Study on the Automatic Composition of Document Service Based on Semantic and Multi-agent Method

Haiyan Hu[1], Xianxue Meng[1], Xiaolu Su[*]

[1] Agricultural Information Institute of Chinese Academy of Agriculture Science, 100081 Beijing, P.R.China. {huhaiyan, meng}@@mail.caas.net.cn
*Corresponding author, Key Laboratory of Digital Agricultural Early-warning Technology Ministry of Agriculture, Beijing 100081, P.R.China. suxiaolu@mail.caas.net.cn

Abstract: This paper proposes a method which composes document services over the Internet automatically based on multi-agent and semantic technology, and discusses the implementation of three document service functions (catalog browsing, document information browsing and document full version downloading) with that method. We use service delegate agent to encapsulate services on the internet; use service purchase agent to decide which service delegate agents should be used and then to generate a composition script describing how these services being purchased should be used; use service provider agent to call these services according to the composition script and merge the results, translate the merged result into the system predefined format, finally return it to the requesting user. By doing so, service provider can simply composite existed services on the internet and provide full functional service.

Key words: Document service, Automatic composition，Semantic Web, Multi-agent

1 Introduction

Information services are becoming more and more specialized. There are many standalone indexing, searching, and online storage services on the Internet, provided by many providers respectively.

Traditional online library is a multi-function system, provides catalog, indexing, and electronic document downloading services. This type of system appears obsolete today. Regarding that there are better search services like Google, better storage services like Amazon S3[1], etc, why not simply use them but do everything self? Develop an all in one service is quite a big challenge, even can be done, it must not be as cost effective as those specialized services from above providers, and can not achieve the high quality standard of them. Composing the existing specialized services are the best way to build online library on today's Internet.

The core business of online library service provider is to give users the documents they want. To achieve this purpose, four fundamental functions are necessary, they are, document catalog and indexing, document searching, document metadata offering, and document full-text download. As the owner of the documents, providing

document's full-text version is the only job must be done by provider himself. If everything is ideal, the online library provider can use online storage service stores his documents, and create a database stores the download links and basic metadata of these documents, and need do nothing else but compose 3rd party services on the web. The basic metadata should include a unique identity for each document, and some candidate identity fields such as title, publish date, author, etc. These fields may be used by some service agents to identify documents, preventing duplication of document metadata from multiple sources.

Semantic and multi-agent based Web Services automatic composing have became new hot spot for Web Services [2-6]. This article discusses the research based on multi-agent and semantic technology which automatically composes directories, document information and document texts in document services.

2 Technical Considerations

The best way to implement service composition system is multi-agent architecture. Using one single agent represents a service. Let these delegate agents promotes his service to the purchasers in a marketplace. After a process of competition, reasonable service consumption relations are established, from which the service cooperation framework can be derived out.

BDI model [7] is a multi-agent architecture that is widely used. A BDI model has three basic parts:

(1) The Belief. Belief is a set of theories about the world, which include the knowledge about the environment, the knowledge about other agents the agent may contact with, and the knowledge about the agent himself. It is the objective basis of the decision-making of the agent.

(2) The Desire. Desire is the motivation of agent, which represents the state the agent willing to achieve and to hold.

(3) The Intention. Intention is the current target of the agent. It is the most urgent desire among all the desires of that agent. It represents the direction of the agent's mind, and guides the actions the agent currently taking.

The system takes JADEX as the basic framework of agent's interaction [8]. Regarding that JADEX do not support ontology inference, it must be extended to meet our requirements. We created a new interface named "IMOWLBelief", two new classes named "MOWLBelief" and "MTypedOWLElement". The class "MTypedOWLElement" defines the attributes of the Ontology and their getters and setters, encapsulates OWL operations. The class "MOWLBelief" is extended from the class "MBelief". To make these newly defined classes accepted by JADEX framework, the JIBX binding file (binding.xml) must be changed. After doing that change, believes can be presented using OWL.

3 Architecture of Service Composition

Each kind of service is composited by a service provider agent, a service purchase agent, and a number of service delegate agents. Each service delegate agent delegates a single service source on the web. Service delegate agent 'knows' every detail of the service it delegated, in another word, that service delegate agent encapsulates its service. The desire of service delegate agents is to sell its service to service purchase agent as much as possible. A service delegate agent can trade with more than one service purchase agents, sells different functions to each of them. For example, a service delegate agent delegates a service source which provides catalogs as well as document metadata, it can sell catalog function to catalog service purchase agent as well as sell metadata function to metadata service purchase agent. Each service purchase agent serves for a single function, so in our system there should be four service purchase agents, work for document catalog, document search, document metadata, and document download respectively. Like service purchase agents, service provider agents are also dedicated to those four functions respectively.

Service purchase agents decide which service the service delegate agents delegated should be used in service composition. While done, the chosen services can be composited to serve users. Service purchase agents will generate a composition script to describe how these services it purchased should be used. A service provider agent is responsible to calling these services according to the script generated by the service purchase agent who work for the same function as it does, and then merges the results, and translates the merged result into the system predefined format; finally return it to the requesting user. The script generation will consume a large amount of computation resources, so it shouldn't be done frequently.

4 Functions of Service Composition

A typical document service provider provides catalog browsing; document searching; document information browsing; and document full version downloading. Regarding that the search service may returns a huge amount of data, and it is not fit for compositing, so we will only discuss composition of the other three functions.

4.1 Catalog Browsing

Catalog browsing service needs a catalog. In traditional way, catalog is created manually with some kind of classification method. Regarding that there may be different users expecting different classification methods, and the best practice is to provide each user what he needs. Creating catalogs manually is an exhausted task, and is apparently not cost effective. If we can reuse the catalogs currently exist on the web, it will become practicable to provide multiple catalogs with different classification method. The catalogs currently on the web are not created for your needs, so those catalogs must be customized.

In catalog service, the catalog itself is the belief of service delegate agent, and the desire of service delegate agent is to sell the service it delegated as much as possible. The service purchase agent will assess each delegate agent and calculate how much its catalog can complement the catalogs already purchased. The service purchase agent will purchase the services which can contribute the most content to current catalogs.

At the initial state, the service purchase agent does not purchase any service. The service delegate agents are added into system every time a new service is discovered. After created, each service delegate agent updates service information periodically to handles the possible changes of that service.

The periodical update of service information may introduce changes to existing service, and those changes may affects the purchase contract already signed. When changes occurred, the service delegate agent must notify the service purchase agent. While notified, the service purchase agent will expire the currently purchase contract, assess the changed service again, together with all other services not yet assessed, and decide how to purchase from all available services. Then the service purchase agent must regenerate service composition script.

The service purchase agent sorts the service delegate agents by the entry numbers in the catalogs their services provides, in descendent order, assesses them one by one. The catalog provides the more useful entries will be scored higher. After all services are assessed, the service purchase agent starts purchase from the service with highest score, until the service scored lower than threshold. The services with zero score will be excluded, until they got some changes and then triggered reassessment processes. The services scored above zero but lower than the threshold will not be purchased in the first round, and will be reassessed in the second round.

The second round purchase based on the result of first round purchase and composition. All the services scored above zero in the first round but not been purchased will be rescored in this round. There is no threshold of score in the second round, if a service is rescored more than zero against the new basis, it will be purchased. The initial score in the first round will be recorded for future use, but not the second round score.

The composition operation uses some simple rules. For example, there is a rule looks like this: “if equalsIgnoreCase (X.entryName, Y.entryName) and notEmpty(intersect (X.parents, Y.parents)) then X sameAs Y ”; or “if equalsIgnoreCase (X.entryName, Y.entryName) and isEmpty(X.parents) and isEmpty(Y.parents) then X sameAs Y ”. The catalogs the services provided may have multiple language versions. The same catalog entry or the same article title in different languages can be linked up automatically. The service purchase agent handles the merging of catalog entries, generates a script to tell the service provider agent how to composite these catalogs. When user requests, the service provider agent retrieves and composites catalogs according to the script generated by service purchase agent, then extracts out the part which meets the user’s requirement, converts them to standard format, and return to the user. Let the service purchase agent handle catalog composition and the service provider agent do the real composition job is based on such considerations: the service purchase agent can present each catalog entry as an OWLClass, and each document as OWL Individual, in its belief, to convenient inference of catalog entry relations; the service provider agent can treat all catalog entries and documents as individuals at the same time, to

convenient inference of merge operation. These two believes can not be mixed up, because there is some thing in one is represented as class and in another is represented as individual; if put them together, the merged belief will violate the OWL DL constraint, and can not be used in any inference. The service provider agent reads the script generated by service purchase agent, and revises its own belief according to that script. Considering mapping multiple languages to each other, the link between them needs a new predicate, to express that they can be translated to each other. Two language versions of one document are handled as two linked documents; two language versions of one catalog entry are also handled as two linked ones.

4.2 Document Information (Metadata) Browsing

Metadata of one document can be acquired from multiple sources, and they may not consist with each other. Data from multiple sources may have multiple structures, and these structures must be identified correctly and mapped to a standard structure. Some of the data is acquired from online libraries, this part appears well arranged, there is some other data come from none academic sources such as online book stores, may be organized according to their business needs, and not comply with their logical meanings.

There is still some fragmental data distributed in online forums, bokees, and other online information sources. This part of data often published personally, and just organized by the intendancy of these people, but may have some in-depth opinions. Because this part of data often appears in html pages, and can not be acquired from some kind of services, it can only be collected from search engine service. But search engine services are never designed to provide such data, and they may return many irrelevant data as well as some useful data, some the results the search engine returned must be filtered first.

Document metadata service composition is not the same as catalog service composition. The service purchase agent must organize the service delegate agents by each document, and then judge which source should be used for each document. There is no standard schema which all service delegate agent comply with. Fusion on attributes can only be done dynamically using inference.

There are 3 steps in the fusion process of each attribute:

(1) To judge whether current attribute can be merged with some attributes of another object.

(2) If two attributes from two objects can be merged, whether their value equal or compatible to each other.

(3) If both values not compatible to each other, determine which value should be kept.

There are two occasions that two different attributes can be merged. One is that they have the same meaning: attributes with the same meaning may have different names, for example, ‘title’, ‘topic’, and ‘caption’ may be the same, and can be merged, another is that one attribute can be deduced from another, for example the attribute ‘age’ can be calculated from the attribute ‘birth date’ but not vice versa, so the attribute ‘age’ can be removed. In the first occasion, a synonym table can handles the mapping well; in the second occasion, there must be a conversion function fit for that

calculation, if such a function can be found by the service purchase agent, the merge will be allowed.

While mapped, compatibility between the values of attributes to be merged must be assessed. Because values are often digits, and mathematical calculation often needed, the assessment can not be completed by inference, but can only by dedicatedly designed functions. These functions are owned privately by the service purchase agent. The values to be assessed must have the data type which is acceptable to the assessment function. If some of the values are not in proper data type, data type conversion must be taken first, fail in conversion means not compatible. If compatible, the one with more accuracy will be kept, if not, there should be a method to choose the applicable one among them. If a value of attribute has more than one source, the value with the support of most of sources has the highest reliability; the less frequently changed values have higher reliability. For example, the 'age' attribute must be changed every year, but the 'birth date' attribute needn't do that change, so 'birth date' is more reliable than 'age'.

To get document metadata from search engine, the system need calling the search service with the title of that document as keyword. The result returned from search service includes the URL of target web page and a brief abstract of it. The search engine often break the title into words before search, so the result returned contains many irrelevant pages which just have some of the words we want to search. These entries are useless and should be eliminated first. While filtered, the remaining entries will be examined one by one, the target web page will be washed out peripheral contents and html tags, only core text content will be kept. The core text will be added to the description of that document.

4.3 Document Full Version Downloading

Document download service provides the binary versions of documents to users. There are many online store services on the web, they are often more reliable and accessible to common users than the server maintained by document provider itself. Most of the online storage services need user login first. Using multiple online storage providers to balance data flow and secure availability is also a good idea. So there is also motivation on composition of document storage services.

In document online storage service composition, the return of the composited service is the binary file of document, it needn't further processing, and can return to user directly. So in this part service provider agent is not necessary. Service purchase is also simpler than the other two parts. The service purchase agent will give higher privilege to the services which can provide storage of larger number of documents, and which has the higher download speed. The privilege is dynamic, if a service already has a number of live calling, its privilege will be lowered to make it less possible to be called again. This dynamic privilege system will balance the load.

5 Conclusion

Traditional way of “all in one” document service system has its downside: waste resources on developing already exist functions on the internet; make the providers lost focus from their core business; and create obstacles to integrating them together.

This paper proposes a method a of automatically compositing existed document services based on multi-agent architecture and semantic technology, and discusses the composition of three common document service functions, and provides a solution to solve the problems traditional document services have.

Acknowledgement

The research was supported by the 12th five-year national key technology support program, super-class scientific and technical thesaurus and ontology construction faced the Foreign Scientific and technical literature（2011BAH10B01）and special fund of basic commonweal research institute project of information institute of CAAS.

Reference

1. http://aws.amazon.com/s3/
2. Lian, Wenjuan;Liang, Yongquan;Zeng, Qmgtian. “Integrating semantics and agent technology to automatic Web service composition”[C]，IEEE Symposium on Web Society, 2nd Symposium on Web Society. pp.201-206 (2010)
3. SANDEEP KUMAR;NIKOS E. MASTORAKIS. “Multi-Agent Negotiation based Semantic Web Service Composition Models”. WSEAS International Conference on Software Engineering, Parallel and Distributed Systems (SEPADS'10). Recent Advances in Software Engineering, Parallel and Distributed Systems, pp.214-223 (2010)
4. Hao Tian; “Agent-based semantic web service composition model”. Second Pacific-Asia Conference on Circuits, Communications and System. v.2, pp.67-70 (2010)
5. Mohamad El Falou, Maroua Bouzid, Abdel-Illah Mouaddib, Thierry Vidal, "Automated Web Service Composition: A Decentralised Multi-agent Approach,", 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, wi-iat, vol. 1, pp.387-394 (2009)
6. Nouredine, G.; Hassina, S. “Composition of web service based multi-agent”. International Conference on Multimedia Computing and Systems (ICMCS 2009), pp.51-55 (2009)
7. http://baike.baidu.com/view/1258913.htm
8. Lars Braubach, Alexander Pokahr, Winfried Lamersdorf. „Jadex: A BDI Agent System Combining Middleware and Reasoning”. Software Agent-Based Applications, Platforms and Development Kits, pp143-168 (2005)