Semantic Web Support for the Business-to-Business E-Commerce Pre-Contractual Lifecycle[1]

David Trastour, Claudio Bartolini, Chris Preist
HP Laboratories Bristol
HPL-2003-174
November 21st , 2003*

E-mail: david.trastour@hp.com, claudio.bartolini@hp.com, chris.preist@hp.com

semantic web, e-commerce, matchmaking, automated negotiation, DAML+OIL, service description

If an e-services approach to electronic commerce is to become widespread, standardisation of ontologies, message content and message protocols will be necessary. In this paper, we present a lifecycle of a business-to-business e-commerce interaction, and show how the Semantic Web can support a service description language that can be used throughout this lifecycle. DAML+OIL is a sufficiently expressive and flexible service description language to be used not only in advertisements, but also in matchmaking queries, negotiation proposals and agreements. We also identify which operations must be carried out on this description language if the B2B lifecycle is to be fully supported. We do not propose specific standard protocols, but instead argue that our operators are able to support a wide variety of interaction protocols, and so will be fundamental irrespective of which protocols are finally adopted.

* Internal Accession Date Only
[1] Computer Networks, Vol.42, iss.5, pp 661-73, 5 Aug. 2003
Approved for External Publication

Semantic Web Support for the Business-to-Business E-Commerce Pre-Contractual Lifecycle

David Trastour Claudio Bartolini Chris Preist

Hewlett-Packard Laboratories
Filton Road, Stoke Gifford, Bristol, BS34 8QZ, UK

Abstract

If an e-services approach to electronic commerce is to become widespread, standardisation of ontologies, message content and message protocols will be necessary. In this paper, we present a lifecycle of a business-to-business e-commerce interaction, and show how the Semantic Web can support a service description language that can be used throughout this lifecycle. DAML+OIL is a sufficiently expressive and flexible service description language to be used not only in advertisements, but also in matchmaking queries, negotiation proposals and agreements. We also identify which operations must be carried out on this description language if the B2B lifecycle is to be fully supported. We do not propose specific standard protocols, but instead argue that our operators are able to support a wide variety of interaction protocols, and so will be fundamental irrespective of which protocols are finally adopted.

Key words: Semantic Web, E-Commerce, Matchmaking, Automated Negotiation, DAML+OIL, Service Description

1 Introduction

Electronic commerce is having a revolutionary effect on business. It is changing the way businesses interact with consumers, as well as the way they interact with each other. Electronic interactions are increasing the efficiency of purchasing, and are allowing increased reach across a global market.

Email addresses: david.trastour@hp.com (David Trastour), claudio.bartolini@hp.com (Claudio Bartolini), chris.preist@hp.com (Chris Preist).

With the increasing availability of the web, a more open e-commerce environment is developing, allowing businesses to trade more flexibly with each other. Some of this openness is achieved by competition between web portals, while some competition occurs within a single web portal, acting as a marketplace for buyers and sellers to meet.

The e-commerce community is creating new infrastructures to support high-level business-to-business (B2B) and business-to-consumer interactions on the web. The effort is concentrated on defining a new generation of electronic data interchange protocols, mostly based on XML (like OASIS, BizTalk and RosettaNet) and on creating new kinds of e-business services such as agent-mediated B2B e-commerce, and knowledge-driven customer relationship management.

On the other hand, with the semantic web initiative, the World Wide Web Consortium (W3C) is developing a range of proposals aimed at supporting intelligent information intensive operations over the web. The emphasis is on enriching the web's data markup languages with knowledge representation features, to permit inference over the content of web pages (prominent initiatives include DAML+OIL, and RDF). Its goals include the production of internet-scale inference mechanisms, knowledge markup languages, and active information-seeking services.

The goal of this paper is to explore how semantic web technology can provide support to the business-to-business e-commerce lifecycle. In particular, we study the early (pre-contractual) phases of the lifecycle. We present a conceptual framework for modelling business-to-business interactions. Within it, we experiment with semantic web technology (in particular DAML+OIL) as a mean to express semantically rich descriptions of services and goods. It is important to note that the environment here presented can be used both for automated interactions, but also to provide structured interactions between humans

This paper is structured as follows. In section 2 we introduce our conceptual framework. In section 3 we explore the phases of matchmaking and negotiation in detail, with particular attention to the operations that are carried out on the messages that participants exchange. In section 4, we identify the need for a declarative language for service descriptions, derive requirements for it and show that DAML+OIL satisfies them. We also present a set of example service descriptions used at various stages in the B2B e-commerce interaction lifecycle. In section 5, we specify the operations that are required during the B2B lifecycle, and demonstrate that they can be straightforwardly be implemented on a description logics reasoner (section 6). We then discuss related work (section 7) and we conclude presenting our future work intentions (section 8).

2 E-Services Framework

We have developed a lifecycle model to help us understand the interactions which take place between businesses engaged in e-commerce. This model (based on that in [1]) follows the lifecycle of an interaction between two (or more) parties and has the following stages:

Matchmaking: A trader locates other traders that it could potentially do business with. This is done by some traders placing *advertisements*, and others making queries over these advertisements.

Negotiation: The trader enters into negotiation with one or more of these potential business partners, to see if they can agree mutually acceptable terms of business. This is done through an interchange of *negotiation proposals* describing constraints on an acceptable deal. The outcome of this is an *agreement,* specifying the terms that both parties consider acceptable. These terms could include a definition of the good or service being traded, price, delivery date, etc.

Contract Formation: The *agreement* is transformed into a legally binding *contract*.

Contract Fulfillment: The parties carry out the agreed transaction, within the parameters specified in the contract. The transaction may be automatically monitored, and parties would be warned if any behaviour outside the agreed terms of the contract takes place.

If this open e-commerce environment is to become pervasive, interactions throughout this lifecycle must be standardised by the industries using it. Standardisation must take place at three levels:

(1) Standards for business-specific ontologies which describe goods, services and contracts being traded. These ensure that when one trader uses a set of terms to describe a given good, another trader will be able to interpret them accurately.
(2) Standards for specifying the format of advertisements, proposals, contracts and other constructs which are used during B2B interactions. These standards would specify the syntax of these constructs, with the semantics being defined by the ontologies. Hence, these standards need not be business-specific.
(3) Standards that specify the protocols which traders use to interact with each other during different phases of the B2B lifecycle. These determine the messages that are sent back and forth containing the standard constructs described above.

The ARPA knowledge-sharing project [2] was the first to tackle these standardisation issues, albeit in the domain of information exchange rather than

e-commerce. Ontolingua [3] provides a tool for defining standard ontologies, KIF [4] a language for representing information and KQML [5] a set of messages for exchanging this information. The FIPA [6] agent standardisation effort has defined a messaging language, and protocols for conducting B2B interchanges such as auctions. While some of the ideas developed in these efforts are clearly important (such as the notion of advertising and facilitators), they do not provide appropriate primitives for defining the constructs used in e-commerce.

The focus of this paper is primarily on standardisation of B2B constructs (point 2). As a result of this, we assume the existence of appropriate domain-specific ontologies. We believe that the semantic web provides an opportunity to develop a service description language that can be used throughout the lifecycle of interaction, rather than just at the advertising phase.

In the industry standard arena, initiatives such as UDDI [7], ebXML [8] and WSDL [9] are gaining momentum. However, they offer limited support for ontologies, semi-structured data and constraints which are essential when modelling B2B constructs as we shall discuss in section 4, and have argued more fully in [10].

In this paper, we demonstrate that a DAML+OIL [11] based service description language is sufficiently expressive and flexible that it can be used not only in advertisements, but also in matchmaking queries, negotiation proposals and agreements. We also identify which operations must be carried out on this description language if the B2B lifecycle is to be fully supported. We do not propose specific standard protocols, but instead argue that our operators are able to support a wide variety of interaction protocols, and so will be fundamental irrespective of which protocols are finally adopted.

3 Matchmaking and Negotiation

In this paper, we focus on the first two stages of the e-commerce lifecycle presented above: Matchmaking and Negotiation. We now describe these phases in detail, showing the different interaction protocols that can be used. We identify commonalities between the different protocols, to allow us to develop a service description language and set of operators able to support them all. We also identify commonalities between the different stages, to allow the same service description language and operator set to support both stages in the lifecycle.

As [12] demonstrates, different protocols may be appropriate in different situations, depending on the expected message flow. Hence, it is not appropriate

to standardise on a unique protocol for all agent systems. Instead, we should allow choice from a variety of such protocols, but standardise aspects of the roles which are common to all of them. Protocol specifications determine where information is stored, and how appropriate messages are passed to access it. Role specifications determine how the information is represented, accessed and used.

3.1 Matchmaking

Matchmaking is the process whereby potential trading partners become aware of each other's existence [13]. A buyer wishing to purchase access to a service must locate potential service providers able to meet its needs. The buyer's requirements may initially be not fully specified, and the service providers may be able to offer a range of services. The process of matchmaking should not result in the service becoming fully specified: this is the purpose of the negotiation phase which follows. Instead, the matchmaking phase should result in a buyer (or service provider) having a list of potential trade partners, each with an associated partially specified service description. This description defines the set of possible services the provider can offer which are of interest to the buyer.

There are many different protocols which can be used to accomplish this. Work on information brokerage using KQML [5] or the FIPA proposed standards [6] have identified the majority of these.

Despite these different agent architectures and communication protocols that can be used to achieve the matchmaking process, we can identify clear roles which are common to all of them. We have a *repository* of information about services or service requirements, which is maintained by the *repository host*. Agents adopting *advertiser* role are willing to *advertise* descriptions of services in the repository. These are usually, though not always, service providers. (They may be buyers, advertising a service request, or may be marketplaces offering environments where such services can be traded.) Similarly, agents adopting the *seeker* role wish to locate appropriate advertisers. Seekers can *query* a repository, via the repository host, and may be able to *browse* the repository.

We now show how these roles are assigned on some canonical examples.

(1) A buyer agent broadcasts its requirements to all agents in the system, irrespective of their abilities. Those agents able to meet the buyer's needs reply with information about what they are able to offer (as in the start of the Contract Net negotiation protocol [14]). The roles are as follows: the buyer adopts the seeker role, and all service providers adopt the role

of advertiser and repository host. However, the repository is local to the service provider, and only contains information about their own service offerings. The buyer must broadcast their query.

(2) An agent community has a centralised facilitator agent, which provides a yellow-pages service. Service providers send advertisements, consisting of descriptions of the service they offer, to the facilitator. Buyers send queries, and receive lists of providers potentially able to satisfy their requirements in response. UDDI provides a simple example of such a service. For use-case analyses of these and variants on this protocol, see [10]. In this third protocol, the facilitator agent plays the role of repository host.

For the advertiser to place an advert, it must be able to specify the set of services it is interested in trading. In many cases, these services will not be fully specified immediately. Because of this, it needs a language which is rich enough to allow an abstraction of a service to be advertised, together with constraints over that abstraction. Similarly, for a seeker to make a query, it must be able to specify, as an abstraction together with constraints, the set of services it is interested in. When a query is made, this is treated as a constraint on the acceptable set of services to the seeker. The repository host must identify which advertisements are compatible with the query. An advertisement is compatible with a query if there is at least one instantiation of the advertisement which is also an instantiation of the query. The repository host responds with a list of all such advertisers and their advertisements. Ideally, it would also return an abstraction of a service, together with constraints, specifying the most general solution acceptable to both the seeker and the advertiser.

3.2 Negotiation

The negotiation stage of the e-commerce interaction lifecycle refines the abstract service specification from the matchmaking phase to a concrete agreement between two parties. Negotiation can be one-to-one, one-to-many or many-to-many, and as a result, many different protocols have been designed to carry this out. Negotiation protocols determine the interchange of messages which take place during negotiation, and the rules by which the negotiators must abide. One-to-one protocols [15] include the shop-front, where a seller simply offers a good at a fixed price, and iterated bargaining, with buyer and seller taking turns to exchange proposed agreements. One-to-many protocols include the English auction, the Dutch auction and the Contract Net [14]. Many-to-many protocols include the Continuous Double Auction and the Call Auction [16].

In the same way we analysed the different protocols for matchmaking in section 3.1, we can analyse the different negotiation protocols and identify roles

and behaviours common to all. Because of the rich variety of negotiation protocols available, this is more complex than the matchmaking case. We present our full analysis of the commonalities in [17], and use this analysis to define a general software framework for negotiation. This framework abstracts away from the particular message interchange required for a given protocol, and can be parameterised with rules to implement different protocols.

From the analysis of the negotiation protocols presented above, and others, we can identify certain abstract roles, data structures and behaviours common to all. In each case there are at least two *negotiation participants* trying to make a deal with each other. In addition, there is at least one (possibly more) *negotiation host*, responsible for enforcing the rules of the negotiation and ensuring it goes smoothly. Before negotiation can begin, the parties must have agreed what the negotiation will be about (usually as a result of the matchmaking process). Hence, this places a restriction on the parameters and values to be negotiated. We call this restriction the *negotiation template*. The negotiation template refers to a common ontology accepted by all participants in the negotiation. It defines a schema for valid *negotiation proposals* that participants submit to each other. The schema declares which fields are admissible and how their values are constrained. A proposal is a further refinement of the negotiation space that represents a configuration of parameters that would be acceptable to the submitter. The result of the negotiation process is an *agreement*. That is a configuration of parameters that is non-ambiguous and can be used during the execution phase to instantiate the service. Therefore we can define the negotiation process as the process through which participants move from a pre-agreed *negotiation template* to an *agreement*, via an exchange of *negotiation proposals*. A single negotiation may involve many parties, resulting in several agreements between different parties and some parties who do not reach agreement. For example, a stock exchange can be viewed as a negotiation where many buyers and many sellers meet to negotiate the price of a given stock. Many agreements are formed between buyers and sellers, and some buyers and sellers fail to trade.

Reviewing the three example protocols described above, we can represent them in terms of our abstract roles and behaviours.

The Contract Net manager is both negotiation participant (as buyer) and negotiation host. The contractor is a negotiation participant (as seller). The bid specification within the task announcement defines the negotiation template, and the bids made by contractors are negotiation proposals. The award specification is the final agreement, determined by the contract net manager.

During iterated bargaining, there are two negotiation participants. One, possibly both, will also play the role of negotiation host (to ensure that the other party submits proposals at the appropriate time, in an appropriate format.)

The negotiation template will be agreed prior to the negotiation, usually as the output of the matchmaking process. Each proposal is a fully instantiated version of the template, and when one party agrees to the proposal of the other, that proposal becomes the agreement.

In an auction, there is one seller participant (who remains silent after specifying the good for sale), many buyer participants, and the auctioneer who acts as the negotiation host. The negotiation template is fully instantiated to define exactly what good/service is for sale in the auction, except for the price which remains undetermined. The seller participant lodges a proposal with the auctioneer which states that they are willing to sell the good, and the minimum price they will accept. Buyers announce proposals, in the form of versions of the negotiation template with the price instantiated to their current bid. When no more proposals arrive, the last proposal is used as an agreement, provided the price is higher than the minimum price in the buyer's original proposal.

In [17] we have standardised some aspects of negotiation protocol checking, and parameterise these actions with declarative rules to enact different negotiation protocols. We also have developed a taxonomy of these negotiation rules. While a full presentation of this aspect of our work is beyond the scope of this paper, we now describe the three rules that are relevant to service description:

Validation Rule: When participants submit proposals, they first need to be validated with respect to the negotiation template. The validation step consists in making sure that the proposal is a more constrained form of the agreement template. That is, the constraints over the parameters in the proposal must be tighter that the corresponding ones in the agreement template. The constraints represent acceptable values to the proposing participant. (Often, these constraints will be a single acceptable value of a parameter.)

Improvement Rule: Given a a set of existing proposals within a negotiation, the improvement rule specifies what new proposals may be posted. For example, auctions often have a 'bid improvement' rule that requires any new proposal to buy to be for a higher price than previous proposals.

Agreement Formation Rule: If an agreement is to be made, there must be at least two valid proposals which are compatible with each other. Proposals are compatible if there is an identical fully-instantiated form of each.

Much work has gone into standardising the different protocols used in negotiation (for instance [6]) though this (rightly) accepts that many different protocols must be available. However, these standards do not standardise the agreement formation process, or the validation process. We propose that these actions, which are common to all negotiations, should also be standardised. Hence we introduce a language for describing templates, proposals and agree-

ments and operations on this language to carry out proposal validation and agreement formation. Furthermore, we design this language to be sufficiently general and flexible to cover the matchmaking phase.

4 Description of Services

In the previous section we have highlighted the information constructs that are exchanged in messages during matchmaking and negotiation, irrespective of what protocol is used. We have categorized them as advertisements, queries, negotiation proposals, negotiation templates and agreements. We have also identified the operations that are carried out on these constructs: matching, proposal validation, protocol checking and agreement formation. If these constructs and operations are to be standardised, we wish to build the constructs from a declarative language for describing services. Furthermore, we need to show that this declarative language can support the required operations over it. We now identify the requirements on this language, and then show that a DAML+OIL based language satisfy these requirements.

4.1 Requirements

A description language for the B2B e-commerce lifecycle should satisfy the following requirements:

(1) Descriptions should offer a high degree of flexibility. During the e-commerce lifecycle, services will be described with different degrees of complexity and completeness. For instance a negotiation proposal may be very descriptive in some aspects, but leave others less specified and open for further negotiation.
(2) Descriptions should express restrictions and constraints. Whether it is an offer or a request, it is often 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.
(3) Descriptions should easily lend themselves to performing the operations described in the negotiation and matchmaking sections. In particular, matching of advertisements with queries during matchmaking; validation of negotiation proposals against the negotiation template; and compatibility checking of two negotiation proposals to determine if an agreement can be made.
(4) Descriptions need to share a common semantics. Moreover descriptions should be able to use vocabularies created by different standard bodies or

industry sectors. Therefore support for interoperable ontologies is needed.

4.2 DAML+OIL based service descriptions

DAML+OIL [11] is a description logic based ontology language developped by the DAML DARPA programme.

DAML+OIL meets the requirements introduced in section 4.1:

(1) DAML+OIL is based on RDF [18] which supports semi-structured data.
(2) Constraints can be expressed with DAML+OIL property restrictions. DAML+OIL provides a rich set of description logic class constructors to build class expressions. However, it only supports unary constraints on datatypes which may not be sufficient for general e-commerce applications (for instance if the shipping cost needs to be linked with the dimensions of the good). A more expressive description logic supporting n-ary datatype constraints (like $\mathcal{SHOQ}(\mathbf{D_n})$ [19]) may then be needed. DAML+OIL will be expressive enough for e-commerce applications where commodity goods are being traded (for instance all standard corporate purchases) since no such constraints occur.
(3) DAML+OIL is a good candidate for expressing descriptions that will be subject to the operations of matching, proposal validation and agreement formation described in section 3. As we will see in the next section, all our operations can be expressed in terms of the subsumption operation [20]. DAML+OIL descriptions lend themselves very well to this operation and mature tools exist (such as the $\mathcal{SHIQ}$ reasoners Racer [21] and FaCT [22]) that can perform this operation on DAML+OIL models.
(4) DAML+OIL is an ontology language and provides clear semantics. Ontology editor tools such as OilEd [23] make the generation of new DAML+OIL ontologies for service descriptions much easier.

4.3 E-commerce constructs

In this section, we explain how we use DAML+OIL to describe the various descriptions that are used in the e-commerce lifecycle. While other more general efforts like DAML-S [24] already use DAML+OIL in their service descriptions, we show here how DAML+OIL is suitable for e-commerce, and especially automated negotiation [1] .

[1] We are using DAML-S Service Profile in our implementation (see section 6) but for the purpose of clarity we describe the e-commerce constructs in DAML+OIL only.

We recognise that service description ontologies and domain specific ontologies will have an important role to play in order to achieve the semantic level of agreement between the various parties. For the sole purpose of the following examples, we define a simple ontology for the sale and delivery of computers. To keep the descriptions concise, we have chosen to use the description logics notation.

The description ontology: We use the `ECConstruct` class as a common superclass for `Advertisement`, `Query`, `Template` and `Proposal`. As we will see later, an agreement is not modelled as a class but as an instance. More precisely, an agreement is an instance of a particular negotiation template.

$$\begin{aligned} ECConstruct &\sqsubseteq \top \\ Advertisement &\sqsubseteq ECConstruct \\ Query &\sqsubseteq ECConstruct \\ Template &\sqsubseteq ECConstruct \\ Proposal &\sqsubseteq ECConstruct \end{aligned}$$

The sale ontology: Two services are defined in this ontology: `Sale` and `Delivery`. A `Sale` describes the sale of one `Product` through the object property, for a unit price and a quantity given by the respective datatype properties.

We have chosen to model the service of `Sale` to include the buyer and seller roles as properties. In doing so, we allow the buyer (resp. the seller) to specify who they are and who they would like to do business with.

$$\begin{aligned} Sale \sqsubseteq\ & (=1\, buyer.Participant) \sqcap \\ & (=1\, seller.Participant) \sqcap \\ & (=1\, item.Product) \sqcap \\ & (=1\, quantity.positiveInteger) \sqcap \\ & (=1\, unitPrice.nonNegInteger) \sqcap \\ & (=1\, delivery.Delivery) \\ Delivery \sqsubseteq\ & (=1\, location.Place) \sqcap \\ & (=1\, date.date) \end{aligned}$$

The PC ontology: The `PC` class is a subclass of `Product` and must have one `Processor` and one amount of `memory`.

$$\begin{aligned} PC \sqsubseteq\ & Product \sqcap \\ & (=1\, hasProcessor.Processor) \sqcap \\ & (=1\, memory.positiveInteger) \\ Processor \doteq\ & \{PentiumIII, Pentium4, Athlon\} \end{aligned}$$

The Participant ontology: Public information about prospective advertisers and negotiators is organized in an ontology, following the yellow pages model. The ontology is built from information that individuals and/or companies are requested to provide at registration time. Such information is then used at matchmaking and negotiation time to verify compatibility of advertisements and proposals. For instance a buyer requiring service provision from an ISO9001 certified company, will only be matched with advertiser that declare to have ISO9001 certification. For the purpose of the examples, we define some disjoint classes `R1`, `R2`, and `R3` that will represent participant identities.

We now give define the e-commerce constructs. The examples will use the ontologies we have just defined.

4.3.1 Advertisement

An advertisement is expressed as a DAML+OIL class defined as the boolean combination of a set of restrictions over abstract properties and datatype properties. In Description Logics terms, advertisements are expressed as T-Boxes.

The following example shows an advertisement where `R1` would like to buy some PCs. More precisely, `R1` is advertising for the `Sale` and `Delivery` service. The restrictions over the `Sale` concept are that:

- items must be PCs with at least 128 Mb of memory;
- quantity of PCs being bought will be less than 200;
- unit price must be less than 700;
- the seller must be located in Europe.

The restrictions on the `Delivery` service are the following:

- goods must be delivered before the 15/12/2001;
- goods must be delivered in Bristol.

In description logics notation, this advertisement can be written as:

$$
\begin{aligned}
Advert1 \doteq\ & Advertisement \sqcap Sale \sqcap \\
& \forall buyer.R1 \sqcap \\
& \forall seller.(\forall isLocated.Europe) \sqcap \\
& \forall item.(PC \sqcap \forall memory.over128) \sqcap \\
& \forall unitPrice.below700 \sqcap \\
& \forall quantity.below200 \sqcap \\
& \forall delivery.(Delivery \sqcap
\end{aligned}
$$

$\forall date.before20011215 \sqcap$
$\forall location.Bristol))$

As we can see from the ontology of the `Sale` service, we require both the `buyer` and the `seller` roles to be part of the information that is specified in the agreement. In the advertisement of a seller (resp. buyer), the `seller` (resp. `buyer`) property will be restricted to to be the identifier of the participant. Hence the restriction $\forall buyer.R1$ in the example.

4.3.2 Query

A Query is similar to an Advertisement. It is also a T-Box. We give an example of a Query where the seeker is looking for buyers and sellers of PCs with a processor whose speed is greater than 1,2 GHz. Because the roles are left unspecified, the matches will be done against all buyers and sellers. This way of matchmaking could lead to a many-to-many negotiation.

$$Query1 \doteq Query \sqcap Sale \sqcap \forall item.(PC \sqcap \forall hasProcessor.(\forall speed.over1200)$$

4.3.3 Negotiation Template

After matchmaking, some parties can choose to enter into negotiation to determine the exact terms of service delivery. The negotiation template represents what is in common between all parties and is the starting point for negotiation. It also serves as a guide to scope the negotiation: negotiation proposals must comply with this template. In DAML+OIL terms, they would have to be subclass of this template.

$$\begin{aligned} Template1 \doteq\ & Template \sqcap Sale \sqcap \\ & \forall item.(PC \sqcap \forall memory.256or512 \sqcap \\ & \forall hasProcessor.\{Pentium4\}) \sqcap \\ & \forall unitPrice.below700 \sqcap \\ & \forall quantity.between100and200 \sqcap \\ & \forall delivery.(\forall date.before20011215)) \end{aligned}$$

4.3.4 Negotiation Proposal

As stated above, a negotiation proposal must be a subclass of the negotiation template associated with the ongoing negotiation. We now give an example of negotiation proposal which satisfies the template `Template1`:

$$\begin{aligned} Proposal1 \doteq\ & Proposal \sqcap Template1 \\ & \forall item.(PC \sqcap \forall memory.512) \sqcap \\ & \forall unitPrice.bellow500 \sqcap \\ & \forall quantity.over150 \end{aligned}$$

4.3.5 Agreement

When a negotiation terminates with an agreement acceptable to both parties, this agreement must specify the service that is going to be exchanged in an exact and non-ambiguous manner. Hence, whereas a negotiation proposal is a T-box, an agreement must be a fully-instantiated instance of the negotiation template. For this reason, we model an agreement as an A-Box.

An agreement is then represented in RDF. The type of the RDF resource representing the agreement must be the DAML+OIL class defining the negotiation template for the particular negotiation.

5 Operations over descriptions

We now return to the operations over descriptions which we identified in section 3 as essential to support a variety of matchmaking and negotiation protocols. In this section, we present specifications of these operations, together with examples of their operation, and identify the core functionality required by a reasoner to execute them.

Matchmaking: Recall from section 3 that the matchmaking process requires a repository host to take a query or advertisement as input, and to return all advertisements which may potentially satisfy the requirements specified in the input query or advertisement. Formally, this can be specified as:

Let α be the set of all advertisements in a given advertisement repository. For a given query or advertisement, Q, the matchmaking algorithm of the repository host returns the set of all advertisements which are compatible, $matches(Q)$:

$$matches(Q) = \{A_i \in \alpha | compatible(A_i, Q)\}$$

A set of descriptions are compatible if their intersection is satisfiable:

$$compatible(D1, \ldots, Dn) \Leftrightarrow \neg(D1 \sqcap \ldots \sqcap Dn \sqsubseteq \bot)$$

For example, consider the following advertisement from a European company `R2`:

$$\begin{aligned} Advert2 \doteq\ & Advertisement \sqcap Sale \sqcap \\ & \forall seller.R2 \sqcap \\ & \forall buyer.\neg R3 \sqcap \\ & \forall item.PC \sqcap \\ & \forall quantity.over100 \sqcap \end{aligned}$$

The intersection of this advertisement with `Advert1` above is satisfiable, as `AgreementBetweenR1andR2` is an instance of both advertisements. Hence,

$$Advert1 \in matches(Advert2)$$

Validation Rule: Recall from section 3 that the negotiation host, on receiving a proposal P, must initially check that it is valid. It is valid if it is a more constrained version of the negotiation template T for this negotiation. In description logic, this means that the negotiation host must check that T subsumes P. Formally, this can be specified as:

$$valid_T(P) \Leftrightarrow P \sqsubseteq T$$

Agreement Formation: Recall from section 3 that agreement formation requires the negotiation host to identify all pairs of proposals which are compatible. Protocol specific rules are then used to determine exactly which of these pairs are used to form an agreement, and how exactly to generate the final agreement. Compatibility can be determined using the compatibility operator defined for matchmaking. Hence, the first stage of agreement formation can be specified as follows:

Let Φ be the set of all valid proposals currently registered with the negotiation host.

$$\begin{aligned} & potentialAgreements(\Phi) = \\ & \{(P_i, P_j) | compatible(P_i, P_j) \wedge i \neq j\} \end{aligned}$$

Protocol validation and protocol-specific aspects of agreement formation are beyond the scope of this discussion. For a full discussion of these operations, together with a rule-based approach to standardising them, see [17]. When an agreement is formed, it can be verified a posteriori that the agreement subsumes the proposals that were used to form it and therefore the original negotiation template.

Note that only two atomic operations are required to define the operations specified above:

- satisfiability ($\neg(X \sqsubseteq \bot)$)
- subsumption ($X \sqsubseteq Y$).

A standard description logics reasoner is able to carry out both of these. Satisfiability lies at the core of such a reasoner, as all other reasoning or inference techniques are transformed into satisfiability checks. The subsumption operator is already defined by the DAML+OIL `subClassOf`, because our service descriptions are expressed as DAML+OIL classes (i.e. description logics concepts). A description logics reasoner can check whether two concepts subsume each other [20]. Hence, a description logics reasoner provides a good platform to implement the operations required in the B2B e-commerce interaction lifecycle.

6 Implementation

We have a prototype system based on the Jade [25] agent platform for the matchmaking and negotiation phases. There are four types of agents in this system: Seeker, Provider, Negotiation Host and Negotiation Participant.

The negotiation rules are processed by the Jess rule-engine [26]. To see some examples of rules in Jess format for the English Auction or the Continuous Double Auction, see [17].

To implement the operations presented in section 5 for the validation rule, the improvement rule and the agreement formation rule, we are using the Racer [21] reasoner system. Racer supports almost all of the DAML+OIL description logic and is one of the few reasoners to provide some support for concrete domains. Racer currently supports integers and rationals, which covers most of our needs.

We have chosen to implement the improvement rule using the description logic reasoner rather than the rule engine. To this end, we have added a construct which is internal to the negotiation host, the Validation Template. For a negotiation in a given state, the Validation Template is the intersection of the negotiation template with generated class expressions representing improvement rule constraints. Because the state of the negotiation evolves with time (for instance the value of the current highest price), the negotiation host must generate a validation template each time a new proposal is received. By checking that the Validation Template subsumes the incoming proposal, the negotiation host checks both rules at the same time: the validation rule and

the improvement rule.

In this implementation, we have chosen to use DAML-S Service Profiles as the basis to represent the e-commerce constructs presented above. DAML-S [24] is a DAML+OIL service description ontology. It aims at facilitating discovery, execution, interoperation, composition and execution monitoring of web services. DAML-S defines the notion of Service Profile (what the service does), Service Model (how the service works) and Service Grounding (how to use the service). In this work, we are only concerned by the fact that a service is represented by input and output properties of the Service Profile. Since DAML-S is in DAML+OIL it is possible to apply property restrictions on the service parameters in a similar way than in the examples presented above. We model the participants with DAML-S Actors and use the more general `providedBy` and `requestedBy` properties (instead of `seller` and `buyer`).

In terms of performance requirements, the matchmaking and negotiation phases are very different. To find a match for a particular advertisement, the reasoner needs to check the satisfiability of the intersection of the advertisement with each advertisement that has been previously submitted. With the dataset we have (around 100 advertisements and queries), the time spent by the reasoner is barely noticeable. We need to design a way to automatically generate large amounts of meaningful data to put the system under a bigger load and study its behaviour. For negotiation, the number of descriptions to manipulate is function of the number of participants in the negotiation. Compared to matchmaking, the negotiation phase uses few but more complex descriptions (which current reasoners can handle).

7 Related work

Work on service description for use in matchmaking is an important part of developing open agent-based systems. However, work on developing such description languages [27–30] has focussed on their application in matchmaking and brokering, ignoring the potential role of negotiation. Matchmaking is not used to locate potential trade partners, but rather to determine the functionality of another agent prior to execution. As a result of this, agents advertise exact specifications of their service (with some small amount of flexibility left to the discretion of the service user). This works for a cooperative community of agents, but will not work for a competitive environment such as in e-commerce. Instead, we treat the advertisement as an invitation to trade, and so it will be less constrained. Additionally, we define appropriate operations to support the negotiation phase, to refine an advertisement to a final agreement, and use the same service description language throughout this process.

DAML-S [24] and the Web Service Modeling Framework (WSMF) [31] are general approaches to modelling electronic services. Their scope is not limited to B2B e-commerce, as they define general mechanisms to enrich Web Services with declarative semantic information. It is important to note that DAML-S and WSMF are not incompatible with our approach. In fact, both DAML-S and WSMF are good candidate technologies for implementing our framework. In particular, WSMF provides a mediation architecture that would allow trading partners to interact even when they use heterogeneous ontologies or product catalogs. As mentioned in the previous section, we have used some DAML-S concepts in our implementation. However, these standards are still at the early stages and support tools are not mature or non-existent.

8 Conclusions and Future Work

In this paper, we have presented an analysis of the B2B e-commerce interaction lifecycle in terms of roles, information constructs and operations necessary to carry out the interactions. We have argued that a variety of protocols can be used for matchmaking and negotiation, but that the same information constructs and operations can be used to support them all. For this reason, we advocate standardization of these constructs and operations, as opposed to standardization on a single protocol. We have assessed the requirements on an appropriate service description language for this, and have argued that DAML+OIL meets these requirements. We have shown how DAML+OIL can be used to represent advertisements, queries, negotiation templates, proposals and agreements. Furthermore, and more importantly, we have shown that the key operations necessary to support B2B interactions can be expressed in terms of satisfiability and subsumption - two operations which existing Description Logics reasoners are capable of executing. Hence, the Semantic Web provides an ideal framework for the standardization of the B2B e-commerce interaction lifecycle.

Research on automation of negotiation requires the ability to assess the likely utility of a given advertisement or negotiation proposal. In our service description language, such proposals and advertisements can be complex structures. Up to now, most work on negotiation has assumed that only one parameter (usually price) is being negotiated. Some work has been carried out on multi-attribute negotiation (e.g. [32]) but this assumes a relatively simple utility model. If we are to be able to assign utilities to complex proposals, then research on tools to help people assess the value of different proposals (preference extraction) will be necessary. It will also be necessary to represent the relative utilities of a space of possible proposals. The application of Multi-Attribute Utility Theory to negotiation [33] is a promising approach to do this. We are currently working on ways of extending this work to assign utilities to complex

service descriptions.

In this paper, we have not addressed the operations involved in moving from the matchmaking phase to the negotiation phase. If only one matchmaking query is made, and only one advertisement selected, then this process is straightforward: the negotiation template is taken to be the intersection between the query and the advertisement. However, if many queries are made and many advertisements are matched, then the problem becomes more complex. Clusters of potentially compatible participants must be formed, together with appropriate negotiation templates. We hope to explore this issue in the future.

References

[1] N. R. Jennings, P. Faratin, M. J. Johnson, T. J. Norman, P. O'Brien, M. E. Wiegand, Agent-based business process management, International Journal of Cooperative Information Systems 5 (2&3) (1996) 105–130.

[2] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, W. R. Swarton, Enabling technology for knowledge sharing, AI Magazine 12 (3) (1991) 36–56.

[3] T. R. Gruber, Ontolingua: A mechanism to support portable ontologies, Tech. rep., Knowledge Systems Laboratory, Stanford University, Stanford, United States (1992).

[4] M. R. Genesereth, Knowledge Interchange Format. Principles of Knowledge Representation and Reasoning, in: Proceedings of the Second International Conference, Cambridge, MA, Morgan Kaufmann, 1991, pp. 599–600.

[5] T. Finin, R. Fritzson, D. McKay, R. McEntire, KQML as an Agent Communication Language, in: Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94), ACM Press, Gaithersburg, Maryland, 1994, pp. 456–463.

[6] FIPA, The Foundation for Intelligent Physical Agents. Interaction Protocol Library Specification (2000).

[7] UDDI, Universal Description Discovery and Integration. Technical White Paper (2000).

[8] D. Nickull, B. Eisenberg, ebXML technical architecture specification, Tech. rep., UN/CEFACT (2000).

[9] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, WSDL. Web Services Description Language (2000).

[10] D. Trastour, C. Bartolini, J. González-Castillo, A semantic web approach to service description for matchmaking of services, in: Proceedings of the International Semantic Web Working Symposium (SWWS), 2001.

[11] Joint US/EU ad hoc Agent Markup Language Committee, DAML+OIL (March 2001), http://www.daml.org (2001).

[12] C. Preist, S. Pearson, An adaptive choice of messaging protocol in multi-agent systems, in: Proceedings of the Third International Conference on Multi Agent Systems, 1998.

[13] K. Decker, K. Sycara, M. Williamson, Middle-agents for the internet, in: Proceedings of the 15th International Joint Conference on Artificial Intelligence, Nagoya, Japan, 1997.

[14] R. G. Smith, The Contract Net protocol: High-level communication and control in a distributed problem solver, in: Proceedings of the 1st International Conference on Distributed Computing Systems, IEEE Computer Society, Washington, DC, 1979, pp. 186–192.

[15] S. Parsons, C. Sierra, N. Jennings, Agents that reason and negotiate by arguing, Journal of Logic and Computation 8 (3) (1998) 261–292.

[16] M. P. W. Peter R. Wurman, W. E. Walsh, A parametrization of the auction design space, Games and Economic Behavior 35 (2001) 304–338.

[17] C. Bartolini, C. Preist, N. R. Jennings, Architecting for reuse: A software framework for automated negotiation, in: Proceedings of the Third International Workshop on Agent-Oriented Software Engineering (AOSE-2002), 2002.

[18] O. Lassila, R. Swick, Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation (1999).

[19] J. Z. Pan, I. Horrocks, Semantic web ontology reasoning in the $\mathcal{SHOQ}(\mathbf{D_n})$ description logic, in: Proceedings of the 2002 Description Logic Workshop (DL 2002), 2002.

[20] I. Horrocks, P. F. Patel-Schneider, Comparing subsumption optimizations, in: E. Franconi, G. De Giacomo, R. M. MacGregor, W. Nutt, C. A. Welty, F. Sebastiani (Eds.), Collected Papers from the International Description Logics Workshop (DL'98), CEUR, 1998, pp. 90–94.

[21] V. Haarslev, R. Möller, Description of the RACER system and its applications, in: Proceedings International Workshop on Description Logics (DL-2001), 2001.

[22] I. Horrocks, FaCT and iFaCT, in: P. Lambrix, A. Borgida, M. Lenzerini, R. Möller, P. Patel-Schneider (Eds.), Proceedings of the International Workshop on Description Logics (DL'99), 1999, pp. 133–135.

[23] S. Bechhofer, I. Horrocks, C. Goble, R. Stevens, OilEd: a reason-able ontology editor for the semantic web, in: Working Notes of the 2001 Int. Description Logics Workshop (DL-2001), 2001, pp. 1–9.

[24] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. L. Martin, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, H. Zeng, DAML-S: Semantic Markup for Web Services, in: Proceedings of the International Semantic Web Working Symposium (SWWS), 2001.

[25] F. Bellifemmine, A. Poggi, G. Rimassa, Jade - A FIPA compliant agent framework, in: Proc. 4th International Conference on Practical Applications of Intelligent Agents and Multi-Agent Systems, 1999.

[26] E. J. Friedman-Hill, Jess, the expert system shell for the Java platform, Tech. rep., Sandia National Laboratories, Livermore, CA (2001).

[27] K. P. Sycara, M. Klusch, S. Widoff, J. Lu, Dynamic service matchmaking among agents in open information environments, SIGMOD Record 28 (1) (1999) 47–53.

[28] Y. Arens, C. Hsu, C. A. Knoblock, Query processing in the SIMS information mediator, in: A. Tate (Ed.), Advanced Planning Technology, AAAI Press, Menlo Park, California, 1996, pp. 61–69.

[29] D. Kuokka, L. Harada, On using KQML for matchmaking, in: V. Lesser (Ed.), Proceedings of the First International Conference on Multi–Agent Systems, MIT Press, San Francisco, CA, 1995, pp. 239–245.

[30] M. H. Nodine, J. Fowler, B. Perry, Active information gathering in InfoSleuth, in: Proceedings of the International Symposium on Cooperative Database Systems for Advanced Applications CODAS, 1999, pp. 15–26.

[31] D. Fensel, C. Bussler, A. Maedche, Semantic Web enabled Web Services, in: Proceedings of the First International Semantic Web Conference, 2002.

[32] P. Faratin, C. Sierra, N. R. Jennings, Negotiation decision functions for autonomous agents, Int. Journal of Robotics and Autonomous Systems 24 (3-4) (1998) 159–182.

[33] M. Barbuceanu, M. S. Fox, Cool: A language for describing coordination in multiagent systems, in: Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, CA, 1995, pp. 17–24.