

# Agent-Mediated Knowledge Sharing for Services Management

Michael F. Nowlan  
Dept. of Computer Science  
Georgetown University  
Washington, DC 20057  
[mfn3@georgetown.edu](mailto:mfn3@georgetown.edu)

M. Brian Blake  
Dept. of Computer Science  
Georgetown University  
Washington, DC 20057  
[blakeb@cs.georgetown.edu](mailto:blakeb@cs.georgetown.edu)

## Abstract

*For service-oriented architectures that span multiple businesses, organizations must transfer information back-and-forth about their available services. Because of the potential large volume, it is unreasonable and impractical to expect human practitioners to handle the number of interactions desired and/or required on a continual basis. Intelligent agents offer the adaptability and flexibility to handle the knowledge transfer that must occur in order to share web service offerings. Agents that operate in this domain will require specialized communication protocols that effectively transfer service-oriented information. This paper introduces an architecture and specialized communication procedures designed for this sort of knowledge sharing environment. We show that these procedures perform reasonably when evaluated using current agent communication technologies.*

## Keywords

Intelligent Agent, Web Services, Communication, Architecture.

## 1. Introduction

Service-oriented computing is facilitating an environment where businesses expose capabilities that can be of widespread use and benefit. There are barriers to the discovery and integration of such beneficial services when prospective consumers are unaware of new service availability or the ease with which they can be deployed within their local organizations. In addition, human users might first consult with other users before adopting new services. As such, the process of discovery is limited by the timeliness of human consultants.

Intelligent agents are adaptive software components with the ability to act as proxies for human counterparts and proactively act in a user's or business's best interest. IBM's RAISE project further defines intelligent agents as human proxies containing a brain, body, society and human-agent interaction [11]. Our work suggests the use of intelligent agents to assist organizations in managing the discovery and introduction of new web services. We developed a federation of intelligent agents that work together through communication to incorporate the web services of-best-fit into each of their business processes.

In this paper, we focus mostly on the agent "society", or communication between agents, and use this as the catalyst for all agent actions and functions. By allowing agents access to each other's knowledge base, incorrect decisions are limited, and correct decisions are more quickly executed. This sharing is applicable for both cross-domain businesses, such as travel agencies, and businesses operating in the same domain.

Although the need for widely applied collaboration between agents of a business domain is apparent, the obstacles to fostering such an atmosphere remain numerous. The first obvious problem is that similar businesses do not operate within the same firewalls or business architectures. Also, with no set language or communication protocol established, businesses would need to proactively set down rules for inter-communication. Although related work shows that agents have the capabilities of proactively discovering web services [5], similar but unconnected businesses will not often take the effort to establish lines of communication.

In this paper, we address several research questions with regards to agents that collaborate to recommend services. Relevant questions are:

1. *What are the required pieces of information that agents must share with each other to enable collaborative recommendations?*
2. *What reasons, besides recommendation, are there for agents to initiate communication between each other?*

This paper expands on the earlier related work [15] by analyzing and comparing the agent communication procedures against data from current intelligent agent frameworks and inter-process communication frameworks.

This paper proceeds in the next section with an overview of the work in our concentrations of intelligent agents and web service recommendation. In Section 3, we outline the abilities and functions of our agents and the federation they inhabit in order to motivate why agent-to-agent communication is fundamental to accuracy. Section 4 offers a close examination of the various types of communications between agents and introduces a new messaging model for agents that collaborate on web service recommendation. Finally, we evaluate our

approach based on its comparison to current agent messaging benchmarks.

## 2. Related Work

There are two intersecting threads within our work: web service recommendation and intelligent agent communication. Recommender systems have traditionally been associated with the electronic commerce domain [6][7]. In most cases, a recommender system uses the input of a consumer or user and finds relevant web services. Woogole [8] is a project that allows users to create a query to find relevant web services. This approach is similar to the search that takes place in our approach; however, in our work, we proactively create customized queries from the WSDL files and SOAP messages that traverse through the service-oriented architecture. Several recent papers addressing recommender systems for web services either incorporate web services in their implementation [4] or suggest web services at the human-user level [14]. Our approach tries to suggest services that may facilitate inter-organizational workflow. In addition, our work contributes to the general research area [17] where intelligent agents are used to generate the recommendations.

Collaborative filtering [3] is one way to augment web service discovery. We leverage this notion in our work by making agents the collaborators instead of humans. Upon finding services through the recommendation model, our agents take an extra step by implementing the web service into the business process. In this sense, our work is closely aligned with the Business Process Execution Language (BPEL) [13].

## 3. Agents Within the Framework

Our agent federation is composed of multiple agents representing business processes in a networked setting. Moreover, we believe agents in this federation should have the same implementation as the web services that they manage. The agents themselves are web services [12], capable of receiving requests for information and providing results for those requests. The ultimate goal of agents is to be able to recommend the most relevant web services to the individuals they represent. Although this singular notion is the starting point, agents may have multiple functions they can perform.

### 3.1 Internal Agent Functions

The main internal function of agents is the ability to query Universal, Description, Discovery and Integration (UDDI) [19] registries to discover useful and relevant services. To determine relevancy, an agent uses syntactic methods of comparing the similarity between the web

services definitions (Web Service Description Language Documents) [20] and pertinent strings extracted from a business model. The syntactic matching method is a one to one string comparison called TSM-LP [16]. Extracting descriptive strings and determining their similarity to the local web services is the second capability of agents and it enables agents to compose a bag of words that describe the operational domain of their business. Thirdly, an agent categorizes the web services existing locally in order to stratify the services for efficient and fast searching. Finally, depending on the uniqueness of each category, an agent adjusts the strictness of its similarity method (TSM-LP) in order to make more accurate recommendations in various domains. Figure 1 illustrates the internal functions of an agent. (Although making a recommendation is an internal function, it is performed after collaboration with other agents and thus, it is described in the next section.)

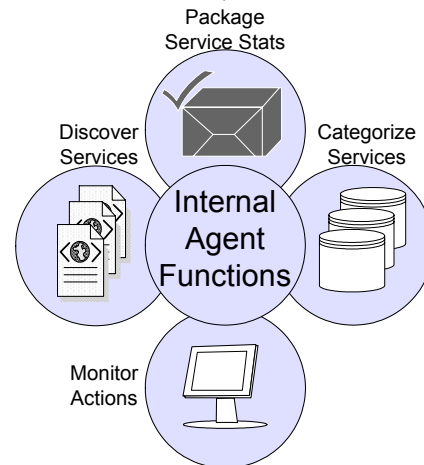


Figure 1. Internal Agent capabilities

### 3.2 Agent Communications and Collaboration

Making accurate web service recommendations is the primary function of agents, and to do that, communication and collaboration with other agents is necessary. Our federation can be thought of as a collection of “friendly” agents that work together on operations. The federation allows agents to ask other agents for help with a particular recommendation. In this way, agents help each other with both the discovery and the recommendation of web services. Consulting agents agreeing on a particular service provides additional strength to that recommendation. After capturing information about the current business process, an agent will tag several local services that are most relevant to its user. However, before recommending or integrating these services, an agent will query the other agents with the extracted information describing its process in an attempt to discover new services it did not know existed, and/or to seek confirmation or disapproval. Figure 2 depicts one such query an agent makes to its peers. After running this

material against their own repositories, they report back to the first agent with recommendations for inclusion or exclusion. If there is overlap in the suggestions of the external agents and its own recommendation, the initiating agent recommends this service or services for inclusion into its business model.

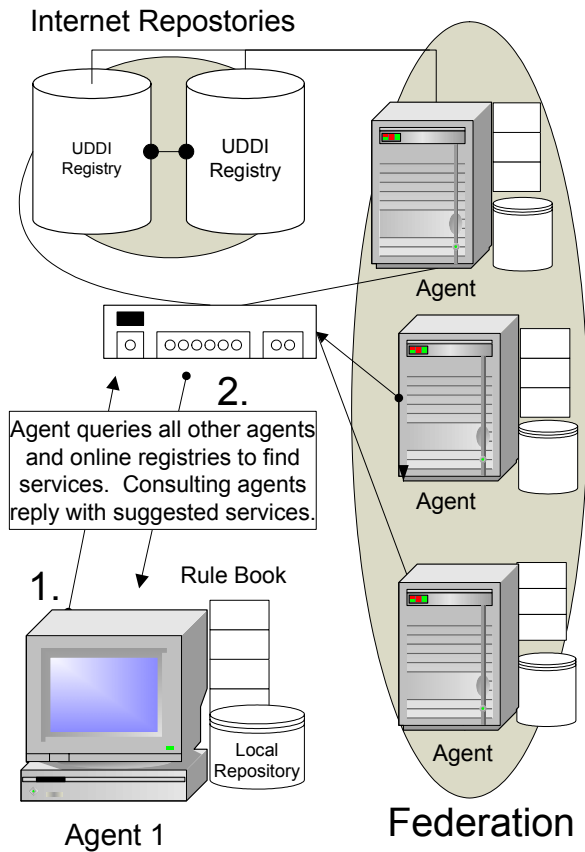


Figure 2. An Agent Querying its Peers

#### 4. Service-Oriented Agent Communication

Agents in the federation are implemented as web services, and their interaction protocols are implemented via the exchange of SOAP messages. An innovation in this paper is the development of an information model associated with the necessary agent-communicated data within the federation. By analyzing the interaction protocols, we introduce this new communication data model. The data attributes in Table 1 represent the result of our analysis and the aggregation of information that is passed repetitively throughout the interaction protocols defined in later detail in this paper. We identify these attributes as eXtensible Markup Language elements relevant to the communication data model. By examining our communication protocols we were able to determine which pieces of information were in constant transition, and using this knowledge and the new elements, we created the *Service Recommendation Markup Language*

(SRML). The SRML schema wraps these data attributes into an organized collection, enabling the agents to interpret SOAP messages with increased brevity. The schema works to aggregate response time by allowing faster indexing of the information being sent once it reaches its local destination. Most importantly, the schema allows message exchanging in a defined and repeatable way. The SRML schema fosters communication between businesses in a simple, and easily understood format, making cross-domain collaboration possible. Table 2 outlines the SRML schema used by agents to interpret SOAP messages.

Table 1. SRML Message Information Attributes.

Keyword (Data Store)	Xml Tag	Description
Bag of Words	<i>WordBag</i>	String collection describing a SOC action or user file.
Category Name	<i>CatName</i>	The name of a category of services
Category Self-Similarity	<i>SimPcnt</i>	Percentage of self-similarity of a category
Task ID	<i>TkID</i>	Identifier for an action.
Agent ID	<i>AgID</i>	Identifier for an Agent.
Group of Services	<i>ServList</i>	A list of candidate or reference web services
Response Constraint	<i>RespC</i>	Time constraint for a query response
Entrance Threshold	<i>EnterT</i>	Threshold of similarity needed to recommend to another agent.
Alignment Threshold	<i>AlignT</i>	Category similarity threshold needed to create an agent agreement.
End Message	<i>Close</i>	Boolean to end exchange.
Category Equivalency List	<i>CatHash, initCat, extCat</i>	List equating Categories of two Agents (Initiating Agent and External Agent)

##### 4.1 Agent-to-Agent Communication Protocols

Communication makes the tasks of web service discovery, management and recommendation a collective process, rather than an independent action. Furthermore, using agents to do these actions is highly beneficial because it automates functions and increases scalability. The primary reason, or task, for communication is *Service Recommendation*. Agents group the services in their local repository into *categories* of similar services. It is likely that over time, one agent's local categories will resemble another agent's, and an alignment of these categories makes recommendation easier so that agents can search within roughly similar categories across their separate

domains. This second task is called *Category Alignment*. A third reason for communication is linked to the second task and is collaboration for category self-similarity, or *Category Analysis*. After categories have been aligned, as in task 2, it is also beneficial to determine how unique the services are in two equivalent categories so that the agent can accurately judge the strength of relevancy matches. The last reason for communication is a *Call for Services*. One agent sends this call out to multiple agents, or one agent in particular, asking for web services to populate its own local repository. Also, newly created agents use this communication to initially create their local repositories.

**Table 2.** New Agent Exchange Model: SRML Schema.

```

<xs:schema xmlns:xsd=
"http://www.w3.org/2001/XMLSchema"
  xmlns:altres=
"http://servicecentricity.georgetown.edu/SRML"
  targetNamespace=
"http://servicecentricity.georgetown.edu/SRML"
</xs:schema>
  <annotation>
    <documentation xml:lang="en">
      The Service Recommendation Markup
      Language for Agent-Based SOAP messages
    </documentation>
  </annotation>
  <xs:element name="SRML">
    <xs:complexType>
      <xs:attribute name="TkID" type="xs:date"/>
      <xs:attribute name="AgID" type="xs:string"/>
      <xs:attribute name="Close" type="xs:string"/>

      <xs:sequence>
        <xs:element name="WordBag"/>
        <xs:element name="CatName"/>
        <xs:element name="SimPcnt"/>
        <xs:element name="ServList"/>
        <xs:element name="RespC"/>
        <xs:element name="EnterT"/>
        <xs:element name="AlignT"/>
        <xs:element name="CatHash">
          <xs:complexType>
            <xs:element name="initCat"/>
            <xs:element name="extCat"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

The types of messages we use to complete communication protocols extend the FIPA ACL

Communication library [10]. The protocols follow an ordered exchange of the message types to carry out a communicative act. Different message types contain different pieces of information. **Call for Proposal**, **Accept Proposal**, **Agree** and **Confirm** are called the “handshake” message types because they are used at the beginning of communicative acts to establish a working relationship. These messages contain little information other than task identification, agent identification and the type of act. **Inform**, **Propose** and **Request** messages often contain the information an initiating agent wants to send to a consulting agent. **Inform Reference** and **Subscribe** messages contain the response information a consulting agent shares with the initiating agent. A **Failure** message ends communication between agents. The collection and exchange of different sets of these message types make up the four communication protocols.

## 4.2 Collaborative Recommendation Protocol

The cornerstone to our approach is timely and effective recommendation of web services. In order to do this, an agent gathers information from their business model’s daily routine and collects these descriptive strings into a “bag of words”. The agent finds web services in its local repository that are similar to the extracted words using a matching approach called TSM-LP [16]. The beginning of communications usually start with a “handshake” between agents in the form of a **Call for Proposal** and **Accept Proposal**. The agent then sends the extracted strings to all accepting agents in the federation (**Request**), who, in turn, do the exact same procedure on their local repositories. After that, the consulting agents respond to the original agent with the services that they found to be most similar to the bag of words (**Inform**). With its own original recommendation and the suggested services of all the other agents, the original agent finds the most commonly appearing services and recommends them for inclusion into its business process. The original agent also sends a message to the collaborating agents to share the result of the recommending process (**Inform**). This message informing the consulting agents of the outcome enables agents to learn over time. The initiating agent stores this service as a “good” recommendation for the extracted strings in its knowledge base. In this way, agents develop knowledge bases consisting of descriptive strings and a web service that is applicable in the presence of such strings. Both initiating and consulting agents learn from the collaboration. This protocol is loosely depicted in the previous Figure 2, and it is also pictured below in Figure 3.

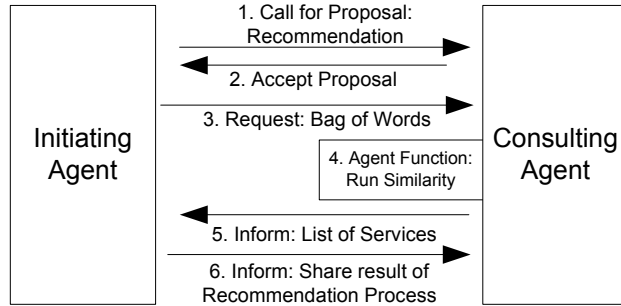


Figure 3. Collaborative Recommendation Protocol

### 4.3 Category Alignment Protocol

Based on the varying natures of business domains, different agents will have different local repositories. One way to organize local repositories is through stratification into categories of similar services. Although the number and makeup of these categories can vary a great deal, it is likely that two unique agents will have at least a few categories that contain very similar services. Because of this likelihood, we created the Category Alignment Protocol.

This message exchange starts with a “handshake” in the form of the initiating agent asking to align categories, and the consulting agent agreeing to carry out the alignment (note: this handshake is slightly different with regards to the communicative acts that are used, **Agree** and **Confirm**). Upon confirmation, the initiating agent sends a Category Name and a collection of services to the consulting agent (**Request**). The consulting agent internally determines which one of its local categories shares the highest similarity to the service list through TSM-LP [16]. The consulting agent then sends a Category Name back to the initiating agent to share the result of the alignment procedure (**Inform Reference**). Finally, the initiating can **Subscribe** to the consulting agent to receive an updated category list if the agent’s categories should change in the future. This update is sent as an **Inform Reference** message. Figure 4 depicts this message exchange.

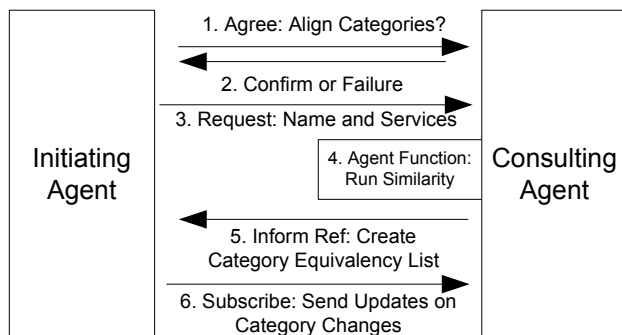


Figure 4. Category Alignment Protocol

### 4.4 Category Analysis Protocol

Once agents have aligned their local categories with each other, it is sometimes necessary to exchange further statistical information about these categories to provide a more comprehensive understanding of their similarities (and differences). It is not enough to simply know that two categories are alike, or similar. Agents need to be able to compare the sensitivities or uniqueness of their categories in a numerical or quantitative way. Uniqueness is defined as the variability of web service message “part” names within a category. A category with mostly the same part names is considered to have a low uniqueness and a high self-similarity. Category sensitivity is important because it indicates how relevant a recommended service is. If a service is recommended from a category with high uniqueness (i.e. great variability), then it is possible that the service is being recommended simply because it is the only one sharing any similarity to the extracted strings. This protocol deals with sharing these category sensitivities between agents after they have aligned their repositories.

Agents engaging in this protocol have already collaborated for Category Alignment, thus, only an informal “handshake” is required in the form of the initiating agent **Proposing** the protocol by sending a Category Name and its self-similarity percentage. The consulting agent receives this message and using the Category Equivalency List (produced by Category Alignment) shares the self-similarity of its corresponding local category (**Inform Reference**). The initiating agent appends this percentage to the Category Equivalency List. Agents can then choose whether or not to update the percentages for all the categories in their Equivalency List. Figure 5 depicts this message exchange.

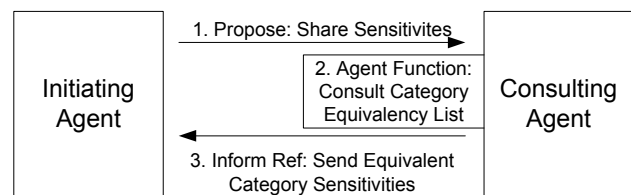


Figure 5. Category Analysis Protocol

### 4.5 Service Sharing Protocol

As business needs evolve over time, agents will increase and reduce the services in their local categories dynamically. Agents may find that certain categories are obsolete to their business model. Conversely, agents may also find certain categories to be overwhelmingly applicable and desire to proactively find more services of the same nature. In these cases, an agent can ask other agents for a small collection of services of a given type.

Whereas, in the recommendation protocol, consulting agents send services based on similarity to a business action, this protocol deals simply with the sharing of services based on a stated category (i.e. type of service). This protocol is also invoked when a new agent registers to the federation, and needs to populate its local repository from scratch.

The “handshake” for this protocol is nearly identical to that of the recommendation protocol, with the only difference in the semantics of the proposal (**Call for Proposal** and **Accept Proposal**). After agreement to communicate, the initiating agent sends a Category Name or descriptive word for the type of services it desires (**Request**). The consulting agent responds with the Category Name from its local repository and a Service List of the most frequently used services in that category (**Inform Reference**). The initiating agent takes this Service List and stratifies the services, if needed, into local categories based on the nature of the services. Figure 6 depicts this message exchange.

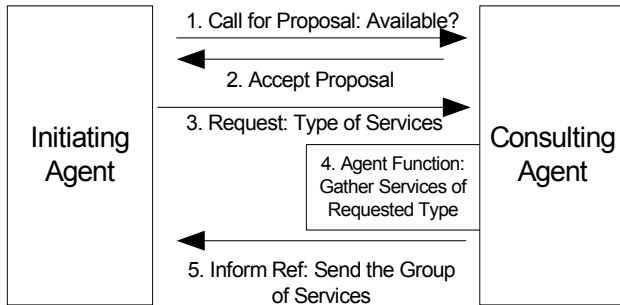


Figure 6. Service Sharing Protocol

## 5. Performance Analysis and Evaluation

In order for the agent federation to be feasible in practice, the recommendations must be presented to the user in a reasonable timeframe. Considering the average time it takes a user to scan a new web site or read a medium-sized message, the agent-federated recommendation time is based on a 5-6 second response time ceiling. In evaluating the agent federation, it is important to measure the system response time against this benchmark.

At the time of this paper, only the recommendation scoring functionality has been developed which is reported in more detail in other work [16]. Of importance to this analysis is the fact that the time for creating a recommendation score (i.e. internal agent function) is approximately 2 seconds. Although this time is well below the reasonable response time of 5-6 seconds to the user, when the necessary messaging overhead is taken into account, the time approaches 5-6 seconds.

In this analysis and of importance to the focus of this paper, we are interested in the combined performance of agent communication *and* the recommendation score

generation. In order to determine an estimated communication response time, we investigated related work containing performance measurements of tools associated with agent communication. Gigaspaces [18] is a collaboration framework typically used for inter-process (i.e. inter-agent) communication. Although industrial performance reports [18] show that Gigaspaces performs under 4 ms (irrespective of number of entries) when reading the shared space, it does not address the inter-process messaging. JADE is perhaps the leading agent framework.

Both Cortese et. al.[8] and Ahuja et.al. [1] agree that the inter-process messaging overhead is about 8-10ms per concurrent process (agents) considering a reasonable network infrastructure and hardware. The values given in Cortese et. al. is based on JADE, Ahuja et. al. based their measures on GigaSpaces and CORBA. In our analysis, we build on the experimental benchmarks from this related work to establish an estimated agent communication time. We then add the experimental recommendation generation time (calculated in this work) to develop and estimate for the agent communication protocols. Based on related work, exchanging 7 character messages for 10,000 trips on a reasonable hardware platform results in the following performance times:

- When the number of agents is between 0 and 100:  
**Message Time** = 10ms \* (Number of Agents)
- When number of agents is between 100 and 2000:  
**Message Time** = 8ms \* (Number of Agents)

In order to analyze, the protocols introduced in this paper, we used the sum of messages required for the protocol. We also estimated message size based on a typical SRML message populated with typical recommendation request information. For example, the recommendation protocol, the category analysis/alignment protocol, and the service sharing protocol required 5, 7, and 4 messages respectively. The related work suggests the communication of 7 characters with 10,000 trips while the agent communication is on the order of 7000 character messages for 10 trips. Consequently, after extrapolating data size and number of round-trips per recommendation, the following communication-only performance measures were calculated for the recommendation protocol:

- When number of agents is between 0 and 100;  
**Message Time** = 5ms \* (Number of Agents)
- When number of agents is between 100 and 2000;  
**Message Time** = 4ms \* (Number of Agents)

In further analysis, we assumed that the other protocols (i.e. the service sharing protocol, category analysis protocol, and category alignment protocols) that happen concurrently with the recommendation protocols would add additional overhead. Through related calculations we

found that if all of the peripheral protocols are executed for each recommendation, then the response time is tripled. Figure 7 shows the estimated communication/messaging time as it relates to concurrent agents. Other measures show the performance impact associated with executing the other services (i.e. category analysis, alignment, and service sharing) 50% of the time and 25% of the time.

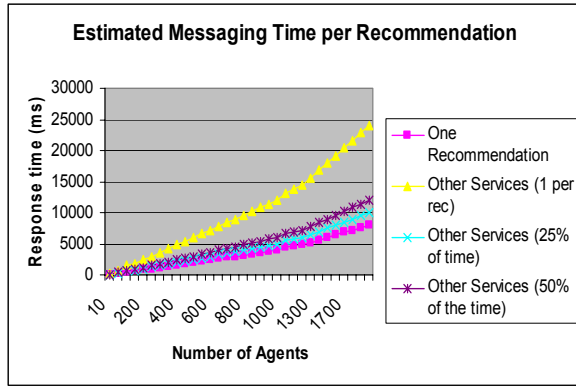


Figure 7. Communication Time per Recommendation.

Figure 8 expresses the *total* estimated recommendation time, which includes the calculation the recommendation score. A tradeoff for system performance and maintaining current information is updating other services only 25% of the time. Using this level of service, the recommendation response time is about 6 seconds for 250 collaborating agents. This result would suggest that our framework should optimize performance by limiting agent collaboration to the top 250 most relevant agents within the federation. The authors feel that this is a reasonable quality of service that assures both the fidelity of the recommendations and the speed.

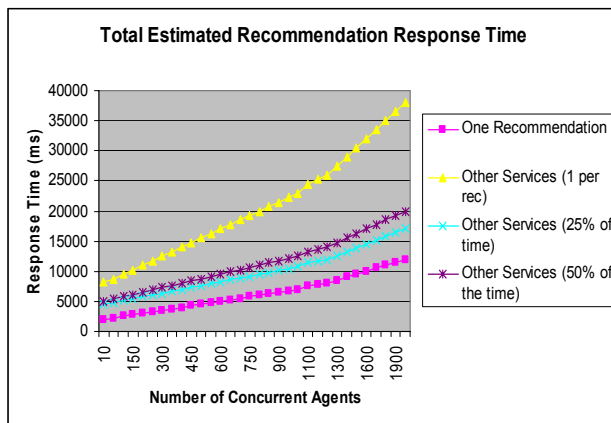


Figure 8. Estimated Recommendation Time

## 6. Discussion

The agent federation and communication protocols are effective because they enable communication between agents using service-oriented approaches. By introducing SRML, there is a generic data structure for agents to share knowledge in the form of recommendation irrespective of the domain (e.g. Entertainment, Sports, Travel, and Finance).

As a limitation to this approach, it may be necessary to regulate how many and which agents can be queried for a single business action. The performance analysis shows that in order to get a 6 second response time, it will require that the 250 most relevant agents be consulted for on-demand recommendation generation. Furthermore, limiting the number of consulting agents for any one query might aid an initiating agent in receiving more specific recommendations, but it also limits the scope of the knowledge base.

The agent federation suggests that an initiating agent must provide a time constraint for response when sending a query. This time constraint has to set under a second to assure that the overall recommendation performance meets feasible levels. In future, work we must investigate the practicality of such a stringent constraint in terms of the real-time operations of this environment.

## 7. Conclusion

In this work, we propose agent-to-agent communication and consultation to provide relevant recommendations of web services. Collaboration is a necessary and primary member of an agent's rule-based decision making. By forcing constant communication between agents, agents become more knowledgeable about the services they recommend and their relation to other business processes in the federation. Our federation allows agents to learn "on the fly" and incorporate new web services into their business domain continually. We have laid out a "friendly" architecture to enact this communication protocol, allowing for accurate and comprehensive web service recommendation in real time. The agent-to-agent schema we developed enables fast and uniformed message protocols within the architecture. By limiting recommendation response time, we make certain that only agent's that can be of service are included in collaborative measures. Finally, by conforming to the 5-6 second window of collaboration the message protocols and the SRML schema support the effectiveness of this approach.

## 8. References

- [1] Ahuja, S.P., Eggen, R, and Anjani, K.J. A Performance Evaluation of Distributed Algorithms on Shared Memory

- and Message Passing Middleware Platforms, *Informatica*, Volume 29, Number 3, pp 327-333
- [2] Alechina, Natasha, Logan, Brian, Whitsey, Mark (2004). "Modeling Communicating Agents in Timed Reasoning Logics." *Proceedings of the Ninth European Conference on Logics in Artificial Intelligence (JELIA 2004)* (pp. 95-107).
- [3] Badrul Sarwar, B., Karypis, G., Konstan, J. And Riedl, J., "Item-based Collaborative Filtering Recommendation Algorithms", *In the Proceedings of the 10 International World Wide Web Conference*, Hong Kong, 2001.
- [4] Birukou, A., Blanzieri, E., D'Andrea, V., Giorgini, P., Kokash, N., and Modena, A. IC-Service: A Service-Oriented Approach to the Development of Recommendation. *In the Proceedings of the 22nd Annual ACM Symposium on Applied Computing*, ACM Press, Seoul, Korea, March 11 - 15, 2007.
- [5] Blake, M.B., Fado, D.H, and Mack, G.A. "Proactive Service Discovery and Execution using Intelligent Agents", 4th IEEE International Conference on Web Services (ICWS 2006), Chicago, IL, 2006
- [6] Burke, R.D., "Hybrid Recommender Systems: Survey and Experiments", *User Model. User-Adapt. Interact.* 12(4): 331-370 (2002)
- [7] Burke, R.D., "The Wasabi Personal Shopper: A Case-Based Recommender System". *Proceedings of AAAI/IAAI 1999*: 844-849
- [8] Cortese, E., Quarta, F., and Vitaglione, G. Scalability and Performance of JADE Message Transport System, *Proceedings of the 2002 AAMAS Workshop on AgentCities*, Bologna, 16th July, 2002
- [9] Dong, X., Halevy, A.Y., Madhavan, J., and Nemes, E., Zhang, J. Similarity Search for Web Services. VLDB 2004
- [10] FIPA (2006), Agent Communicative Act Specification Library, Foundation for Intelligent Physical Agents, <http://www.fipa.org/specs/fipa00037/index.html>
- [11] Grosz, B. "Building Commercial Agents: An IBM Research Perspective," in Second International Conference on The Practical Applications of Intelligent Agents and Multi-Agent Technology, The Practical Applications Company, London, UK. April 1997.
- [12] Huhns, M., "Agents as Web Services", IEEE Internet Computing, Volume 6, No. 4, 2002
- [13] Leymann, Frank, Roller Dieter. "Business Processes in a Web services world." IBM Software Group. 2002.
- [14] Manikroa, U.M. and Prabhakar, T.V., "Dynamic Selection of Web Services with Recommendation System", *Proceeding of the International Conference on Next Generation Web Services Practices*, August 2005, Seoul, Korea
- [15] Nowlan, M.F. and Blake, M.B., "Intelligent Agent Communication and Collaboration for Web Services Management", WETICE 2007
- [16] Nowlan, M.F., Kahan, D.R., and Blake, M.B. "Using Naming Tendencies to Syntactically Link Web Service Messages", in *Data Engineering Issues in E-Commerce and Services (DEECS)*, Lecture Notes in Computer Science, Vol. 4055, pp. 90-99, Springer-Verlag, June 2006
- [17] Proceedings of the Multi-Agent Information Retrieval and Recommender Systems, IJCAI-2005 Workshop, Edinburgh, Scotland, July 2005
- [18] The GigaSpaces Platform: Performance report (2007): <http://www.gigaspace.com/download/GigaSpacesPerformanceReport.pdf>
- [19] Universal, Description, Discovery and Integration: <http://www.uddi.org>
- [20] Web Services (2006): <http://www.w3.org/2002/ws/desc/>