A semantic-augmented multi-level matching model of web services

Liu, M; Gao, Q.; Shen, W.; Hao, Q.; Yan, J.

Publisher's version / Version de l'éditeur:
https://doi.org/10.1007/s11761-009-0045-8
Service-Oriented Computing and Applications, 3, 3, pp. 205-215, 2009-06-01

A semantic-augmented multi-level matching model of web services

NRCC-52698

Liu, M; Gao, Q.; Shen, W.; Hao, Q.; Yan, J.

January 2010

A version of this document is published in / Une version de ce document se trouve dans:
Service-Oriented Computing and Applications, 3, (3), pp. 205-215, June 01, 2009, DOI: 10.1007/s11761-009-0045-8

A Semantic-augmented Multi-level Matching Model of Web Services

Min Liu[1,2], Qi Gao[1],Weiming Shen[2], Qi Hao[2], Junwei Yan[1]

[1]*CIMS Research Center, Tongji University, Shanghai 200092, P.R. China*

[2]*National Research Council Canada, London, Ontario N6G 4X8, Canada*

{lmin,2much, jwyan}mail.tongji.edu.cn; { weiming.shen,qi.hao}@nrc.gc.ca

Abstract

Semantic Web Services, augmenting Web service descriptions using Semantic Web technology, were introduced to facilitate the publication, discovery, and execution of Web services at the semantic level. Semantic matchmakers enhance the capability of UDDI service registries in the Semantic Web Services architecture by applying some matching algorithms between advertisements and requests described in OWL-S to recognize various degrees of matching for Web services. This paper proposes a novel semantics-enhanced Web service framework and a multi-level matching model for Web services. The matching process is achieved at five levels: syntactic, static semantic, dynamic semantic, qualitative service, and dependable service. A case study on collaborative design is used to demonstrate the proposed approach.

Keywords: Semantic Web Services, Similarity, Matching Degree, Collaborative Design.

1. Introduction

The combination of Web services, ontology and the Semantic Web has resulted in the emergence of Semantic Web Services [1].

Semantic Web Services (SWS) [2], augmenting Web service descriptions using Semantic Web technology [3], were introduced to facilitate the publication, discovery, and execution of services at the semantic level. Semantic Web service description languages, such as OWL-S [4] and Web Service Modeling Ontology (WSMO) [5], were proposed as abstractions of syntactic Web service description languages such as WSDL. OWL-S has been widely used as Semantic Web service description languages and submitted to W3C for possible standardization [6]. It describes the categories, inputs, outputs, and consequences of Web services in terms of concepts defined in OWL ontology. It also provides the grounding constructs for specialization into WSDL constructs for compatibility with existing Web services, which are described by WSDL documents.

To support programmatic service discovery,

Semantic matchmakers, which are usually software agents that accept and keep track of the descriptions of available services from providers and match them against the requirements from service consumers [3], enhance the capability of UDDI service registries in the Semantic Web Services architecture, by applying some matching algorithms between advertisements and requests described in OWL-S to recognize various degrees of matching for Web services.

In this paper, we propose a semantics-enhanced web service framework and a multi-level matching model for Web services. The matching process is checked through a set of rules that are organized into five levels: syntactic, static semantic, dynamic semantic, qualitative and dependable levels. Each rule compares a specific pair of attributes of interacting web services and operations. Furthermore, a service-similarity algorithm is proposed to address the various degrees of matching for Web services in the qualitative matching level.

The remainder of this paper is organized as follows: Section 2 gives an overview of the related work. Section 3 describes a semantics-enhanced Web service framework and a Web service operation model. Section 4 addresses the details about the multi-level matching model and the service-similarity assessment method for Web services. Section 5 presents a case study in collaborative design domain including the ontology design and matching process. A result of simulation is also demonstrated. Section 6 concludes the paper with some discussion on future work.

2. Related Work

In the development of integrated systems of large scale distributed and heterogeneous applications, the Web service architecture framework has been setdown as shown in Figure 10, which includes the service broker, the service provider and the service requester using WSDL, UDDI and SOAP protocols [11].

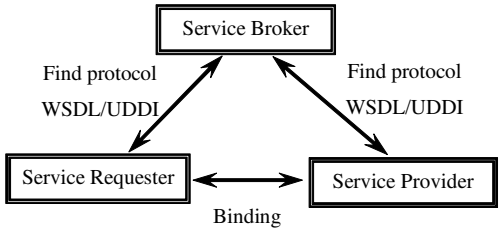

Figure 10 Web Service Architecture

In this Web service architecture, Web services rely on a set of related specifications to define how reusable components should be specified (through the Web-Service Description Language [8]), how they should be advertised so that they can be discovered and accessed (through the Universal Description, Discovery, and Integration [9]), and how they should be invoked at run time (through the Simple Object Access Protocol [10]).

However, this category-based service-discovery method (e.g. UDDI) is clearly insufficient [7] because it relies on the shared common-sense understanding of the application domain by the developers who publish and consume the specified services.

Semantic Web Services (SWS) were proposed to address this kind of problem. In SWS, ontology is a formal and explicit specification of a shared

conceptualization [12], and is expected to play a central role to empower Web services with semantics. Using the Semantic Web, computers will be able to understand pieces of information on Web pages rather than merely presenting them to users, and would be able to autonomously assist users in manipulating such information.

In SWS, the problem of service discovery and matching is analogous to the problem of component retrieval and information retrieval [7]. First, a WSDL specification declares a "software component" including a specification of its interface signature and a specification of where the actual implementation exists and how it can be used. Second, a WSDL specification usually includes a set of natural-language descriptions of the service and its elements. Therefore, given only a textual description of the desired service, a semantic information-retrieval method [7] can be used to identify and order the most relevant WSDL specifications based on the similarity of their element descriptions with the query under question.

Since WSDL does not provide formal specifications of the ontology of the data types of the available services and the functional semantics of their operations, it is not possible to guarantee that a retrieved service can fulfill all requirements of the requester. But WSDL is extensible and, in fact, the OWL-S effort aims at extending WSDL with such semantic specifications. However, until such extensions become standard and actual services with such semantic specifications are published, the issue of programmatically discovering relevant services among the multitude of published services makes the problem of service-similarity degree extremely relevant. Therefore, a service profile matching algorithm is proposed by researchers at Carnegie Mellon University [14][15] to be used by matchmakers.

3. Semantics-enhanced Web Service Framework and Service Operation Model

The Web services stack of standards is designed to support the reuse and interoperation of software components on the Web. A critical step in the process of developing applications based on Web services is service discovery, i.e. the matching and identification of existing Web services that can potentially be used in the context of a new Web application. However, discovery through catalog-style browsing (e.g. UDDI) is clearly insufficient. This paper presents a novel Web service framework and service operation model to improve the matching and discovery ability of Web service based on Semantic Web Services Framework.

3.1 Semantics-enhanced Web Services Framework

Semantic Web Services possess the potential to help unify the computing resources and knowledge scattered on the Internet into a large platform for collaborative design. To facilitate the publication and discovery of semantic Web services, an architectural framework, as an extended version of the standard Web

Services model, is proposed for the development of collaborative design systems (Figure 1).

In this model, the UDDI service registry is strengthened by the matchmaker, and the WSDL service description is enriched by the OWL-S semantic Web service description. Access to WSDL documents on the Internet is still necessary for service requesters to properly ground and bind to the service providers. However, the matchmaker does not need to store the copies (or URL) of such documents locally because only the semantic Web services description from OWL-S documents is used in the matching process.

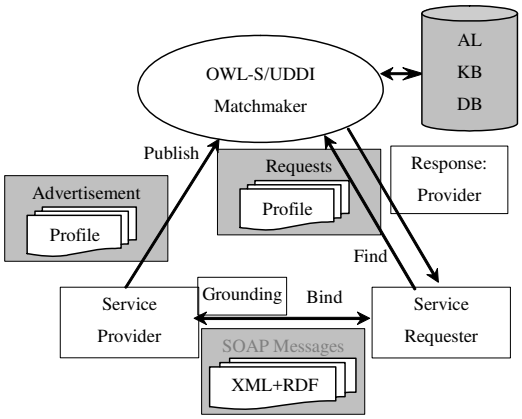

Figure 1 Semantic-enhanced Web Service Framework

SOAP messages, which are used for communication between Web services, are to be augmented with RDF statements so that XML data transmitted from a sender will be meaningful to the recipient. Each parameter of the request and response messages between service consumers and providers consists of the serialized XML data and its corresponding RDF statements which explain the meaning of such data. RDF statements are also typically encoded in XML format and the format is termed RDF/XML according to the RDF specification. As RDF/XML can naturally be included as extra attachments in SOAP messages, RDF-augmented SOAP messages will not cause a compatibility issue with non-semantic aware recipients because the recipients can simply ignore the part of an unrecognized attachment.

In the matchmaker, the Application Logic (AL) is the primary component of a semantics-enhanced service. It performs the functions advertised in its service description. It handles and processes requests initiated by service consumers, and delegates tasks to service providers by sending out request messages.

The Knowledge Base (KB) provides intelligent assistance to AL. It consists of an inference engine and the copies of ontology and RDF (or OWL) statements downloaded from the Semantic Web. Multiple domain ontology, e.g., a unit-of-measurement ontology, a material-property ontology and a structural engineering ontology is downloaded and stored locally in the knowledge base during the initialization process of a semantics-enhanced service. The knowledge base uses its inference engine and the premises from RDF (or OWL) statements to answer queries initiated from AL.

The database component (DB) provides information processing assistance to AL.

3.2 Multi-level Model of Web Service Operation

In the semantics-enhanced web service framework, the semantic description of web services and the description level of service operation are important for

evaluating their matching scores. The operation ontology, a meta-data ontology, is used as a template to define Web service operations and provides concepts that allow for the description of other concepts [6, 7].

Each operation, defined by a set of nonfunctional and functional attributes, is an instance of the operation ontology. Nonfunctional (e.g., qualitative) attributes include a set of metrics that measure the quality of the operation (e.g., time, availability, and cost). Functional attributes describe the syntactic and semantic features of an operation. We identify three groups of functional attributes: syntactic, static semantic and dynamic semantic. Syntactic attributes represent the structure of a service operation, e.g., the list of input and output parameters that define the operation's messages. Semantic attributes refer to the meaning of the operation or its messages. Among them, static semantic attributes describe features that are not related to the execution of the operation, such as the operation's category (i.e., domain of research). Dynamic semantic attributes refer to the way and the constraints under which the operation is executed. The dynamic semantic attribute generally refers to the business logic of the operation, i.e., the results returned by the operation given certain parameters and conditions.

L4: Dependable Service Level	Dependability of Operation	Self-managing Attributes	Self-optimization
			Self-configuration
			Self-healing
			Self-monitoring
		Trust Attributes	Availability
			Confidentiality
		Security Attributes	Identification
			Encryption
L3: Qualitative Service Level	Quality of Operation	Runtime Attributes	Response Time
			Running Statistics
		Business Attributes	Cost
			Reputation
L2: Dyanmic Semantic Level	Dynamic Semantics	Behavior	Exact
			Plugin
			Exact Post
			Plugin Pre
			Plugin Post
L1: Static Semantic Level	Static Semantics	Operation	Serviceability
			Prov. & Cons. Type
			Category
			Purpose
		Message	Message Type
			Data Types
			Business Role
			Language
			Unit
L0: Syntatic Level	Syntax	Operation	Mode
			Binding
		Message	Number of Parameters

Figure 2 Semantic-augmented multi-level model of Web Service operation

Based on the operation ontology, the proposed multi-level model for web service operation in matchmaker contains a set of rules that are organized into five levels (Figure 2). Each rule at a certain level compares a specific feature of services. L0 compares syntactic attributes, such as the number of parameters in the message. L1 compares static semantic attributes, including the static semantics of messages and the static semantics of operations. L2 compares dynamic semantic attributes. L3 focuses on the qualitative service and contains business and runtime attributes. L4 emphasizes the dependable service and contains the security, trust and self-managing attributes of the Web service operation.

4. Operation-based Multi-level Matching Model for Web Services in Matchmaker

In the multi-level model for Web service operation (Figure 2), the matching rules of Web service operation in matchmaker are divided into five levels. Therefore,

the Web service matching rules are also organized into syntactic matching level, static semantic matching level, dynamic semantics matching level, qualitative level and dependable level. A service request is matched in serial with the advertisement service through the above rules.

4.1 Syntactic Matching Level

The syntactic-matching focuses on the matching of WSDL specifications and it is a natural extension of the signature-matching method for component retrieval [7]. It involves the comparison of the operations' set offered by the Web service, which, in turn, is based on the comparison of the data types communicated by these messages, the operations' input and output messages, the operation and Web services.

- Matching of data types

The basis of service, operation and message matching is the matching of the individual data types. To assess the degree of similarity between two service data types, this method performs a domain-specific comparison of the "trees" corresponding to the XML syntax of these data types specifications. This comparison is based on three heuristics rules: Heuristic 1: Two simple data types are compared on the basis of their programming-language type (matchSimpleDataTypes); Heuristic 2: Complex data types are compared on the basis of their constituent elements and the XML grouping organization among them (getCompositeDataTypes); Heuristic 3: Complex data types (matchIdenticalTypes), imported from the same namespace, are considered identical if they have the same name. Based on these rules, the comparison algorithm for matching two lists of Data Types (sourceList and targetList) is described as follows:

```
int matchOfDataTypes (sourceList(m), targetList(n))
    matrix = construct a m*n matrix
    for (int i = 0; i<m; i++)
        for (int j = 0; j<n; j++)
            sourceType = sourceList(i)
            targetType = targetList(j)
            if (both sourceType and targetType are
        primitive)
                matrix[i][j] =
            matchSimpleDataTypes (sourceType,
            targetType);
            if (both types share the same name and
        namespace)
                matrix[i][j] =
            matchIdenticalTypes(sourceType,
            targetType);
            if (either sourceType or targetType is
        complex)
                newSourceList =
            getCompositeDataTypes(sourceType);
                newTargetList =
            getCompositeDataTypes(targetType);
                matrix[i,j] = matchOfDataTypes
            (newSourceList, newTargetList) +

                organizationChange(sourceType,
            targetType);
    return the match-degree with the maximum score;
```

- Matching of messages

After evaluating the data-type matching scores, the structures of the query-service messages against the target-service messages are compared. Given a source message and a target message, it is clear that there are many possible correspondences between their parameter lists. The objective of this step, then, is to evaluate all pair-wise mappings resulting from all possible permutations of the messages' parameter lists and to identify the parameter correspondence that maximizes the sum of their individual data-type matching scores. Therefore, the comparison algorithm for matching two messages (msg1, msg2) of Web service is described as follows:

```
int matchOfMessages (msg1, msg2)
    list1 = list of data types associated to msg1;
    list2 = list of data types associated to msg2;
    score = matchOfDataTypes (list1, list2)
    return score;
```

- Matching of operations

The matching process of operations is based on the process of matching their request and response (and exceptions when applicable) messages. The matching score between two operations is the sum of the matching scores of their input and output messages. The comparison algorithm for matching two Web service operations (op1, op2) is described as follows:

```
int matchOfOperations (op1, op2)
    score = matchOfMessages (op1 input, op2 input) +
        matchOfMessages (op1 output, op2 output)
    return score
```

- Matching of web services

Web services define a set of operations. The following comparison algorithm matchOfWebServices is used to match all operations between the WSDL specifications from the source service and target service in a pair-wise fashion to identify the best source-target operation correspondence.

```
int matchOfWebServices (service1, service2)
    m = number of operations in service1
    n = number of operations in service2
    operationMatrix = construct m*n matrix
    for (int i = 0; i<m; i++)
        for (int j = 0; j<n; j++)
            operationMatrix[i][j] =
        matchOfOperations(list1[i],list2[j])
    return the match-degree with the maximum score
```

4.2 Static Semantic Matching Level

The semantic description of service operations are semantically described at two levels: static and dynamic. The static semantics of an operation models "non-computational" properties of an operation, that is, properties that are independent of the execution of the operation. The static semantics is described at two "granularities": operation and message.

4.2.1 Static Semantics of Operations

The static semantics at the operation granularity is defined by the following attributes:

- Serviceability

This attribute gives the type of assistance provided

by the operation. Examples of values for this attribute are “part-architecture” and “material-property”. FEA assistance is another example of service that provides finite element analysis support to needy product design companies.

- Provider and consumer types

The provider of an operation may be corporations (“global”, “state,” “local,” etc.) or nonprofit agencies (“individual” and “community”). For example, partModeling service may be provided by the design department of a corporation or by volunteers (nonprofit community). The consumer type specifies the group of companies (e.g., automotive manufacturing, equipment manufacturing).

- Category

The category *C* of an operation *op* describes the area of Web service community of *op*. It is defined by a tuple (*Domain, Synonym, Specialization, Overlap*). *Domain* gives the area of interest of the community (e.g., “partdesign”). It takes its value from a vertical ontology for domain names. *Synonym* contains a set of alternative domain names for *C*. For example, “3D/2D-modeling” is a synonym of “partmodelling.” *Specialization* is a set of specializations of *C's* domain. For example, “partModeling” and “part” are specializations of “productDesign.” This means that *C* provides part modelling services for parts. *Overlap* contains the list of categories that overlap with *C's* category. It is used to provide a peer-to-peer topology for connecting operations with “related” categories. We say that *Category* overlaps with category if composing *op* is “meaningful.” By meaningful, we mean that the composition service provides a value-added service (in terms of categories).

- Purpose

The purpose describes the goal of the operation. It is defined by four attributes: *Func*, *Syn*, *Spec*, and *Overlap*. The *Func* describes the business functionality offered by the operation. Examples of functions are “partModeling,” “modelAnalyzing,” and “virtualAssembly.” The *Syn, Spec and Overlap* attributes work as they do for categories. The *Overlap* contains the list of purposes that are related to the purpose of the current operation.

4.2.2 Static Semantics of Messages

Each message within an operation is semantically described via a message type *MT*. *MT* gives the general semantics of the message. For example, a message may represent a “purchase order” or an “invoice.”

Message types do not capture the semantics of parameters within a message. Below, we define a set of attributes to model the semantics of message parameters:

- Data type

It gives the range of values that may be assigned to the parameter. We use XML Schema's built-in data types as the typing system. Built-in (or simple) types are predefined in the XML Schema specification. They can be either primitive or derived types. Unlike primitive types, derived types are defined in terms of

other types. For example, integer is derived from the decimal primitive type. Complex data types can also be adopted in this model.

- Business role

It gives the type of information conveyed by the message parameter. For example, an address parameter may refer to the first (street address and unit number) or second (city and zip code) line of an address. Business roles take their values from a predefined taxonomy. Every parameter would have a well-defined meaning according to that taxonomy. An example of such taxonomy is Rosetta Net's business dictionary [13]. It contains a common vocabulary that can be used to describe business properties.

- Unit

It refers to the measurement unit in which the parameter's content is provided. For example, a weight parameter may be expressed in "Kilograms" or "Pounds." An eligibility period parameter may be specified in days, weeks, or months. We use standard measurement units (length, area, weight, etc.) to assign values to parameters' units. If a parameter does not have a unit (e.g., address), its unit is equal to "none."

- Language

The content of a message parameter may be specified in different languages. For example, an English-Chinese-translation operation takes an English word as input and returns its Chinese translation as output. We adopt the standard taxonomy for languages to specify the value of this attribute.

4.3 Dynamic Semantic Matching Level

The dynamic semantics of an operation models computational or execution-related features of that operation and it generally refers to the way and the constraints in which an operation is executed.

The dynamic semantics or business logic of an operation op refers to the outcome expected after executing op, given a specific condition. Service providers may decide beforehand which "effects" are made visible to users. The business logic of an operation is defined by a set of rules where each rule R_{ik}^{m} has the following format:

$$R_{ik}^{m} = \frac{(\mathrm{Pr}eParameter_{ik}^{m}, \mathrm{Pr}eCondition_{ik}^{m})}{(PostParameter_{ik}^{m}, PostCondition_{ik}^{m})}$$

$\mathrm{Pr}eParameter_{ik}^{m}$ and $PostParameter_{ik}^{m}$ are sets of parameters. Each parameter is defined by name, data type, business role, unit, and language. The elements of $\mathrm{Pr}eParameter_{ik}^{m}$ and $PostParameter_{ik}^{m}$ generally refer to op's input and output parameters. However, they may in some cases refer to parameters that are neither an input nor output of op. For example, assume that the address of every citizen registered with the Department on the Aging is stored in the department's database. In this case, this parameter should not be required as input for the orderMeal operation since its value could be retrieved from the database.

$\mathrm{Pr}eCondition_{ik}^{m}$ and $PostCondition_{ik}^{m}$ are conditions over the parameters in $\mathrm{Pr}eParameter_{ik}^{m}$ and

$PostParameter_{ik}^{m}$, respectively. They are specified as predicates in first-order logic. The rule R_{ik}^{m} specifies that if $\Pr eCondition_{ik}^{m}$ holds when the operation op starts, then $PostCondition_{ik}^{m}$ holds after op reaches its End state. If $\Pr eCondition_{ik}^{m}$ does not hold, there are no guarantees about the outcome of the operation. The following is an example of a pre and post-condition of a rule associated with the operation *registerCompanySearch*:

$$\frac{income < 1{,}000{,}000 \wedge size \geq 200 \wedge zip = 22{,}044}{approved = true \wedge duration = 6}$$

The rule uses income (unit = {year, US dollar}), companySize, zip, approved, and duration (unit = {month}) as parameters. It states that companies with a yearly income of less than 1,000,000 dollars, a minimum company size of 200, and living in area code 22,044 are eligible for the company index for a 6-month period.

4.4 Qualitative Level

Qualitative level focuses on quality of services and contains business attributes and runtime attributes. One of the most important operations in the qualitative service level is the runtime matching of the ideal service profile of a service consumer against the service profiles registered by several service providers. Therefore, a service profile matching algorithm is proposed for use by matchmakers, which is inspired by the one proposed by Semantic Web Services researchers at Carnegie Mellon University [14, 15] and in the domain of Computational Mechanics [2].

In detail, an OWL-S profile description is a set of OWL-S statements that semantically describe a service, which is either needed by a service requester or offered by a service provider. In the OWL-S specification, the elements of a profile description that are relevant to the interoperation of Web services are the taxonomic type of profile, i.e., whether a service belongs to a certain class and *hasInput*, *hasEffect* and *hasOutput* properties.

For each pair of service profiles, the degree of matching is calculated by using the weighted average of the matching scores between the pairs of the profile types, the input parameters, the effects of service, and the output parameters. Mathematically, the degree of match between a pair of service profiles is

$$D_S = \frac{\sum_i W_i * d_i^p}{\sum_i W_i} \qquad (1)$$

Where D_S is the degree of match between two service profiles, W_i and d_i^p represent the weight and the matching scores between the profile types, the input parameters, the effects of services and the output parameters. By default, equal weights are assigned to the matching scores in matchmaking operations. Service consumers may request higher weights for certain pairs of profile descriptions if compatibility between those pairs is more important.

For each pair of the ideal service request concept C_R and the advertised concept C_A, the matching

score between C_R and C_A with respect to C_R, $d(C_R, C_A)$ is defined as:

$$d(C_R, C_A) = \begin{cases} 1.00 & \text{if } C_R \text{ is the same as } C_A \\ 0.75 & \text{if } C_A \text{ subsumes } C_R \\ 0.25 & \text{if } C_R \text{ subsumes } C_A \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

If we collectively call the profile types, the input, the effects of service and the output parameters, the value of $d(C_R, C_A)$ =1.00 signifies that the ideal parameter perfectly matches the advertised parameter. The value of 0.75 signifies that the advertised parameter is more general than the ideal parameter, and that the advertised service is not specifically made for the requester. The value of 0.25 signifies that the ideal parameter is more general than the advertised parameter, and that the advertised service may not completely fulfill the consumer's request. The value of 0 signifies that the two parameters are incompatible and the advertised service is not recommended for the requester [17].

4.5 Dependable Level

Composite Web services have high dependability requirements that call for trust mechanisms, security problems and fault tolerance due to both the specifics of the Web services architecture and the limitations of the Internet, which is not a reliable media [19]. The autonomy of component Web services raises challenging issues in specifying matching and composition processes and, in particular, dependable behaviors of composite services.

Comparing some other web services matching methodologies [1][2][3], in this paper we propose an additional dependable level which is built upon the qualitative level and contains the security, trust and self-managing attributes of operation.

The self-managing attributes are responsible for configuring services internally (Self-configuration), for healing over internal failures (Self-healing), for optimizing their own behavior (Self-optimization). Self-configuration is an important part of the self-managing attributes. Autonomic elements configure themselves, based on the environment in which they find themselves and the high-level tasks to which they have been set, without any detailed human intervention in the form of configuration files or installation dialogs.

Some standards, such as Trusted Platform Modules (TPM) and the Trusted Computing Module Software Stack (TCMSS) [17], will be added directly into the trust and security attributes as they are instrumented into Integrated Circuits (ICs), systems, and applications. Through these standards, TPM instruments, with core security technologies, can generate and store keys securely for use in digital certificates and encryption. These operations are accessed and controlled through standard TSS interfaces and are readily available to security management software for file/folder encryption, secure e-mail, identity and access management, and remote access.

5. Case Study: Product Collaborative Design

In the collaborative product design domain (as shown in Figure 3), the performance of a product prototype should be tested to assure the product performance before a new product is put into production. However, it is very expensive and time-consuming to manufacture a fully functioning prototype of the product. Many manufacturers use CAE tools to simulate and optimize the testing process of the prototype, use CAD tools to help design the product, and FEA tools to help analyze the computer model of the objective product after the engineers understand completely the physics action of product design.

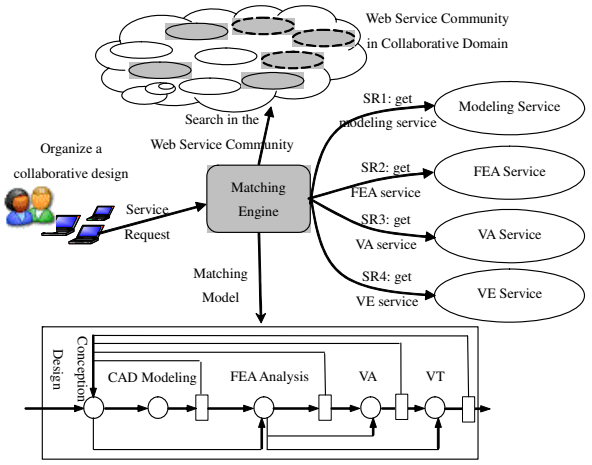

Figure 3 the Design and Analysis Process in a Collaborative Design Community

Figure 3 describes the design and analysis process model of a new product in a collaborative design community, which involves the conceptual design phase, the CAD modeling phase, the FEA analysis phase, the virtual assembly (VA) phase, the virtual testing (VT) phase, as well as related multi-iterative processes. After these processes, a more explicit computer model of the new product is built.

5.1 Web Service matching architecture

Many resources exist in the service application environment of the above design model, such as computers, mainframe, storage equipment, FEA software tools, CAD tools, and virtual testing systems. They are encapsulated on the Web into separate computing, modeling, data storage and data analysis services. Figure 4 depicts a Web Service matching architecture for collaborative design and its application environment.

In the Web Service matching architecture and its application environment, shown in Figure 4, the matching engine uses the embedded similarity degree algorithms, described in Section 4, to search the required Web service in the Web service community for the Web client at the levels of syntactic WSDL, operation static semantics, message static semantics, dynamic semantics, and qualitative similarity.

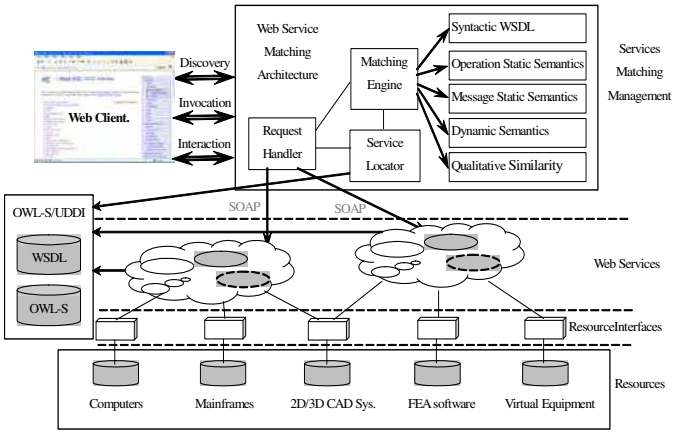

Figure 4 A Web Service Matching Architecture and Service Application Environment

5.2 Web Service Design

As an example, a CAD modeling requester sends a part-modeling message (partModelingMessage) to

Matching Engine (ME). After accepting this message, ME checks the validity of the user, which is reached by the checkUser operation from an Information System's checkIperationsPT service, and figure 5 gives the check process flow model in detail. If the response message is OK, ME starts the matching process. In the calling process, the WSDL-based description of the product modeling service is shown partly as follows:

```
<wsdl:definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:gridservicesoapbinding="http://www.gridforum.org/na
mespaces/2003/03/OGSI/bindings"
…
    ...
    <containers>  //description of service containers
        <container name=”modeling”
    messageType=”partModelingMessage”/>
        <container name=”modeling”
    messageType=”modelBuildMessage”/>
        <container name=”analyse”
    messageType=”modelAnalyseMessage”/>
        ...
    </containers>
    ...
    <wsdl:message name="partModelingMessageRequest">
    //description of request message
        <wsdl:part name="partID" type="xsd:string"/>
        <wsdl:part name="modelFileName"
    type="xsd:string"/>
    </wsdl:message>
    <wsdl:message
    name="partModelingMessageResponse">
      ...
    </wsdl:message>
    <wsdl:portType name="partModelingPT"> // service
    interface description
      <wsdl:operation name="ModelingRequest"
    parameterOrder="partID modelFileName ">
      <wsdl:input message="impl: partModelingRequest "
    name="partModelingRequest "/>
      <wsdl:output
    message="impl:partModelingResponse"
    name="partModelingResponse "/>
     </wsdl:operation>
      ...
    </wsdl:portType>
    <wsdl:binding name="partModelingSoapBinding"
type="impl: partModelingPT "> //dynamically bing the
service instances
        ...
    </wsdl:binding>
    <wsdl:service name="partModelingService"> // name of
domain service
        <wsdl:port binding="impl:
    partModelingSoapBinding"
    name="partModelingService">
            <wsdlsoap:address
        location="http://localhost/ogsa/services/partModel
        ingService"/>
        </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

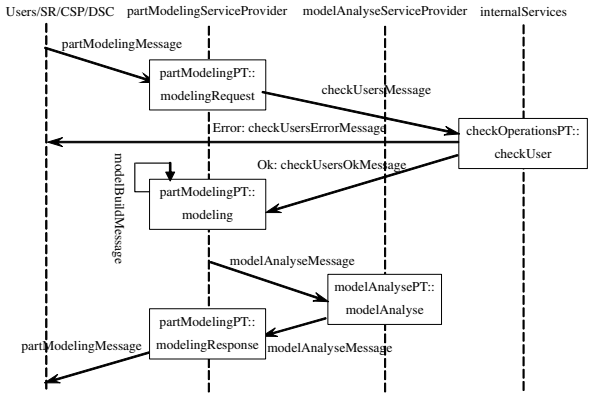

Figure 5 the Business Process Flow model for Product Design Process

5.3 Ontology Design in Domain

Ontologies play a central role in collaborative design and manufacturing: they enable fluent and consistent flows of data both inside and outside the company. They offer mature tools and an XML definition language like OWL which eases the integration with already implemented web services. Referring to MASON Architecture [18], we designed the domain ontology for Product Collaborative Design partly shown in figure 6. This ontology is built upon four main concepts: organization, operation, resource and cooperation. This figure is just a small subset of the whole ontology which contains 210 base concepts and 30 properties binding them currently and is still updating.

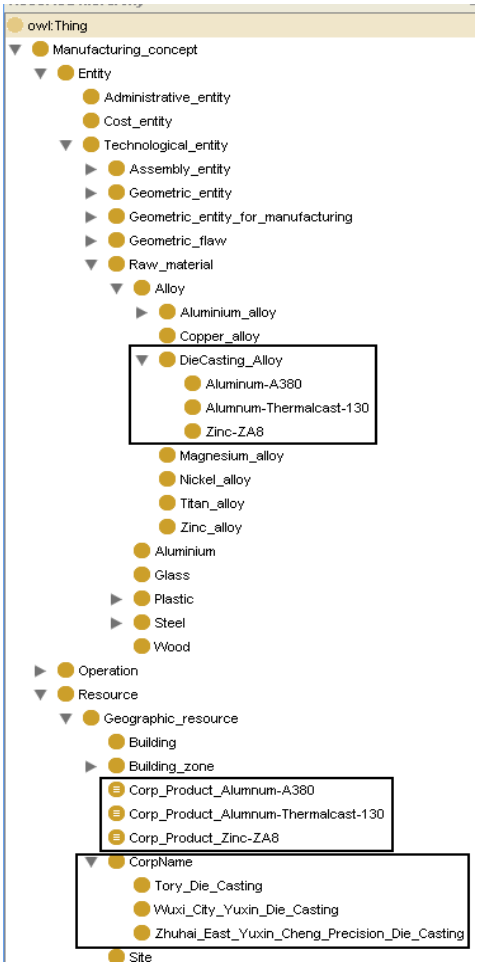

Figure 6 Collaborative Manufacturing Ontology

5.4 Experimental Result

The data types, messages, and operations of the request service are firstly referenced according to the matchOfDataTypes, matchOfMessages, matchOf Operations, and matchOfWebServcices algorithms. If successful, ME returns the advertisement service to the requester. Otherwise, the matching process continues respectively in the static semantics, dynamic semantics, and qualitative service levels. If a proper service cannot be found, a list of similar services, calculated by the matching degree equation (1), will be returned to the requester. Figure 7 describes a web services matching process. At first the service request is transferred to a

service template which contains properties at different levels. By using the ontology reasoner, Jena, these properties will be matched to the most suitable concept in the collaborative manufacture ontology described in section 5.3. Then we synthesize the matching results of different levels to a final web service matching score.

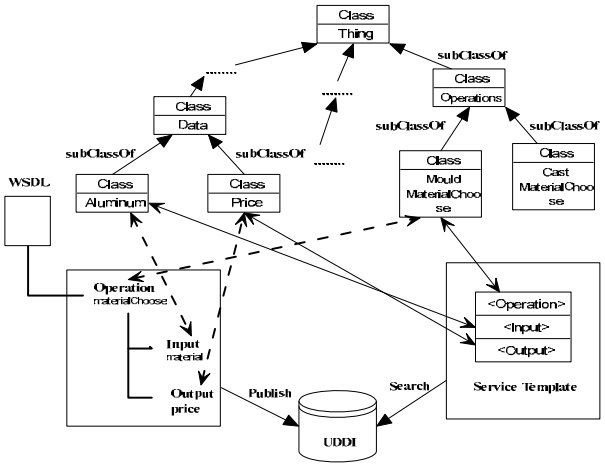

Figure 7 Web Services Matching Process

In order to judge the feasibility of the proposed multi-level matching architecture for semantic web services (abbreviated "multi-level matching"), we evaluate the performance of the matching algorithm through a comparison with the keyword-based matching algorithm on UDDI (abbreviated "Keyword-based matching") and the matching algorithm proposed by Paolucci and Sycara [14, 15] (abbreviated "Semantic Matching"). We simulated the completeness of Service Matching Query and the correction of Service Matching Query shown respectively in figure 8 and 9.

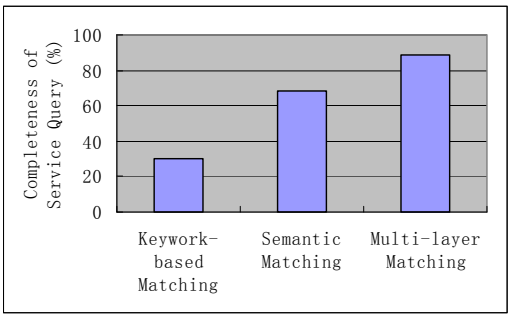

Figure 8 Completeness of Service Matching Query

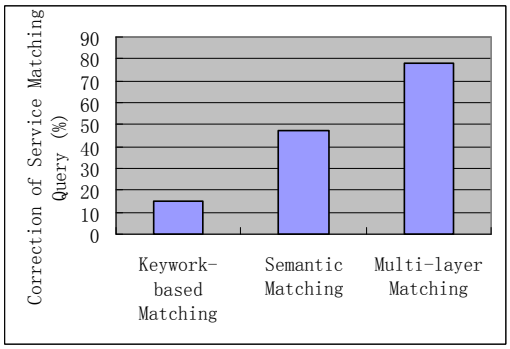

Figure 9 Correction of Service Matching Query

The Simulating Result shows that the completeness of Service Matching Query for three algorithms is respectively 30%, 68% and 89%, and the correction of Service Matching Query is 15%, 47% and 78%. Compared with the keyword-based matching algorithm on UDDI and the matching algorithm proposed by Paolucci and Sycara, the performance of the proposed multi-level matching architecture for semantic web services has greatly improved. Moreover, the completeness and correction of the multi-layer matching algorithm are higher e at 75%.

6. Conclusion

In this paper, a novel semantics-enhanced web service framework and a multi-level matching model for Web services is proposed. In the multi-level

matching model for web services, the matching process is implemented through a set of rules that are organized into five levels: syntactic, static semantic, dynamic semantic, qualitative and dependable levels. Each rule compares a specific pair of attributes of interacting web services and operations. A service-similarity matching algorithm is described in the qualitative matching level.

In term of future research, we are working in two directions: (1) developing the above semantic similarity algorithms such as fuzzy-set-based matching to improve the veracity of Web service matching; (2) investigating self-managing aspects in the dependable model of service operations.

Acknowledgements

The research work presented in this paper was partially supported by the National High Technology Research and Development Program of China (Grant No. 2007AA04Z104 and 2007AA10Z206) and the State Scholarship Fund of China Scholarship Council.

Reference

[1] B. Medjahed, A. Bouguettaya, "A Multilevel Composability Model for Semantic Web Services", *IEEE Transactions on Knowledge and Data Engineering*, 2005, 17(7), 954-968.

[2] T. Vacharasintopchai, W. Barry, V. Wuwongse, W. Kanok-Nukulchai, "Semantic Web Services Framework for Computational Mechanics", *Journal of Computing in Civil Engineering*, 2007, 21(2), 65-77.

[3] M. Burstein, C. Bussler, T., Finin, M.N. Huhns, M. Paolucci, A.P. Sheth, S. Williams, M. Zaremba, "A semantic Web services architecture", *IEEE Internet Computing*, 2005, 9(5), 72-81.

[4] The OWL Services Coalition, The OWL Services Coalition, OWL-S 1.1 beta release. http://www.daml.org/services/owl-s/1.1B/, July 2004.

[5] D. Romana, U. Kellera, H. Lausena, J. de Bruijn, R. Lara, M. Stollberg, A. Polleres, C. Feier, C. Bussler, D. Fensel, "Web Service Modeling Ontology", *Applied Ontology*, 2005, 1(1), 77–106.

[6] D. Martin, M. Burstein, J. Hobbs, et al., OWL-S: Semantic markup for Web services, http://www.w3.org/Submission/OWL-S, Nov. 22, 2004.

[7] E. Stroulia, Y.Q. Wang, "Structural and Semantic Matching for Assessing Web-service Similarity", In*ternational Journal of Cooperative Information Systems,* 2005, 14(4), 407–437

[8] E. Christensen, F. Curbera, et al. Web Services Description Language, http://www.w3.org/TR/wsdl

[9] T. Bellwood, L. Clement, D. Ehnebuske, et al. UDDI Technical Paper, http://www.uddi.org/pubs/UDDI-V3.00-Published-20020719.doc, July 19, 2002

[10] N. Mitra, Simple Object Access Protocol, http://www-.w3.org/TR/2003/REC-soap12-part0-20030624, 7/24/03

[11] Y.S. Tan, V. Vellanki, J. Xing, et al., "Service domains", *IBM Systems Journal*, 2004, 43(4), 734~755

[12] T. Berners-Lee, J. Hendler, O. Lassila, "The Semantic Web", *Scientific America*, 2001, 284(5), 34-43.

[13] B. Medjahed, B. Benatallah, A. Bouguettaya, et al. "Business-to-Business Interactions: Issues and Enabling Technologies", *The VLDB Journal*, 2003, 12(1), 59-85.

[14] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara, “Semantic matching of web services capabilities”, *The 6th IEEE International Symposium on Wearable Computers*, Seattle, Washington, USA, Oct. 7-10, 2002.

[15] K. Sycara, M. Paolucci, A. Ankolekar, N. Srinivasan. “Automated discovery, interaction, and composition of semantic Web services”, *Journal of Web semantics*, 2003, 1(1), 27–46

[16] P. Buche, C. Dervin, O. Haemmerle, R. Thomopoulos, ”Fuzzy Querying of Incomplete, Imprecise, and Heterogeneously Structured Data in the Relational Model Using Ontologies and Rules”, *IEEE Transactions on Fuzzy Systems*, 2005, 13(3), 373-383.

[17] Wikipedia. Trusted Platform Module. http://en.wikipedia.org/ wiki/Trusted_Platform_Module, Oct. 06, 2007

[18] S. Lemaignan, A. Siadat, et al., ”MASON: A Proposal For An Ontology Of Manufacturing Domain”， Distributed Intelligent Systems: Collective Intelligence and Its Applications, 2006. DIS. IEEE Workshop on,2006

[19] F. Tartanoglu, V. Issarny, et al., “Dependability in the Web Services Architecture”, Architecting Dependable Systems, 2003, p. 90-109.