MULTI-DIMENSIONAL MATCHMAKING FOR ELECTRONIC MARKETS

DANIEL J. VEIT and CHRISTOF WEINHARDT
Information Management and Systems, Universität Karlsruhe
(TH), Karlsruhe, Germany

JÖRG P. MÜLLER
Intelligent Autonomous Systems, SIEMENS AG CT,
Munich, Germany

Matchmaking is the process of mediating demand and supply in markets based on profile information. In electronic marketplaces and in negotiations, matchmaking plays a key role. The issue is to find the most appropriate agent for a task, the best bid in a multi-attribute auction, or the best present good for a request. In most real-world markets, multi-dimensional matchmaking is required, i.e., the ability to combine different dimensions and sub-dimensions of decision making to define an overall relevance. This task requires the interplay of multiple matchmaking algorithms. Another central aspect is the possibility to design relevance computation processes for multi-attribute objects easily. The realization of this issue makes multi-dimensional matchmaking processes to be easily integrated into industrial marketplace solutions. The work described in this paper aims on general multi-dimensional matchmaking objectives. These matchmaking objectives are implemented and deployed for industrial applications. The main contributions of this paper are: (i) the definition of multi-dimensional matchmaking in general; (ii) an implementation of configurable multi-dimensional matchmaking as a application dependent EJB^{TM} component, which is configurable using XML; (iii) the definition and implementation of different relevance (i.e., distance) functions for general usage and specific domains; (iv) the description of a process guiding application developers to design matchmaking applications (enterprise java beans); and (v) a report on experiences deploying the EJB matchmaker for the human resource area within a large-scale agent-based software.

INTRODUCTION

Electronic markets have gained importance in the past few years. This tendency is mainly driven by the rise of electronic commerce. The deployment

Address correspondence to Daniel Veit, Information Management and Systems, Universität Karlsruhe (TH), Englerstr. 14, 76131 Karlsruhe, Germany. E-mail: veit@iw.uni-karlsruhe.de

possibilities of electronic markets range from goods to contracts over all types of objects, which can be described in an electronic way. In next-generation electronic markets, autonomous agents are likely to play an important role.

An electronic market is based on the assumption that there are clear and well-defined measures which allow to qualify an item with respect to another item. In general, this is not the case. Most kinds of items traded on electronic markets today are homogenous items whose only negotiable factors are quantity and price. The huge majority of items, which are candidates to be traded on electronic marketplaces, are items which have negotiable properties that are not measurable in figures or a discrete structure, but have to be measured in non-numeric continuous values or even complex documents containing both figures and continuous describing values. One crucial task in electronic markets is to provide mechanisms which allow to find the best fitting counterparts for a negotiation. In general, this should be an ordered list of possible counterparts, which is descending in quality of the respective starting basis. In the case of a homogeneous item, e.g., a stock, a price, or quantity information, is easy to qualify with respect to the position. This is different when dealing with heterogeneous items, such as a human resource market does.

After the best fitting counterparts are found, negotiation starts. In a negotiation, the same problem rises again. If attributes are negotiated whose comparison is non-trivial, elaborated measures must be defined to compute a relevance. In the phase where counterparts are distinguished, this is a non-recurring task in which the relevance of each possible counterpart towards the own position is determined once, whereas in a negotiation every bid or iteration of the bidder must be judged with respect to the opposite position.

Consequently, functions to judge relevance between a request and an offer must be supported to meet the following desiderata:

- Take as input a starting point description format which is possibly complex (i.e., contains several dimensions and sub-dimensions for specification).
- Provide mechanisms (relevance functions) to compute the relevance of instances of this format towards the other position.
- Provide different (possibly domain specific defined) description formats for the requester and the provider side.
- Provide a mapping from the requester description format to the provider description format which explains how relevance is computed.
- Evaluate efficiently in order to support decisions which are similar to how a specialist would decide.

In this paper, we refer to a research project in which we focused on the topic of matchmaking as a coordination mechanism in the selection phase of a negotiation in an elelctronic market. In the next section, we focus on the

objectives for matchmaking in general. In the section after that, we introduce the GRAPPA matchmaking framework. "Matchmaking Implementation" gives an overview on the implementation of GRAPPA (Generic Request Architecture for Passive Provider Agents), which is a generic EJB-based matchmaker based on XML data structures. The next section briefly summarizes the application of GRAPPA in the Human Resource Domain. The project called HrNetAgent is a large-scale agent-based software prototype for matchmaking between vacant positions and applicants. In "Evaluation" we provide an evaluation of the matchmaking approach made in the HrNet-Agent Project. In the last section, we conclude and describe issues of further research and application.

MATCHMAKING OBJECTIVES

Matchmaking is not only a key task in multi-agent systems, it is also a crucial function in marketplaces and electronic negotiations. The provider who will enable the most effective matches between demand and supply will gain a competitive advantage and increase the acceptance and popularity of their marketplaces.

Definition

We understand matchmaking as a function which accepts as input a set of offers (candidate profiles) and a request (centroid profile), and provides as output a ranked list of the k best offers with respect to the request. Each element of the list provides an overall relevance of the offer towards the request. This relevance is computed from the distances obtained in the sub-dimensions of the profiles. In the GRAPPA section, we will explain this process in more detail. Each centroid profile is wrapped by an agent. Candidates can provide their profiles either by defining a single agent, which carries its profile, or by selecting an agent which wraps a larger amount of candidate profiles stored in a database. Thus, matchmaking can be regarded as a *k-nearest neighbors problem*: In an n-dimensional vector space, the k nearest neighbors to a given profile (represented as a point in that vector space) needs to be computed. This problem is well understood in theory and solutions are known, e.g., from Information Retrieval (Salton 1989). However, the requirements stated in the previous section, turn the development of generic solutions to this problem into a challenge.

Related Work

In this section, we will present some work done by different groups, which concerns matchmaking among profiles which are carried by autonomous

agents. These profiles are mostly refered to as *Agent Service Descriptions*. Kuokka and Harada (1996) considered matchmaking in the context of emerging information integration technologies, where potential providers and requesters send messages describing their capabilities and needs of information (or goods). They presented two matchmakers: COINS (COmmon INterest Seeker), which is based on free text matchmaking using a distance measure from information retrieval (Salton 1989), and SHADE (SHared DEpendency Engineering), which uses a subset of KIF (Genesereth and Fikes 1992) and a structured logic text representation called MAX (Kuokka 1990). While COINS focused on at e-commerce, SHADE aimed at the engineering domain.

Complementing the theoretical work in Decker, Sycara, and Williamson (1997), Sycara and coworkers addressed the matchmaking problem in practice. They developed and implemented the LARKS matchmaker (LAnguage for Advertisement and Request for Knowledge Sharing) described in Sycara and Klusch (1998). In LARKS, the matchmaking process runs through three major steps: (1) context matching, (2) syntactical matching, and (3) semantic matching. Step 2 is divided into a comparison of profiles, a similarity matching, and a signature matching. Compared to previous approaches, LARKS provides higher expressiveness for service descriptions. Like those, however, LARKS has a static scheme for service descriptions, which restricts its application to agents that comply with this fixed description format.

In the context of electronic auctions, Weinstein and Birmingham (1997) introduce a service classification agent which has meta-knowledge and access to nested ontologies. This agent dynamically generates unique agent and auction descriptions, which classify an agent's services and auction subjects, respectively. A requester obtains from it the name of the best auction to its needs. The ontology issue is also addressed by Maedche and coworkers in (Maedche et al. 2002).

In IMPACT (Subrahmanian et al. 2000), so called Yellow Pages Servers play the role of matchmaker agents. Offers and requests are described in a simple data structure which represents a service by a verb and one or two nouns (e.g., *sell:car, create:plan(flight)*). The matchmaking process computes the similarity of descriptions from the shortest paths in directed acyclic graphs that are built over the sets of verbs and nouns, respectively, where edges have weights reflecting their distance.

Central aspects of market designs and negotiation protocols are introduced by Ströbel in (Ströbel 2000, Ströbel 2001). The author focuses on roles in matching and negotiation situations in electronic markets and shows important aspects in today's electronic negotiation research in his SILK-ROAD software prototype, which has been developed at the IBM Research Labs in Rüschlikon, Zurich, Swizerland.

The GRAPPA matchmaking approach integrates negotiation role definitions and profiles and differs from existing matchmaking approaches in various respects:

- The definition of the demand and supply profiles flexibly adapted to enable a wide range of applications.
- Our model does not enforce a specific matchmaking method. Instead, arbitrary (possibly nested) description schemes can be defined from basic types using various forms of aggregation, such as lists, sets, or records, and linked with suitable distance functions in a flexible way.
- Existing multi-stage matchmaking approaches measure similarity in several steps subsequently and classify the matching object after applying different matchmaking methods in a sequence. In contrast, our approach clusters the attributes of demand and supply profiles into clusters and computes local subdistances for these clusters ''in parallel,'' before combining them to a single global distance.
- GRAPPA uses XML to describe supply and demand profiles schemas, and thus can be easily applied to a range of existing and future data repositories. Also, future extensions of GRAPPA may take advantage of tools for XML.
- GRAPPA provides an open framework to incorporate new matchmaking algorithms and to reuse them within industrial matchmaking solutions.

THE GRAPPA MATCHMAKING FRAMEWORK

Figure 1 illustrates the structure of the GRAPPA matchmaking framework. It consists of three major parts. Its core is the *matchmaking engine* described in the following sub-section. It is complemented by the *matchmaking library* and the *matchmaking toolkit*.

GRAPPA Matchmaking Engine

The matchmaking engine accepts a set of supply profiles (candidate instances) and a demand profile as input. The supply profiles, which have to be provided as instances of the matchmakers candidate class, are either stored in the matchmakers service repository or retrieved from different data sources. In the case of the matchmaker, it does not keep a service repository. The request, which has to be provided as an instance of the matchmakers centroid class, is matched against each of the candidate instances.

The candidate structure, as well as the centroid structure, are multi-dimensional. They consist of complex types constructed from a domain
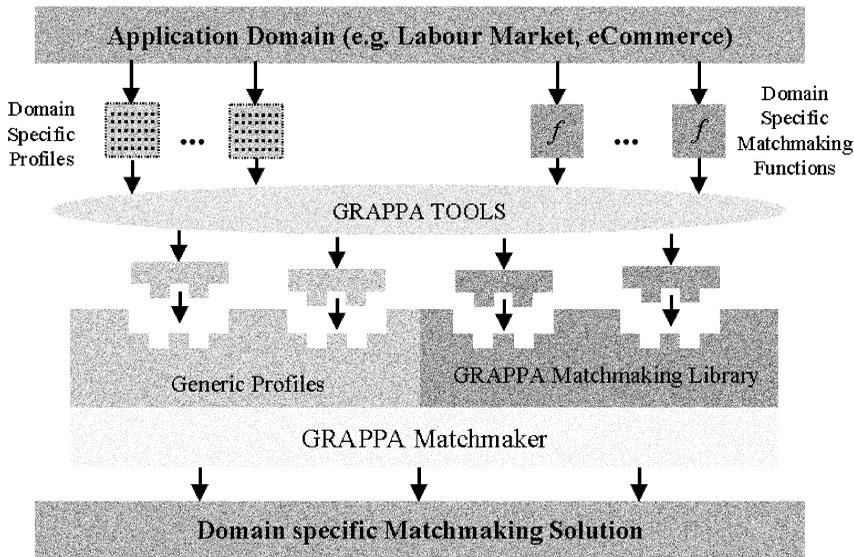
FIGURE 1. GRAPPA matchmaking framework.

specific set of basic types under application of four complex type constructors: list, array, record, and set.

The overall distance, a real value between 0 and 1, is obtained by recursively computing the distance values for different profile sub-types and propagating them upwards to compute the values for their parents (see the next sub-section as well as Eiter, Veit, Müller, and Schneider [2001] for details). For the basic types (the atomic attributes of the centroid and the candidate), the specific distance function for the particular type is applied and the result is propagated upwards. Then, at the next higher level, all basic distances between the atomic types in this level are merged to one distance value for this complex type under application of aggregate functions. For in depth discussion of the distance function issue see Eiter, Veit, Müller, and Schneider [2001].

The result of the recursive computation of distance values is:

(i) An overall distance (real value between 0 and 1), which reflects the quality of the considered candidate instance for the current centroid instance.

(ii) A structure (in XML), which consists of the individual distance results in each layer.

The best k candidates (with respect to the current centroid) are returned as the result of the match. This list is ranked using the value from (i).

The agent (or the agent's principal) can then recur into the XML structure described in (ii) to obtain an explanation on how the particular overall result arose (e.g., which aspects of the match contributed to a good or bad overall result).

GRAPPA Matchmaking Library

The GRAPPA Matchmaking Library hosts an extensible collection of pre-defined profile schemas and (general-purpose or domain-specific) distance functions. The profiles schemas can be used as a basis for application-specific profiles; the distance functions provide uniform interfaces that allow us to flexibly combine them to develop specific matchmaking solutions.

It is essential for a matchmaking system to provide powerful distance functions. Currently, we provide distance functions for FreeText, Weighted Keyword, Interval, TimeInterval, DateInterval, Boolean, and Number basic values (i.e., instances of basic types). All distance functions have the property to take two basic values as input and to provide a real number between 0 and 1 as output (distance). Additionally, domain specific distance functions can be integrated as we shall describe in the section on the HRNET Agent project. On top of these basic functions, we define aggregate distance functions. Currently, WeightedAverage, Average, Minimum, and Maximum are supported as pre-defined aggregate functions. As for basic functions, it is possible to define domain specific aggregate functions and integrate them into a domain specific matchmaker.

As an example for a basic distance function in GRAPPA, we show the default distance function for free text. This distance function is based on a cosine similarity measure developed in information retrieval (Salton 1989). Any free text document T can be associated with a document vector dv by removing stopwords and performing stemming on the remaining words. For each word, its frequency in T is assigned (called the *term frequency* (*tf*)). Given a collection of N documents, the *document frequency* (*df*) of a word stem is the number of documents in which it occurs. The *term-frequency-inverse-document-frequency factor* (*tf-idf factor*)

$$tf \cdot \log(N/df)$$

has proven to be a useful weight for a word stem. The documents are represented in a document space by their document vector consisting of the *tf-idf* factors.

The similarity of two documents T_1 and T_2 is computed as the cosine between the corresponding document vectors in the document space (Salton 1989; Vorhees 1994):

$$sim(T_1, T_2) := \frac{\sum_{j=1}^{k} w_{1,j} \cdot w_{2,j}}{\sqrt{\sum_{j=1}^{k} (w_{1,j})^2 \sum_{j=1}^{k} (w_{2,j})^2}},$$

where

$$w_{i,j} = tf_{i,j} \cdot \log \frac{N}{df_j}$$

for $i=1$ or $i=2$ is the weight for term j in document i, and k is the dimension of the document space.

GRAPPA Toolkit

The GRAPPA Toolkit provides a set of tools which enable the development of a multi-dimensional matchmaker for specific applications mainly through configuration without much coding work. To guide the marketplace designer we have defined a five-step process to obtain a domain specific matchmaking solution:

(1) Define the centroid and candidate schemas (basic entries) in XML.
(2) Define the clusters of attributes in XML (pseudo-orthogonalization); clustering can be recursive.
(3) Associate the clusters of the centroid with clusters of the candidate by applying appropriate distance function.
(4) Combine the results of the distance functions to an overall distance value (e.g., weighted sum).
(5) Apply feedback regarding the quality of the matches, e.g., by adaptively changing weights or matching functions.

These steps are explained in more detail in the following section.

MATCHMAKING IMPLEMENTATION

In this section, we describe key issues of implementation of the matchmaking framework. We focus on the requirements described in the Introduction; in particular, the framework has been designed to be extensible and to be integrated easily into commercial e-business platforms.

Basic Entities and Processes

In this section, we describe the basic computational concepts used in our implementation. The centroid profile encapsulates the structure of the ori-

ginating request. This structure is defined by means of an XML document type definition (DTD). Requests on the domain-specific matchmaker must conform to this DTD.

Candidate profiles are the data instances on which the matchmaking will be processed (e.g., records in an applicants database). Like the centroid profile, this structure must be defined in an XML document type definition. To compute a match between centroid and candidate profile based on various distance functions, a 1:1-mapping between the elements of the centroid and the candidate is required. An excerpt from an example for a candidate and centroid structure is found in Figure 2. Especially if the centroid and the candidate provide a different number of top-level attributes, a orthogonalization process is required, which will be described in the following. In this process, attributes that together describe a separate semantic aspect relevant for matchmaking are determined and grouped into a cluster. An attribute may be contained in more than one cluster. If this is this case, they are duplicated to preserve orthogonality.

As described in the section "GRAPPA Matchmaking Library," the matchmaking library provides a number of built-in distance functions that can be used to create domain-specific matchmakers. All distance functions will need to implement an interface `DistanceFunction`. This way, domain-specific distance functions implementing custom semantics required for specific matchmaking solutions can be provided.

Aggregate functions are used by the matchmaker to compute a multi-attribute result from the values returned by the basic distance functions. The matchmaking library contains the aggregate function WeightedSum to compute a weighted sum of the basic distances as an overall result. Different aggregate functions implementing thresholds, hard constraints, or averages can be included similarly as additional distance functions using the interface `AggregateFunction`.

Centroid Job-Profile	Candidate Person
Max_age	Age
Max_wage	Expected_wage
Job_description	Soft_skills Hard_skills

FIGURE 2. Example of candidate and centroid profile.

862

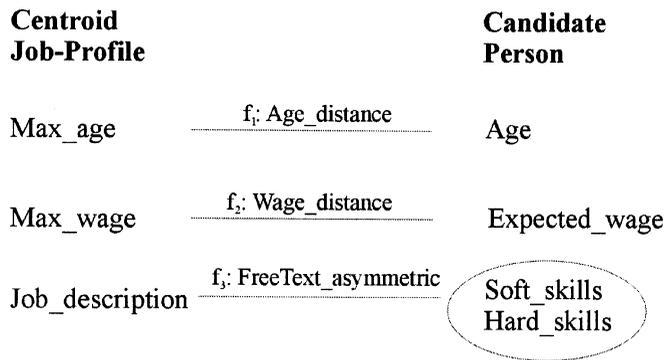| Centroid Job-Profile		Candidate Person
Max_age	f_1: Age_distance	Age
Max_wage	f_2: Wage_distance	Expected_wage
Job_description	f_3: FreeText_asymmetric	Soft_skills Hard_skills

FIGURE 3. Example after orthogonalization and association of distance functions.

In Figure 3 three clusters were formed and three distance functions, f_1, f_2, and f_3, were associated to the clusters.

Configuration

In order to customize the generic matchmaker for a domain, we provide a configuration process: First, the structure of centroid (request to the matchmaker) and candidate (data instances) profiles have to be defined. These structures must be provided as XML document type definitions (DTD). During the matchmaking process, instances of the centroid and candidate profiles must comply with these DTDs. Then, the main configuration file (XML document) has to be created. In this document, the clustering of attributes, and association of distance functions with each pair of centroid/candidate cluster pair are specified. Additionally, this document contains the specification of the aggregate parameters whose meaning depends on the corresponding aggregate function (in the case of the weighted sum, the aggregate parameter would represent simply the weight).

Furthermore, for each cluster pair, a constraint type (soft or hard) must be declared. This type determines the impact of an absolute inequality of a cluster-pair on the overall result. So a distance of 0 of one cluster comparison with the comparison type "strong" would lead to an overall result of 0 independent of any other comparison result. Each cluster and its associated distance function is considered as one dimension in the configuration. The configuration entries for each dimension are grouped as 5-tuples consisting of: (i) cluster from centroid; (ii) cluster from candidate; (iii) distance function; (iv) aggregate parameter; and (v) constraint type. As mentioned above, the computation of a distance value between two clusters can also be made by dividing each cluster into its subitems; compute the distance of these subitems directly and use an aggregate function to obtain an overall result for this cluster.

In this case, the corresponding dimension entry in the configuration file would have a number of sub-dimension entries and, instead of a distance function, an aggregate function for the overall result has to be specified. The configuration file with a structure as described above is written in XML using a special configuration document type definition `configuration.dtd`, provided as a part of the matchmaking library.

Dimension entries in configuration file for the example above:

```
<Configuration>
   <Dimension>
     <LeftName>job_profile</LeftName>
     <RightName>Person </RightName>
     <Dimension>
     <Left Name>max_age</LeftName>
     <RightName>Age</RightName>
     <DistanceFunction>Age_Dist</DistanceFunction>
     <AggregateParameter>50<AggregateParameter>
     <Type>weak</Type>
     </Dimension>
     <Dimension>
     <LeftName>max_wage</LeftName>
     <RightName>Expected_wage</RightName>
     <DistanceFunction>Wage_Dist</DistanceFunction>
     <AggregateParameter>50</AggregateParameter>
     <Type>weak</Type>
     </Dimension>
     <AggregateFunction>WeightedSum</AggregateFunction>
     <AggregateParameter>30</AggregateParameter>
     <Type>strong</Type>
   </Dimension>
   <Dimension>
     <LeftName>job_description</LeftName>
     <RightName>soft_skills</RightName>
     <RightName>hard_skills</RightName>
     <DistanceFunction>FreeText_Asymmetric
     </DistanceFunction>
     <AggregateParameter>70</AggregateParameter>
     <Type>strong</Type>
   </Dimension>
</Configuration>
```

Using this configuration information, a domain-specific instance of the matchmaker can be created automatically.

Matchmaking Java-Library

The matchmaking library consists of a number of classes covering configuration aspects, such as dimension entries and the main configuration file, and a class for the generic matchmaker, which operates with the configuration and carries out the matchmaking process between instances of corresponding classes for centroid and candidates. The matchmaking result is covered by a class which provides detailed information about the matchmaking process, such as information about each cluster-cluster comparison, reasons for "disqualifying" a candidate, output to XML, etc. The following paragraphs provide a short description of the most important classes included in the matchmaking library.

The dimension-class encapsulates the dimension entry (a 5-tuple) in the configuration file with declaration of the clusters in centroid and candidate, distance function (resp. sub-dimensions an aggregate function), the aggregate parameter, and type. Since a dimension can have multiple sub-dimensions, this class is defined recursively.

The config-class is the main configuration class, encapsulating the configuration file and administering the dimension entries. During construction, an object of this class loads all required distance and aggregate functions, creates dimension entries, and builds the necessary internal structure for the matchmaker. Additionally, this class provides the application with two factory objects for loading centroid and candidate data. Instances of the matching class must be initialized with a configuration object and it will then perform the domain-specific matchmaking on centroid and candidates. Additionally, this class can have a number of candidate providers from which candidates can be drawn.

The candidate provider interface automates the access of candidate data for the matchmaker. Implementing this class, the user has the ability to preview and preselect candidates from various data sources before matchmaking so that unnecessary comparisons can be avoided.

Deployment as Enterprise Java Beans Component

In addition to the classes of the basic matchmaker described above, the library provides a set of classes for the deployment of a domain-specific matchmaker in an application server to provide matchmaking functionality in the context of a commercial e-business platform and to make it available from the Web or from other clients. These classes are two wrapper classes for the MMConfig and the matching class with some extra functionality suitable for EJB components.

Config, ConfigEJB, ConfigHome

These are classes for the ConfigBean as a wrapper for the MMConfig class. This Enterprise Java Bean is realized as an entity bean to provide a persistent storage for domain-specific configurations of the matchmaker.

Matchmaker, MatchmakerEJB, MatchmakerHome

These classes represent the MatchmakerBean session bean with the matchmaking functionality. In combination with an instance of the ConfigBean, this class performs the matchmaking in the application server.

APPLICATION: HRNETAGENT

Due to the open, flexible architecture of the GRAPPA framework, it can be applied to a wide range of matchmaking problems in all sorts of (agent- or human-operated) electronic marketplaces. In this section, we provide a brief description of one industrial project in which GRAPPA has been applied successfully. The Siemens "Human Resource Network Agent" (HRNet-Agent) project.

The HRNetAgent is an application of GRAPPA for matching corporate job profiles with profiles of job applicants (i.e., persons looking for employment) stored in various databases. The current version of HRNet-Agent is a prototype system that has been developed for the German Labor Exchange Office (BfA), and demonstrates the feasibility of a partially automated approach to employment relaying. Based on its success, a full-fledged system is planned for the near future. The potential return on investment is huge: Reducing the relaying time of unemployed persons (currently, there are about four million people in Germany without employment) just by one day on average will save the German government more than a hundred million dollars a year.

Figure 4 shows the architecture of the HRNetAgent system. A company specifies its job profiles to a designated GUI-Agent, which takes the role of a requester agent in the system. The GUI-Agent queries the matchmaker by sending to HRNetAgent the description of the open position, which should be filled. The scheme for specifying the open positions is the centroid. The backend of HRNetAgent consists of a collection of data sources wrapped by information agents, and by a search controller that coordinates a number of search agents. For example, one data source is the central database of the German Federal Labor Exchange Office[1] in which the names of all currently unemployed persons in Germany are stored. Other examples may be corporate skills databases; other databases can also be easily integrated. Note that the database wrapper agents play the role of virtual provider agents in our architecture.
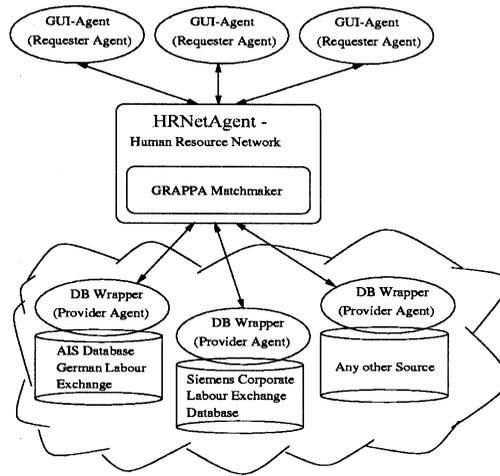
FIGURE 4. HRNetAgent system overview.

The HrNetAgent human resource market is designed to find appropriate applicants from heterogeneous sources (e.g., the employee data of different companies). The results are displayed for the user in a homogenous way. These properties fulfill main points of the desiderata for future job markets formulated by Maier et al. (Maier, Kronewald, and Mertens 2000).

Wrapper agents perform the task of query translation, connection handling, and result translation. They return a pre-selection of profiles to the matchmaker based on conditions extracted from the centroid profile. In HRNetAgent, the centroid and candidate schemes are converted to XML-DTDs, which are considered as the document classes of these types. Matchmaking thus is done on a pre-selection of candidates. The most successful candidates for a job profile are stored in the local service repository for fast access by the application. In addition, HRNetAgent offers an automated notification service via SMS, fax, or e-mail.

EVALUATION

We have been investigating the GRAPPA matchmaker quality in a field study. The GRAPPA matchmaking in the HrNetAgent defines a ranking of the candidates for a given job position (centroid). To prove the quality of the matches computed by GRAPPA, a field study with eleven human resource (HR) experts from three companies was performed: Siemens AG CT, Munich; Lufthansa AG, Frankfurt; and the University Hospital of the Justus-Liebig-University Giessen. We designed a test which allows to identify the HR experts' opinions on a set of fifty candidates towards a given job position. In the next step, we computed the correlation between the

HR experts' opinions and the GRAPPA matchmaker ranking on the same candidates. Finally, to show the GRAPPA matchmakers ranking quality in comparison to other state-of-the-art automated ranking methods, we compared the GRAPPA results with the results the Microsoft Index Server computed on the candidates.

For comparison of the results, we used precision and recall measures from information retrieval (IR), which are also used in (Salton 1989), and adapted those measures for the automated matchmaking evaluation domain. The following key results were achieved.

GRAPPA reached high recall and reasonable precision values. The candidates who were selected by the HR experts to be the best fitting applicants for the position also reached high scores by the matchmaker. In the set of applicants who did not fit so well (the worst third of applicants), the correlation between the matchmakers' results and the HR experts' opinions was lower. In all cases, the GRAPPA matchmaker outperformed the Microsoft Index Server by far.

Discussing the results of the empirical evaluation, the GRAPPA matchmaker reached high recall and moderate precision values. This combination enables an application of the HrNetAgent as a pre-selection tool for HR experts to reduce large sets of applicants to the best fitting subset before manually matching. The empirical evaluation of the GRAPPA matchmaker is shown in detail in (Veit, Müller, and Weinhardt 2002).

CONCLUSIONS

In this paper, we have described a generic approach to matchmaking in agent- or human-mediated electronic marketplaces. The focus of this work is on achieving the flexibility and openness required to build a matchmaking framework that can be easily applied to different vertical marketplaces and that can be integrated into a broad range of industrial marketplace platforms. Currently, the matchmaking framework described earlier in this paper will be developed to product stage. It will be used both to enable matchmaking in agent-based marketplace applications within Siemens, and for "non-agent enabled" electronic marketplaces. Our hope is that this will be a starting point allowing us to push the deployment of agents into mainstream e-business systems.

A current restriction of the system is that it only provides 1:N matchmaking, i.e., the matchmaker will always consider one demand profile and multiple supply profiles and vice versa. It cannot deal with matching problems as they occur in continuous double auctions, where the best matches combining multiple demand and supply profiles need to be identified (see, e.g., [Sandholm 2000]). Future work will include the development of corresponding matching functions for N:M matchmaking.

Also, the system currently provides only very basic feedback mechanisms that can be used to adapt the matchmaking configuration. We believe that learning capability will be required to achieve robust and good matchmaking behavior, and we are planning to incorporate feedback rules into the system in the future.

One future research objective will be to apply multi-dimensional matchmaking for relevance computation in electronic negotiations. Bichler (Bichler 2000) states that multi-attribute auctions use a mechanism which determines a winning bid among different bids. This mechanism can be seen as an application of multi-dimensional matchmaking. In our future work, we intend to apply the GRAPPA matchmaking system to multi-attribute auctions in practice. Finally, more elaborate methods are required to test the performance of complex matchmaking solutions as the one described in this paper.

NOTE

1. http://www.arbeitsamt.de

REFERENCES

Bichler, M. 2000. An experimental analysis of multi-attribute auctions. *Decision Support Systems* 29: 249–268.

Decker, K., K. Sycara, and M. Williamson. 1997. Middle-agents for the Internet. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence* (IJCAI-97), 578–583.

Eiter, T., and H. Mannila. 1997. Distance measures for point sets and their computation. *Acta Informatica* 34:109–133.

Eiter, T., D. Veit, J. Müller, and M. Schneider. 2001. Matchmaking for structured objects. In *Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery* (DaWaK), Munich, Germany, 186–194.

Genesereth, M. R., and R. E. Fikes. 1992. Knowledge Interchange Format, Version 3.0 Reference Manual. Technical Report Logic-92-1, Computer Science Department, Stanford University. http://www-ksl.stanford.edu/knowledge-sharing/papers/kif.ps (06/11/2002).

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

Kuokka, D., and L. Harada. 1996. Integration information via matchmaking. *Journal of Intelligent Information Systems* 6(2/3):261–279.

Maedche, A., S. Staab, N. Stojanovic, R. Studer, and Y. Sure. 2002. *Spinning the Semantic Web – Bringing the World Wide Web to its Full Potential*, eds. D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster. Cambridge, MA: The MIT Press (in press).

Maier, M., K. Kronewald, and P. Mertens. 2000. Vernetzte jobbörsen und unternehmensnetzwerke – eine vision, *Wirtschaftsinformatik* 42:124–131.

Müller, J. P. 1997. The design of intelligent agents. *Lecture Notes of Articial Intelligence*, Vol. 1077. Berlin: Springer.

Müller, J. P., and M. Pischel. 1999. Doing business in the information marketplace: a case study. In *Proceedings of the 3rd Intl. Conference on Autonomous Agents* (Agents-1999), ACM Press, 139–146.

Nodine, M., W. Bohrer, and A. H. H. Ngu. 1999. Semantic brokering over dynamic heterogenous data sources in InfoSleuth. In *Proceedings of the Fifteenth International Conference on Data Engeneering*, 358–365.

Poggi, A., and G. Rimassa. 2000. An agent model and platform for realizing efficient and reusable agent software. In *Proceedings of the 4th Intl. Conference on Autonomous Agents* (Agents-2000), ACM Press, 74–75.

Salton, G. 1989. *Automatic Text Processing*. Barcelona: Addison-Wesley.

Sandholm, T. 2000. eMediator: a next generation electronic commerce server. In *Proceedings of the 4th Intl. Conference on Autonomous Agents* (Agents-2000), ACM Press, 341–348.

Ströbel, M. 2000. On auctions as the negotiation paradigm of electronic markets. *Journal of EM – Electronic Markets* 10(1).

Ströbel, M. 2001. Design of roles and protocols for electronic negotiations. *Electronic Commerce Research Journal*, Special Issue on Market Design 1(3):335–353.

Subrahmanian, V. S., P. Bonatti, J. Dix, T. Eiter, S. Kraus, F. Ozcan, and R. Ross. 2000. *Heterogenous Agent Systems*. Cambridge, MA: The MIT Press.

Sycara, K., J. Lu, and M. Klusch. 1998. Interoperability among heterogenous Software Agents on the internet. Technical Report CMU-RI-TR-98-22, The Robotics Institute, Carnegie Mellon University, Pittsburgh.

Veit, D., J. P. Müller, M. Schneider, and B. Fiehn. 2001. Matchmaking for autonomous agents in electronic marketplaces. In *Proceedings of the 5th Intl. Conference on Autonomous Agents* (Agents-2001), Montreal, 65–66.

Veit, D., J. P. Müller, and Ch. Weinhardt. 2002. An empirical evaluation of multidimensional matchmaking. In *Proceedings of the third International Workshop on Agent Mediated Elecronic Commerce* (AMEC IV), First International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Bologna, Italy, in press.

Vorhees, E. 1994. Query expansion using lexical-semantic relations. In *Proceedings of the ACM SIGIR'94 Conference*, 61–69.

Weinstein, P. and W. Birmingham. 1997. Service classification in a proto-organic society of agents. In *Proceedings of the IJCAI-97*, Workshop on Artificial Intelligence in Digital Libraries. http://www-personal.umich.edu/∼peterw/paper.protoorganic.ps

Wiederhold, G. 1993. Intelligent Integration of information. In *Proceedings of ACM SIGMOD Conference on Management of Data*, Washington, DC, 434–437.