Explanation Services and Request Refinement in User Friendly Semantic-Enabled B2C E-Marketplaces

Simona Colucci[1], Tommaso Di Noia[1], Eugenio Di Sciascio[1], Francesco M. Donini[2], Azzurra Ragone[1], and Raffaele Rizzi[1]

[1] Politecnico di Bari, Via Re David, 200, I-70125, Bari, Italy
{s.colucci, t.dinoia, disciascio, a.ragone}@poliba.it, raffaele@raffaelerizzi.com
[2] Università della Tuscia, via San Carlo, 32, I-01100, Viterbo, Italy
donini@unitus.it

Abstract. This paper presents an approach aimed at fully exploiting semantics of supply/demand descriptions in B2C and C2C e-marketplaces. Distinguishing aspects include logic-based explanation of request results, semantic ranking of matchmaking results, logic-based request refinement. The user interface has been designed and implemented to be immediate and simple, and it requires no knowledge of any logic principle to be fully used.

1 Introduction

Do we really need semantics in e-marketplaces? "Where's the beef" after requesting the annotation effort needed to take advantage of semantic-enriched descriptions? As it is well-known the currently most prominent e-commerce application on the web, EBay, does not use semantics, neither does Google, and both work –fine, you may add. As a matter of fact semantic web technologies open extremely interesting new scenarios, including: formalization of annotated descriptions that are machine understandable and interoperable, without being biased by usual drawbacks of natural language expressions; the possibility to reason on descriptions and infer new knowledge; the validity of the Open World Assumption, overcoming limits of structured-data models. Nevertheless there are seriuos issues that should not be underestimated: the annotation effort is considerable, though promising results are being obtained on automated extraction and ontology mapping and merging [21]; computational complexity is often demanding also for simple reasoning tasks; interaction with semantic-based systems is often cumbersome and requires skills that most end users do not have –and are not willing to learn. Moreover we believe that the effort of annotation should be rewarded with inferences smarter than purely deductive services such as classification and satisfiability, which, although extremely useful show their limits in approximate searches (see Section 3).

In this paper we show that a semantic-enabled marketplace can provide valued-added services in terms of explanations to users requests, ranking of offers and request refinement, and that use of such systems can be made easy and immediate. Main contributions of this paper include: full exploitation of non-standard inferences for explanation services in the query-retrieval-refinement loop; semantic-based ranking in the request answering; design and implementation of a completely graphical and usable interface, which requires no prior knowledge of any logic principles, though fully exploiting it in the back-office.

2 The Need for Semantics in the Matchmaking Process

We start revisiting the rationale of knowledge-based approaches to the matchmaking process between demand and supplies. Let us observe that in a generic marketplace where supplies and demands are multiattribute descriptions, all the attributes identifying the supplies as well as describing the demand, should be considered related with each other at least via some implication and disjointness relations. Considering a simple example, if a requester is looking for *a room for two persons* and in the marketplace there is an advertisement for a *double room; use of mini-bar included*, then the user should be fully satisfied by the advertisement. From the previous example we can observe that:

– Both the demand and the supply follow a product-centric structure. There is a product to be searched / advertised with some features describing it.
– The match and classification process are not simply based on features comparison after the identification of the product category[1]. If we grouped supplies (and demands) with respect to product categories `Bedroom` and `DoubleRoom` then we would not identify the equivalence between a double room and a bedroom for two persons. Also features must be taken into account during the classification process as they characterize the semantics of classification items.

If we model products and related features in an ontology $\mathcal{T}$ using a logical language, we can exploit its formal semantics during the classification and matching processes. In particular we can identify the compatibility of a supply with respect to a demand checking the satisfiability of their logical conjunction. If it is satisfiable then they are compatible. On the other hand if the information modeling a supply imply (are classified by) the one of the demand, then the latter is completely satisfied by the former. Formally, a request R (conversely a resource O) is satisfiable w.r.t. $\mathcal{T}$ if there is at least one interpretation in all the interpretations for $\mathcal{T}$ which is also an interpretation for R (conversely for O). For what concerns classification, given R and O both satisfiable w.r.t. $\mathcal{T}$, we say that O is classified by R if all the interpretations for O are also interpretations for R.

Formally, let $\mathcal{M}$ be the interpretations set for $\mathcal{T}$and $\mathcal{M}_\mathsf{R}$ the set of interpretations in $\mathcal{M}$ that satisfy the request R (respectively $\mathcal{M}_\mathsf{O}$ for the resource O). We have R (conversely O) is satisfiable if $\mathcal{M}_\mathsf{R} \not\equiv \emptyset$ and R classifies O if $\mathcal{M}_\mathsf{O} \subseteq \mathcal{M}_\mathsf{R}$.

[1] This is a typical approach in current marketplaces – see eBay among others.

Given R and O both satisfiable w.r.t. an ontology, logic based approaches to matchmaking proposed in the literature [20, 18, 19] use classification/implication and satisfiability to grade match results in five categories. We recall such a classification list:

1. Exact. $\mathcal{M}_{\mathsf{R}} = \mathcal{M}_{\mathsf{O}}$ – The demand is semantically equivalent to the supply. All the interpretations for R are also interpretations for O.
2. Full - Subsumption. The information within the supply semantically implies the one within the demand. $\mathcal{M}_{\mathsf{O}} \subseteq \mathcal{M}_{\mathsf{R}}$ – All the interpretations for O are also interpretations for R.
3. Plug-In. The information within the demand semantically imply the one within the supply. $\mathcal{M}_{\mathsf{R}} \subseteq \mathcal{M}_{\mathsf{O}}$ – All the interpretations for R are also interpretations for O.
4. Potential - Intersection. The information within the supply are semantically compatible with the one in the demand. $\mathcal{M}_{\mathsf{R}} \cap \mathcal{M}_{\mathsf{O}} \neq \emptyset$ – Some interpretations for R are also interpretations for O.
5. Partial - Disjoint. The information within the supply are semantically incompatible with the one in the demand. $\mathcal{M}_{\mathsf{R}} \cap \mathcal{M}_{\mathsf{O}} = \emptyset$ – No interpretation for R is also an interpretation for O.

2.1 Fully Exploiting Semantics

Largest part of logic-based approaches only allow, as shown before, a categorization within match types. But while exact and full matches can be rare (and basically equivalent), a user may get several potential and partial matches. Then a useful logic-based matchmaker should provide a –logic– ranking of available resources vs. the request, but what we get using classification and satisfiability is a boolean answer. Also partial matches, as pointed out in [19], might be just "near miss", *e.g.*, maybe just one requirement is in conflict, but a pure satisfiability check returns a hopeless *false* result, while it could be interesting to order "not so bad" offers according to their similarity to the request.

One may be tempted to revert to classical and well assessed Information Retrieval (IR) algorithms to get a rank for approximate matches (*e.g.*, so-called Hybrid approaches [17]), but regardless of well-known limits of unstructured text retrieval, there is something IR algorithms cannot do, while a logic approach can: provide explanations for match results and suggest revision of requests. We illustrate the rationale of such an approach by computing what is needed in order to "climb the classification list" presented above and reach a **Full** or an **Exact** match.

In particular, if we get a Partial match we could revise R relaxing some restrictions, in order to reach a Potential match. Once we get a Potential match we can hypothesize what is not specified in O in order to reach a Full match and subsequently we can suggest to the user what is not specified in the relaxed R but it is in O. The ideal match sequence should be then:

$$\textbf{Partial} \rightarrow \textbf{Potential} \rightarrow \textbf{Full} \ (\rightarrow \textbf{Exact})$$

Now consider a demand and a supply as depicted in Figure 1(b) and the ontology in Figure 1(a) representing the knowledge domain related to pictures content. Because of the disjoint relations between daffodils and tulips we have a

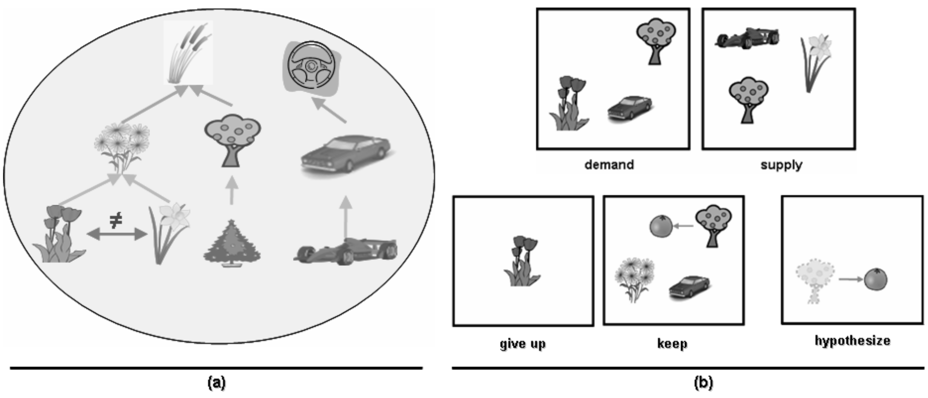


Fig. 1. (a) The reference Ontology – (b) A *demand* and a *supply* in the pictures marketplace; the contracted demand (*keep*) and the information to be given up (*give up*); features to be hypothesized in O in order to have a Full match with K(*hypothesize*)

Partial match between the demand and the supply. A contraction of the demand is needed in order to continue the matchmaking process. Notice that we have to contract less information as possible. Then in the contracted request we maintain the information about flowers and we give up only the one on tulips (see Figure 1(b)). After the contraction, a Potential match occurs between the contracted demand and the supply. We call the new contracted demand K (for Keep). With respect to O now we have $\mathcal{M}_{\mathsf{K}} \cap \mathcal{M}_{\mathsf{O}} \neq \emptyset$.

Now to reach a Full match we should reduce $\mathcal{M}_{\mathsf{O}}$. This is possible hypothesizing some unspecified characteristic H (for Hypothesis) in O such that $\mathcal{M}_{\mathsf{O}} \cap \mathcal{M}_{\mathsf{H}} \subseteq \mathcal{M}_{\mathsf{R}}$. If we hypothesize H as depicted in Figure 1(b) we obtain that the conjunction of O and H implies K, producing a Full match.

The previous example shows that some revision and hypotheses are needed in order to perform an extended matchmaking process and go through match classes.

– **Partial → Potential.** Contract R to K, giving up elements G conflicting with O: extend $\mathcal{M}_{\mathsf{R}}$ to $\mathcal{M}_{\mathsf{K}}$.
– **Potential → Full.** Make hypotheses on O adding missing characteristics H with respect to K: reduce $\mathcal{M}_{\mathsf{O}}$.

Observe that we are not asking the requester to actually go through the whole process. Yet our approach has a twofold advantage: the requester can use provided information to actually revise her request and moreover the information we extract is also all what is needed to compute a semantic distance between R and O. Once we know what has to be contracted and hypothesized it is possible to compute a match degree based on K, H, G and $\mathcal{T}$, that is what is needed in order to reach a Full match taking into account the semantics modeled in the ontology $\mathcal{T}$. In case of multiple resources, we can use this match degree as a score to rank such resources according to R. With respect to the previous example, the match degree is a function $\varphi(\mathsf{G}, \mathsf{K}, \mathsf{H}, \mathcal{T})$ combining all the causes for a non-Full match. Notice that φ needs also $\mathcal{T}$ to compute the *match_degree*; in fact in $\mathcal{T}$ the semantics of K, H, R and O are modeled, which should be taken into account when evaluating how to weigh them with respect to O and R. Before making the final step beyond, moving from a Full to an Exact match, some considerations are needed. In an *Open World* semantics, what is not specified in a formula has not to be interpreted as a constraint of absence. It is a "don't care" specifications. This can be due to basically two reasons:

– the user really does not care about the unspecified information.
– the user does not own that knowledge. She is not aware that it is possible to specify some other characteristics in the request, or she simply did not consider further possible specifications. She is not necessarily an expert of the marketplace knowledge domain.

In the second case a further refinement makes sense. A way to do this is to present to the user all the knowledge modeled in $\mathcal{T}$ and ask her to refine the query, adding characteristics found in $\mathcal{T}$. This approach has at least two main drawbacks:

1. The user must be bored browsing all $\mathcal{T}$ in order to find something interesting to be added to the request.
2. She can choose something in $\mathcal{T}$ which is not in any offer in the marketplace. Then after the query refinement she is not able to see any change in the list ranked using $\varphi(\mathsf{G}, \mathsf{K}, \mathsf{H}, \mathcal{T})$.

To avoid the above drawbacks, we might suggest to the requester *only* those characteristics able to change the ranked list of offers within the marketplace.

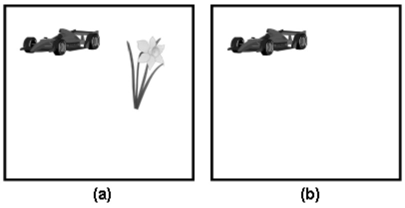


Fig. 2. Fake bonuses (a) and real bonuses (b)

Then (in an ideal marketplace where the only offer is O) we could suggest to the user to refine the contracted request adding features represented in Figure 2(a) showing B' (for Bonus) *i.e.*, what is specified in O but is not in K. But notice that in B' we have *daffodils*, that is the source of inconsistency of the original request R with the original O. Then it would be very strange if the user refined her request by adding something which is in conflict with her initial preferences. We call B' ***fake bonuses*** because of the above observation. The user is likely to refine adding at most $\mathsf{B} = \mathit{sportCar}$ (see Figure 2(b)) specification. Adding B to K the "distance" from a Full match is reduced but we do not reach an Exact match.

Full $\rightarrow$ quasi-Exact. Suggest to the requester what should be added to K looking at non requested features B (for bonus) in O –Reduce $\mathcal{M}_{\mathsf{K}}$.

In the following we will refer to Description Logics (DL) and model a DL-based framework to cope with the issues introduced here.

3 Non-standard Reasoning Services

Description Logics (DLs) are a family of logic formalisms for Knowledge Representation [2] whose basic syntax elements are *concept names*, *role names*, *individuals.*

In the following, we assume the reader be familiar with DLs syntax and semantics. DL-based systems usually provide at least two basic reasoning services:

1. *Concept Satisfiability*: $\mathcal{T} \models \mathsf{R} \not\equiv \bot$ –Given a TBox $\mathcal{T}$ and a concept R, does there exist at least one model of $\mathcal{T}$assigning a non-empty extension to R?
2. *Subsumption*: $\mathcal{T} \models \mathsf{R} \sqsubseteq \mathsf{O}$ –Given a TBox $\mathcal{T}$and two concepts R and O, is R more general than Oin any model of $\mathcal{T}$?

Matchmaking services outlined in the previous section call for other non-standard inferences we briefly recall hereafter. Let us consider concepts O and R, if their conjunction $\mathsf{O} \sqcap \mathsf{R}$ is unsatisfiable in the TBox $\mathcal{T}$ representing the ontology, *i.e.*, they are not compatible, one may want to retract specifications in R, G (for *Give up*), to obtain a concept K (for *Keep*) such that $K \sqcap \mathsf{O}$ is satisfiable in $\mathcal{T}$. In [10] the Concept Contraction problem was defined as follows:

Definition 1. *Let $\mathcal{L}$ be a DL, O, R, be two concepts in $\mathcal{L}$ and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$, where both O and R are satisfiable in $\mathcal{T}$. A* Concept ContractionProblem *(CCP), identified by $\langle \mathcal{L}, \mathsf{O}, \mathsf{R}, \mathcal{T} \rangle$, is finding a pair of concepts $\langle G, K \rangle \in \mathcal{L} \times \mathcal{L}$ such that $\mathcal{T} \models \mathsf{R} \equiv G \sqcap K$, and $K \sqcap \mathsf{O}$ is satisfiable in $\mathcal{T}$. Then K is a* contraction *of R according to O and $\mathcal{T}$.*

Obviously, there is always the trivial solution $\langle G, K \rangle = \langle \mathsf{R}, \top \rangle$ to a CCP, that is give up everything of R. On the other hand, when $\mathsf{R} \sqcap \mathsf{O}$ is satisfiable in $\mathcal{T}$, the "best" possible solution is $\langle \top, \mathsf{R} \rangle$, that is, give up nothing — if possible. Hence, a Concept Contractionproblem amounts to an extension of a satisfiable one. Since usually one wants to give up as few things as possible, some minimality in the contraction must be defined [15].

If the offered resource O and the request R are compatible with each other, *i.e.*, they *potentially match*, the partial specifications problem still holds, that is, it could be the case that O — though compatible — does not imply R. Then, it is necessary to assess what should be hypothesized (H) in O in order to completely satisfy R. In [10] the Concept Abduction problem was defined as follows:

Definition 2. *Let $\mathcal{L}$ be a DL, O, R, be two concepts in $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$, where both O and R are satisfiable in $\mathcal{T}$. A* Concept Abduction Problem *(CAP), identified by $\langle \mathcal{L}, \mathsf{R}, \mathsf{O}, \mathcal{T} \rangle$, is finding a concept $H \in \mathcal{L}$ such that $\mathcal{T} \models \mathsf{O} \sqcap H \sqsubseteq \mathsf{R}$, and moreover $\mathsf{O} \sqcap H$ is satisfiable in $\mathcal{T}$. We call H a* hypothesis *about O according to R and $\mathcal{T}$.*

Obviously the definition refers to satisfiable O and R, since R unsatisfiable implies that the CAP has no solution at all, while O unsatisfiable leads to counterintuitive results ($\neg \mathsf{R}$ would be a solution in that case). If $\mathsf{O} \sqsubseteq \mathsf{R}$ then we have

$H = \top$ as a solution to the related CAP. Hence, Concept Abduction amounts to extending subsumption. On the other hand, if $\mathsf{O} \equiv \top$ then $H \sqsubseteq \mathsf{R}$.

Intuitively, Concept Abduction and Concept Contractioncan be used for respectively subsumption and satisfiability explanation. For Concept Contraction, having two concepts whose conjunction is unsatisfiable, in the solution $\langle G, K \rangle$ to the CCP $\langle \mathcal{L}, \mathsf{R}, \mathsf{O}, \mathcal{T} \rangle$, G represents "why" $\mathsf{R} \sqcap \mathsf{O}$ are not compatible. For Concept Abduction , having R and O such that $\mathsf{O} \not\sqsubseteq \mathsf{R}$, the solution H to the CAP $\langle \mathcal{L}, \mathsf{R}, \mathsf{O}, \mathcal{T} \rangle$ represents "why" the subsumption relation does not hold. H can be interpreted as *what is specified in R and not in O*. It is intuitive that adding new constructors increases DL languages expressiveness. Nevertheless, it is a well known result [6] that this usually leads to an explosion in computational complexity of inference services. Hence a trade-off is necessary. Here we refer to the $\mathcal{ALN}$ (**A**ttributive **L**anguage with unqualified **N**umber restrictions) subset of OWL-DL.

Ontologies are usually designed as *simple-TBox* in order to express the relations among objects in the domain. With a *simple-TBox* the left side is represented by a concept name in all the axioms (for both inclusion and definition). Notice that as in $\mathcal{ALN}$ only *unqualified existential restriction* is allowed, the restriction on the *ObjectProperty* must be `<owl:Thing/>`. For $\mathcal{ALN}$ algorithms have been implemented to solve Concept Abduction and Concept ContractionProblems [8] and `MaMaS` (**Ma**tch **Ma**ker **S**ervice) reasoner[2] supports such inference services. In [19] polynomial algorithms were introduced to provide a semantic-based score for, respectively, potential and partial matches. In particular, the `rank potential` algorithm (from now on `rp` for compactness), which will be used later on, computes, given a set of $\mathcal{ALN}$axioms $\mathcal{T}$ and two $\mathcal{ALN}$concepts O and R both satisfiable in $\mathcal{T}$, a *semantic distance* of O from R with respect to the ontology $\mathcal{T}$. Notice that we write *the distance of R from* O rather then *the distance between O and R* because of the non-symmetric behavior of `rp` (see [19] for further details). Recalling the definition, $\mathtt{rp}(\mathsf{O}, \mathsf{R})$ corresponds to a numerical measure of what is still missing in O w.r.t. R. If $\mathsf{O} = \top$ we have the maximum value for $\mathtt{rp}(\mathsf{O}, \mathsf{R})$, that is the maximum (potential) mismatch of O from R. The value returned by $\mathtt{rp(}\top, \mathsf{R}\mathtt{)}$ hence amounts to how specific is a complex concept expression R with respect to an ontology $\mathcal{T}$, what we call the *depth* of R: $\mathtt{depth}(\mathsf{R})$. Such a measure is not trivially the depth of a node in a tree for at least two main reasons:

1. An $\mathcal{ALN}$ontology, typically, is not a simple terms taxonomy tree, *i.e.*, its structure is not limited to simple IS-A relations between two atomic concepts[3].
2. An $\mathcal{ALN}$complex concept is generally the conjunction of both atomic concepts and role expressions.

We remark that even though $\mathcal{ALN}$ is less expressive than the $\mathcal{SHOIN}(D+)$ supported by OWL DL, it allows the minimum set of operators needed to model requests and offers in a marketplace, in order to deal with concept taxonomy, disjoint groups, role restrictions ($\mathcal{AL}$), and number restriction ($\mathcal{N}$) to represent quantity.

[2] http://sisinflab.poliba.it/MAMAS-tng/DIG
[3] It can be better represented as a labeled oriented graph.

4 Algorithms and Their Rationale

With respect to the match classification presented in Section 2 we show how it is possible to exploit both standard and non-standard inference services for DL in order to identify match classes and go from a Partial match to a Full (or Exact) match, and use the obtained information to provide a semantic-based score measuring similarity w.r.t. the request.

Using Subsumption and Concept Satisfiability we can rewrite match classes in terms of Description Logics. Given an ontology $\mathcal{T}$and a request R and an offer O, expressed as DL complex concepts, both satisfiable w.r.t. $\mathcal{T}$, we have:

Exact: $\mathcal{T} \models \mathsf{R} \equiv \mathsf{O}$ – **Full:** $\mathcal{T} \models \mathsf{O} \sqsubseteq \mathsf{R}$ – **Plug-In:** $\mathcal{T} \models \mathsf{R} \sqsubseteq \mathsf{O}$
Potential: $\mathcal{T} \models \mathsf{R} \sqcap \mathsf{O} \not\equiv \bot$ – **Partial:** $\mathcal{T} \models \mathsf{R} \sqcap \mathsf{O} \equiv \bot$

Both Concept Abduction and Concept Contractioncan be used to suggest guidelines on what, given O, has to be revised and/or hypothesized to obtain a Full match with R.

[**Partial→Potential**] If $\mathsf{R} \sqcap \mathsf{O} \equiv \bot$ – Partial match – then solving the related Concept ContractionProblem we have $\mathsf{R} \equiv \mathsf{G}_{\mathsf{R}} \sqcap \mathsf{K}_{\mathsf{R}}$ such that $\mathsf{K}_{\mathsf{R}} \sqcap \mathsf{O} \not\equiv \bot$ w.r.t. $\mathcal{T}$. That is, we contract R to K_{R} such that there is a Potential match between the contracted request and O.

[**Potential→Full**] Once we are in a Potential match, we can formulate hypotheses on what should be hypothesized in O in order to completely satisfy the contracted R. If we solve the related Concept Abduction Problem, we can compute an hypothesis H such that $\mathsf{O} \sqcap \mathsf{H} \sqsubseteq \mathsf{K}_{\mathsf{R}}$ and reach a Full match with the contracted request.

The above concepts can be formalized in the following simple algorithm:

```
Algorithm. $retrieve(\mathsf{R}, \mathsf{O}, \mathcal{T})$
input $\mathsf{O}, \mathsf{R} \equiv \mathsf{K} \sqcap \mathsf{G}$ concepts satisfiable w.r.t. $\mathcal{T}$
output $\langle \mathsf{G}, \mathsf{H} \rangle$, i.e., the part in R that should be retracted
and the part in O that should be hypothesized to have a
full match between O and K (the contracted R)
begin algorithm
1:    if $\mathcal{T} \models \mathsf{R} \sqcap \mathsf{O} \equiv \bot$ then
2:       $\langle \mathsf{G}, \mathsf{K} \rangle = contract(\mathsf{O}, \mathsf{R}, \mathcal{T})$;
3:       $\mathsf{H}_{\mathsf{K}} = abduce(\mathsf{O}, \mathsf{K}, \mathcal{T})$;
4:       return $\langle \mathsf{G}, \mathsf{H} \rangle$;
5:    else
6:       $\mathsf{H} = abduce(\mathsf{O}, \mathsf{R}, \mathcal{T})$;
7:       return $\langle \top, \mathsf{H} \rangle$;
end algorithm
```

Notice that $\mathsf{H} = abduce(\mathsf{O}, \mathsf{R}, \mathcal{T})$ [rows 3,6] determines a concept H such that $\mathsf{O} \sqcap \mathsf{H} \sqsubseteq \mathsf{R}$, $\langle \mathsf{G}, \mathsf{K} \rangle = contract(\mathsf{O}, \mathsf{R}, \mathcal{T})$ [row 2] determines two concepts G and K such that $\mathsf{R} \equiv \mathsf{G} \sqcap \mathsf{K}$ and $\mathcal{T} \models \mathsf{K} \sqcap \mathsf{O} \not\equiv \bot$ following minimality criteria as suggested in [10, 8]. The information extracted by `retrieve` algorithm are what is needed to compute a similarity score between R and each supply. We refer here to a match degree function with values in [0..1].

$$\varphi = \left(1 - \frac{m}{N_{\mathsf{R}}}\right) \cdot \left(1 - \frac{n}{N_{\mathsf{K}}}\right) \tag{1}$$

where $N_{\mathsf{R}} = \mathtt{rp}(\top, \mathsf{R})$, $m = \mathtt{rp}(\mathsf{K}, \mathsf{R})$, $n = \mathtt{rp}(\mathsf{O}, \mathsf{K})$ and $N_{\mathsf{K}} = \mathtt{rp}(\top, \mathsf{R})$. The first factor in the previous formula represents how much of the original R has to be contracted with respect to its length and then compute a score for G; the second one gives an estimation of how many information required in K is underspecified in O.

[**Full→quasi-Exact**] Now we can try to get as close as possible to an Exact match, suggesting to the user, in a request refinement stage, what is specified in O and has not been originally requested by the user. In order to overcome the suggestion of "fake bonuses", we have to identify which part of O generated the inconsistency with R before contracting. We can solve a Concept Contraction-Problem between O and R contracting O. That is we have $\mathsf{O} \equiv \mathsf{G}_{\mathsf{O}} \sqcap \mathsf{K}_{\mathsf{O}}$ such that $\mathsf{K}_{\mathsf{O}} \sqcap \mathsf{R} \not\equiv \bot$ w.r.t. $\mathcal{T}$. In [10], among others, the conjunction minimal solution to a CAP is proposed for DLs admitting a normal form with conjunctions of concepts. A solution belonging to such solution is in the form $\mathsf{B} = \sqcap_{j=1..k} C_j$, where C_j are DL concepts and is **irreducible**, *i.e.*, B is such that for each $h \in 1, ..., k$, $\sqcap_{j=1..h-1,h+1..k} C_j$ is not a solution for the CAP. The algorithm presented in [10] and implemented in `MaMaS`, allows to compute an irreducible solution.

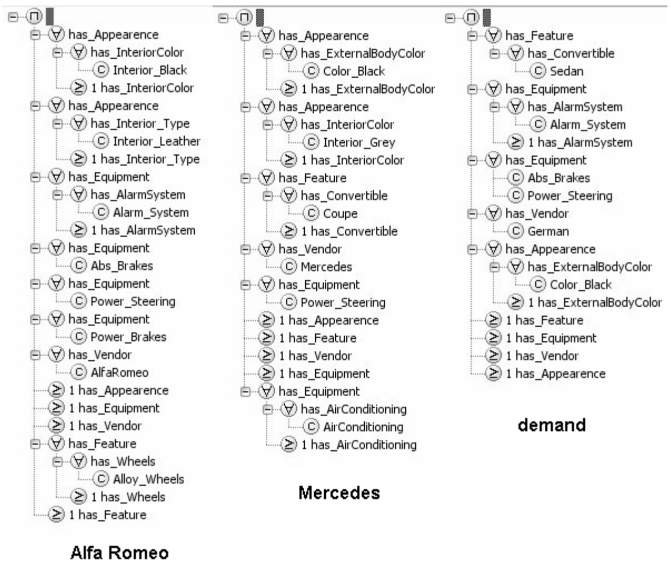
In the following the algorithm *computeBonus*$(\mathsf{O}, \mathsf{R}, \mathcal{T})$ is presented, able to compute what should be hypothesized in the requester preferences in order to get a better match result, and –if possible– an Exact match (see 2). It takes as input an offer O, a request R and the ontology $\mathcal{T}$they refer to.

```
Algorithm. computeBonus$(\mathsf{O}, \mathsf{R}, \mathcal{T})$
input O and R DL concepts both satisfiable w.r.t. $\mathcal{T}$ reference ontology
output $B_{irr}$ a set of DL concepts representing bonuses
begin algorithm
1:    $B = \emptyset$;
2:    $B_{irr} = \emptyset$;
3:    $\langle \mathsf{G}_{\mathsf{R}}, \mathsf{K}_{\mathsf{R}} \rangle = contract(\mathsf{O}, \mathsf{R}, \mathcal{T})$;
4:    $\langle \mathsf{G}_{\mathsf{O}}, \mathsf{K}_{\mathsf{O}} \rangle = contract(\mathsf{R}, \mathsf{O}, \mathcal{T})$;
5:    $\mathsf{B} = abduce(\mathsf{K}_{\mathsf{R}}, \mathsf{K}_{\mathsf{O}}, \mathcal{T})$;
6:    for each $C_j \in \mathsf{B}$
7:        $B_{irr} = B_{irr} \cup \{C_j\}$;
8:    return $B_{irr}$;
end algorithm
```

The problem of ***fake bonuses*** is taken into account in rows 3-5 of *compute-Bonus*. In row 3, a Concept ContractionProblem is solved, contracting R in K_{R} and identifying in G_{R} the source of inconsistency with O. In row 4 the same is performed for O identifying in K_{O} the part of the offer which is compatible with R and in G_{O} the incompatible one and then likely to contain the ***fake bonuses***. In row 5 we compute B, solution of Concept Abduction Problem such that $\mathsf{K}_{\mathsf{R}} \sqcap \mathsf{B} \sqsubseteq \mathsf{K}_{\mathsf{O}}$. Notice that adding B to K_{R} we are neither in a Plug-In match nor in an Exact one with respect to the contracted request K_{R}. In fact, we would have a Plug-In match if $\mathsf{K}_{\mathsf{R}} \sqcap \mathsf{B} \sqsubseteq \mathsf{O}$ rather than K_{O} and we could have an Exact match adding also ***fake bonuses*** which are isolated now in $\mathsf{G_O}$. We notice here that an obvious improvement in the definition of users preferences and query results can be obtained explicitly determining features that are mandatory to satisfy user's request and features that she considers less stringent. We hence implemented the possibility to define explicitly strict and negotiable constraints, following the approach presented in [9]. For the sake of conciseness we do not report here the modifications to the algorithms, but refer the reader to that paper.

4.1 Illustrative Example

In order to better clarify our approach we propose a simple illustrative example. Let us consider the a demand and two supplies as depicted in Figure 3[4]. Computing $retrieve(Demand, Mercedes, \mathcal{T})$ and



```
⊓
  ∀ has_Appearence
    ∀ has_InteriorColor
      C Interior_Black
    ≥ 1 has_InteriorColor
  ∀ has_Appearence
    ∀ has_Interior_Type
      C Interior_Leather
    ≥ 1 has_Interior_Type
  ∀ has_Equipment
    ∀ has_AlarmSystem
      C Alarm_System
    ≥ 1 has_AlarmSystem
  ∀ has_Equipment
    C Abs_Brakes
  ∀ has_Equipment
    C Power_Steering
  ∀ has_Equipment
    C Power_Brakes
  ∀ has_Vendor
    C AlfaRomeo
  ≥ 1 has_Appearence
  ≥ 1 has_Equipment
  ≥ 1 has_Vendor
  ∀ has_Feature
    ∀ has_Wheels
      C Alloy_Wheels
    ≥ 1 has_Wheels
  ≥ 1 has_Feature
Alfa Romeo
```

```
⊓
  ∀ has_Appearence
    ∀ has_ExternalBodyColor
      C Color_Black
    ≥ 1 has_ExternalBodyColor
  ∀ has_Appearence
    ∀ has_InteriorColor
      C Interior_Grey
    ≥ 1 has_InteriorColor
  ∀ has_Feature
    ∀ has_Convertible
      C Coupe
    ≥ 1 has_Convertible
  ∀ has_Vendor
    C Mercedes
  ∀ has_Equipment
    C Power_Steering
  ≥ 1 has_Appearence
  ≥ 1 has_Feature
  ≥ 1 has_Vendor
  ≥ 1 has_Equipment
  ∀ has_Equipment
    ∀ has_AirConditioning
      C AirConditioning
    ≥ 1 has_AirConditioning
Mercedes
```

```
⊓
  ∀ has_Feature
    ∀ has_Convertible
      C Sedan
  ∀ has_Equipment
    ∀ has_AlarmSystem
      C Alarm_System
    ≥ 1 has_AlarmSystem
  ∀ has_Equipment
    C Abs_Brakes
    C Power_Steering
  ∀ has_Vendor
    C German
  ∀ has_Appearence
    ∀ has_ExternalBodyColor
      C Color_Black
    ≥ 1 has_ExternalBodyColor
  ≥ 1 has_Feature
  ≥ 1 has_Equipment
  ≥ 1 has_Vendor
  ≥ 1 has_Appearence
demand
```

Fig. 3. The demand an supplies description used for the example

$computeBonus(Mercedes, Demand, \mathcal{T})$ we obtain results as shown in Figure 4(a) and a match $\varphi = 70\%$.

Computing $retrieve(Demand, Alfa, \mathcal{T})$ and $computeBonus(Alfa, Demand, \mathcal{T})$ we obtain results as shown in Figure 4(b) and a match degree $\varphi = 60\%$. If we refine the request adding $Power\_Brakes$ and $Interior\_Leather$ taken from the bonus of $Alfa$, we have the rank in the results list changes with a match degree of 70% for $Alfa$ and 60% for $Mercedes$.

[4] In Figure 4 and 5 the logic-based representation of the descriptions is depicted only for illustrative purpose. The GUI of the tool we present in Section 5 uses a more user friendly approach to composition and visualization of descriptions.

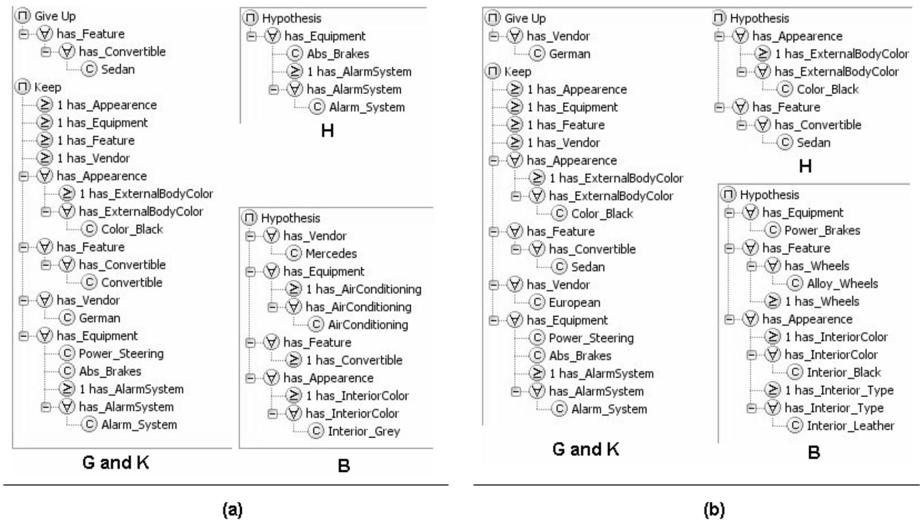


Fig. 4. (a) Matchmaking results for Mercedes - (b) Matchmaking results for Alfa

5 Prototype System

Based on the theoretical framework presented in the previous sections we developed an application fully exploiting semantic-based matchmaking procedures. The information presented within the system interface is ontology independent, *i.e.*, it is built on the fly once the reference ontology –hence the chosen marketplace domain– has been specified. Then, if the marketplace changes, the new ontology can be loaded to dynamically present to the user the new domain knowledge available and ready to be used. The GUI is divided in three main sections: in the first one (see Figure 5) the user is guided through the browsing of the ontology in order to select characteristics of the good to be searched (in other words to compose the request); in the second one (Figure 5(b)) the set of desired/undesired characteristics is shown; in the third one, a list of the offers within the marketplace is presented (see Figure 6(a)), ranked with respect to the match degree with the request. For each supply a graphical representation is presented both of the match explanation (see Figure 6(b)) and of the bonuses the user can select and add to refine the request. As shown in Figure 5(a), in order to browse the ontology an approach different from the classical tree-view is used. In the left side of the browsing panel root classes are presented. Those classes (this is the only configuration information to be provided to the tool) are the starting point for the navigation and features search. Double-clicking on a Class name, only its Subclasses are shown (in the right side of the panel). The classes navigation path is visible in the upper side of the panel. Then the user is able, whenever she wants, to go back to any level she visited previously and continue the navigation exploring other branches of the classes hierarchy. Once the user finds the characteristics she was looking for, she can drag them

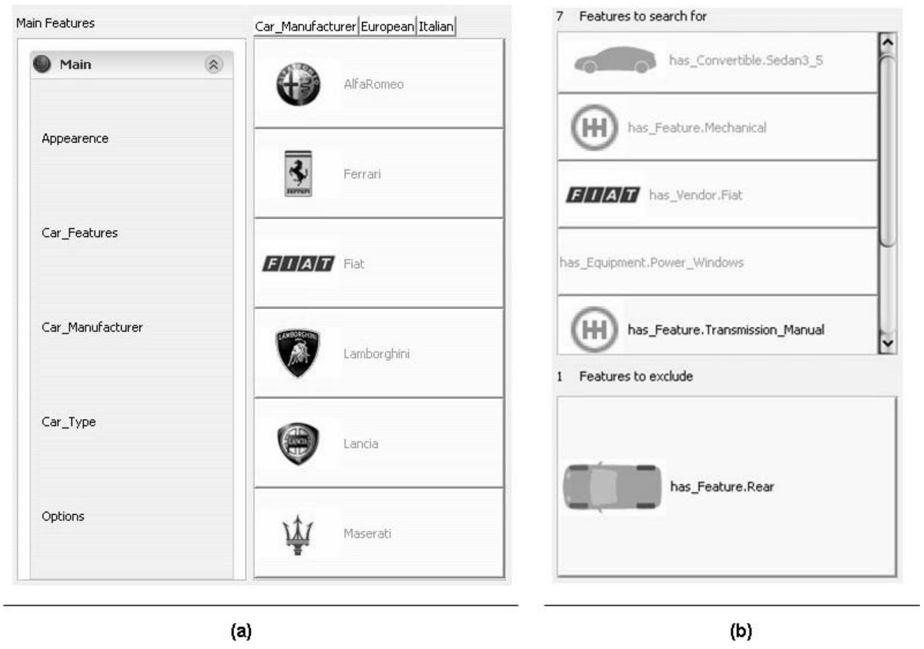
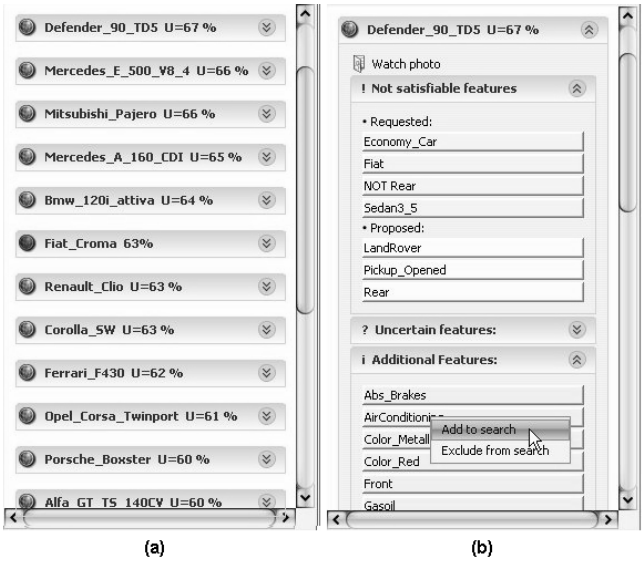


Fig. 5. (a) Ontology Browser - (b) Query Panel

in the query panel. Exploiting the domain/range relations, the query is formulated using also the ObjectProperties defined within the ontology. In the query panel, the user can express preferences both positive and negative. The panel is divided in two areas (see Figure 5(b)). In the upper one the user collects characteristics she would like to find in the resource to be matched; in the lower one characteristics explicitly not required are set. Both in the navigation and query panel, ancillary information are associated to the ontology classes and visualized ,*e.g.*, an image icon associated to each class, in order to improve the interface usability. Using `<rdfs:comment/>` it is possible to specify the image to be shown as an icon for the class. If no ancillary information are present, then only the class name, and the ObjectProperty having the class as range, are displayed (see Figure 5(b)).

The results panel (see Figure 6) is structured as a ranked list. Each item in the list represents a supply in the marketplace whose match degree is computed with respect to the demand. Each supply is visualized as a list of sub-panels, where the user is able to see:

– **Not Satisfiable Features:** what are the causes of inconsistency. In this panel both $\mathsf{K_R}$ and $\mathsf{K_O}$ (see Section 4) are shown. Then the user is able to see what in her request is in conflict with the supply (Requested) and why the conflict arises (Proposed).
– **Uncertain Features H:** what is specified in the contracted request K and is not in the supply O.


(a)

Defender_90_TD5 U=67 %
Mercedes_E_500_V8_4 U=66 %
Mitsubishi_Pajero U=66 %
Mercedes_A_160_CDI U=65 %
Bmw_120i_attiva U=64 %
Fiat_Croma 63%
Renault_Clio U=63 %
Corolla_SW U=63 %
Ferrari_F430 U=62 %
Opel_Corsa_Twinport U=61 %
Porsche_Boxster U=60 %
Alfa_GT_TS_140CV U=60 %

(b)

Defender_90_TD5 U=67 %
Watch photo
! Not satisfiable features
• Requested:
Economy_Car
Fiat
NOT Rear
Sedan3_5
• Proposed:
LandRover
Pickup_Opened
Rear
? Uncertain features:
i Additional Features:
Abs_Brakes
AirConditioni
Color_Metall
Color_Red
Front
Gasoil
Add to search
Exclude from search

Fig. 6. System Results (a) and (b)

– **Additional Features:** what is over-specified in the supply but is not requested, in other words the bonuses. This sub-panel offers an additional feature with respect to the previous two. For each bonus presented the user is able to add it to the "features to search for" or to the "features to exclude" and refine the query.

The reasoner is not embedded within the tool. This one communicates with the inference engine via a DIG 1.1 interface over HTTP. Since the tool exploits both standard and non-standard inference services as presented in Section 4, we use `MaMaS` reasoner system, which exposes a standard DIG 1.1 interface enhanced with additional tags to support the above mentioned services.

6 Related Work

Studies on matchmaking system go a long way back. For a detailed report on general matchmaking issues and systems we refer the interested reader to [19]. Here we concentrate only on relevant work in semantic matchmaking. In [13, 16] subsumption based matchmaking was initially introduced. In [20] a language, LARKS, was proposed specifically designed for agent advertisement. The matching process was carried out through five progressive stages, going from classical IR analysis of text to semantic match via Θ-subsumption. The notion, inspired by Software Engineering, of *plug-in* match was introduced to overcome in some way the limitations of a matching approach based on exact match. No ranking was devised but for what is called relaxed match, which basically reverts again to a IR free-text similarity measure. So a basic service of a semantic approach, such as inconsistency check, seems unavailable with this type of match. In [23] a matchmaking framework was proposed, which operated on service descriptions in

DAML+OIL and was based on the FaCT reasoner. An extension to the approach in [20] was proposed in [18] where two new levels for service profiles matching are introduced. Notice that there, the *intersection satisfiable* level is introduced, whose definition corresponds to the one of *potential matching* proposed in [19]. The approach presented does not introduce a ranking method to measure proximity of service descriptions. In [19] properties that a matchmaker should have in a DL based framework, were described and motivated, and algorithms to classify and rank matches into classes were presented, *i.e.*, *Full match:* all requested characteristics are available in the description examined; *Potential match:* some part of the request is not specified in the description examined; *Partial match:* some part of the request is in conflict with the description examined. The algorithms compute a semantic distance between each description w.r.t. a request in each class. In [3] the Difference Operator in DLs was proposed for matchmaking in the framework of web services The approach uses the Concept Difference, followed by a set covering operation optimized using hypergraph techniques. A DL-based system, which allows the treatment of negotiable and strict constraints has been proposed in [9]. That approach is complementary to the one proposed here, as we can extend preferences selection while maintaining a separation between strict and negotiable constraints. The need to overcome simple deductive based semantic-matchmaking is now increasingly acknowledged. Recent approaches try to tackle this isssue adopting fuzzy-DLs as in Smart [1] or hybrid approaches, as in the OWLS-MX matchmaker [17]. Such approaches, anyway, relaxing the logical constraints, do not allow any explanation or automated revision service.

7 Conclusion and Future Work

In this contribution we have presented a formal approach and a system that –in our opinion– clearly show the benefits of semantic markup of descriptions in an e-marketplace. We have presented algorithms to provide logic-based explanation services, semantic ranking of matchmaking results, and request refinement, and shown that semantic-based techniques, with their inherent advantages, can be implemented in a usable way, which does not require specific expertise to be used to their full power.

We are carrying out preliminary tests on the system, with the aid of human volunteers. The domain we selected was one of used cars. Experiments are related to evaluate both the theoretical approach and the usability of the tool. The evaluation of different match degree functions is also under investigation.

References

1. S. Agarwal and S. Lamparter. smart - a semantic matchmaking portal for electronic markets. In *Proceedings of International IEEE Conference on E-Commerce Technology*, 2005.
2. F. Baader et al. editors. *The Description Logic Handbook.* Cambridge University Press, 2002.

3. B. Benatallah et al. Request Rewriting-Based Web Service Discovery. In *Proceedings of ISWC'03*, 2003.
4. B. Benatallah et al. Semantic Reasoning for Web Services Discovery. In *Proc. of Workshop on E-Services and the Semantic Web at WWW'03*, 2003.
5. A. Borgida. Description Logics in Data Management. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):671–682, 1995.
6. R.J. Brachman and H.J. Levesque. The tractability of subsumption in frame-based description languages. In *Proceedings of the Fourth National Conference on Artificial Intelligence (AAAI-84)*, 1984.
7. S. Colucci et al. Concept Abduction and Contraction in Description Logics. In *Proceedings DL'03*, 2003.
8. S. Colucci et al. Uniform Tableaux-Based Approach to Concept Abductiona and Contraction in ALN DL. In *Proceedings of DL'04*, 2004.
9. S. Colucci et al. Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications*, 4(4):345–361, 2005.
10. T. Di Noia et al. Abductive matchmaking using description logics. In *Proceedings of IJCAI 2003*, 2003.
11. T. Di Noia et al. Semantic matchmaking in a P-2-P electronic marketplace. In *Proceedings of SAC '03*, 2003.
12. T. Di Noia et al. A system for principled Matchmaking in an electronic marketplace. In *Proceedings of WWW '03*, 2003.
13. E. Di Sciascio et al. A Knowledge-Based System for Person-to-Person E-Commerce. In *Proceedings of ADL-2001*, 2001.
14. F. M. Donini et al. Reasoning in Description Logics. In Gerhard Brewka, editor, *Principles of Knowledge Representation*, Studies in Logic, Language and Information, pages 193–238. CSLI Publications, 1996.
15. P. Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. Bradford Books, MIT Press, Cambridge, MA, 1988.
16. J. Gonzales-Castillo et al. Description Logics for Matchmaking of Services. In *Proceedings of ADL'01*, 2001.
17. M. Klusch et al. Owls-mx: Hybrid owl-s service matchmaking. In *Proceedings of AAAI Fall Symposium on Agents and the Semantic Web*, 2005.
18. L. Li and I. Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In *Proceedings of WWW '03*, 2003.
19. T. Di Noia et al. A system for principled Matchmaking in an electronic marketplace. *International Journal of Electronic Commerce*, 8(4):9–37, 2004.
20. M. Paolucci et al. Semantic Matching of Web Services Capabilities. In *Proccedings of ISWC'02*, 2002.
21. P. Shvaiko and J. Euzenat. A Survey of Schema-based Matching Approaches. *Journal on Data Semantics*, 4, 2005.
22. K. Sycara et al. LARKS: Dynamic Matchmaking Among Heterogeneus Software Agents in Cyberspace. *Autonomous agents and multi-agent systems*, 5:173–203, 2002.
23. D. Trastour et al. Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. In *Proceedings of WWW'02*, 2002.