Matchmaking for Information Agents

Daniel Kuokka Larry Harada *
Lockheed Palo Alto Research Labs, O/96-20, B/255
3251 Hanover Street, Palo Alto, CA 94304
kuokka@aic.lockheed.com, harada@aic.lockheed.com

Abstract

Factors such as the massive increase in information available via electronic networks and the advent of virtual distributed workgroups for commerce are placing severe burdens on traditional methods of information sharing and retrieval. Matchmaking proposes an intelligent facilitation agent that accepts machine-readable requests and advertisements from information consumers and providers, and determines potential information sharing paths. We argue that matchmaking permits large numbers of dynamic consumers and providers, operating on rapidly-changing data, to share information more effectively than via current methods. This paper introduces matchmaking, as enabled by knowledge sharing standards like KQML, and describes the SHADE and COINS matchmaker implementations. The utility and initial results of matchmaking are illustrated via example scenarios in engineering and consumer information retrieval.

1 Introduction

The trend toward computer-based tOOlS for many aspects of commerce has led to a rapid increase in distributed virtual workgroups, such as multi-vendor design teams and virtual corporations. In addition, the advent of the Internet, personal computer networks, and interactive television networks has led to an explosion of information available on-line from thousands of new sources. These phenomena offer great promise for obtaining and sharing diverse information conveniently, but they also present a serious challenge. The sheer multitude, diversity, and dynamic nature of on-line information sources makes finding and accessing any specific piece of information extremely difficult.

To address this problem, several exciting new technologies have been developed. The standards and protocols of the World Wide Web, as well as its associated browsers, have provided a hugely successful dissemination framework for previously disassociated information. Furthermore, integration frameworks from CAD vendors and telecommunications companies provide information connectivity where there was none before. However, both of these employ address-based messaging or browsing paradigms—the users must know where the information exists. Unfortunately, as users try to make the transition from adventurous explorers to goal-driven information seekers, it becomes very difficult to find desired information. The pearls are lost in a sea of irrelevant information.

In response to this problem, two common solutions have appeared: clearinghouses and exploration agents. Clearinghouses, such as CommerceNet and MCC's EINet Galaxy, are central servers at which individual information providers can register. Since there are relatively few clearinghouses, consumers are able to effectively locate desired information. Exploration agents, such as Lycos [Mauldin and Leavitt, 1994] and the World Wide Web Worm [McBryan, 1994], "crawl" the network compiling a master index. The index can then be used as the basis for keyword searches much like a manually-created clearinghouse.

These approaches provide very useful solutions to the overflow of information, but several problems remain. First, as the number and size of clearinghouses grow, they degenerate into a duplication of the network, itself (an interesting phenomenon is that many clearinghouses are becoming cross-indexed, allowing each to benefit from the knowledge-base of the others). Thus, inefficiencies and difficulties in locating a specific piece of information are still present. Also, exploration is a computationally inefficient approach (in terms of bandwidth, processor, and memory utilization), so it is usually applied sparingly, and therefore provides a limited index of the subject network.

More fundamentally, the above approaches make the assumption that information producers are (mostly) passive, forcing consumers to drive the process. This necessarily imposes several handicaps:

*This work was supported in part by ARPA Contract DAAA 15-91-C0104 (Shared Knowledge-Based Technology for the Re-Engineering Problem), monitored by the U.S. Army Research Laboratory, Advanced Computational and Information Systems Directorate. The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

- Information consumers must know of or arduously locate all relevant providers. However, today's networks are composed of millions of potential information sources, each of which may provide information dynamically. Thus, discovering all sources is very difficult.
- Information providers have no way to contribute their efforts. Even though producers often have a stake in delivering their information, and would therefore be willing to assist in the process, this potential goes unutilized.
- Once a connection is made, there is no means by which a provider can notify a consumer of new knowledge or updates to past queries. Thus, in contexts where information is updated frequently and dynamically, approaches where the provider is passive simply can't work.

2 Matchmaking

A different approach to addressing this problem is called matchmaking. Matchmaking is based on a cooperative partnership between information providers and consumers, assisted by an intelligent facilitator [Genesereth, 1992] utilizing a knowledge sharing infrastructure [Patil *et al.,* 1992]. Information providers take an active role in finding specific consumers by *advertising* their information capabilities to a matchmaker. Conversely, consumers send *requests* for desired information to the matchmaker. The matchmaker attempts to identify any advertisements that are relevant to the requests and notifies the providers and consumers as appropriate.

Matchmaking is an automated process depending on machine-readable *messaging* and *content* languages. The main advantage of this approach is that the providers and consumers can continuously issue and retract advertisements and requests, so information does not tend to become stale. This is particularly critical in situations where information changes rapidly, as in product development and crisis management, and in situations where the shear magnitude of providers and consumers would cause the clearinghouse to be updated nearly continuously. A matchmaker is somewhat like a blackboard, except that it exists as a separate agent, the shared information tends to be highly structured in terms of knowledge-sharing protocols, and specific matchmaking algorithms are used. (The term agent is used in this paper to refer to a tool or program, possibly under the guidance of a human, that consumes or provides information to other agents.)

The content language must allow broad classes of information (i.e., many different documents) to be conveyed succinctly; otherwise, very many highly-specific messages, essentially duplicating the clients' databases, would be required. Whereas this provides useful representational economy and efficiency, it dictates that advertisements and requests are only approximate versions of the actual information. Thus, matchmaking is approximate, and false positive and false negative matches (depending on whether the advertisements and requests are over- or under-general) are likely to occur.

As variations on the general theme, matchmaking can follow many different specific modes. For example, the consumer might simply ask the matchmaker to *recommend* a provider that can likely satisfy the request. The actual queries then take place directly between the provider and consumer. The consumer might ask the matchmaker to forward the request to a capable provider with the stipulation that subsequent replies are to be sent directly to the consumer (called *recruiting).* Or, the consumer might ask the matchmaker to act as an intermediary, forwarding the request to the producer and forwarding the reply to the consumer (called *brokering).*

An implicit form of the last case, called *content-based routing,* is also possible, where a consumer simply *subscribes* to information as if the matchmaker were the source. The providers, rather than advertising their capabilities, simply send out changes as they occur. The matchmaker then routes the specific changes on to the subscriber. These different modalities (which correspond to several existing KQML message types as described in section 3) are shown in Figure 1.

An additional variation is on the persistency of the request. Consumers often desire to be told not only about providers that have already advertised a relevant capability, but also about any providers that advertise a capability in the future. In this case, the consumers would issue a persistent version of the above requests. This is essential, for instance, in the Parameter Manager example described in Section 5, in which a new consumer or provider with pertinent constraints may come on-line at any time.

There are many potential modes of matchmaking beyond those summarized in figure 1. As pointed out previously, one of the benefits of matchmaking is that it allows providers to take a more active role in information retrieval. Whereas the above schemes allow providers to advertise their capabilities dynamically, providers still cannot identify potential consumers unless the consumer actually issues a specific statement of interest. A useful extension would be to allow a provider to request the names of consumers that have posted related interests. This raises serious privacy considerations (imagine a consumer asking for a list of automobile dealerships only to be bombarded by sales offers from all of the dealerships), but in some cases, it may be desirable for a matchmaker to present a consumer with potential information providers spontaneously. Anonymity of the consumer is maintained, yet providers have an avenue for solicitations.

To evaluate and test the matchmaking approach, two prototype matchmakers have been built. The first matchmaker was designed and prototyped as part of the SHADE system [Kuokka and Harada, 1995a; McGuire *et al,* 1993]. The SHADE matchmaker supports many modes of operation over formal, logic-based representations. The second matchmaker, created as an element of the COINS system (Common Interest Seeker) operates over free-text information, supporting fewer modes. The implementation of each of these systems is outlined in the following sections. Other researchers are also working on facilitators, such as the ABSI facilitator [Singh, 1993],

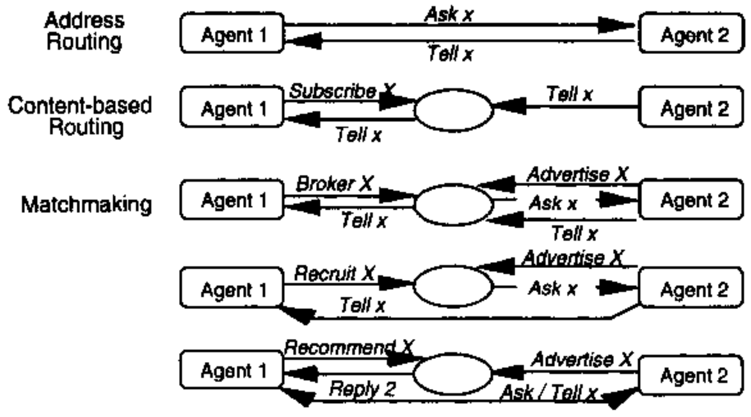

Figure 1: Modes of information routing

which also perform certain matchmaking functions. A detailed description of protocol issues related to matchmaking is covered in [Kuokka and Harada, 1995b].

The decision to implement two distinct matchmakers rather than a single, fully capable matchmaker was initially motivated by non-technical issues. However, it turns out that the resulting modularity is appropriate and beneficial in the agent-based world. This approach allows many matchmakers, each created by researchers with specific technical expertise, to be specialized for specific classes of languages. Such a distributed approach may also address pragmatic issues of scalability, but little effort has been applied in this area to date. If desired, a single, multi-language matchmaker may be implemented via a simple dispatching agent that farms out requests to the appropriate subcontracting agent.

3 SHADE Matchmaker

Matchmaking depends heavily on several technologies: an appropriate *messaging language* in which the client agents express their requests (e.g., the form of advertisement or request), an expressive *content representation* used to encode the actual information to be advertised and requested, and an effective matching algorithm.

The SHADE matchmaker communicates in terms of KQML (Knowledge Query and Manipulation Language [Finin *et al*., 1993]) messages. Advertisements are sent using the KQML advertise performative (message type). Requests are sent using the recommend, recruit, and broker performatives. The matchmaker also supports content-based routing via the KQML performatives tell and subscribe.

For example, the KQML message

```
(advertise :sender p .-receiver mm :lang kqml
   :content
   (ask-one :lang kif
      :content
      (subcomponent-of ?x ?y)))
```

advertises the capability to answer queries (ask-one) about the component hierarchy, and the message

```
(recruit-all :sender c :receiver mm :lang kqml
   :content
   (ask-one :lang kif
      :content
      (subcomponent-of gimbal ?x))))
```

asks the matchmaker to locate an agent that can answer the query: "What is the parent component of the gimbal?[11]

As its content language, the SHADE matchmaker supports two logic-based representations: a subset of KIF [Genesereth and Fikes, 1992], used in the above example, and a structured logic representation called MAX [Kuokka, 1990] augmented to support string patterns as terms. KIF is supported since it provides an expressive, standardized shared language with well-defined semantics. MAX is supported since it is more appropriate for representing highly structured data such as objects and frames. It essentially allows partial templates of frames to be advertised and requested. Furthermore, with the string matching augmentation, it provides a convenient means for advertising and requesting semi-structured text, such as outlines. For example, the message

```
(subscribe :sender c :receiver mm :lang kqml
   :content
   (ask-about :lang max
      :content
      [(trouble-report ?x)
       (match ?x [(subject ".*gimbal.*")])]))
```

subscribes to trouble report objects that have the string "gimbal" in their subject field.

The actual matching of advertised and subscribed content fields is performed by a Prolog-like unification algorithm. If strings are present in the logic forms, a regular expression pattern matcher is used for term unification.

Advertisements and requests must match based solely on their content; there is no knowledge base against which inference is performed. For example, an advertisement containing the term "engine" would not match an isomorphic request containing the term "propulsion system," since the matchmaker does not know that an engine is a subclass of a propulsion system. To address this issue, the SHADE project has also been developing technology to define *ontologies* [Gruber, 1993]—knowledge bases that define shared concepts. As ontologies become available, future versions of the matchmaker will include the capability to perform limited inference based on the specific ontology specified in the KQML message, allowing the above example to match.

This path must be followed carefully, however, since an arbitrary amount of inference or knowledge may be required to match any given advertisement and request. The matchmaker could quickly be transformed from a communication facilitator to a multi-domain reasoning engine. This would violate a key tenet of the agent-based approach—that of utilizing many different domain-specific agents. Therefore, the tendency to enhance the capabilities of the matchmaker must be tempered by the desirable separation of functionality underlying the network of agents.

The SHADE matchmaker is implemented entirely as a declarative rule-based program within the MAX forward-chaining agent architecture. This allows features of the matchmaker (e.g., support for additional KQML performatives) to be added as additional rules. For example, the rule that implements the broker request is shown below.

```
(rule
 [(pre
   [(message broker-one ?broker)
    (match ?broker [(content ?content)
                    (language kqml)
                    (sender ?sender)
                    (receiver ?receiver)])
    (match ?content [(perf ?bperf)
                     (language ?blang)
                     (content ?bcontent)])
    (advertisement ?bperf ?advertisement)
    (match ?advertisement
           [(language ?alang)
            (content ?acontent)
            (advertiser ?advertiser)])
    (not-match ?sender ?advertiser)
    (match ?blang ?alang)
    (match ?bcontent ?acontent)])
  (post
   [(advertisement ?bperf ?advertisement)
    (brokering ?advertiser ?sender)
    (kqml-message [(perf reply)
                   (receiver ?sender)
                   (sender ?receiver)
                   (content received)])
    (kqml-message [(perf ?bperf)
                   (receiver ?advertiser)
                   (sender ?receiver)
                   (content ?bcontent)])])])
```

In addition, the SHADE matchmaker supports meta-level queries about its operation. This feature allows other agents to subscribe to the message level actions of the matchmaker in addition to content level information. For example, another SHADE agent called the Bird's Eye View agent uses this feature to subscribe to all advertisements and requests, regardless of content. The Bird's Eye agent then displays the system of agents and message traffic. The meta-reasoning capabilities are also used to provide reasons for match failures in certain cases (the Space Imaging application described in section 6 uses this feature).

Count	Concept
97	matchmak
53	inform
45	content
38	advert
33	agent
33	match

Table 1: A portion of document vector for this paper

4 COINS Matchmaker

Motivated by the utility of the SHADE matchmaker on structured information and by the need for similar functionality over the huge amount of text available on-line, a second matchmaker has been created that operates on free-text as its content language. This matchmaker was initially conceived as the central part of a system called COINS (COmmon INterest Seeker), which allows users to easily advertise and request information about their interests. However, by architecting the system as a set of agents, the COINS matchmaker is also useful as a general purpose facilitator.

As with the SHADE matchmaker, the COINS matchmaker is accessed via the standard KQML messages advertise and broker. The content language is either free-text or a concept vector (a weighed list of stemmed words in the document). An example of a portion of the concept vector for this paper is shown in Table 1.

To determine if a free-text request matches a free-text advertisement, the content of each is converted into a concept vector using the SMART [Salton, 1989] information retrieval system. The SMART matching algorithm is then used to determine the degree of match. Finally, an adjustable cutoff measure is used to make the match binary. Thus, other than supporting a different content language, the COINS matchmaker works much like the SHADE matchmaker.

SMART employs an inverse document frequency scheme so the COINS matchmaker must maintain and use a local concept corpus. This functions somewhat like the ontology of the SHADE matchmaker in that it is a knowledge base of shared concepts allowing the match process to be more effective.

5 Application: Collaborative Engineering

The SHADE and COINS matchmakers are being used as a central component of several research projects. The

SHADE project, itself, is exploring broader infrastructure issues in support of distributed engineering. To this end, SHADE has developed a testbed for collaborative engineering to motivate and test infrastructure components such as the matchmaker.

To illustrate the utility of a matchmaker in an engineering environment, consider the following scenario showing a few steps in the design of a satellite. The engineering team consists of a Systems Engineer, responsible for specifying the overall architecture of the satellite; a Designer, responsible for designing the geometry and structure of a gimbal on the satellite; and a Mass Specialist, who allocates the mass budgets to individual subsystems.

The participants use a number of engineering tools that consume and produce complex engineering information. Each participant uses the Project Coordination Assistant (PCA) [Kuokka, 1994], which allows engineers to view and manipulate textual and structured data on the project such as the satellite component hierarchy and trouble reports, and the Parameter Manager (ParMan) [Kuokka and Livezey, 1994], which allows many different engineers to define constraints over shared parameters. In addition, each engineer may use any number of CAD tools specific to his or her discipline.

Initially, the Systems Engineer uses the PCA to request notification about any unresolved problems. This is translated into the following matchmaker subscription:

```
(subscribe :sender syseng :receiver mm
  :content
  (tell :lang max
    :content
    [(newpage [(item ?newitem)])
     (match ?newitem [(text "problem")])
     (oldpage ?opage)
     (not-match ?opage [(item ?newitem)])]))
```

Notice that the form of the content is designed such that a literal will match the interest template only the first time it is added to the page. Otherwise, if a pattern that matches the interest template exists within a page, every subsequent change to that page would result in a repeated notification, even if the pattern, itself, did not change.

Next, using the PCA, the Systems Engineer adds the mass budget attribute (mass-bgt) to the gimbal object in the global satellite topology database, resulting in a new page being sent to the matchmaker.

```
(tell .-sender syseng :receiver mm :lang max
  :content
  [(newpage
    [(item [(text "Structure")
            (item [(text "Gimball")
                   (item [(text "mass")])
                   (item [(text "mass-bgt")])
                   ...]) ...])])
   (oldpage
    [(item [(text "Structure")
            (item [(text "Gimball")
                   (item [(text "mass")])
                   ...]) ...])])])
```

By virtue of a previous subscription to changes to the gimbal (similar to the System Engineer's subscription to unresolved problems), the Gimbal Designer receives notice of the new mass budget attribute.

The new parameter is relevant to his subsystem, so the designer imports the new parameter to his ParMan tool and enters the constraint that the actual mass of the gimbal must be less than the mass budget. Since the ParMan tool is designed to handle distributed constraints, it must attempt to locate other agents that have constraints over the new parameter. This results in the following messages being sent to the matchmaker.

```
(recruit-all :sender gimball-pm :receiver mm
  :content
  (subscribe :lang kqml
    :content
    (stream-about :lang kif
      :content
      (mass gimbal-1)))

(advertise :sender gimball-pm :receiver mm
  :content
  (subscribe :lang kqml
    :content
    (stream-about :lang kif
      :content
      (mass gimbal-1)))
```

Since the mass specialist is responsible for allocating the mass budgets, she also defines a constraint, which results in similar advertisements and requests being posted by her ParMan agent. The matchmaker matches the advertisements and requests for the mass budget posted by each ParMan and routes the requests on to the other ParMan agents. This allows each ParMan agent to locate all other sources of relevant constraints, and ultimately identify that the budget as supplied by the mass specialist is inconsistent with the actual mass of the gimbal.

At this point, we assume that the designer cannot restructure the gimbal to meet the budget demands, so he posts an open problem via PCA. This results in the following message being sent to the matchmaker.

```
(tell :sender gimball :receiver mm :lang max
  :content
  [(newpage [(item [(text "Problems")
                    (item [(text
     "Gimbal 1 cannot satisfy mass budget")])
                    (item [(text
     "Dual controller hysteresis occurring")])
                    ...]) ...]
   (oldpage [(item [(text "Problems")
                    (item [(text
     "Dual controller hysteresis occurring")])
                    ...]) ...])])
```

This matches with the System Engineer's earlier subscription, so he receives notification of this problem. Thus, by facilitating the dynamic connection of relevant information sources, a problem that might have gone unnoticed for many days was identified and propagated to concerned participants within minutes. A diagram, generated by the Bird's Eye View agent based on its

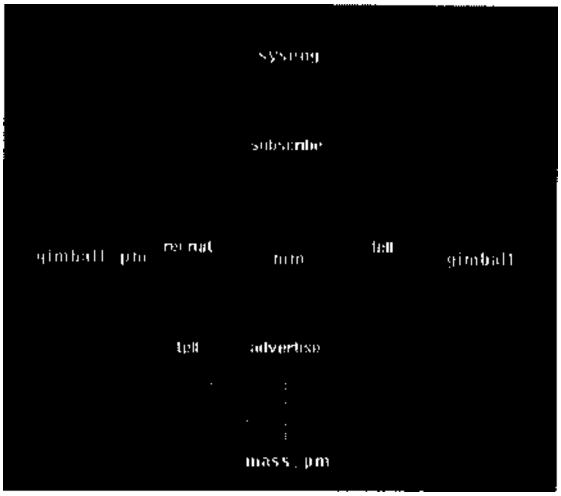

Figure 2: Key message traffic among agents

meta-level subscription to the matchmaker is shown in Figure 2.

The key to the matchmaking approach is that it avoids the need to identify a priori all potential information transfer paths, which is impossible in general due to the dynamic nature of engineering teams. This is especially important if the project is considered in its full context, where there are hundreds of engineers, scores of gimbals, hundreds of other components, and thousands of parameters and constraints. Many engineers might be using the Parameter Manager to state their constraints on the parameters of specific interest to them. When any one engineer decides to add a constraint, he has no way of knowing exactly which other engineers are impacted, and therefore whom should be notified. This is solved by each Parameter Manager sending advertisements and subscriptions to the matchmaker for the specific parameters of concern, allowing all agents to locate the new sources and sinks of information for this specific, unforeseeable engineering need.

The matchmaker is also vital to the operation of collaboration tools like the PCA. As described above, when the Systems Engineer created a monitor for problem reports, a KQML subscribe message was sent to the matchmaker. As other changes were made to the PCA data, they were forward to the matchmaker. This allowed the matchmaker to route relevant changes to subscribers. Not shown in the scenario is that as changes are made to PCA pages, the PCA also transforms the semi-structured text into concept vectors and sends them as subscriptions to the COINS matchmaker. As other agents do the same, the matchmaker monitors for pairs of concept vectors that match (according to the approximate concept vector match criterion). If a pair is found, pointers are returned to the PCA and the other agent. The PCA then adds them to a dynamic list of relevant documents. Thus, the matchmaker is used to locate other information relevant to the contents of the PCA information base.

The matchmaker has been used by several other engineering-related projects as well. The Cosmos project [Mark and Dukes-Schlossberg, 1994], which is creating a knowledge-based commitment reasoner to determine impacts of engineering changes, uses the matchmaker to provide indirection between a set of dynamic clients and the server. The ARPA Simulation Based Design project uses the matchmaker to provide change subscription and notification services over its large, object-oriented product model. In this application, if an object for which a subscription has been issued changes, the user will receive automatic notification. Other applications of the matchmaker, such as its use to locate relevant pages in a large distributed engineering notebook, are in earlier stages of development.

6 Application: Information Retrieval

The functionality of matchmaking goes beyond engineering teams. For example, the matchmaker is an integral part of a prototype information retrieval system being developed to support Lockheed's SII (Space Imaging, Inc) project, an effort to sell high-resolution satellite imagery on the commercial market. Since there are multiple satellite image providers, and numerous value-added post-processors, this task consists of locating data available from multiple dynamic sources in response to specific queries. Therefore, the SII prototype uses the SHADE matchmaker.

The system works as follows. As new classes of images become available, the data sources issue advertisements in terms of the image attributes (e.g., geographic area, resolution, spectral bands, and cloud cover). When a user requests a specific kind of image, a front-end agent issues a query to the matchmaker that describes the desired attributes. The matchmaker compares the advertisements to the queries, and sends any matches to the fronr-end agent. This first-pass match is used to locate servers for further, more specific, queries. In addition, when a source database is not appropriate, the matchmaker returns a failure reason.

The matchmaker is important to the SII application not only because there are multiple sources of data, but also because the data is constantly being updated as satellites circle the earth. The matchmaker allows each data source to advertise and retract its image capabilities dynamically, permitting the matchmaker to suggest sources even if the specific image hasn't yet been collected. Only an automated system like the matchmaker can offer the up-to-the-minute location of data required by SII.

7 Conclusions

The growth of information available via electronic networks presents both an unprecedented opportunity and a difficult challenge. Rather than relying on traditional techniques that are consumer-driven, matchmaking allows both of the stake holders (i.e., information providers and consumers) to contribute to information gathering activities. Thus, information providers can seek specific consumers much like consumers currently find specific providers. In addition, since matchmaking is an automated approach, it better addresses the dynamic nature of electronic information, which is based on huge numbers of potential information providers and consumers. The need for such an approach is underscored by the rapid adoption of the SHADE and COINS prototype matchmakers by several projects.

However, matchmaking is still an experimental approach, and many questions remain. Additional support is required for formal languages such as object and terminological representations, and a capability to load relevant knowledge bases and ontologies is needed to permit matchmaking based on subsumption reasoning and inference (however, the matchmaker cannot become the reasoning engine to the world). Also, further expansion into free-form human languages and graphics is needed, going beyond the current concept vector abstraction of text. Looking beyond the content language, the experiments with matchmaking to date have already begun to stretch the KQML messaging substrate. Further augmentations are required to support additional modalities and to clarify the semantics of the existing message types. And finally, as applications grow in size and complexity, techniques to distribute the matchmaker load will be required. Yet, in spite of these open issues, matchmaking is a promising approach to supporting information access in heterogeneous and dynamic environments.

Acknowledgments

The initial concept of matchmaking and the KQML formalization stemmed from efforts by the entire SHADE team, including Jim McGuire, Jay Weber, Greg Olsen, Rich Pelavin, Tom Gruber, and Marty Tenenbaum, as well as the efforts of the KQML committee. We also thank Brian Livezey for many insights and Morton Hirschberg for his invaluable support. This work was supported in part by ARPA prime contract DAAA15-91-C0104, monitored by the U.S. Army Research Laboratory.

References

[Finin et at, 1993] T. Finin, J. Weber, G. Wiederhold, M. Genesereth, R. Fritzson, D. McKay, J. McGuire, R. Pelavin, S. Shapiro, and C. Beck. Draft specification of the KQML agent-communication language. Technical report, The ARPA Knowledge Sharing Initiative External Interfaces Working Group, 1993.

[Genesereth and Fikes, 1992] M. Genesereth and R. Fikes. Knowledge Interchange Format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992.

[Genesereth, 1992] M. Genesereth. An agent-based framework for software interoperability. In Proceedings DARPA Software Technology Conference, 1992.

[Gruber, 1993] T. Gruber. A translation approach to portable ontology specifications. Knowledge Acquisition, 5(2), 1993.

[Kuokka and Harada, 1995a] D. Kuokka and L. Harada. A communication infrastructure for concurrent engineering. Artificial Intelligence in Engineering, Design, Analysis, and Manufacturing, 1995.

[Kuokka and Harada, 1995b] D. Kuokka and L. Harada. On using KQML for matchmaking. In International Conference on Multiagent Systems, 1995.

[Kuokka and Livezey, 1994] D. Kuokka and B. Livezey. A collaborative parametric design agent. In Proceedings of the National Conference on Artificial Intelligence, pages 387-393, Menlo Park, CA, 1994. AAAI Press.

[Kuokka, 1994] D. Kuokka. An evolution of collaborative design tools. In AAAI-94 Workshop on Models of Conflict Management in Cooperative Problem Solving. AAAI Tech. Report WS-94-04, 1994.

[Kuokka, 1990] D. Kuokka. The Deliberative Integration of Planning, Execution, and Learning. PhD thesis, School of Computer Science, Carnegie Mellon University, 1990.

[Mark and Dukes-Schlossberg, 1994] W. Mark and J. Dukes-Schlossberg. Cosmos: A system for supporting engineering negotiation. Concurrent Engineering: Research and Applications, 2(3), 1994.

[Mauldin and Leavitt, 1994] M. Mauldin and J. Leavitt. Web-agent related research at the CMT. In Proceedings of the ACM Special Interest Group on Notworked Information Discovery and Retrieval (SIGNIDR-94), 1994.

[McBryan, 1994] 0. McBryan. WWWW—the World Wide Web Worm, http://www.cs.colorado.edu/home /mcbryan/WWWW.html, 1994.

[McGuire et al, 1993] J. McGuire, D. Kuokka, J. Weber, J. Tenenbaum, T. Gruber, and G. Olsen. SHADE: Technology for knowledge-based collaborative engineering. Concurrent Engineering: Research and Applications, 1(3), 1993.

[Patil et al, 1992] R. Patil, R. Fikes, P. Patel-Schneider, D. McKay, T. Finin, T. Gruber, and R. Neches. The DARPA Knowledge Sharing Effort: Progress report. In Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann, 1992.

[Salton, 1989] G. Salton. Automatic Text Processing—The Analysis, Transformation and Retrieval of Information by Computer. Addison-Wesley, Reading, MA, 1989.

[Singh, 1993] N. Singh. A CommonLisp API and facilitator for ABSI (revision 2.0.3). Technical Report Logic-93-4, Stanford University Computer Science Department Logic Group, 1993.