Intention Interpretation for Service Request Using User Models

* Chiung-Hon Leon Lee, Alan Liu, Yen-Ru Cheng
* Department of Computer Science and Information Engineering
ChungChou Institute of Technology
Department of Electrical Engineering and Center for Telecommunication Research
National Chung Cheng University
Taiwan, R.O.C.
leonlee@dragon.ccut.edu.tw, aliu@ee.ccu.edu.tw, kencheng.69@gmail.com

Abstract

A method to extract user intention from the service request string entered by the user is proposed and applied in semantic Web service systems. Imprecise user intentions are modeled with computable goal models. Three kinds of background knowledge are used when extracting the user intention: domain ontology, goal structure, and user profiles. Domain ontology is constructed to represent the concepts in the application domain, the goal structure indicates the system capabilities, and the user profiles are for the intelligent system to infer the user's preference. The input terms will be parsed into different word sense sets. With the domain ontology, possible senses of the terms will be identified. The goal structure is used for mapping the user's requests to system capabilities. The user models will be used to aid the selection of a personalized candidate goal model for representing the user intention. The system can base on the selected goal model to generate and perform a series of actions for achieving the goal model.

Keywords: user intention, semantic Web service, goal model, domain ontology, user profiles.

1 Introduction

Web services are a good choice for loosely coupled architectures and reusable software components. By the technologies of SOAP, WSDL, and UDDI, Web services can be published, located, and invoked via the Internet. Although there are several techniques and standards which have been proposed to facilitate the Web services access, to a Web service system, it is too naïve to wish that one can directly use the results returned from the service providers to satisfy the user's request [1]. The system might have to interact with the user to elicit enough information for Web services acquisition; discover, select, and compose services; process information returned from the services; and demonstrate the results to the user.

In [2, 3], we proposed an approach to model the user's intention with goal that makes the service requester intention-aware. An intention-aware system has the capabilities to receive the user's request, interpret the request, and map the request to a series of system actions for satisfying the request. This paper continues our previous works to propose an intention interpretation approach by using user models for semantic Web service systems. The problems explored in this paper are challenges in the representation and the extraction of the services request intention. The contribution of this paper is that we draw attention to the similarity between the process of selecting, composing, and executing Web services and the intention-based Human-Computer Interaction (HCI) systems [4]. We represent a step forward in building intention-aware semantic Web service systems.

One of the most attractive characteristics of the Web services is service composition. The atomic services can be further assembled into value-added composite services for solving more complex problems. There are several proposals for creating service compositions in standardized and systematic fashions, such as Web Services Flow Language, XLANG, and BPEL4WS [1]. However, these standards describe Web service content in terms of XML syntax which lacks both a well defined semantics and sufficient expressive power.

The Semantic Web services [5] solve problems on the semantic level of Web services and address Web service descriptions as a whole. The semantic markup languages such as OWL-S and its previous release DAML-S [6] are proposed to describe the capabilities and contents of the Web services in a computer interpretable language and improve the quality of service discovery, invocation, composition, monitoring, and recovery.

Since the service composition language and semantic markup language have been proposed, for facilitating the services access, the agent technology is widely used by the Web service researchers [6, 7, 8, 9]. An agent is a software entity that has some properties like human beings such as autonomy, reasoning, learning, and knowledge level communication [10]. The agent represents the user to discover, interact, and compose Web services to satisfy the user's requests.

Because the agent has to delegate the user to do something for serving the user, how the agent understands the user's service request becomes an important capability. If the system misunderstands the user's request, the results return to the user will be wrong. Once a service-oriented system becomes intention-aware, the time of the user-system interaction can be reduced and the system usage will become more convenient.

In the research of HCI or Human-Robot Interaction (HRI), there are several approaches proposed to help the system identify the user's intention such as visual recognition of hand and body gesture, conversational interaction, force-feedback tactile glove, or fusing the input from multimodal [11-12]. These literatures focus on how to interpret and fuse the information from the input devices for getting the user's intention. In this paper, we claim that the system can interpret the user's intention more specifically and individually if the system has the background knowledge about the application domain, system's capabilities, and system users.

In [13], the authors proposed a method to extract keyword and concept linguistic features from the semantic context for intention modeling and action intention prediction. A Naïve Bayes classifier is used to find the proper concept of corresponding keywords. The user's intention is classified into two levels: action intention and semantic intention. Action intentions are the basic actions performed on the computer such as mouse click and keyboard typing. Semantic intentions concern with the aims which the user wants the system to achieve such as "To buy a book about Java programming for me," "To book a flight for me," etc.

In this paper, we are interested in how to provide a mechanism to let the intelligent system interpret user semantic intention from entered keywords. This allows the user to use a semantic Web service system like using traditional search engines by entering keywords. Based on the interpreted user intention, the agent can perform a series of actions to achieve the user's request.

The concept of proposed approach is shown in Figure 1. We assume that the entered string contains the user intention. The intention is defined as an object or goal that guides the system to perform one or more system functions for achieving it. A set of goal modes are defined from the system requirements point of view as the abstract descriptions of the system capabilities. The goal structure is composed of the goal models and plays the role which bridges the gap between service requests and system capabilities. For example, assume that a user entered a query string "query book java how to program", after interpretation process, the system might map the entered string to a Query_Book goal model and interpret the user intention as "To query a book which the book title is *Java How to Program*". Based on the goal model, the requester agent can generate a plan to satisfy the goal model. When executing the plan, the agent will send a service request to a service broker agent for getting a book query service or a booklist about Java programming. Finally, the agent will display the booklist to the user for satisfying the Query_Book goal.

Because the Web services requester does not have a priori knowledge about the service providers and Web services, how to map the service request to related Web services is an important challenge in Web services research. In software requirements engineering [14], the requirements are elicited from the user and the system will be designed to provide a set of services to achieve the elicited requirements. Because the services are defined by the system designer and the user generally has no idea about how the designer names and designs the system functions, the service request terms might mismatch with the system capabilities. In this paper, we propose a goal-based intention extraction process for mapping the service request to the system capabilities. The service request terms will be mapped to a predefined goal model and the system can base on the matched goal model to generate a set of actions to satisfy the user intention represented in that goal model.

The rest of this paper is organized as follows. The background knowledge used for mapping the service request terms to goal structure is introduced in Section 2. In Section 3, the process of the goal driven user intention extraction approach is proposed. An implementation of proposed method is described in Section 4. Finally, the conclusions are given in Section 5.

2 Background Knowledge for Intention Interpretation

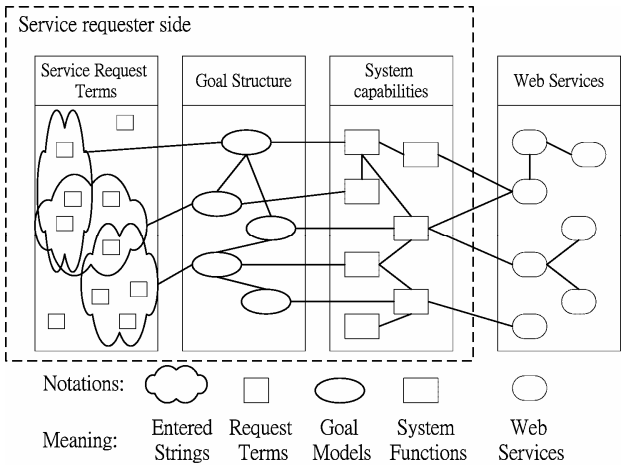

Figure 1 The concept of intention mapping

Background knowledge is an essential part when acquiring user intention from the keywords entered by the user. We take three types of background knowledge into consideration to support the intention extraction process. First, a goal structure is used to link the possible user intentions and the system capabilities. Next, the domain ontology is used to identify the possible senses of input keywords. To construct the domain ontology, a lexical dictionary WordNet [15] is used as the basis of the ontology. Finally, a user profile is kept for recording the user information for selecting personalized goal models.

2.1 Goal Models

One of the most important issues in user intention extraction is how to transfer the service requests based on the user's point view to the system capabilities based on the designer's point of view. We use the goal model proposed in [2] to facilitate this transformation. The user-entered service request string will be mapped to related goal models, and the system based on the matched goal models to generate a series of actions to achieve the goals.

We assume that the system requirements are elicited, analyzed, and represented by the use case approach [16, 17]. The user cases will be extended with goal models to facilitate the construction of the goal structure. The goal-driven use case (GDUC) approach [18] is adapted to discover the goal models. The GDUC approach is divided into three steps: identifying actors and use case, extending the use case with goal, and analyzing goal relationships.

The procedure of identifying actors and use case is similar to general use case approaches for finding the system actors and use cases. The actors represent the information provider, system function user, and system cooperator, etc., such as a real user, hardware, or virtual software programs. In this paper, we are interested in the actual user rather than other actors, because the goal models extended from the use cases are the basis of user intention representation.

A use case is a sequence of actions performed by the system that yields an observable result of value to a particular actor. The use case extension step is to extend each use case with a goal model. A goal model is composed of three parts: contents to represent the query variables, properties to describe attributes of the goal, and constraint to indicate the system awareness of the environment before and after the goal has been achieved.

The contents play an important role for mapping the entered request string to related goal models. The verb in the request string will be interpreted as an action which should be performed to satisfy the goal. The adjectives and adverbs play the role of the constraints to the execution of the action. The nouns are assigned as the parameters of the action.

To obtain goal models of the system is a gradual process. Through the use case based requirement analysis, the goal models can be found from coarse to delicate. For instance, assume that a user requires a bookstore Web service system to buy a book, the service requester has to generate a bookstore service query request, receive the information of bookstore store services from service broker, select and invoke the bookstore service, receive book information from the service provider, present the book information to the customer, invoke a book order service, and present the order result to the customer.

In some situations, the user might only want to query a book, so some query book steps can be integrated into one abstract step "query book" to represent another use case which performs the query book activity. Sometimes, the user might want to buy an inexpensive book. In other words, the system needs to sort the queried booklist according to the book price for the user to select the inexpensive books more conveniently. This requirement can be seen as nonfunctional requirement to extend the "buy book" scenario.

Figure 2 shows the use case diagram of the bookstore example. The actor "Customer" represents the user who wants to use the system to query or buy books. Three use cases are identified, "Buy a book", "Query books", and "Get inexpensive books". The relationship between "Buy a book" and "Query books" is an "include" relationship. It represents the fact that when a user wants to buy a book, he might need to query the book and get detail information about the book first. The relationship between "Buy a book" and "Get inexpensive books" is an "extend" relationship. It represents that if the user wants to buy an inexpensive book, the system should sort the books by price for the user to select an inexpensive one. BuyBook is

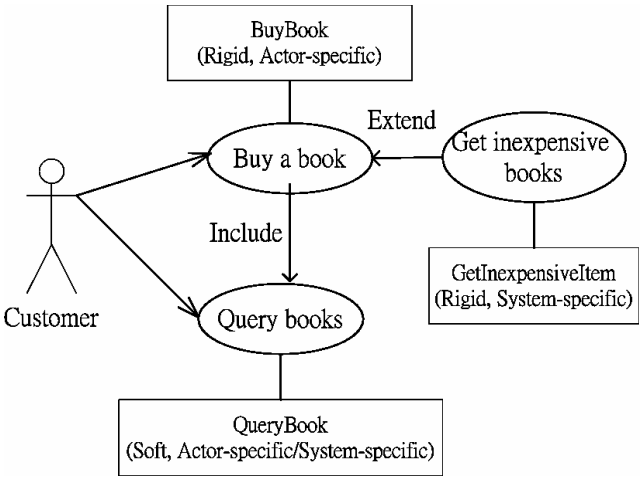

Figure 2 A bookstore use case diagram

a rigid actor-specific goal because this goal is initiated by the user and has to be satisfied or otherwise fail. QueryBook is a soft actor or system specific goal, because the goal could be initiated by the user or system to support the first goal and could be satisfied to a degree. GetInexpensiveItem is a system specific goal because it only initiated by the system to support the BuyBook goal.

2.2 Domain Ontology

A key component in user-entered data parsing and interpretation is the domain ontology which is a conceptualization of the application domain in a machine-readable form [19]. The ontology comprises the classes of entities, relations between entities and the axioms which apply to the entities which exist in that domain.

How to represent concepts and concept relationships in a machine readable form is important issue when constructing the domain ontology, especially, when we prefer to process the content of information rather than just to present information to users. The Web Ontology Language (OWL) [20] is designed for this purpose. OWL provides additional vocabulary along with a formal semantics to facilitate greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S). Although OWL gives a good way to represent the domain ontology, how to construct a domain dependent ontology is still a big challenge.

We assume that the linguistic features in the input string may indicate a user's intention. Two types of linguistic features proposed in [13] are considered: keyword and concept. A keyword feature is a single word extracted from the stop-word excluded input string. For example, a sentence "Buy a book about Java programming" is parsed to keyword features "Buy book Java programming".

Only using keywords to represent user's intention is too specific. For example, "buy" and "purchase" is of the same meaning sometimes; however, they are different keyword features. In order to solve this problem, we extract relationships between keywords from the WordNet to facilitate domain ontology construction. WordNet is a lexical dictionary representing the underlying lexical concept in various forms: synonyms (similar terms), hypernyms (generalization terms), and hyponyms (specification terms), etc. These relationships are suitable for domain ontology construction. The most representative relationship (i.e. hypernyms) is selected as the concept feature. However, not all concept generalization in WordNet is suitable for the linguistic feature. For example, the hypernyms of the word "java" in the hierarchy could be "java-island-land-object-entity", and "object" and "entity" may be too abstract for domain ontology construction.

2.3 User Profile

We divide the user's profile into static records and dynamic records. The static records include the user's ID, name, interest, birthday, career, etc. Such information can be derived when the user first registers to the system. A set of rules can be derived from the static records to filter the candidate goal models. For example, if the goal selection module receives two goal models "Query_Book" and "Book_Flight" and the user is a student, then the system will select "Query_Book" and remove the other goals because the agent can infer that it is rare to have a student who has time and money to go to Java Island.

The dynamic records save a table of periodic patterns and pattern related count to support the goal selector ranking related goal models. The periodic patterns are derived from the successfully performed user goals in a user usage period. Such user goals have been performed successfully and the view field of the goal model is user-specific. A method to mine the periodic patterns and how to use the user profile to select a goal model is described in the next section.

3 User Intention Interpretation

"Understanding" a service request refers to the computer's ability to transform the verbal form into machine-readable semantics. User intention interpretation involves the extraction of *key concepts* from the entered string, as well as inferring the *informational goal*. There are many research issues in natural language understanding (NLU), and we follow the state-of-the-art NUL techniques to restrict the application domain and entered string format for limiting the scope of intention understanding [21].

There are two main tasks in the user intention interpretation process: keywords abstraction and goal model generation. The former is to parse and interpret the entered keywords for finding possible senses of the keywords. The latter is mapping the interpreted keywords to the related goal models.

Sycara and her colleagues proposed an abstraction algorithm for mapping the query message from the service requester to an advertisement of service providers [22]. Our user intention extraction algorithm is based on Sycara's algorithm. The difference is that our work is to get a goal model to represent the user intention but the abstraction algorithm is to generate a service profile for discovering a proper Web service. In our approach, the requester agent can base on the goal model to make a plan for intention satisfaction.

Figure 3 shows the procedure of user intention extraction process in three main steps: keywords abstraction, goal generation, and goal selection. The domain ontology, goal structure, and user profiles are used to support the process. The selected goal model will be used to represent the user intention. Based on the selected goal model, the system can generate a series of actions to satisfy the goal.

3.1 Keywords Abstraction

The keywords abstraction step receives the service request string from the user and refers to domain ontology for generating a set of word sense. The keywords abstracting process has three steps.

1. Separating the input string into a set of verb triples (*Verb, Constraint, Parameters*).
2. Generating possible senses of terms in *Parameters*.
3. Eliminating impossible senses of terms.

Step 1 extracts the verb from the input string, and separates the other terms into nouns, adjectives, and adverbs according to their part in the phrase. The template of the verb triple is (*Verb*, *Constraint*, *Parameters*) where *Verb* stores the query term whose part of speech is verb; *Constraint* contains the query term whose part of speech is adjective or adverb; *Parameters* holds the other query terms. For a bookstore example, the service request string "Buy book java how to program inexpensive" could be parsed into several verb triples as shown in Table 1. In this example, the terms 'buy', 'book' and 'program' could be interpreted as a verb or a noun; 'java' is viewed as a noun; 'how' is an adverb, and the term 'inexpensive' is viewed as an adjective. The verb 'query' will be generated automatically if there is no verb interpreted in the input string. Here, we focus on atomic tasks, so we limit that at

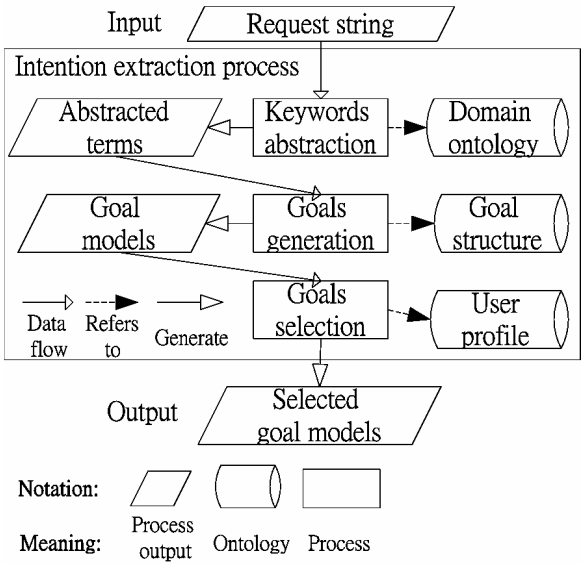

Figure 3 User Intention Extraction Process

Table 1 An example of verb triples

Verb	*Constraint*	*Parameters*
buy		book java how to program inexpensive
buy	inexpensive	book java how to program
program	inexpensive	buy book java how to
query		buy book java how to program inexpensive
query	inexpensive	book java program

most one term will be in the variables *Verb* and *Constraint*. We assume that the terms of *Parameters* should be seen as whole. The verb triples containing the verb 'book' will not be considered as candidate because the parameters will be broken into two parts: 'buy' and 'java how to program inexpensive.' The adverb 'how' will not be treated as a *Constraint* in the same reason.

Step 2 gives the word senses to the query terms of *Parameters* by referring to the application domain ontology.

In Step 3, the word senses of the terms in the *Parameters* are ranked by measuring the semantic similarity among terms. We assume that the word sense of a term is influenced by the context. For example, the term 'java' has three senses including island, coffee, and programming language. If the other terms in the *Parameters* include 'computer' or 'software', we can guess that the sense '*programming language*' is more possible than the others.

How to bind the terms in the entered string with correct concept is one of most important research issues in natural language understanding (NLU) [23]. In [24], the authors discussed that the extent to which shortest path lengths in *is-a* hierarchies can be used to measure conceptual distance. We adapt this idea for binding the terms in the *Parameters* to one of the concept nodes in the ontology.

In the domain ontology, if a term is polysemous, multiple paths might exist between two terms. In our approach, the senses of a polysemous term will be ranked by comparing the minimum length of concept distance. The distance of two concepts in the ontology is defined as follows.

Let C_1 and C_2 be two concepts represented by the nodes N_a and N_b in the domain ontology, respectively, with an *is-a* semantic relationship. A measure of the conceptual distance *CD* between C_1 and C_2 is given by

$CD(C_1, C_2)$ = minimum number of edges separating N_a and N_b. (1)

A term in the query string can be mapped to several

concepts in the ontology; we define the term distance *TD* between two terms as follows.

$$TD(T_1, T_2) = \min (CD(\boldsymbol{C}_i, \boldsymbol{C}_j)), \quad (2)$$

where $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ is any combination of the concepts mapped by any two terms.

Assume that there are *n* terms in the *Parameters,* the multiple terms distance *MTD* between term T_1 and other terns is defined as follows.

$$MTD(T_1, T_2 \wedge T_3 \wedge ...T_n) = \min(TD(T_1, T_j)_{j=2,3,...,n}) \quad (3)$$

We choose one term to compare with the other terms in the *Parameters* iteratively for deriving the *MTD*, the concept which mapped by *MTD* will be selected as the concept of the chosen term. If the terms cannot be found in any node of the ontology for mapping, the concept distance will be set as the maximum concept distance in the ontology.

3.2 Goal Generation

The goal generation step uses the annotated verb triples and the predefined goal structure to generate the candidate goal models. The *Verb* term in verb triple and synonyms of the *Verb* term will be used as index to search the goal structure and find the matched goal models. The *Constraint* term in verb triple is treated in similar way. Each verb triple corresponds to a goal model. The goal model generating process follows the steps listed below:

1. Generating candidate goal models by *Verb* term matching.
2. Pruning the redundant goal models by comparing *Parameters* terms.
3. Filling parameters fields of the candidate goal model.
4. Extending candidate goal models by matching the term in *Constraint*.

First step generates candidate goal models by matching the terms in the *Verb* field of the verb triple with the action field in the goal model. For example, assume that the application domain is the bookstore, if the term 'buy' or 'purchase' has been identified in the *Verb* field, the generated candidate goal models will be "Buy_Book", "Buy_Software", and "Buy_Toy", etc. because the action field of these goal models are the same.

Step 2 uses the relationship between the action and parameters to prune the redundant goal models. For example, assume that three verbs 'buy', 'book', and 'program' are identified in the entered string, goal models generated could be "Buy_Book", "Buy_Software", "Book_Flight", and "Travel_Planning" etc. If one term in parameters is identified as an ISBN of a book, all of the generated goal models will be eliminated except "Buy_Book".

In the research of NLU, the rule-based approach is a widely used technique in which a set of heuristic rules are designed by domain experts for mapping the content words in the parsed constituents into meaningful entries in the semantic frame [25]. The advantage of rule-based approach is easy implementation. However, due to extensive handcrafting and heuristic design in the approach, extending or shifting the old design to a new domain often involves significant time and effort on the part of the experts. The statistical natural language processing approach [26] can be adopted as an alternative. The shortage of statistical approach is that training the statistical approach based component need a lot of annotated corpora but hand-annotation of corpora may be costly. For this reason, in our approach, we construct a set of elimination rules for filtering redundant goal models. The elimination rules can be generalized as follows.

"**If** the parameters extracted from the verb triple do not belong to the attributes of the object in the candidate goal model

Then eliminate the goal model".

In step 3, since the goal models have been generated, the *Parameters* of the verb triple will be used to fill the parameters fields of the goal models and the identified attributes from the *Parameters* will be attached to the object of the parameters fields.

Finally, if there is any constraint-goal model related to *Constraint* term, the pruned candidate goal models will be attached with the constraint-goal model. For example, if the *Constraint* term is "inexpensive", a constraint-goal model "Get_Inexpensive_Item" will be attached to the candidate goal models.

3.3 Goal Selection

The goal selection step receives candidate goal models and selects a reasonable goal model by refereeing the user profile. The static records in the user profile will be used to filter the generated goal models, the dynamic records are used to rank the filtered goal models.

There are three steps in the goal selection phase.

1. Filtering impossible candidate goal models by referring the static records.
2. Ranking the selected goal models by referring the dynamic records.
3. Interacting with the user to determine a final selected goal model.

How to construct the dynamic records for ranking the candidate goal models is a research challenge. In this paper, we use periodic patterns to rank the goal models. Mining

the periodic patterns from a sequence of successfully performed goals is a sequential pattern mining problem [27]. The assumption of this approach is that in a system usage period, a successfully performed goal has some relationship with previously performed goals.

A pattern is a sequence S = s_1, s_2, ….s_n, such that *n* is the length of the pattern. Assume that a successfully performed goal sequence is {(g_1, g_2, G_1), (g_1, g_3, G_2), (g_4, g_5, G_3), (g_1, G_1), (g_1, g_3, G_2)} which G and g are used to represent the user-specific goal and extended system-specific goal. For example g_1 and G_1 might be "Get_Inexpensive_Item" and "Buy_Book". This goal sequence will be transferred into a pattern {A, B, C, D, B} if we use upper-case alphabet symbols to represent the pairs of the user-specific goal with its extended system-specific goals. Because the same user-specific goal might be extended with different system-specific sub-goals, this kind of goals is treated as different goals in the periodic pattern. For example (g_1, g_2, G_1) and (g_1, G_1) will be represented in different symbols A and D.

Figure 4 shows the periodic patterns extracted from the original pattern {A, B, C, D, B}. The first group of periodic patterns is extracted beginning with the first element of the original pattern and the second group is extracted beginning with the second element of the original pattern, for instances. The length of the periodic patterns is from 2 to the length of the original pattern.

Assume that four original patterns are {A, B, C, D, B}, { B, C, D, B}, {A, B, C, B}, and {A, B, D, E}, the table of dynamic record derived from these patterns is shown in Table 2. The system will use the weighted count sum (*WCS*) of the decomposed patterns to rank the input goal models. The *WCS* is described as follows.

$$WCS = \sum (P)^{(L-2)} * C \tag{4}$$

L and *C* are the length and count of the matched periodic pattern. *P* is a parameter for the weight of previous successfully performed goals and can be set by the system designer. In this paper, we set *P* to 2/3. For example, if the candidate goal models generated are "B", "C", and "D" and the previous successfully performed user-specific goal sequence are "AB," the *WCS* of these goals are calculated as follows.

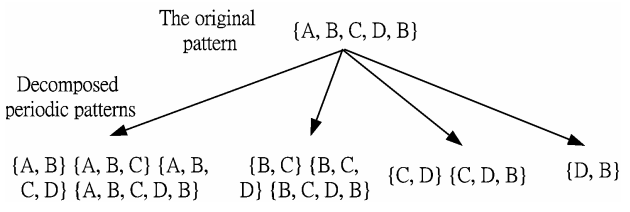

Figure 4 The periodic patterns extracted from the performed goal series

Table 2 The periodic pattern table of dynamic record

Length = 5		Length = 4		Length = 3		Length = 2	
ABCDB	1	ABCD	1	ABC	2	AB	3
		BCDB	2	BCD	2	BC	3
		ABCB	1	CDB	2	CD	2
		ABDE	1	BCB	1	DB	2
				ABD	1	CB	1
				BDE	1	BD	1
						DE	1

$$WCS_B = 0$$

$$WCS_C = (\frac{2}{3})^{(2-2)} * 3 + (\frac{2}{3})^{(3-2)} * 2 = 4.33$$

$$WCS_D = (\frac{2}{3})^{(2-2)} * 1 + (\frac{2}{3})^{(3-2)} * 1 = 1.67$$

After the ranking process, the ranked goals will be "C", "D", and "B".

The last step of goal selection is to display the candidate goal models. Because the goal model contains semantic information, the designer can base on the information to get better user-system interaction. If there are more than one candidate goal models selected, the goal model will be shown to the user and the user can modify the content of the goal model and select a proper goal model to represent user's intention.

4. Implementation

We apply the proposed techniques in implementing an intention-driven query tool (IDQT), the tool is designed for the intelligent service requester agent. The requester agent performs the tasks such as accepting the service request string from the user, extracting the user's intention from the request string, generating and executing a plan to satisfy the user's intention, interacting with the service broker to derive the answer, and processing and integrating the results from the service broker, etc.

For evaluating the proposed approach, we develop a query model for Web services which is inspired by Ouzzain's research [28]. The query model has three levels: the query level, the goal level, and the service level. In the query level, the system provides a query interface to the user for entering query string like using the search engine. The user's input query string is not a complete string. It is composed of different query variables and the position of the query variable is not limited and fixed. Three application domains are considered in our model: bookstore, vehicles scheduling, and flight booking. Three vehicles are considered in the domain: taxi, bus, and train. In our approach, we assume that the user has some background knowledge about the application domain and already knows which kind of services that the system can

perform. This assumption is a general assumption for many state-of-the-art natural language understanding approaches to avoid the user to overshoot or undershoot the capabilities of query interface [29]. The user interface of the intention extractor is shown in Figure 5.

In the goal level, the intention extractor receives the user's input string and performs the intention extraction process which proposed in Section 3. There are 21 goal models constructed in the application: 13 are actor-specific and 8 are system-specific. The intention extraction process uses the goal model to represent user's intention, and translates the goal model into a query document. Figure 6 shows a goal selection dialog for the user to select a goal model to represent the user intention.

In the service level, we suppose that a Web service broker agent like the broker mentioned in Sycara's research [22] can receive query documents which generated by the service requester and help us discover the related Web service and return the results. If the known Web service is unable to satisfy the user's request, the system will request the broker to discover and provide more Web services. Figure 7 shows a query book results which returned by Amazon Web Services (AWS) (http://www.amazon.com) in the bookstore application domain.

Figure 5 User interface of intention extractor

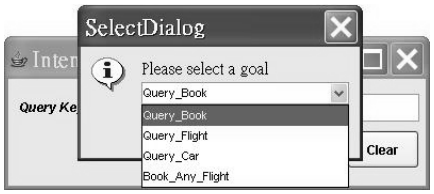

Figure 6 Goal selection dialog for goal selection

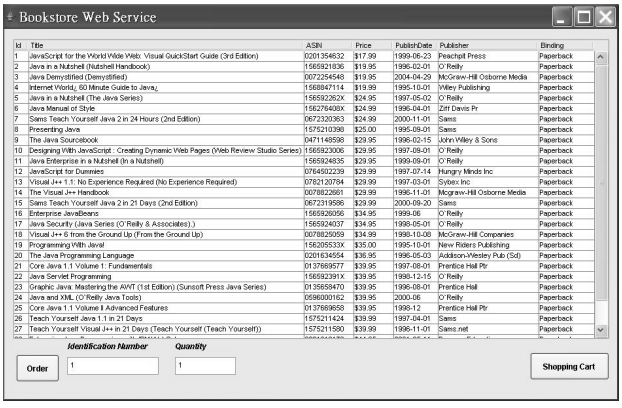

Figure 7 A result of query book goal

We evaluate the ability of IDQT by entering sentences in original requirement specification. Our experiment results show that the IDQT can correctly translate the user service request into related goal model if the request similar to the original string stored in the goal model.

5. Conclusion

We proposed a method to represent the imprecise user intention with goals, and an intention extraction process is also introduced. The background knowledge, domain ontology, goal structures, and user profiles are used to facilitate the intention extraction process.

By the proposed approach, the system can transfer the entered string into goal models. Based on the goal model, the system can perform a series of actions to achieve the goal. Once the goal has been achieved, the user intention is satisfied.

Acknowledgment

This research was supported in part by the Department of Industrial Technology, Ministry of Economic Affairs (Taiwan) under Grant 95-EC-17-A-02-S1-029 and by the National Science Council under grant NSC94-2213-E-194-010 and NSC95-2752-E-008-002-PAE.

References

[1] H. Wang, J.Z. Huang, Y. Qu and J. Xie, "Web services: problem and future directions," *ELSEVIER J. Web Semantics*, vol. 1, no. 3, April 2004, pp. 309-320.

[2] C.H.L Lee and A. Liu "Model the query intention with goals," Proc. of the USW2005, Mar. 2005, pp. 535-540.

[3] C.H.L Lee and A. Liu "Toware intention-aware semantic Web services," Proc. of the IEEE SCC2005, July. 2005, pp. 69-76.

[4] C. Breazeal, "Social Interactions in HRI: The Robot View," *IEEE Trans. Systems, Man*, and *Cybernetics*, May 2004, pp. 181–186.

[5] M. Paolucci and K. Sycara, "Autonomous Semantic Web Services," *Internet Computing, IEEE* , vol. 7, no. 5, Sept.-Oct. 2003, pp. 34-41.

[6] K. Sycara et al., "Automated discovery, interaction and composition of Semantic Web services," *ELSEVIER J. Web Semantics*, vol. 1, no. 1, 2003, pp. 27-46.

[7] J.M. Vidal, P. Buhler, and C. Stahl, "Multiagent systems with workflows," *Internet Computing, IEEE*, vol. 8, no. 1, 2004, pp. 76-82.

[8] N. Gibbins, S. Harris, and N. Shadbolt, "Agent-based semantic Web services," *ELSEVIER J. Web Semantics,* vol. 1, no. 2, Feb. 2004, pp. 141-154.

[9] R. Sreenath and M. Singh, "Agent-based service selection," *ELSEVIER J. Web Semantics,* vol. 1, no. 3, Apr. 2004, pp. 261-279.

[10] K. Sycara et al, "Distributed Intelligent Agents," *IEEE Expert,* Dec. 1996, pp. 36-45.

[11] I. Marsic, A. Medl, and J. Flanagan, "Natural communication with information systems," *Proceedings of the IEEE*, vol. 88 , no. 8 , Aug. 2000, pp. 1354-1366.

[12] S. Iba, C.J.J. Paredis, and P.K. Khosla, "Intention aware interactive multi-modal robot programming," *Proc. of the IEEE RSJ Conf.* 2003, pp. 3479-3484.

[13] Z. Chen, et al., "User intention modeling in web application using data mining", *World Wide Web: Internet and Web Information Systems*, vol. 5, 2002, pp. 181-191.

[14] P. Loucopoulos and V. Karakostas, *System requirements engineering,* McGraw-Hill, London, 1995.

[15] C. Fellbaum, *WordNet: An electronic Lexical Database*, MIT Press, Boston, 1998.

[16] I. Jacobson, *Object-Oriented Software Engineering*, Addison Wesley Longman, Reading, Mass., 1992.

[17] J. Rumbaugh, I. Jacobson, and G. Booch, *The Unified Modeling Language Reference Manual*, Addison Wesley Longman, Inc. 1999.

[18] J. Lee and N.L. Xue, "Analyzing user requirements by use cases: a goal-driven approach", *IEEE Software*, vol. 16, 1999, pp. 92-101.

[19] B. Chandrasekaran, J.R. Josephson, and V.R. Benjamins, "What are ontologies, and why do we need them," *IEEE Intelligent Systems and Their Applications*, vol. 14, no. 1, 1999, pp. 20-26.

[20] D.L. McGuinness and F.V. Harmelen, *OWL Web Ontology Language Overview*, Technical Report, W3C, 2004.

[21] H.M. Meng and K.C. Siu, "Semiautomatic acquisition of semantic structures for understanding domain-specific natural language queries," IEEE trans. on Knowledge and Data Engineering, vol. 14, no. 1, 2002, pp. 172-181.

[22] K. Sycara, M. Paolucco, J. Soudry, and N. Srinivasan, "Dynamic discovery and coordination of agent-based semantic Web services," *IEEE Internet Computing*, vol. 8, no. 3, 2004, pp. 66-73.

[23] J. Allen, *Natural language understanding 2nd ed.*, The Benjamin/Cummings Publishing Company, Inc., Redwood City, 1994.

[24] R. Rada, H. Mili, E. Bichnell, and M. Blettner, "Development and application of a metric on semantic nets," IEEE Trans. Systems, Man, and Cybernetics, vol. 9, no. 1, 1989, pp. 17-30.

[25] H.M. Meng and K.C. Siu, "Semiautomatic acquisition of semantic structures for understanding domain-specific natural language queries," *IEEE Trans. Knowledge and Data Engineering*, vol. 14, no. 1, 2002, pp. 172-181.

[26] C.D. Manning and H. Schutze, *Foundations of statistical natural language processing*, The MIT Press, Cambridge, 1999.

[27] W.G. Aref, G. Elfeky, and A.K. Elmagarmid, "Incremental, online, and merge mining of partial periodic patterns in time-series database," *IEEE Trans. Knowledge and Data Engineering*, vol. 16, no. 3, 2004, pp. 332-342.

[28] M. Ouzzani and A. Bouguettaya, "Efficient access to Web services," *IEEE Internet Computing*, vol. 8, no. 2, 2004, pp. 34-44.

[29] C.W. Thompson, P. Pazandak, and H.R. Tennant, "Talk to your semantic Web," *IEEE Internet Computing*, vol. 9, no. 6, 2005, pp. 75-78.

Biographies

Chiung-Hon Leon Lee received the Ph.D. degree in Electronic Engineering from the National Chung Cheng University in Taiwan in 2006. He is an assistant professor at Department of Computer Science and Information Engineering, ChungChou Institute of Technology, Taiwan. His research interests are in agent-based software engineering, Web services, knowledge representation, and fuzzy time series.

Alan Liu received the Ph.D. degree in Electrical Engineering and Computer Science from the University of Illinois at Chicago in 1994. He is an associate professor at Department of Electrical Engineering, National Chung Cheng University in Taiwan. His research interests in artificial intelligence and software engineering include knowledge acquisition, requirements analysis, intelligent agents, and applications in embedded systems and robotic systems. He is also a member of IEEE, ACM, and TAAI.

Yen-Ru Cheng received the BS degree in Electrical Engineering from the Tatung Institute of Technology in 1994 and the MS degree in Electrical Engineering from the National Chung Cheng University in Taiwan in 1996. He is a Ph.D. student in the Electrical Engineering Department at National Chung Cheng University. His research interests are artificial intelligence, software engineering, intelligent agents, requirements engineering, and embedded software engineering.