JIIS, 6, 261–279 (1996)
© 1996 Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

Integrating Information via Matchmaking

DANIEL KUOKKA AND LARRY HARADA kuokka@aic.lockheed.com
Lockheed Palo Alto Research Labs, O/96-20, B/255, 3251 Hanover Street, Palo Alto, CA 94304

Abstract.
Trends such as the massive increase in information available via electronic networks, the use of on-line product data by distributed concurrent engineering teams, and dynamic supply chain integration for electronic commerce are placing severe burdens on traditional methods of information sharing and retrieval. Sources of information are far too numerous and dynamic to be found via traditional information retrieval methods, and potential consumers are seeing increased need for automatic notification services. Matchmaking is an approach based on emerging information integration technologies whereby potential producers and consumers of information send messages describing their information capabilities and needs. These descriptions, represented in rich, machine-interpretable description languages, are unified by the matchmaker to identify potential matches. Based on the matches, a variety of information brokering services are performed. We introduce matchmaking, and argue that it permits large numbers of dynamic consumers and providers, operating on rapidly-changing data, to share information more effectively than via traditional methods. Two matchmakers are described, the SHADE matchmaker, which operates over logic-based and structured text languages, and the COINS matchmaker, which operates over free text. These matchmakers have been used for a variety of applications, most significantly, in the domains of engineering and electronic commerce. We describe our experiences with the SHADE and COINS matchmaker, and we outline the major observed benefits and problems of matchmaking.

Keywords: Information, brokering, retrieval, agents

1. Introduction

The advent of the Internet, large consumer-oriented networks, and company Intranets has led to an explosion of information available on-line from thousands of new sources. In addition, structural changes in manufacturing and commerce such as interdisciplinary design teams, virtual corporations, and electronic commerce have increased the need for sharing electronic information. These phenomena present a serious challenge to the information consumer and provider. 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 these changes, several exciting new technologies have been developed. The protocols, servers, and browsers of the World Wide Web provide a revolutionary means to deliver and access on-line information. Integration frameworks from CAD vendors and telecommunications companies provide information connectivity where there was none before. Other efforts, such as the ARPA Intelligent Integration of Information (I3) program (Hull and King, 1995), are striving to make databases more accessible.

Ironically, since most of these technologies are focused on making more information available, their success has has contributed even more to the overflow of on-line information without offering assistance in locating and keeping up-to-date on that information. Nearly all of these approaches employ address-based requests (common to both the browsing and database query paradigms) in which the users must know where the information exists.

For example, whereas the Web is extremely useful for browsing, 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 get lost in a sea of irrelevant information.

In response to this problem, several solutions have appeared: clearinghouses, mediators, and exploration agents. Clearinghouses, such as CommerceNet, Yahoo, and MCC's EINet Galaxy, are central servers where individual information providers can register. Since there are relatively few clearinghouses, consumers are able to locate desired information relatively effectively. Mediators (Wiederhold, 1992) are systems that combine, transform, or abstract information in other sources. For example, the mediators of TSIMMIS (Chawathe et al., 1994) provide database transparency, allowing queries to be directed to an intermediary which handles many different databases. Similarly, SIMS(Arens et al., 1993) handles queries that span multiple databases. Since many databases can be fronted by a mediator, the problem of finding data is simplified. Exploration agents such as Lycos (Mauldin and Leavitt, 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. Mediators are aimed mainly at presenting a value-added view of other data, not at integrating a huge number of sources. 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 and out-of-date 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:

- Information consumers, or their agents, 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 information sources relevant at any given time is 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 un-utilized.

- 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 are very inefficient.

What is needed is an automated, knowledge-based information resource that dynamically connects information consumers with information providers, scalable to the scope of the Internet and World Wide Web. Matchmaking is an example of such a service/ other projects such as Agent Based Software Integration (Genesereth, 1992, Singh, 1993) have created similar systems as well. This article describes matchmaking in general, describes two implemented matchmakers, and discusses their use and benefits drawing on experiences from two applications.

2. Matchmaking

Unlike the traditional model of information pull, Matchmaking is based on a cooperative partnership between information providers and consumers, assisted by an intelligent facilitator (the *matchmaker*). Information providers and consumers update the matchmaker (or network of matchmakers) with their needs and capabilities. The matchmaker, in turn, notifies consumers or producers of promising "partners". Matchmaking is an automated process depending on machine-readable communication among the consumers, providers, and matchmaker. Thus, communication must occur via rich, formal knowledge sharing languages (Patil et al., 1992).

The main advantage of this approach is that the providers and consumers can continuously issue and retract information needs and capabilities, so information does not tend to become stale and the flow of information is flexible and dynamic. This is particularly critical in situations where sources and information change rapidly, as in commerce, product development and crisis management, and in situations where the shear magnitude of providers and consumers would cause a manually-maintained clearinghouse to be updated nearly continuously.

Unlike a mediator, a facilitator does not add value or transform information—it focuses on locating and propagating information. However, matchmakers may employ mediators, and they often serve as the intermediary for all communication, giving the appearance of value-added processing. Automated matchmaking places requirements on those information sources and consumers that use it, in that they must describe their capabilities and needs formally and dynamically. We use the term *agent* to refer to an information source or consumer communicating at this level. Since facilitators and mediators also use these langauges, they are also classified as agents.

There are two distinct levels of communication with a matchmaker: the message type (sometimes called the *speech act*) and the content. The former denotes the intent of the message (e.g., query or assertion) while the latter denotes the information being exchanged (e.g., what information is being queried or asserted). There are a variety of message types. For example, information providers can 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. As variations on this general theme, 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. 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.

As pointed out previously, one of the benefits of matchmaking is that it allows providers to take a more active role in information retrieval. Thus, just as a request can be viewed as an effort to locate an information provider, an advertisement can be viewed as an effort to locate a consumer's 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). Fortunately, the various modes of matchmaking can include exchanges that preserve either party's anonymity.

In addition to a formal protocol of advertisement and request, matchmaking depends on rich, machine interpretable descriptions of the information being advertised or requested. The description language used for matchmaking must allow large collections of information (i.e., a database) to be conveyed *succinctly*, since it is usually impractical to advertise every detail (e.g., each fact in the database). Whereas this provides useful representational economy and efficiency, it dictates that advertisements and requests are only partial descriptions of the actual information. Thus, matchmaking is often imperfect, and false positive and false negative matches are likely to occur depending on whether the advertisements and requests are too general or too specific.

Since the content of requests and advertisements may not align perfectly, satisfying a request might involve aggregating or abstracting the information to produce an appropriate result. For example, if a source advertises information about automobiles while a consumer requests information about Fords, some knowledge and inference is required to deduce that a Ford is an automobile. Such transformation of data is an important capability, but its addition to a matchmaker must be carefully weighed. If knowledge about automobiles were added to a matchmaker, similar knowledge could be added about animals, airplanes, detergents, and every other possible topic. Obviously, this would quickly lead to an impractically large matchmaker. Therefore, a matchmaker strictly does not contain any domain knowledge. However, as mentioned above, a matchmaker is free to use other mediators and data sources in determining partners. Thus, it could farm out the automobile/Ford example to an automobile knowledge base to determine if a match exists.

3. Implementation

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, 1995, McGuire et al., 1993), a testbed for integrating heterogeneous tools in large-scale engineering projects. It operates over formal, logic-based representations, and is designed to support many different types of requests and advertisements. A second matchmaker was created as an element of the COINS system (Common Interest Seeker). The emphasis of this matchmaker is on matching free text rather than formal representations. The implementation of each of these systems is outlined in the following sections.

Both matchmakers run as processes accepting and responding to advertisements and requests from other processes. Communication occurs via KQML (Knowledge Query and Manipulation Language (Finin et al., 1993)), which defines specific message types (historically known as *performatives*) and semantics for advertising and requesting information.

KQML messages types include simple queries and assertions (e.g., `ask`, `stream`, and `tell`), routing and flow instructions (e.g., `forward` and `broadcast`), persistent queries (e.g., `subscribe` and `monitor`), and information brokering requests (e.g., `advertise`, `recommend`, `recruit`, and `broker`), which allow information consumers to ask a facilitator to find relevant information producers. The knowledge carried by a KQML message is referred to as the content, and may be in any content language. Thus, KQML supports the different content languages of the SHADE and COINS matchmakers. Even though KQML defines many other parameters such as `:language`, `:ontology`, `:reply-with`, we include only those parameters vital to the examples.

These matchmakers were developed separately due to the differences between their content languages (logic vs. free text), and the resulting radical impact on the matching algorithms. They could, in principle, be integrated, but just as a matchmaker uses other agents for domain-specific inference, it is preferable to keep them separate, rather than creating one huge matchmaker. If desired, a single, multi-language matchmaker may be implemented via a simple dispatching agent that farms out requests to the appropriate matchmaker based on the `:language` parameter. 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.

3.1. SHADE Matchmaker

The SHADE matchmaker is intended to dynamically connect information sources and consumers that speak KQML carrying formal, logic-based content languages. It allows agents to advertise their information capabilities, locate information sources, and request constant updates.

Advertisements are sent to the matchmaker using the KQML `advertise` performative (message type). Requests can take a variety of forms: `recommend`, `recruit`, and `broker`. The matchmaker also supports content-based routing via the KQML performatives `tell` and `subscribe`. These different modalities are illustrated in Figure 1.

For example, the KQML message

```
(advertise :sender p :receiver mm :language kqml :content
  (ask-one :language 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 :language kqml :content
  (ask-one :language 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?”

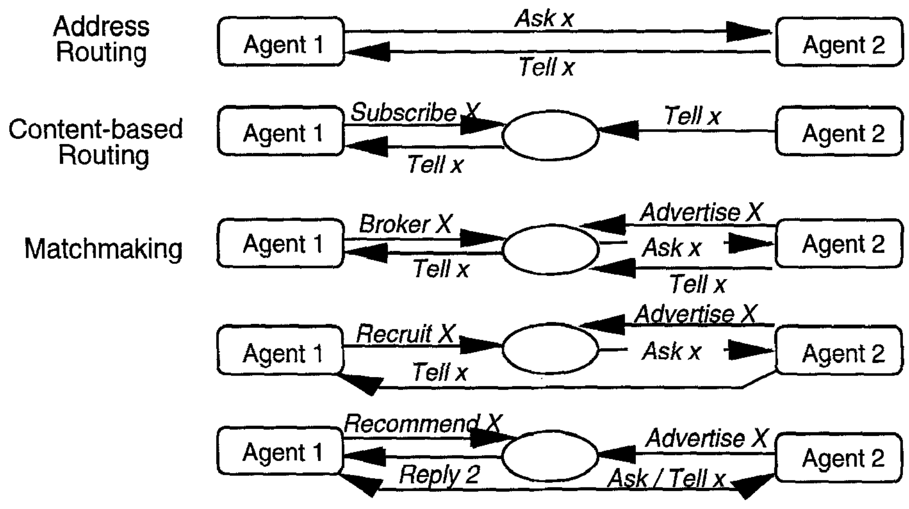

Figure 1. Modes of information routing

As its content language, the SHADE matchmaker supports two logic-based representations: a subset of KIF (Knowledge Interchange Format(Genesereth and Fikes, 1992)), used in the above example, and a structured logic representation called MAX(Kuokka, 1990). KIF is a form of full first order predicate calculus, and MAX is a conjunctive predicate logic augmented to support frames (conjunctive sets of literals) as first-class terms. It also supports string patterns as terms. For example, the message

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

subscribes to trouble report frames 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 and no 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 engines are a subclass of propulsion systems.

The matchmaker does not perform inference since an arbitrary amount of deduction or knowledge may be required to match any given advertisement and request. To do so would quickly transform the matchmaker 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. As described above, such inference can be delegated to domain-specific agents. Techniques for doing this efficiently are still under development.

The SHADE matchmaker is implemented entirely as a forward-chaining rule-based program (also using MAX). 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)])])])
```

At the core of the matching algorithm is a recursive call to match: `(match ?bcontent ?acontent)`. This performs a constrained bi-directional logical unification of the advertised content with the requested content. Several generic constraint predicates are supported, such as `greater-than` and `in-set`. Thus, both advertisements and requests can be more general than a set of ground literals.

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 4.2 uses this feature).

Table 1. A portion of a document vector for this paper

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

Finally, a important precondition of matchmaking is a shared *ontology*. Shared ontologies are needed to ensure that terms (such as "engine") have clear and consistent semantics. Otherwise, a match may be found or missed based on an incorrect interpretation of the requests. A number of projects, including SHADE, have been developing technology to define ontologies (Gruber, 1993), but further discussion is beyond the scope of this article.

3.2. COINS Matchmaker

There is a serious need for matchmaking over the huge amount of unstructured text emerging on the World Wide Web and other wide-area networks, since it is not practical to expect all Web documents to be expressed in KIF where traditional matching algorithms can be used. To determine the practicality and efficacy of matchmaking in this domain, 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 standard KQML messages. However, it does not use the brokering performatives such as *advertise* and *recruit*. Instead, it uses the content-based routing paradigm where clients send *tell*s, *ask*s, and *subscribe*s. The COINS matchmaker can handle two content languages: either free text or a document vector. A *document vector* is a weighted list of stemmed words in the document (a portion of the document vector for this article is shown in Table 1). Thus, document vectors are a space-efficient means of sending a large document. Upon receiving a free-text document, the COINS matchmaker immediately converts it into a document vector for processing, so a content of type `document vector` is isomorphic to a content of type `text`. For the local client to produce a document vector, it must use COINS-compatible stemming and noise word elimination techniques.

An example of a request (subscribe) is shown below. The content is a document vector that represents a summary of a satellite project called MACE. The content document vector contains unbound variables for the document name, owner, and match weight, followed by a list of concepts and frequencies (truncated for exposition). This allows the subscription to refer to all unknown documents that match the specified concepts, much like `(subcomponent-of ?satellite ?x)` specifies all unknown subcomponents of the satellite.

```
(subscribe :sender mace :receiver coins-mm :language kqml
  :reply-with mace-overview :content
  (stream-all :language document-vector :content
    (?document ?owner ?weight mass 4 engineer 6 control 9 parm 7
     h3 6 capabl 5 design 8 vibrat 1 linear 1 ul 6 mace 7 mount 1
     glob 5 structur 1 ent 1 gimb 5)))
```

When the matchmaker receives a tell message with a document vector (or document) that matches the above content, it will be forwarded to the subscriber. For example, shown below are several such tells representing documents exported by an agent for a related satellite project (FSAT). Notice that the reply is not the actual matched documents, since the COINS matchmaker does not store the document. Instead, it is a document vector, including the name of the document, the owner (as an email address), and the strength of the match.

```
(tell :sender coins-mm :receiver mace :language document-vector
  :in-reply-to mace-overview :content
  (/aic/shade/demos/fsat/demo.txt harada@aic.lockheed.com 0.68
   mass 4 engineer 6 control 9 parm 7 h3 6 capabl 5 design 8))

(tell :sender coins-mm :receiver mace :language document-vector
  :in-reply-to mace-overview :content
  (/aic/shade/demos/fsat/jitter.txt harada@aic.lockheed.com 0.48
   mass 4 engineer 6 control 9 parm 7 h3 6 capabl 5 design 8))
```

To process and match free text and document vectors, the SMART (Salton, 1989) information retrieval system is used. First, if the content type is free text, it is converted into a document vector using SMART's stemming and noise word removal (the text-to-document-vector library supplied to clients uses these same facilities). Then, the document vectors are compared using an inverse document frequency algorithm, where the frequency of a concept in a document is divided by its frequency in the entire corpus to determine its information content (how well it describes the document with respect to the corpus). An adjustable cutoff measure is used to make the comparison binary.

Since the COINS matchmaker uses an inverse document frequency scheme, it must maintain 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 across multiple clients. Unlike an ontology, however, there is no notion of semantics; the corpus simply gives an estimate of the relevance of the word. Therefore, even though "control" is the most frequent word in the subscription above, words like "vibrat*e*" and "gimb*al*" carry more information, due to their much smaller frequency across all documents.

The COINS matchmaker clients do not have to agree on a shared ontology explicitly, other than using a common natural language (obviously, COINS would not be successful at matching an English document to a French document). However, use of domain-specific ontologies would give better results, since words with different specific meanings in multiple domains would be used consistently. Furthermore, as COINS receives more advertisements and requests, it revises its corpus to obtain better estimates of the information content of words. In essence, this is the COINS matchmaker's attempt to incrementally learn the ontology already shared by its clients.

4. Test Applications and Results

The SHADE and COINS matchmakers have been tested as part of several applications. The main applications to date have been in collaboration support for large-scale engineering (as part of the SHADE(Kuokka and Harada, 1995), Cosmos(Mark and Dukes-Schlossberg, 1994),, Integrated Weapons Systems Database, and Simulation-Based Design(Davis et al., 1993) projects) and electronic commerce (for a satellite imagery clearinghouse and tools for supply chain integration). These are described below.

4.1. Collaborative Engineering

The first application of the matchmaker was in the domain of collaborative engineering. To understand the need for matchmaking in engineering, consider a large satellite project where there are engineers responsible for requirements, structural design, structural analysis, dynamics analysis, thermal analysis, optical analysis, etc. In addition, these disciplines are typically duplicated for each major subsystem (e.g., payload-one versus payload two). Thus, there can be many hundreds of engineers working separately, but since the various subsystems and analyses interact with each other across the entire system, any one decision can impact many other engineers. But exactly who is impacted is generally not known by the decision maker.

In addition to working on agent communication infrastructure, the SHADE project is attempting to integrate such design teams by building assistant agents—software tools that interact with the human engineer and with the other assistant agents. Many of these are created from scratch, such as Lockheed's Parameter Manager(Kuokka and Livezey, 1994), while others are created by *wrapping* existing tools, such as SDRC's I-DEAS solid modeler and Wolfram's Mathematica.

Several experiments have been run using matchmaking to facilitate the interactions among SHADE collaborative engineering tools. To illustrate one set of experiments, consider a small portion of an engineering team designing a satellite. The key participants are the 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). The PCA is a shared engineering notebook that allows engineers to view and manipulate textual and structured data on the project, such as the satellite component hierarchy and trouble reports. Via the matchmaker, it provides notification to interested parties (indicated via advertisements) upon changes to its pages. Participants also use the Parameter Manager (ParMan) (Kuokka and Livezey, 1994), which gives each engineer an interface to enter constraints over shared parameters. In addition, each engineer may use any number of CAD tools specific to his or her discipline. In our example, the I-DEAS solid modeler is used by the designer. The engineering team and their tools are shown in Figure 2.

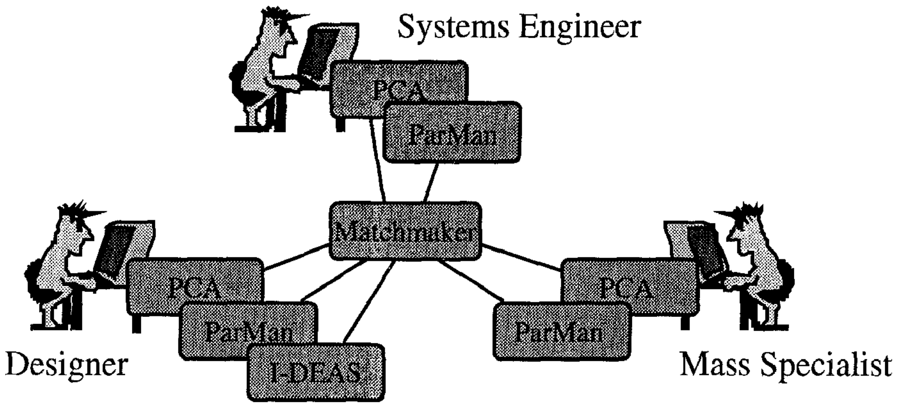

Figure 2. A collaborative engineering team and associated agents.

A few critical tasks in the work flow of this team illustrate the use and value of matchmaking. First, consider the use of matchmaking to propagate trouble reports. The systems engineer must be kept informed of major problems spanning the entire project. Often, this is achieved inefficiently via large meetings and hierarchical communication channels. However, matchmaking and associated assistant agents present an alternative. The systems engineer can use the PCA to request notification about any unresolved problems. This, in turn, depends on the SHADE matchmaker to locate such dynamic information. For example, a request by the systems engineer to be kept informed of "problems" would result in the following matchmaker subscription:

```
(subscribe :sender syseng :receiver mm :content
  (tell :language 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 (by virtue of the `not-match` condition). 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.

Now, assume that a designer in some other organization (or even at an entirely different subcontractor company) uncovers a key problem. Rather than waiting for a weekly meeting to present the problem to his boss, which then would be propagated through many levels of management before (undependably) reaching all interested parties, he posts an open problem via PCA. This results in the following message being sent to the matchmaker.

```
(tell :sender gimball :receiver mm :language 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")])
           ...])
    ...])])
```

The above message matches the System Engineer's earlier subscription, so the matchmaker forwards the message to him. Thus, by facilitating the dynamic connection of relevant information sources, a problem that might have gone unnoticed for many weeks was identified and propagated to concerned participants within minutes. This exchange is summarized in Figure 3.

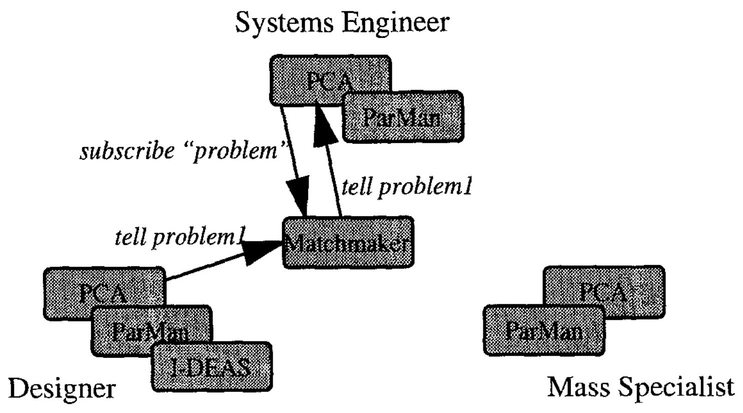

Figure 3. Using the matchmaker to propagate trouble reports

Now consider the use of the matchmaker to facilitate more detailed design. There are many attributes of the satellite design that impact multiple engineers. For example, the mass of spacecraft is tightly controlled since it impacts launch cost, structural requirements, dynamic stability, and many other factors. To tightly monitor and optimize this critical attribute, each engineer on the project uses ParMan to enter key constraints and to be kept informed of potential conflicts. ParMan, in turn, uses the matchmaker to locate other ParMan agents with constraints on related parameters.

To illustrate this process, assume that the gimbal designer performs a redesign of the gimbal, resulting in a modification of its mass. 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 :language kqml :content
    (stream-about :language kif :content
      (mass gimbal-1))))

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

Elsewhere, a mass specialist is responsible for allocating mass budgets, so she also defines a constraint over the gimbal mass, which results in similar advertisements and requests being posted by her ParMan agent. The matchmaker matches the advertisements and requests for the gimbal mass posted by each ParMan and routes the requests to the other ParMan agents. This allows each ParMan agent to locate all other sources of relevant constraints. In this case, once each ParMan agent queries the other for gimbal mass constraints, both discover that the new mass budget for the gimbal is insufficient for the new gimbal design. This is flagged to the respective users, and the design conflict is rapidly identified, preventing expensive revisions at a later date. This exchange is summarized in Figure 4.

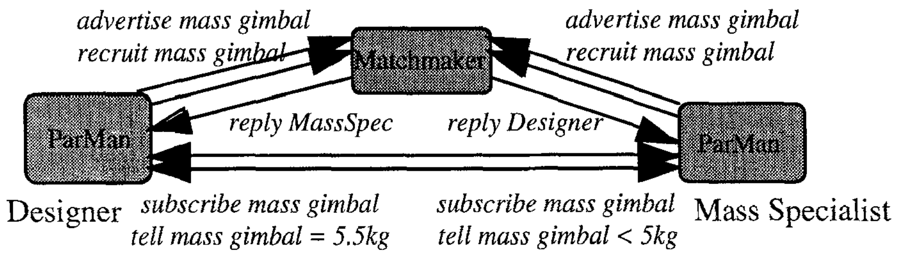

Figure 4. Matchmaking for electronic comerce in satellite image delivery.

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. Along these lines, the Integrated Weapons System Database (IWSDB) project is an effort to put on-line the huge volume of engineering data generated and used by the F-22 project. Much of this information is in the form of design documents, specifications, and catalogs; i.e., largely unstructured free text. Therefore, the IWSDB project is using the COINS matchmaker to provide engineers transparent access to a myriad of information sources.

The key benefit of the matchmaking approach is that it eliminates the need to identify a priori all potential information transfer paths, which is usually impossible 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. Each of these represents a potential information transfer path.

Whereas there are typically tools for each individual engineering domain (since these tend to be circumscribed tasks), there is less support for integrating these dynamic information sources and consumers. The typical approach is to rely on hierarchical management structures or pre-defined communication paths. Unfortunately, the former is too inefficient, while the latter is too inflexible. The matchmaking approach, which allows collaborators to locate each other directly and dynamically provides a much needed third alternative.

The benefits of matchmaking are particularly evident when the Parameter Manager is considered. Many engineers might be using ParMan 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 who 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. A tool like ParMan simply wouldn't work without the matchmaker.

This style of use is also underscored by the IWSDB application, in which engineers use the COINS matchmaker to locate and access documents relevant to their information needs. Even though elements of this functionality are provided by other information retrieval systems, such as WAIS, the matchmaking approach allows the user to be unconcerned with the specific data sources, and it provides automated notification as data sources are updated. Note also that matchmaking can be used to provide very task specific exchange of information, such as in the ParMan application, or very general, opportunistic information sharing, as in the IWSDB example.

4.2. *Electronic Commerce*

Another set of applications that have adopted the SHADE and COINS matchmakers are in the electronic commerce domain. Upon becoming available, the matchmaker quickly became 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. In this application, there are multiple satellite image providers (such as Landsat and SPOT), and there are many more value-added postprocessors that overlay information such as geographic, industrial, and political features, agricultural analyses, and utility routes. The problem is to provide a single interface (the Customer Service Center) that an information consumer can use to specify, locate, and preview imagery available from multiple, dynamic sources. Therefore, the SII prototype uses the SHADE matchmaker to route a customer's request to the appropriate databases. The set of agents is shown in Figure 5.

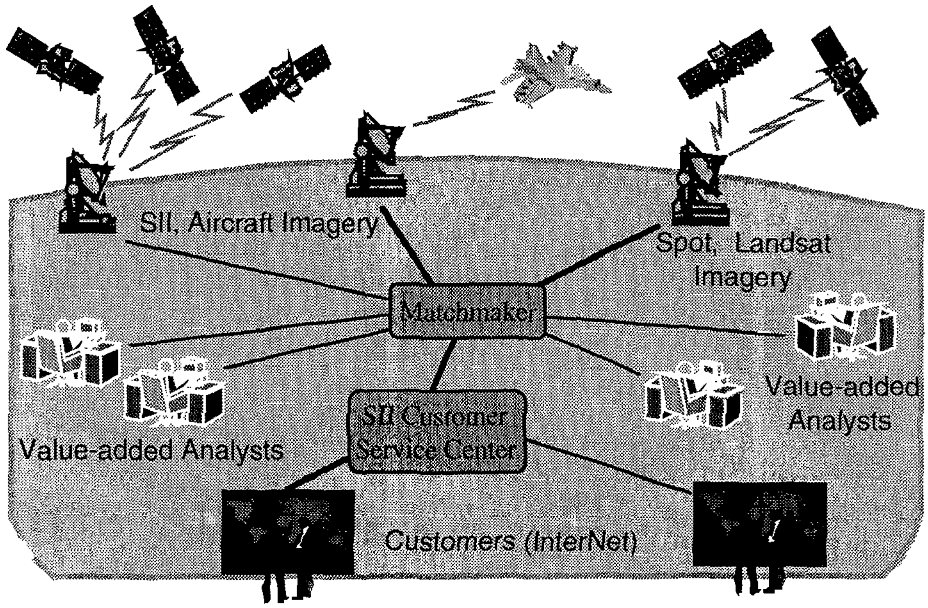

Figure 5. Matchmaking for electronic commerce in satellite image delivery.

A functional prototype, which has been fielded for customer use via the World Wide Web, has been built using the SHADE matchmaker (as well as other elements of the SHADE infrastructure). 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). An example of such a message is shown below.

```
(advertise :sender landsat-db :receiver mm :content
  (stream-about :language kif :content
    (and (image ?img ?footprint ?res ?date)
         (resolution ?img ?res)
         (< ?res 10)
         (date ?img ?date)
         (>= ?date 19860101)
         (<= ?date 19891231))))
```

The above message says that the Landsat database can answer queries about images with resolution of 10 m taken between 1986 and 1989.

The customer uses a World Wide Web based customer service center(CSC) to specify and request specific kinds of images. The CSC issues a request to the matchmaker describing the the desired attributes.

```
(broker-all :sender sii-csc :receiver mm :content
  (stream-about :language kif :content
    (and (image ?img ?footprint ?res ?date)
         (footprint ?img ?footprint)
         (overlap ?footprint #latlong#)
         (resolution ?img ?res)
         (<= ?res 10)
         (date ?img ?date)
         (>= ?date 19850101))))
```

In this message, the SII customer (via the CSC) is looking for all images with resolution of 10 m or better taken after 1984 that cover any portion of the specified region (the details of the footprint representation have been suppressed). The matchmaker compares the broker request to the advertisements, and sends it on to the appropriate databases. The databases then respond directly to the CSC.

```
(tell :sender landsat-db :receiver sii-csc :content
  (and (image O-224 #latlong# 10 19880914)
       (image O-973 #latlong# 10 19890601)))
```

The content of the reply is a set of KIF literals that include an object reference for the image (allowing the client to access this large data structure via a more efficient transport protocol), a structure specifying the image footprint, the resolution, and the date the image was taken. When no source database is appropriate, the matchmaker returns a failure reason. This allows the user to relax constraints in order to locate some imagery.

Matchmaking is also playing a central role from the outset in the AIMS project (Agile Infrastructure for Manufacturing Systems), an ARPA-funded effort focusing on tools for forming, contracting, and operating virtual corporations. At the core of AIMS is a cost, capability, and availability server (CCA), which member companies use to advertise their up-to-the-minute manufacturing capacity and price. By using such a service, other companies need not maintain a large manufacturing capability, since they can take advantage of the under-utilized capacity in the broader community. Due to the large number of manufacturing shops, and the dynamic nature of their workload and pricing, the matchmaker is useful in quickly forming virtual corporations.

The matchmaker is important in electronic commerce applications because there are multiple sources of data. In the Space Imaging domain, even though there are relatively few sources of raw images, there are a great many services that can post-process the images in order to overlay other useful information. A customer often may not know exactly who provides these services, so the matchmaker is used to provide that service location. This aspect of matchmaking is also a central theme of the AIMS project. If a client company must locate a machine shop with specific machining capability and availability, currently there is little recourse but to depend on fragmented catalogs or previous experience. Matchmaking provides a powerful index to a large number of suppliers.

Another important aspect of Matchmaking in these applications is its support for dissemination of dynamic information. In the space imagery delivery application, the satellite databases are constantly being updated as satellites circle the earth. Only an automated system can offer the up-to-the-minute location of data required. In fact, the dynamic nature of matchmaking would allow advertisement (and subsequent retraction) of satellite tasking opportunities, thereby allowing customers to request images taken in near real time. In the AIMS application, matchmaking similarly can describe the up-to-the-minute status of each supplier, which might be very important in an agile manufacturing setting. Shops can advertise their current availability as well as their capabilities, allowing the client company to locate manufacturing capacity immediately.

Finally, matchmaking can reduce the cost and bandwidth associated with transactions over wide-area networks. Image databases have complex schemata and overlapping data availability. Matchmaking allows these databases to be (approximately) characterized, and therefore can direct the user to appropriate data sources without excessive blind queries to each database. This reduces both query charges and message traffic.

5. Conclusions

The growth of information available via electronic networks presents both an unprecedented opportunity and a difficult challenge. Rather than relying on traditional information indexing techniques, which are static and consumer-driven, matchmaking allows both of the stake holders (i.e., information providers and consumers) to participate in 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 rapidly changing nature of electronic information, which is generated dynamically by huge numbers of potential information providers and consumers.

These benefits have been demonstrated by integrating the SHADE and COINS matchmakers with several prototype applications in concurrent engineering and electronic commerce. Projects that have adopted matchmaking include the SHADE testbed, Cosmos, Simulation-Based Design, the F-22 Integrated Weapons Systems Database, Space Imaging, Inc., and AIMS. In fact, the need for and utility of matchmaking is underscored by the rapid adoption of the SHADE and COINS prototype matchmakers by many of these projects.

Experiments have shown matchmaking to be most useful in two different ways: locating information sources or services that appear dynamically, and notification of information changes. Collaborative engineering tools like ParMan depend heavily on this capability to ensure that design constraints are propagated to those affected. Similarly, emerging electronic commerce systems like the image broker of SII depend on matchmaking to locate the best, most up-to-date images. A third benefit, that of allowing producers of information to actively seek potential consumers, has only been partially demonstrated, but this capability would certainly attract the attention of many industries.

Even though matchmaking has proven very useful in the above applications, several important shortcoming have been uncovered. Whereas queries can be expressed succinctly, expressing the content of a knowledge base (as in an advertisement) is a much harder problem. Current formal content languages are adequate for the simple examples shown above, but to go beyond advertising simple attributes quickly strains what can be represented. Additional research is required on ever more powerful content languages. The COINS matchmaker is, of course, not limited by representation, but the efficiency and efficacy of free-text matching becomes a limiting factor.

Our experiments revealed a number of other issues as well. There is inadequate support for transactions, error recovery and status checking. Keeping a network of agents consistent with their matchmaker is an important but largely unexplored issue. The semantics of many KQML message types are not well-defined. As applications grow in size and complexity, techniques to distribute the matchmaker load will be required. Finally, domain-specific knowledge and inference is needed in many cases, in order to compute matches based on subsumption and deductive reasoning. Work on utilizing domain-specific agents in support of matchmaking is a high priority (with the caveat that the matchmaker cannot become the reasoning engine to the world).

Yet, in spite of these open issues, matchmaking is a promising approach to supporting information access in very large, heterogeneous, and dynamic environments. Otherwise, the information explosion will simply change the problem from finding a needle in a haystack to finding a needle in a pile of pins.

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

Arens, Y., C.Y. Chee, C.N. Hsu, and C. Knoblock. Retrieving and integrating data from multiple information sources. *International Journal of Intelligent and Cooperative Information Systems*, 2(2), 1993.

Advanced Research Projects Agency. Reference architecture for the intelligent integration of information. Prepared by the Program on Intelligent Integration of Information, Richard Hull and Roger King, eds., 1995.

Chawathe, S., H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *Proceedings of IPSJ Conference, Tokyo, Japan*, 1994.

Davis, M., R. Evans, G. Davis, and G. Jones. Simulation based design for submarines. In *Proceedings of the Submarine Technology Symposium, JHU/APL*, 1993.

Finin, T., 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, M. An agent-based framework for software interoperability. In *Proceedings DARPA Software Technology Conference*, 1992.

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

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

Kuokka, D. and L. Harada. A communication infrastructure for concurrent engineering. *Journal of Artificial Intelligence in Engineering, Design, Analysis, and Manufacturing*, 1995.

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

Kuokka, D. 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.

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

McGuire, J., 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.

Mauldin, M. 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.

Patil, R.,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, G. *Automatic Text Processing—The Analysis, Transformation and Retrieval of Information by Computer*. Addison-Wesley, Reading, MA, 1989.

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

Wiederhold, G. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3), 1992.