A Semantic Web Approach to Service Description for Matchmaking of Services

David Trastour, Claudio Bartolini and Javier Gonzalez-Castillo
david_trastour@hp.com, claudio_bartolini@hp.com, javgon@hplb.hpl.hp.com
HP Labs, Filton Road, Bristol BS34 8QZ, UK

Abstract. Matchmaking is an important aspect of e-commerce interactions. Advanced matchmaking services require rich and flexible metadata that are not supported by currently available industry standard frameworks for e-commerce such as UDDI and ebXML. The semantic web initiative at W3C is gaining momentum and generating technologies and tools that might help bridge the gap between the current standard solutions and the requirement for advanced matchmaking services.
In this paper we examine the problem of matchmaking, highlighting the features that a matchmaking service should exhibit and deriving requirements on metadata for description of services from a matchmaking point of view. We then assess a couple of standard frameworks for e-commerce against these requirements. Finally, we report on our experience of developing a semantic web based matchmaking prototype. In particular, we present our views on usefulness, adequacy, maturity and tool support of semantic web related technologies such as RDF and DAML.

Keywords. Semantic Web; E-Commerce; Matchmaking Services; Automated Negotiation; Electronic Marketplaces; Ontology.

1. Introduction

E-commerce is done faster, on a global scale, and with fewer human interventions than traditional trade. Electronic interactions are increasing the efficiency of purchasing, and are allowing increased reach across a global market. With the proliferation of offers comes the problem of finding and selecting potential counterparts for service provision/consumption to engage in negotiation with them.

In the business-to-business (B2B) e-commerce arena, the last couple of years have seen a continuous flourishing of E-marketplaces. E-marketplaces aggregate buyers and sellers in a single virtual location to create dynamic trading exchanges. In doing so, they somehow simplify the problem of discovering potential counterparts for business. Still businesses come together based on the services they require or provide, and matchmaking - i.e. the process of matching service offers with service requests - might be a difficult task depending on the degree of flexibility and expressiveness of the service descriptions.

By analysing the features that we would like an advanced matchmaking service to have, we derived requirements for a language for service descriptions in the context of matchmaking. These requirements are: high degree of flexibility and expressiveness; ability to express semi-structured data; support for type and subsumption; ability to express constraints over ranges of possible values as well as definite values of a specification.

We have studied industry standard frameworks for e-commerce such as UDDI and ebXML, to see whether the solutions they propose meet our requirements. We found that their main shortcoming is that they do not allow much flexibility and expressiveness in the service descriptions.

After looking at the industry standards, to develop our prototype of an advanced matchmaker, we have taken an approach to service matchmaking based on semantic web technologies. Our approach aims at providing a richer service description, while making use of existing ontologies and the ways of combining and extending them. In this paper, we report about our experience in applying semantic web related technologies to the service description problem. In particular, we investigate the use of RDF as a basis for a service description language[1], and we discuss how well it meets our requirements. In addition, we discuss our experience with some semantic web tools.

The remainder of the paper is structured as follows. In section 2 we describe the features of the matchmaking service; in section 3 we derive the requirements for a language to express descriptions of advertisements and queries to be used in matchmaking; in section 4 we assess current industry standards e-commerce frameworks, such as UDDI and ebXML against the requirements; in section 5 we describe our experience in applying semantic web technologies to the development of a matchmaking service prototype; in section 6 we present related work and in section 7 our future work intentions, to conclude in section 8.

2. Matchmaking

Matchmaking is the process by which parties that are interested in having exchange of economic value are put in contact with potential counterparts.

The matchmaking process is carried out by matching together features that are required by one party and provided by another. In the traditional way of doing business, this process is executed either through brokers, by actively seeking counterparts in directory services such as the yellow pages, or by looking at advertisements on media.

With the possibilities opened by e-commerce, the number of potential counterparts is multiplied. Therefore, who is seeking for a business counterpart is faced with the problem of filtering out relevant from irrelevant information.

2.1. Advertising, querying and browsing

The minimal functionalities that a matchmaking service provides are the features of *advertising* a service, and *browsing* or *querying* a repository of advertised services.

[1] The descriptions that we consider in this work do not involve behavioural aspects of a service, as these are not necessarily required for matchmaking. Therefore, “service description language” here and in the rest of the paper is shorthand for “language to express service parameters”.

2.1.1. Advertising

A party describes the features of the service or product that it is providing or requesting. Such description is published in an *advertisement* in the matchmaking service.

An advertisement defines a space of possible realizations of a service. The level of detail used to describe the service is completely up to the advertiser. It is even possible to advertise more specific and more general descriptions for analogous services at the same time.

The advertiser will add contact details to the advertisement to make it possible for a potential counterpart to follow up. Along with the service features and contact information, corollary information might be expressed on negotiable terms and condition as well as the rules of engagement for the negotiation process. Moreover, the advertiser can also specify visibility rules for the advertisement. The matchmaking services will take them into account when delivering information to interested parties that are browsing or querying the repository.

2.1.2. Querying

To find out a relevant advertisement among the currently available ones, a party can submit a *query*. The query expresses constraints over aspects of advertised services that the submitter is interested in. The query expression will be use to filter out the existing advertisements that are not important to the submitter.

2.1.3. Browsing

The matchmaker offers the possibility of *browsing* the currently available advertisements. The matchmaker maintains an advertisement repository, where posted advertisements are stored. In finding out about advertised services, browsing parties can make use of this information to tune the adverts that they will submit in turn, so as to maximize the likelihood of matching.

To facilitate browsing, the matchmaking services may provide a classification of adverts and of the terms used in them. Many current catalogue-based marketplaces organise products in predefined hierarchical categories, making this classification often too rigid.

2.2. Information in Advertisements and Queries

Functional aspects apart, descriptions of advertisements and queries have much in common. Both usually contain constraint expressions over the structure and the value of the attributes in the service descriptions.

We present an example to give a flavour of what information is contained through advertisements and queries. Let us consider a typical advertisement for the services of sale, shipping and insurance of a given good in a B2B marketplace. The advertisement has to contain parameters to describe aspects of all the services. For the sale itself, it is necessary to have a description of the good touching on its characteristic attributes. For instance, in a B2B marketplace for flowers, prospective sellers sort their offering by variety, stem length, colour, region of provenience and price. Furthermore, product ratings provided by the growers themselves can be present, along with photos and descriptions for most products.

When services such as payment and shipping are provided together with the sale, descriptions are further complicated. The advertisement may then present aspects such as delivery date and location, or form of payment. Finally, for complex business interactions, a behavioural specification of the collaborative business process that includes definition of roles (e.g. payer, payee, insurer or shipper) and their interactions [14] could be included. An analysis of that is beyond the scope of this paper.

2.3. Use Cases for a Matchmaking Service

In this section, we sketch a short list of very simple use cases to make progress towards the definition of the features of a matchmaking service. In all use cases, the result is that the party requesting the matchmaking service obtain information on published advertisements. The party is then responsible for following up by getting in contact with the publisher of the advertisement.

2.3.1. Use Case 1: Browsing

Party browses the advertisement repository. Party manually finds what it wants by drilling down through the categories.

2.3.2. Use Case 2: Volatile query

Party submits a query to the matchmaker (advertisements repository). The matchmaker immediately returns matching advertisements that are currently present in the repository.

2.3.3. Use Case 3: Persistent query

Party submits a *persistent* query to the matchmaker (advertisements repository). The persistent query is a query that will remain valid for a length of time defined by the party itself. The matchmaker immediately returns matching advertisements that are currently present in the repository. Within the validity period of the query, whenever an advertisement is added or updated that matches the query, the matchmaker will notify the party. The party can decide to remove the persistent query from the matchmaker before the validity period is ended.

2.3.4. Use Case 4: Advertisement

Party posts an advertisement to the matchmaker. This advertisement describes what the party requires or provides and is publicly available to all parties. As with the persistent query, the advertisement is persistent and has a validity period. The matchmaker returns all matching advertisements that are currently present in the repository. Within the validity period of the advertisement, whenever an advertisement is added or updated that matches the query, the matchmaker will notify the party. The party can decide to remove the advertisement from the matchmaker before the validity period is ended.

2.3.5. Use Case 5: Advertisement with visibility rules

Same as the previous case, except that the party adds visibility rules to the advertisement. These visibility rules define who can see the advertisement based on publicly available attributes of the requestor such as identity or business category.

2.4. How the Matchmaking Service Operates over Advertisements and Queries

As it is apparent from the use cases, the job of the matchmaker is to match together compatible advertisements and return advertisements that satisfy a query. To clarify this point it is worth specifying that:

- Two advertisements are compatible when there exists a realization of a service that has all the characteristics expressed in both service. The matchmaking service will match service requests with compatible service offers.
- An advertisement satisfies a query when there exists a realization of a service that satisfies all the constraints that are expressed in the query.

The job of the matchmaker is therefore to perform operations over the language constructs. In the following section, we start to investigate what are the properties that such a language should possess.

3. Requirements

From the analysis carried out in section 2, we derive a set of requirements for a language to express service descriptions in the context of a matchmaking service.

The first observation that we can make is about the potential complexity of the descriptions. While some aspects of the description can be expressed with simple attribute-value pairs, some others might require more structuring. Levels of specifications can be nested so to form grouping and tree/graph structures. This requires a flexible and expressive metadata model.

Requirement 1: High degree of flexibility and expressiveness

As it happens in traditional business, advertisements are very often under-specified to leave open some aspects of the service to a successive stage of negotiation. Moreover, advertisers should be allowed not to mention some details of the service they provide or require, because they might not have the information, they might not want to disclose it, or simply might not be interested in it.

Requirement 2: Ability to express semi-structured data

When publishing advertisements or submitting queries, it is essential to be able to work at different levels of generality. When querying the repository for services of a certain type, we need to make sure that all the instances of service types that are subsumed by the requested type are retrieved. As an example, when we require flowers, we expect be matched with anyone providing roses.

Requirement 3: Support for types and subsumption

In querying and advertising, it is usually the case that what is expressed is not a single instance of a service, but rather a conceptual definition of the acceptable instances. A natural way of describing this is by expressing constraints over the parameters of the service.

Requirement 4: Ability to express constraints

As an aside, we note that the descriptions must be understandable by all the participants. This is difficult because the participants can potentially use their own formats to internally represent their products or services. In order for them to interoperate and to provide a powerful subsumption mechanism there is a need for using ontologies. An ontology goes beyond the simple specification of a set of terms; it also expresses relationships between them. There are many ontology efforts, either reference ontologies such as WordNet [8], or domain ontologies, i.e. developed for vertical industries (see TranXML [20] for the transportation as an example). To design an ontology is beyond the scope of our work. We only require descriptions to refer to an ontology in order to mediate between diverse information sources.

In the remainder of this paper, we take into consideration existing language for knowledge representation both from industry standard framework for e-commerce and from the W3C semantic web initiative.

4. Standard Frameworks for E-Commerce

In this section, we assess some industry standard frameworks for e-commerce with respect to the requirements that we identified in the previous section. The standards we considered were UDDI and ebXML.

4.1. Universal Description Discovery and Integration (UDDI)

UDDI is a cross-industry effort driven by a set of major platform and software providers, as well as marketplace operators and e-business leaders. The aim of UDDI is to create a global, platform-independent, open framework to enable businesses to discover each other, define how they interact over the Internet, and share information in a global registry that will more rapidly accelerate the global adoption of B2B e-commerce [4].

When trying to implement a matchmaking service based on UDDI, we incurred in the following problems:

- there is no classification or organisation of UDDI data structures, the tModels (cf. Req. 3);
- tModels only provide a tagging mechanism. UDDI is only intended to provide a first level filter. Further discrimination is done in direct communication with the service provider (cf. Req. 2);

- searching is only done by string equality matching on some fields such as name, location or URL (cf. Req. 3 and 4);
- the description schemata are not extensible (cf. Req. 1 and 2).

4.2. *E-Business eXtensible Markup Language (ebXML)*

ebXML is a set of specifications that together aim to enable a modular electronic business framework. ebXML specifications have XML messaging as a common basis. ebXML is a joint initiative of the United Nations (UN/CEFACT) and OASIS, developed with global participation for global usage.

We briefly considered ebXML as a platform for our matchmaking service. ebXML defines core components like name, address and suchlike information. However, ebXML is very focussed on defining business processes definition and business documents payload. The data model of the Core Component vocabulary does not look very rich and they do not provide support for semi-structured data (cf. Req. 1), inheritance (cf. Req. 3) and constraints (cf. Req. 4).

4.3. Other e-commerce frameworks

In terms of requirements for discovery and matchmaking, none of the other frameworks for e-commerce that we looked at (RosettaNet [17], eCo [6], BizTalk [12]) seemed to provide anything beyond a basic ontology definition.

5. Semantic Web Technologies

As we argued in the previous section, the metadata models used by the main industry standards do not meet the requirements that we stated. Therefore, for the development of a prototype of an advanced matchmaking service, we turned our attention to the semantic web initiative at the W3C consortium.

5.1. Semantic Web

> *The Semantic Web is a vision: the idea of having data on the Web defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration and reuse of data across various applications.*
> [From the Semantic Web activity statement]

The Semantic Web vision from the semantic web activity statement [18] fits well with the context that we set for our matchmaker. Moreover the efforts currently underway to develop metadata tools and languages promise to offer appropriate responses to the problem that arise in the development of a prototype matchmaker. This becomes evident when comparing the requirements that we collated in section 2 with some of the W3C specifications, namely the Resource Description Framework (RDF) and the Darpa Agent Markup Language (DAML). We have experimented with both RDF and DAML as well as with some of the related tools currently available or even in course of development, and here we report on our experience with them.

5.2. RDF

RDF is a general-purpose knowledge representation language, and its flexible data model seems to fulfil our needs.

5.2.1. Expression of the service parameters in RDF

The basic RDF data model [10] consists of three object types: *resource*, *property* and *statements*. Resources are the central concept of RDF. They are used to describe anything, from web pages to people. Properties express specific aspects, characteristics, attributes, or relations used to describe a resource. Statements are composed of a specific resource together with a named property and the value of that property for that resource. The value can be a resource in turn. Alternatively, the value can be a *literal*, a primitive term that is not evaluated by an RDF processor. RDF models consist of a bag of statements and are represented as directed labelled graphs, as in the example in Figure 1.

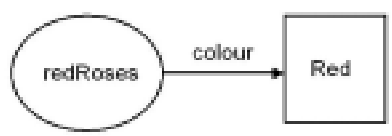

Figure 1: An RDF statement

Since RDF is about neutrally representing knowledge rather than associating specific semantic to a representation, we need to state what interpretation we are going to attach to the representation that we use. The interpretation will depend on the context. For example, let us consider the RDF statement in Figure 1. If it appears in an advertisement for the sale of roses, our matchmaker would interpret it as stating that the colour of the roses for sale *is* red. The same statement in an advertisement for the purchase of roses would be interpreted as stating that the colour of the roses is *required* to be red.

As one would expect, the matchmaker considers the two advertisements as compatible.

5.2.2. Ontology

As we hinted at in section 3, the terms that are expressed in the advertisements have to be defined in an ontology. The design of such an ontology is beyond the scope of this paper. However, we will just underline here that RDF models are typically enriched by an RDF Schema (RDFS) [5]. The RDFS Specification describes how to use RDF itself to define vocabularies of RDF terms. During the development of the matchmaking prototype we made the assumption that there exists some ontology description in RDFS. Our schema borrowed from various ontologies to express concepts such as delivery services or descriptions of flowers. In our example of the B2B flowers marketplace, the ontology describes concepts such as flower type, stem length, region, colour quality, as well as service related concepts such as delivery date and location. It could also include rules of engagement for the negotiation of other terms and conditions of the service. The automatic – or semi-automatic - merging of ontologies is a difficult problem [13], which we do not take into consideration in this work.

As an aside, it is worth noting that the ontology might import or include information from UDDI yellow or white pages or ebXML registries. In this way, we get the best of the two worlds: extended reach thanks to the industry standard frameworks and expressiveness from the semantic web.

5.2.3. Advertisements in RDF

We describe an advertisement as an RDF graph that defines a space of possible realizations of one or more services, not by expressing a particular realization of the service(s). Therefore, some of the aspects of the advertisement need to be expressed through constraints (cf. Req 4 in the previous section).

Figure 2 represents an example of an advertisement. Together with the advertisement, terms are represented that belong to the ontologies and would be shared by the different parties involved in a matchmaking session. For the sake of our example, we have modelled the services of sale and delivery. The service of sale defines a set of items, the total price of the sale and whether the intention of the advertiser is to buy or to sell. The product can specify a quantity. In turn, the quantity is expressed by a measure that can be either volume or weight. The delivery service defines charge, delivery date, origin and destination locations.

Each advertisement is represented as an RDF resource of type `Advertisement` and as a result has its own URI. It designates the root node of the description. Properties from this resource will characterize the types of services that are required or provided. By using our example ontologies, it is possible to add further details and form a full RDF advertisement. In Figure 2[2], the RDF sub-graph representing the advertisement is highlighted in colour. The advertisement is for the wholesale purchase and delivery of 100 kg roses. The root of the advertisement is the `myAdvert` resource. Navigating the graph from `myAdvert`, all the relevant information is reachable. For instance, the desired quantity and colour of the item for sale can be read as properties of the `redRoses` node.

As we argued above, advertisements express constraints over the possible realizations of a service. RDF is useful in dealing with two kinds of constraints. Typing constraints are used to say that a node must be of a certain type, or any subtype or supertype of it. We express them by using the `rdf:type` relation. In our example, the node `redRoses` expresses that a potential matching advert needs to have a node of type `Rose`, or a subclass of it. Equality constraints on values are obtained by specifying a value for a literal. In our example, the buyer is only interested in buying exactly 100 kg of flowers.

---

[2] Obtained with the Protégé ontology editor [19].

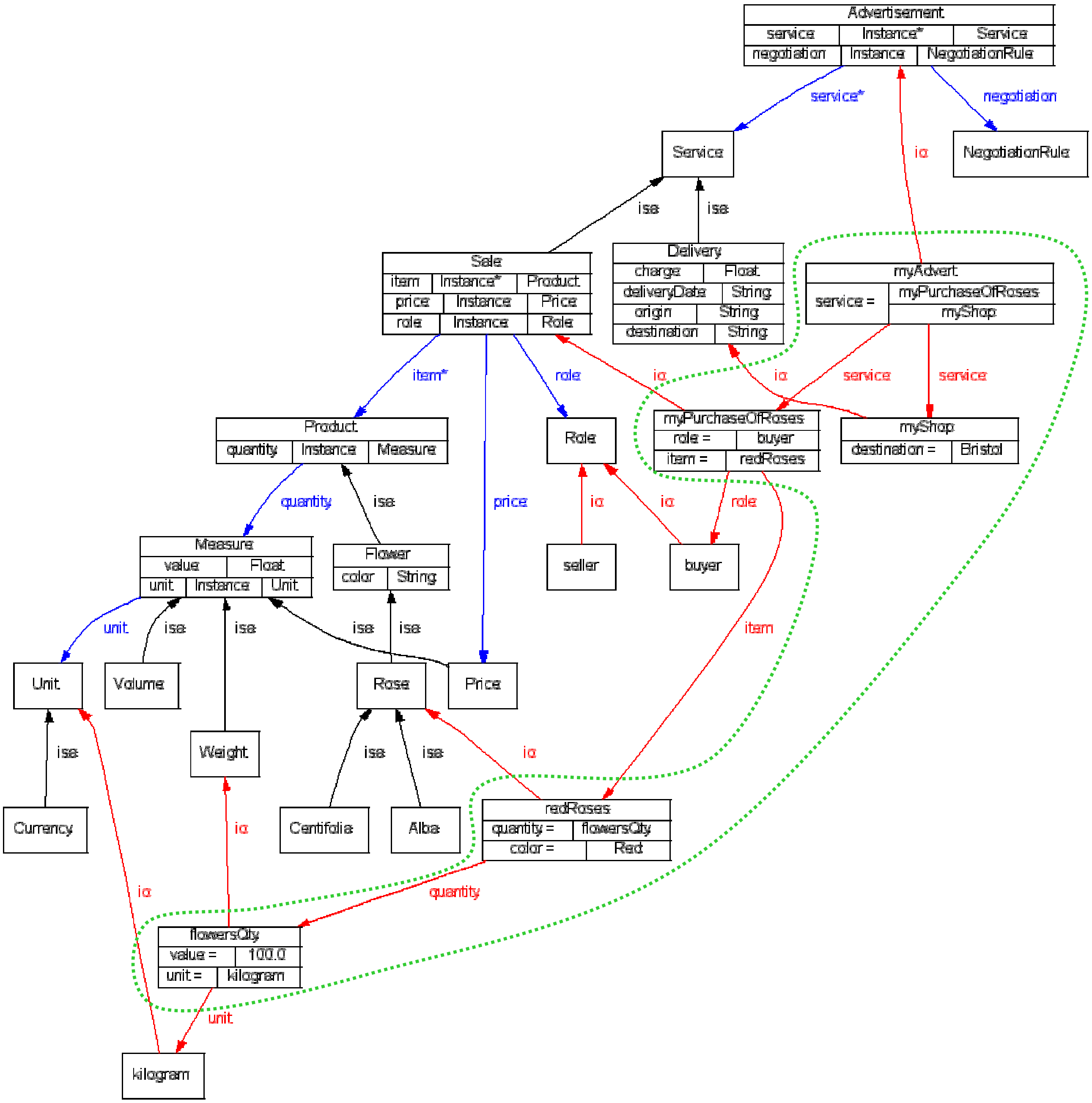

Figure 2: An advertisement enriched with ontologies

5.2.4. Matching

With the design decisions that we made, matching of advertisements is reduced to matching of RDF graphs. We have implemented a matching algorithm, based on the visitor pattern [7]. The algorithm is implemented in Java and based on the Jena RDF API [11]. Advertisements match when their root nodes (that must be instances of `Advertisement`) match, and all their respective sub-nodes do too.

Following the visitor pattern, we have defined a default matching rules and a mechanism for it to be overridden for nodes of particular types. The association of overriding matching rule with a node type is done by annotating the RDFS schema. Precisely, a `matchingRule` property is added to the node representing a Class resource. The value of the property is a literal that expresses the fully qualified name of the java class that implements the matching rules for the resources of that type.

The default matching rule is presented in Figure 3. The main idea is to traverse simultaneously two advertisements by finding recursively the nodes that share a common type and making sure there are no incompatibilities between the advertisements. In the following algorithm description, we ignore the possibility of cycles, for simplicity of explanation.

```
Two Advertisements match when:
   Their root node match.

Two nodes match when:
   One of them is a subtype of the other.
   AND
      IF
        A matching rule is defined for the most specialised common type
        between the two nodes, the matching rule is evaluated positive.
      ELSE (default matching rule)
        FOR each property p1 that appears in one node such that there
        exists p2 in the other node where p1 = p2 or p2 is a sub-
        property of p1,
           The two object nodes from p1 and p2 match.
```

Figure 3: Matching algorithm for two RDF graphs

Let us go back to our flower advertisement (presented without the related ontologies in figure 4) and examine it in light of the matching algorithm. Our buyer is interested in having her advertisement matched against compatible advertisements from sellers. In the rest of this sub-section, we show some advertisement that would match our buyer's and some that would not. Our buyer advertised for the purchase of 100 kg of red roses.

Among the published advertisements in the matchmaking repository we consider the following:

Table 1: Matching of Advertisements

Sale and delivery of	*Result*	*Justification*
flowers	Hit	`Rose` is a subclass of `Flower`
roses	Hit	No problem
100 kg of blue roses	Miss	Mismatch on the colour property
daffodils	Miss	`Daffodil` is a subclass of `Flower` but is disjoint with `Rose`: mismatch on the types
100 kg of long stem centifolia roses	Hit	`Centifolia` is a subclass of `Rose`
100 kg of short stem alba roses	Hit	`Alba` is a subclass of `Rose`[3]
Up to 300 kg of roses	Hit	Compatible constraint. See discussion

[3] Actually it should be a miss because Alba is variety of white roses. It is a limitation of the design of our ontology. See the discussion subsection.

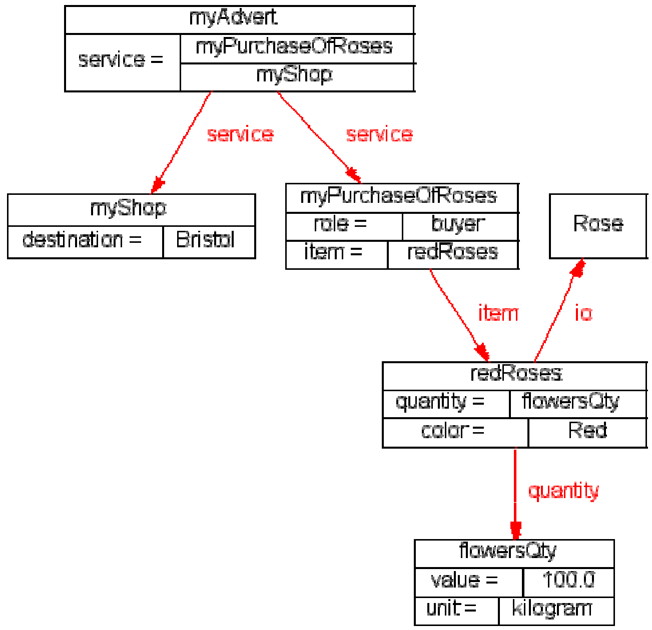

Figure 4: Buyer's advertisement

The first six examples are easily expressed by the syntax illustrated so far. However, for the last advertisement, we need to devise some mechanism to express and resolve the constraint expression that appears in the seller's advertisement.

In our first prototype of a matchmaker, we have designed a proprietary syntax to express the constraints directly in the value slot of the instance. In our example, to express an advertisement for the sale of a quantity of roses up to 300 kg, we set the literal corresponding to the `rdf:value` to be the literal "`LessOrEqualThan 300`". The matching algorithm parses the value string and interprets the constraint. The drawback with this approach is that is based on sole syntax. To overcome this problem, we envisage to design a constraint ontology that will allow us to annotate nodes in the `Advertisement` graph as representing an instance of a particular kind of constraint (such as `LessOrEqualThan 300` in the example). We expect this to have a minimal impact on the design and implementation of the matching algorithm.

The discussion so far has been focussed on advertisements only. As we observed in section 2, advertisement and queries have much in common. Our bias is that they can and should use the same filtering mechanism and possibly be expressed through the same language constructs. The constraint system we have just introduced provides this filtering mechanism. The last example could be seen as both an advertisement and a query.

5.3. Discussion

At the end of section 3, we listed the following requirements for a description language to be used in matchmaking services:

- high degree of flexibility and expressiveness;
- ability to express semi-structured data;
- support for types and subsumption;
- support for constraints.

Based on our experience, RDF is offers valuable support to meeting the first three. On the other hand, we find that it falls short on our requirement 4: support for constraints. To overcome this problem we had to design a proprietary mechanism. One of the problems we had to express advertisements in RDF is the one raised in the Alba example in the previous sub-section. Alba is a variety of roses that comes in white colour only. If a seller advertises the sale of Alba roses, his advertisement should not be matched with the one proposing to purchase red roses. Our matching algorithm would mistakenly match the two advertisements because the ontology does not express the restriction of the colour property of Alba class to be white. We found this difficult to express in RDFS, as it does not seem possible to restrict the range of a property for a subset of its original domain

5.3.1. DAML

More recently, we started to look at DAML+OIL [9] to enrich our RDF descriptions. DAML+OIL looks promising to overcome both of the hurdles that we encountered with RDF. We have started to experiment and express the concepts of a service description as DAML+OIL classes. The service description is defined as the boolean combination – intersection, disjunction or complement – of a set of restrictions over datatypes and abstract properties. These restrictions are expressed either through DAML+OIL restrictions or through XML Schema restrictions.

On the constraint support side, DAML+OIL allows us to define concepts using restrictions, for instance existential qualifiers, universal qualifiers or cardinality over properties. Rich datatype definitions can also be used in these restrictions and are defined in XML Schema, leaving us in particular the possibility to express ranges.

Because DAML+OIL classes can be restricted on the target value/class of a property, it is possible to create richer ontologies. Our example ontology for flower could be more complete by adding the fact that Alba is a sort of Rose whose colour is always white.

5.3.2. RDF Tools

RDF tools that provide an interesting set of features start to become available. There are many APIs allowing to manipulate RDF models, either providing a low-level triple-store abstraction or a providing a higher graph abstraction. We chose the second type of implementation, and more particularly Jena, as we felt it would be more suited to our problem.

Protégé [19] is a very good tool to design ontologies in RDF, even though it does not support all the features of RDF (the absence of multi-class membership is an example). Protégé also provides some interesting features not present in RDF that look very promising for some future DAML support.

6. Related work

RDFSuite [1] and KRAFT [15] are highly relevant to our present work. For a discussion on how they relate, see the section on future work.

Reynolds [16] presented an RDF framework for resource discovery. In the context of his framework, the directed graph query language (DGQL) is a simple query language for RDF based on graph matching. However, now DGQL can only perform equality and indifference tests. What we propose to do is based on a wider array of possibilities for constraint matching.

7. Future work

As we highlighted in the discussion in section 5.3, an important part of follow up work for our matchmaker will be the design of a constraint ontology and the implementation of a mechanism to integrate constraint solving with the current matching algorithm. We envisage that RDFPath - or approaches of its kind - will be useful to apply the visitor pattern to the task of matching sub-graphs.

We also would like to extend this work and enlarge its scope from matchmaking to other phases of the e-commerce process. In particular, because constraints expressed in matchmaking advertisements may be similarly useful in expressing negotiation proposals [2], we intend to extend our matching algorithm and metadata model to cope with automated negotiation. In this framework, negotiation proposals are expressed as service descriptions the same way advertisements are.

Matchmaking does not necessarily require expressing behavioural aspects of a service. However, in the B2B environment, collaborative business processes are a fundamental concept. Therefore, we plan to extend our work to include aspects of service behaviour.

We follow with interest the work on RDFSuite and RQL in particular [1]. RDFSuite provides a suite of tools for RDF storage and querying. They manage to achieve good results in scalability because of their use of database technology. When we will tackle the problem of persistence of the advertisements repository, we plan to investigate the use of RDFSuite.

KRAFT [15] is an architecture for supporting virtual organization that uses constraints as a knowledge exchange medium. KRAFT is highly relevant to our work, especially as we move onto defining an ontology of constraints.

8. Conclusion

Our experience in prototyping an advanced matchmaking service made us to realize that there is a gap between what standard frameworks for e-commerce provide today and what could be achieved through the usage of semantic web technologies. We believe that in the near future automated matchmaking and negotiation will achieve results at a level of complexity far beyond what is possible today. And semantic web tools and technologies will play a primary role in making that happen.

9. References

[1] Alexaki, V; et al. The ICS-FORTH RDFSuite: Managing Voluminous RDF Description Bases, Proceedings of the Second International Workshop on the Semantic Web (SemWeb'2001), May 2001.

[2] Bartolini, C; Priest, C. A Framework for Automated Negotiation. HP Labs, Technical Report. 2001.

[3] Bechhofer, S; Gobbe, C. Delivering Terminological Services. University of Manchester.

[4] Boubez, T; et al. UDDI Data Structure Reference V1.0, UDDI Open Draft Specification, Sep 2000.

[5] Brickley, D; Guha, R.V. Resource Description Framework (RDF) Schema Specification 1.0, W3C Candidate Recommendation, Mar 2000; available at http://www.w3.org/TR/rdf-schema/.

[6] CommerceNet, Inc. eCo Architecture for Electronic Commerce Interoperability, 1999.

[7] Gamma, E; Helm, R; Johnson, R; Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software. Reading MA: Addison-Wesley, 1995.

[8] Fellbaum, C. WordNet: An electronic lexical database. The MIT Press. 1998.

[9] Hendler, J; McGuinness, D.L. The DARPA Agent Markup Language, *IEEE Intelligent System*s, vol. 16, no. 6, Jan./Feb., 2000, pp. 67–73.

[10] Lassila, O; Swick, R. 1999. Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation, Feb 1999; available at http://www.w3.org/TR/REC-rdf-syntax/.

[11] Mc Bride, B. Jena: Implementating the RDF Model and Syntax Specification, Proceedings of the Second International Workshop on the Semantic Web (SemWeb'2001), May 2001.

[12] Microsoft. BizTalk Framework 2.0: Document and Message Specification, MSDN Online, Jun 2000.

[13] Omelayenko B., Syntactic-Level Ontology Integration Rules for E-commerce, Proceedings of The 14th International FLAIRS Conference (FLAIRS-2001), May 2001.

[14] Piccinelli, G., Mokrushin L. Dynamic Service Aggregation in Electronic Marketplaces. Special Issue on Electronic Business Systems of the Computer Networks journal. HP Labs, Technical Report. 2001.

[15] Preece, A; et al. KRAFT: Supporting Virtual Organisations thought Knowledge Fusion, *AAAI-99* Workshop on Artificial Intelligence for Electronic Commerce, 1999.

[16] Reynolds, F. An RDF Framework for Resource Discovery, Proceedings of the Second International Workshop on the Semantic Web (SemWeb'2001), May 2001.

[17] RosettaNet Organization. http://www.rosettanet.org, 2001

[18] Semantic Web Activity. http://www.w3.org/2001/sw/, 2001.

[19] Stanford Medical Informatics. The Protégé Project. http://protege.standford.edu, 2001

[20] Transentric. TranXML: the Common Vocabulary for Transportation Data Exchange. 2001.