Enabling Agent Negotiation in e-Trading Environments Through Knowledge Beads

Simon Fong
Department of Electrical and Electronic Engineering
Universidade de Macau
Av. Padre Tomás Pereira S.J., Taipa
Macau SAR
ccfong@umac.mog

Sofia Zhuang
Department of Software Engineering
Universidade de Macau
Av. Padre Tomás Pereira S.J., Taipa
Macau SAR
syz@umac.mo

Abstract – **Online trading environments such as business-to-business (B2B) e-commerce enable companies to tender projects and/or to procure goods with low costs and rapid responses. Agent-based architectures are an effective platform for such purposes because they provide mechanisms to allow companies to advertise their deals, exchange information, and perhaps to bargain over the deals before they can be closed in some kind of automated fashion. This is a classical problem of doing automated agent negotiation in which the process could be complex. In this paper, we examine the possibilities of applying an object-oriented ontology called Knowledge Beads (KB) for enabling agent negotiation and representing the knowledge collected over the negotiation process. We propose a KB-supported agent architecture and discuss its features for supporting B2B trading. In particular, we focus upon KB's use of inheritance capability as a knowledge representation medium, and show how it contributes to the fusion of proposal formation, negotiation actions and knowledge management.**

I. INTRODUCTION

In today's electronic trading environment, the concept of efficiently finding the right suppliers or buyers, at the right time, for the right deal in a streamlined procurement process is a driving goal. To achieve this goal, a company must be able to: (1) precisely define what they want to buy/sell, (2) negotiate the deal to their best interest, and yet (3) gain insight into the whole trading process as feedbacks to their knowledge management facility. Many would advocate the use of MRP or ERP software systems to achieve this objective. The drawbacks with this approach range from an enormous initial investment to regimenting dated practices as argued in [1]. We opt for a more flexible agent-based infrastructure that has functions tailor-made to suit situations where exact matches are not available.

For the first challenge, we need a flexible and dynamic mechanism for defining a non-ambiguous and meaningful bill-of-material (BOM). This BOM should be built on some ontology that can be commonly known across the participating agents. In order to tackle this problem, Knowledge Bead (KB) has been proposed for defining elements of a product or even BOM with criteria, inheritance and rules embedded within [2]. We will explain KB in the latter part of this paper.

Once we have an ontology-based foundation, namely KB, we could have a common ground for agents to express their requirements, their proposals, counter-proposals and etc. Tender specifications, bidding or proposal forms, and even their communication protocols can be built upon KB. Therefore, solutions for the second challenge, which is automated negotiation, can be made possible when they have a common ontological platform. In some primitive e-trading environment where only standard commodities are traded, we can use simple database indexing techniques and catalogues to store and display the products usually by part-numbers. However, in situations where the goods/deals being traded are complex in nature, and on which intelligent agents would have to negotiate, an ontological tool like KB is preferred.

Our objective in this paper is to ponder on how the concept of KB can be applied to the design of an agent-based online trading system that has both dynamic negotiation ability and knowledge management functionality. Achieving both agent negotiation and knowledge management is believed to be possible when the "knowledge" that is used to describe the product and the deal are commonly represented by an object-oriented ontology called Knowledge Bead [2].

We discuss the approach of applying KB from ground-up. This paper is organized in such a way that we first describe the definition of KB, then show how KB can be used to form a BOM with weights, introduce the features of KB-enabled agents, and ended with a conclusion.

II. DEFINE KNOWLEDGE BEADS WITH ONTOLOGY

Knowledge beads can be characterized as:
KB = definition + behavior + data + weight

KNOWLEDGE BEAD could be a composite object, or a simple, atomic part object in most cases; both have their own methods and data.

DEFINITION means a static unique description; this could be a UPC (Universal Product Code) or an unique index implemented at the ontology databases for referencing this KB.

BEHAVIOR is described by a set of possible actions in the use of KB engineering; some typical ones include KB formation, duplication, attributes alteration, pruning and linking to other KB. They are analogous to class functions in object-oriented programming, and are usually inherited from base classes.

DATA consist of the defined data contained in the knowledge beads.

WEIGHT is a relative weight or priority indicating how important this KB is in the BOM. At the same level under the same parent KB, the weights from all the child KB's add up to 100%, otherwise their weights would have to be normalized.

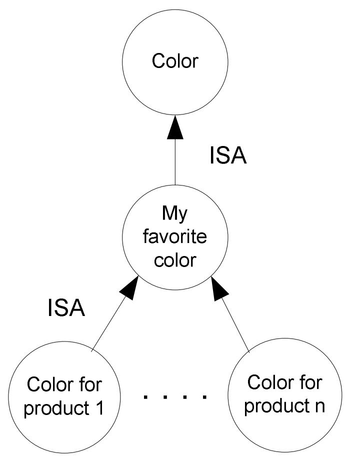


Fig. 1. The color class hierarchy diagram

Borrowing from object-oriented modeling, inheritance and hierarchy are two powerful features that KB have, for the use of building BOM. They are natural choices for expressing the meaning and construction of a BOM. For instance, inheritance has a long history in knowledge representation [3]. Below is an example of how simple rules are defined for my favorite products

2.1. Inheritance Feature

Assume that color is an attribute of a product description, and we have to set criteria (rules) based on the value of that attribute. Hence color becomes a factor that determines my requirement or preference (in a more relaxed situation). Each criterion carries a definition of my favorite color for each product in this simple example.

CRITERION (1) – The car must be RED.
CRITERION (2) – The shirt must be RED.
CRITERION (3) – The door must be RED.
CRITERION (4) – The radio must be RED.

From these criteria we can conclude that I like everything in RED colour as it is my favourite. With inheritance there is no need to change every criterion every time I change my favourite colour. By defining a root criterion for my favourite colour that will be commonly used by the other criteria, the attributes and hence the requirements will be inherited when it is referenced. This encourages consistency of definition since it is needed to be defined only once

CRITERION(0) – My favourite colour is RED.
CRITERION(1) – The car must be IN MY FAVOURITE COLOUR.
CRITERION(2) – The shirt must be IN MY FAVOURITE COLOUR.
CRITERION(3) – The door must be IN MY FAVOURITE COLOUR.
CRITERION(4) – The radio must be IN MY FAVOURITE COLOUR

We can see that such inheritance has an advantage in defining meaning and setting requirements in building BOM, as it is quite often that many of the same attributes needed to be defined. Implementing CREITERION(0) as a default in a base class, then letting other objects (or KB's in our context) to be inherited from it, realizes such advantage. When the attributes of CRITERION(0) that are defined in a base KB is called by its descendent CRITERION(1), its attributes are inherited by CRITERION(1) and similarly for the rest of the rules that made references to CRITERION(0).

2.2. Hierarchy and Weighted Criticality

In the design of KB, a product description is composed of fundamental elements that have meanings, rules and weights. When building a BOM, a hierarchy of KB's that makes up all the necessary parts of a BOM is formed. As shown in Figure 2, there is a very simplified BOM made up of a variety of KB's of products on the bill and a KB for the required delivery date. In turn each KB of product in the BOM is also formed by some KB's that serve as default attributes, such as unit price, quantity and model number. We can take a step further to generalize these product KB's into some specific product domains. For instance, in the computer industry, products are usually computers. A base class or a domain KB as what we called here would consists of all the necessary attribute KB's that are required to describe a computer. Hence we have a domain KB called "Computer" as shown in Figure 3 that can instantiate a specialized KB called "My favorite computer" to be used in a particular BOM. We advocate that different industries would be keeping their own domain KB's so to reflect the unique characteristics of their products. A central ontology library is needed to register the entire domain KB's from all the different industries, so that agents can refer to such when they are communicating among each other using KB's. In our example shown in Figure 3, the KB <My favorite computer> is inherited from a ready-to-use domain KB <Computer> that would have all the definitions of attributes for a computer. Specifically, the KB <My favorite computer> has certain criteria set for defining what it is meant to be "My favorite computer". They are such that the CPU speed must be at least 2 GHz, the chassis is of slim tower type and the brand must be Dell, for example

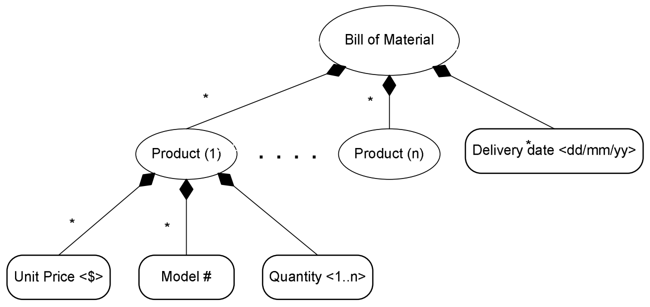


Fig. 2. A BOM composed by KB's

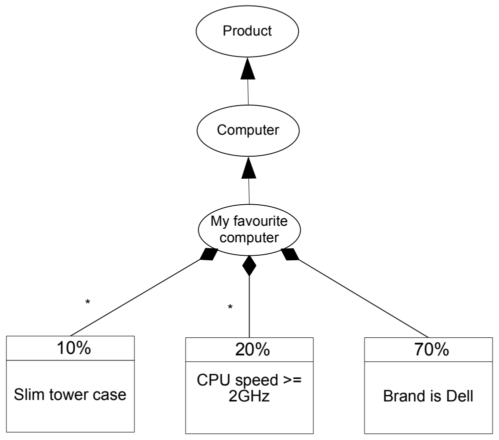


Fig. 3. Defining My-favorite-computer using KB's

Each KB when being used as leaves at the bottom of a tree for product description carries a criterion and a weight specifying how important the criterion is. With a hierarchical structure holding the KB's we can do a proper weighted criticality calculation, in order to help agents making decision in matchmaking based on the priority of each criterion.

So in Figure 3, according to the above rules which item is better: (1) a fast IBM computer in slim casing or (2) a slow computer not in slim casing but the brand is Dell? Neither one of the rules meet all the three criteria. That is, no perfect match happens in this case, as in most real-life situations. However, with weighted criticality, the preference is still on (2) - a slower and fatter sized Dell computer because the brand has the highest priority in this example. This concept reinforces the need to have hierarchy because weighted criticality cannot realize without a hierarchy and balanced weights. This approach can also be adapted in the area of probabilistic uncertain knowledge [4]. Furthermore, in most situation of matchmaking, it is likely that agents do not always get the ideal deals having all the criteria fulfilled. Weighted criticality calculation is about choosing the closest match(s) based on the preferences specified for each criterion.

2.3. Negotiating at the same points of reference

When two agents communicate they need to use the same points of references otherwise communication subjectivity and ambiguity could arise. When a buyer agent is dealing with a seller agent in the market place it is preferred that we reduce most of the ambiguous requirements in order to prevent error and to save time. An example is illustrated as follows. A buyer wants a powerful computer and it has to be the latest model. A seller has one that he thinks will meet the criteria of the buyer. They are matchmaking a deal based on abstract criteria. Without a precise definition that both agents can understand on the meaning of criteria, the negotiation process will go hay-wired and perhaps wrong agreement may result. The buyer's requirement (read-point of references) for the computer:

NEW – It has been out on the market for less than 3 months
POWERFUL – It has a Pentium CPU whose speed can go up to 3GHz.

The seller's description (read-point of references) for the computer:

NEW – It has been out on the market for 1 year
POWERFUL – It has a Xeon 1.8 GHz processor

When they meet to close a deal, chances are that the deal will not be successful though they use the same words (new and powerful) to define the criteria. The issue is that although qualitatively both the buyer and seller use the same terms, quantitatively they are defined differently. The solution then is to ask both to quantify their criteria based on the same points of reference. We can learn from this analogy that as long as the buyer and seller use the same points of reference, they will be able to effectively communicate and negotiate. What critical is that they must use the common points of references for negotiation and matchmaking. In other words, how the rules are defined for the e-trading agent will not render the negotiation process useless as long as the same conditions at the atomic level (read-points of references) are used in the building up those rules. In our implementation these points of references in the atomic level are known as knowledge beads.

2.4. Hierarchy of Inheritable Rule-based intelligence and Knowledge Beads

We have seen in the previous sections the advantages of inheritance. As pieces of information are linked, the structure must allow unrestricted relationship network. The only exception is that at the atomic level, the information must be defined using the same points of reference. These points of references shall be called knowledge beads. Knowledge beads are the basic building blocks in building the rules. In the following discussion we will show an application of these techniques on how a computer as a product is described by KB's.

III. BILL-OF-KNOWLEDGE

The illustration here is kept simple focusing on the principles of hierarchy and inheritance. Let us assume a scenario that a 'computer' is one of the products on the BOM. In e-trading environment, my Internet agent will meet other suppliers' agents online for procuring goods on my BOM. Before the orders can be confirmed, agents would have to exchange information making each other understand what is being demanded and what can be offered. Usually the product specifications, terms and rules on which negotiation is based are expressed in text during a matchmaking process regardless of the language implementation, e.g. XML/RDF/DAML [5]. In an example of text based information analysis at the atomic level are 'words' that one would find in the agent communication. These words are treated as atomic (lowest) level attributes that can be converted from knowledge beads. The process of implementing knowledge beads in RDBMS is described elsewhere.

A collection of knowledge beads is a called a bill of knowledge. The example below shows how a preferred product 'computer' is defined with the weights of my choice on each attribute. The 'computer' inherits the general attributes of 'product' that is a part of BOM. It therefore is assumed to carry default variables like delivery date, model number, etc. It may inherit also from a base KB 'colour' so that the whole computer including its parts would have to default to the colour of my choice. However, overriding the choice of colour on any attribute is possible. For example, we can have an all beige colour computer, but a black-coloured keyboard.

Through the exchange of records using predefined data formats for example XML for bill of knowledge, agents can communicate their information and criteria by building attributes upon attributes until the whole knowledge about the product is known. This differs from the approach of Knowledge Interchange Format (KIF) which necessitates the exchange of rules between agents [6]. Knowledge Beads use the approach of attribute exchange instead.

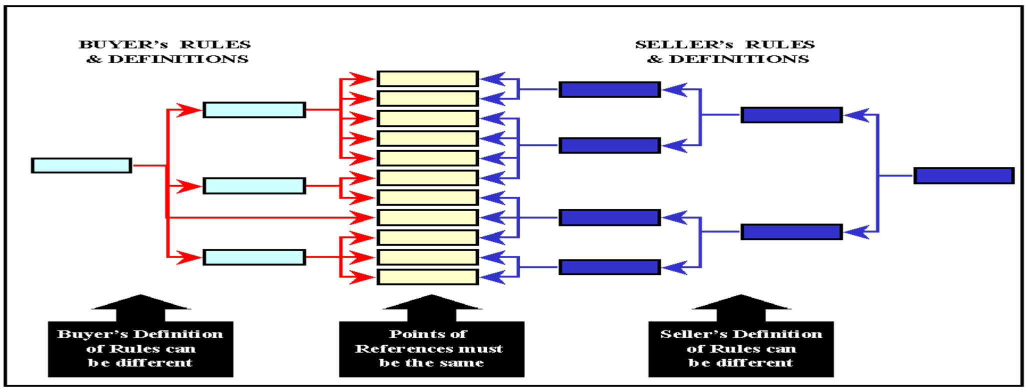
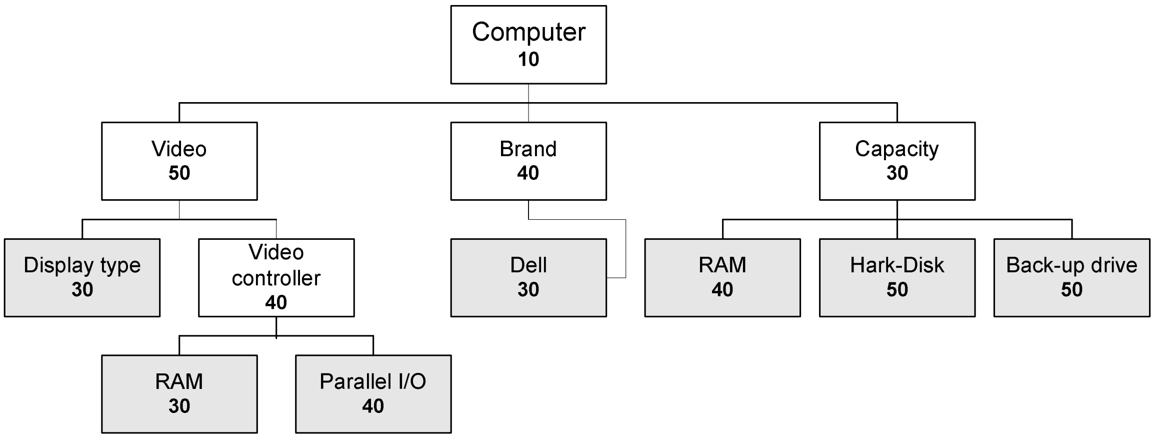

BUYER's RULES & DEFINITIONS
SELLER's RULES & DEFINITIONS
Buyer's Definition of Rules can be different
Points of References must be the same
Seller's Definition of Rules can be different

Fig. 4. Points of Reference



Exploded leaves of Bill-of-knowledge

Parent KB	Child KB	Weightage
Video	Display type	D
Video controller	RAM	F
Video controller	Parallel I/O	G
Brand	Dell	H
Capacity	RAM	I
Capacity	Hard disk	J
Capacity	Back-up drive	K

Bill-of-knowledge network links

Parent KB	Child KB	Weightage
Computer	Video	50/(50+40+30) = A
Computer	Brand	40/(50+40+30) = B
Computer	Wheels	30/(50+40+30) = C
Video	Display type	[30/(30+40)]*A = D
Video	Video controller	[40/(30+40)]*A = E
Video controller	RAM	[30/(30+40)]*E = F
Video controller	Parallel I/O	[40/(30+40)]*E = G
Brand	Dell	30*B = H
Capacity	RAM	[40/(40+50+50)]*C = I
Capacity	Hard disk	[50/(40+50+50)]*C = J
Capacity	Back-up drive	[50/(40+50+50)]*C = K

Fig. 5. Bill-of-knowledge

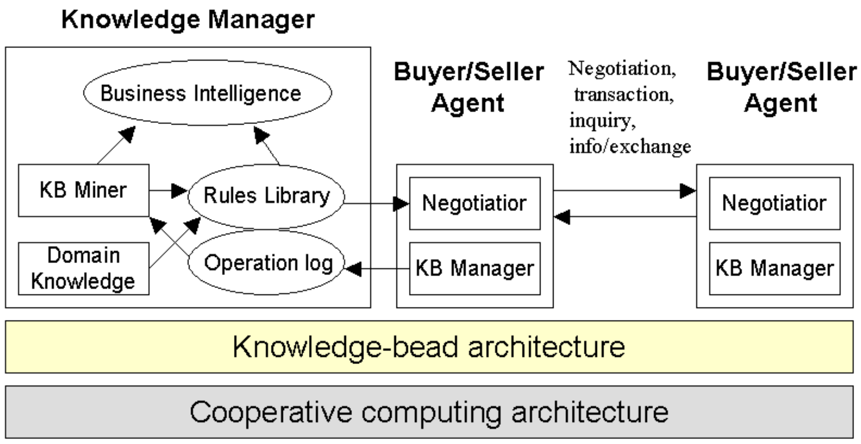


Fig. 5. The infrastructure of negotiation agents and knowledge manager

The basic principle of a bill of knowledge is that it contains both information and relationship. Because information is built one onto another, a hierarchy exists, as shown in Figure 5. With the ability to ‘bud-graft’ in the bill of knowledge, inheritance happens. Sometimes there is a need for us to use alternative rules instead of the primary rules for overriding the default inherited. Some criteria are more important thus in the bill of knowledge we must also incorporate the relative weightage for the information defined.

In real world applications, the definitions of non-leaf KB in the bill of knowledge usually are:

1. Inheritable
2. Have a hierarchy for weighted criticality
3. Have a primary attribute definition
4. Have alternate attribute definitions
5. Have a time-phased date-effective attributes definition mechanism.

These can be achieved on the knowledge beads data structure when we add fields that allow us to define primary-alternate levels and date-effective to the simple knowledge bead data structure in the illustration above. Details on how the weighted criticality is calculated can be found in [2].

IV. KB-ENABLED TRADING AGENTS

By using KB’s our proposed agent system attempts to reach beyond the simple matching of relevant attributes and price quotes. It works by enabling online traders to stipulate their business intentions in detail for complex deals. With the aid of KB, such intentions are defined by multiple items (products and services), multiple attributes, constraints, preferences, tradeoffs and the specification of negotiation mechanisms. Because the process takes place automatically on the trader's behalf, negotiations can be conducted simultaneously with tens or even hundreds of potential partners rather than with just one or two. Business will move ahead even if such partners can only contribute part of the deal (partial deals) in some incomplete situations. This flexibility leads to the best available transaction for all sides.

Achieving greater control Price is only one of the many variables that both buyer and seller can register within the KB-enabled trading environment. Each participant defines the deal content, preferences and constraints (such as delivery dates, supplier/buyer ratings) and tradeoffs permitted in a deal - in effect appointing an automatic 'Negotiating Agent' who strives to close an advantageous deal. These 'agents' work on all sides, constantly matching and adjusting buyers and sellers intentions. With control over many parameters, the dealer has the flexibility to configure negotiation set-up in the most convenient way.

All sides can benefit The agents operate over/within a common market specific language built on KB, which gives buyers and sellers the freedom to interrelate automatically:

- Traders can trade easily, free of the tedious manual interactions with other parties that can involve adjusting the deal, submitting counter offers, incorporating business rules etc.
- Buyers can shop for whole deals, scan a wide array of offerings and find the offers that best match their requirements.
- Sellers can compete on their strong points - matching products to market needs, differentiating themselves and receiving instantaneous wide exposure. This increases their sales volume.
- The agents provide a high level of flexibility and interplay on a fully automated level, which allows for broad-based, simultaneous activity.

Self-learning ability During the negotiation process, when an agent encounters some new “ways” of business rules or new elements it will learn about it and be able to apply them in the future. Such new “ways” could be a new specification of a product, strategy, constraints, rule, preference or attributes represented in KB which makes a common platform among all the agents. With an example following up from Figure 5, a seller may introduce a new attribute that the buyer was never aware of before. This will substantially change the weights and bill-of-knowledge for the buyer. For example, while the buyer agent was looking for a product with video-type SVGA, the seller has new stocks available that come with XGA with little difference in price. A KB of video-type having its value as XGA will be absorbed by the buyer agent into registering it in its domain library. Effectively, that newly registered KB of XGA could be used for procurement subsequently.

V. NEGOTIATING AGENTS AND KNOWLEDGE MANAGER

Each side appoints a 'Negotiating Agent'. These 'Agents' meet on a common ground within the KB-enabled trading environment to match the buyer and seller intentions. Their tasks include:

- Acting solely on the trader's behalf to adjust the deal content
- Carrying out the trader's negotiation strategy
- Evaluating other parties offers and reacting accordingly
- Maintaining the confidentiality of the trader's proprietary information
- Dynamically accessing various approved parties' business rules and decision data

'Agents' automatically adjust the deal content between themselves according to their permitted tradeoffs and business rules. This is the true meaning of automatic negotiation when bargaining for goods and services.

On the other hand, the data accumulated are stored in some operation log at either the seller or buyer side. This log contains valuable information regarding the deals, the preferences and the characteristics of the other agents, and the negotiating strategies used by the other agents. Through a special data miner that discovers patterns from the KB, some useful business intelligence could be generated and new rules or definition of some new products (new description of a product) can be manifested too. This new information will in turn help create or modify new rules for the negotiating agents to use. The detailed operation of such KB-enabled agent systems are described in another paper.

VI. EXPECTED SIGNIFICANCE

The most expected significance out of this research is perhaps the multiplying benefits that are anticipated from converging agent negotiation and knowledge management at the same time. Each technology has been established for some time and they are maturing over the last decades. However, either they usually lead on individual direction in a sense that they take on different grounds and be applicable to certain domains. With the research results of Knowledge Beads that serve as a common fundamental building block for the agents to use for negotiation and for the knowledge manager to interpret, there should exist opportunities and possibilities for these two separate technologies to complement each other when put into use together. For example, the data left behind the agents could be data-mined for business intelligence for knowledge management. Any new tactics or new product description learnt from the other agents could be put back into the bill-of-knowledge subsequently. This newly learnt intelligence can also be applied into the rule-based agents so to increase its level of sophistication.

By using knowledge-based agent technology, we could anticipate the following benefits:

- The levels of negotiation could be more refined, sophisticated and accurate
- More complex B2B negotiation can be possible
- Agents can learn by themselves by their own experience or from other agents via KB
- Agents make use of KB to negotiate under circumstances where no perfect match is likely; this should always give the best possible match
- In the reactive mode, good decision support is provided for users through knowledge management
- Allow Knowledge library to be expanded easily using KB that is scalable in nature
- B2B e-Commerce can be automated and yet be operated intelligently by knowledge-driven agent technology

VII. CONCLUSION

An object-oriented concept called knowledge bead has been proposed for specifying the product description in BOM. Such a concept facilitates inheritance and hierarchical data definition that yields flexibility and adaptability in e-trading agent systems. Flexibility is resulted from using KB to represent the meaning of everything under the sun. Adaptation is supported by bill-of-knowledge through which agents can easily compose and restructure the necessary ontological meanings during agent interaction. It is believed that integrating KB into some prevalent forms of B2B agent communication standards such as cascading XML and EDI is possible. An infrastructure of negotiation agents and knowledge manager has been proposed that addresses the possibilities of converging knowledge management and agent negotiation. We are hoping that this paper on KB will inspire a new generation of communication format and item definition for e-trading agents, who in turn would be more capable to facilitate sophisticated negotiation. For BOM's that consist of only simple commodities, a relational database is sufficient to store their parts numbers in the form of static identities. However, for BOM and proposals that carry composite goods (or even bundle sales), a meaningful hierarchy of definition would be needed. Counter-offer or proposals would be a matter of revising the tree of KB rather than re-specifying the whole BOM. KB is generic enough to handle this job, and can ensure that the meanings of the attributes that the agents negotiate upon are always consistent. For future works, we would mainly focus in these two directions: (1) Formulate theorems and methodology for KB and their operations where appropriate, and (2) Development of the preference profiles and the case-based negotiation algorithms. These elements will allow the agents to dynamically change their preferences and evaluation criteria in a changing world and situation, and to select among sets of known successful negotiation strategies based on the current mission constraints

VIII. REFERENCES

[1] N. Ivezic, T. Potok, L. Pouchard, "Manufacturing Multiagent Framework for Transitional Operations", *IEEE Internet Computing*, Vol.3 (5), 1999, pp.58-59.

[2] K. Chin, S. Fong, "Hierarchy of Inheritable Rule-based Intelligence and Knowledge Beads Mapping-approach for Building a Multi-purpose Intelligent Matching Agent", *Asia Pacific Web Conference '99*, Hong Kong, 27 September 1999.

[3] D. Poole, A. Mackworth, and R. Goebel "Computational Intelligence, A Logical Approach", *Oxford University Press*, 1999.

[4] D. Poole, "Context-specific approximation in probabilistic inference", *Proc. Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98)*, Madison, Wisconsin, July 1998, pp. 447-454..

[5] D. Kuokka, L. Harada, "Matchmaking for Information Integration", *Journal of Intelligent Information Systems*, 1996.

[6] B. Grosof, Y. Labrou, H. Chan, "A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML", *ACM International Conference of E-Commerce*, 1999, pp.68-77.