Towards automated procurement via agent-aware negotiation support[*]

A. Giovannucci, J. A. Rodríguez-Aguilar
IIIA-CSIC, Campus UAB
08193 Bellaterra, Barcelona, Spain
{andrea,jar}@iiia.csic.es

A. Reyes, F. X. Noria, J. Cerquides
Intelligent Software Components, S. A.
08190 Sant Cugat del Vallès, Barcelona, Spain
{toni,fxn,cerquide}@isoco.com

Abstract

Negotiation events in industrial procurement involving multiple, highly customisable goods pose serious challenges to buying agents when trying to determine the best set of providing agents' offers. Typically, a buying agent's decision involves a large variety of constraints that may involve attributes of a very same item as well as attributes of multiple items. In this paper we describe iBundler*, an agent-aware negotiation service to solve the winner determination problem considering buyers' and providers' constraints and preferences.*

1. Introduction

Consider the problem faced by a buying agent when negotiating with providing agents. In a negotiation event involving multiple, highly customisable goods, buying agents need to express relations and constraints between attributes of different items. Moreover, it is common practice to buy different quantities of the very same product from different providing agents, either for safety reasons or because offer aggregation is needed to cope with high-volume demands. This introduces the need to express business constraints on providing agents and the contracts they may have assigned. Not forgetting the provider side, providing agents may also wish to impose constraints or conditions over their offers. These may only be valid if certain configurable attributes (e.g. quantity, delivery days) fall within some intervals, or assembly and packing constraints need to be considered. Once a buying agent collects all offers, he is faced with the burden of determining the winning offers. The problem is essentially an extension of the combinatorial auction (CA) problem, which can be proved to be NP-complete[14]. It would be desirable to relieve buying agents from solving such a problem. In this paper we have tried to make headway in this direction by fully describing *iBundler* (extended the work in [13]),an agent-aware decision support service acting as a combinatorial negotiation solver (solving the winner determination problem) for both multi-item, multi-unit negotiations and auctions. Thus, the service can be employed by both buying agents and auctioneers in combinatorial negotiations and combinatorial reverse auctions[15] respectively. Furthermore, it extends current CA models by accommodating both operational constraints and attribute-value constraints. At this aim, new ontological issues needed to be considered in order to empower the expressiveness offered by negotiation objects and offers to incorporate buyers' and providers' business constraints. Therefore, our approach required the extension of state-of-the-art ontologies for automated negotiation. To the best of our knowledge, *iBundler* represents the first agent-aware service for multi-item negotiations[1], since agent services have mostly focused on infrastructure issues related to negotiation protocols and ontologies.

The paper is organised as follows. Section 2 introduces the market scenario where buyers and traders are to negotiate, along with the requirements, preferences, and constraints they may need to express. Next, a formal model of the problem faced by the buyer (auctioneer) based on the description in section 2 is provided. Thereafter, section 4 details the computational realisation of the agent service as an agency. Finally, section 5 summarises our contributions.

2. Market requirements

While in direct auctions, the items to be sold are physically concrete (they do not allow configuration), in a negotiation involving highly customisable goods, buyers need to express relations and constraints between attributes of different items. On the other hand, multiple sourcing is common practice, either for safety reasons or because offer aggregation is needed to cope with high-volume demands. This introduces the need to express constraints on providers and on the contracts they may be awarded. Not forgetting

∗ Partially supported by project Web-i(2) (TIC-2003-08763-C02-01).

1 Awarded the Best Application prize of the 2003 Agentcities Technology Competition (http://www.agentcities.org).

the provider side, providers may also impose constraints or conditions over their offers.

Consider a buyer intending to buy 200 chairs (any colour/model is fine) for the opening of a new restaurant, and at that aim we employ an e-procurement solution that launches a reverse auction. If we employ a state-of-the-art CA solver, a possible resolution might be to buy 199 chairs from provider A and 1 chair from provider B, simply because it is 0.1% cheaper and it was not possible to specify that in case of buying from more than one provider a minimum of 20 chairs purchase is required. On the other hand the optimum solution might tell us to buy 50 blue chairs from provider A and 50 pink chairs from provider B. Why? Because although we had no preference over the chairs' colour, we could not specify that regarding the colour chosen all chairs must be of the same colour. Although simple, this example shows that without modelling natural constraints, solutions obtained are seen as mathematically optimal, but unrealistic.

Next, we identify the capabilities required by buyers in the above-outlined negotiation scenario to express their preferences and constraints:

Negotiate over multiple items. A negotiation event is usually started with the preparation of a request for quotation (RFQ) form, which details the requirements (including attribute values as well as drawings and technical documentation) for the list of requested items (goods or services).

Offer aggregation. An RFQ item can be multiply sourced (acquired from several providers), either because not a single provider can satisfy the whole demand or because of buyers' explicit constraints (see below).

Business sharing constraints. Buyers might be interested to restrict the number of providers that may have each RFQ item awarded, either for security or strategical reasons. It is also common practice to constraint the contract volume a single provider may gain per item.

Constraints over single items. Every RFQ item is described by a list of negotiable attributes. Since: a) there exists a degree of flexibility in specifying each of these attributes (e.g. several values are acceptable); and b) multiple offers referring the very same item can be finally accepted; buyers need to impose constraints over attribute values. For instance, say that the deadline for the reception of item A is 2 weeks. Although items may arrive any given day within 2 weeks, once the first units arrive, the rest of units might be required to arrive no more than 3 days later.

Constraints over multiple items. In daily industrial procurement, accepting certain configuration for one item might affect the configuration of a different item (e.g. to ensure compatibility between products). Hence, buyers need to express constraints and relationship between attributes of different RFQ items.

Specification of providers' capacities. Buyers cannot risk to award contracts to providers beyond their capabilities. At this aim, they must require to have providers' capacities per item declared.

Analogously, next we detail the expressiveness of the bidding language required by providers:

Multiple bids per item. Providers might be interested in offering alternate conditions/configurations for the very same good, i.e., offering alternatives for a same request. A common situation is to offer volume-based discounts. This means that a provider submits several offers and each offer only applies for a minimum (maximum) number of units.

Combinatorial offers. Economy efficiency is enhanced if providers are allowed to offer (bid on) combination of goods. They might lower the price, or improve service assets if they achieve to get more business.

Multi-unit offering. Each provider requires to specify his willingness to sell over/below a minimum/maximum number of units.

Homogeneous combinatorial offers. Combinatorial offering may produce inefficiencies when combined with multi-unit offering. Thus a provider may wind up with an award of a small number of units for a certain item, and a large number of units for a different item, being both part of the very same offer (e.g. 10 chairs and 200 tables). It is desirable for providers to be able to specify homogeneity with respect to the number of units for complementary items.

Packing constraints. It is often not possible to serve an arbitrary number of units (e.g. a provider cannot sell 27 units because his items come in 25-unit packages). Thus, providers require to specify their packing sizes.

Complementary and exclusive offers. Providers usually submit XOR bids, i.e., exclusive offers that cannot be simultaneously accepted. Also, they may wish to indicate that an offer is selected only if another offer is also selected. This type of bidding, hereafter referred to as AND bids, allows to express volume-based discounts (e.g. first 1000 units at \$2.5 p.u. and then \$2 each).

Although many more constraints might be considered, we believe these do address well the nature of the problem.

3. Formal model

In this section we provide a formal model of the problem faced by the buyer (auctioneer) based on the description in section 2. But before, some definitions are in place.

[Items] The buyer (auctioneer) has a vector of items $\Lambda = \langle \lambda_1, \ldots, \lambda_m \rangle$ that he wishes to obtain. He specifies how many units of each item he wants $U = \langle u_1, \ldots, u_m \rangle, u_i \in I\!R^+$. He also specifies the minimum percentage of units of each item $M = \langle m_1, \ldots, m_m \rangle, m_i \in [0,1]$, and the maximum percentage of units of each item $\bar{M} = \langle \bar{m}_1, \ldots, \bar{m}_m \rangle, \bar{m}_i \in [0,1], \bar{m}_i \geq m_i$, that can be allocated to a single seller. Furthermore, he specifies the minimum number of sellers $S = \langle s_1, \ldots, s_m \rangle, s_i \in I\!N$, and the maximum number of sellers $\bar{S} = \langle \bar{s}_1, \ldots, \bar{s}_m \rangle, \bar{s}_i \in I\!N, \bar{s}_i \geq s_i$, that can have simultaneously allocated each item. Finally, a tuple of weights $W = \langle w_1, \ldots, w_m \rangle, 0 \leq w_i \leq 1$, contains the degree of importance assigned by the buyer to each item.

[Item attributes] Given an item $\lambda_i \in \Lambda$, let $\langle a_{i_1}, \ldots, a_{i_k} \rangle$ denote its attributes.

[Sellers' capacities] Let $\Pi = \langle \pi_1, \ldots, \pi_r \rangle$ be a tuple of providers. Given a provider $\pi_i \in \Pi$ the tuple $C^i = \langle c_1^i, \ldots, c_m^i \rangle$ stands for the minimum capacity of the seller, namely the minimum number of units of each item that the seller is capable of serving. Analogously, the tuple $\bar{C}^i = \langle \bar{c}_1^i, \ldots, \bar{c}_m^i \rangle$ stands for the maximum capacity of the seller, i.e. the maximum number of units of each item that the seller is capable of providing.

[Bid] The providers in Π submit a tuple of bids $B = \langle B^1, \ldots, B^n \rangle$. A bid is a tuple $B^j = \langle \Delta^j, P^j, M^j, \bar{M}^j, D^j \rangle$ where $\Delta^j = \langle \Delta_1^j, \ldots, \Delta_m^j \rangle$ are tuples of bid values per item, where $\Delta_i^j = \langle \delta_{i_1}^j, \ldots, \delta_{i_k}^j \rangle \in I\!R^k, 1 \leq i \leq m$, assigns values to the attributes of item λ_i; $P^j = \langle p_1^j, \ldots, p_m^j \rangle, p_i^j \in I\!R^+$, are the unitary prices per item; $M^j = \langle m_1^j, \ldots, m_m^j \rangle, m_i^j \in I\!R^+$, is the minimum number of units per item offered by the bid; $\bar{M}^j = \langle \bar{m}_1^j, \ldots, \bar{m}_m^j \rangle, \bar{m}_i^j I\!R^+, \bar{m}_i^j \geq m_i^j$, is the maximum number of units of each item offered by the bid; and $D^j = \langle d_1^j, \ldots, d_m^j \rangle$ are the *bucket* or *batch* increments in units for each item ranging from the minimum number of units offered up to the maximum number of units. Given a bid $B^j \in B$, we say that B^j does not offer item $\lambda_i \in \Lambda$ iff $m_i^j = \bar{m}_i^j = 0$.

In order to model homogeneity constraints, we define a function $h : B \to 2^\Lambda$. Given a bid $B^j \in B$, $h(B^j) = \{\lambda_{j_1}, \ldots, \lambda_{j_k}\}$ indicates that the bid is homogeneous with respect to the items in $h(B^j)$. In other words, if the buyer (auctioneer) picks up bid B^j, the number of units allocated for the items in $h(B^j)$ must be the same.

Furthermore, in order to relate sellers to their bids we define function $\rho : \Pi \times B \to \{0, 1\}$ such that $\rho(\pi_i, B^j) = 1$ indicates that seller π_i is the owner of bid B^j. This function satisfies the following properties: $\forall B^j \in B \;\; \exists \pi_i \in \Pi$ such that $\rho(\pi_i, B^j) = 1$; and given a bid $B^j \in B$ if $\exists \pi_i, \pi_k \in \Pi$ such that $\rho(\pi_i, B^j) = 1$ and $\rho(\pi_k, B^j) = 1$ then $\pi_i = \pi_k$. These conditions impose that each bid belongs to a single seller.

[XOR bids] Let $xor : 2^B \to \{0, 1\}$ be a function that defines whether a subset of bids must be considered as an XOR bid. Only bids owned by the very same seller can be part of an XOR bid. More formally $xor(\mathcal{B}) = 1 \Leftrightarrow \exists!\, \pi \in \Pi$ such that $\rho(\pi, B^i) = \rho(\pi, B^j) = 1 \; \forall B^i, B^j \in \mathcal{B}, B^i \neq B^j$. Thus, f.i. if $\exists B^j, B^k \in B \; xor(\{B^j, B^k\}) = 1$ both bids are mutually exclusive, and thus cannot be simultaneously selected by the buyer.

[AND bids] Let $and : \cup_{i=1}^n B^i \to \{0, 1\}$ be a function that defines whether an ordered tuple of bids must be considered as an AND bid. Thus, given an ordered tuple of bids $\langle B^{j_1}, \ldots, B^{j_k} \rangle$ such that $and(\langle B^{j_1} \ldots B^{j_k} \rangle) = 1$ then the buyer can only select a bid $B^{j_i}, 1 < i \geq k$, whenever $B^{j_1}, \ldots, B^{j_{i-1}}$ are also selected. Furthermore, all bids in an AND bid belong to the very same seller. Put formally, $and(\mathcal{B}) = 1 \Leftrightarrow \exists!\, \pi \in \Pi$ such that $\rho(\pi, B^i) = \rho(\pi, B^j) = 1 \; \forall B^i, B^j \in \mathcal{B}, B^i \neq B^j$. AND bids are intended to provide the means for the buyer to express volume-based discounts. However, they should be regarded as a generalisation of the bidding via price-quantity graphs in [9].

Based on the definitions above we can formally introduce the decision problem to be solved to provide support to the buyer (auctioneer):

[Multi-attribute, multi-unit combinatorial reverse auction] The multi-attribute, multi-unit combinatorial reverse auction winner determination problem (MMCRAWDP) accounts to the maximisation of $\sum_{j=1}^n y_j \cdot \sum_{i=1}^m w_i \cdot V_i(q_i^j, p_i^j, \Delta_i^j)$ subject to the following constraints:

1. $q_i^j \in 0 \cup [m_i^j, \bar{m}_i^j]$. This constraint forces that when bid B^j is selected as a winning bid, the allocated number of units of each item q_i^j has to fit between the minimum and maximum number of units offered by the seller.

2. $q_i^j \bmod d_i^j = 0$. The number of allocated units q_i^j to a bid B^j for item λ_i must be a multiple of the batch d_i^j specified by the bid.

3. $\sum_{j=1}^n q_i^j = u_i$ (there is no free disposal). The total number of units allocated for each item must equal the number of units requested by the buyer.

4. $\forall \pi_k \in \Pi \;\; q_i^j \cdot \rho(\pi_k, B^j) \in \{0\} \cup [c_i^k, \bar{c}_i^k]$. For each item, the number of units allocated to a seller cannot exceed his capacities.

5. $\forall \pi_k \in \Pi \;\; q_i^j \cdot \rho(\pi_k, B^j) \in \{0\} \cup [m_i \cdot u_i, \bar{m}_i \cdot u_i]$. The total number of units allocated per seller cannot exceed or be below the maximum and minimum percentages that can be allocated per seller specified by the buyer.

6. $\forall \lambda_{j_t}, t \in h(B^j) \;\; q_i^j = q_t^j$. For homogeneous bids, the number of units allocated to the items declared homogeneous must be the same.

7. $\forall \lambda_i \in \Lambda \;\; \sum_{k=1}^r x_i^k \in [s_i, \bar{s}_i]$. The number of sellers to be awarded each item cannot exceed or be below the maximum and minimum number of total sellers specified by the buyer.

8. $and(\langle B^{j_1}, \ldots, B^{j_k} \rangle) = 1 \Rightarrow y^{j_1} \geq \ldots \geq y^{j_k}$. Bids being part of an AND bid can only be selected if the bids preceding them in the AND bid are selected too.

9. $\forall B' \subseteq B$ such that $xor(B') = 1 \;\; \sum_{B^j \in B'} y^j \leq 1$. XOR bids cannot be jointly selected.

10. $a \cdot v_{i,l} + b \geq \delta_{i,l}^j \geq a' \cdot v_{i,l} + b'$ where $a, b, a', b' \in I\!R$. Intra-item constraints are modelled through this expression.

It indicates that only those bids whose value for the attribute item related to the decision variable that satisfy the expression can be selected.

11. $c \cdot v_{i,l} + d \geq v_{j,k} \geq c' \cdot v_{i,l} + d'$ where $c, d, c', d' \in \mathbb{R}$. Inter-item constraints are modelled through this expression. It puts into relation decision variables of attributes belonging to different items.

where

- $y_j \in \{0,1\}, 1 \leq j \leq n$, are decision variables for the bids in B;
- $x_i^k \in \{0,1\}, 1 \leq i \leq m, 1 \leq k \leq r$, are decision variables to decide whether seller π_k is selected for item λ_i;
- $q_i^j \in \mathbb{N} \cup \{0\}, 1 \leq j \leq n, 1 \leq i \leq m$, are decision variables on the number of units to select from B^j for item λ_i;
- $V_i : \mathbb{R}^+ \times \mathbb{R}^+ \times \mathbb{R}^{i_k} \rightarrow \mathbb{R}, 1 \leq i \leq m$, are the bid valuation functions for each item; and
- $v_{i,l}$ stands for a decision variable for the value of attribute a_l of item λ_i.
- $a, b, a', b', c, c', d, d'$ are buyer-defined constant values.

There are several aspects that make our model differ from related work. Firstly, traditionally all CA models assume that the buyer (auctioneer) equally prefers all items. Such constraint is not considered in our model, allowing the buyer to express his preferences over items. Secondly, multi-attribute auctions and CAs with side constraints have been separately dealt with. On the one hand, Bichler [2] extensively deals with multi-attribute auctions, including a rich bidding language. On the other hand, Sandholm et al. [15] focus on multi-item, multi-unit CAs with side constraints where items are not multi-attribute. We have attempted at formulating a model which unites both. Lastly, to the best of our knowledge neither inter-item nor intra-item constraints have been dealt with in the literature at the attribute level [9] though they help us better cope with multiple sourcing scenarios.

4. Agent-aware negotiation service

4.1. Winner determination

Since the CA problem is known to be NP-complete, solving methods are of crucial importance. In general terms, we identify three main approaches in the literature to fight this complexity. Firstly, attempts to make the CA design problem tractable through specific restrictions on the bidding mechanism have taken the approach of considering specialised structures that are amenable to analysis(e.g. [10]). But such restrictions violate the principle of allowing arbitrary bidding, and thus may lead to reductions in the economic outcome. A second approach sacrifices optimality by employing approximate algorithms (e.g. [8]). However, and because of the intended, actual-world usage of our service, it is difficult to accept the notion of sub-optimality. A third approach consists in employing an exact or complete algorithm that guarantees the global optimal solution if this exists. Although theoretically impractical, the fact is that effective complete algorithms for the CA problem have been developed. Many of the works reviewed in the literature adopt global optimal algorithms as a solution to the CA because of the drawbacks pointed out for incomplete methods. Basically two approaches have been followed: traditional Operations Research (OR) algorithms, and new problem specific algorithms (e.g. [16]). The fact is that the CA problem is an instance of the multi-dimensional knapsack problem (MDKP) [7]), a mixed integer program well studied in OR. In fact, our formulation of the problem can be regarded as similar to the binary multi-unit combinatorial reverse auction winner determination problem in [15] with side constraints[9]. Besides, expressing the problem as a mixed integer programming problem with side constraints enables its resolution by standard algorithms and commercially available, thoroughly optimised software.

With these considerations in mind, the core of our service has been modelled and implemented as a mixed integer programming problem: a version using ILOG CPLEX 7.1 in combination with SOLVER 5.2; and another version using using iSOCO's Java MIP modeller —that integrates GLPK (http://www.gnu.org/directory/GNU/glpk.html). In both cases it takes the shape of a software component. Hereafter we shall refer to this component as the *iBundler* solver.

4.2. Architecture

The *iBundler* service has been implemented as an agency composed of agents and software components that cooperatively interact to offer a negotiation support service. *iBundler* can act as a combinatorial negotiation solver for both multi-item, multi-unit negotiations and auctions. Thus, the service can be employed by both buyers and auctioneers in CAs. Figure 1 depicts the components of the agency, along with the fundamental connections of buyers and providers with the service. Next we make explicit the main functionality of its members:

[Logger agent]. It represents the interface of the *iBundler* agency to the world. The Logger agent is in charge of facilitating registration and deregistration with the *iBundler* service to users (both buyers and providers) as well as their subsequent access to the service via log in and log out.

[Manager agent]. Agent devoted to providing the solution of the problem of choosing the set of bids that best matches a user's requirements. There exists a single Manager agent per user (buyer or auctioneer), created by the Logger agent, offering the following services: brokering service to for-

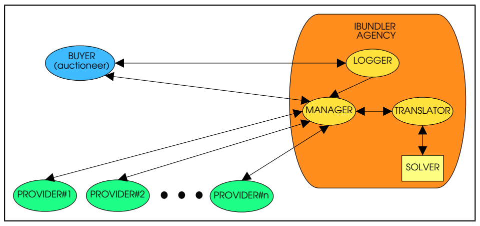

Figure 1. Architecture of the *iBundler* agency

ward buyers' requirements (RFQs) to selected providers capable of fulfilling them; collection of bids; winner determination in a combinatorial negotiation/auction; and award of contracts on behalf of buyers. Furthermore, the manager agent is also responsible for: bundling each RFQ and its bids into a negotiation problem in FIPA-compliant (http://www.fipa.org) format to be conveyed to the Translator agent; and to extract the solution to the negotiation problem handled back by the Translator agent. Observe that figure 1 shows the interplay of buying and providing agents with the Manager as the sole access point to *iBundler*.

[Translator agent]. It creates an XML document representing the negotiation problem in a format understandable by the Solver departing from the FIPA-compliant description received from the Manager. It also translates the solution returned by the Solver into an object of the ontology employed by user agents. It is the bridge between the language spoken by user agents and the language spoken by Solver.

[Solver component]. The *iBundler* component itself extended with the offering of an XML language for expressing offers, constraints, and requirements. The XML specification is parsed into an MIP formulation and solved using available MIP solvers as described above.

Our design manages to separate concerns among the three members of the agency. On the one hand, the Manager is strictly devoted to coordination. It represents the façade of the service. Besides, since every negotiation requested by a buyer makes the agency create an instance of the Manager, the service can cope with scalability issues. Thus, if the service is heavily accessed, Managers can synchronise to queue tasks for the Translator. This is in charge of relieving both Managers and Solver from the burden of translating FIPA-compliant specifications into the XML language required by Solver. We pursued to have Solver exclusively dedicated to handle computationally expensive negotiation problems as it is. Notice too that Solver has been implemented as a software component because it was intended to serve for two purposes: as the core component of the *iBundler* agency, and as the winner determination component in a commercial sourcing application[12].

To implement the *iBundler* agency we used the following technologies: JADE [1] as the software tool to implement agents, and as the platform where the agency resides (connected to the Agentcities network as a node); Tomcat 4.1 (http://jakarta.apache.org/tomcat/) as J2EE server to build web interfaces for human traders; and FIPA in building agents, messages, ontology and protocols.

4.3. Interaction Protocol

Figure 2 depicts the interaction protocol involved in the interplay of buyers and provides with *iBundler*. It is expressed in AUML (Agent Unified Modeling Language)[11] following the FIPA interaction protocol library specification compiled in [5]. Observe that the specification in figure 2 involves four roles, namely: *buyer*, *manager*, *translator*, and *provider*. Whereas multiple agents can act as providers, the remaining roles can be uniquely adopted by a single agent each. Notice too that the *iBundler* interaction protocol is composed of several interleaved interaction protocols, namely:

[IP-RFQ] Held between a buyer and the manager agent created by the Logger agent after registration. The buyer delivers an RFQ to his manager agent requesting to obtain the optimal set of offers from the available providers.

[IP-CFP] Prior to delivering the optimal set of offers, the manager interacts with the available providers to request their offers under the rules of this CFP interaction protocol. If no offers are received the manager refuses to deliver the optimal set of offers in the context of the IP-RFQ interaction protocol. Otherwise, the manager agrees on providing the service and proceeds ahead by starting out an instance of the IP-Request-Solution interaction protocol. The protocol winds up with the notification of contract awards to selected providers according to the buyer's decision. Notice that the manager mediates between the buyer and the providers.

[IP-Request Solution] This interaction protocol held between the manager and the translator agent within the *iBundler* agency aims at calculating the optimal set of offers considering the offers submitted by providers, along with the buyer's requirements and constraints. The result delivered by the translator is further conveyed by the manager to the buyer in the context of the interleaved IP-RFQ interaction protocol.

[IP-AWARD] At the end of the IP-RFQ interaction protocol the buyer obtains the optimal set of offers. Notice though that he may request also to receive all offers, as explained later on in section 4.4 when describing all agent actions. Thereafter, the buyer initiates the IP-AWARD interaction protocol in order to request the manager to award contracts to selected providers. Observe that the contract award distribution is autonomously composed by the buyer, and thus the buyer may decide to either ignore or alter the optimal set

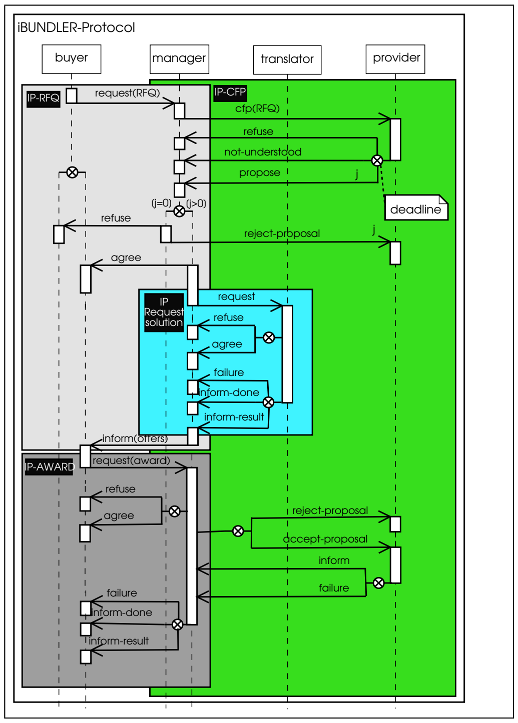

Figure 2. *iBundler* interaction protocol

of offers recommended by the *iBundler* agency. Whatever the decision conveyed to the manager, it follows the buyer's directions conveying the awards that terminate the IP-CFP held with providers. Once providers confirm their acceptances the manager acknowledges them to the buyer, ending up the IP-AWARD protocol, and the iBundler-Protocol.

4.4. Ontology

Although research on automated negotiation in multi-agent systems has concentrated on the design of negotiation protocols and their associated strategies, ontological aspects of negotiation protocols have recently started to attract researchers' attention (see [17] and the results of the ADMIT project [4]). In [17] we find an ontological approach to automated negotiation founded on the following concepts: *negotiation protocol* (rules followed by participants during a negotiation process), *party* (participants, be them either human agents, software agents or even organisations of agents), process (way to reach an agreement on some issue by modifying negotiation attributes), (negotiation) *object*, *offer* (possible combination of values associated to the negotiation attributes which represent an expression of will), *negotiation rule* (set of rules that govern a specific negotiation protocol). Although satisfactory enough for most concepts, particularly as to negotiation protocols regarded as processes and rules, in this work we had to enrich the concepts of *offer* and *object* in order to accommodate the expressiveness required by the actual-world constraints described in section 2 for bids and RFQs respectively. To the best of our knowledge, no ontology defined in prior work allows us the expressiveness that buying and providing agents require. In other words, there is no adequate ontology for multi-item, multi-unit combinatorial reverse auctions with side constraints. Thus we had to define an *ad-hoc* ontology for the *iBundler* service.

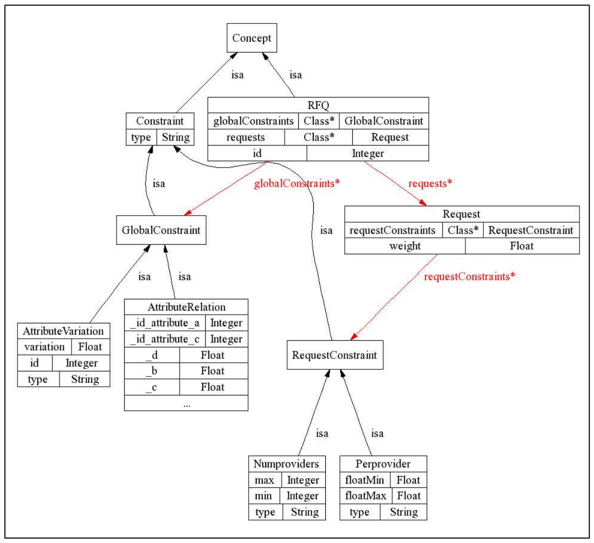

Figure 3. RFQ concept representation

The ontology has been defined with the aid of *Protege 2000* (http://protege.stanford.edu). Furthermore, the conversion from ontological objects to Java classes is realised via the *beangenerator* Protege 2000 plug-in (http://www.swi.psy.uva.nl/usr/aart/beangenerator). The automatically-generated Java classes fulfil with the JADE specification in [3]. Figures 3, 4, 5, and 6 provide graphical representations (as shown by the Ontoviz Protégé plug-in) of the core concepts in the *iBundler* ontology, namely, and respectively, the *RFQ*, *ProviderResponse*, *Problem*, and *Solution* concepts. The *RFQ* concept is employed by buying agents to express their requests for bids (via *request* in IP-RFQ). Figure 3 shows that an RFQ is composed of a sequence of *Request* concepts, one per requested item. A sequence of global constraints (*GlobalConstraint* concept) relating separate, requested items may be part of an RFQ. There are two types of *GlobalConstraint* concepts: constraints that allow to express linear relationships between different attributes of the very same or separate item(s) (*AttributeRelation* concept) and constraints on the values of an item's attribute (AttributeVariation concept). A sequence of constraints on individual items (*RequestConstraint* concept) may be also

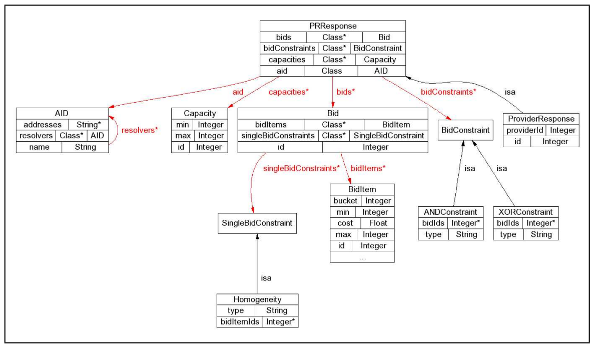

Figure 4. Bid representation as part of the provider response concept

part of an RFQ. Constraints on individual items can serve to limit the range of providers (*NumProviders* concept) to which the item can be awarded or the range of percentage of units to be awarded to the very same provider (*PerProvider* concept). Notice that all the constraints specified in an *RFQ* stand for the buyer's business rules.

On the provider side, providers express their offers in terms of the *ProviderResponse* concept (via a *propose* in IP-CFP), which in turn is composed of several elements: a list of *Bid* concepts (each *Bid* allows to express a bid per either a single requested item or a bundle of items); constraints on the production/servicing capabilities of the bidding provider (*Capacity* concept); and constraints on bundles of bids formulated with the *BidConstraint* concept (each *BidConstraint* in turn can be of exclusive –xor– or volume-based discount type –and–, corresponding respectively to the *XOR* and *AND* concepts). Whereas constraints on bundles of bids put into relation separate bids, constraints on individual bids (expressed as *SingleBidConstraint* concepts) allow to relate the values offered for separate items within the very same bid. As an example, homogeneity constraints can be declared by providers within some bid to make buyers aware that the quantity of items they can select per item must be the same, or else the provider will not concede his bid. Such constraint maps to *Homogeneity*, a particular type of *SingleBidConstraint*.

Once the manager collects all offers submitted by providers, he wraps up the *RFQ* concept as received from the buyer along with the offers as *ProviderResponse* concepts to compose the negotiation problem to be solved by the *Solver* component (via *request* in IP-Request-Solution). The resulting concept, *Problem*, is depicted in figure 5.

Finally, the solution produced by the Solver component is transformed by the translator agent into a *Solution* concept (see figure 6) that is handed over to the manager (via *inform-result* in IP-Request-Solution). The *Solution* concept contains the specification of the optimal set of offers calculated by Solver. Thus *Solution* contains a list of *SolutionPerProvider* concepts, each one containing the bids selected in the optimal bid set per provider, as a list of *BidSolution* concepts, along with the provider's agent identifier, as an *AID* concept. Each *BidSolution* in turn is composed of a list of *BidItemFixed* concepts containing the number of units selected per bid along with the bid's total cost.

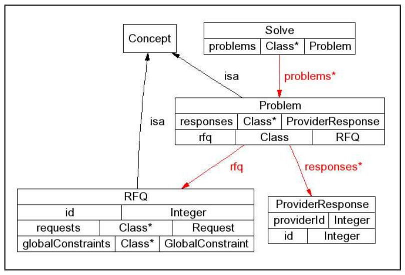
Concept
Solve
problems | Class* | Problem
isa
problems*
Problem
responses | Class* | ProviderResponse
rfq | Class | RFQ
isa
rfq
responses*
RFQ
id | Integer
requests | Class* | Request
globalConstraints | Class* | GlobalConstraint
ProviderResponse
providerId | Integer
id | Integer

Figure 5. Problem

So far we have concentrated on ontological concepts referring to entities with a complex structure that can be defined in terms of slots. Hereafter we shall draw our attention to agent actions, i.e. the special concepts that indicate actions that can be performed by agents in the *iBundler* agency, as well as buyers and providers.

Thus, the Logger agent offers the services associated to the following actions:

[Login] Action requested by trading agents when logging in with the *iBundler* agency.

[Logout] Action requested by trading agents when logging out of the *iBundler* agency.

[Register] Action requested by trading agents when signing for the *iBundler* agency. They must provide information about themselves. At the end of the registration, the Logger provides them with a username and a password.

[Unregister] Action requested by trading agents when unregistering with the *iBundler* agency. They must provide information about themselves. All the brokering information associated to them is erased by the Logger.

As to the manager, it offers four core services via:

[GetAllBids] The buyer specifies an RFQ along with a list of providers. The manager agent forwards the query to all the providers, delivering back all the responses to the buyer.

[Solve] The buyer sends to the manager an *RFQ* along with a list of *ProviderResponse* concepts representing providers' offers. The manager composes a *Problem* out of the *RFQ* and *ProviderResponses* to subsequently request the translator agent for a *Solution*. In this way, the buyer is relieved from the intricate construction of a *Problem* involving the creation of crossed references between *RFQ* and the *Bid* concepts in each *ProviderResponse*. Once the *Solution* is received by the manager it is forwarded to the buyer.

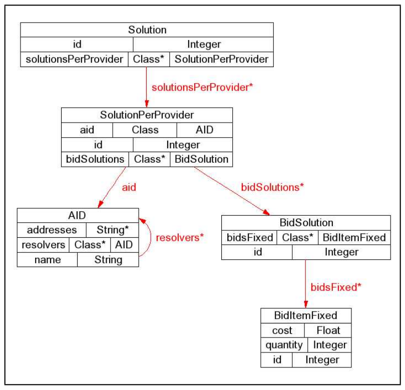

Figure 6. Solution

[Manage] The buyer sends to the manager an *RFQ* along with a list of providers. The manager sends a filtered version of the *RFQ* (removing the buyer's private constraints) to available providers, collects all their offers, constructs a *Problem* to ask the translator for a *Solution*, which is conveyed to the buyer once calculated.

[Buy] The buyer constructs a *Solution* concept and subsequently asks the manager to the bids and providers in the list of *SolutionsPerProvider*. Notice that the buyer may employ the same *Solution* concept recommended by the manager as an optimal solution.

Finally, providers do offer their services through:

[RequestForQuotations] Offer request received from the manager for a filtered version of the *RFQ* sent by the buyer.

[BuySolution] Order to buy selected offers received from the manager.

5. Contributions

This paper describes the implementation of the *iBundler* service, an innovative agent-aware decision support service for negotiation scenarios that operates as a winner determination solver for both multi-item, multi-unit negotiations and auctions. Although the current implementation is largely inspired on the interaction with professional buyers through the development of the sourcing [6] solution described in [12], it is our belief that *iBundler* is general enough to effectively empower agents to conduct from simple to largely sophisticated negotiations in open agent environments. Notice that the implementation of *iBundler* contributes along two main directions. On the one hand, we have incorporated actual-world side constraints to the winner determination problem for CAs auctions. On the other hand, we have realised a new ontology that accommodates both operational constraints and attribute-value constraints for buying and providing agents, offering a highly-expressive bidding language.

References

[1] F. Bellifemine, A. Poggi, and G. Rimassa. Developing multi-agent systems with jade. In *Intelligent Agents VII*, number 1571 in LNAI, pages 89–103. Springer-Verlag, 2001.

[2] M. Bichler and J. Kalagnanam. Bidding languages and winner determination in multi-attribute auctions. Technical Report RC22478, IBM, 2002.

[3] G. Caire. Jade tutorial. application-defined content languages and ontologies. Technical report, TILAB, 2002.

[4] A. Consortium. Demonstration documentation for checkpoint 1. Technical Report Deliverable D5.4, IST-2000-28385, August 2002.

[5] FIPA. Fipa interaction protocol library specification. Technical Report DC00025F, FIPA.

[6] A. Group. Making e-sourcing strategic: from tactical technology to core business strategy. Technical report, Aberdeen Group, 2002.

[7] R. C. Holte. Combinatorial auctions, knapsack problems, and hill-climbing search. In *Lecture Notes in Computer Science*, volume 2056. Springer-Verlag, Heidelberg.

[8] H. H. Hoos and C. Boutilier. Solving combinatorial auctions using stochastic local search. In *AAAI-2000*, 2000.

[9] J. Kalagnanam and D. C. Parkes. *Supply Chain Analysis in the eBusiness Era*, chapter Auctions, Bidding and Exchange Design. 2003. forthcoming.

[10] F. Kelly and R. Steinberg. A combinatorial auction with multiple winners for universal service. *Management Science*, 46:586–596, 2000.

[11] J. Odell, H. van Dyke Parunak, and B. Bauer. Extending uml for agents. In *Proceedings of the Agent-Oriented Information Systems Workshop*, pages 3–17. 17th National Conference on Artificial Intelligence, 2000.

[12] A. Reyes-Moro, J. A. Rodríguez-Aguilar, M. López-Sánchez, J. Cerquides, and D. Gutiérrez-Magallanes. Embedding decision support in e-sourcing tools: Quotes, a case study. *Group Decision and Negotiation*, 12:347–355, 2003.

[13] J. A. Rodríguez-Aguilar, A. Giovanucci, A. Reyes-Moro, F. X. Noria, and J. Cerquides. Agent-based decision support for actual-world procurement scenarios. In *2003 IEEE/WIC International Conference on Intelligent Agent Technology*, Halifax, Canada, October 2003.

[14] M. H. Rothkopt, A. Pekec, and R. M. Harstad. Computationally manageable combinatorial auctions. *Management Science*, 8(44):1131–11147, 1995.

[15] T. Sandholm, S. Suri, A. Gilpin, and D. Levine. Winner determination in combinatorial auction generalizations. In *First Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02)*, pages 69–76, Bologna, Italy, July 2002.

[16] T. W. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135:1–54, 2002.

[17] V. Tamma, M. Wooldridge, and I. Dickinson. An ontology for automated negotiation. In *First Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02)*, Bologna, Italy, July 2002.