IADIS International Journal on WWW/Internet
Vol. 6, No. 1, pp. 37-49
ISSN: 1645 – 7641

DESIGNING AND IMPLEMENTING DATA MART FOR AN AGENT-BASED E-COMMERCE SYSTEM

Michał Drozdowicz *Department of Mathematics and Information Sciences, Technical University of Warsaw, Warsaw, Poland*

Maria Ganzha, Maciej Gawinecki, Paweł Kobzdej, Marcin Paprzycki *Systems Research Institute Polish Academy of Sciences, Warsaw, Poland*

Maria.Ganzha@ibspan.waw.pl

ABSTRACT

Successful data mining, forecasting and automatic reasoning require vast amounts of high quality domain specific data stored in an easily accessible data store. Here we present a solution for gathering and storing historical data for use by the e-shop subsystem of a model agent based e-commerce system. We discuss the agent communication scenarios feeding information to the system and propose a data mart design for storing data. For each table of the data mart we discuss the way that data stored within it can be used in adapting behaviors of the e-shop.

KEYWORDS

e-commerce, software agent, data warehousing, knowledge management, data mining, adaptive behavior

1. INTRODUCTION

Success stories of companies such as Wal-Mart and Procter & Gamble show that integrated and effective analysis of retail data can provide a competitive advantage [0, 0]. In the area of e-commerce, Amazon.com uses data analysis for content personalization, targeted e-mail campaigns and product recommendations [0]. While substantial work has been devoted to data warehousing and mining in e-commerce ([0, 0, 0, 0]), most publications concentrate on analyzing data gathered through visitor's clickstreams. In this paper we consider issues involved in collecting, storing and processing historical data for use in an agent-based e-commerce system (see [0, 0] for more details). The main difference is that in such system there are no clickstreams to store and extract knowledge from. The interaction between the "client" and the system has been replaced by interactions (a) between the client and an agent

(the *Client Agent*) representing her/his interests, and (b) interactions between agents representing the *Client Agent* (i.e. *Buyer Agents*) and the e-shop representing agents. Therefore, we present a solution for storing history of selected events that take place within the e-shop, for use in automatic reasoning and decision process automation. We start with discussion of possible and employed approaches to monitoring state of the e-shop. Next, we present the design of a data mart for efficient storage of large quantities of data collected during the functioning of the e-shop along with basic ways of taking advantage of the information stored therein. We also indicate how particular collected data items can be used to support adaptive processes in the e-shop.

2. SYSTEM DESCRIPTION

The system under consideration is an agent-based virtual marketplace, where agents representing buyers engage in price negotiations with agents representing sellers. Conceptualization of the proposed system has been represented as a use case diagram in Figure 1. Since the detailed description of the system can be found in [0, 0], here, we only briefly describe typical scenarios in the system.

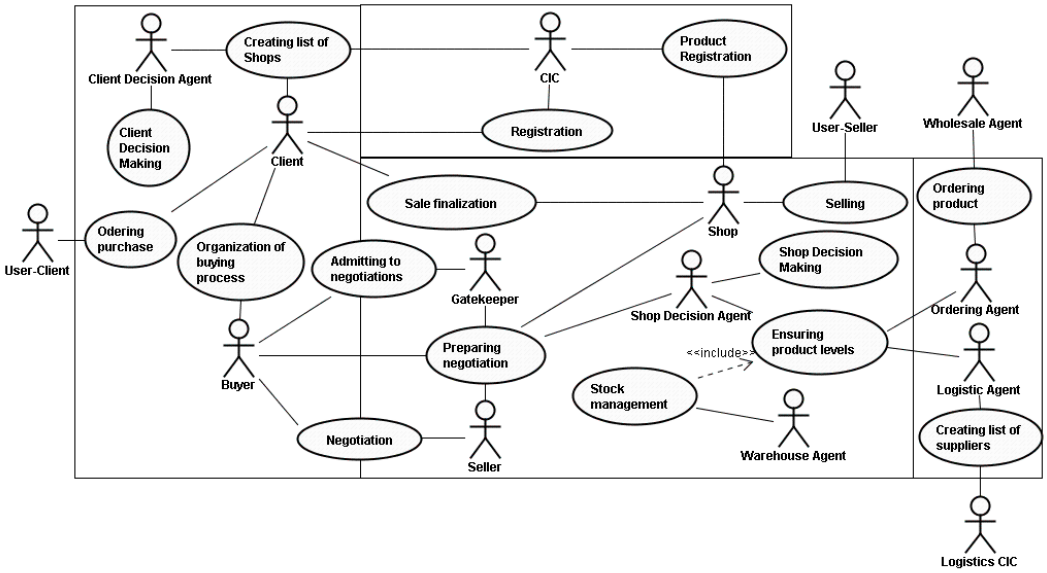

Figure 1. Use case of the e-commerce system

2.1 Client Subsystem

First, let us consider agents supporting the *Client* in her shopping needs. The *Client Agent* (*CA*) is responsible for direct *Client* support. Based on the description of the product provided by the *User-Client* it obtains a list of shops to be contacted. This is done by querying a yellow-

pages agent called the *Client Information Centre* (*CIC*). Obtained shop list is adjusted on the basis of trust considerations (see, [0]). As a result, the *CA* sends *Buyer Agents* to selected shops to involve in price negotiations. After receiving offers, the *CA* asks the *Client Decision Agent* (*CDA*) to determine which offers are acceptable and among them which is "the best." If there exists such offer, it finalizes the purchase. If no appropriate offer can be found, the *CA* decides further course of actions, which may involve trying to negotiate for a better price, or informing the *User-Client* that purchase under specified conditions is impossible. It should be noted, that to allow the client to decide on which offer to pursue after every finished negotiation the winning client is given a reservation period, during which it can purchase the item on the negotiated terms [4].

2.2 Shop Subsystem

The activities of the shop's part of the system can be split into three major processes – (1) selling products, (2) managing trust towards customers, (3) managing product inventory, and (4) managing negotiation mechanisms. Selling products is handled by two agents: the *Gatekeeper Agent* (*GA*) that registers and authorizes entrance of buyers interested in taking part in a negotiation, and the *Seller Agent* (*SeA*), who represents the shop during the negotiation (for details on the negotiation process see, [0, 0, 0]). The finalization of purchase as well as registering (and de-registering) products with the *CIC*, and coordinating work of other agents in the e-shop is the duty of the *Shop Agent* (*SA*). Another major issue is managing trust towards the shop's clients, which is the main parameter in the decision of admitting buyers to negotiations and fixing the negotiation period length to minimize the potential losses caused by unreliable or malicious customers (see, [0]). This process is performed by the *Shop Decision Agent* (*SDA*). Managing stock levels is a task handled by a group of agents, supervised by the *Warehouse Agent* (*WA*) that stores and manages information about current inventory and product reservations. It also uses sales forecasts prepared by the *SDA* to proactively ensure desired level of all products. Stock supply orders are carried out by the *Logistic Agent* (*LA*) that interacts with a wholesaler-related equivalent of the *CIC* to receive a list of wholesalers that sell specific products and dispatches requests for product sale and delivery proposals using workers from a pool of *Ordering Agents* (*OA*). More details about the logistics subsystem can be found in [0, 0]. The third major functionality of the shop subsystem – management of price negotiation mechanisms is performed by the *Shop Decision Agent*, and incorporates selecting the most appropriate negotiation mechanisms and strategies, as well as updating prices of products along with other negotiation parameters.

3. KNOWLEDGE MANAGEMENT

Let us now focus our attention on processes involved in gathering and storing knowledge needed for accurate forecasting and decision making within the shop, i.e. data needed for the *SDA* to manage negotiation parameters and trust towards customers, and to calculate sales forecasts for the logistics subsystem. To effectively perform these tasks it is essential for the *SDA* to have access to complete knowledge both about the state and the history of the system as well as the conditions of the environment the system is operating in.

3.1 Information Gathering

The task of monitoring the state of the system (including storing information about events taking place within it) can be performed in several ways:

1) The monitoring module is active – it queries selected information sources and stores their responses (delivered in a "digested" form).
2) The information source is active – change in the state of the system or occurrence of an event result in a message, containing complete information about a specific event, being sent to the monitoring module.
3) Similar to 2), but the active source sends only a brief notification about the change or the event; while the full information is available in response to an additional request of the monitoring module.
4) Points 2) and 3) can be modified so that messages / notifications are sent only to entities that had previously subscribed to the specific information sources.

The first approach handles well situations where we can accept a delayed reaction to events taking place in the system, as the message is not received immediately after occurrence of an event, but only after it is requested. It can also be used when, within the system, there is no interest in specific events, but in sequences of events that occurred over a period of time. The second approach, on the other hand, lets the monitoring module acquire information about an event immediately after its occurrence. The disadvantage of this method is that the receiver is not able to specify/decide whether it needs the full information or just a notification. Thus this approach can overwhelm the system with large number of unnecessary messages. As a remedy, if this is acceptable in the system, it is possible to use the third approach. The last method is useful when the number of monitoring modules is large, or when the list of message receivers may be modified over time.

Let us now turn our attention to the information that is to be collected within the e-shop of the proposed agent-based system. The information collected by the *SDA* can be divided into:

1) Originating inside of the e-shop – received either from the *SA* or the *WA* agents, describing specific events related to the functioning of the e-shop, such as buyer registration, negotiation closing, transaction finalization, or restocking deliveries. These messages are meant to provide the most complete insight into activities of individual agents within the system and thus should be delivered to the *SDA* immediately after occurrence of an event. Let us also note that most messages passed within the e-shop are rather small, while the *SDA* is the only receiver of the collected data. Therefore in this case the second approach to information gathering is possible.
2) Originating outside the e-shop – consisting of information periodically requested by the *SDA* and concerning the number of queries received from clients by the *CIC* and number of shops selling a specific product. Since the *CIC* is not an active source of information, the *SDA* has to be. Therefore the first approach to data gathering should be applied – the *SDA* requests periodically the needed data and receives it in a digested form.

Let us now describe in more detail the above indicated information gathering related scenarios of communication between the *SDA* and other agents in the e-shop and the outside environment.

3.1.1 Shop Agent

Because the *Shop Agent* coordinates actions of all major agents in the e-shop, it has the necessary information to inform the *SDA* about the following events (this list is not exhaustive):

1) *Buyer Agent* registering for admission to a negotiation. Message contains id's of the client and of the product of interest, time of registration, information whether the *BA* has migrated from the client's system or has been created by the shop. Note that creation of a *BA* within the shop requires *SDA's* permission (involves trust considerations [3]).
2) *BA* being removed from the shop. This can be caused either by lack of trust towards the customer (note that trust can change while the *BA* is within the e-shop [0]) or expiration of a timeout within which the *BA* was supposed to perform some action (e.g. the *BA* was to confirm reception of rules of negotiation). Notification about removal of the *BA* contains the id of the customer and the product, time of *BA's* registration and removal, and the reason for the removal.
3) After a negotiation is completed, its log is sent to the *SDA* (this is the only potentially large message). Such log contains complete information about the negotiation: its identifier, time of its beginning and conclusion, the negotiation template and strategy used, along with all the parameters. Moreover, the log includes list of buyers admitted to the negotiation with the time they received the negotiation rules and list of participants i.e. those buyers from that actually joined the negotiation. For each participant, the log contains the time they accepted the negotiation rules and details of their bids. Furthermore, for the winning bid the *SDA* is informed about the transaction id and length of the reservation.
4) *SDA* is also notified about the final result of the transaction (was it finalized, or did the buyer resign and when, or did the reservation lapsed). The message contains id's of the transaction and the client, the result, and the exact time of the event.
5) *SA* informs the about stock shortage. The *SA* withdraws the product from the *CIC* and sends a notification containing the id of the product to the *SDA*.

3.1.2 Warehouse Agent

The second agent reporting to the *SDA* is the *Warehouse Agent*. It notifies the *SDA* about situations linked to warehouse management:

1) The delivery of a product to the warehouse, described by the id of the product, time of the order, time of the delivery, the amount and price of the product delivered
2) Problems with maintaining the stock at the level recommended by the *SDA*. This situation may occur for example when wholesellers cannot provide the *WA* with large enough quantity of product items (to fulfil the *SDA* Forecast). These messages carry only the identifier of the product affected.

3.1.3 CIC

Additionally, the *SDA* retrieves from the *CIC* two types of information about the ("business") environment:

1) *CIC* is queried for the list of shops selling specific products (direct competitors). Here, the request is exactly the same as one of the client looking for shops selling a specified product [0].
2) *CIC* is asked about the number of clients' queries for a specific product.

It should be noted, that while providing *SDA* agents with information about shops selling products is the basic functionality of the *CIC*, giving away statistical usage information, such as the number of received queries, should be thought-of as an additional, paid, service of the *CIC*.

3.2 Data Storing

Looking at the above list of information provided to the *SDA* and their granularity (up to a single bid offered) along with the need of storing the complete history of the e-shop, it seems obvious that the amount of acquired data may become a problem. For instance, if data is not stored efficiently, the *SDA* will have the necessary data but, due to its amount, will not be able to extract knowledge from it. To overcome this issue we have designed multi-dimensional data mart structured according to the star / fact constellation schema proposed in [0]. The remaining parts of this paper are devoted to the description of the design of such data mart, focusing primarily on fact tables.

3.2.1 Bid Fact Table

The *Bid Fact Table* (represented in Figure 2) holds information about every bid made by a client during negotiations. This table is populated with data provided by the *SA* in negotiation logs and contains the amount and value of a single offer connected to dimensions such as: the client who made the bid, the product, the date and time of the offer, the negotiation template and strategy used and the bid status. The bid status dimension stores information on whether the bid was above or below the negotiation minimum price and if it was a winning offer.

Using the content of the *Bid Fact Table*, the negotiation process can be analyzed concerning a single negotiation, a specific client or a product; and further grouped and summarized by the strategy and the template used in the negotiation, or by the bid status information such as whether the bid was above or below the minimum acceptable price. By aggregating data contained in this table it is possible to calculate such measures as the amount of bids, products offered and the unit price across any combination of dimensions. Knowledge that is possible to extract from data stored in the *Bid Fact Table* can be used primarily for analysis of the negotiation behavior of clients, for use in fraud detection and for updating clients trust level, for example clients taking part in many negotiations but rarely posting any offers can be suspected of exploiting the process only to gather information about the selling strategy or the minimum price levels of the shop (this information can be then used by the competition to adjust their selling strategies). Another important use of this information is to investigate the effect of employing different negotiation mechanisms, parameter values and selling strategies on the profit generated by sales of a specific product or of a product group.

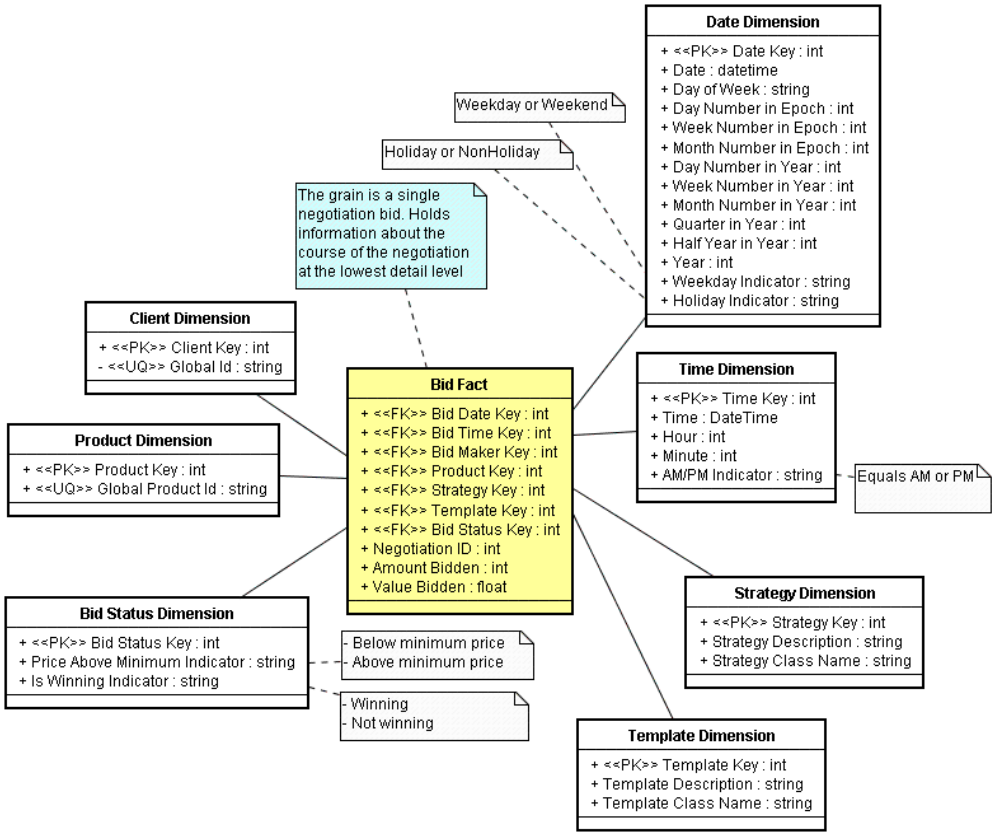

Figure 2. Bid Fact Table diagram.

3.2.2 Transaction Fact Table

The largest in the number of columns table in the schema is the *Transaction Fact Table* (depicted in Figure 3), which contains information about the complete sale process flow of every transaction in the system (with the exception of the bidding process persisted in the *Bid Fact Table*). The *Transaction Fact Table* is built around the Accumulating Snapshot pattern [0] and focuses on steps of the sales process, the client progress (from registering with the *Gatekeeper Agent* till the end of its interaction with the shop). Here, every event is described by appropriate dimension table links and by corresponding measures, which are mostly 0 or 1 at the most detailed level but can that be easily added-up in various combinations by grouping along selected dimensions (e.g. the number of client registrations for negotiations concerning a specific product in a specific period of time). Most of dimensions are date and time demarcated and therefore can be used to reason about the length of consecutive steps of the process (for example the length of the negotiation).

Most important uses of information contained in the *Transaction table* are: (1) identifying potentially fraudulent client behavior and determining trust towards customers based on the time difference between successive process events (e.g. to identify clients purposely deferring acceptance of the negotiation protocol to delay the negotiation start as a form of a Denial of

Service attack) ; (2) analysis of the correlation between employing different negotiation strategies and the point at which the client-shop interaction has finished, giving insight into the effectiveness of the selling strategy and the client sensitivity to negotiation parameter changes (by aggregating 0-1 event measures along strategy and template dimensions); (3) the shopping habits of specific clients like the frequency of shop visits and their outcomes, and products they are most interested in (grouping information by customers and products); and (4) the potential demand for specific products measured by the number of clients registering for and participating in negotiations. The *Transaction Fact Table* is populated using data received by the *SDA* throughout the whole sale process (see section 0): client registration (1)), client dismissal (2)), negotiation log (3)) and reservation finalization (4)) – after each step of the process (such as the acceptance of the negotiation rules or the end of the negotiation) appropriate event measures are set from 0 to 1 and references to date and time dimensions specifying the transition moment are set to the time of the event (e.g. the date and time the client accepted the rules).

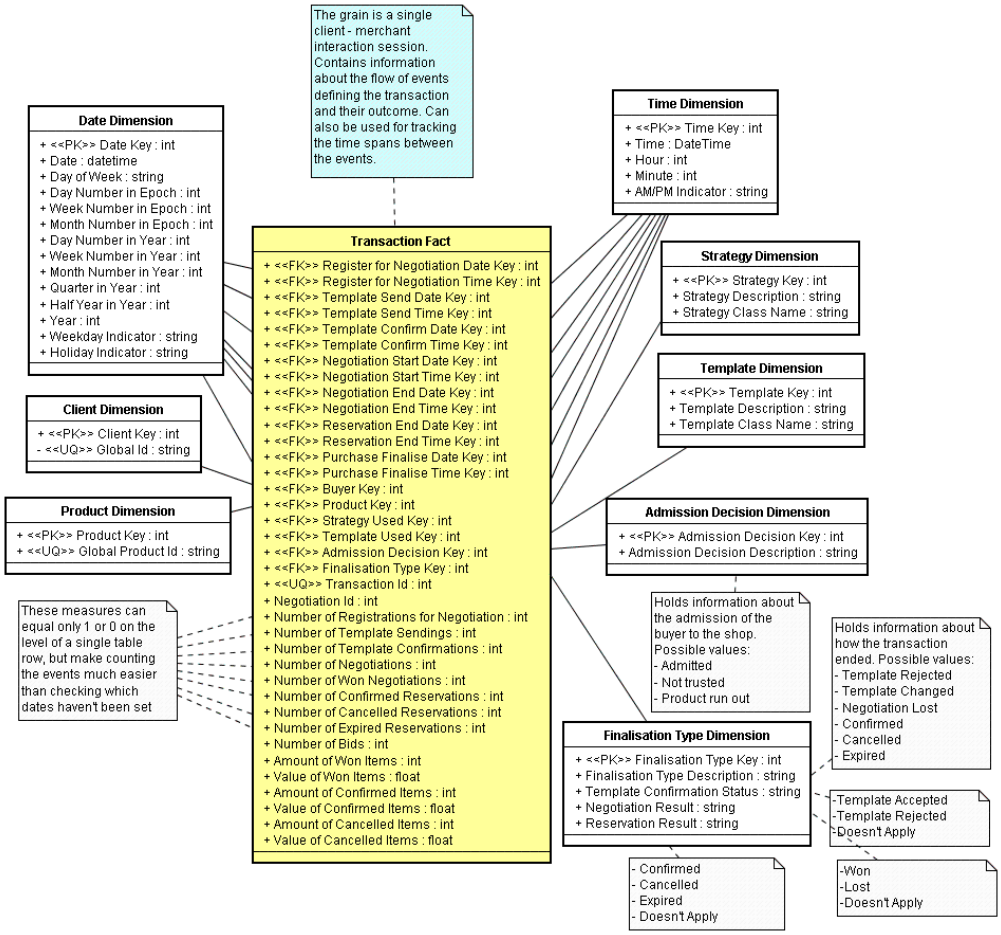

Figure 3. Transaction Fact Table diagram.

3.2.3 Negotiation Fact Table

The *Negotiation Fact Table* (described in Figure 4) contains general information about price negotiations. Every row in the table holds metrics describing a single negotiation and references to dimensions allowing data to be grouped and summarized by the negotiation end date and time, product, used strategy and template. The table does not contain any client-specific data (e.g. particular bids that occurred during a negotiation) and therefore does not link directly to the customer dimension. The negotiation is described by the offered product quantity and starting and minimum prices. Other measures include: quantity and value of the items reserved in winning bids, quantity and value of actually purchased items, total number of bids, number of winning bids and number of finished transactions. This information, although available also through combining data in the *Bid Fact Table* and the *Transaction Fact Table*, has been considered important enough to be aggregated and stored in a more easily accessible manner. One of the key reasons is the expected efficiency of operations that can be executed on this table (rather than finding needed information within multiple tables).

Utilizing data contained in the *Negotiation Fact Table* the *SDA* can analyze the impact of the negotiation parameters on the outcome of negotiations and their profitability. Other useful knowledge that can be deduced from this table is the real demand for specific products and whether the offered amount has been sufficient to satisfy it. Data from this table is also important for assessing accuracy of sales forecasting algorithms. Table's information is extracted from the negotiation log and from messages concerning reservation finalization.

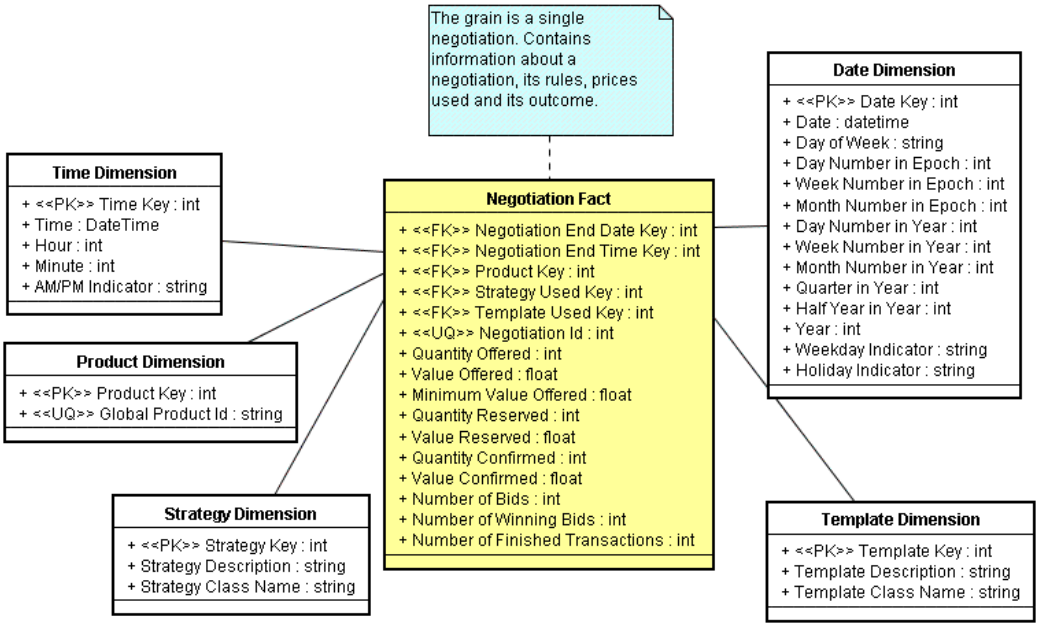

Figure 4. Negotiation Fact Table diagram.

3.2.4 Supply Fact Table

The *Supply Fact Table* (found in Figure 5) accumulates data about wholesale orders and deliveries of stock to the shop's warehouse. These events are described by the amount of

ordered and delivered product. Data is organized along the following dimensions: the id of the wholesaler that carried out the order, the ordered product and references to the date and time dimensions describing the order and delivery events.

The most important knowledge that can be extracted from the data in this table is the historical wholesale price of sold products – essential in estimating the shop's profitability; time it took to carry out the order and delivery in accordance to the initial terms (two key metrics for rating the wholesalers). Furthermore, information contained here can be used to adjust trust toward particular wholesalers. Information gathered in the *Supply Fact Table* is extracted from product delivery messages received from the *WA*.

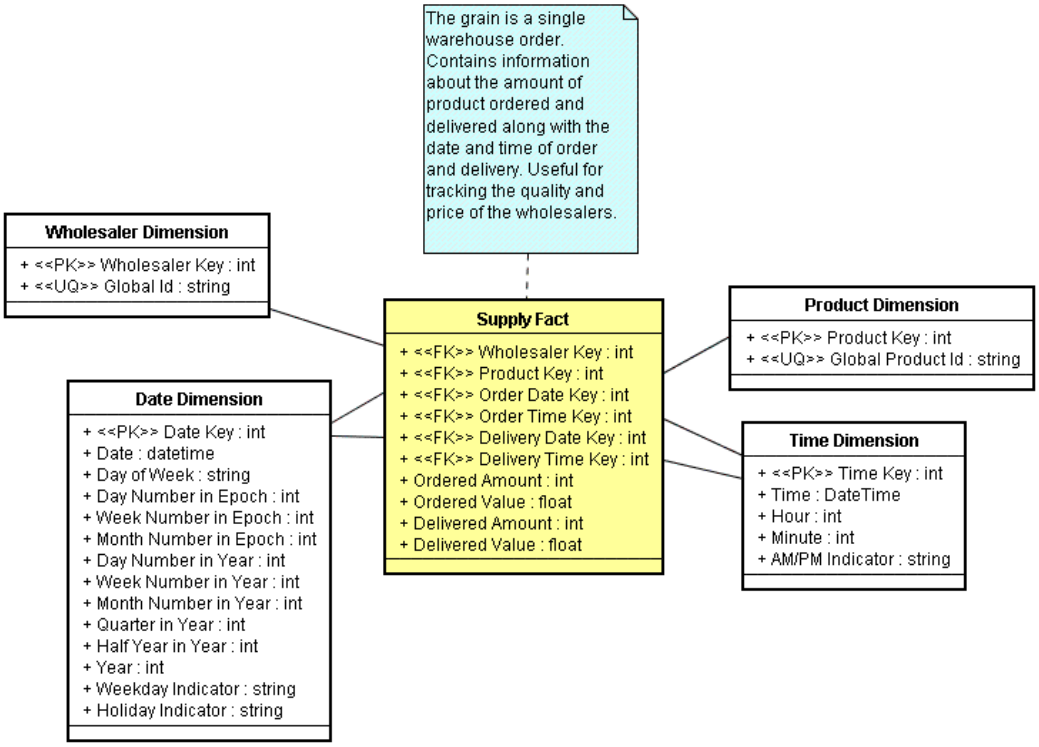

Figure 5. Supply Fact Table diagram.

3.2.5 Demand Fact Table

Information about the potential demand for products acquired from the *CIC* agent is stored in the *Demand Fact Table* (presented in Figure 6). Data available in this table can be used to approximate the global demand for a specific product and the level of competition. Moreover, if combined with information about the number of clients registering in the shop can serve as a base for estimating the shop's image among the customers. This table is filled with data received by the *SDA* from the *CIC*.

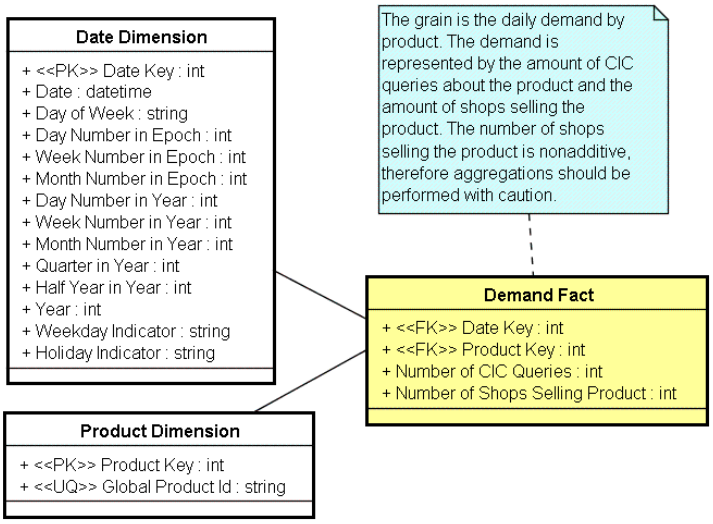

Figure 6. Demand Fact Table diagram.

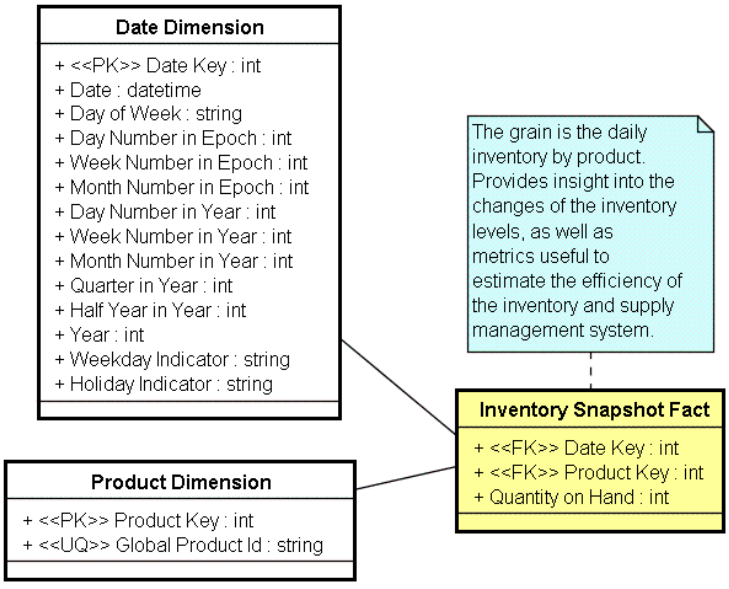

Figure 7. Inventory Snapshot Fact Table diagram.

3.2.6 Inventory Snapshot Fact Table

The *Inventory Snapshot Fact Table* (Figure 7) holds information about stock levels for every product detailed at the level of a single day. This data could be derived by combining information available in the *Negotiation Fact Table* and the *Supply Fact Tables* by comparing the amount of items supplied and sold. However, due to the awkwardness and potential inefficiency of such solution, a helper table storing snapshots of the inventory has been introduced.

The most important use of the data stored in the *Inventory Snapshot* table should be considered calculating stock management performance metrics. The data in the table comes from periodically aggregated information about sales and supplies.

4. CONCLUDING REMARKS

In this paper we have discussed challenges and solutions involved in gathering and efficiently storing historical data concerning all aspects of functioning of an e-shop, which is a part of a model agent-based e-commerce system. After a brief description of the system design we have focused on management of knowledge for use in data mining and automatic decision making. We have listed possible approaches to gathering information about the state of the system and explained decisions made while designing the module. We have also discussed the communication scenarios resulting in generation of usable information. Next, we have presented a complete data mart design for easily accessible storing of the data about the system events, along with basic examples of knowledge possible to extract from the stored data. Currently, we have completed the initial implementation of the data mart and are in final testing of data extraction and storage supporting processes. The next step will be integrating it with the remaining parts of the system and looking into utilization of the above described knowledge extracting procedures.

REFERENCES

Ansari, S. et al, 2001. Integrating E-Commerce and Data Mining: Architecture and Challenges. *Proceedings of the 2001 IEEE International Conference on Data Mining*. San Jose, CA, USA, p. 27.

Anthes, G., 2005. Modeling Magic: IT-based operations research builds better supply chains at Procter & Gamble. *Computerworld*, Feb 07, 2005.

Bădică, C. et al., 2006. Towards Trust Management in an Agent-based E-commerce System - Initial Considerations. *Proceedings of the MISSI 2006 Conference*. Wroclaw, Poland, pp. 225-236.

Bădică, C. et al., 2007. Developing a Model Agent-based E-commerce System. In: Jie Lu et. al. (eds.) *E-Service Intelligence - Methodologies, Technologies and Applications*, pp. 555-578. Springer, Berlin, 2007.

Bădică, C. et al., 2007. Implementing Rule-Based Automated Price Negotiation in an Agent System. *Journal of Universal Computer Science*, Vol. 13, No. 2, pp. 244-266.

Ganzha, M. et al, Implementing Commodity Flow in an Agent-Based Model E-commerce System. *Proceedings of the PPAM Conference.* in press

Ganzha, M., et al., 2004. JADE-based Multi-agent E-commerce Environment: Initial Implementation. *Analele Universităţii din Timişoara, Seria Matematică-Informatică*, Vol. XLII, 2004, pp. 79-100.

Gawinecki M. et al., 2006. Managing Information and Time Flow in an Agent-based E-commerce System. *Proceedings of the Fifth International Symposium on Parallel and Distributed Computing.* Timisoara, Romania, pp. 352-359.

Gawinecki M. et al., 2007. Introducing interaction-based auctions into a model agent-based e-commerce system — preliminary considerations. Proceedings of the EATIS Conference. Faro, Portugal.

Hays C. L., 2004. What Wal-Mart Knows About Customers' Habits. *The New York Times*; Nov 14, 2004.

Kimball, R. and Ross, M., 2002. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, 2nd Edition*; John Wiley & Sons, New York, NY, USA.

Kohavi, R. et al., 2004. Lessons and Challenges from Mining Retail E-Commerce Data. *Machine Learning*, Vol. 57, No. 1-2, pp. 83–113.

Kohari, R. and Provost, F., 2001. Applications of Data Mining to Electronic Commerce. *Data Mining and Knowledge Discovery*, Vol. 5, No. 1-2, pp. 5-10.

Paprzycki, M and Ganzha, M., 2007. Adapting Price Negotiations to an E-commerce System Scenario. *Proceedings of the CISIM Conference*. Ełk, Poland, pp. 380-386.

Perez, J., C., 2005. Amazon Turns Ten. *PCWorld*. July, 2005.

Raghavan, 2005. N R S.*;* Data mining in e-commerce: A survey. *S¯adhan¯a,* Vol. 30, No. 2-3, pp. 275–289.

Serzysko, T. et al, 2007. Introducing Commodity Flow to an Agent-Based Model E-commerce System. *Proceedings of the 2007 IAT Conference*. Silicon Valley, CA, USA, pp. 294-298.