

1 Inf Syst Front
2 DOI 10.1007/s10796-007-9043-6

4 Agent-mediated knowledge sharing for intelligent 5 services management

6 Michael F. Nowlan · M. Brian Blake

8 © Springer Science + Business Media, LLC 2007

11 **Abstract** For service-oriented architectures that span mul-
12 tiple businesses, organizations must transfer information
13 back-and-forth about their available services. Because of
14 the potential large volume, it is unreasonable and imprac-
15 tical to expect human practitioners to handle the scale of
16 interactions desired and/or required on a continual basis.
17 Intelligent agents offer the adaptability and flexibility to
18 handle the knowledge transfer that must occur in order to
19 share service offerings. Effectively transferring service-
20 oriented information in this domain requires autonomous
21 systems that adapt to heterogeneous environments. This
22 work introduces an architecture and specialized communi-
23 cation procedures designed for this sort of knowledge
24 sharing environment. We show that these procedures
25 perform reasonably when evaluated with current agent
26 communication technologies.

27 **Keywords** Recommendation · Web services ·
28 Agent communication · Collaboration · Management

29 1 Introduction

30 Service-oriented computing is facilitating an environment
31 where businesses expose capabilities that can be of
32 widespread use and benefit. There are barriers to the

discovery and integration of such beneficial services when
prospective consumers are unaware of new service avail-
ability 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 (Grosz 1997). 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 cross-domain
businesses, such as travel agencies.

Although the need for widely applied collaboration
between agents of a business domain is apparent, the
obstacles to fostering such an atmosphere remain numer-
ous. The first obvious problem is that multiple 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 (Blake et al. 2006), similar but

M. F. Nowlan · M. B. Blake (✉)
Department of Computer Science, Georgetown University,
333 St. Mary's Hall, 37th and O Street, NW,
Washington, DC 20057, USA
e-mail: mb7@georgetown.edu

M. F. Nowlan
e-mail: mfn3@georgetown.edu

68 unconnected businesses will not often take the effort to
69 establish lines of communication.

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

- 73 • What are the required pieces of information that agents
74 must share among each other to enable collaborative
75 recommendations?
- 76 • What other communication procedures are required that
77 may complement recommendation?

78 This paper proceeds in the next section with an overview
79 of the work in our concentrations of intelligent agents and
80 web service recommendation. In Section 3, we outline the
81 abilities and functions of our agents and the federation they
82 inhabit in order to motivate why agent-to-agent communi-
83 cation is fundamental to efficiency. Section 4 offers a close
84 examination of the various types of communications
85 between agents and introduces a new messaging model
86 for agents that collaborate on web service recommendation.
87 Section 5 discusses how agent access web services
88 registries. Finally, we evaluate our approach based on its
89 comparison to current agent messaging benchmarks.

90 2 Related work

91 There are two intersecting threads within our work: web
92 service recommendation and intelligent agent communi-
93 cation. Recommender systems have traditionally been associ-
94 ated with the electronic commerce domain (Burke 1999,
95 2002). In a typical recommendation session in a web
96 setting, a database is used to store transaction history as it
97 relates to a particular user. The profiles and transaction
98 history of multiple users are correlated such that users of
99 similar interests are matched. Consequently, the purchases
100 of one user can suggested to other users with similar overall
101 preferences.

102 In this work, a recommender system is used in much the
103 same way. The system uses input from many consumers
104 and/or providers to distribute relevant web services to other
105 relevant stakeholders within the entire federated communi-
106 ty. The information that agents store and disseminate are
107 represented as a group of keywords that can be compared
108 with keywords extracted from candidate services. Woogle is
109 a project that allows users to create a query to find relevant
110 web services (Dong et al. 2004). This approach is similar to
111 the search that takes place in our approach; however, in our
112 work, we create customized queries from the WSDL files
113 and SOAP messages that traverse through the service-
114 oriented architecture. Several recent papers addressing
115 recommender systems for web services either incorporate
116 web services in their implementation or suggest web

services at the human-user level (Birukou et al. 2007; 117
Manikrao and Prabhakar 2005). Our approach tries to 118
suggest services that may facilitate inter-organizational 119
workflow. 120

Collaborative filtering is where the preferences of 121
multiple users are exploited to identify the most specific 122
product/service for a specific user with a similar profile as a 123
related group of users (Sarwar et al. 2001). We leverage this 124
notion to enhance web service discovery in our work by 125
making agents the collaborators instead of humans. As 126
such, our work contributes to the general research area 127
where intelligent agents are used to generate the recom- 128
mendations (Kaelbling and Saffiotti 2005). Recommendation, 129
in related studies (Maximilien and Singh 2002), is 130
associated with the reputation of the business entity or the 131
web services that the entity provides. Managing reputation 132
is a large body of work that captures historical data and 133
context such that the best service can be selected based on a 134
community of opinions (Malaga 2001; Rezgui et al. 2003). 135
The notion of recommendation proposed in this paper is not 136
associated with reputation as the approach builds relevance 137
based on the usage of specific context-related strings in the 138
users' actions and those used to represent the candidate 139
services. 140

Upon finding services through the recommendation 141
model, our agents take an extra step by implementing the 142
web service into the business process. In this sense, our 143
work is closely aligned to the goals of the Business Process 144
Execution Language (BPEL; Leymann and Roller 2002). 145
This paper expands on the earlier related work by analyzing 146
the performance of the agent interactions using baselines 147
from related frameworks (Nowlan and Blake 2007a, b). In 148
addition, this paper formalizes the data models required for 149
agents to be able to manage discovery in registries. 150

151 3 An agent-based recommendation framework

This work extends earlier work that also investigates web 152
service recommendation but focuses on the algorithms for 153
matching web services to user preferences (Blake and 154
Nowlan 2007). In this work, the focus is on the *commu-* 155
nication of distributed agents that act as proxies for 156
organizations to find web services that possibly might fulfil 157
an ongoing need. By understanding the ultimate goals of 158
recommending web services, this work uses the low-level 159
interaction protocols of agents and data interchange models 160
are examined to produce standard communicative acts 161
among distributed agents. 162

A typical scenario for this approach would be an 163
organization that manages a long-standing supply chain 164
for some business purpose. In a service-oriented environ- 165
ment, such a supply chain potentially uses web services as 166

167 sub-capabilities. In this dynamic environment, new web
 168 service-based capabilities come available on the network on
 169 an ongoing basis. Organizations participating in these types
 170 of business processes require automated systems that can
 171 identify the entry of new capabilities and suggest those
 172 capabilities for use in existing business processes. These
 173 automated processes or intelligent agents correlate the
 174 subject matter associated with a business process with the
 175 potential capabilities of new web services. The centrepiece
 176 of this work is the collaboration between multiple agents
 177 that leads to knowledge sharing about available web
 178 services. In general, there are several types of data that
 179 must be shared

- 180 • *Subject matter associated with new web services.* The
 181 subject matter of the service is derived from the text
 182 strings that describe the service name, input/output
 183 messages, and other specification data.
- 184 • *Subject matter of the business process.* Since business
 185 processes residing on service-oriented architectures
 186 revolve around electronic exchange of information,
 187 intelligent agents can scan the underlying text informa-
 188 tion and keywords and use them to compare with
 189 keywords associated with new web services that enter
 190 the network.
- 191 • *Categories of web services.* Since it is impractical to
 192 search the entire web service repository each time,
 193 intelligent agents maintain a subset of keywords that
 194 describe a particular category of services. These key-
 195 words can be used to categorize new web services and
 196 effective constrain the search.

197 **3.1 Internal agent functions**

198 Our agent federation is composed of multiple agents
 199 representing business processes in a networked setting.
 200 Moreover, we believe agents in this federation should have
 201 the same implementation as the web services that they
 202 manage. The agents themselves are web services (Huhns
 203 2002), capable of receiving requests for information and
 204 providing results for those requests. The ultimate goal of
 205 agents is to be able to recommend the most relevant web
 206 services to the individuals they represent. Although this
 207 singular notion is the starting point, agents may have
 208 multiple functions they can perform.

209 To support recommendation, the main internal function
 210 of agents is the ability to query Universal, Description,
 211 Discovery and Integration (UDDI) registries to discover
 212 useful and relevant services (UDDI 2006). To determine
 213 relevancy, an agent uses syntactic and semantic methods of
 214 comparing the similarity between the web services defi-
 215 nitions (Web Service Description Language Documents
 216 [WSDL]) and pertinent strings extracted from a business

model (W3C 2002). Extracting descriptive strings (i.e. 217
 messages, semantic information, and text) and determining 218
 their similarity to the local web services is the second 219
 capability of agents and it enables agents to compose a bag 220
 of words that describe the operational domain of their 221
 business. Thirdly, an agent categorizes the web services 222
 existing locally in order to stratify the services for efficient 223
 and fast searching. Since precision is better in a heteroge- 224
 neous repository (many unique strings), then depending on 225
 the uniqueness of each category, an agent adjusts the 226
 strictness of its similarity method in order to make more 227
 accurate recommendations in various domains. Figure 1 228
 illustrates the internal functions of an agent, as previously 229
 mentioned (although making a recommendation is an 230
 internal function, it is performed after extensive collabora- 231
 tion with other agents and thus, it is described in the next 232
 section). 233

3.2 Agent communication and collaboration 234

Our federation can be thought of as a collection of 235
 “friendly” agents that work together on operations. Com- 236
 munication and collaboration are necessary components of 237
 this environment. The federation allows agents to ask other 238
 agents for help with a particular recommendation. In this 239
 way, agents help each other with both the discovery and the 240
 recommendation of web services. Consulting agents agree- 241
 ing on a particular service provides additional strength to 242
 that recommendation. After capturing information about the 243
 current business process, an agent will tag several services 244

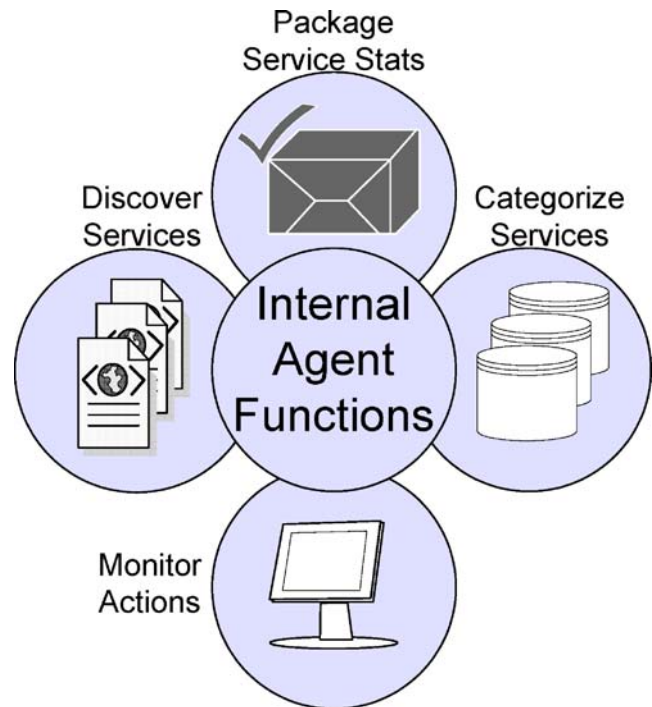


Fig. 1 Internal agent capabilities

Print will be in black and white

245 that are most relevant to its user. However, before
 246 recommending or integrating these services, an agent will
 247 query the other agents in an attempt to discover new
 248 services it did not know existed, and/or to seek confirma-
 249 tion or disapproval. Figure 2 depicts one such query an
 250 agent makes to its peers. First, an agent sends its extracted
 251 information to its peers.

252 Agents in the federation are implemented as web
 253 services, and their interaction protocols are implemented
 254 via the exchange of SOAP messages. An innovation in this
 255 paper is the development of an information model
 256 associated with the necessary agent-communicated data
 257 within the federation. By analyzing the interaction proto-
 258 cols, we introduce this new communication data model.
 259 The data attributes in Table 1 represent the result of our
 260 analysis and the aggregation of information that is passed
 261 repetitively throughout the interaction protocols defined in
 262 later detail in this paper. We identify these attributes as
 263 eXtensible Markup Language (XML) *elements* relevant to
 264 the communication data model. By examining our commu-
 265 nication protocols we were able to determine which pieces
 266 of information were in constant transition, and using this
 267 knowledge and the new elements, we created the *Service*

Table 1 SRML message information attributes

Keyword	XML	Description	
Bag of words	<i>WordBag</i>	String collection describing a SOC action or user file	t1.1
Category name	<i>CatName</i>	The name of a category of services	t1.2
Category self-similarity	<i>SimPcnt</i>	Percentage of self-similarity of a category	t1.3
Task ID	<i>tkID</i>	Identifier for an action	t1.4
Agent ID	<i>agID</i>	Identifier for an agent	t1.5
Group of services	<i>ServList</i>	A list of candidate or reference web services	t1.6
Response constraint	<i>RespC</i>	Time constraint for a query response	t1.7
Entrance threshold	<i>EnterT</i>	Threshold of similarity needed to recommend to another agent	t1.8
Alignment threshold	<i>AlignT</i>	Category similarity threshold needed to create an agent agreement.	t1.9
End message	<i>close</i>	Boolean to end exchange	t1.10
Category equivalency list	<i>CatHash, initCat, extCat</i>	List equating categories of two agents (initiating agent and external agent)	t1.11
			t1.12
			t1.13

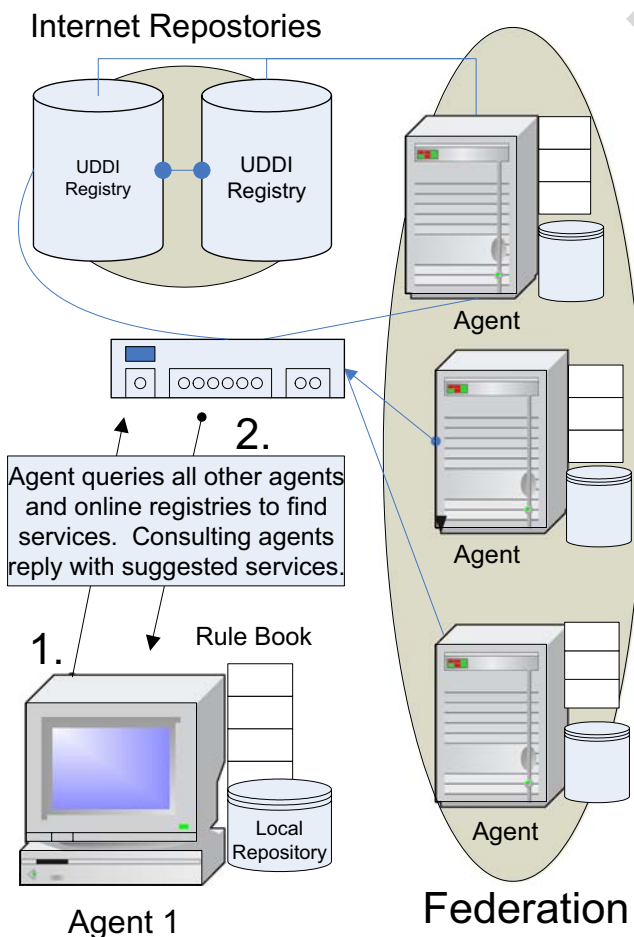


Fig. 2 Agent-mediated sharing for service management

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 informa-
 tion 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 mark-up format,
 making cross-domain collaboration possible. Figure 3 out-
 lines the SRML schema used by agents to interpret SOAP
 messages. This schema will be exploited in later sections.

4 Agent-to-agent communication protocols

Communication makes the tasks of web service discovery and
 recommendation a collective process, rather than an indepen-
 dent action. Sharing knowledge in a service-oriented archite-
 cture can be divided into two categories, service-based
knowledge acquisition and *knowledge mediation*. In a
 knowledge acquisition scenario, agents acquire and dissemi-
 nate information about services and capabilities. In the
 knowledge acquisition category, an agent recommends a
 service to a user (i.e. *Service Recommendation*). This is the
 primary function of an agent in the federation. An agent may
 also send a request to multiple agents, or one agent in
 particular, asking for web services to populate its own local
 repository (i.e. *Service Sharing*). Newly created agents use this
 communication to initially populate their local repositories.

Print will be in black and white

```

<xs:schema xmlns:xsd = "http://www.w3.org/2001/XMLSchema"
  xmlns:alters = "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>
  
```

Q1 Fig. 3 New agent exchange model: SRML schema

295 In the knowledge mediation category, agents must
 296 reconcile their shared knowledge. Agents group the
 297 services in their local repository into *categories* of similar
 298 services. An agent must internally evaluate its services and
 299 analyze them to enhance recommendation (i.e. *Category*
 300 *Analysis*). It is likely that over time, one agent's local
 301 categories will resemble another agent's, and an alignment
 302 of these categories makes recommendation easier so that
 303 agents can search within roughly similar categories across
 304 their separate domains. This second task is called *Category*
 305 *Alignment*. The taxonomy for service-based knowledge
 306 sharing is illustrated in Fig. 4.

307 The types of messages we use to complete communica-
 308 tion protocols extend the FIPA ACL Communication
 309 library. The protocols follow an ordered exchange of the
 310 message types to carry out a communicative act. Different
 311 message types contain different pieces of information. *Call*
 312 *for Proposal*, *Accept Proposal*, *Agree* and *Confirm* are
 313 called the "handshake" message types because they are
 314 used at the beginning of communicative acts to establish a
 315 working relationship. These messages contain little infor-

mation other than task identification, agent identification 316
 and the type of act. *Inform*, *Propose* and *Request* messages 317
 often contain the information an initiating agent wants to 318
 send to a consulting agent. *Inform Reference* and *Subscribe* 319
 messages contain the response information a consulting 320
 agent shares with the initiating agent. A *Failure* message 321
 ends communication between agents. The collection and 322

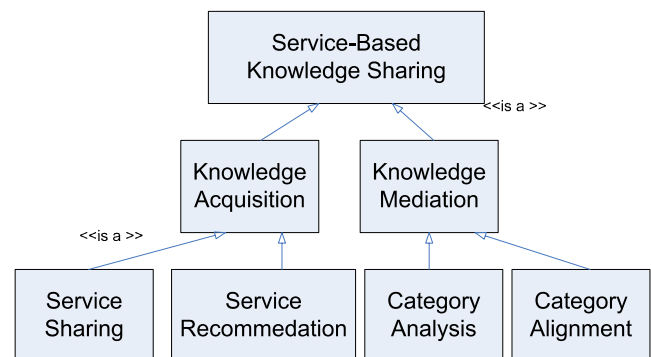


Fig. 4 Focused taxonomy of knowledge sharing

Print will be in black and white

323 exchange of different sets of these message types make up
 324 the four communication protocols.

325 4.1 Collaborative service recommendation protocol

326 The cornerstone to our approach is timely and effective
 327 recommendation of web services. In order to do this, an
 328 agent gathers information from their business model's daily
 329 routine and collects these descriptive strings into a "bag of
 330 words". The agent finds web services in its local repository
 331 that are similar to the extracted words using a matching
 332 approach called TSM-LP (Nowlan et al. 2006). The
 333 beginning of communications usually start with a "hand-
 334 shake" between agents in the form of a *Call for Proposal*
 335 and *Accept Proposal*. The agent then sends the extracted
 336 strings to all accepting agents in the federation (*Request*),
 337 who, in turn, do the exact same procedure on their local
 338 repositories. After that, the consulting agents respond to the
 339 original agent with the services that they found to be most
 340 similar to the bag of words (*Inform*). With its own original
 341 recommendation and the suggested services of all the other
 342 agents, the original agent finds the most commonly
 343 appearing services and recommends them for inclusion
 344 into its business process. The original agent also sends a
 345 message to the collaborating agents to share the result of
 346 the recommending process (*Inform*). This protocol is
 347 loosely depicted in the previous Fig. 2, and it is pictured,
 348 in more detail, in Fig. 5. Figure 6 illustrates an agent
 349 message for Step 3 (i.e. Request: Bag of Words) in the
 350 protocol shown in Fig. 5.

351 4.2 Category alignment protocol

352 Based on the varying natures of business domains, different
 353 agents will have different local repositories. One way to
 354 organize local repositories is through stratification into
 355 categories of similar services. Although the number and
 356 makeup of these categories can vary a great deal, it is likely
 357 that two unique agents will have at least a few categories
 358 that contain very similar services. Because of this likeli-
 359 hood, 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 communi-
 cative 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. 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 7 depicts this message exchange.

4.3 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 cate-
 gories. 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 recom-
 mended service is. This protocol deals with sharing these
 category sensitivities between agents after they have
 aligned their repositories.

Agents engaging in this protocol have already collabo-
 rated 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 8 depicts this message exchange.

4.4 Service sharing protocol

As business needs evolve over time, agents will increase
 and reduce the services in their local categories dynamical-
 ly. Agents may find that certain categories are obsolete to
 their business model. Conversely, agents may also find

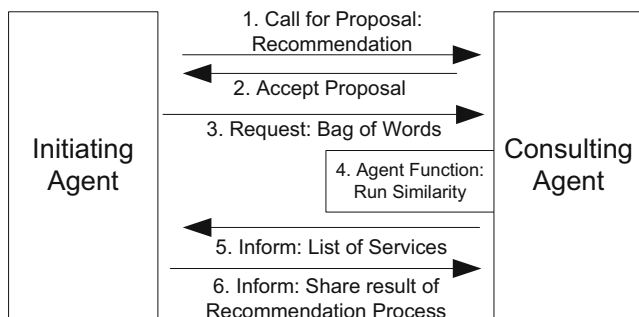


Fig. 5 Collaborative recommendation protocol

```
<SRML tkID = 201 agID = ebay08 close = "N">
  <WordBag/> paperback, journal, article, technology,
  service-oriented
  <CatName/> BookPurchase
  <RespC units = "seconds"/> 360
</SRML>
```

Q1 Fig. 6 Sample SRML message for request: Bag of words

409 certain categories to be overwhelmingly applicable and
 410 desire to proactively find more services of the same nature.
 411 In these cases, an agent can ask other agents for a small
 412 collection of services of a given type. In the recommenda-
 413 tion protocol, consulting agents send services based on
 414 similarity to a business action, this protocol deals simply
 415 with the sharing of services based on a stated category (i.e.
 416 type of service). This protocol is also invoked when a new
 417 agent registers to the federation, and needs to populate its
 418 local repository from scratch.

419 The “handshake” for this protocol is nearly identical to
 420 that of the recommendation protocol, with the only
 421 difference in the semantics of the proposal (*Call for*
 422 *Proposal* and *Accept Proposal*). After agreement to
 423 communicate, the initiating agent sends a Category Name
 424 or descriptive word for the type of services it desires
 425 (*Request*). The consulting agent responds with the Category
 426 Name from its local repository and a Service List of the
 427 most frequently used services in that category (*Inform*
 428 *Reference*). The initiating agent takes this Service List and
 429 stratifies the services, if needed, into local categories based
 430 on the nature of the services. Figure 9 depicts this message
 431 exchange.

432 5 Connecting agents to service registries

433 The Universal Description, Discovery, and Integration
 434 (UDDI) specification is a standard for the design and
 435 implementation of web service registries. Web services in
 436 these registries can be defined based on five UDDI entities,
 437 *Business Entity*, *Business Service*, *Binding Template*,
 438 *TModels*, and *KeyedReference*. The Business Entity

describes the organization by name and other attributes. 439
 The Business Service describes the web service by name 440
 and key. The Binding Template has references to the 441
 interface and messages associated with a particular Busi- 442
 ness Service. Both TModels and KeyedReferences serve as 443
 meta-data for the Business Entities, Business Services, and 444
 Binding Templates. TModels and KeyedReferences have a 445
 hierarchical relationship which allows the ability for 446
 implementing meta-data for meta-data. An overview of 447
 the UDDI entities is illustrated in Fig. 10. 448

449 The approach presented in this paper builds on the 449
 UDDI specification with regards to how agents store 450
 category information about web services. Businesses in a 451
 web service registry will naturally expose one or many web 452
 services. Based on the category for which a business’s 453
 services belong, the agent federation uses KeyReferences, 454
 specifically CategoryBags to define the type. Each Business 455
 Service and corresponding Binding Templates are defined 456
 by TModels. To connect business entity-based meta-data 457
 with business services meta-data, the TModels are then 458
 associated with relevant CategoryBags. Based on this 459
 approach, business entities and business services can be 460
 extracted independently. Furthermore, the connection of 461
 TModels to CategoryBags allows web services to be 462
 correlated with their corresponding business entities. These 463
 relations are illustrated in Fig. 11. We have experimented 464
 on manipulating these data models using Hewlett Packard’s 465
 Systinet Registry. 466

6 Performance assessment and evaluation 467

In order for the agent federation to be feasible in practice, the 468
 recommendations must be presented to the user in a 469
 reasonable timeframe. Considering the average time it takes 470
 a user to scan a new web site or read a medium-sized 471
 message, the agent-federated recommendation time is based 472
 on a 5–6 s response time ceiling. In evaluating the agent 473

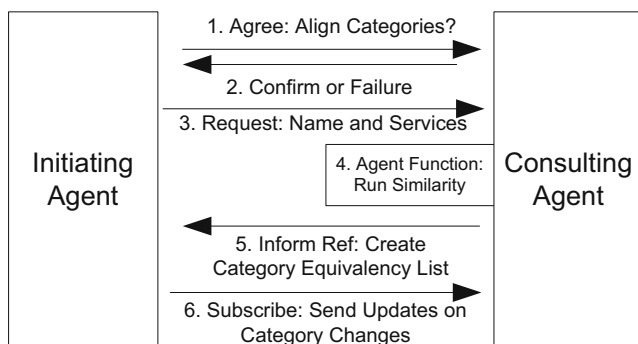


Fig. 7 Category alignment protocol

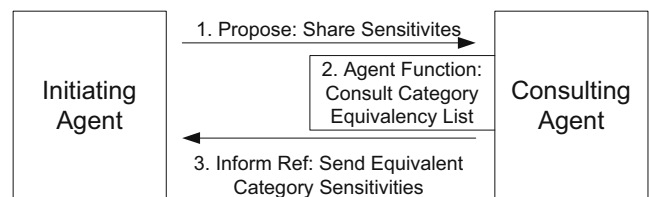


Fig. 8 Category analysis protocol

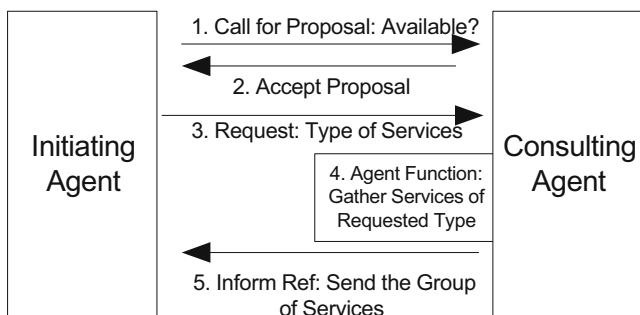


Fig. 9 Service sharing protocol

474 federation, it is important to measure the system response
475 time against this benchmark.

476 At the time of this paper, only the recommendation
477 scoring functionality has been developed which is reported
478 in more detail in other work (Nowlan et al. 2006). Of
479 importance to this analysis is the fact that the time for
480 creating a recommendation score (i.e. internal agent
481 function) is approximately 2 s. Although this time is well
482 below the reasonable response time of 5–6 s to the user,
483 when the necessary messaging overhead is taken into
484 account, the time approaches 5–6 s.

485 In this analysis and of importance to the focus of this
486 paper, we are interested the combined performance of agent

487 communication *and* the recommendation score generation. 487
488 In order to determine an estimated communication response 488
489 time, we investigated related work containing performance 489
490 measurements of tools associated with agent communica- 490
491 tion. GigaSpaces is a collaboration framework typically 491
492 used for inter-process (i.e. inter-agent) communication (The 492
493 GigaSpaces Platform 2007). Although industrial perfor- 493
494 mance reports show that Gigaspaces performs under 4 ms 494
495 (irrespective of number of entries) when reading the shared 495
496 space, it does not address the inter-process messaging. 496
497 JADE is perhaps the leading agent framework. 497

498 Both Cortese et al. (2002) and Ahuja et al. (2005) agree 498
499 that the inter-process messaging overhead is about 8– 499
500 10 ms per concurrent process (agents) considering a 500
501 reasonable network infrastructure and hardware. The 501
502 values given in Cortese et al. (2002) is based on JADE, 502
503 Ahuja et al. (2005) based their measures on GigaSpaces 503
504 and CORBA. In our analysis, we build on the experimen- 504
505 tal benchmarks from this related work to establish an 505
506 estimated agent communication time. We then add the 506
507 experimental recommendation generation time (calculated 507
508 in this work) to develop and estimate for the agent 508
509 communication protocols. Based on related work, ex- 509
510 changing 7 character messages for 10,000 trips on a 510

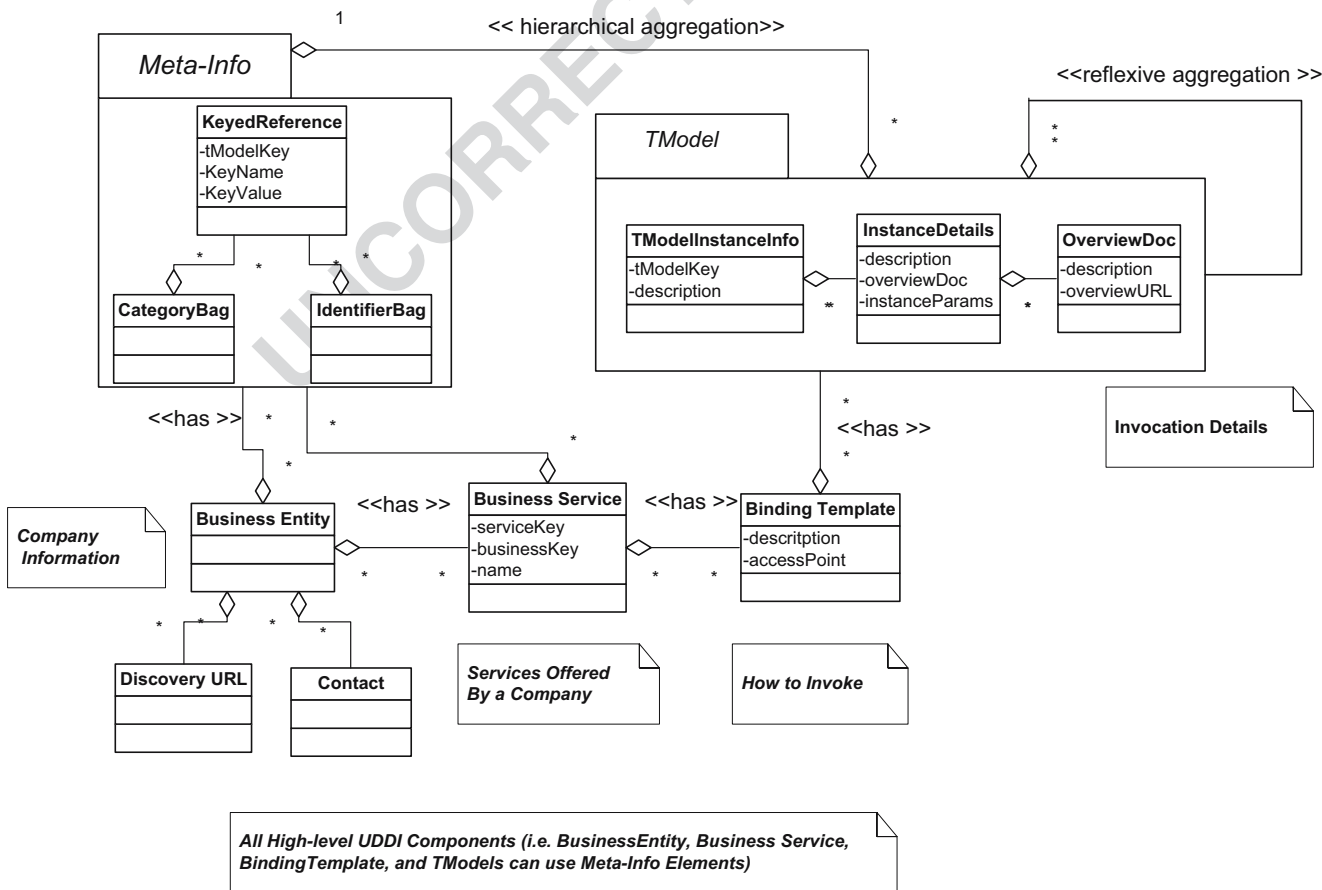


Fig. 10 Overview of UDDI model

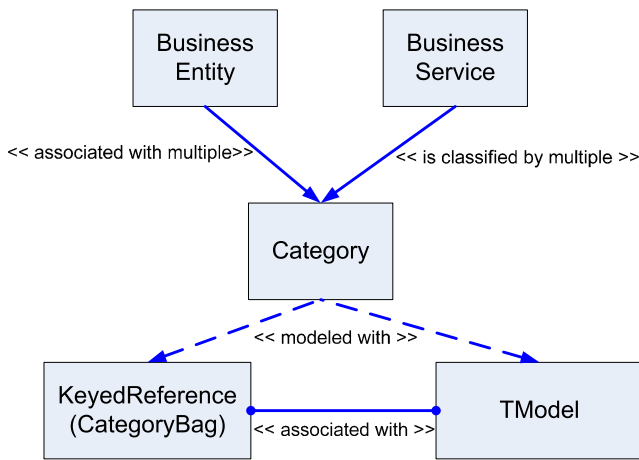


Fig. 11 Integrating the agent federation with the UDDI model

511 reasonable hardware platform results in the following
512 performance times:

When the number of agents is between 0 and 100 :
 Message Time = 10 ms × (Number of Agents)
 When number of agents is between 100 and 2,000 :
 Message Time = 8 ms × (Number of Agents)

515 In order to analyze, the protocols introduced in this
516 paper, we used the sum of messages required for the
517 protocol. We also estimated message size based on a
518 typical SRML message populated with typical recommen-
519 dation request information. For example, the recommen-
520 dation protocol, the category analysis/alignment protocol,
521 and the service sharing protocol required five, seven, and
522 four messages respectively. The related work suggests the
523 communication of seven characters with 10,000 trips
524 while the agent communication is on the order of 7,000
525 character messages for 10 trips. Consequently, after
526 extrapolating data size and number of round-trips per
527 recommendation, the following communication-only per-

formance measures were calculated for the recommenda- 528
 tion protocol: 529

When the number of agents is between 0 and 100 :
 Message Time = 5 ms × (Number of Agents)
 When number of agents is between 100 and 2,000 :
 MessageTime = 4 ms × (Number of Agents)

In further analysis, we assumed that the other protocols 532
 (i.e. the service sharing protocol, category analysis proto- 533
 col, and category alignment protocols) that happen concu- 534
 rrently with the recommendation protocols would add 535
 additional overhead. Through related calculations we found 536
 that if all of the peripheral protocols are executed for each 537
 recommendation, then the response time is tripled. Figure 12 538
 shows the estimated communication/messaging time as it 539
 relates to concurrent agents. Other measures show the 540
 performance impact associated with executing the other 541
 services (i.e. category analysis, alignment, and service 542
 sharing) 50% of the time and 25% of the time. 543

Figure 13 expresses the total estimated recommendation 544
 time, which includes the calculation the recommendation 545
 score. A tradeoff for system performance and maintaining 546
 current information is updating other services only 25% of 547
 the time. Using this level of service, the recommendation 548
 response time is about 6 s for 250 collaborating agents. This 549
 result would suggest that our framework should optimize 550
 performance by limiting agent collaboration to the top 250 551
 most relevant agents within the federation. The authors feel 552
 that this is a reasonable quality of service that assures both 553
 the fidelity of the recommendations and the speed. 554

The agent federation and communication protocols are 555
 effective because they enable communication between 556
 agents using service-oriented approaches. By introducing 557
 SRML, there is a generic data structure for agents to share 558
 knowledge in the form of recommendations irrespective of 559
 the domain (e.g. entertainment, sports, travel, and finance). 560
 Another benefit of the approach is that agents in the future 561
 may be able to integrate with web services by allowing the 562

Print will be in black and white

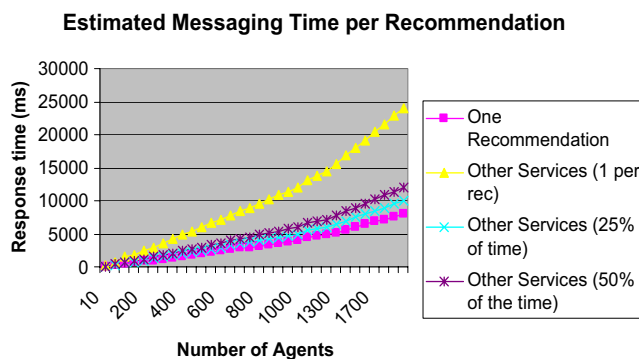


Fig. 12 Communication time per recommendation

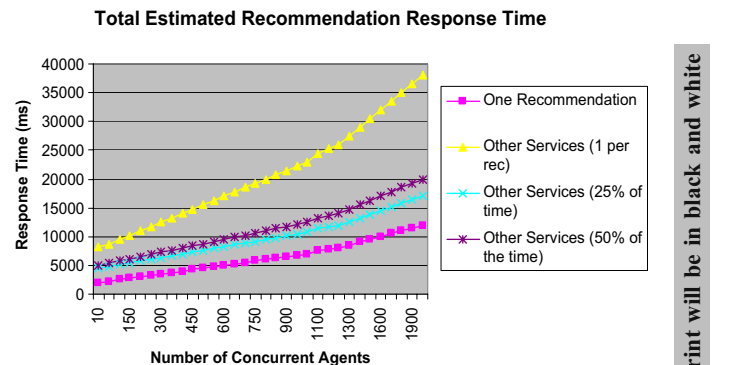


Fig. 13 Estimated recommendation time

Print will be in black and white

563 web services to generate the SRML documents directly
564 from their WSDL representations.

565 As a limitation to this approach, it may be necessary to
566 regulate how many and which agents can be queried for a
567 single business action. The performance analysis shows that
568 in order to get a 6 s response time, it will require that the
569 250 most relevant agents be consulted for on-demand
570 recommendation generation. Furthermore, limiting the
571 number of consulting agents for any one query might aid
572 an initiating agent in receiving more specific recommenda-
573 tions, but it also limits the scope of the knowledge base.

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

581 7 Conclusion

582 In this work, we propose agent-to-agent communication
583 and consultation to provide relevant recommendations of
584 web services. Collaboration is a necessary and primary
585 member of an agent's rule-based decision making. By
586 forcing constant communication between agents, agents
587 become more knowledgeable about the services they
588 recommend and their relation to other business processes
589 in the federation. Our federation allows agents to learn "on
590 the fly" and incorporate new web services into their
591 business domain continually. This architecture enacts this
592 communication protocol, allowing for comprehensive web
593 service recommendation in real time. The agent-to-agent
594 schema we developed enables fast and uniformed message
595 protocols within the architecture. By limiting recommenda-
596 tion response time, we assure that only agent's that can be
597 of service are included in collaborative measures. Finally,
598 by conforming to the 5–6 s window of collaboration, the
599 message protocols and the SRML schema support the
600 effectiveness of this approach.

602 **Acknowledgement** This work is partially funded by National
603 Science Foundation, Arlington, Virginia, USA, under award number
604 0548514.

605 References

606 Ahuja, S. P., Eggen, R., & Jha, A. K. (2005). A performance evaluation
607 of distributed algorithms on shared memory and message passing
608 middleware platforms. *Informatica*, 29(3), 327–333.
Q2 609 Alechina, N., Logan, B., & Whitsey, M. (2004). *Modeling communi-
610 cating agents in timed reasoning logics*. Proceedings of the Ninth

European Conference on Logics in Artificial Intelligence (JELIA 611
2004) (pp. 95–107). 612
Birukou, A., Blanzieri, E., D'Andrea, V., Giorgini, P., Kokash, N., & 613
Modena, A. (2007). IC-service: A service-oriented approach to 614
the development of recommendation systems. *Proceedings of the* 615
22nd annual ACM symposium on applied computing: 2007 Mar 616
11–15; Seoul, Korea. New York, USA: ACM Press. 617
Blake, M. B., & Nowlan, M. F. (2007). Recommending web services 618 Q3
via an agent federation. *International Transactions on Systems*
Science and Applications, 3(2), (in press). 619
Blake, M. B., Fado, D. H., & Mack, G. A. (2006). Proactive service 620
discovery and execution using intelligent agents. *Proceedings of* 621
the 4th IEEE international conference on web services; 2006 Sep 622
18–22; Chicago, USA (pp. 463–470). Washington, USA: IEEE 623
Computer Society. 624
Burke, R. (1999). The wasabi personal shopper: A case-based 625
recommender system. *Proceedings of the 16th national confer-* 626
ence on artificial intelligence and the 11th innovative applica- 627
tions of artificial intelligence conference innovative applications 628
of artificial intelligence; 1999 Jul 18–22; Orlando, USA (pp. 629
844–849). Menlo Park, USA: American Association for Artificial 630
Intelligence. 631
Burke, R. (2002). Hybrid recommender systems: Survey and experi- 632
ments. *User Model User-Adapt Interact*, 12(4), 331–370. 633
Cortese, E., Quarta, F., & Vitaglione, G. (2002). Scalability and 634
performance of jade message transport system. In Proceedings of 635
the AAMAS Workshop on AgentCities, Bologna, Italy. Available 636
from: [http://sharon.cselt.it/projects/jade/papers/Final-ScalPerf](http://sharon.cselt.it/projects/jade/papers/Final-ScalPerfMessJADE.pdf) 637
[MessJADE.pdf](http://sharon.cselt.it/projects/jade/papers/Final-ScalPerfMessJADE.pdf). 638
Dong, X., Halevy, A., Madhavan, J., Nemes, E., & Zhang, J. (2004). 639
Similarity search for web services. *Proceedings of the 30th VLDB* 640
Conference; 2004 Aug 19–Sep 3; Toronto, Canada (pp. 372– 641
383). St Louis, USA: Morgan Kaufmann Publishers. 642
FIPA. (2006). Agent Communicative Act Specification Library, 643 Q2
Foundation for Intelligent Physical Agents, [http://www.fipa.org/](http://www.fipa.org/specs/fipa00037/index.html) 644
[specs/fipa00037/index.html](http://www.fipa.org/specs/fipa00037/index.html). 645
Grosz, B. (1997). Building commercial agents: An ibm research 646
perspective. In: Crabtree, B. (ed.) *Proceedings of the second* 647
international conference on the practical applications of intelligent 648
agents and multi-agent technology; 1997 Apr 21–23; London, 649
England. Blackpool, UK: Practical Application Company Ltd. 650
Huhns, M. N. (2002). Agents as web services. *IEEE Internet* 651
Computing, 6(4), 93–95. 652
Kaelbling, L. P., & Saffiotti, A., (eds.) (2005). Multi-agent information 653
retrieval and recommender systems. *Proceedings of the 19th* 654
international joint conference on artificial intelligence; 2005 Jul 655
30–Aug 5; Edinburgh, Scotland. Denver, USA: Professional 656
Book Center. 657
Leymann, F., & Roller, D. (2002). Business process in a web services 658
world [Online]. Available from: URL: [http://www.ibm.com/develop-](http://www.ibm.com/developerworks/library/ws-bbelwp/) 659
[erworks/library/ws-bbelwp/](http://www.ibm.com/developerworks/library/ws-bbelwp/), 2002 Aug 1 [cited 2007 June 16]. 660
Malaga, R. A. (2001). Web-based reputation management systems: 661
Problems and suggested solutions. *Electronic Commerce Re-* 662
search, 1(4), 403–417. 663
Manikrao, U. S., & Prabhakar, T. V. (2005). Dynamic selection of web 664
services with recommendation system. *Proceedings of the* 665
international conference on next generation web services 666
practices; 2005 Aug 22–26; Seoul, Korea (pp. 117). Washington, 667
USA: IEEE Computer Society. 668
Maximilien, E. M., & Singh, M. P. (2002). Conceptual mode of web 669
service reputation. *ACM SIGMOD Record*, 31(4), 36–41. 670
Nowlan, M. F., & Blake, M. B. (2007a) Intelligent agent communi- 671
cation and collaboration for web service management. *Proceed-* 672
ings of the 16th IEEE international workshops on enabling 673
technologies: infrastructures for collaborative enterprises. Paris, 674
France. IEEE Press (Jun 18–20). 675
676

- 677 Nowlan, M. F., & Blake, M. B. (2007b). Agent-mediated knowledge
678 sharing for inter-organizational services management. *Proceed-*
679 *ings of the 2007 IEEE international conference on services*
680 *computing*. Salt Lake City, USA. IEEE Press (Jul 9–13).
- 681 Nowlan, M. F., Kahan, D. R., & Blake, M. B. (2006). *Using naming*
682 *tendencies to syntactically link web service messages*. *Data*
683 *engineering issues in E-commerce and services* (Vol 4055/2006,
684 pp. 90–99). Berlin (Germany): Springer (lecture notes in
685 computer science).
- 686 Rezgui, A., Bouguettaya, A., & Malki, Z. (2003). A reputation-based
687 approach to preserving privacy in web. *Proceedings of the 29th*
688 *international conference on very large data bases*; 2003 Sep 9–
689 12; Berlin, Germany. Berlin: Springer.
- 690 Sarwar, B., Karypis, G., Konstan, J., & Reidl, J. (2001). Item-based
691 collaborative filtering recommendation algorithms. *Proceedings*
692 *of the 10th international conference on world wide web*; 2001
693 *May 1–5; Hong Kong, China*. New York, USA: ACM Press.
- 694 GigaSpaces (2007). The GigaSpaces Platform: Performance Report.
695 Available from: <http://www.gigaspace.com/download/GigaSpacesPerformanceReport.pdf>.
- 696
- 697 OASIS (2006). Universal, Description, Discovery and Integration.
698 Oasis UDDI [Online]. Available from: <http://www.uddi.org/>.
- W3C (2002). Web services description working group [Online]. Available 699
from: <http://www.w3.org/2002/ws/desc/>, cited June 16, 2007. 700
- M. F. Nowlan** is currently an undergraduate research assistant in the 704
Department of Computer Science at Georgetown University pursuing 705
a Bachelor of Science in Computer Science. He currently conducts 706
research in the intelligent exchange of messages and data for web 707
services in service-oriented architectures. He has co-authored 6 journal 708
articles and refereed proceedings in these areas. 709
- M. B. Blake** is Associate Professor and Chair of Computer Science at 712
the Georgetown University. He received his Ph.D. in Information and 713
Software Engineering from George Mason University and a Bachelor 714
of Electrical Engineering from Georgia Institute of Technology. His 715
current research focuses on service-oriented computing, intelligent 716
agents for workflow management, distributed data management and 717
software engineering education. Dr. Blake has published over 80 718
journal articles and refereed proceedings in these research areas. 719

UNCORRECTED PROOF

AUTHOR QUERY

AUTHOR PLEASE ANSWER QUERY.

- Q1. Tables 2 and 3 were changed to Figures 3 and 6. Citations for the following were also changed and Figures were renumbered. Please check if appropriate.
- Q2. References “Alechina et al. 2004 and FIPA 2006” was listed but not cited. Please check.
- Q3. Please provide bibliographic update for Blake and Nowlan 2007.

UNCORRECTED PROOF