PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 36/05

A Multi-Agent Architecture for a Dynamic Supply Chain Management

José Alberto R. P. Sardinha
Marco S. Molinaro
Patrick M. Paranhos
Pedro M. Cunha
Ruy L. Milidiú
Carlos J. P. de Lucena

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900
RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 36/05 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena November, 2005

A Multi-Agent Architecture for a Dynamic Supply Chain Management *

José Alberto R. P. Sardinha, Marco S. Molinaro,
Patrick M. Paranhos, Pedro M. Cunha, Carlos J. P. de Lucena

sardinha@inf.puc-rio.br, milidiu@inf.puc-rio.br, lucena@inf.puc-rio.br

Abstract. Supply chain management (SCM) is a very challenging problem that is leveraging the e-commerce explosion. Today's supply chains are essentially static, because they rely on long-term relationships among key trading partners. Dynamic practices are vital because they offer better matches between suppliers and customers as market conditions change. This paper presents a flexible architecture for dealing with the next generation of SCM problems, based on a distributed multi-agent architecture of a dynamic supply chain. We define intelligent agent roles that tackle sub problems of a dynamic SCM, and present a solution for combining these results in a distributed environment. The agents' typical tasks are bid planning, production scheduling, supplier negotiation, among others. We also present an implementation of this architecture used in the international test bed for SCM solutions, the Trading Agent SCM competition, as well as some experimental results.

Keywords: Multi-Agent Systems, Supply Chain Management, Trading Systems.

Resumo. O gerenciamento da cadeia de suprimentos (*Supply Chain Mangement – SCM)* é um problema desafiador que está impulsionando o comércio eletrônico. As cadeias de suprimentos atuais são essencialmente estáticas, pois elas dependem de relacionamentos de longo prazo entre os parceiros de negócios. Práticas dinâmicas são vitais para oferecer um melhor casamento entre fornecedores e clientes toda vez que há alterações nas condições do mercado. Esse artigo apresenta uma arquitetura flexível para lidar com os problemas do gerenciamento da cadeia de suprimentos, baseado num arquitetura multi-agente para uma cadeia de suprimentos dinâmica. Papéis de agentes inteligentes são definidos para atacar subproblemas de uma cadeia dinâmica, e uma solução para combinar esses resultados é apresentado para um ambiente distribuído. Os papéis típicos são planejamento de lances, planejamento da produção, negociação com fornecedores, entre outros. Uma implementação da arquitetura também é apresentada para o ambiente do Trading Agent Competition, junto com resultados experimentais.

Palavras-chave: Sistemas Multi-Agentes, Gerenciamento da Cadeia de Suprimentos, Sistemas para Comércio.

---

* This work has been partially supported by the ESSMA Project under grant 552068/2002-0 (CNPq, Brazil).

In charge for publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3114-1516 Fax: +55 21 3114-1530
E-mail: bib-di@inf.puc-rio.br

1 Introduction

The main goal of the Supply Chain Management (SCM) is to plan and coordinate activities in a supply chain [1]. These activities may have many participants and coordination is a key aspect when efficient trading is required. Most of the supply chains in the industry are essentially static, because they rely on long-term relationships among key trading partners. The adoption of dynamic practices can offer better matches between suppliers and customers as market conditions change.

Multi-agent systems [2,3,4] is a technology used in many intelligent systems that execute in a distributed and unpredictable environment. The system's goal is achieved when software agents [4] react to events, execute strategies, interact, and participate in organizations. Supply chains are a good example of complex and open environments where agents have to deal with many unknown situations. Multi-agent systems are probably the most suitable technology for dealing with decision making strategies in such environments, because they permit an easy combination of various artificial intelligence techniques.

The primary goal of this work is to present a multi-agent architecture for a generic supply chain that uses a direct sales model [1], which links customers directly to a manufacturer through the Internet. However, we used the design of the proposed architecture to implement an agent entry for the Trading Agent Competition (TAC SCM) [5] called LearnAgentsSCM [6]. This system competed against 32 entries, and was able to classify to the quarter-finals of the 2005 competition.

The remainder of the paper is organized as follows. The second section presents related work. The third and forth section present the design of the multi-agent system and the distributed knowledge base. The following sections describes the TAC SCM game and the algorithms used in the agents. The final section presents the empirical results and concluding remarks.

2 Related Work

The TAC SCM competition encourages the development of high quality research into the trading agent and supply chain management problem. Consequently, many agent entries present [7] extremely interesting designs and algorithms for the supply chain problem.

The system RedAgent [8] also uses a multi-agent system to implement a manufacturer in a supply chain. RedAgent has a distributed architecture, and uses heuristic-based agents to deal with individual aspects of the TAC SCM game, such as component procurement and production of customer orders. The system presented an excellent result in the 2003 competition by ending up in first place. However, the proposed architecture is not generic enough for a direct sales SCM model. The system only presents the viewpoint of a manufacturer operating in the chain, and explains an internal market communication mechanism used to determine the allocation of resources.

The systems DeepMaize [9], Botticelli [10] and TacTex [11] also present very interesting designs and excellent results in the competition. These systems also present the viewpoint of a manufacturer operating in the chain, and not an overall design. The systems introduce perspectives of operations management problem in an environment defined by other operations in the chain.

There is also some literature in Operations Management that is not related to TAC SCM but deals directly with supply chains [1,12]. The main focus of this type of work is to describe the many aspects of a supply chain, and present a perspective of a central optimizer applied to: (i) a particular firm in a specific environment, or (ii) multiple (normally two) interacting participants in the chain.

This paper presents a multi-agent architecture in a dynamic supply chain, where customers interact directly to a manufacturer through the Internet. Consequently, our architecture is designed for a generic supply chain that uses a direct sales model.

3 The SCM Multi-Agent Design

A typical supply chain [1] may involve a variety of participants, such as: (i) Customers, (ii) Retailers, (iii) Wholesalers/Distributors, (iv) Manufacturers, and (iv) Component/Raw Material suppliers. The objective of every supply chain is to maximize the overall value generated, which is normally measured through profitability. Chopra and Meindl [1] advocate that all processes in a supply chain can be broken down into the following four cycles:

i. Customer order cycle – processes directly involved in receiving and filling customer's order;

ii. Replenishment cycle – processes involved in replenishing retailer inventory with a distributor;

iii. Manufacturing cycle – processes involved in replenishing distributor inventory;

iv. Procurement cycle – processes necessary to ensure that materials/components are available for manufacturing according to the schedule.

In a direct sales model, such as the one used by Dell Inc, the manufacturers fill customer orders directly. Retailers, Wholesalers and Distributors are bypassed in this type of supply chain, which ends up with only three participants – Customers, Manufacturers and Suppliers. This is probably the most dynamic practice used by the market nowadays, and this is the main reason why we decided to implement our multi-agent system based on this SCM model.

The architecture proposed uses a multi-agent approach in order to build a flexible and general design for a dynamic supply chain. Each agent can be implemented with a different AI technique, which permits a system designer to test many diverse strategies and decide the optimal combination of these techniques. The agents also use a distributed knowledge base as a key component for collaboration. Agents store results and information in the knowledge base so that other agents can use it to solve their problems. Figure 1 presents the architecture of the multi-agent system.

The main focus of the proposed design is to tackle separately important sub-problems of a supply chain: (i) procurement of components, (ii) production and delivery of finished goods, and (iii) direct sales of finished goods to customers.

The customer agents typically represent real customers and firms that are willing to buy finished goods. This agent must implement a strategy for selecting finished goods based on its preferences. This decision affects all sub-problems, but has a stronger influence in the direct sales sub-problem.

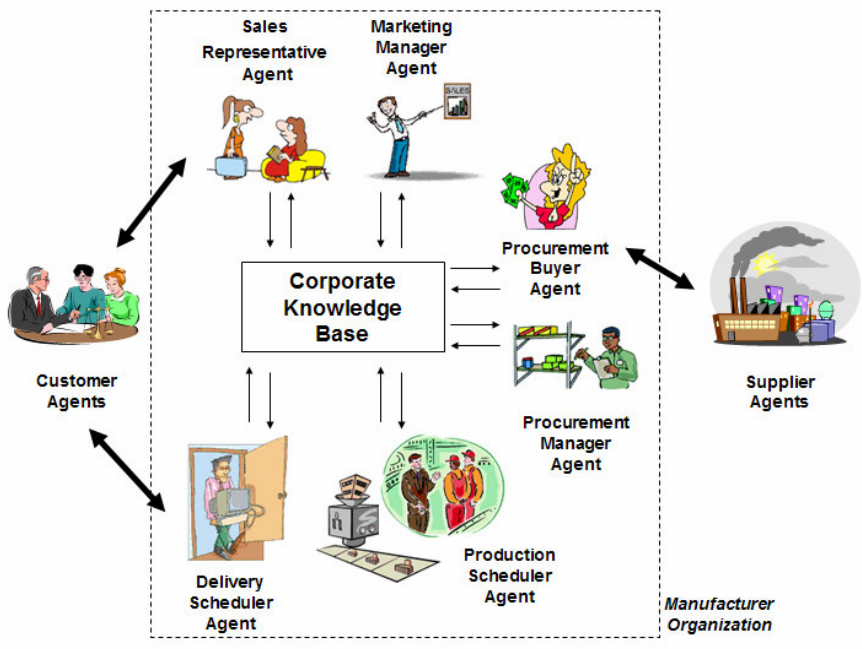

Figure 1. The Multi-Agent Architecture

The supplier agents are responsible for selling materials/components to the manufacturers, and it directly influences the procurement sub-problem. This agent is normally a manufacturer and could also use the design of the manufacturer proposed in figure 1.

The manufacturer of finished goods, also called manufacturer organization, is composed of the following agents:

- Sales Representative agent – is responsible for fulfilling customer orders. Typically, this agent has to decide the price of finished goods based on current demand and the probability of winning customer orders;
- Marketing manager agent – has to select the best customers and market segments in order to maximize the manufacturer's profitability;
- Production scheduler agent and Delivery scheduler agent – have to optimize the schedule of the production and delivery of finished goods respectively;
- Procurement buyer agent – negotiates components with supplier based on characteristics such as price and delivery time.
- Procurement manager agent – decides when to buy components based on the current inventory and component demand.

The design of the manufacturer tackles the sub-problems mentioned above with the following agents: (i) procurement of components - Procurement buyer agent and Procurement manager agent, (ii) production and delivery of finished goods - Production scheduler agent and Delivery scheduler agent, and (iii) direct sales of finished goods to customers - Sales representative agent and Marketing manager agent.

These agents use the corporate knowledge base to exchange important information for decision making purpose. This collaboration has the ultimate goal of achieving the maximum profitability for the manufacturer organization

4 The Corporate Knowledge Base

The corporate knowledge is extremely important for the agents in the manufacturer organization to collaborate and exchange important information for decision making purpose. This key component of our architecture facilitates the collaboration of agents in a distributed environment, and is implemented using OWL [13] – a Web Ontology Language.

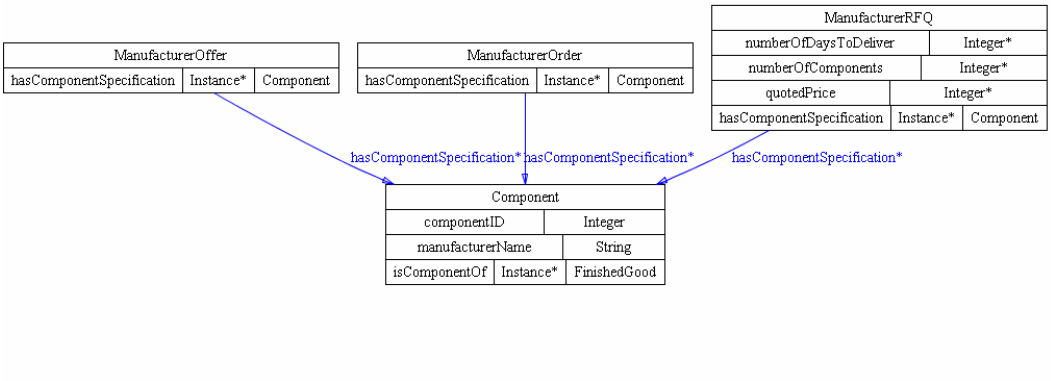

Figure 2. An example of the Corporate Knowledge Base

For brevity reasons, we will only present a part of the ontology created for the manufacturer organization. The design of this knowledge base assumes that transactions between manufacturers-customers and manufacturers-suppliers are conducted through a request for quote (RFQ) mechanism. Figure 2 presents the OWL classes ManufacturerOffer, ManufacturerOrder and ManufacturerRFQ. The object properties presented in figure 2 relates these classes to the class Component. The instances of these classes store information related to the procurement process.

In our ontology, we have also defined classes and properties for information related to the production and delivery of finished goods, and information related to the direct sales of finished goods to customers.

5 The TAC SCM Game

The TAC SCM game [5] is a simulation of a direct-sales supply chain with three main entities: Customers, Manufacturers and Suppliers. The participants of the competition have to build agents that simulate small manufacturers. Every game has 220 simulated days, and six agents compete with each other for both suppliers and customers. At the end of the game, the agent with the highest sum of money is declared the winner.

Each agent is able to produce and store 16 different PC configurations in their own production facility, by using different combinations of components. In addition, these production facilities have limited daily capacity, and each type of PC produced requires a different number of production cycles. Each agent starts with no money, no pending orders and no inventory, just with an unlimited credit line.

The TAC SCM server assumes the role of both ends of the supply chain (the suppliers and customers) using fixed strategies, fully described in [14]. Both procurement for components and sales of finished goods take place on markets common to all agents,

and transactions on both of these markets are conducted through a request for quote (RFQ) mechanism, on a daily basis.

On the sales side, customers send RFQs to all agents informing the required computer specification, the quantity of PCs, the maximum price they are willing to pay, and when products should be delivered (also called due date). Agents may respond to these offers by sending bids. The customers select the bid with lowest price from all of the bids received.

Once the offer is accepted, the agent is responsible for delivering the order according to the specified quantity and due date, and will incur in a monetary penalty for late deliveries.

Negotiations with suppliers are carried in similar fashion. Agents send requests for quotes based on a given quantity of a component and delivery date. Suppliers respond to these RFQs with information about the price and availability of components. Once the agent accepts a supplier offer, the latter tries to deliver on the accorded date.

5.1 LearnAgentsSCM – A Multi-Agent Architecture for the TAC SCM

The LearnAgentsSCM architecture uses the design of the manufacturer presented in figure 1. We decided to use this design in our 2005 entry, instead of a single agent, because we believe it facilitates the development of modular entities that are distributed and reusable. The strategies presented in this section must illustrate the modularity of our architecture.

We decided to use the TAC domain to test our architecture because it resembles a competitive SCM market, and can be used as a benchmark for evaluating the performance of our system. The strategies presented in this section are not specific for the TAC environment, and our goal is to present a more general SCM system using the proposed multi-agent design and a request for quote (RFQ) mechanism for trading goods in the supply chain.

Strategies used in the Sales Representative

The main goal of the Sales Representative Agent is to decide the bid price for every customer RFQ. Two strategies were tested in this agent. The first strategy uses the algorithm bellow to classify the customer RFQs in three different classes based on a level of interest. Consequently, a different strategy is adopted for pricing each RFQs class.

The RFQs that present a higher level of interest are also the ones that are more difficult to win. Therefore, the agent selects a lower bid price for the response. The criterion for clustering the RFQs in terms of its level of interest only uses the monetary penalty of a customer order, because we believe that orders with higher penalties also present a higher risk for the manufacturer when the resources are low.

1. Divide RFQs in 3 classes (C1, C2, C3) such that C1 contains the RFQs with lower penalties , C3 the ones with highest penalties and C2 the rest of RFQs.
2. The bid strategy of each RFQ is defined by the class that it belongs to:
   a. C1 – average of previous the highest and lowest market prices of that type of product
   b. C2 – increment factor on C1's price
   c. C3 – discount on the maximum price the client is willing to pay
3. The factor used on C3 is updated based on this class acceptance rate.

The second strategy for pricing RFQs tries to select bid prices that maximize the expected profit. We used in this strategy a simple stochastic programming model presented below. The model simplifies that actual problem by considering a group of RFQs of the same type of PC to be a single RFQ. Consequently, RFQs of a same type of PC receive the same bid price.

$$\begin{aligned} \max \quad & \sum_{i=1}^{16} w_i(p_i)\, p_i q_i \\ s.t. \quad & \sum_{i=1}^{16} w_i(p_i) q_i c_i \le C \qquad (1) \\ & q_i \le Q_i \\ & p_i \in P_i \\ & q_i \in \mathbb{N} \end{aligned}$$

where

$w_i(.)$ – the probability of winning an order of type i given its price

p_i – the price of product type i

P_i – set of quantized prices of product type i

q_i – quantity of product type i that should receive bid

c_i – number of cycles for producing a unit of product type i

C – available cycles

Q_i – quantity of product type i in current RFQs

Restriction (1) is the factory cycle constraint, where C is the estimated factory cycles available for new orders in the next few days. We also assume that orders always have enough components to be produced, justifying the absence of a component restriction in the model. The set P_i is composed of 4 prices between the previous highest and lowest market prices. Then, this model is solved through a dynamic programming approach.

After using the qualifying round to test both pricing strategies, we noticed that the second one presented better results. Hence, this strategy was adopted in the follwing rounds.

Although we tested both strategies separately, a mix of both is a possible future experiment. For instance, one could use the price from the second strategy as the base

price for the first one, applying then increment factors. This was not adopted in this version of our system, because it seamed relatively complex to implement during the competition.

Strategy used in the Marketing Manager

The main goal of the Marketing Manager agent is to decide which customer's RFQs to respond, trying to maximize the organization profitability subject to resource constraints. This agent uses a model that considers the probability of winning a bid given the price chosen by the Sales Representative. The implementation uses the concept of partial orders [8]. The decision on what RFQs to bid is based on a factory schedule for the next 8 days, including both customer orders and RFQs. This model is then solved using a greedy strategy that chooses the most profitable RFQs (in terms of its profit per factory cycles) and allocate them on the earliest date possible.

The Marketing Manager agent is also responsible for adding additional RFQs for future demand. This strategy is used when the agent believes that future market demand can reach high levels and the prediction of available factory cycles is low.

```
while there is still an available order (or RFQ) to allocate, do
        get the most profitable order
        if finished good inventory can fulfill the order, then
                mark order as allocated
                mark order to send bid
                update available inventory
        else if order can be produced by factory, then
                mark order as allocated on the first day that has
                available resources
                mark order to send bid
                update resources on that day
        else
                remove order from list
        end if
end do
```

Strategy used in the Procurement Buyer

The Procurement Buyer agent is responsible for negotiating prices with suppliers in the procurement process. RFQs are sent to all suppliers when a component is requested by the Procurement Manager. When offers are received in response to the RFQs, this agent has a fixed strategy of selecting the offer with cheapest price. The information gathered in this negotiation process is also used to estimate the replacement price for components in inventory. At the end of the game, the Procurement Manager also informs when the procurement of components must stop, and consequently, the replacement price is set to zero.

Strategy used in the Procurement Manager

The Procurement Manager agent decides the quantity and due date of each component to buy. It uses a specific strategy for each different phase of the game:

- First days: the main goal of the agent in this phase is to build a little inventory for a near future so that the factory can start its production. In the first days, a fixed quantity and due dates is used in this decision, because little information is available to calculate future demand.

- After the first days: the goal is to maintain a target inventory level. This level is ramped up in the beginning of the game to prevent the agent from buying a large amount of components in this phase. The manufacture organization has a pre-defined lifetime of one year in the TAC Game, which is clearly different from real world situations. Hence, we implemented a strategy that is specific for this scenario. When the year is finishing in the game, there is no positive effect to have any type of inventory. Therefore, we need to ramp down our target inventory level in the end of the game, trying to finish with no available inventory. A multi-resolution procurement model was then used. First pass is responsible for deciding "short due date" orders to make sure that there are enough components for the next 20 days. Second pass is responsible for "long due date" orders, taking the future inventory to the desired level. Both phases are solved through an implicit enumeration procedure that searches for due date and quantity on a reduced quantized search space. This strategy also takes into account the future demand and component prices.

Strategy used in the Production Scheduler and Delivery Scheduler

The goal of the Production Scheduler and Delivery Scheduler agents is, respectively, to plan the production of finished goods and delivery of these goods to customers.

Both production and delivery are decided at the same time by a unified model. In the current implementation, we reuse the model adopted in the Marketing Manager with some minor differences. This model does not use the calculated probabilities of winning a customer order and does not include the RFQs sent by the customers. This solver uses the same greedy strategy algorithm as the Marketing Manager agent, promoting code reuse. Once the production is completely planned, the delivery of orders is scheduled for the day after its production.

Empirical Results and Concluding Remark

The 2005 TAC SCM competition had 32 entries from different research institutions around the world, and only the 24 best agents were classified for the final rounds. As mentioned in the previous section, we implemented an entry for the 2005 TAC SCM called *LearnAgentsSCM*. The preliminary rounds of the competition presented many negative scores from agents, which indicated that a more conservative strategy could present nice results in the final rounds.

Figure 3 presents the average position of LearnAgentsSCM in the preliminary rounds compared to the highest score in the games. The x-axis represents the score of the top agent, and the y-axis depicts the average position of our agent. The two first two ranges (-0M,0-1M) clearly shows that our agent performs well when the market is extremely competitive. However, our current implementation does not excel when market conditions are favorable, and this is due to our conservative approach.

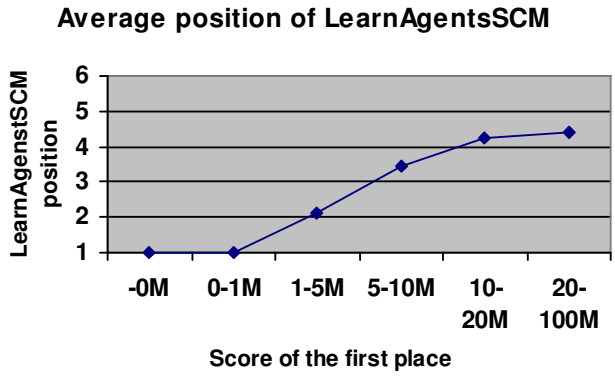

Figure 3. The average position of LearnAgentsSCM

Our system classified to the quarter-finals because it finished the seeding round in the 16th place. Figure 4 depicts the average number of orders each agent received in the group A of the quarter-final round. The results clearly present a low number of orders received by our system, and this is due to the adoption of a more conservative strategy. Unfortunately, this strategy presented a poor result in the quarter-finals.

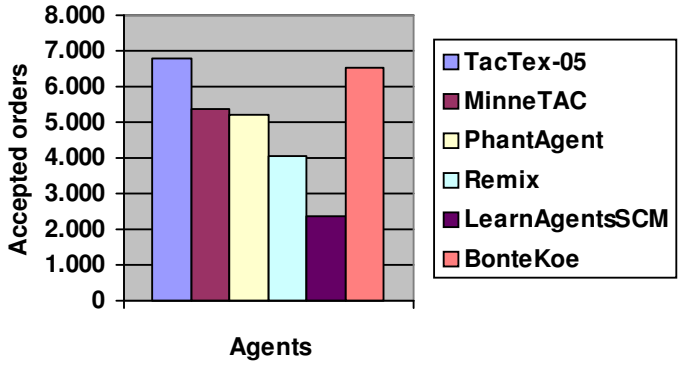

Figure 4. The average number of orders received by each agent in group A of the 2005 TAC SCM quarter-finals

Position	Agent	Average Score
1	TacTex-05	17.78 M
2	MinneTAC	11.91 M
3	PhantAgent	7.026 M
4	Remix	6.686 M
5	LearnAgentsSCM	4.683 M
6	BonteKoe	-2.200 M

Table 1. Final scores of the 2005 Quarter-finals

The primary goal of this work is to present a multi-agent architecture for a generic supply chain that uses a direct sales model, which links customers to a manufacturer through the Internet. A multi-agent design is used in the architecture, because we believe it facilitates the development of modular entities that are distributed and reus-

able. The design was also used to implement an agent entry for the Trading Agent Competition. This system competed against 32 entries, and was able to classify to the quarter-finals of the 2005 competition.

References

1. Chopra, S., and Meindl, P.. Supply Chain Management – Strategy, Planning, and Operations. Second Edition Pearson Prentice Hall, 2004.
2. Wooldridge, M. Intelligent Agents. In: Weiss, G. Multiagent systems: a modern approach to distributed artificial intelligence. The MIT Press, Second printing, 2000.
3. Ferber, J. Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence. Addison-Wesley Pub Co, 1999.
4. Garcia, A., Lucena, C. J. and Cowan, D. Agents in Object-Oriented Software Engineering. Software: Practice and Experience, Elsevier, pp. 1-32, May 2004.
5. Arunachalam, R. and Sadeh, N.. The Supply Chain Trading Agent Competition. Electronic Commerce Research and Applications, Volume 4, Issue 1, Pages 66-84, Spring 2005.
6. Sardinha, J. A. R. P., Paranhos, P. M., Cunha, P. M., Molinaro, M. S., Milidiú, H., Milidiú, R. L., Lucena, C. J. P.. LearnAgentsSCM - A multi-agent system for the TAC Supply Chain Management. Poster Session at TAC 2005, Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, August 2005.
7. TAC Research Reporst. http://tac.eecs.umich.edu/ researchreport.html. Accessed in November 2005.
8. Keller, P. W., Duguay, F., Precup, D.. RedAgent-2003: An Autonomous Market-Based Supply-Chain Management Agent. Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'04), Volume 3, pp. 1182-1189, 2004.
9. Kiekintveld, C., Wellman, M., Singh, S., Estelle, J., Vorobeychik, Y., Soni, V., Rudary, M.. Distributed feedback control for decision making on supply chains. Fourteenth International Conference on Automated Planning and Scheduling, Whistler, BC, 2004.
10. Benisch, M., Greenwald, A., Grypari, I., Lederman, R., Naroditskiy, V., Tschantz, M.. Botticelli: A Supply Chain Management Agent. Third International Joint Conference on Autonomous Agents and Multi-Agent Systems, New York, 2004.
11. Pardoe, D., Stone, P.. Bidding for Customer Orders in TAC SCM: A Learning Approach. AAMAS-04 Workshop on Trading Agent Design and Analysis, New York, 2004.
12. Simchi-Levi, D., Kaminsky, P., Simchi-Levi, E.. Designing and Managing the Supply Chain. McGraw-Hill, second edition, 2002.
13. W3C. OWL Web Ontology Language Reference. W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-ref/. Accessed in November 2005.
14. Collins, J., Arunachalam, R., Sadeh, N., Eriksson, J., Finne, N., Janson, S.. The Supply Chain Management Game for the 2005 Trading Agent Competition.

School of Computer Science, Carnegie Mellon University, Technical Report CMU-ISRI-04-139, December 2004.