values. The upper ontology contains the concepts SemanticAttrProfile, SemanticAttribute, and SemanticValue and is used to derive a *semantic attribute ontology* of a domain. Figure 3.3 (f) shows a semantic attribute ontology for the semantic attribute ElectronicsAward in the ElectronicsAppliance domain. The ontology defines vocabularies for the kinds of awards and the providers in the domain can use them as the values for the attribute ElectronicsAward.

(iii) *Upper structural ontology* (Figure 3.3 (c)) models the structure of fundamental static knowledge about a Web Service. It is used to derive a *structural ontology* of a domain which contains a number of StructuralConcepts including SalesDetails, ProductDetails, and DeliveryDetails. The concept ProductDetail in the upper ontology models either tangible products (e.g. a desktop PC from an electronics appliance vendor) or intangible products (e.g. information obtained from a search engine). The concepts SalesDetail and DeliveryDetail can respectively model information about payment and channels for service delivery. Figure 3.3 (g) shows a structural ontology for services in the ElectronicsAppliance domain. It defines possible payment methods and vocabularies of products within the domain, with relevant product details such as model, years of guarantee, and price. Such product details are defined with the concept DataElement, meaning that they are data concept that may be used from other external ontologies to add details to the structural ontology.

(iv) *Upper behavioural ontology* (Figure 3.3 (d)) models functional capability of a Web Service in terms of its operations. Each operation requires some inputs and produces different outputs and effects, sometimes when particular conditions are satisfied (The DAML-S Services Coalition, 2002). The upper ontology is used to derive a *behavioural ontology* of a domain. The concept Operation in the upper ontology may have some Precondition that must hold before the service can function. Outputs and effects of the Operation may be ConditionalOutput or ConditionalEffect if there are some behavioural constraints associated with them; otherwise they will be UnconditionalOutput and UnconditionalEffect. By modelling behavioural capability as a collection of operations, the behavioural profile can then be used as a semantic specification for WSDL interface specification of a Web Service (Christensen et al., 2001). Figure 3.3 (h), shows a behavioural ontology for Web Services in the ElectronicsAppliance domain. It has an operation Sell which may require CustomerInfo and Payment detail as inputs. The precondition ValidAcceptedCreditcard says that the operation will function only when the customer provides a valid credit card (i.e. one of the credit cards accepted by the service). This precondition is an equivalentClass to the behavioural constraint AcceptedCreditCard in the operational rule-based ontology in Figure 3.3 (i) (see below). The operation may return any of the unconditional or conditional outputs/effects. The conditional output OrderedProductWithShippingFee specifies that the operation may reply with the ordered product and a shipping fee which has to be paid. But this depends on whether the customer is located in a valid shipping location (i.e. the condition ValidLocationWithShippingFee); otherwise there is no fee as there will be no shipping. ValidLocationwithShippingFee is defined as an equivalentClass to the behavioural constraint ValidShippingLocationWithShippingFee in the operational rule-based ontology.

(v) *Upper operational rule-based ontology* (Figure 3.3 (e)) models constraints on the provision of the service and is used to derive an *operational rule-based ontology* for a domain. Each rule states a constraint or policy of the activity of the service and is modelled by the concept ServiceConstraint which may require some inputs and will be evaluated into an output value (i.e. it reads as IF (inputs are true) THEN (return output)). The concept BehaviouralConstraint refers to the constraint that requires at least one input to be evaluated and returns a Boolean output value. The concept OperationalConstraint may or may not require input and may return a non-Boolean output value. BehaviouralConstraint is aimed for describing preconditions and conditions associated to outputs and effects in the behavioural profile (Sriharee and Senivongse, 2003). We assume that service consumers will be guided by the discovery framework to supply inputs needed for evaluation of any service constraints within the domain of interest. Figure 3.3 (i) shows an operational rule-based ontology for the ElectronicsAppliance domain with the policies on product shipping and on credit cards used for shipping payment. For ProductShippingPolicy, the service may have operational constraints on ServiceShippingLocation (which returns the locations covered in the shipping area), on DeliveryDayShipping (which returns the number of days required for shipping), and on ShippingServiceCharge (which returns the shipping charge based on the shipping location and value of payment). The behavioural constraint ValidShippingLocationWithShippingFee may return true or false depending on whether the customer is located in the area of shipping. For CreditCardFeeChargePolicy, the behavioural constraint AcceptedCreditCard may return true or false depending on whether the customer presents a credit card that is one of those accepted.

These domain ontologies will be defined by domain experts. We assume all service providers in the same domain share the same capability-based ontologies and do not consider the case that ontologies may change and that different groups of experts may define different domain ontologies. Service providers will publish their profiles according to these shared ontologies. We can use OWL (W3C, 2004) as an ontology language since several tools exist and it is recommended by W3C. Note that our approach also accommodates numerical constraints on the concepts defined in the structural or operational rule-based ontologies. For example, an electronics appliance vendor named PowerBuy may want to publish in the structural profile that the price of its PCs is between

20,000-80,000 bahts (e.g. 20000 $\leq$ Price $\leq$ 80000 bahts), or, in the operational rule-based profile, that the delivery day is no more than 3 days (e.g. DeliveryDay $\leq$ 3 days). Figure 3.3 (j) gives additional ontology for representing a simple formal expression for such numerical constraints (Sriharee et al., 2004b). Moreover, PowerBuy may use an existing rule language such as SWRL (Horrocks et al., 2003) to represent such rules.

3.3 Ontology-Based Profiles of Web Service

This section gives an example of the semantic attribute profile, structural profile, behavioural profile, and operational rule-based profile of a service called PowerBuy which is a service in the electronics appliance domain.

3.3.1 Semantic Attribute Profile

Figure 3.4 shows part of the semantic attribute profile of PowerBuy service which is an instance of the semantic attribute ontology of the electronics appliance domain (in Figure 3.3 (f)). It specifies that the attribute ElectronicsAwards has a value ThailandElectronicsAssociationAwardYear2005.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:sa="http://www.newregistry.com/semanticattronto#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:sad="http://www.newregistry.com/electronicsapplianceSaonto#"
    xmlns="http://www.powerbuy.com/PowerbuySaProfile#"
    xml:base="http://www.powerbuy.com/PowerbuySaProfile">
  <owl:Ontology rdf:about="">
    <owl:imports
rdf:resource="http://www.newregistry.com/electronicsapplianceSaonto"/>
  </owl:Ontology>
  <sad:ThailandElectronicsAssociationAward
            rdf:ID="ThailandElectronicsAssociationAwardYear2005"/>
  <sad:ElectronicsApplianceAttributeProfile
            rdf:ID="PowerBuySemanticAttributeProfile">
    <sa:hasSemanticAttr>
      <sad:ElectronicsAward rdf:ID="ElectronicsAwards2005">
        <sa:hasSemanticValue
            rdf:resource="#ThailandElectronicsAssociationAwardYear2005"/>
      </sad:ElectronicsAward>
    </sa:hasSemanticAttr>
  </sad:ElectronicsApplianceAttributeProfile>
</rdf:RDF>
```

Figure 3.4 Example of semantic attribute profile.

3.3.2 Structural Profile

Figure 3.5 shows part of the structural profile of PowerBuy service which is an instance of the structural ontology of the electronics appliance domain (in Figure 3.3 (g)). The profile specifies that PowerBuy sells laptops and desktops. For desktops, the price starts from 15,000. Payments by FirstChoice and Aeon cards are accepted.

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.powerbuy.com/PowerbuyStProfile#"
    xmlns:std="http://www.newregistry.com/electronicsapplianceStonto#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:st="http://www.newregistry.com/structuralonto#"
  xml:base="http://www.powerbuy.com/PowerbuyStProfile">
  <owl:Ontology rdf:about="">
    <owl:imports
rdf:resource="http://www.newregistry.com/electronicsapplianceStonto"/>
  </owl:Ontology>
  <std:ElectronicsApplianceStructuralProfile rdf:ID="PowerBuyStructuralProfile">
    <st:hasStructuralConcept>
      <st:ProductDetails rdf:ID="PowerBuyProductDetails">
        <st:hasProduct>
          <std:Laptop rdf:ID="PowerBuyLaptop"/>
        </st:hasProduct>
        <st:hasProduct>
          <std:Desktop rdf:ID="PowerBuyDesktop">
            <std:hasPrice>
              <std:Price rdf:ID="PriceDesktop">
                <std:hasNumericalConstraint>
                  <st:NumericalConstraint rdf:ID="PriceConstraint">
                    <st:hasUnit
                    rdf:resource="http://www.newregistry.com/structuralonto#Baht"/>
                    <st:hasOperator
                      rdf:resource="http://www.newregistry.com/
                              structuralonto#GreaterThanAndEqual"/>
                    <st:hasLiteralVar1
                         rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
                    >15000</st:hasLiteralVar1>
                  </st:NumericalConstraint>
                </std:hasNumericalConstraint>
              </std:Price>
            </std:hasPrice>
          </std:Desktop>
        </st:hasProduct>
      </st:ProductDetails>
    </st:hasStructuralConcept>
    <st:hasStructuralConcept>
      <st:SaleDetails rdf:ID="PowerBuySaleDetails">
        <st:paymentMethod>
          <std:FirstChoiceCreditcard rdf:ID="PBFirstChoice"/>
        </st:paymentMethod>
        <st:paymentMethod>
          <std:AeonCreditcard rdf:ID="PBAeonCard"/>
        </st:paymentMethod>
      </st:SaleDetails>
    </st:hasStructuralConcept>
  </std:ElectronicsApplianceStructuralProfile>
</rdf:RDF>
```

Figure 3.5 Example of structural profile.

3.3.3 Behavioural Profile

Figure 3.6 shows part of the behavioural profile of PowerBuy service which is an instance of the behavioural ontology of the electronics appliance domain (in Figure 3.3 (h)).

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.powerbuy.com/PowerbuyBhProfile#"
    xmlns:rld="http://www.newregistry.com/electronicsapplianceRlonto#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rl="http://www.newregistry.com/ruleonto#"
    xmlns:bhd="http://www.newregistry.com/electronicsapplianceBhonto#"
    xmlns:bh="http://www.newregistry.com/behaviouralonto#"
  xml:base="http://www.powerbuy.com/PowerbuyBhProfile">
  <owl:Ontology rdf:about="">
    <owl:imports
      rdf:resource="http://www.newregistry.com/electronicsapplianceBhonto"/>
  </owl:Ontology>
  <bhd:ValidLocationWithShippingFee
       rdf:ID="PowerBuyValidLocationWithShippingFee"/>
  <bhd:Sell rdf:ID="PowerBuySell">
    <bh:hasOutput>
      <bhd:OrderedProduct rdf:ID="PowerBuyOrderedProduct"/>
    </bh:hasOutput>
    <bh:hasEffect>
      <bhd:ProductDelivered rdf:ID="PowerBuyProductDelivered"/>
    </bh:hasEffect>
    <bh:hasPrecondition>
      <bhd:ValidAcceptedCreditcard rdf:ID="PowerBuyValidAcceptedCreditcard"/>
    </bh:hasPrecondition>
    <bh:hasInput>
      <bhd:CustomerInfo rdf:ID="PowerBuyCustomerInfo"/>
    </bh:hasInput>
    <bh:hasInput>
      <bhd:Payment rdf:ID="PowerBuyPayment"/>
    </bh:hasInput>
    <bh:hasEffect>
      <bhd:ProductDeliveredAfterReorder
          rdf:ID="PowerBuyProductDeliveredAfterReorder"/>
    </bh:hasEffect>
    <bh:hasOutput>
      <bhd:OrderedProductWithShippingFee
          rdf:ID="PowerBuyOrderedProductWithShippingFee">
        <bh:hasCondition rdf:resource="#PowerBuyValidLocationWithShippingFee"/>
      </bhd:OrderedProductWithShippingFee>
    </bh:hasOutput>
  </bhd:Sell>
  <bhd:ElectronicsApplianceBehaviouralProfile
         rdf:ID="PowerBuyBehaviouralProfile">
    <bh:hasOperation rdf:resource="#PowerBuySell"/>
  </bhd:ElectronicsApplianceBehaviouralProfile>
</rdf:RDF>
```

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Figure 3.6 Example of behavioural profile.

The profile specifies that PowerBuy provides operation Sell which requires CustomerInfo and Payment as inputs, gives OrderedProduct or OrderedProductWithShippingFee as outputs, and gives ProductDelivered or ProductDeliveredAfterReorder as effects. The precondition is the customer must hold ValidAcceptedCreditCard.

The output OrderedProductWithShippingFee is a conditional output because it is under a condition ValidLocationWithShippingFee. This means if the customer's location is in a certain area, shipping fee must be paid for the ordered product. This condition is associated with ValidShippingLocationWithShippingFee is the operation rule-based profile of PowerBuy. Such an association is specified in the behavioural profile (Figure 3.7). In a similar manner, the precondition ValidAcceptedCreditCard is associated with the condition AcceptedCreditCard defined in the operation rule-based profile of PowerBuy.

```
<owl:Class rdf:ID="ValidLocationWithShippingFee">
    <rdfs:subClassOf rdf:resource="#ValidLocation"/>
    <owl:sameAs rdf:resource="http://www.newregistry.com/
            electronicsapplianceRlonto#ValidShippingLocationWithShippingFee"/>
  </owl:Class>
  <owl:Class rdf:ID="ValidAcceptedCreditcard">
    <rdfs:subClassOf
       rdf:resource="http://www.newregistry.com/behaviouralonto#PreCondition"/>
    <owl:sameAs rdf:resource="http://www.newregistry.com/
           electronicsapplianceRlonto#AcceptedCreditcard"/>
  </owl:Class>
```

Figure 3.7 Example of associations of conditions in behavioural profile.

3.3.4 Operational Rule-Based Profile

Figure 3.8 shows part of the operational-rule based profile of PowerBuy service which is an instance of the operational rule-based ontology of the electronics appliance domain (in Figure 3.3 (i)).

```
<?xml version="1.0"?>
<rdf:RDF  xmlns="http://www.powerbuy.com/PowerbuyRlProfile#"
    xmlns:rld="http://www.newregistry.com/electronicsapplianceRlonto#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rl="http://www.newregistry.com/ruleonto#"
    xml:base="http://www.powerbuy.com/PowerbuyRlProfile">
  <rld:AcceptedCreditcard rdf:ID="PowerBuyAcceptedCreditcard"> <!--Behavioural Constraint-->
    <rl:hasOutputParameter>
      <rld:Boolean rdf:ID="PBBoolean"/>
    </rl:hasOutputParameter>
    <rl:hasInputParameter>
      <rld:CreditcardPayment rdf:ID="PowerBuyCreditcardPayment"/>
    </rl:hasInputParameter>
  </rld:AcceptedCreditcard>
  <rld:ValidShippingLocationWithShippingFee
     rdf:ID="PowerBuyValidShippingLocationWithShippingFee "> <!--Behavioural Constraint-->
    <rl:hasOutputParameter>
      <rld:Boolean rdf:ID="PBBoolean"/>
    </rl:hasOutputParameter>
    <rl:hasInputParameter>
      <rld:ShippingLocation rdf:ID="PowerBuyShippingLocation"/>
    </rl:hasInputParameter>
  </rld:ValidShippingLocationWithShippingFee>
<rld:CreditcardFeeChargePolicy rdf:ID="PowerBuyCreditcardFeeChargePolicy">
  <rl:hasServiceConstraint rdf:resource="#PowerBuyAcceptedCreditcard"/>
</rld:CreditcardFeeChargePolicy>
<rld:ProductShippingPolicy rdf:ID="PowerBuyProductShippingPolicy">
    <rl:hasServiceConstraint
            rdf:resource="#PowerBuyValidShippingLocationWithShippingFee"/>
    <rl:hasServiceConstraint>
       <rld:DeliveryDayShipping rdf:ID="PowerBuyDeliveryDayShipping">
          <rl:hasOutputParameter>
              <rld:DeliveryDay rdf:ID="PowerBuyDeliveryDay"/>
          </rl:hasOutputParameter>
          <rl:hasInputParameter>
              <rld:ShippingLocation rdf:ID="PowerBuyShippingLocation"/>
          </rl:hasInputParameter>
       </rld:DeliveryDayShipping>
    </rl:hasServiceConstraint>
    <rl:hasServiceConstraint>
      <rld:ServiceShippingLocation rdf:ID="PowerBuyServiceShippingLocation">
        <rl:hasOutputParameter rdf:resource ="#PowerBuyShippingLocation"/>
       </rld:ServiceShippingLocation>
    </rl:hasServiceConstraint>
    <rl:hasServiceConstraint>
      <rld:ShippingServiceCharge rdf:ID="PowerBuyShippingServiceCharge">
         <rl:hasOutputParameter>
           <rld:ServiceCharge rdf:ID="PowerBuyServiceCharge"/>
        </rl:hasOutputParameter>
         <rl:hasInputParameter rdf:resource="#PowerBuyShippingLocation"/>
      </rld:ShippingServiceCharge>
    </rl:hasServiceConstraint>
</rld:ProductShippingPolicy>
<rld:ElectronicsApplianceRuleProfile rdf:ID="PowerBuyRuleProfile">
    <rl:hasRule rdf:resource="#PowerBuyCreditcardFeeChargePolicy"/>
    <rl:hasRule rdf:resource="#PowerBuyProductShippingPolicy"/>
</rld:ElectronicsApplianceRuleProfile>
</rdf:RDF>
```

Figure 3.8 Example of operational rule-based profile.

The profile specifies AcceptedCreditCard which is the constraint of the precondition ValidAcceptedCreditCard in Figure 3.6., and ValidShippingLocationWithShippingFee which is the constraint for the conditional output OrderedProductWithShippingFee in Figure 3.6. The profile also specifies other constraints, i.e. DeliveryDayShipping, ServiceShippingLocation, ShippingServiceCharge. The constraints may specify inputs and/or outputs where necessary.

The profile here does not specifically instantiate the constraints; it does not specify, for example, that the accepted credit cards are Amex and Visa only, or that the shipping location that incurs a shipping fee is Europe. On the other hand, this profile serves as a template for the publisher of PowerBuy. The discovery framework will provide a GUI that is aware of the profile and allows the publisher to instantiate the constraints (see Chapter 6). In the case that PowerBuy specifies such instantiated constraints directly in the profile (e.g. by using SWRL), the constraints will have to be extracted and translated into a rule-based script (e.g. Jess script (Jess, 2003)) for constraint evaluation.

Ontology reasoning also plays a part in constraint evaluation. For example, if the customer's credit card is CitiBank visa card, ontology reasoning will infer that CitiBankVisa is Visa and therefore the input parameter CreditCardPayment of AcceptedCreditCard will instead become Visa, before being passed to constraint evaluation.

3.4 Using External Ontology in Ontology-Based Profiles

This work uses shared domain ontologies to describe service descriptions in a particular domain. Domain experts are responsible for defining domain ontologies for each type of the ontology-based service profiles. Domain experts may adopt well-defined external ontologies that are created by reputable organisations (such as DAML.org, CYC.com, etc.) when building shared domain ontologies.

We give two examples of how external ontologies can be utilised. In Section 3.4.1, an example of how a domain expert may use an external ontology when creating a shared domain ontology is presented. Another example of how to associate a service to an external service classification ontology is given in Section 3.4.2.

3.4.1 Using External Ontology in Shared Domain Ontology

A number of working groups have published ontologies as shared knowledge for public to adopt when describing information. Domain experts may use such ontologies when building shared domain ontologies. Figure 3.9 depicts an award ontology which is an OWL-based external ontology adapted from its DAML counterpart located at http://www.ai.sri.com/daml/ontologies/sri-basic/1-0/Awards.daml. This award ontology defines shared semantics for any awards in terms of properties such as date range, organisation or school, granting foundation, title etc. These properties are defined with domain and range of their values. A domain expert, when creating a shared domain ontology, may adopt only some properties and put more restrictions on those properties where appropriate.

```
xmlns:base = "http://www.ai.sri.com/owl/ontologies/sri-basic/1-0/Awards#"
<owl:ObjectProperty rdf:ID="date-range">
  <rdfs:domain rdf:resource="#Award"/>
  <rdfs:range rdf:resource="http://www.ai.sri.com/owl/.../Date.owl#Date-range"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID ="organisation-or-school">
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"/>
  <rdfs:domain rdf:resource="#Award"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="granting-foundation">
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"/>
  <rdfs:domain rdf:resource="#Award"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="title">
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral"/>
  <rdfs:domain rdf:resource="#Award"/>
</owl:DatatypeProperty>
<owl:Class rdf:ID="Award"/>
```

Figure 3.9 Part of external ontology specifying the award concept.

A domain expert may adopt this award ontology when defining a shared semantic attribute profile for the electronics appliance domain (c.f. Figure 3.3 (f)). In Figure 3.10, the concept ThailandElectronicsAssociationAward is defined as a subconcept of the concept

ElectronicsAwardValue and the concept Award in the external ontology. The concept ThailandElectronicsAssociationAward adopts only the properties title, date-range, and granting-foundation, each with a restriction on cardinality.

With well-defined external ontologies, domain ontologies and, as a result, ontology-based service profiles will be specified with common well-defined semantic structure, and discovery can benefit from semantics of external ontologies.

```
 xmlns:base="http://www.newregistry.com/electronicsapplianceSaonto#"

<owl:Class rdf:ID="ThailandElectronicsAssociationAward">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="ElectronicsAwardValue">
       <rdfs:subClassOf rdf:resource="http://www.newregistry.com/
                  semanticattronto.owl#SemanticValue"/>
    </owl:Class>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="http://www.ai.sri.com/owl/.../Awards#Award"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1
      </owl:cardinality>
      <owl:onProperty rdf:resource="http://www.ai.sri.com/owl/.../Awards#title"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1
      </owl:cardinality>
      <owl:onProperty
          rdf:resource="http://www.ai.sri.com/owl/.../Awards#date-range"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int">1
      </owl:cardinality>
      <owl:onProperty
        rdf:resource="http://www.ai.sri.com/owl/.../Awards#granting-foundation"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Figure 3.10 Part of semantic attribute ontology for electronics appliance domain adopting the award ontology.

3.4.2 Associating Capability Ontology with External Ontology

This example shows how a capability ontology of a service domain can be associated with an external ontology of service taxonomy. At present, industrial bodies, e.g. UNSPSC.org and NAICS.org, provide ontologies for classification of products and services.

These taxonomies have been adopted by standard UDDI for specifying business and service categories. It is possible to associate such external ontologies with a capability ontology for a particular domain (c.f. Section 3.2).

According to Figure 3.3 (a), Figure 3.11 shows a capability ontology of the electronics appliance domain. It specifies semantics attribute, structural, behavioural, and operational rule-based ontologies for the domain. Also, it links to the external ontology of UNSPSC classification located at http://www.daml.org/2004/05/unspsc/unspsc. In doing so, service discovery can be enhanced as a service consumer may query for services under a particular UNSPSC class. This query allows all services belonging to such a class as well as its subclasses to be discovered.

```
  xmlns:unspsc="http://www.daml.org/2004/05/unspsc/unspsc#"
  xmlns:base="http://www.newregistry.com/businessdomain#"

  <owl:Class  rdf:ID=”ServiceDomain”/>
  <owl:Class rdf:ID="ElectronicsAppliance">
    <rdfs:subClassOf rdf:resource="#ServiceDomain"/>
  </owl:Class>

 <ElectronicsAppliance rdf:ID="ElectronicsApplianceDomainDesc">
   <hasSemanticAttributeOntology
         rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
     http://www.newregistry.org/electronicsapplianceSaonto.owl
   </hasSemanticAttributeOntology>
   <hasStructuralOntology rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
     http://www.newregistry.org/electronicsapplianceStonto.owl
   </hasStructuralOntology>
   <hasBehaviouralOntology rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
     http://www.newregistry.org/electronicsapplianceBhonto.owl
   </hasBehaviouralOntology>
   <hasOperationalRule-BasedOntology
        rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
     http://www.newregistry.org/electronicsapplianceRlonto.owl
   </hasOperationalRule-BasedOntology>
   <relatedToUNSPSCService
        rdf:resource=" http://www.daml.org/2004/05/unspsc/
                    unspsc#ElectricalAccessoriesOrSuppliesManufactureServices"/>
   <relatedToUNSPSCProduct  rdf:resource="
        http://www.daml.org/2004/05/unspsc/unspsc#ConsumerElectronics"/>

 </ElectronicsAppliance>
```

Figure 3.11 Capability ontology of electronics appliance domain.

3.5 Semantic Query

A service consumer will submit a query to the semantic matchmaker through the provided GUI (see Chapter 6). The query will be translated into an XML-based profile which links to RDF resources defined in the upper ontologies and domain ontologies. This XML-based query profile will be parsed to match against the providers' profiles. Domain experts will define an ontology for a domain which specifies a schema for the query within the domain.

The query can be represented as a set of relation expressions which extend from the triple <*subject*, *property*, *object*> of RDF expressions (Costello and Jacobs, 2003). Generally, *subject* refers to a particular aspect of the service to be queried, *property* refers to a particular property of such aspect that is of interest, and *object* refers to the value of the property. In this research, *object* may be simple lexical value or ontological value and may or may not have an associated constraint. In summary, each relation expression is in one of the following forms.

- Relation expression for single property value. This relation expression is in the form *property(subject, object)* where *object* is a single literal or ontological value. For example, a query for a service whose simple attribute Description has keywords "electronics, retail" can be specified as hasDescription(si:Description, "electronics, retail") where si:Description is a simple attribute in the simple attribute ontology whose value is linked to a resource that represents simple attribute value in the extended UDDI information model. This form of relation expression is applicable to a query on the simple attribute, semantic attribute, structural, behavioural, and operational rule-based profiles.
- Relation expression for multiple ontological values. This relation expression is in the form *property(subject, list of objects)* where *object* is an ontological value. For example, a query for a service that provides Product Desktop and Laptop can be specified as hasProduct(st:Product, std:Desktop, std:Laptop). This form

of relation expression is applicable to a query on the semantic attribute and structural profiles.

- Relation expression for numerical constraint. This relation expression is in the form *property(subject, object, logicaloperator, literalvalue1, [literalvalue2], unit)* where *object* is associated with the numerical constraint that follows. For example, a query for a service that provides Desktop product with Price, ranging between 15,000 and 30,000 bahts, can be specified as hasPrice(std:Desktop, std:Price, *Between*, 15000, 30000, baht), where std:Desktop is defined in the structural domain ontology. This form of relation expression is applicable to a query on the structural and operational rule-based profiles.
- Relation expression for numerical constraint with parameters. This relation expression is in the form *property(subject, object, list of parameter values, logicaloperator, literalvalue1, [literalvalue2], unit)* where *object* is associated with the numerical constraint that follows based on some *parameter values*. For example, a query for a service whose DeliveryDay is constrained to be less than or equal to 3 days only if delivery is in Bangkok can be specified as hasDeliveryDayShipping(rld:DeliveryDayShipping, rld:DeliveryDay, Bangkok, *LessThanOrEqual*, 3, day), where rld:DeliveryDayShipping is a service constraint defined in the operational rule-based ontology of the domain. This form of relation expression is applicable to a query on the operational rule-based profile.

Figure 3.12 presents an example of XML-based query for a service that advertises its description with keywords “electronics” and “retail”. The service receives an award from Thailand Electronics Association and also sells desktops and laptops. The advertised price of a desktop is between 15,000 and 30,000 bahts. In addition, the service can provide shipping of the product to Bangkok area within 3 days and accept American Express credit card for purchasing.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

```
<qr:serviceQuery  xmlns:qr="http://www.newregistry.com/query.xml"
   xmlns:qrd="http://www.newregistry.com/electronicsappliancequery.owl#"
   xmlns:bh="http://www.newregistry.com/behaviouralonto.owl#"
   xmlns:bhd="http://www.newregistry.com/electronicsapplianceBhonto.owl#"
   xmlns:sa="http://www.newregistry.com/semanticattronto.owl#"
   xmlns:sad="http://www.newregistry.com/electronicsapplianceSaonto.owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns="http://www.newregistry.com/userquery.xml#"
   xmlns:st="http://www.newregistry.com/structuralonto.owl#"
   xmlns:std="http://www.newregistry.com/electronicsapplianceStonto.owl#"
   xmlns:rl="http://www.newregistry.com/ruleonto.owl#"
   xmlns:rld="http://www.newregistry.com/electronicsapplianceRlonto.owl#"
   xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xml:base="http://www.newregistry.com/userquery.xml">
      <qr:relationExpression    ID= “relationexpression1”
              typeExp= “QSingPropValue”>
             <qr:property>si:hasDescription</qr:property>
             <qr:subject>si:Description</qr:subject>
             <qr:object>electronics, retail</qr:object>
      </qr:relationExpression>
      <qr:relationExpression    ID= “relationexpression2”
              typeExp= “QSingPropValue”>
             <qr:property>sa:hasAward</qr:property>
             <qr:subject>sa:Award</qr:subject>
             <qr:object>sa:ThailandElectronicsAssociationAward</qr:object>
      </qr:relationExpression>
      <qr:relationExpression    ID= “relationexpression3”
              typeExp= “QMultiOntoValue”>
             <qr:property>st:hasProduct</qr:property>
             <qr:subject>std:ElectronicsApplianceProductDetails</qr:subject>
             <qr:object>std:Desktop, std:Laptop</qr:object>
      </qr:relationExpression>
      <qr:relationExpression    ID= “relationexpression4”
              typeExp= “QNumConst”>
             <qr:property>st:hasPrice</qr:property>
             <qr:subject>std:Desktop</qr:subject>
             <qr:object>std:DesktopPrice</qr:object>
             <qr:constObj  logicalOp= “st:Between”  literalVal1 = “15000”
                           literalVal2 = “30000” unit = “st:Baht” />
      </qr:relationExpression>
      <qr:relationExpression    ID= “relationexpression5”
              typeExp= “QNumConstParam”>
             <qr:property>qrd:hasDeliveryDayShipping</qr:property>
             <qr:subject>rld:DeliveryDayShipping </qr:subject>
             <qr:object>rld:DeliveryDayConst</qr:object>
              <qr:parametervalues>rld:Bangkok</qr:parametervalues>
             <qr:constObj  logicalOp= “rl:LessThanOrEqual” literalVal1 = “3”
                           unit = “rl:Day” />
      </qr:relationExpression>
      <qr:relationExpression    ID= “relationexpression6”
              typeExp= “QSingPropValue”>
             <qr:property>qrd:acceptCreditcard</qr:property>
             <qr:subject>bh:Operation</qr:subject>
             <qr:object>bhd:AmericanExpress</qr:object>
      </qr:relationExpression>
</qr:serviceQuery>
```

Figure 3.12 Semantic query profile.

CHAPTER IV

MATCHMAKING AND RANKING OF SEMANTIC WEB SERVICES

4.1 Matching Criteria for Integrated Service Profile

This section explains matching criteria for determining whether an integrated service profile of a provider matches the query of a consumer. Matching is based on the comparison between two relation expressions, one in a particular profile of the provider and the other in the query. The provider's profiles and the query can be seen as a collection of these relation expressions as mentioned in Chapter III.

The following Sections 4.1.1-4.1.6 explain matching criteria which consider all aspects of the integrated service profile. Results from these will lead to a classification of matching types and assignment of their ordinal scale in Section 4.1.7. It is assumed that any services that do not publish any aspects requested in the query will not be considered in the matching process.

4.1.1 Matching Ontological Concepts

As most profiles (except for the simple attribute profile) are ontology-based, matching by subsumption and equivalence is the basis for matching ontological concepts in the query and the provider's profile (Baader et al., 2003). This approach has been adopted in (Sycara et al., 2002; Paolucci et al., 2002; Trastour et al., 2002; Li and Horrocks, 2003; Di Noia et al., 2003). In (Resnik, 1995; Andreasen et al., 2003), a weaker match of ontological concepts, called partial match, is defined for two concepts that have a shared node in IS-A taxonomy and do not have a subsumption relationship between them. The degree of matching is determined between two concepts as described below.

For two relation expressions of the same property, one in the query and the other in the provider's profile, let C_Q be the property value specified in the query and C_P be the one in the profile:

(i) *If $C_Q \equiv C_P$ then C_P is an exact match for C_Q*, where $\equiv$ means is equivalent to. For example from Figure 4.1, the provider who sells Desktop will be an exact match for the query that also requests for Desktop.

(ii) *If $C_P \sqsubseteq C_Q$ then C_P is a specialised match for C_Q*, where $\sqsubseteq$ means is subsumed by (i.e. C_P is more specific than C_Q). In this case, the query may specify a generic concept while the profile defines a specific concept. For example from Figure 4.1, the profile that sells either Notebook or Desktop will be a specialised match for the query that requests for PC.

(iii) *If $C_Q \sqsubseteq C_P$ then C_P is a generalised match for C_Q.* This means the concept in the query is more specific than, and is subsumed by, the one in the profile. For example from Figure 4.1, the profile that sells PC will be a generalised match for the query that requests for a Desktop.

(iv) *If $(C_Q \not\sqsubseteq C_P) \wedge (C_P \not\sqsubseteq C_Q) \wedge (C_Q \sqsubseteq C_C) \wedge (C_P \sqsubseteq C_C)$ then C_P is a partial match for C_Q*, where $\not\sqsubseteq$ means is not subsumed by and C_C is a node in the same IS-A taxonomy. This means it is acceptable for the concept in the profile to be a match for the concept in the query provided that the two concepts have common characteristics through a common parent concept. For example from Figure 4.1, the profile that sells Laptop will be partial match for the query that requests for Desktop.

(v) *If none of the above relationships exist then C_P is a failed match for C_Q.*

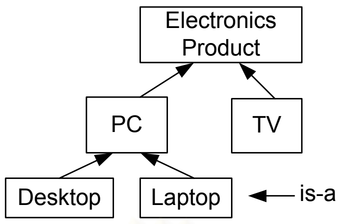


Figure 4.1 Fragment of ontology of the domain.

4.1.2 Matching Numerical Constraints

As mentioned earlier, service providers and consumers may put numerical constraints on relation expressions in the context of the structural or operational rule-based ontologies. For example, the relation expression on the price of the product PC may be published or queried with such a constraint that the price is between 30,000 and 50,000 bahts. Or the relation expression on the rule for the number of delivery day for shipping may be published or queried with a constraint that it is less than 3 days. Matching two numerical constraints compares the intervals of the possible values that are defined in the constraints. The degree of matching for numerical constraints can be determined as described below.

For two relation expressions of the same property, let N_Q be a nonempty set of numerical constraint values of the relation expression in the query (R_Q), and N_P be a nonempty set of numerical constraint values of the relation expression in the profile (R_P):

(i) *If* $N_P \subseteq N_Q$ *then* R_P *is an exact match for* R_Q.

(ii) *If* $N_Q \subseteq N_P$ *then* R_P *is a plug-in match for* R_Q.

(iii) *If* $(N_P \cap N_Q \neq \phi) \wedge (N_P \subsetneq N_Q) \wedge (N_Q \subsetneq N_P)$ *then* R_P *is a weak match for* R_Q.

(iv) *If* $N_P \cap N_Q = \phi$ *then* R_P *is a failed match for* R_Q.

4.1.3 Matching Sets of Ontological Values

Service providers or consumers may publish or query multiple ontological values on any relation expressions related to the semantic attribute and structural ontologies. For example, the relation expression on the award that the service has obtained may be published or queried with both values ThailandElectronicsAssociationAward and ThailandMagazineAward. Or the relation expression on the product for sale by the service may be both Desktop and TV. Matching two sets of ontological values comes down to matching each of the values in the two sets based on ontological matching (Constantinescu and Faltings, 2003).

For two relation expressions of the same property, let D_Q be a nonempty set of ontological values of the relation expression in the query (R_Q), and D_P be a nonempty set of ontological values of the relation expression in the profile (R_P):

Definition. The profile will satisfy a set of ontological values match on the query if there exists an ontological match (Section 4.1.1) between each concept in the query and a concept in the profile. This is denoted by

$$SetOfOntoValsMatch(R_Q, R_P) = true \Leftrightarrow$$

$$\forall i,\ \exists j: (i \in D_Q) \wedge (j \in D_P) \wedge (i \otimes j).$$

where $\otimes$ means having a kind of the ontological match in Section 4.1.1 (i.e. *exact*, *specialised*, *generalised*, *partial*).

4.1.4 Matching Service Constraints

Service providers or consumers may publish or query on the values of service constraints in the operational rule-based ontology. There are two cases for considering matching between two sets of constraint values: matching operational constraints and matching behavioural constraints. For example, the operational constraint ServiceShippingLocation of a service may be published to return the concepts Bangkok, Chiang Mai, and Phuket on evaluation. If the query is for the service that provides shipping to Bangkok and Chiang Mai, the service would match. The case of behavioural constraints is more complex as they are conditions that are associated with the behavioural profile (i.e. precondition, conditional output, conditional effect). So matching of behavioural constraints will be used for determining matching of precondition, conditional output, and conditional effect. Although behavioural constraints require input parameters to evaluate to either true or false, the behavioural profile concerns only when they evaluate to true, by which the precondition will hold and the conditional output and conditional effect will result. For example, in Figure 3.3 (h), the conditional output OrderedProductWithShippingFee will result only if the condition ValidLocationWithShippingFee is true. Since ValidLocationWithShippingFee is an equivalentClass to the behavioural constraint ValidShippingLocationWithShippingFee, we first evaluate this behavioural constraint by using the location (specified in the query) as the input parameter (say, Bangkok and Chiang Mai). If the location is among the valid values, defined by the service, for this constraint (e.g. Bangkok, Chiang Mai, Phuket), it will evaluate to true which means the equivalent ValidLocationWithShippingFee is also true, and the result is the output OrderedProductWithShippingFee will be produced. In other words, the output OrderedProductWithShippingFee of this service satisfies the query based on the location input from the query. For behavioural constraints, matching is therefore considered against the values of the input parameter of the constraint.

Since the values related to the evaluation of a service constraint may in fact be either a range of numerical values or a set of ontological concepts, we can adopt the matching rules in Sections 4.1.2 and 4.1.3 here. For two relation expressions of the same property, let O_Q be a nonempty set of ontological outputs or of ranges of numerical output for an operational constraint specified in the query (R_Q), O_P be a nonempty set of ontological outputs or of ranges of numerical output for an operational constraint in the profile (R_P), I_Q

be a nonempty set of ontological inputs or of ranges of numerical input for a behavioural constraint specified in the query (R_Q) where the constraint is evaluated to true, and I_P be a nonempty set of ontological inputs or of ranges of numerical input for a behavioural constraint specified in the profile (R_P) where the constraint is evaluated to true:

Definition. The profile will satisfy a set of constraints match on the query if, depending on whether the constraints are operational or behavioural, the output or input for each of the constraint evaluation of the query matches one in the profile. This is determined by

(i) SetOfOperationalConstrsMatch(R_Q, R_P) = true $\Leftrightarrow$

$$\forall i,\ \exists j: (i \in O_Q) \wedge (j \in O_P) \wedge (i \odot j)$$

(ii) SetOfBehaviouralConstrsMatch(R_Q, R_P) = true $\Leftrightarrow$

$$\forall i,\ \exists j: (i \in I_Q) \wedge (j \in I_P) \wedge (i \odot j)$$

where $\odot$ means either having a kind of the ontological match in Section 4.1.1 (i.e. *exact*, *specialised*, *generalised*, *partial*) or having a kind of the numerical constraint match in Section 4.1.2 (i.e. *exact*, *plug-in*, *weak*).

4.1.5 Matching Behavioural Profiles

Matching behavioural profiles determines whether the behavioural capability of the service can satisfy or realise the behavioural requirement in the query (Liskov and Wing, 1994; Zaremski and Wing, 1997; Wickler, 1999). Intuitively, the service will satisfy the query if, given the precondition and input from the query, the service can accept and operate successfully, giving out satisfied output or effect to the query. Ontological matching in Section 4.1.1 is used to determine matching for each relation expression in the behavioural profile as follows:

(i) *Operation, Precondition, Output, and Effect Match*. An ontological concept signifying either an operation, precondition, output, or effect in the profile will match to its counterpart in the query if they have a kind of ontological match in Section 4.1.1 (i.e. *exact*, *specialised*, *generalised*, *partial*).

(ii) *Input Match.* Ontological match in Section 4.1.1 is also used to determine matching between one input concept in the profile and another in the query. It is interesting to note that, unlike other aspects, *generalised* match is a better match than *specialised* match in the case of input. This means the service's operation can perfectly operate with the query's input which is more specific than what it expects. This is compared to the case when the service's operation expects a more specific input than what supplied by the query.

Let $\mathbb{R}_Q$ and $\mathbb{R}_P$ be sets of behavioural relation expressions which comprise an operation, a set of inputs, a set of outputs, a set of preconditions, and a set of effects.

Definition. The profile will satisfy a behavioural match on the query if all the behaviour expected by the query is among the behaviour that the profile exhibits. This is determined by

$$\textit{BehaviouralMatch}(\mathbb{R}_Q, \mathbb{R}_P) = \text{true} \Leftrightarrow (\mathbb{R}_Q \subseteq \mathbb{R}_P) \wedge$$
$$(\forall i, \exists j : (i \in \mathbb{R}_Q) \wedge (j \in \mathbb{R}_P) \wedge (i \otimes j))$$

where $\otimes$ means having a kind of the ontological match in Section 4.1.1 (i.e. *exact*, *specialised*, *generalised*, *partial*).

Note that the service may publish more behavioural information than the query. For example, the service may require more number of inputs than those in the query. We do not consider this number issue in the matching process; as long as the service has the inputs that can match to the query's inputs, the service satisfies the query in that respect. Suppose that finally the consumer selects to use this service, the consumer can study from the WSDL of the service to find out what more inputs are needed.

Since a behavioural profile contains relation expressions that relate to many aspects of the behavioural model, we then have to determine matching for all of associated semantic elements. In the case that semantic elements are preconditions, outputs, or effects with associated behavioural constraints in the operational rule-based ontology, they will match only if they can also satisfy behavioural constraint match as mentioned in Section 4.1.4.

4.1.6 Matching Simple Attributes

Matching of simple attributes in the simple attribute profile is based on comparing descriptive string values of the attributes and determining their similarity. We adopt an approximate string matching technique called *q*-grams (Ukkonen, 1992; Gravano et al., 2001; Navarro, 2001). The basic idea of q-grams is *sliding* a window of length *q* over the characters of string σ. To achieve a better comparison, words with no information value are removed from the descriptive string before processing. It is also possible to provide a list of keywords for a particular domain to help specifying attribute values when publishing or querying. Matching descriptive attribute values can be implemented by extracting terms, which are likely to match the listed keywords, from the profile and the query. Extraction can be implemented by substring match, and later use *q*-grams for computing similarity.

Let *q* be length of *q*-grams, σ be a set of *n* keyword terms extracted from the value of a simple attribute in the query, a set G_{σ_i} be *q*-grams of a string σ_i where $\sigma_i \in \sigma$, Ω be a set of *m* keyword terms extracted from the value of the same simple attribute in the profile, and a set G_{Ω_j} be *q*-grams of string Ω_j where $\Omega_j \in \Omega$. The similarity score between σ and Ω is computed by

$$SimSimpleAttribute\ (\sigma, \Omega) = \left|(\bigcup_{i=1}^{n} G_{\sigma_i}) \cap (\bigcup_{j=1}^{m} G_{\Omega_j})\right| / \left|(\bigcup_{i=1}^{n} G_{\sigma_i})\right|$$

For example, given σ = {electronics, retail}, Ω = {retail}, and length *q* is 3, *q*-grams of the string "electronics" is {##e,#el,ele,lec,ect,ctr,tro,ron,oni,nic,ics,cs#,s##}, and *q*-grams of the string "retail" is {##r,#re,ret,eta,tai,ail,il#,l##}. Therefore the similarity score between σ and Ω is 0.38. The similarity score helps classify the types of simple attribute matching such as: *StrongMatch* [0.75, 1], *OptimisticMatch* [0.50 - 0.75), *RelaxedMatch* [0.25, 0.50), and *Fail*ed[0, 0.25).

4.1.7 Ordinal Scale of Profile Matching

Table 4.1 summarises the classification of matching and ordinal scale which represents match scores, from the strongest to the weakest match.

Table 4.1 Types of matching and match scores

Classification of matching	Match score
Ontological match (Section 4.1.1)	exact = 4, specialised = 3, generalised = 2, partial = 1, failed = 0
Input match of behavioural profile (Section 4.1.5)	exact = 4, generalised = 3, specialised = 2, partial = 1, failed = 0
Numerical constraint match (Sections 4.1.2 and 4.1.4 (for numerical values))	exact = 3, plug-in = 2, weak = 1, failed = 0
Set of values match (Sections 4.1.3 and 4.1.4 (for ontological values))	satisfied = 1, failed = 0
Simple attribute match (Section 4.1.6)	strong = 3, optimistic = 2, relaxed = 1, failed = 0

4.2 Example of Matchmaking

Suppose there is a query for a Web Service of a retail electronics shop which sells desktop computers with the price range between 15,000 – 30,000 bahts. The shop must receive Thailand Electronics Association Award and accept credit card payment. The consumer also needs the computer to be delivered to Bangkok within 3 days. Fee charge for delivery is acceptable but should be less than 200 bahts. We present this query ($\mathbb{Q}$) with the relation expressions below. Note that each expression is subscripted by a profile symbol; α, γ, ρ , and β represents simple or semantic attribute profile, structural profile, operational rule-based profile, and behavioural profile respectively. The superscript denotes the context of the relation expression; *S* refers to a simple attribute, *C* refers to a constraint which may be either a numerical, behavioural, or operational constraint, and ϕ refers to a single concept. For the behavioural profile, the superscripts Δ, *I, O, P, E* respectively refer to operation, input, output, precondition, and effect. The question marks indicate requirements that the service consumer specifies in the query:

$\mathbb{Q}$ = {hasDescription(Description, ?"electronics, retail"$)_{\alpha}^{S}$,

hasAward(Award, ?ThailandElectronicsAssociationAward$)_{\alpha}^{\phi}$,

hasProduct(ProductDetails, ?Desktop$)_{\gamma}^{\phi}$,

hasPrice(Desktop, Price, ?*Between*, ?15000, ?30000, ?baht$)_{\gamma}^{C}$,

hasDeliveryDayShipping(DeliveryDayShipping, DeliveryDay, ?Bangkok, ?*LessThanOrEqual*, ?3, ?day$)_{\rho}^{C}$,

hasShippingServiceCharge(ShippingServiceCharge, ServiceCharge, ?Bangkok, ?*LessThanOrEqual*, ?200, ?baht$)_{\rho}^{C}$,

hasOperation(BehaviouralProfile, ?SellElectronicsProduct$)_{\beta}^{\Delta}$,

hasInput(SellElectronicsProduct, ?CreditcardPayment$)_{\beta}^{I}$, hasPrecondition(SellElectronicsProduct, ?ValidAcceptedCreditcard$)_{\beta}^{P}$,

hasOutput(SellElectronicsProduct, ?OrderedProduct$)_{\beta}^{O}$,

hasEffect(SellElectronicsProduct, ?ProductDelivered$)_{\beta}^{E}$ }

The integrated service profiles of two candidate services S_1 and S_2 are in Figure 4.2. These are instance profiles, so ontological matching is considered from the base concept of each individual resource. Some IS-A hierarchies that represent knowledge in the profile ontologies of the ElectronicsAppliance domain are shown in Figure 4.3. Note that it is possible that a single concept in the query may match to multiple concepts in the profile. For example, the query that asks for a product Desktop could match to both product Desktop and Laptop which are published in the profile, but with a different strength (i.e. exact vs. partial match). We consider the strongest match in this case.

By comparing $\mathbb{Q}$ against S_1 and S_2 and assuming that any evaluation required to evaluate the behavioural constraints of the two services are valid, matching results between $\mathbb{Q}$ and S_1, and $\mathbb{Q}$ and S_2 are reported as follows:

Match($\mathbb{Q}$, S_1) = {$\mathit{strong}_{hasDescription}$, $\mathit{exact}_{hasAward}$, $\mathit{exact}_{hasProduct}$, $\mathit{plug\text{-}in}_{hasPrice}$, $\mathit{exact}_{hasDeliveryDayShipping}$, $\mathit{exact}_{hasShippingServiceCharge}$, $\mathit{exact}_{hasOperation}$, $\mathit{exact}_{hasPrecondition}$, $\mathit{exact}_{hasInput}$, $\mathit{partial}_{hasOutput}$, $\mathit{exact}_{hasEffect}$}

Match($\mathbb{Q}$, S_2) = {$\mathit{relaxed}_{hasDescription}$, $\mathit{partial}_{hasAward}$, $\mathit{exact}_{hasProduct}$, $\mathit{plug\text{-}in}_{hasPrice}$, $\mathit{plug\text{-}in}_{hasDeliveryDayShipping}$, $\mathit{exact}_{hasShippingServiceCharge}$, $\mathit{specialised}_{hasOperation}$, $\mathit{partial}_{hasPrecondition}$, $\mathit{partial}_{hasInput}$, $\mathit{partial}_{hasOutput}$, $\mathit{partial}_{hasEffect}$}

```
xml:stl="http://www.powerbuy.com/PowerbuyStProfile">
xml:bhl="http://www.powerbuy.com/PowerbuyBhProfile">
xml:rll="http://www.powerbuy.com/PowerbuyRlProfile">
$S_1$ = {si:hasDescription(si:Description, uddi:Detail="In 2003, PowerBuy
is the first retail electronics store in Thailand—our goal was to provide the
community with a new technology that would change the face of consumer
electronics forever.")
sa:hasAward(sa:Award, sa:ThailandElectronicsAssociationAward)
st:hasProduct(st:ProductDetails, stl:PowerBuyDesktop)
st:hasProduct(st:ProductDetails, stl:PowerBuyLaptop)
st:hasProduct(st:ProductDetails, stl:PowerBuyTV)
st:hasPrice(stl:PowerBuyDesktop, stl:PowerBuyDesktopPrice)
stl:PowerBuyDesktopPrice(
  st:hasNumericalConstraint(
     st:hasOperator(st:GreaterThanOrEqual)
     st:hasLiteralVar1(15000)
     st:hasUnit(st:baht)   ))
rl:hasServiceConstraint(rll:PowerBuyProductShippingPolicy,
  rll:PowerBuyShippingServiceCharge)
  rll:PowerBuyShippingServiceCharge(
    rl:hasInputParameter(rll:PowerBuyShippingLocation)
    rl:hasOutputParameter(rll:PowerBuyServiceCharge) )
   Constraint1: If rll:PowerBuyShippingLocation=rld:Bangkok Then
                rll:PowerBuyServiceCharge=100 baht
   Constraint2: If rll:PowerBuyShippingLocation=rld:Chiangmai Then
                rll:PowerBuyServiceCharge=500 baht)
       Note: Constraints are defined in database.
rl:hasServiceConstraint(rll:PowerBuyProductShippingPolicy,
     rll:PowerBuyDeliveryDayShipping)
  rll:PowerBuyDeliveryDayShipping (
    rl:hasInputParameter(rll:PowerBuyShippingLocation)
    rl:hasOutputParameter(rll:PowerBuyDeliveryDay))
   Constraint1: If rll:PowerBuyShippingLocation=rld:Bangkok Then
                rll:PowerBuyDeliveryDay<=2 day
   Constraint2: ….
       Note: Constraints are defined in database.
bh:hasOperation(bh:BehaviouralProfile,
   bhl:PowerBuySellElectronicsProduct)
   bhl:PowerBuySellElectronicsProduct (
    bh:hasPrecondition(bhl:PowerBuyValidAcceptedCreditcard)
    bh:hasInput(bhl:PowerBuyCreditcardPayment)
    bh:hasOutput(bhl:PowerBuyOrderedProductWithoutShippingFee)
      bhl:PowerBuyOrderedProductWithoutShippingFee(
      bh:hasCondition(bhl:PowerBuyValidLocationWithoutShippingFee))
    bh:hasOutput(bhl:PowerBuyOrderedProductWithShippingFee)
      bhl:PowerBuyOrderedProductWithShippingFee(
         bh:hasCondition(bhl:PowerBuyValidLocationWithShippingFee))
    bh:hasEffect(bhl:PowerBuyProductDelivered))
}
```

```
xml:stl="http://www.comworld.com/ComWorldStProfile">
xml:bhl="http://www. comworld.com/ComWorldBhProfile">
xml:rll="http://www. comworld .com/ComWorldRlProfile">
$S_2$ ={si:hasDescription(si:Description, uddi:Detail="Comworld Co., Inc is
Thailand's number-one specialty retailer of personal computers and entertainment
software and appliances.")
sa:hasAward(sa:Award, sa:ThailandMagazineAward)
st:hasProduct(st:ProductDetails, stl:ComWorldDesktop)
st:hasPrice(stl:ComWorldDesktop, stl:ComWorldDesktopPrice)
  stl: ComWorldDesktopPrice(
     st:hasNumericalConstraint(
       st:hasOperator(st:GreaterThanOrEqual)
       st:hasLiteralVar1(10000)
       st:hasUnit(st:baht)   ))
rl:hasServiceConstraint(rll:ComWorldProductShippingPolicy,
  rll: ComWorldShippingServiceCharge)
  rll: ComWorldShippingServiceCharge(
     rl:hasInputParameter(rll:ComWorldShippingLocation)
     rl:hasOutputParameter(rll:ComWorldServiceCharge))
     Constraint1: If rll:ComWorldShippingLocation=rld:Bangkok Then
                  rll:ComWorldServiceCharge=150 baht
     Constraint2: If rll:ComWorldShippingLocation=rld:Phuket Then
                  rll:ComWorldServiceCharge=500 baht
       Note: Constraints are defined in database.
rl:hasServiceConstraint(rll:ComWorldProductShippingPolicy,
  rll:ComWorldDeliveryDayShipping)
   rll:ComWorldDeliveryDayShipping(
     rl:hasInputParameter(rll:ComWorldShippingLocation)
     rl:hasOutputParameter(rll:ComWorldDeliveryDay))
     Constraint1: If rll:ComWorldShippingLocation=rld:Bangkok Then
                 rll:ComWorldDeliveryDay<=5 day
       Note: Constraints are defined in database.
bh:hasOperation(bh:BehaviouralProfile,
   bhl:ComWorldSellComputer&HardwareProduct)
   bhl:ComWorldSellComputer&HardwareProduct (
     bh:hasPrecondition(bhl:ComWorldValidECreditPayment)
     bh:hasInput(bhl:ComWorldECreditPayment)
     bh:hasOutput(bhl:ComWorldOrderedProductWithoutShippingFee)
      bhl:ComWorldOrderedProductWithoutShippingFee(
       bh:hasCondition(bhl:ComWorldValidLocationWithoutShippingFee))
     bh:hasOutput(bhl:ComWorldOrderedProductWithShippingFee(
        bh:hasCondition(bhl:ComWorldValidLocationWithShippingFee))
     bh:hasEffect(bhl:ComWorldProductDeliveredFromStock)
        bhl:ComWorldProductDeliveredFromStock (
           bh:hasCondition(bhl:ComWorldLocationWithinStockArea))
     bh:hasEffect(bhl:ComWorldProductDeliveredByPartner)
           bhl:ComWorldProductDeliveredByPartner (
              bh:hasCondition(bhl:ComWorldLocationOutOfStockArea))
}
```

Figure 4.2 Integrated service profiles of two candidate services.

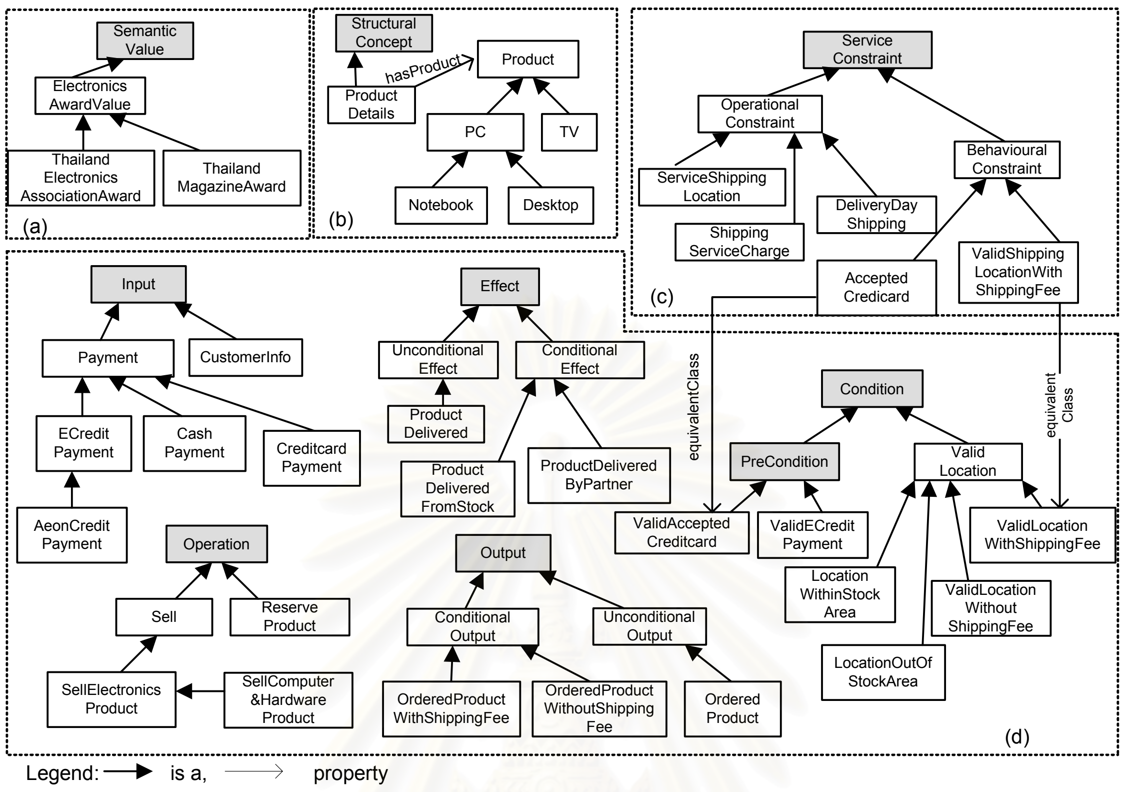


Figure 4.3 Fragment of ontologies for the profiles.

In our matchmaking process, Web Services that cannot fulfill any of the relation expressions within the query will be filtered out. In other words, only the Web Services that match with all the relation expressions within the query, regardless of the strength of matching (i.e. the ordinal scale), will be considered as matched services. These services may also have more of other capabilities that are not of the consumer's concern. In the following section, service consumers may refine search results by specifying a match preference.

4.3 Ranking Methodology

After the matchmaking process discovers all the Web Services whose characteristics and capabilities match to what expected by the query, the ordinal scale of match types in Section 4.1.7 is used by the ranking process to rank all those matched services based on user preference criteria (e.g. (Larichev, 2001)). Service consumers can specify any of the following preference criteria for matching and ranking:

(a) *Match Preference*. This criterion can be set to define a preference when considering matching on a particular relation expression. The preference is specified in terms of the weakest acceptable match type. For example, the service consumer may query for the product PC and set a match preference to *specialised*. So the services that publish the product with the same concept, i.e. PC (by *exact* match), and more specific concepts, i.e. Desktop and Laptop (by *specialised* match), will match to the query.

(b) *Feature Priority Preference*. This criterion can be set to define a significance that one relation expression has over the others within the same profile. This preference setting can help overcome a problem of conflicting ordinal scale of match types among several relation expressions. For example, the query specifies two relation expressions on Award and Description in the context of attributes. Suppose two candidate services have ordinal scale match as (*specialised*, *strong*) and (*exact*, *optimistic*) respectively, this can be problematic for ranking. By specifying a feature preference such that the consumer gives priority to Award over Description, the second candidate service will be ranked higher than the first one.

(c) *Profile Priority Preference*. This criterion can be set to define a significance that one profile of the service has over the others. It can be used in a similar way to the feature priority preference but is for problematic ranking across profiles. For example, the query specifies a relation expression on Award in the semantic attribute profile and another on Product in the structural profile. Suppose two candidate services have ordinal scale match as (*specialised*, *exact*) and (*exact*, *specialised*), this can be problematic for ranking. By specifying a profile priority preference such that the consumer gives priority to the semantic attribute profile over the structural profile (denoted by $\gamma \prec \alpha$), the second candidate service will be ranked higher than the first one.

Similarly to resolving ordinal scale conflict, feature priority preference and profile priority preference can be used to refine ranking. When two services have the same ranking order and a priority preference is set, the priority can be used to break the tie by further determining *sub-ranking*. In other words, sub-ranking is only for a more refined ordering within the same rank order with a similar match score, and it will not be considered if the priority preference is not set. In such a case, the services will be assumed to be ranked as equal; further ranking consideration, should the need arises, is left to the consumer.

Some parts of the example in Section 4.2 are taken to show how ranking is applied. Assume that the query now consists of the following six relation expressions: hasDescription, hasProduct, hasDeliveryDayShipping, hasOperation, hasOutput, and hasEffect. The service consumer sets a match preference on each relation expression as ($relaxed_{hasDescription}$, $exact_{hasProduct}$, $plug\text{-}in_{hasDeliveryDayShipping}$, $specialised_{hasOperation}$, $partial_{hasOutput}$, $partial_{hasEffect}$). The feature priority preference is set for the behavioural profile and specifies the priority, from the least to the most, as (effect, output, operation), which is denoted by $E \prec O \prec \Delta$. The profile priority preference is set to give equal priority to the structural, operational rule-based, and behavioural profiles, and these three has a priority over the attribute profiles (denoted by $\alpha \prec (\gamma \approx \rho \approx \beta)$).

Ranking methodology consists of the following steps:

(i) For each relation expression in the query, determine possible match scores for each of them with regards to their match preference. For six relation expressions of the query $\mathbb{Q}$ that we now focus, the relation expression hasDescription requires simple attribute matching with a match preference set to *relaxed*. Therefore its possible matches, from the strongest to the weakest, according to Section 4.7 are {*strong*, *optimistic*, *relaxed*}. This corresponds to the $MatchScores_{hasDescription}$ = {3, 2, 1}. Possible match scores for other relation expressions will be determined in a similar manner. Hence, $MatchScores_{hasProduct}$ = {4}, $MatchScores_{hasDeliveryDayShipping}$ = {3, 2}, $MatchScores_{hasOperation}$ = {4, 3}, $MatchScores_{hasOutput}$ = {4, 3, 2, 1}, and $MatchScores_{hasEffect}$ = {4, 3, 2, 1}.

(ii) For each profile with several relation expressions related to it, consider as follows:

- Define all possible match patterns for the profile. A match pattern is an n-tuple of the match scores from all related relation expressions which is denoted by

$MatchPattern = (ms_1, \ldots , ms_n)$

where n *= the number of related relation expressions*

ms_i *= a match score value taken from* $MatchScores_i$, $i = 1..n$.

The *MatchPatterns* for the behavioural profile in the example will be (4,4,4), (4,4,3), (4,4,2), (4,4,1), (4,3,4), (4,3,3), (4,3,2), (4,3,1), (4,2,4) etc. *MatchPattern* = (4,4,4) says that there might be a Web Service with a behavioural profile that matches to the query with a match score 4 on operation, match score 4 on output, and match score 4 on effect.

- Classify all possible match patterns to their rank order. This is determined by the summation of the match scores in each *MatchPattern*, denoted by *MatchPatternScore(MatchPattern)*. For example, *MatchPatternScore*((4,4,4)) = 12, *MatchPatternScore*((4,4,3)) = 11, and *MatchPatternScore*((4,4,2)) = 10. Different values of *MatchPatternScore* will determine the rank orders, and different *MatchPattern*s which have the same *MatchPatternScore* will falls into the same rank order. We can compute the number of possible rank orders by

$$\mathit{NumberofRankOrders} = \mathit{Max}(\mathit{MatchPatternScore}_1, \ldots, \mathit{MatchPatternScore}_n) - \mathit{Min}(\mathit{MatchPatternScore}_1, \ldots, \mathit{MatchPatternScore}_n) + 1$$

where n = the number of possible match patterns

In the example, the maximum *MatchPatternScore* is 12 (from *MatchPattern* = (4,4,4)) and the minimum is 5 (form *MatchPattern* = (3,1,1)). So the number of rank orders in the behavioural profile is 8. Figure 4.4 (a) shows only the top three rank orders; the highest *MatchPatternScore* = 12 will be the top rank order 1, followed by the lower scores with lower rank orders. Each rank order has a number of match patterns assigned to it. From this assignment, we can see that a Web Service whose behavioural profile has *MatchPattern* = (4,4,4) would be ranked higher than the one with *MatchPattern* = (4,4,3).

- Refine ranking by determining sub-ranking based on the specified feature priority preference. If the service consumer specifies feature priority preference, it can help determine relative ranking between match patterns within the same rank order. In Figure 4.4 (a), pattern number 2 (i.e. *MatchPattern* = (4,4,3)) is under the same rank order as pattern number 3 (i.e. *MatchPattern* = (4,3,4)) so primarily they are ranked equal. But with the feature priority preference $E \prec O \prec \Delta$ set for the behavioural profile, sub-ranking can be performed. Figure 4.4 (a) also shows an example of a sub-ranking table for rank order 2 (*MatchPatternScore* = 11) and rank order 3 (*MatchPatternScore* = 10). In the sub-ranking table for rank order 2, pattern number 2 wins over pattern number 3 (because it has a higher match score for output), pattern number 2 wins over pattern number 4 (because it has a higher score match for operation), and pattern number 3 wins over pattern number 4 (because it has a higher score match for operation).

Apart from the behavioural profile β, the query $\mathbb{Q}$ in our example above also involves the simple attribute profile α, structural profile γ, and operational rule-based profile ρ. We have to determine match patterns, rank orders, and sub-ranking tables for these profiles as well. In summary,

For α, rank order 1 (*MatchPatternScore* = 3) : *MatchPattern* = (3)

rank order 2 (*MatchPatternScore* = 2) : *MatchPattern* = (2)

rank order 3 (*MatchPatternScore* = 1) : *MatchPattern* = (1)

For γ, rank order 1 (*MatchPatternScore* = 4) : *MatchPattern* = (4)

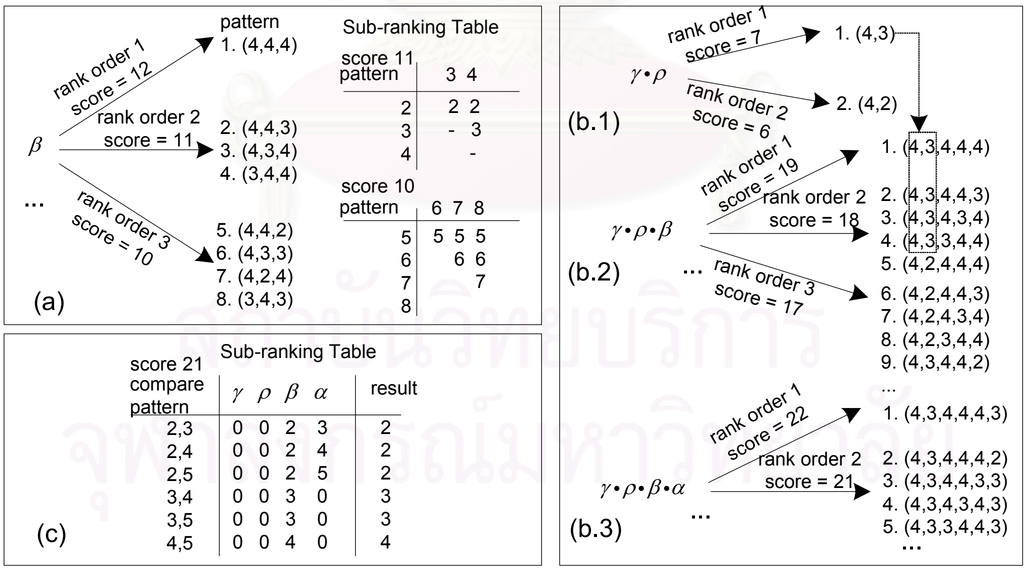
For ρ, rank order 1 (*MatchPatternScore* = 3) : *MatchPattern* = (3)

rank order 2 (*MatchPatternScore* = 2) : *MatchPattern* = (2)

Remark: For each of these three profiles, each of its rank orders has only one *MatchPattern*, therefore sub-ranking tables are not necessary.

(iii) Combine different profiles and determine match patterns, rank orders, and sub-ranking. This step is similar to step (ii) but is done across the profiles and the process is incremental. In Figure 4.4 (b.1), we start with combining the structural profile γ and operational rule-based profile ρ first. All possible match patterns are defined based on all *MatchPattern* under these two profiles (which have been generated in step (ii)). Therefore, we obtain *MatchPattern* = (4,3) and *MatchPattern* = (4,2) for the combination $\gamma \bullet \rho$ with *MatchPatternScore* = 7 and *MatchPatternScore* = 6 respectively. The match patterns from $\gamma \bullet \rho$ will be used to define match patterns when the behavioural profile is added to the combination. Figure 4.4 (b.2) shows some match patterns and rank orders for the combination $\gamma \bullet \rho \bullet \beta$. And subsequently, the simple attribute profile is combined into $\gamma \bullet \rho \bullet \beta \bullet \alpha$ in Figure 4.4 (b.3).

After the profiles are combined, we can similarly determine relative sub-ranking for match patterns within the same rank order. According to the profile priority preference $\alpha \prec (\gamma \approx \rho \approx \beta)$ set by the example, the sub-ranking table for rank order 2 (*MatchPatternScore* = 21) in the final combination $\gamma{\bullet}\rho{\bullet}\beta{\bullet}\alpha$ can be created; part of it is shown in Figure 4.4 (c). Each row of the table compares two match patterns and determines which one wins over the other with regards to each profile. In the case that there are conflicts, the profile priority preference is considered. Considering pattern number 2 (i.e. *MatchPattern* = (4,3,4,4,4,2)) and pattern number 3 (i.e. *MatchPattern* = (4,3,4,4,3,3)), the two are equal (i.e. no one wins) with regards to the profiles γ and ρ. However, pattern number 2 wins over pattern number 3 regarding to β, while pattern number 3 wins under α. This conflict is resolved by the profile priority preference; the sub-ranking table shows the overall result such that pattern number 2 wins over pattern number 3 because β has higher priority than α. Pattern number 2 hence will be ranked higher than pattern number 3 in the sub-ranking.



Sub-ranking Table

score 11

pattern	3	4
2	2	2
3	-	3
4		-

score 10

pattern	6	7	8
5	5	5	5
6		6	6
7			7
8			

Sub-ranking Table

score 21 compare pattern	γ	ρ	β	α	result
2,3	0	0	2	3	2
2,4	0	0	2	4	2
2,5	0	0	2	5	2
3,4	0	0	3	0	3
3,5	0	0	3	0	3
4,5	0	0	4	0	4

Figure 4.4 Example of ranking.

.

Looking back at the matching results in Section 4.2 and the shortened query $\mathbb{Q}$ with the relation expressions: hasDescription$^{S}_{\alpha}$, hasProduct$^{\phi}_{\gamma}$, hasDeliveryDayShipping$^{C}_{\rho}$, hasOperation$^{\Delta}_{\beta}$, hasOutput$^{O}_{\beta}$, hasEffect$^{E}_{\beta}$ }, we obtain the match results for the service S_1 and S_2 as follows.

Match($\mathbb{Q}$, S_1) = {$strong_{hasDescription}$, $exact_{hasProduct}$, $exact_{hasDeliveryDayShipping}$, $exact_{hasOperation}$, $partial_{hasOutput}$, $exact_{hasEffect}$}

Match($\mathbb{Q}$, S_2) = {$relaxed_{hasDescription}$, $exact_{hasProduct}$, $plug\text{-}in_{hasDeliveryDayShipping}$, $specialised_{hasOperation}$, $partial_{hasOutput}$, $partial_{hasEffect}$}

Match score for S_1 is (3+4+3+4+1+4) = 19 and for S_2 is (1+4+2+3+1+1) = 12. So S_1 is in a higher rank and closer to $\mathbb{Q}$ than S_2.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER V

MATCHMAKING AND RANKING EVALUATION

5.1 Matchmaking Analysis

Evaluation of matchmaking is based on relevance evaluation which concerns precision and recall of match results (Baeza-Yates and Ribeiro-Neto, 1999). Since our matchmaking process will return only the Web Services that can satisfy all capabilities requested in the query (i.e. they can match all relation expressions in the query but can also do more), this is called "plug-in match" in WSMO (WSMO, 2004). As a result, we assume precision is 1 as all services returned are definitely relevant to the query. However, the recall value may be low as there may be some relevant services that are not returned because they match only some relation expressions of the query (at least one). This is called "intersection match" in WSMO. We can apply intersection match to our matchmaking process instead and consider Web Services with intersection match as relevant to the query, so that the recall value will be increased. But by doing so, it is possible that a Web Service with plug-in match may be ranked lower than another Web Service with intersection match, because the former may match all requested capabilities but with low scores whereas the latter may match only some of the requested capabilities but with high scores. The Web Service with plug-in match may also be shifted down to lower sub-rank within the same rank order by the presence of another Web Service with intersection match which has the same match pattern score. The shift-down means the possibility that a service consumer will prefer and select the Web Service with intersection match instead of the one with plug-in match is high.

An experiment is conducted to study the effect of the shift-down. The matchmaking is tested under two scenarios *(i)* when plug-in match is used and *(ii)* when intersection match is used. We consider the case of the query $\mathbb{Q}$ with six relation expressions and the

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

match preference as well as the priority settings as in Section 4.3. There are 192 possible match patterns in total, with 11 rank orders. We select 50 match patterns out of 192 as the samples for observing their shift-down behaviour. These 50 match patterns are selected from each of the 11 rank orders in such a way that the 50 samples would reside in all rank orders in a normal distribution (Weis, 2004).

We start with the plug-in match scenario first. For each of the 50 sample match patterns, we compute a *relative distance* which reflects approximately how further down the sample is ranked, in relation to the match pattern at the top rank order. Since there may be multiple samples at a particular rank order, we compute an average of their relative distance values to represent a relative distance of any sample at that rank order. Suppose that we have the rank orders with match patterns assigned to each of them as in Figure 5.1. Some of the match patterns are the sample match patterns that we will observe. A relative distance of any samples at a particular rank order is computed by

RelativeDistanceOfSampleWithinRankOrder =

the number of match patterns in the same rank order that are ranked higher

RelativeDistanceOfAnySamplesAtRankOrder =

the number of all match patterns in all higher rank orders +

Average(*RelativeDistanceOfSampleWithinRankOrder*$_1$, …,

RelativeDistanceOfSampleWithinRankOrder$_n$)

where n = the number of samples at that rank order

From Figure 6, *RelativeDistanceOfAnySamplesAtRankOrder 1* = 0

RelativeDistanceOfAnySamplesAtRankOrder 2 = *1* + *Average*(*1*,*1*) = 2

RelativeDistanceOfAnySamplesAtRankOrder 3 = *6* + *Average*(*0*,*2*,*3*,*6*) = 8.75

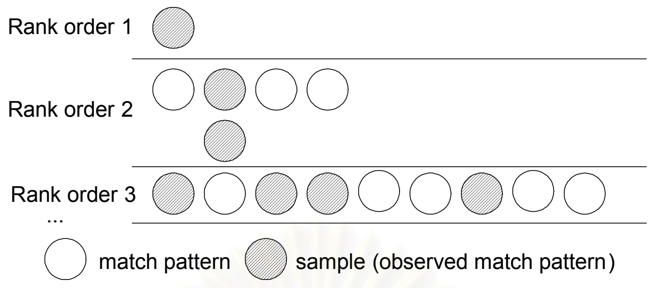


Figure 5.1 Example of relative distance.

Determining a relative distance of any samples at a particular rank order helps simulate the shift-down effect. When the intersection match scenario is used, there will be more match patterns and these patterns will scatter in all rank orders and interleave with the match patterns of the plug-in scenario. Therefore, the position of a particular sample may shift down. When the relative distance of that sample within the rank order is increased and the number of match patterns in all higher rank orders is increased, the relative distance of any sample in that rank order is too.

We can repeat the calculation of relative distance of the 50 sample match patterns under the intersection match scenario and compare the result with that of the plug-in match scenario. Table 5.1 shows the comparison. The percentage of shift range is calculated from the change in relative distance when using intersection match compared to the relative distance under plug-in match scenario. Match patterns in higher rank order will be less affected by the intersection match, but for those in lower rank orders, the shift range increases exponentially.

Table 5.1 Result of relative distance.

Rank order	No.of samples	Relative distance of any samples at rank order (plug-in match)	Relative distance of any samples at rank order (intersection match)	Shift range (%)
1	1	0	0	0
2	2	2	2	0
3	4	10	10	0
4	6	27.33	28.83	5.49
5	7	56.86	65.43	15.08
6	10	93.1	123.1	32.22
7	7	130.86	204.43	56.22
8	6	157.17	308.83	96.50
9	4	178	449.75	152.67
10	2	188.5	630.5	234.48
11	1	191	802	319.90

Remark:	*No.of relation expressions*	*No. of match pattern*	*No. of rank orders*	*Recall*
plug-in	6	192	11	0.107 (192/1783)
intersection	6	1783	22	1.00

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

The shift-down effect is interesting because, in real situations, matched services that are returned as search results may not be distributed evenly in every rank order. The distribution depends on the profiles of the services that are actually published to the semantic registry. For example, under the plug-in scenario, a search result may return two matched services, S_1 and S_2. S_1 falls in rank order 1 and is listed first whereas S_2 falls in rank order 4 and is listed second. Under the intersection match scenario, S_2 may be overtaken by other services with intersection match which happen to fall in rank order 4 as well. This will result in S_2 being pushed further down in the search result list, e.g. S_2 may be now listed fifth instead of second. This means the chance that S_2 will be picked by the service consumer is lower. The trade-off between recall value and relative distance should therefore be considered. We can improve the situation by allowing service consumers to specify which of the requested relation expressions must be matched (minimum requirement) in order to improve recall with less shift range.

5.2 Ranking Capability Analysis

By applying ordinal scale for ranking, the ranking is coarse-grained. As seen earlier, there are a number of match patterns that fall into the same ranking order. The ranking algorithm classifies them as equally ranked or "unjudgeable" as it is not able to determine which one should be ranked higher or lower than another.

This research is interested in the capability of the ranking algorithm. Ranking capability here refers to the ability of the algorithm to classify Web Services into different rank orders. We consider each pair of match patterns and if the algorithm can rank them, it has ranking capability over the pair. Intuitively, ranking capability can be determined by the proportion of the number of unjudgeable pairs of match patterns over the total number of match pattern pairs. This analysis can be performed on real integrated service profiles data, but in practice, the proportion may vary depending on the contents of the service profiles that are published. So it is difficult to realise the capability of the algorithm. In this thesis, we instead simulate a ranking capability analysis on all possible match patterns that an integrated service profile can match. Assume that service matches are evenly distributed; that is, for every possible match pattern, there is some service that matches the query by that pattern. Ranking capability can be computed by

Ranking capability = 1- $((\sum_{j=1}^{R}\sum_{i=1}^{P_j}(P_j - i))/\sum_{k=1}^{P}(P-k))$

where P_j is the number of match patterns in rank order *j*

R is the number of all possible rank orders.

P is the number of all possible match patterns = $\sum_{j=1}^{R} P_j$

From the formulae above, we can realise that

$\sum_{i=1}^{P_j}(P_j - i)$ is the number of match pattern pairs within a rank order *j*, i.e. the number of unjudgeable pairs

$\sum_{j=1}^{R}\sum_{i=1}^{P_j}(P_j - i)$ is the total number of unjudgeable match pattern pairs

$\sum_{k=1}^{P}(P-k)$ is the total number of match pattern pairs

We conduct an experiment in the most complex case where the largest number of match patterns will be involved. We assume the maximum number of match scores at 4 scales (i.e. 4, 3, 2, 1) and the match preference is set to the weakest level (i.e. 1). The experiment varies the number of the relation expressions that are specified in the query and needed to be matched. The result is in Table 5.2.

Table 5.2 Result from ranking capability experiment.

No.of relation expressions	1	2	3	4	5	6	7	8
No.of match patterns	4	16	64	256	1024	4096	16384	65536
No.of rank orders	4	7	10	13	16	19	22	25
No.of unjudgeable pairs of match patterns	0	14	258	3918	57640	849770	1.264E7	1.895E8
No.of pairs of match patterns	6	120	2016	32640	523776	8386560	1.342E8	2.147E9
Ranking capability	1	0.883	0.872	0.879	0.889	0.898	0.905	0.912

With the number of relation expressions = 1, ranking capability is 1. This is obvious because each rank order will have only a single match pattern and therefore there is no unjudgeable pair within the same rank order. When the number of relation expressions increases, ranking capability is still satisfactory and getting closer to 1. In real situations where match distribution may not cover all rank orders, the result of ranking capability analysis may differ. Nevertheless, feature priority preference and profile priority preference should help refine ranking, and intuitively should improve ranking capability.

CHAPTER VI

SEMANTIC WEB SERVICES DISCOVERY ARCHITECTURE

This section illustrates the architecture of the semantic Web Services discovery framework and the prototype. The architecture is in Section 6.1. Section 6.2 discusses how a service provider can create the integrated service profile. Section 6.3 shows the user interface for publishing and querying.

6.1 Overview of Architecture

The semantic Web Services discovery architecture is depicted in Figure 6.1. The architecture integrates together the UDDI registry and a semantic registry. The UDDI registry here is extended to accommodate a variety of simple attributes that result from the survey (see Table 3.1). The semantic registry will accommodate ontology-based profiles. Hence, service consumers can have a mixture of the traditional way of attribute query and semantic query. All the upper ontologies proposed by this research will be stored in the ontology repository within the framework. Domain experts can load these upper ontologies in order to derive shared domain ontologies by using an ontology editor such as Protégé (Protégé, 2001) (1). Such domain ontologies may be stored somewhere in the network, e.g. at the domain experts' organisations. However, the domain experts are required to store the URLs of these domain ontologies with the ontology repository. This is for the semantic registry to be able to preprocess domain ontologies and perform reasoning, and also for service providers to load such domain ontologies in order to derive their ontology-based profiles (2). Via a GUI, service providers can publish their profiles through the publishing proxy (3) and instantiate service constraints (4). The publishing proxy will store simple attributes to UDDI, and generate all ontology-based profiles and store them in the ontology repository. The profiles are pre-processed to extract knowledge and reason further by the parser module which is integrated with an inference engine (e.g. (Jena, 2003)). The result is stored in the semantic relation database which is designed to handle facts from the profiles, more facts from inference, and service constraint expressions (6). Using a database is powerful as it can deal with huge information and its pre-processing helps prepare information for future retrieval.

The architecture can provide the service consumers with a GUI that corresponds to the ontologies of the domain so that the consumers can specify query onto the profiles more easily (5). The query will go through the querying proxy to the matching module. While performing matching, the matching engine may interact with the constraint evaluation engine. Constraint expressions in the database can be translated into other language such as Jess (Jess, 2003), RuleML (Iwaihara et al., 2002), or SWRL (6), and then a rule engine is used to evaluate them. The matching engine also performs ranking. Matched services will be ranked and reported in an XML document which will be returned to the consumer (7).

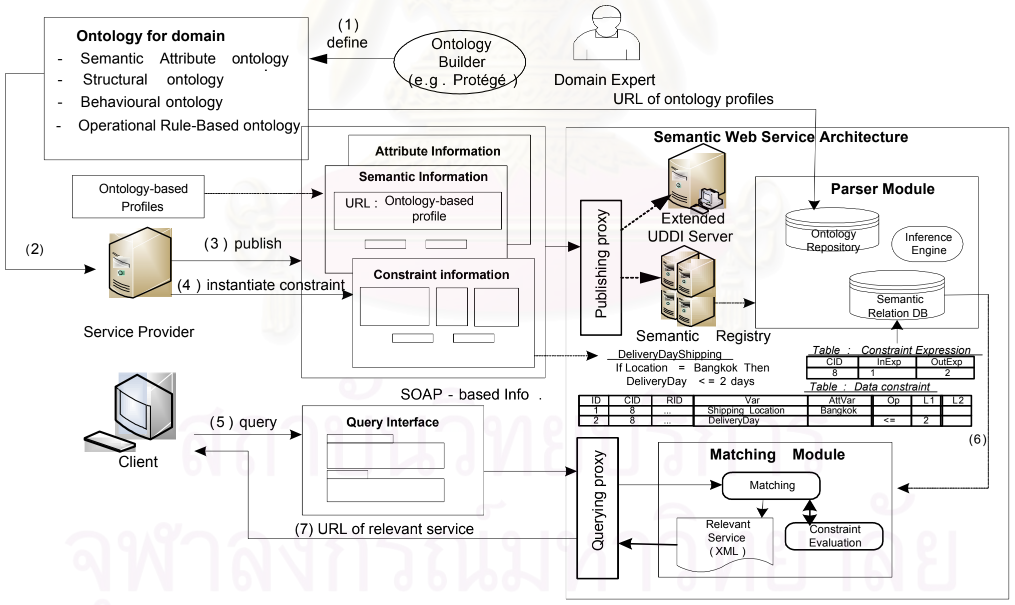


Table : Constraint Expression

CID	InExp	OutExp
8	1	2

Table : Data constraint

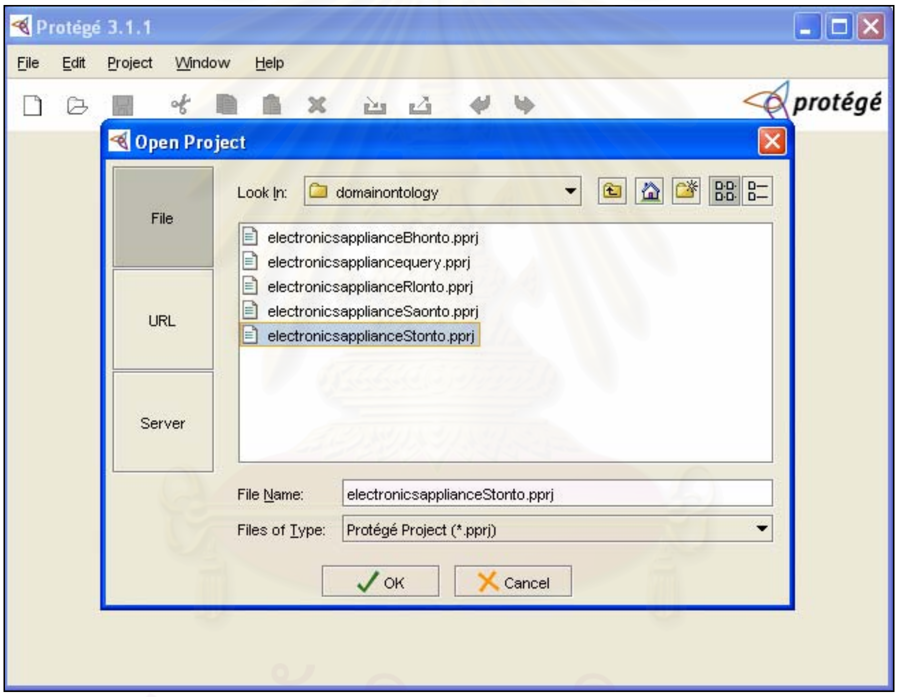
ID	CID	RID	Var	AttVar	Op	L1	L2
1	8	...	Shipping Location	Bangkok			
2	8	...	DeliveryDay		<=	2	

Figure 6.1 Semantic Web Services discovery architecture.

6.2 Creating Ontology-Based Profiles

Domain experts can use an ontology editor (e.g. Protégé, OIL-ED, RDF editor or even XML editor) to create capability-based ontologies for a domain. This requires basic knowledge about OWL and RDF languages.

Figure 6.2 presents the user interface of Protégé for opening an existing structural ontology for the electronics appliance domain (the ontology file is at a local URL). The tool can also open an ontology located at a particular URL.



Look In: domainontology

electronicsapplianceBhonto.pprj
electronicsappliancequery.pprj
electronicsapplianceRlonto.pprj
electronicsapplianceSaonto.pprj
electronicsapplianceStonto.pprj

File Name: electronicsapplianceStonto.pprj

Files of Type: Protégé Project (*.pprj)

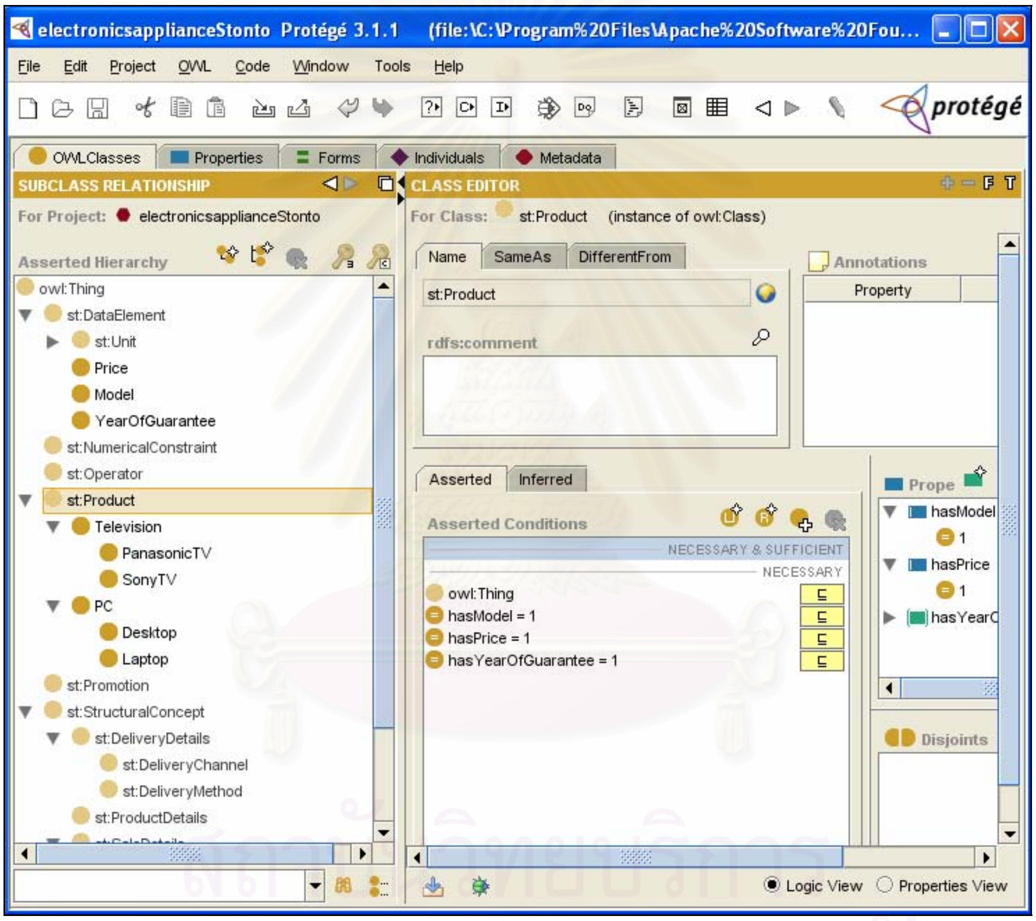
สถาบันวิทยบริการ

Figure 6.2 Opening structural domain ontology for electronics appliance domain.

จุฬาลงกรณ์มหาวิทยาลัย

Figure 6.3 shows an example content of the structural domain ontology for the electronics appliance domain. This is based on the upper structural ontology, e.g. product details, delivery details, and relevant data elements are defined for this domain.

Based on the capability-based ontologies of the domain, service providers in the domain can create instances of the profiles. For example, PowerBuy can create its own structural profile based on the structural ontology in Figure 6.3 and define specific information such as the models of the product, range constraint on the product price, or the number of years of guarantee.


electronicsapplianceStonto Protégé 3.1.1 (file:\C:\Program%20Files\Apache%20Software%20Fou...
File Edit Project OWL Code Window Tools Help
protégé
OWLClasses Properties Forms Individuals Metadata
SUBCLASS RELATIONSHIP
CLASS EDITOR
Asserted Hierarchy
Name SameAs DifferentFrom
Annotations
Property
Asserted Inferred
Asserted Conditions
Prope
Disjoints
Logic View Properties View

For Project: electronicsapplianceStonto

- owl:Thing
  - st:DataElement
    - st:Unit
    - Price
    - Model
    - YearOfGuarantee
  - st:NumericalConstraint
  - st:Operator
  - st:Product
    - Television
      - PanasonicTV
      - SonyTV
    - PC
      - Desktop
      - Laptop
  - st:Promotion
  - st:StructuralConcept
    - st:DeliveryDetails
      - st:DeliveryChannel
      - st:DeliveryMethod
    - st:ProductDetails

For Class: st:Product (instance of owl:Class)

st:Product

rdfs:comment

NECESSARY & SUFFICIENT

NECESSARY

owl:Thing ⊑
hasModel = 1 ⊑
hasPrice = 1 ⊑
hasYearOfGuarantee = 1 ⊑

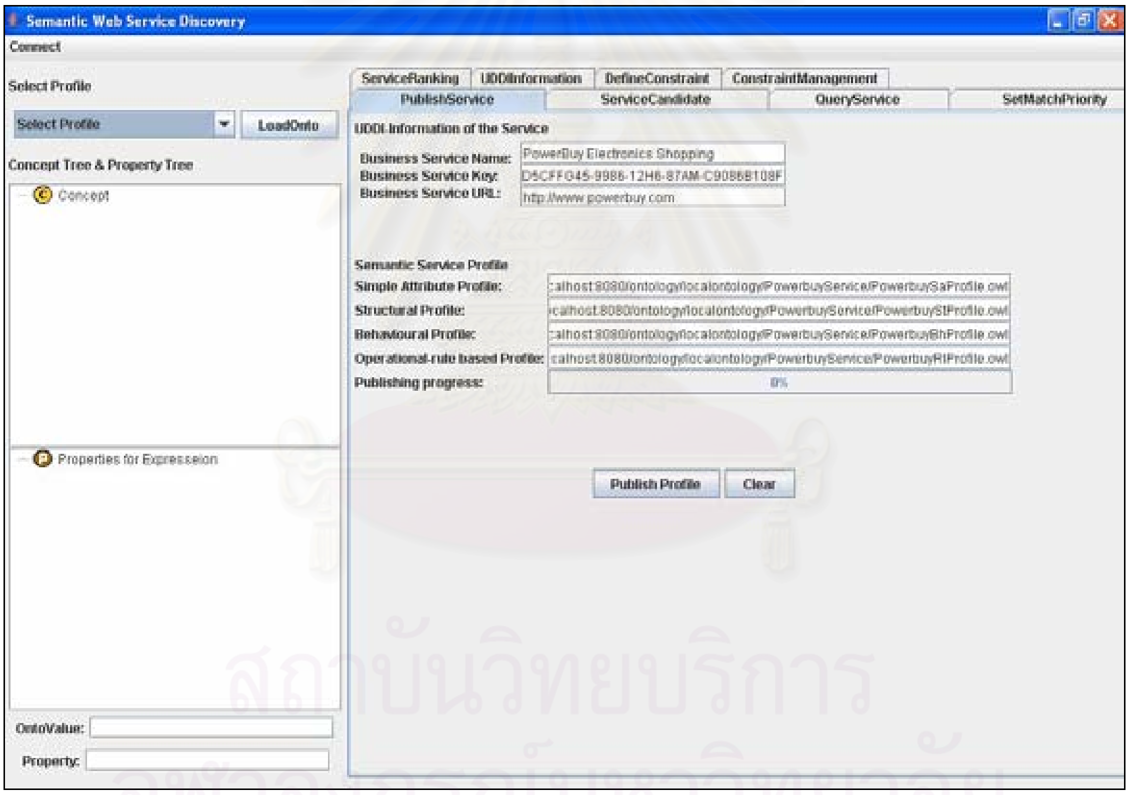
hasModel
1
hasPrice
1
hasYearO

Figure 6.3 Example of structural ontology for electronics appliance domain.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

6.3 Publishing and Querying Semantic Web Services

The application for publishing and querying semantic Web Services is developed with Java. Figure 6.4 shows the graphical user interface for publishing a Web service of PowerBuy shop. The publisher specifies URLs of all ontology-based profiles of the service and also some references to the corresponding UDDI information entries of the service. This is so that a correspondence between service information in our semantic registry and that in UDDI can be maintained. Our approach requires that the service providers publish their services to UDDI registry before publishing with the semantic registry.



UDDI Information of the Service

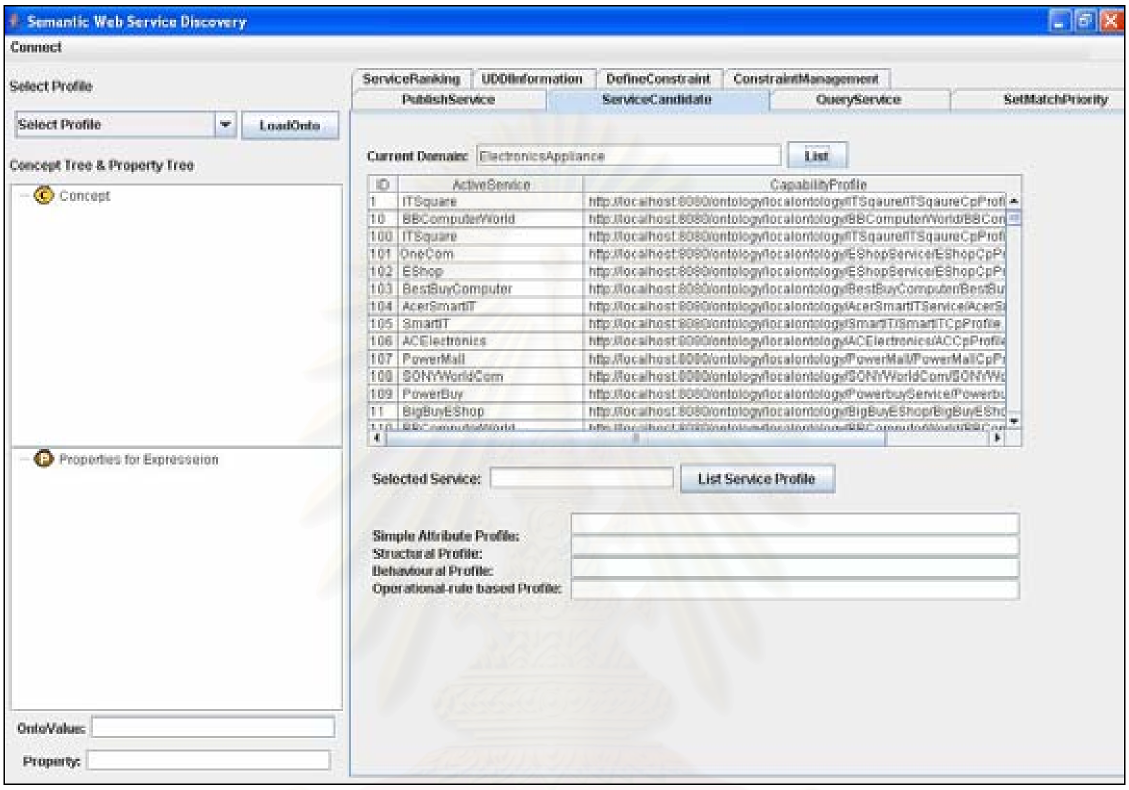
Business Service Name: PowerBuy Electronics Shopping
Business Service Key: D5CFFG45-9986-12H6-87AM-C9086B108F
Business Service URL: http://www.powerbuy.com

Semantic Service Profile

Simple Attribute Profile: :alhost:8080/ontology/localontology/PowerbuyService/PowerbuySaProfile.owl
Structural Profile: :calhost:8080/ontology/localontology/PowerbuyService/PowerbuyStProfile.owl
Behavioural Profile: :alhost:8080/ontology/localontology/PowerbuyService/PowerbuyBhProfile.owl
Operational-rule based Profile: calhost:8080/ontology/localontology/PowerbuyService/PowerbuyRlProfile.owl
Publishing progress: 0%

Figure 6.4 Publishing ontology-based profiles of PowerBuy.

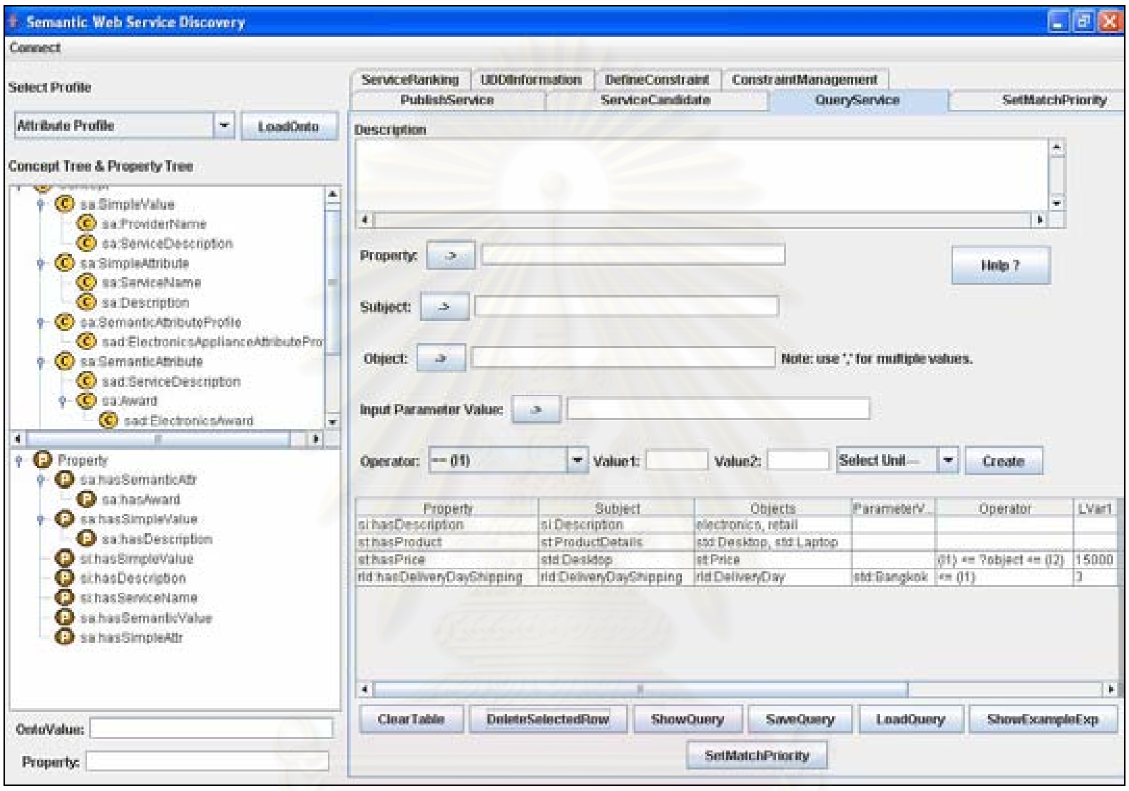
Service providers can browse the published services which are stored in a database of the semantic registry. Figure 6.5 shows an example of published services in the electronics appliance domain.



ID	ActiveService	CapabilityProfile
1	ITSquare	http://localhost:8080/ontology/localontology/ITSqaure/ITSqaureCpProfi
10	BBComputerWorld	http://localhost:8080/ontology/localontology/BBComputerWorld/BBCon
100	ITSquare	http://localhost:8080/ontology/localontology/ITSqaure/ITSqaureCpProfi
101	OneCom	http://localhost:8080/ontology/localontology/EShopService/EShopCpPr
102	EShop	http://localhost:8080/ontology/localontology/EShopService/EShopCpPr
103	BestBuyComputer	http://localhost:8080/ontology/localontology/BestBuyComputer/BestBu
104	AcerSmartIT	http://localhost:8080/ontology/localontology/AcerSmartITService/AcerS
105	SmartIT	http://localhost:8080/ontology/localontology/SmartIT/SmartITCpProfile
106	ACElectronics	http://localhost:8080/ontology/localontology/ACElectronics/ACCpProfile
107	PowerMall	http://localhost:8080/ontology/localontology/PowerMall/PowerMallCpPr
108	SONYWorldCom	http://localhost:8080/ontology/localontology/SONYWorldCom/SONYWo
109	PowerBuy	http://localhost:8080/ontology/localontology/PowerbuyService/Powerbu
11	BigBuyEShop	http://localhost:8080/ontology/localontology/BigBuyEShop/BigBuyEShc
110	BBComputerWorld	[illegible]

Figure 6.5 Example of published services in electronics appliance domain.

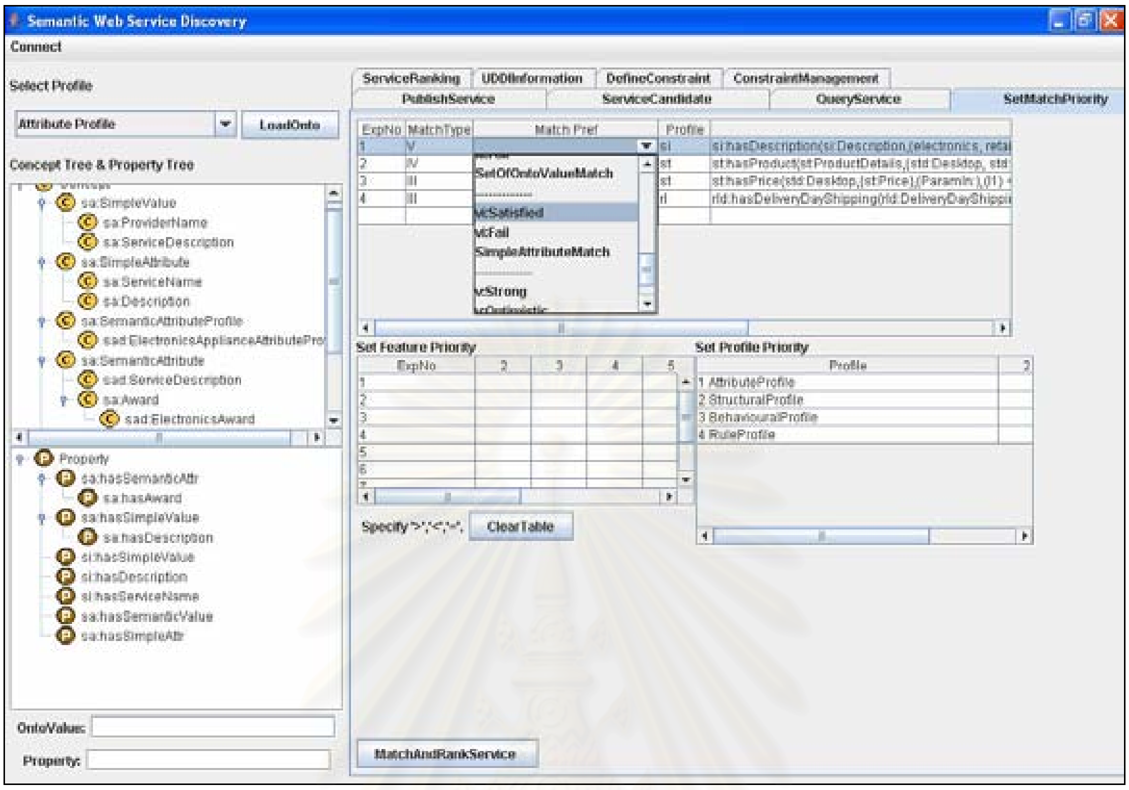
A graphical user interface is provided for specifying a query. Our prototype is integrated with Jena for querying ontology and MySQL is chosen for database management. Service consumers can specify relation expressions on different aspects of the expected service. The user interface will be aware of the domain ontologies and will provide the consumers with the templates for specifying requirements. Relation expressions and necessary information in a template will be used to generate an XML-based query profile which will be stored and can be retrieved later. Figure 6.6 shows an example of the template that allows service consumers to specify requirements such as description of the service, product required, and relevant constraints. In this figure, the constraints are specified for the price range of the desktops and the number of delivery days for the consumer's location in Bangkok.



Property	Subject	Objects	ParameterV...	Operator	LVar1
si:hasDescription	si:Description	electronics, retail			
st:hasProduct	st:ProductDetails	std:Desktop, std:Laptop			
st:hasPrice	std:Desktop	st:Price		(I1) <= ?object <= (I2)	15000
rld:hasDeliveryDayShipping	rld:DeliveryDayShipping	rld:DeliveryDay	std:Bangkok	<= (I1)	3

Figure 6.6 Querying a service in electronics appliance domain.

For query, service consumers can specify all preference criteria to help with matching and ranking. Figure 6.7 shows that a service consumer specifies the required match type for the match preference which is classified according to the classification of matching in Table 4.1.



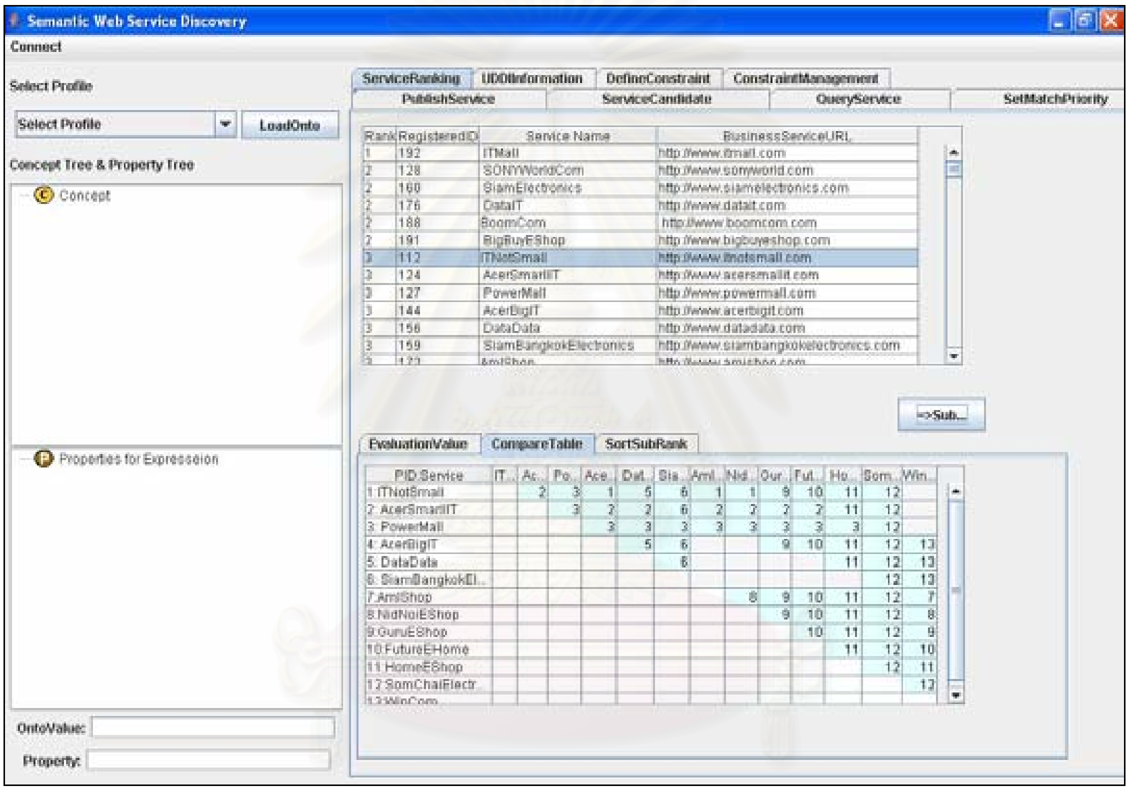
ExpNo	MatchType	Match Pref	Profile	
1	V		si	si:hasDescription(si:Description,(electronics, retai
2	IV		st	st:hasProduct(st:ProductDetails,(std:Desktop, std:
3	III		st	st:hasPrice(std:Desktop,(st:Price),(ParamIn:),(I1) <
4	III		rl	rld:hasDeliveryDayShipping(rld:DeliveryDayShippi

ExpNo	2	3	4	5
1				
2				
3				
4				
5				
6				
7				

Profile	2
1 AttributeProfile	
2 StructuralProfile	
3 BehaviouralProfile	
4 RuleProfile	

Figure 6.7 Specifying a match type for match preference.

Figure 6.8 shows an example of the returned services that match with a query. The table at the top of the screen lists the returned services with their rank orders. For example, ITMall is ranked highest among other matched services, whereas the next five services, from SONYWorldCom to BigBuyEShop, are equally ranked second, and so on. As shown in this table, ITNotSmall which is ranked third is currently selected. This effectively results in the services which are ranked equal to ITNotSmall being listed, in the table at the bottom of the screen, by sub-ranking. The columns of this table refer to all services that are equally ranked third in this example (i.e. ITNotSmall, AcerSmartIT, PowerMall etc.). The rows of this table refer to pattern IDs (PIDs) of the services in the corresponding columns (i.e. PID 1 corresponds to ITNotSmall, PID 2 to AcerSmartIT, PID 3 to PowerMall etc.). The number in each cell refers to the PID that wins (i.e. ranked higher) by sub-ranking. For example, the number 2 in the cell of PID 1 and AcerSmartIT means the service with PID 2 (i.e. AcerSmartIT) wins over or is ranked higher than the service with PID 1 (i.e. ITNotSmall) by sub-ranking. When the cell has no designated number, it is unjudgeable which service is ranked higher in sub-ranking. This is the case of, for instance, the service with PID 4 (i.e. AcerBigIT) and AmiShop.



Rank	RegisteredID	Service Name	BusinessServiceURL
1	192	ITMall	http://www.itmall.com
2	128	SONYWorldCom	http://www.sonyworld.com
2	160	SiamElectronics	http://www.siamelectronics.com
2	176	DataIT	http://www.datait.com
2	188	BoomCom	http://www.boomcom.com
2	191	BigBuyEShop	http://www.bigbuyeshop.com
3	112	ITNotSmall	http://www.itnotsmall.com
3	124	AcerSmartIT	http://www.acersmallit.com
3	127	PowerMall	http://www.powermall.com
3	144	AcerBigIT	http://www.acerbigit.com
3	156	DataData	http://www.datadata.com
3	159	SiamBangkokElectronics	http://www.siambangkokelectronics.com
3	171	AmiShop	http://www.amishop.com

PID Service	IT...	Ac...	Po...	Ace...	Dat...	Sia...	Ami...	Nid...	Our...	Fut...	Ho...	Som...	Win...
1 ITNotSmall		2	3	1	5	6	1	1	9	10	11	12	
2 AcerSmartIT			3	2	2	6	2	2	2	2	11	12	
3 PowerMall				3	3	3	3	3	3	3	3	12	
4 AcerBigIT					5	6			9	10	11	12	13
5 DataData						6					11	12	13
6 SiamBangkokEl...												12	13
7 AmiShop								8	9	10	11	12	7
8 NidNoiEShop									9	10	11	12	8
9 GuruEShop										10	11	12	9
10 FutureEHome											11	12	10
11 HomeEShop												12	11
12 SomChaiElectr...													13
13 WinCom													

Figure 6.8 Match results with ranking orders and sub-ranking orders.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER VII

SUMMARY

In this chapter, we conclude our work and discuss its limitations. Directions for future work are also given.

7.1 Summary and Contribution

The contribution of this work is the integrated service profile which is a combination of the traditional attribute-based description and ontology-based specifications for use in matchmaking of Web Services. Ontology-based specifications involve many aspects of the service capability in terms of semantic attribute, structural, behavioural, and operational rule-based profiles. This makes Web Services metadata model richer for service providers to publish and for service consumers to query. Publishing and querying can be made on all or any individual profiles. The integrated service profile is in accordance with the Service-Oriented Model part of the Web Services Architecture (Booth et al., 2004), which models a Web Service to have information about the provider, the syntax and semantics of the service, the tasks within the service, and a business policy.

Our matchmaking and ranking scheme may be considered as having three dimensions. The first dimension is that the matchmaking process considers only the services that match to any degree with all aspects specified in the query. Those that match with only some aspects will be filtered out. The second dimension is the refinement on the first dimension such that those matched services will be further filtered out if the service consumer specifies a match preference which signifies the minimum strength of matching that will be accepted. The third dimension is on ranking matched services that are returned as search results. The services that are equally matched, or are in the same rank order, will be sorted in the sub-ranking process according to the feature priority preference and profile priority preference of the service consumer.

By filtering out the services that do not satisfy any single aspect requested in the query, it can be seen that all of the required aspects within the query are considered as crucial behaviour. User preferences are only for sorting matched results in sub-ranking. Other approach to matchmaking and ranking may be by considering services that match with at least one aspect within the query as matched services, and then applying user preferences to sort those matched services into different rank orders and into different sub-ranking orders. We prefer the current approach because the algorithm is simpler and it can guarantee that all matched services that are returned as search results can satisfy the service consumer all round.

The proposed matching scheme gives an intuitive ordinal scale based on ontological subsumption which considers semantic compatibility. For each service, the pattern of the matching scores on relevant profiles is useful for determining which of the services is closer to a particular query. This approach is better than computing an average of the matching scores on relevant profiles because the degree of matching on each profile is preserved and this information is valuable for ranking and sub-ranking. Ranking matched services can be performed across profiles and refined under user preference setting. An analysis on matchmaking and ranking processes gives some insight of the methodology and gives confidence over the practicality of the approach.

Due to the richness of the profile, overheads exist when publishing and querying ontology specifications. However, the discovery architecture tries to facilitate in some way such as providing GUI that can guide providers and consumers to publish and query, the use of a powerful database to store facts, and preprocessing of knowledge extraction from the profiles.

7.2 Limitations

Our approach shares a limitation with other researches in semantic Web Services discovery such that the provision of domain ontologies and semantic service profiles is a manual process. These metadata cannot be created and published automatically, as opposed to the metadata of searching agents such as Google. Automatic metadata provision may require string analysis on some sort of domain or service information (such as (Heß and Kushmerick, 2004)).

The integrated service profile works on the basis of shared domain ontologies. Domain experts are assumed to provide all related domain ontologies, i.e. semantic attribute, structural, behavioural, and operational rule-based ontologies as the common knowledge within the domain. Service providers of a particular domain are also assumed to respect such domain ontologies and follow them when deriving service descriptions. This work does not consider the cases when domain ontologies evolve or when there is more than one set of domain ontologies defined by different experts. Also, ontology reasoning is based on subsumption relationships only.

The integrated service profile should not be considered an extensive set of capability-based profiles. It involves only four aspects of Web Services (i.e. attributes, structure, behaviour, and rules), and hence semantic publishing and querying of services are based only on these aspects. It cannot accommodate publishing and querying on other aspect of service capability such as on the internal structure of the service process (i.e. service workflow).

The prototype of the discovery framework only gives an idea of semantic service discovery; it should not be considered as a single implementation suite. The prototype only exemplifies the steps in publishing and querying Web Services using existing ontology tools and newly-developed software modules. Users, especially domain experts and service providers, are required to have good knowledge about ontology and its languages (i.e. RDF, OWL) in order to use ontology tools and the prototype itself.

Some user interface gives only raw information and is not so intuitive, e.g. sub-ranking order in Figure 6.8 is given as pair-wise order between each pair of services instead of a sorted list of all services in sub-ranking order.

7.3 Future Work

The matchmaking process can be improved in other aspects. Since ontological match is based on concept hierarchy, specialised match and generalised match at the moment cannot distinguish between matched concepts at different levels of the hierarchy. For example, for the query that is looking for a product by specifying an ontological term ElectronicsProduct, a service publishing its product as PC and another as Desktop will both satisfy specialised match to the query. However, since PC is closer to ElectronicsProduct than Desktop according to the concept hierarchy, the former service should be preferred to the latter. The matchmaking process can be refined to consider depth of concepts (i.e. distance-based analysis) in the hierarchy as the concept that are closer to the concept specified in the query should be preferred. Also, in some cases, we may combine specialised match with exact match because specialised match can guarantee the behaviour of exact match.

Measuring the performance of matchmaking process can be conducted. It is expected that the process will consume time to prepare and infer all ontologies that will be used, and to compute possible match patterns, rank orders, and sub-ranking tables. Nevertheless, these can be performed at initialisation time prior to publishing and querying. Service providers and consumers should work conveniently on the pre-processed information. The idea is confirmed by (Srinivasan et al., 2004) which reports a performance evaluation of their ontology-based matchmaking process.

To enhance discovery capability, it is possible to bridge our ontology-based metadata model to other similar models, e.g. to bridge our behavioural metadata model to OWL-S Process Model and WSMO capability model, and our operational rule-based metadata model to SWRL rule model. More types of profiles can also be introduced to the integrated service profile such as the service composition profile to allow query based on the internal structure of the service process.

Since our approach uses ontology-based metadata model as information model therefore, metadata management (e.g. maintenance of metadata) is another important issue that should be considered and included in the discovery framework. The prototype can also be improved into a complete framework that is more convenient to use.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

REFERENCES

Andreasen, T., Bulskov, H., Knappe, R.. On Ontology-Based Querying. Proceedings of Workshop on Ontologies and Distributed Systems. (August 2003).

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F.. The Description Logic Handbook : Theory, Implementation, and Applications. Cambridge University Press, 2003.

Baeza-Yates, R. and Ribeiro-Neto, B. Modern Information Retrieval, MA: Addison-Wesley for ACM, 1999.

Bansal, S., Vidal, J.M.. Matchmaking of Web Services Based on the DAML-S Service Model. Proceedings of the 2nd International Joint Conference on Autonomous Agents & Multiagent Systems (AAMAS 2003), 2003.

Bernstein, A., Klein, M.. Discovering services: Towards High-Precision Service Retrieval. Workshop on Web Services, E-Business, and the Semantic Web (WES 2002), 2002.

Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.. Web Services Architecture W3C Working Group Note 11 February 2004 [Online]. 2001. Available from: http://www.w3.org/TR/ws-arch/

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Bruijn, D. J., Lausen, H., Polleres, A, Fensel, D.. The Web Service Modelling Language WSML: An Overview. DERI Technical Report 2005-06-16, (June 2005).

Christensen, E. et al.. Web Services Description Language (WSDL) 1.1 [Online]. Available from: http://www.w3.org/TR/wsdl/

Chris Preist. A Conceptual Architecture for Semantic Web Services. International Semantic Web Conference, (November 2004).

Constantinescu, I., Faltings, B.. Efficient Matchmaking and Directory Services. IEEE/WIC International Conference on Web Intelligence (WI'03), (October 2003).

Di Noia, T., Di Sciascio, E., Donini, F.M., Mongiello, M.. A System for Principled Matchmaking in an Electronic Marketplace. Proceedings of the 12th International World Wide Web Conference (WWW 2003), 2003.

Dogac, A. et al.. Exploiting Web Services Semantics: Taxonomies vs. Ontologies. IEEE Data Engineering Bulletin, 25, 1, (March 2002).

Dumas, M., O'Sullivan, J., Heravizadeh, M.. Towards A Semantic Framework for Service Description. Proceedings of the 9th International Conference on Database Semantics, Hong-kong, (April 2001).

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Fensel, D. et al. OIL in a nutshell. In: Knowledge Acquisition, Modeling, and Management. Proceedings of the European Knowledge Acquisition Conference (EKAW-2000), R. Dieng et al. (eds.), Lecture Notes in Artificial Intelligence, LNAI, Springer-Verlag, (October 2000).

Gómez-Pérez, A.. Ontological Engineering: A State of the Art. Ontological Engineering: A state of the Art, Expert Update, British Computer Society. 2, 3, (Autumn 1999).

Gravano, L., Ipeirotis, P.G., Jagadish, H.V., Koudas, N., Muthukrishnan, S., Pietarinen, L., Srivastava, D.. Using q-grams in a DBMS for Approximate String Processing. IEEE Data Engineering, 24, 4 (2001): 28-34.

Gruber, T.. A Translation Approach to Portable Ontologies. Knowledge Acquisition, 5, 2 (1993) :199-220.

Hay, D., Healy, K.A.. Defining Business Rules ~ What are they really?. Technical Report 1.3, The Business Rules Group, July 2000 [Online]. Available from: http://www.businessrulesgroup.org/first_paper/br01c0.htm

Heß, A., Kushmerick, N.. Matching Learning for Annotating Semantic Web Services. AAAI Spring Symposium Semantic Web Services, 2004.

Heflin, J., Hendler, J., Luke, S.. SHOE: A Knowledge Representation Language for Internet

Applications. Technical Report CS-TR-4078 (UMIACS TR-99-71), 1999.

Iwaihara, M., Kozawa, M., Narazaki, J., Kambayashi, Y.. A System for Querying and Viewing Business Constraints. Proceedings of International Workshop on Rule Markup Languages for Business Rules on the Semantic Web, Schroeder, M. and Wagner, G. (Eds.), Sardinia, Italy, (June 2002).

Jaeger, M.C., Tang, S.. Ranked Matching for Service Descriptions using DAML-S. CAiSE Workshops, (2004): 217-228.

Jena. Semantic Web Framework: Jena [Online]. 2003. Available from: http://jena.sourceforge.net/index.html.

Keller, U., Lara, R., Polleres, A., Toma, I., Kifer, M., Fensel, D. WSML Deliverable D5.1 v0.1 WSMO Discovery, WSML Working Draft, September 13, 2004 [Online]. Available from: http://www.wsmo.org/2004/d5/d5.1/v0.1/20040913/d5.1v0.1_20040913.pdf

Larichev, O.I.. Ranking Multicriteria Alternatives: The Method ZAPROS III. European Journal of Operation Research, 131, 3 (2001): 550-558.

Li, L., Horrocks, I.. A Software Framework for Matchmaking Based on Semantic Web Technology. Proceedings of the 12th International World Wide Web Conference (WWW 2003), 2003.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Liskov, B.H, Wing, J.M.. A Behavioural Notion of Subtyping. ACM Transactions on Programming Languages and Systems, 16, 6 (1994): 1811-1841.

Lynne, M., Soh, C.. Structural Influences on Global Ecommerce Activity. Journal of Global Information Management, 10, 1 (Jan – Mar 2002): 5 -12.

Martin, D. et al.. Bringing Semantics to Web Services: The OWL-S Approach. Proceedings of 1st International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), (July 2004).

MySQL. MySQL [Online]. Available from: http://www.mysql.com

Navarro, G.. A Guided Tour to Approximate String Matching. ACM Computing Surveys, 33, 1 (2001): 31-88.

Oaks, P., ter Hofstede, A.H.M., Edmond, D.. Capabilities: Describing What Services Can Do. Proceedings of the 1st International Conference on Service Oriented Computing, 15-18 Italy, (December 2003).

Pan, J.Z., Horrocks, I.. OWL-E: Extending OWL with Expressive Datatype Expressions. IMG Technical Report, (April 2004).

Paolucci, M. et al. Semantic Matching of Web Services Capabilities. Proceedings of the 1st

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

International Semantic Web Conference (ISWC 2002), LNCS Vol. 2342, Springer Verlag, 2002.

Paolucci, M., Sycara, K.. UDDI Spec TC V4 Proposal Semantic Search [Online]. 2004. Available from: http://www.oasis-open.org/committees/uddi-spec/doc/req/uddi-spec-tc-req029-semanticsearch-20040308.doc

Protégé. 2001. Available from: http://protege.stanford.edu/

Resnik, P.. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. Proceedings of International Joint Conference on Artificial Intelligence, (November 1995).

Semantic Web. Semantic Web [Online]. 2001. Available from: http://www.w3.org/2001/sw

Sivashanmugan, K., Verma, K., Sheth, A., Miller, J.. Adding Semantics to Web Services Standards. Proceedings of the International Conference on Web Services, 2003.

Sriharee, N., Senivongse, T.. Discovering Web Services Using Behavioural Constraint and Ontology. Proceedings of the 4th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2003), Paris, France, (November 2003): 248-259.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Sriharee, N., Senivongse, T., Teppaboot, C., Futatsugi, K.. 2004a. Adding Semantics to Attribute-Based Discovery of Web Services. Proceedings of International Symposium on Web Services and Applications (ISWS' 04), Las Vegas, Nevada, USA., (June 2004): 790-794.

Sriharee, N., Senivongse, T., Verma, K., Sheth, A.. 2004b. On Using WS-Policy, Ontology and Rule Reasoning to Discover Web Services. Proceedings of the IFIP International Conference on Intelligence in Communication Systems (INTELLCOMM 04), Bangkok, Thailand, (November 2004): 246-255.

Sriharee, N., Senivongse, T.. Enriching UDDI Information Model with an Integrated Service Profile. Proceedings of the 5th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2005), Athens, Greece, (June 2005).

Srinivasan, N., Paolucci, M., Sycara, K.. An Efficient Algorithm for OWL-S based Semantic Search in UDDI. Semantic Web Services and Web Process Composition: First International Workshop, SWSWPC 2004, San Diego, CA, USA, (July 2004).

Sycara, K., Widoff, S., Klusch, M., Lu., J.. LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. Autonomous Agents and Multi-Agent Systems, 5, 2 (June 2002): 173 – 203.

Tapabut, C., Senivongse, T., Futatsugi, K.. Defining Attribute Templates for Descriptions of

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Distributed Services. Proceedings of 9th Asia-Pacific Software Engineering Conference (APSEC 2002), Gold Coast, Australia, (December 2002): 425-434.

The DAML Services Coalition. DAML-S: Web Service Description for the Semantic Web. Proceedings of the 1st International Semantic Web Conference (ISWC 2002), Sardinia, Italy, LNCS Vol. 2342, Springer-Verlag, 2002.

Trastour, D., Bartolini, C., Gonzalez-Castillo, J.. A Semantic Web Approach to Service Description for Matchmaking of Services. Proceedings of the International Semantic Web Working Symposium (SWWS'01), 2001.

Trastour, D., Bartolini, C., Preist, C.. Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. Proceedings of the 11th International World Wide Web Conference (WWW 2002), 2002.

uddi.org. UDDI: Universal Description, Discovery, and Integration of Web Services [Online]. 2001. Available from: http://www.uddi.org

Ukkonen, E.. Approximate String Matching with q-grams and Maximal Matches. Theoretical Computer Science, 92, 1 (1992): 191-211.

Verma, K., Sivashanmugam, K., Sheth, A., Patil, A.. METEOR-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Journal of Information Technology and Management, 2004.

W3C. Web Service Description Language (WSDL) 1.1 [Online]. 2001. Available from: http://www.w3.org/TR/wsdl

W3C. OWL Web Ontology Language Overview [Online]. 2004. Available from: http://www.w3.org/TR/2004/REC-owl-features-20040210/

W3C. Web Services [Online]. 2002. Available from: http://www.w3.org/2002/ws

Weiss, N.A.. Introductory Statistics. 7th edition, MA: Pearson Addison-Wesley, (May 2004).

Wickler, G.. Using Expressive and Flexible Action Representations to Reason about Capabilities for Intelligent Agent Cooperation. Doctoral Dissertation, University of Edinburgh, 1999.

WSA. Web Service Architecture [Online]. 2004. Available from: http://www.w3.org/TR/2004/NOTE-ws-arch-2004020211/#id2280201

WSMO. Web Services Modelling Ontology [Online]. 2004. Available from: http://www.wsmo.org

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

Zaremski, A.M., Wing, J.M.. Specification Matching of Software Components. ACM Transactions on Software Engineering and Methodology, 6, 4 (1997): 333-369.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

APPENDICES

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

APPENDIX
PUBLICATIONS

A.1 International Journal

1. Sriharee, N., Senivongse, T.. Ontological Business Policies for Web Services Discovery. GESTS International Transactions on Computer Science and Engineering, 10, 1: (June 2005): 191-203.
2. Sriharee, N., Senivongse, T.. On Matching and Ranking of Web Services Behaviour in Process-Based Service Discovery. WSEAS Transactions on Computers, 5, 2: (February 2006): 439-446.
3. Sriharee, N., Senivongse, T.. Matchmaking and Ranking Semantic Web Services Using Integrated Service Profile. Accepted to publish in International Journal of Metadata, Semantics, and Ontologies, 1, 2.

A.2 International Conference

1. Sriharee, N., Senivongse, T.. Discovering Web Services Using Behavioural Constraint and Ontology. Proceedings of the 4th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2003), LNCS 2893, Paris, France, (November 2003): 248-259.
2. Sriharee, N., Senivongse, T., Teppaboot, C., Futatsugi, K.. Adding Semantics to Attribute-Based Discovery of Web Services. Proceedings of International Symposium on Web Services and Applications (ISWS' 04), Las Vegas, Nevada, USA., (June 2004): 790-794.
3. Sriharee, N., Senivongse, T., Verma, K., Sheth, A.. On Using WS-Policy, Ontology and Rule Reasoning to Discover Web Services. Proceedings of the IFIP International Conference on Intelligence in Communication Systems (INTELLCOMM 04), LNCS 3283, Bangkok, Thailand, (November 2004): 246-255.
4. Sriharee, N., Senivongse, T.. Enriching UDDI Information Model with an Integrated Service Profile. Proceedings of the 5th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 2005), LNCS 3543, Athens, Greece,

(June 2005): 130-135.

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

BIOGRAPHY

Name	Natenapa Sriharee
Sex	Female
Marital Status	Single
Date of Birth	23 June 1974
Place of Birth	Yasothon
Permanent Address	104/31, Changwattana 43, Changwattana Road, Pakkred, Nontaburi 11120.

Education:

2005	Ph.D. (Computer Engineering), Chulalongkorn University. Funding source: The Commission on Higher Education, Ministry of Education.
2000	M.Sc. (Computer Science), National Institute of Development Administration (NIDA).
1998	B.Sc. (Statistics), Khon Kaen University (KKU).

Work Experience:

August 2001 – present:	Lecturer at Computer Science and Information Systems, King Mongkut's Institute of Technology North Bangkok.
January 2001 – July 2001:	System Analyst, Communication Authority of Thailand.
April 1998 – April 1999:	Computer Audit, Siam Commercial Bank PCL.

สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย