Toward a Flexible Human-Agent Collaboration Framework with Mediating Domain Ontologies for the Semantic Web

Yuri A. Tijerino and Muhammed Al-Muhammed

Brigham Young University
Computer Science Department
TMCB 2228, Provo UT 84602
{yuri, mj47}@cs.byu.edu
http://www.deg.byu.edu/

Abstract. This paper introduces a human-agent interaction framework that allows human and computational mechanisms to interact in a seamless manner to collaboratively perform problem solving tasks in the Semantic Web. The framework is based on a combination of services supported by intermediate domain ontologies and flexible mapping mechanisms that map agent and human internal representations and external communication protocols. The framework is based on a solid conceptual foundation and can be extended to also incorporate other service-based computational mechanisms such as web services. The central focus of the framework is to encourage contribution by anyone or anything, that might possess useful information, knowledge or expertise for the successful completion of collaborative problem solving tasks. Yet, the problem solving participants are not required to go through a rigorous ontological commitment or have a common communication protocol in order to collaborate.

Key words: Human-Agent Interaction, Semantic Interoperability, Ontology Mapping

1 Introduction

In Tim Berners-Lee original Web proposal to CERN [1] the Web was envisioned as an amalgamation of what we know today separately as the Web and the Semantic Web. Because of practical, technological and commercial reasons, however, the human-centered Web became the focus of attention for most researchers and for most commercial applications. However, the ever increasing content in the WWW, or the Web as it is commonly called, is so large that it has created the problem of information overload for most people. The Semantic Web [2] promises to alleviate this problem by allowing computational mechanisms to publish, find, share, understand and use content to help people cope with information overload and take over tasks that require little or no human interaction. However, as much research is being devoted to the many interesting problems of the Semantic Web, very little attention is being paid to allowing people to fully take advantage of the computational power that will become available in the Semantic Web. As a matter of fact, the human-centered Web and the computation-centered Semantic Web exist today in almost two parallel universes with Web users working completely independently of computational mechanisms in the Semantic Web. Perhaps the reason for this is that it is as difficult for humans to understand information published in the Semantic Web as it is for computational mechanisms to understand information published in the Web.

Some researchers have recognized this need for interaction and are proposing novel and unique approaches to allow human users to interact with semantically annotated web resources. Most notable is the CS AKTiveSpace system [3] at the University of Shouthampton, which allows users to "explore" as oposed to browse information about computer science research in the UK. The system is very sophisticated, yet very elegant in the way it visually presents information collected in about 10 million RDF triplets. The main difference between the approach presented in this paper with the one implemented in CS AKTiveSpace is that while CS AKTiveSpace uses a common domain ontology in combination with a miriad of harvesting and screen scraping techniques to store the informaiton in a central repository, our approach allows agents[1] and humans to collaborate in the gathering of the information in a decentralized manner.

Another notable approach is that proposed by the Mangrove system [5] at the University of Washington. This is a system that deals directly with the problem of populating the Semantic Web with useful ontology-based annotations by non-technical people. The approach is interesting because not only it facilitates annontation of HTML pages by "ordinary" people, but it also allows the people to immediately benefit from the annotations. What makes this approach related to the one presented in this paper is that there is human interaction with the Semantic Web pages in a meaningful manner. However, the system doesn't go far enough to promote a post-annotation, two-way interaction mechanism in which both humans and computational mechanisms such as agents collaborate in problem solving efforts. This paper proposes a mechanism that addresses this particular issue.

The vision presented in this paper for this problem is to create inter-operation mechanisms that allow human users to interact with computational mechanisms in a seamless manner by bridging the gap between their internal representations using a mediating domain ontologies.

To illustrate the proposed framework of human-machine collaboration in the Semantic Web, the paper focuses on knowledge management like problems. On one hand, this is the problem people encounter when searching information on

[1] Bradshaw [4] defines a software agent as "a software entity, which functions autonomously and continuously in a particular environment, often inhabited by other entities... an agent that inhabits the same environment with other agents... [is] able to communicate and cooperate with them."

the Web to solve a particular problem or to accomplish a particularly interactive task. On the other hand, this is also a problem that computational-centered mechanisms of the Semantic Web have to deal with, since one of the goals for computational-centered mechanisms such as software agents or web services is to find resources and possibly other software agents or web services that might have needed information or knowledge, or that might be able to provide a particular service. From this perspective, the knowledge management problem that a person has to deal with on the Web and the knowledge management problem that a computational-centered mechanism encounter on the Semantic Web are not that much different. We will discuss this knowledge management problem using as an example a computer shopping application. In this application, a user wishes to purchase a computer and gives a very sketchy set of specifications for the computer. The system finds an agent that specializes in purchasing computers and "knows" what other agents can provide assitance. Before an order is made for a computer, the agent interacts with the human user and other agents to further refine the specifications.

The remainder of this paper is organized as follows. The next section describes the conceptual and philosophical motivation for making possible knowledge sharing between humans and machines. Section 3 describes the proposed framework for allowing human-agent interoperability in the Semantic Web. Section 4 illustrates the most important parts of the framework through a simple example. Section 5 presents some cloncluding remarks.

2 Knowledge Sharing Between Humans and Machines

Knowledge can be viewed as a commodity, which is produced, consumed, refined, stored, retrieved, shipped and recycled in a continuous loop in which both humans and machines play an important role. In other words, knowledge does not exist in an isolated form. It exists in an Economy of knowledge[2]. In early work on so-called expert systems [8], it was noted that knowledge was not something that a particular individual had in a closed system, but instead existed in a distributed manner among many individuals. The task of a so-called "expert" was not only to retrieve pre-stored internal representations of knowledge, but also to collaborate with other experts to create that knowledge through practice. This paper expands this view to include other computational mechanisms such as software agents and web services[3] in the list of experts. In addition, it proposes that by expanding this view, not only can people and machines collaborate in the context of explicit knowledge, but also to some degree in the context of tacit knowledge.

[2] According to Paul Romer [6, 7] knowledge has become the third factor of production in leading world economies, with labor and capital being the first two.

[3] Although the readily could support web services because the framework is built on web service technology, we will focus on human and agent interoperability.

2.1 The Tacit Versus the Explicit Dimension of Knowledge

According to Polanyi[9], tacit or implicit knowledge is the internal representation that is found in an individual's mind. Nonaka[10, 11] further expands this definition by adding that tacit knowledge is that knowledge that cannot be easily verbalized. According with Polanyi, this knowledge is normally gathered through observation, individual experiences, and from interactions with other people. It is also improved through practice, drill, and visualization. Machine learning methods based on non-symbolic representations, such as artificial neural networks, genetic algorithms and Bayesian classification are known to learn from observation and cannot easily explain their conclusions.

Explicit knowledge — also known as objective, verbal, declarative or articulable knowledge — is knowledge that can be communicated in descriptive form. Usually explicit knowledge can be found in the form of ontologies, logic representations, rule-based systems or other formats that are capable of generically transforming "information" into meaningful explicit descriptions. Notice that we do not consider explicit knowledge "information" which is found in the Web such as written text or other media form like video or audio. Textbooks, maps, DVDs, etc. are all examples of information provided in explicit formats, but they are not examples of explicit knowledge. Some machine-centered mechanisms are also capable of explictly knowing. For instance, decision trees, rule-based systems and ontology-based systems can explicitly represent knowledge and explain why some dicision was made. The Semantic Web is the most ambitious and sucessful distributed effort to annotate large bodies of information with explicit knowledge to make it available to both humans and machines. In other words, while the Web is the largest repository of HTML-based *information*, the Semantic Web is the largest repository of ontology-based explicit knowledge.

So why is this all so important? Because the ultimate goal of the Semantic Web is to allow machines to find, share, combine and understand knowledge in the Web way, i.e. without central authority, with few basic rules, in a scalable, adaptable, extensible manner capable of supporting both explicit and tacit knowing.

Nevertheless, to truly take advantage of the information in the Web, available to humans, in combination with the "explicit knowledge" in the Semantic Web, available to machines in the form of ontologically annotated resources, we must devise mechanisms that allow humans to collaborate with machines in a symbiotic partnership. This symbiotic partnership will allow humans access to machines, not as tools, but as partners in knowledge formation. In a similar way, it will also allow machines to benefit from the better ability that humans have to process information into true tacit knowledge.

2.2 Human-Machine Collaboration

Although advances in technologies such as machine learning and artificial intelligence attempt to dote computational mechanisms with quasi-human capabilities, the promise of true machine intelligence is far from being realized. However, these technologies have made it possible to give limited independence to computational mechanisms that perform limited automated tasks with little or no human supervision. The most well-known of these mechanisms are software agents. It is mostly for these agents that the Semantic Web is being designed and created. However, these agents, no matter how intelligent they are portrayed to be, are still tools to serve some purpose useful to people, directly or indirectly. It is the hypothesis of this paper that close interaction between these agents and humans is necessary to make these limited mechanisms more efficient in the exploration, exploitation and creation of knowledge. In other words, human users should be able to interact with agents to provide direction and feedback on their activities, and by so doing be able to create more knowledge from those interactions. Furthermore, agents should be able to interact with humans to inquire information needed to perform these tasks, and by so doing borrow human knowledge needed for more efficient task execution. This interactivity is somewhat different from the paradigm of machines being used exclusively as tools because in here the interaction is bi-directional, thus humans can also be perceived as tools to agents, though in a more limited sense.

3 Human-Agent Interoperability Framework

It is obvious from the discussion that there needs to be a transparent framework that allows interaction between agents and agents and between humans and agents. In order for the framework to be transparent we impose two constraints: 1) the framework should not require a strong ontological commitment and 2) nor should it require a strong pre-agreement between agent and agents or agents and humans on the communication format.

With no strong ontological commitment we mean that the agents and humans do not need to agree on a shared ontology and could have their own internal representation (e.g., a local ontology), which does not need to be the same shared representation. However, they need to be able to readily map their internal representations to a mediating ontology. What makes this a weak ontological commitment, as opposed to a strong one, is that the agents or humans do not need to know *a priori* what the mediating representation is.

With no pre-agreement on communication format, we mean that both the agents and the humans do not need to compose messages in a way that is not natural to the way they do things already. However, in order for this to be posible there needs to be a mechanism that translates the messages between agents and agents and humans an agents in a transparent manner by using the mediating ontology during an initialization phase to understand their message format.

Fulfilling these two requirements would allow for agent-agent and human-agent interoperability on the fly. According to Uschold[12], “the holy grail of semantic integration architectures” is to “allow two agents to generate needed mappings between them on the fly without a-priori agreement and without them having built-in knowledge of any common ontology.” Here we not only attempt to allow agent-agent interoperability in that manner, but also human-agent interoperability, which is also necessary to bring about the Berners-Lee[1] original vision of the Semantic Web. We do not claim that the framwork presented in this paper meets these two requirements in full because it has several limitations.

One limitation of this framework is that it only allows for interoperability of humans with computational mechanisms. In other words, the computational mechanisms such as agents are the target of the interaction as opposed to direct interaction of humans with the semantically annotated pages on the Semantic Web, which is the approach of Shadbolt et al [3] and McDowell et al [5] in the systems we described earlier. Interaction between humans and Semantic Web based computational mechanisms, however, is very important and more natural than direct interaction of humans with semantically annotated pages, because the Semantic Web is originally intended for computational mechanisms and not for humans. On the other hand, if humans can interact more readily with those computational mechanisms, it should be possible for them to performs tasks such as mining the web for tacit or explicit knowledge in the form of specific answers to questions or solutions to real-world problems as opposed to finding information that can be used to obtain that knowledge as is the case of today's Web. For instance, instead of finding multiple semantically annotated web sites where one can book a flight, buy a book or configure and buy a computer, through a "Semantic Web Explorer" such as the CS AKTiveSpace system; it should be possible instead for users to interact with agents that can find those sites and then perform those transactions directly on behalf of and through interaction with the human user.

The framework described in this section meets these challenges head on and consists of four main components: 1) a directory service that allows seamless registration and search of human and agents, 2) a message mapping infrastructure that translates between internal representations of both agents and humans, 3) a communication service that handles agent-agent and agent-human communication, and 4) a trust and security infrastructure in which the agent-agent and user-agent interactions takes place.

3.1 Directory Services

The framework adapts the matchmaking algorithm based on OWL-S[4] with UDDI as proposed by Paolucci et al [14], to allow registration and search of services provided not only by web services, but also to include services provided by agents and human users. However, since this architecture is fairly well-undestood on the context of web services, we will focus here on how to discover, register and search services provided by human users and agents. So for instance, an agent whose task is to shop for products on the internet can be registered in the directory services and advertise that it is a shopping agent. An user who interacts with the agent to buy a product, say a PC, is registered temporarily

[4] OWL-S evolved from DAML-S and is now the basis for SWSL, or the Semantic Web Services Language[13].

in the directory as one who can dissambiguate the request should the agent find them ambiguous.

To allow agents and human users to interoperate on the fly with other agents or human users, they first need to be registered in the UDDI directory. We now describe this process.

Agent Registration: Al-Muhammed [15] presents an agent interoperation method that does not require a shared ontology or pre-agreed message format to allow communication between agents. This approach consists of defining local ontologies for the agent based on an extended data extraction methodology that constructs the local ontology from clues provided in the agent code and the code comments. This methodology has worked fairly well in previous efforts to extract data from structured information sources [16–19] and to dynamically construct ontologies from tables [20].

Conceptually, agent code can also be considered to be structured information source and is treated in a similar way to other structured information sources such as tables or lists. This approach has its limitations, since it requires access to the agent's code, which is not always available. It also assumes that the code is well formed, documented and written with meaningful names for class names, methods and variables, which is not always the case either. Nevertheless, experiments showed that if the assumptions hold true this approach is viable. We use this approach to first extract the agent's ontology. Then we use the agent's ontology to discover the services and map them to OWL-S. Once the agent's ontology and the agent's services have been discovered they are registered in the UDDI directory for other service requesters to see, including human users. This approach is an extension of that developed by Al-Muhammed [15, 21], but in his approach no central service directory is proposed.

Rather than requiring agents to share ontologies, we provide our framework with automated mapping to agent-independent, domain-specific ontologies such as those readily available in the web in the form of DAML, DAML+OIL or OWL. To accomplish this, we utilize the same mapping techniques described by Al-Muhammed [15, 21] to map the agent's services to OWL-S and make them available through the UDDI directory in a similar way as that described by Paolucci [14] for web services. When an agent makes a request in its native communication language, say a Java method call, it is translated using the particular domain ontology, say one written in DAML, and then a service matching the request is found using OWL-S or SWSL service descriptions in the UDDI directory. If the service is that of another agent, then the request is translated to its local ontology and a response is sent back to the requesting agent following the same translation process.

In order to generate local ontologies for the agents, an *Agent Ontology Extraction Engine* uses preconfigured recognizers modeled after data frames [22], which are snippets of knowledge as to how to recognize instances of specific concepts in an ontology. This process is illustrated in Figure 1. This agent ontology includes the names of concepts the agent uses such as class name, parameter names, variable names and the data types of the concepts. To compensate for not having a shared ontology, the framework maps the the agent ontologies of all registered agents to one of the various domain ontologies the framework maintains[5]. This process is described further in Section 3.2.

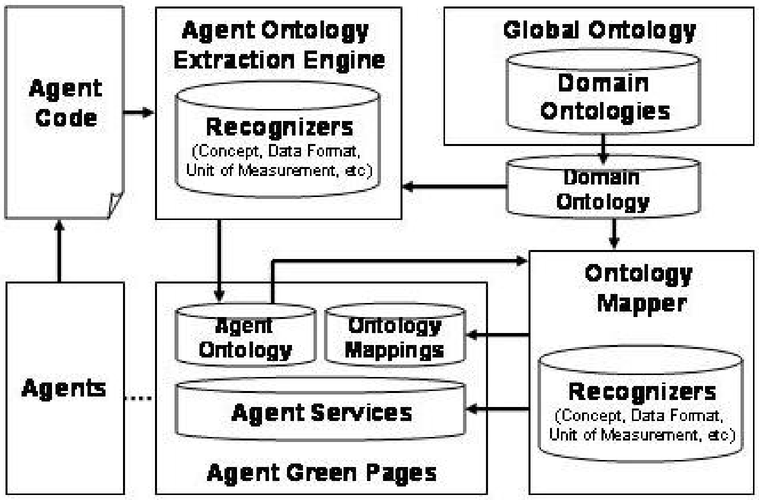

Fig. 1. Agent registration using an intermediate domain ontology and general-purpose concept recognizers.

User Registration: User registration can occur in two different ways. A user might choose to enter information in the UDDI directory and following the OWL-S format to also advertise the basic services provided. Of course this might be a bit unrealistic for most users including experienced ones. A second way is to temporarily register user when she sends a service request that might require interaction with the service provider. We will go in more detail on this process in a later example.

3.2 Ontology Mapping Services

The domain ontologies we propose in our framework consist of two components: 1) a conceptual model of the domain, which describes the domain in terms of con-

[5] We emphasize that there is a major difference between our approach and a shared ontology approach, because an agent's developer needs to know nothing about any other agent's ontology, nor do they need to know anything about the domain ontology. It is the ontology mapper that does the work.

cepts, relationships, constraints and axioms, and 2) recognizers[6] that currently are based on regular expressions[7] to help us recognize concepts, data formats and units of measure from the agent's code. We have experimented with this type of recognizers using various domains with very promising results [17, 23, 24].

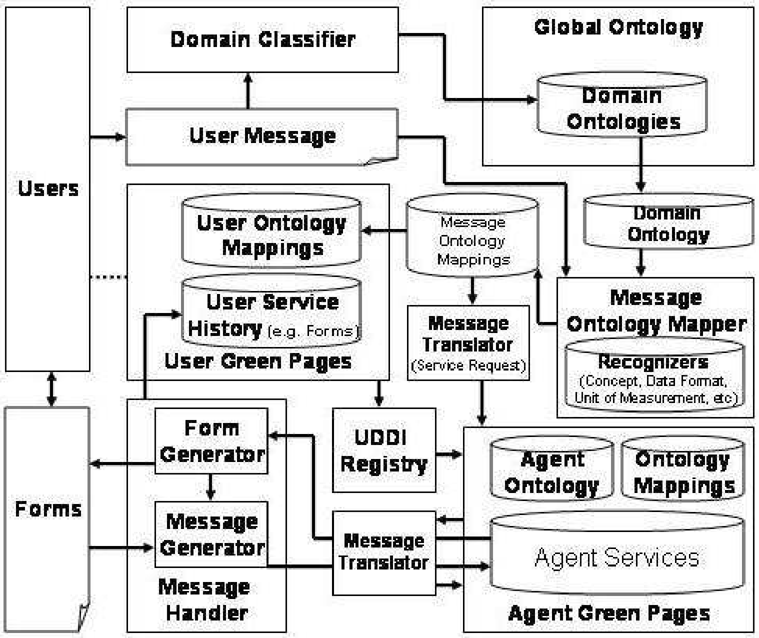

Fig. 2. Message registration and user-agent interaction.

Domain ontologies can either be manually constructed based on ontology engineering techniques [25, 26] or on automatic ontology generation techniques being developed by the authors [20, 27].

Agent Ontology Mapping: As Figure 1 illustrates the *Agent Ontology Extraction Engine* parses the agent code, finds its services, and expresses them in

[6] Recognizers are currently modeled after Data Frames [22] which original purpose is to allow the extraction of concept instances found in structured or unstructured content.

[7] Although data frames are currently based on regular expressions there is ongoing research to make them smarter by incorporating other recognition techniques such as decision trees, neural networks and even latent semantic indexing.

an agent-independent way, as proposed by Al-Muhammed [15, 21], in the green page corresponding to the agent.

The *Ontology Mapper* then uses the *domain ontologies* to translate internal representations of both agents and users into local representations. It uses recognizers to handle the mapping between agent internal representations and human generated messages. In this framework it is not required that all agents and messages be described in terms of the domain ontology, but instead depend on the recognizers to conciliate the differences between their representations by recognizing concept equivalencies and instances. Concept equivalencies deal with equivalent concepts in different ontologies, in this case the domain-specific ontology and the agent-specific ontology. Concept instances are related to information found in messages that can be thought of as instances of a particular concept in the domain specific ontology.

User Message Ontology Mapping: When a user sends a message and the framework has identified a domain for the message, an appropriate domain ontology from the *Global Ontology Repository* is used to allow the *Message Ontology Mapper* to employ recognizers associated with the particular domain ontology to establish message ontology mappings. This results in several mappings of the message to the ontology to occur and be placed in the *Message Ontology Mappings*. Thus now both the user messages and the agent services are described in terms of the intermediate domain ontology.

Once the message is parsed and the mappings are placed in the *Message Ontology Mappings* repository, two things need to happen: 1) the *Message Ontology Mappings* are placed in the *User Ontology Mappings* repository in the corresponding *User Green pages*, and 2) the *Message Ontology Mappings* are transferred to the *Service Request Generator* which generates a service request for the UDDI registry as shown in Figure 2.

3.3 Communication Services

Rather than having agents deal directly with incoming messages, our framework provides for automatic mapping of incoming messages, either agent or user generated, to an appropriate service in the UDDI registry. Then a *Message Translator* makes a mapping between the service and incoming messages by 1) parsing a message and identifying its type and its input and output parameters, and 2) matching the type of the input and output parameters of the message with those in a service provided by an agent or user.

User-agent communication is not much unlike agent-agent communication. The main difference is that there is one more formating step necessary so that humans can generate and reply to agent requests. Figure 2 illustrates this kind of user-agent interaction.

If a user sends a message, it is first parsed and mapped to the appropriate domain ontology and stored in the *User Ontology Mappings* as described in Section 3.2. Then the message is translated and an appropriate agent which can handle the request is identified through the UDDI register. The message is then sent to the agent, which processes the message with the appropriate service. If the agent needs additional information from the user, the agent then composes a message and sends it to the message translator, which translates it and then converts it into a human-readable form using the *Message Handler*'s *Form Generator*[8]. A form is then presented to the user, who takes the necessary action and sends it back to the *Message Handler*. The *Message Handler* then converts the form again into a message, which is translated by the *Message Translator* and sent back to the original agent which made the request.

3.4 Trust and Security Services

There are many trust and security issues that come to mind in regards agent-agent and human-agent interactions. This issues deal with network security, message content, authentication and authorization. These issues are very common in any distributed system, however what makes them unique in this framework of agent and human interactions is that there is no predetermined communication between a server and a client because this framework proposes a highly distributed architecture with no pre-arranged relationships between service providers and requesters. What is needed, therefore, is a trust and security infrastructure that can be customized to this type of distributed architecture.

There are at least three systems that come to mind that are originally intended for supporting distributed agent interactions in the Semantic Web. The first and most favored by the authors is the one proposed by Gavriloaie et al [31], because it does not require registration of the agents or the users in order to interact and build trust. In addition it is based on declarative policies which can be easily maintained as an additional service in the UDDI registry.

The second approach is proposed by Kagal et al [32] and deals with annotating distributed Semantic Web sources with policies. Although this could work with Semantic Web pages and even agents, it would be difficult to maintain for human users. Yet, the third approach is based on Semantic Web languages for policy representation and reasoning as described by Tonti et al [33]. Although these policy based approaches could in theory work with the UDDI registry, they are not flexible enough to allow users and agents to develop the policies on the fly as is the case of the approach proposed by Gavriloaie et al [31].

In addition to these trust issues, there are also issues related to reputation that that need to be incorporated in the framework. This kind of trust deals with beliefs about competencies on the part of both humans and agents. Security on the other hand deals with authorization and verification issues related with the access and transmission of messages across the network, which of course is needed to ensure the validity of such messages. We will leave these issues for others to solve, since they are not the main focus of this paper.

[8] At the BYU Data Extraction Group [28] with which the authors are affiliated, we have experimented with ontology-driven forms with much success [17, 29, 30]. Here we plan to use the results of that research to enable the interaction of humans with the agents through dynamically generated forms.

4 A Simple Example

In this section we illustrate the four services described in sections 3.1, 3.2 and 3.3, but not 3.4 which is not in the main scope of this paper. Let's suppose that the user wishes to purchase a laptop computer for her son.

First, the user needs to identify a service she requires. This can be accomplished in two ways. In the first, the user browses through a directory of services and makes a specific request based on that particular service. In the second, the user issues a text-based request to a specific category of services. We will illustrate this more challenging second option. In this case, he would browse through a category of services and find the one for *Computer Retail.* This category of services are analogous to directories commonly found in Web search engines and portals, for instance Yahoo and Google. Once the specific category of services is found the user can type a request such as "buy the fastest laptop with Centrino technology and 128-bit graphics which is under $2,000.00". Since the user selected a particular category, there are a number of choices of services providers under that category. Furthermore, there is also a domain ontology associated with the category such as one defined in DAML or OWL. In addition, there is also recognizers associated with that particular domain ontology and its concepts. We use these recognizers to parse the user request into concepts in the ontology. This produces concept-instance pairs such as:

```
{ProductPurchase="buy"}{ConstraintFastest="fastest"}
{LaptopComputer="laptop"}{CentrinoCPUType="centrino"}
{128BitGraphicCardType="128-bit graphics"}
{ConstraintUnder="under"}{Price="$2,000.00"}
```

Notice that this is by no means natural language processing. Instead, this is simple concept-instance recognition based on the recognizers associated to the concepts in the domain ontology. First, the user is registered temporarily in the UDDI directory so that additional information can be requested later by the service provider if one is found. Second, the request is categorized as *ProductPurchase* service and the rest of the matched concepts recognized as parameters for the service. This information is then used to search for a service registered in the *Computer Retail* category for the closest matching service. The service does not need to be a 100% match as any number of registered services might provide all or partially matching services.

In this case let's assume that the service is provided by a product purchasing agent that specializes in finding the cheapest computer selling web services based on given specifications. The agent might have the following Java interface in order to search and purchase a computer.

```
void PurchaseComputer{String ComputerModel, Float LowPrice
                      Float HighPrice, String CPUType,
                      Bool EmbeddedEthernet, Bool Modem,
                      String GraphicCardModel, Bool Floopy,
                      Integer CDROMSpeed, ...}
```

The user request is translated to the local ontology of he agent and a request is composed. However, since not all the information the agent requires to provide the service is available in the user request, the agent sends a request to the user for the additional information. This request is translated into a form as illustrated in Figure 2 and presented to the user. The user then fills in the additional information and sends it back to the agent to complete the transaction if all the infomation needed is available. If not, then similar interaction between the agent and the user might occur until the transaction is completed.

Notice that during this process the agent might also interact with other agents, web services or even humans to find the right computer that meets the user's specificiations.

5 Concluding Remarks

We have described a framework for human-agent interaction in the Semantic Web. The framework supports the transfer of both explicit knowledge using a readily available domain ontologies to translate meaning between agents and agents and humans and agents. I also supports the transfer and use of tacit knowledge on the part of the users, who can direct the work of agents and acquire new tacit knowledge in the process. Agents too can acquire tacit knowledge, because by mapping their local ontology to a domain ontology, they can communicate with other agents and users without having to explicitly demonstrate their needs, but instead depending on ontology mapping to translate the meaning.

The framework is based on a combination of services supported by intermediate domain ontologies and flexible mapping mechanisms that map agent and human internal representations and external communication protocols. The framework is based on a solid conceptual foundation and can be extended to also incorporate other service-based computational mechanisms such as web services. The services described here are just part of an evolving framework and not intended to be exahustive, but instead extensible. As mentioned in the text, these services can be extended to also incorporate web services or any other Semantic Web resource that can be mapped to the domain ontologies. This includes documents and web pages. The central focus of the framework is to encourage contribution by anyone or anything, that might possess useful information, knowledge or expertise for the successful completion of collaborative problem solving tasks.

The most interesting and unique contribution of the paper is that no-matter what or who the problem solving participants are, they are not required to go through a rigorous ontological commitment or have a common communication protocol in order to collaborate. We perceive this to be the main problem that overshadows the successful proliferation of the Semantic Web and have tackled it head on. Although, there is still much work to be done to expand the vision of this framework beyond its current form, we are confident that it will make it possible for “ordinary”, non-technical people to reap the full benefits of the Semantic Web to come.

References

1. Berners-Lee, T.: Information management: A proposal (the original www proposal) (1989) http://www.w3.org/History/1989/proposal.html.
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American **36** (2001)
3. Shadbolt, N., Schraefel, M., Gibbins, N., Glaser, H., Harris, S.: Cs aktivespace: Representing computer science in the semantic web. In: Proceedings of the 13th International World Wide Web Conference (WWW2004), New York, NY, ACM (2004)
4. Bradshaw, J., ed.: Software Agents. 2nd edn. MIT Press, Cambridge, Massachusetts (1997)
5. McDowell, L., Etzioni, O., Gribble, S., Halevy, A., Levy, H., Pentney, W., Verma, D., Vlasseva, S.: Mangrove: Enticing ordinary people onto the semantic web via instant gratification. In: Proceedings of the 2nd International Semantic Web Conference (ISWC03), Sanibel Island, Florida, USA (2003)
6. Romer, P.M.: Increasing returns and long run growth. Journal of Political Economy (1986) 1002–37
7. Romer, P.M.: Endogenous technological change. Journal of Political Economy (1990) S71–S102
8. Tijerino, Y.A.: A Task Analysis Interview System Based on Two-Level Task Ontologies. PhD thesis, Osaka University, Osaka Japan (1993)
9. Polanyi, M.: The Tacit Dimension. Routledge & Kegan Paul Ltd, London (1966)
10. Nonaka, I.: The knowledge-creating company. Harward Business Review **69** (1991) 96–104
11. Nonaka, I., Takeuchi, H.: The Knowledge-Creating Company. Oxford University Press, Oxford (1995)
12. Uschold, M.: Creating semantically communication on the world wide web (2002) Keynote address at the *Proceedings of the Semantic Web Workshop at the 11th International WWW Conference.*
13. : Semantic web services language (2004) http://www.daml.org/services/swsl/.
14. Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: Semantic matching of web services capabilities. In: Proceedings of the First International Semantic Web Conference (ISWC02). (2002) 333–347
15. Al-Muhammed, M.: Dynamic matchmaking between messages and services in multi-agent systems. Technical report, Brigham Young University, (Provo, Utah)
16. Embley, D., Tao, C., Liddle, S.: Automating the extraction of data from tables with unknown structure. Data & Knowledge Engineering (2004) (to appear).
17. Chen, X.: Query rewriting for extracting data behind html forms. Technical report, Brigham Young University, Provo, Utah (2004) Currently at www.deg.byu.edu/proposals/index.html).
18. Walker, T., Embley, D.: Automating the extraction of genealogical information from the web. In: Proceedings of Fourth Anual Workshop on Technology for Family History and Genealogical Research, Provo, UT (2004) (Currently Available at www.deg.byu.edu) MyComments =.

19. Tao, C.: Schema matching and data extraction over html tables by cui tao. Master's thesis, Brigham Young University (2003)
20. Tijerino, Y., Embley, D., Lonsdale, D., Nagy, G.: Ontology generation from tables. In: Proceedings of the 4th International Conference on Web Information Systems Engineering, Rome, Italy (2003) 242–249.
21. Al-Muhammed, M., Embley, D.: Towards enabling communication among independent agents in the semantic web. In: Proceedings of 3rd International Semantic Web Conference (ISWC04), Hiroshima, Japan (2004) (Submitted).
22. Embley, D.: Programming with data frames for everyday data items. In: Proceedings of the 1980 National Computer Conference, Anaheim, California (1980) 301–305
23. Embley, D., Tao, C., Liddle, S.: Automatically extracting ontologically specified data from HTML tables with unknown structure. In: Proceedings of the 21st International Conference on Conceptual Modeling (ER'02), Tampere, Finland (2002) 322–327
24. Chartrand, T.: Ontology-based extraction of RDF data from the world wide web. Master's thesis, Brigham Young University, Provo, Utah (2003)
25. Mizoguchi, R., Tijerino, Y.A., Ikeda, M.: Task analysis interview based on task ontology. Expert Systems with Applications **9** (1995) 15–25
26. Mizoguchi, R., Ikeda, M.: Towards ontology engineering. In: proceedings of the Joint 1997 Pacific Asian Conference on Expert Systems / Singapore International Conference on Intelligent Systems, Singapore (1997) 259–266
27. Tijerino, Y., Embley, D., Lonsdale, D., Nagy, G.: Ontology generation from tables. (Journal of World Wide Web Internet and Web Information Systems) Submitted.
28. : Homepage for BYU data extraction research group (2004) URL: http://osm7.cs.byu.edu/deg/index.html.
29. Liddle, S., D.W. Embley, D.S., Yau, S.: Extracting data behind web forms. In: Proceedings of the Joint Workshop on Conceptual Modeling Approaches for E-business: A Web Service Perspective (eCOMO 2002), Tampere, Finland (2002) 38–49
30. Yau, S.: Automating the extraction of data behind web forms. Technical report, Brigham Young University, Provo, Utah (2001) http://www.deg.byu.edu.
31. Gavriloaie, R., Nejdl, W., Olmedilla, D., Seamons, K.E., Winslett, M.: No registration needed: How to use declarative policies and negotiation to access sensitive resources on the semantic web. In: 1st European Semantic Web Symposium, Heraklion, Greece (2004)
32. Kagal, L., Finin, T., Joshi, A.: A policy based approach to security for the semantic web. In: Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida, USA (2003)
33. Tonti, G., Bradshaw, J., Jeffers, R., Montanari, R., Suri, N., Uszok, A.: Semantic web languages for policy representation and reasoning: A comparison of KAoS, Rei, and Ponder. In: Proceedings of the 2nd International Semantic Web Conference, Sanibel Island, Florida, USA (2003) 419–437