A Robust Agent Design for Dynamic SCM Environments

Ioannis Kontogounis[1], Kyriakos C. Chatzidimitriou[2], Andreas L. Symeonidis[1,3], and Pericles A. Mitkas[1,3]

[1] Electrical and Computer Engineering Dept., Aristotle University of Thessaloniki, GR541 24, Thessaloniki, Greece
[2] Department of Computer Science, Colorado State University, Fort Collins, Colorado 80523, USA
[3] Intelligent Systems and Software Engineering Laboratory, Informatics and Telematics Institute – CERTH, GR570 01, Thessaloniki, Greece
`kontogou@auth.gr, kyriakos@cs.colostate.edu, asymeon@iti.gr, mitkas@eng.auth.gr`

Abstract. The leap from decision support to autonomous systems has often raised a number of issues, namely system safety, soundness and security. Depending on the field of application, these issues can either be easily overcome or even hinder progress. In the case of Supply Chain Management (SCM), where system performance implies loss or profit, these issues are of high importance. SCM environments are often dynamic markets providing incomplete information, therefore demanding intelligent solutions which can adhere to environment rules, perceive variations, and act in order to achieve maximum revenue. Advancing on the way such autonomous solutions deal with the SCM process, we have built a robust, highly-adaptable and easily-configurable mechanism for efficiently dealing with all SCM facets, from material procurement and inventory management to goods production and shipment. Our agent has been crash-tested in one of the most challenging SCM environments, the trading agent competition SCM game and has proven capable of providing advanced SCM solutions on behalf of its owner. This paper introduces `Mertacor` and its main architectural primitives, provides an overview of the TAC SCM environment, and discusses `Mertacor`'s performance.

1 Introduction

Current trends in Decision Support (DS) Supply Chain Management (SCM) software tend to integrate Supplier Relationship Management (SRM), Customer Relationship Management (CRM), and Enterprise Resource Planning (ERP) primitives, in order to provide competitive business solutions. DS SCM software efficiently monitors and records all transactions, while supply chain strategies are applied at various stages of the process, in order to reduce cost and improve service levels [1].

Nevertheless, in such systems human expertise is imperative, and this usually leads to their deprecation, from advanced DS systems to mere transactional databases. In addition, the flourishing of virtual organizations and electronic marketplaces, has led to the shift from traditional markets, relying on long-term trading partner relationships, to more dynamic SCM environments, where goods (raw material, end products) are auctioned between interested parties (suppliers, manufacturers, customers), and advanced bidding strategies are employed in order to achieve optimal results. The structure of these auction environments requires computational strength and accurate timing, therefore implying the need for autonomous SCM solutions, which shall identify rapid market changes and handle them in a cost-effective manner, in order to profit from specific economical regimes. Nevertheless, these SCM solutions should also satisfy all security, safety and soundness issues that may arise in such uncertain environments.

Recent research literature acknowledges intelligent agents as the most appropriate technology for trading and auctioning in electronic markets [2]. Equipped with smart strategies and efficient learning techniques, agents can provide robust solutions to deal with uncertainty and complexity. The more dynamic the SCM environment, the more intelligent the agent has to be.

In this context, we introduce `Mertacor`, an agent that employs a robust SCM mechanism for trading within a dynamic SCM environment. `Mertacor` takes over all company activities, aiming to maximize company revenue. Through extensive analysis, a number of key points within the SCM process have been identified and incorporated into the agent's trading mechanism. By the use of heuristics, SCM business rules, scheduling algorithms, data mining techniques and fail-safe mechanisms, `Mertacor` proves extremely capable of trading with other entities, within a dynamic, multi-variate, uncertain environment. `Mertacor` performance has been extensively tested through its participation in one of the most demanding trading agent competitions, the Trading Agent Competition (TAC) SCM game (http://www.sics.se/tac).

The rest of the paper is organized as follows: Section 2 provides an overview of the TAC SCM environment, in order to specify the framework `Mertacor` was tested on. Section 3 describes the functional characteristics of `Mertacor`, while Section 4 delves deeper into the implementation with respect to TAC SCM. Finally, Section 5 discusses `Mertacor` results at the TAC SCM game, while Section 6 summarizes work conducted and concludes the paper.

2 TAC SCM Overview

Within the TAC SCM game [3], agents act as Personal Computer (PC) manufacturers, competing with others on supplier and customer contracts. Throughout the duration of the game, each agent has to: (a) negotiate supply contracts, (b) bid for customer orders, (c) manage daily assembly activities and, (d) ship completed orders to customers.

A maximum number of six agents can connect to the TAC SCM game server, which simulates the suppliers and customers, and provides banking, production, and warehousing services to the competitors. Each agent is running its own PC assembling unit, which has limited production capacity. Sixteen (16) different types of PCs can be assembled, each requiring a different component compilation. The ten (10) different components available (CPUs, Motherboards, Memory, and Hard disk drives) can be procured through sending RFQs (Request For Quote) and issuing orders to the suppliers. Every day customers send RFQs and agents bid on them, depending on their ability to satisfy delivery dates and prices. The bid price should not exceed the reserve price the customer requires, which is between $75 - 125\%$ of nominal price of PC components. The next day, if an agent's quote is a winning offer, customer sends the order to the agent. To get paid, the agent must either assemble the ordered PCs or supply the customer with PCs already stocked in inventory on time. If an agent fails in delivering customers orders, it is charged with a penalty. Winner is declared the agent with the greater revenue at the end of the game. Game length is 220 days, with each day lasting 15 seconds. Fig. 1 provides a schematic representation of the game. A more detailed description of the game can be found at [4].

3 Agent Mertacor

Taking a closer look at the TAC SCM specifications, one can easily distinguish four (4) primary SCM facets: a. *Component Supplies Procurement*, dealing with negotiations on cheap component contracts, b. *Inventory Management*, managing stock requirements c. *Production and Delivery Scheduling*, and d. *Customer Bidding*, dealing with negotiations on PC sales. In order to better manage and efficiently act on each one, `Mertacor` has employed a modular architecture. Each

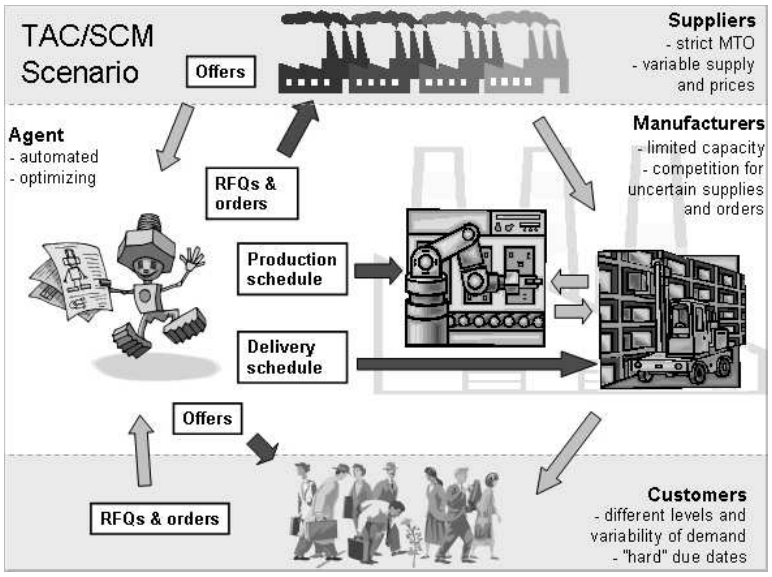

Fig. 1. An overview of the TAC SCM game.

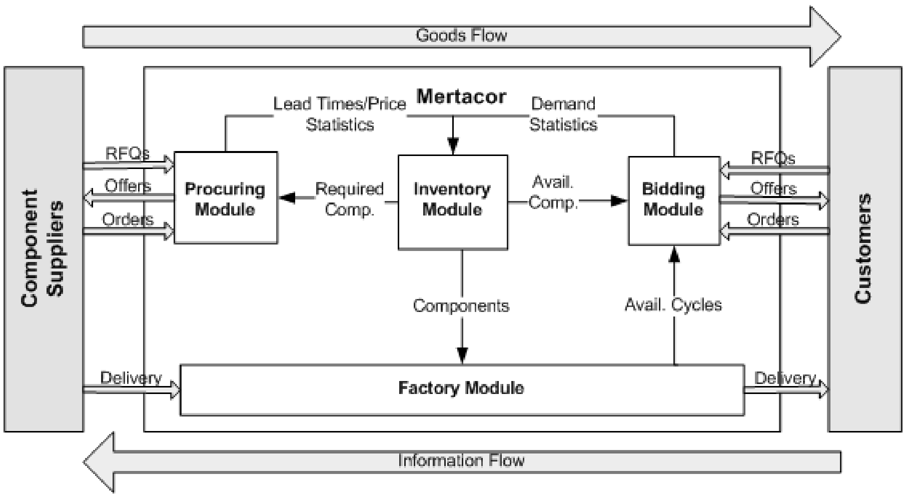

Fig. 2. The architecture of agent Mertacor.

task is delegated to a specific module, while all modules act in close collaboration. `Mertacor`, being a wrapper around the modules, ensures communication with suppliers and customers. Such a modular architecture can be easily applied to other environments also, outside the constraints of the competition. Following other successful paradigms [5, 6], `Mertacor` exploits the integration of techniques from the Operations Research (OR) literature, namely heuristics and adaptive algorithms, as well as statistical modeling. The overall `Mertacor` architecture is illustrated in Fig. 2, where four core modules can be identified:

1. The Inventory Module (IM)
2. The Procuring Module(PM)
3. The Factory Module (FM), and
4. The Bidding Module (BM)

IM constitutes the cornerstone of one's supply chain structure. SCM literature provides many paradigms of IM techniques, i.e. make-to-stock and make-to-order. `Mertacor` realizes an assemble-to-order system (ATO), a hybrid combination of the two aforementioned paradigms, which proves suitable in environments where assembly times are significantly smaller than replenishment times [7]. Additionally, an ATO system eliminates end-product inventory, reduces storage costs, improves forecast accuracy through demand aggregation, and provides quicker response time for order fulfillments through risk pooling.

Both PM and FM are based on heuristics. PM, which is primarily responsible for balancing the need for cheap component procurement to the running needs of the assembly line, attempts prediction of future demands, in order to pre-order affordable components. For FM, which is responsible for producing accurate schedules and for providing the bidder with information on the factory production capacity, a simulation procedure along with some heuristic algorithms was adopted. The simulator creates a projection of what the factory should expect in the near future (usually when conditions are less likely to change) and then diffuses this information to the rest of the modules.

With respect to bidding, a statistical model, capable of predicting the winning price of an order, was developed. Certain customer RFQ properties and the running SCM environment state are used to predict. Training data are derived from logs of previously played games, while some simple but effective fail-safe mechanisms were added, in case the predicted models are invalid. This learning approach proved to be fairly accurate, abiding by the standard rules of a market. Thus, the simulation can be considered realistic and the learning methodology applicable in real life domains.

4 Agent Modules

4.1 Inventory Management

An ATO system works as follows: The main goal of the system is to define certain inventory levels (thresholds) that need to be satisfied and below which replenishment is needed. These thresholds are calculated in real time for each component using the following equation [1, 7]:

$$R = D_{AVG}L_{AVG} + z\sqrt{L_{AVG}D_{STD}^2 + D_{AVG}^2 L_{STD}^2}$$

where D is the demand for specific component, L is the supplier lead time and z is a safety factor denoting the service level.

Demand is given in terms of products from the orders made by the customers. Statistics of product demand may vary making the thresholds more unstable. Nevertheless, by approximating demand in terms of product ranges (high-end, mid-end, low-end), smaller variations can be achieved. Additionally, in ATO no finished product inventory is kept (lower storage costs) and since components are shared along many products: (a) the levels of the thresholds are lowered due to aggregation of demands (b) internal component exchanges are applicable in order to avoid late orders and penalties. Component demand is calculated on the range demand by applying the scheme in Fig. 3, which has been adjusted to the TAC SCM specs. Minimum and maximum levels are also coded as fail-safes. The aforementioned system can handle unique components for the product families and can be extended to become a configure-to-order system (CTO), where there are no pre-specified end-products and the customer can personally select the set of components [7].

4.2 Component Procurement

`Mertacor`'s performance is highly dependent on two factors: (a) having an inventory filled up with cheap components and (b) satisfying the inventory levels,

Comp. ID	Low	Medium	High
1	0.4	0.33	0.0
2	0.0	0.17	0.6
3	0.6	0.17	0.0
4	0.0	0.33	0.4
5	0.4	0.5	0.6
6	0.6	0.5	0.4
7	0.8	0.5	0.4
8	0.2	0.5	0.8
9	0.6	0.5	0.4
10	0.4	0.5	0.6

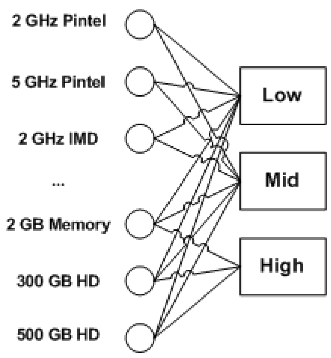

Fig. 3. Mapping range demand to component demand.

since each delayed order implies a penalty and after five days, order cancellation. In order to cope with these requirements, we have developed a simple strategy, which is divided into two distinct phases: initial and standard.

`Mertacor` initial procurement strategy is followed for the first two days. The RFQs sent to the suppliers on Day-0 and Day-1 aim to build an initial inventory so that `Mertacor` can start production immediately and therefore start bidding for orders from the first day. The components procured are predictably expensive, nevertheless their usability is increased, since at the beginning of the game, competition is not that strong, leading to higher product prices. On Day-1, another bundle of RFQs is sent, aiming to acquire relatively large quantities of cheap components for the following days.

On Day-2 `Mertacor` switches to a normal-state procurement strategy, designed to satisfy the aforementioned goals. It uses all five (5) available RFQs per component and per supplier each day, to maximize knowledge of the supplier's selling prices (probing). RFQs sent by our agent can be divided into three categories:

- *Normal procurement RFQs*, aiming to satisfy inventory reorder levels and allow bidding for customer orders in the near future. Quantities for these RFQs are calculated based on the reorder levels of the inventory manager.
- *Critical procurement RFQs*, a special state in which our agent tries to procure components in order to satisfy customer orders.
- *Early procurement RFQs*, in an effort to obtain low-priced components several days before they may be needed. These RFQs indicate quantities that if ordered, they would cause inventory to exceed reorder levels, so that there is no need for normal procurement after some posterior point in the game.

4.3 Production and Delivery Scheduling

The FM is responsible for: (a) generating production and delivery schedules schedules, (b) adjusting inventory levels, and (c) adjusting available factory cycles. This module is also assigned the task of integrating the procuring, inventory and sales parts. FM is the most resource demanding module of the agent. It implements a factory simulator, which simulates the operation of the factory for several days in the future, attempting to produce a draft of what will follow, based on knowledge on future supplier deliveries, customer deliveries, customer orders, future production and delivery schedules. When the schedules are produced, they are communicated to the interested parties. The number of future days `Mertacor` is simulating is defined as *look-ahead time*. For the current implementation the look-ahead time has been specified to fifteen (15) days.

The algorithm used by the factory simulator to predict forecoming inventory needs is iterative. For each day of the look-ahead period starting from today, the agent has to:

1. Update current inventory with supplier deliveries expected today
2. Update inventory with products assembled from factory (last schedule)
3. Remove components that are needed for tomorrows production
4. Remove products that are to be delivered tomorrow
5. Remaining inventory is the starting inventory for next day

This way the agent is aware of the expected inventory levels and factory cycles for the next 15 days, and alerts the bidder not to exceed these levels.

The daily production schedule includes the orders that fit within the daily factory capacity (2000 cycles). In case there are more orders and the capacity is exhausted, a greedy scheduling procedure is employed:

– Customer orders are sorted based on due date
– Orders with the same due date are sorted based on penalty
– Orders with similar due dates and penalties are sorted based on expected profit, that is unit price x quantity

Another parameter taken into account are the potential orders that should be scheduled. Not all available factory capacity for future days is committed to current RFQs, but a constantly decreasing fraction of the factory's nominal capacity. Thus, it is possible to save cycles for profitable RFQs, expected in the next days. If an RFQ can be successfully scheduled, the bidder is given a signal to go ahead and place a bid for that RFQ.

4.4 Bidding

Our bidding strategy is focused on finding the optimal bidding price for each RFQ received and then deciding on which of these RFQs to bid, sorting on anticipated profit. `Mertacor`'s initial hypothesis is that every bid it places will be successful, and in order to realize it, a bidding mechanism based on machine learning techniques has been implemented. Through this mechanism, the market is modeled off-line based on data from past games. Twenty five (25) attributes were initially selected. Through a cross-validation procedure, using multiple linear regression (MLR) and backward elimination based on the F-statistic, the most parsimonious model within "one-standard-error" from the minimum was picked, leaving seven final predictors [8]. The initial set of attributes, which was formulated by intuition, is the same as in [9], while the data mining algorithm that has eventually been selected to model the market is the M5' [10], since it outperformed other similar algorithms with respect to root mean square error (RMSE). The optimal, in our case, β coefficients for MLR can be found in Table 1.

Table 1. The attributes used to predict order prices. These are: the RFQ's `due date`, its `reserve price`, the `highest` and `lowest` prices for the previous `two` days, and the `current demand` of PCs. The value of the intercept was 1515.94.

Feature	Due Date	Res. Price	High-1	High-2	Low-1	Low-2	Demand
β	-8.08	15.69	161.04	66.45	67.97	39.42	9.32

This modeling provides us with the parameters that potentially affect the bidding strategies of the marketplace. Since inputs are normalized, the β values are directly comparable and their signs indicate the correlation between the input and the output. Interesting rules that may be derived are: the higher the highest and lowest prices for the past two days, the higher the current price; the later an order is due, the lower its price; the bigger the reserve price, the higher the offer to the customer; the higher the demand from the customer-side, the higher the prices, since there is less competition.

The bidding module also incorporates two on-line modeling mechanisms: a fail-safe mechanism designed to function complementary to the trained models, handling unexpected circumstances of selling prices, and an overbidding mechanism to help with filling the capacity given by the scheduler.

The former, named the follower for its ability to follow prices on-line, evaluates the minimum and maximum prices for PCs ordered the previous day as provided by the daily price reports, and predicts the approximate level of bidding price for each RFQ. The follower deploys linear interpolation based on the assumption that the maximum price paid corresponds to the maximum customer RFQ reserve price, while the minimum price paid corresponds to the minimum RFQ reserve price. Let P_M be the model price and P_F the follower price. Then, the final bid price is calculated as follows:

$$if(|P_F - P_M|/P_M)\% < threshold(10\%), P_M, else, P_F$$

Experimenting has shown that this fail-safe mechanism has significantly helped `Mertacor` through sudden market changes, especially at the start and end periods of games, when the game unfolded in unpredictable manners. Results showed a 20% improvement in RMSE accuracy compared to other on-line naive mechanisms [11]. Additionally, to support the use of the off-line model versus the on-line, a increase of 13% in RMSE accuracy was measured, favoring the former approach.

As far as overbidding is concerned, a scheme using the k-Nearest Neighbors (k-NN) algorithm [10] was developed to produce a probability of acceptance for each bid placed. Having identified the probability of a RFQ becoming an order, the bidding module signals the scheduler to commit only the fraction of the capacity that corresponds to that probability, letting the remaining capacity for the next RFQ in the row. The probability is calculated as the fraction of the n neighbors that became orders versus the total number of neighbors k (n/k). For the TAC SCM game, a value of k=10 was used. The neighbors/exemplars are RFQs sent to the customers the previous day(s) tagged either as accepted or rejected. The set of attributes used are the attributes selected for the off-line model.

5 Competition Results

`Mertacor` participated in the TAC SCM 2005 competition and performed quite well in all rounds. It came 11th among the 32 teams that participated in the qualifying phase and 10th among the 25 teams in the seeding phase. Going through the results of the qualifying rounds, we came to the conclusion that the reduced `Mertacor` efficiency was due to the fact that our agent was trained to cope with strong competition, accomplished only when six competitors participated. This, unfortunately, was not certain through the preliminary phase. During the finals, though, where the games played were much more competitive than in the previous rounds, `Mertacor` was a top scorer in the quarter finals and placed 3rd in the semi finals. At the final round `Mertacor` competed with the other 5 best scoring agents and finished 3rd, with a positive bank balance (See Table 2).

Table 2. Mean skills of the agents in the finals for a total of 16 games. The skills are: final bank balance (Score), revenue, cost of components, storage costs, delivery performance and factory utilization.

Agent	Score ($)	Revenue ($)	Material ($)	Storage ($)	Del. (%)	Util. (%)
TacTex-05	4.741 M	108.586 M	100.614 M	2.013 M	97,75	87,81
SouthamptonSCM	1.604 M	108.246 M	102.375 M	2.843 M	98,06	87,75
Mertacor	546 272	75.582 M	72.639 M	1.730 M	98,88	60,63
Deep Maize	-220 503	107.681 M	103.309 M	2.645 M	97,31	85,13
MinneTAC	-311 844	81.903 M	79.728 M	1.887 M	99,88	65,00
Maxon	-1.985 M	71.105 M	68.588 M	3.520 M	100	56,19

Even though numerous games must be played in order to evaluate the "true" value of an agent and its game profile with respect to the others, we will restrict our analysis to the results in Table 2 displaying some of the strong points and drawbacks of the developed design. First of all, the ATO system employed, along with the procurement strategy followed, resulted to an agent with the lowest storage costs, high delivery performance rates from agents willing to risk the 100% delivery rate for more profit, and low material costs. One could argue that the last metric also accounts for the inability to compete for orders, but once put into perspective of `Mertacor`'s performance, good inventory management is implied. In addition, the bidding module ensures a high Average Selling Price (ASP) for the agent (2nd with a 0.776 normalized ASP - 0,780 for agent Maxon).

One of the most characteristic drawbacks of the final was the expensive contracts with suppliers, placing `Mertacor` 6th, with 0,726 average normalized CPU buying price (the most expensive component - 0,694 for agent Southampton-SCM). Another bottleneck was the low factory utilization (equivalent to low revenue) that can be interpreted to low throughput (rate of products out of the factory versus components in the factory) causing additional reduction to profit. A balance between high selling prices and high throughput is imperative.

6 Conclusions

In this paper, we have introduced `Mertacor`, a SCM agent designed to participate in the TAC SCM game 2005. The agent employs a combination of OR, heuristic and statistical modeling techniques, in order to manage a wide range of activities in an efficient manner. The architecture proposed is generic, and can be applied to other SCM environments also. Focusing on specific points, one can see that the inventory management system, designed for the IBM PC production line, performed very well, in an uncertain and dynamic environment, outside the assumptions made by the authors. The learning models were able to capture the dynamics of the markets at hand, while the heuristics applied to the supplies and the factory modules worked well enough for the agent to be ranked 3rd in the competition. As far as the TAC community is concerned, we have introduced some novel ideas that could help further improve the game. Future research work on `Mertacor` includes the development of more accurate predictors on the behavior of both customers and suppliers. That, along with some improvements in the heuristics, would allow a bigger factory throughput, which is the confining factor for `Mertacor`.

References

1. Levi, S.D., Kaminsky, P., Levi, S.E.: Designing and managing the supply chain. McGraw-Hill, Illinois (2000)
2. He, M., Jennings, N.R., Leung, H.: On agent-mediated electronic commerce. IEEE Transactions on Knowledge and Data Engineering **15**(4) (2003) 985–1003
3. Arunachalam, R., Sadeh, N.: The supply chain trading agent competition. Electronic Commerce Research and Applications **4** (2005) 63–81
4. Collins, J., Arunachalam, R., Sadeh, N., Ericsson, J., Finne, N., Janson, S.: The Supply Chain Management Game for the 2005 Trading Agent Competition. Technical Report CMU-ISRI-04-139, CMU (2004)
5. Pardoe, D., Stone, P.: TacTex-03: A supply chain management agent. SIGecom Exchanges: Special Issue on Trading Agent Design and Analysis **4**(3) (2004) 19–28

6. He, M., Rogers, A., David, E., Jennings, N.R.: Designing and Evaluating an Adaptive Trading Agent for Supply Chain Management Applications. In: IJCAI-05 Workshop on Trading Agent Design and Analysis. (2005)
7. Cheng, F., Ettl, M., Lin, G.: Inventory-Service Optimization in Configure-to-Order Systems. Technical Report RC 21781, IBM (2001)
8. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer (2001)
9. Pardoe, D., Stone, P.: Bidding for customer orders in tac scm: A learning approach. In: Workshop on Trading Agent Design and Analysis. (2004)
10. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann (2000)
11. Dahlgren, E., Wurman, P.R.: Packatac: A conservartive trading agent. SIGecom Exchanges **4**(3) (2004) 33–40