Trading Agents & Strategies

Wouter de Vries
whrvries@cs.vu.nl
MSc. thesis
31 August 2001

Intelligent Interactive Distributed Systems Group
Department of Artificial Intelligence
Division of Mathematics and Computer Science
Faculty of Sciences
Vrije Universiteit Amsterdam

This thesis contributes to comparing different trading strategies of multiple agents. First, a task model of the trading task is described including explicitely modelled strategies for all decisions made in the trading activity. The task model is incorporated to a software trading agent. A prototype agent is tested in a market situation. A small number of trading strategies is described.

Keywords: agents, trading, negotiation, strategies.

Supervisors:
Prof. Dr. Frances M.T. Brazier, frances@cs.vu.nl
Dr. Niek J.E. Wijngaards, niek@cs.vu.nl
Referee:
Dr. Benno J. Overeinder, bjo@cs.vu.nl

Table of Content

1 Introduction

Trading is a well-known human activity, usually involving two parties, a consumer and a merchant. From the perspective of a consumer, trading is a process in which the consumer tries to acquire goods to satisfy a certain need. From the perspective of a merchant, trading means selling products with as much profit as possible. In this thesis these two different perspectives are combined in one task model, and eventually in one model for a trading agent.

The trading agent described in this thesis, is a software agent that trades on a marketplace. The agent is capable of advertising, and buying and selling products. Trading is a process in which pro-active, reactive and social behavior, autonomy, and adaptivity are key features of its participants.

Section 1.1 describes the background of this thesis, section 1.2 describes the purpose of the thesis, section 1.3 describes the most important rules on the market, and section 1.4 describes the approach.

1.1 Background

The idea of creating a marketplace for trading computer parts by agents (Agent Mediated Marketplace, AMMP) was launched at the course Multi-agent systems in Complex Domains (MiCD) 2000, taught by prof. dr. F. Brazier and dr. N. Wijngaards. This marketplace is managed by a market-manager and consists of a clock, a bank, a storehouse, and a number of trading agents. Two types of agents are distinguished: merchant and consumer agents. Merchant agents have to earn as much money as possible, and consumer agents have to buy parts with a limited budget for the construction of a pre-specified computer (according to a wish list). Some market rules are further specified in section 1.3. The protocol and a detailed description of the market are described in Appendix A.

The objective of the course was for students to design agents that trade in the marketplace, which make use of strategies to fulfill their goals. The agents have to be more or less generic so strategies can be changed without building a completely new agent.

1.2 Purpose

The underlying purpose of this thesis is to compare different trading strategies in a market situation, using a multi-agent approach. This leads to the question:

How to model an agent in such a way that trading strategies can be tested?

To answer this question, a task model is developed in which strategies are explicitly modeled. On the basis of this task model, an agent model of a trading agent is developed. A prototype agent is constructed, which can trade on the Agent Mediated Marketplace and in which trading strategies can be plugged in.

The trading task is refined in such a way that each subtask may have a number of strategies. All these local strategies may be combined, resulting in a large number of different trading strategies.

A trading agent is an agent based on the generic agent model explained in section 5, using the task model of trading described in section 3. Using this task model, the agent's strategies are relatively easy to change, new strategies may be plugged in and the agent may choose its own strategy or change it when it is not satisfactory.

With little adjustment, the trading agent to be designed may function on a different platform, since all components, including input and output components, may be exchanged for others, while the agent still functions correctly.

1.3 Market rules

The Agent Mediated Marketplace, as any other marketplace, has certain rules, constraints, and a communication protocol. A detailed description can be found in Appendix A, but the main points are also highlighted in this section. The market used to test the trading agents and strategies is created for the course Multi-agent systems in Complex Domains 2000/2001, and developed by cooperation of both teachers and students.

The products traded on the market are computer parts. A product is of a specific category (e.g., mouse). The performative attributes of a product are summarized in one attribute, the performance index. The higher the performance index, the better the product.

Two types of agents are present on the market: consumers and merchants. Each agent starts with a number of products in stock and some money. Consumers are given a list of products, a so-called wish list.

The goal of the consumers is to acquire all the products on their wish list. Merchants have to make as much money as they can by selling the products they have in stock. Both products and wish lists are received at the start of the market, so agents don't know what to expect before they enter the market.

Money and products are stored in the bank and in the storage. Agents can receive information about their own bank balance and products. A market manager is present to make sure agents don't cheat, i.e. make a deal and don't deliver the product or pay the money.

Messages are used for communication with other agents and the manager. The market is turn based, so messages send in turn T are received in turn T+1. An immediate reply is received in turn T+2. A clock is present to tell what turn it is, and how many turns are to be expected. The market may come to an end.

At the start of a turn each agent receives the messages that were send to it the previous turn, and all agents start reasoning. When an agent is done reasoning, it signals the manager, and then waits for the other agents to finish. In case an agent takes unusually long to reason, the manager intervenes.

1.4 Approach

The first step to designing and implementing a trading agent is to choose the design method. The need for meta-level reasoning results in the choice for DESIRE (DEsign and Specification of Interacting REasoning components). DESIRE is a formal knowledge-level modeling and specification framework for knowledge-intensive (multi-agent) systems [Brazier, Dunin-Keplicz, Jennings and Treur, 1995, 1997; Brazier, Jonker and Treur, 1998].

This approach leads to the construction of a task model of the trading task in which is described what subtasks are to be done while trading, what information these tasks need and produce, and how this information is exchanged between the tasks. These subtasks (or

processes) are then modeled as components of an agent, resulting in an agent model, in this case of a trading agent.

This functional design is implemented in the Object Oriented language Java ™ (http://java.sun.com) using blackboards.

Section 2 discusses a literature study about negotiation and trading, on which the trading task is based. Section 3 discusses the task model of trading, and section 4 describes how strategies are modeled in this task model. Section 5 describes the agent model, which is based on the task model, and section 6 the detailed design of a trading agent, for the implementation. Section 7 discusses a number of conclusions, and section 8 discusses recommendations and future research.

This thesis will not formulate test criteria for agents, or strategies, or evaluate the agents on the marketplace. Section 7.8 describes a limited number of aspects to consider for such a test.

2 Trading and Negotiation

This section describes a literature research on trading and negotiation. Section 2.1 describes a number of related definitions, section 2.2 describes the basis of the trading task model.

2.1 Definitions

This section describes a number of definitions of important concepts for this thesis. These concepts are defined by pair wise comparison: trading vs. negotiation, marketplace vs. auction, brokering vs. searching, buyer vs. seller, and consumer vs. merchant. Finally agents and strategies are briefly defined.

2.1.1 Trading vs. Negotiation

The main difference between trading and negotiation is best noticeable after describing the six stages in Consumer Buying Behavior [Guttman, Moukas & Maes, 1998; Guttman & Maes. 1998]. These are summarized in figure 2.1

1 Need Identification
2 Product Brokering
3 Merchant Brokering
4 Negotiation
5 Purchase And Delivery
6 Product Service And Evaluation

Figure 2.1. Six Fundamental stages guiding consumer behavior.

The stages are described in section 2.2. The figure is shown in this section to make it visible that negotiation is a part of trading. Based on these six stages, negotiation and trading are defined as follows:

- Negotiation is the process where buyer and seller tell each other their price, and then try to reach a mutually accepted price for the good.
- Trading is the whole process described in these 6 stages, including negotiation.

To make things more complex, there is also the term bargaining. The Object Management Group (OMG) defines bargaining as: bilateral and multi-lateral negotiation [Benyoucef & Keller, 2000].

- bilateral: two parties (one-to-one)
- multi-lateral: more than two parties (many-to-many)

In this thesis the term bargaining is avoided, as it means negotiating. Negotiation in this thesis is bi-lateral, since no other agent knows that two agents are negotiating about an object. Communication is kept private, except messages explicitly sent to other recipients as well. The trading, however, is multi-lateral, since agents may trade with as many other agents as they want.

The OMG also defines bargaining as:

- distributive: one attribute is negotiable (usually the price). The parties have opposing interests. (minimizing and maximizing)

- integrative: multiple attributes of the item are negotiable (price quality, delivery date) not necessarily opposing interests (they try to optimize different attributes).

In this thesis only price negotiation is possible. Extensions to integrative (multi-attribute) negotiations are mentioned in section 8.

Benyoucef and Keller [2000] use the term *Combined Negotiation* for the trading of many goods at the same time. For each product and for each negotiation partner they use one agent. All agents stay in contact with the CNSS (Combined Negotiation Support System). This resembles what is done in this thesis, as the CNNS is integrated in one agent, resulting in one agent that is able to conduct multiple negotiations at one time.

2.1.2 Marketplace vs. Auction

Defining what an *auction* is, is not easy, since there are many types of auctions. This thesis will use the definition by McAfee & McMillan [1987]: An auction is a market institution with an explicit set of rules determining resource allocation and prices on the basis of bids from the market participants. An auction has a number of properties that varies per type. An auction can be one sided or double, open or sealed, first price or n^{th} price, decreasing or increasing, repeated or one-shot, discriminatory or not, and uniform or not (URL: http://www.agorics.com/Library/Auctions/terms.html). Some examples are described below.

At a *marketplace*, multiple merchants sell multiple products and instances of products to multiple consumers. Both sides have a starting price and adapt their price to come to a mutually accepted price. Marketplaces may use a fixed-price system, but for this thesis a dynamic price system is used.

The differences between marketplaces and auctions are summarized in table 2.1.

Auctions are often mistaken for a marketplace. Kasbah [Chavez & Maes, 1996] e.g., is described as being a marketplace. Kasbah is, however, a set of double sided Dutch auctions. Each seller enters Kasbah with one product, sets a price and lowers it. Buyers enter Kasbah searching one product, set a price and raise it. Each buyer and seller conducts multiple auctions until the product is sold or acquired, or the price limit is reached. Based on the properties of auctions and marketplaces used in this thesis, Kasbah is indeed a set of auctions, as sellers on Kasbah sell one product only.

Auction	**Marketplace**
One merchant	Multiple merchants
One product per merchant	Multiple products per merchant
Multiple consumers	Multiple consumers
One side changes price, the other accepts or declines (one-side auctions)	Both side change prices to come together in the middle by so called haggling.

Table 2.1. Difference between auction and marketplace.

As illustration a number of example auction types (URL: http://www.knowledgemedia.org/netacademy/glossary.nsf/kw_id_all/839) and marketplace properties are named below.

Definitions of auction types:

- Single Sided Auctions
  - The English Auction
    An ascending auction where all bidders gather at the same time in the same place to

bid, and the auctioneer solicits progressively higher bids from the audience until only one bidder is left. The winner claims the item, at the price he last bid.
    - The First-Price Sealed Bid Auction
      Bidders submit a single, irrevocable, sealed bid. The bids are opened simultaneously, and the winner is the highest bidder, who claims the item at the price he bid.
    - The Dutch Auction
      A continuous descending auction where bidders can see the current price and must decide if they wish to purchase at that price or wait until it drops. The winner is the first bidder at the current price.
    - The Vickrey Auction
      Similar to the first-price sealed bid auction - the winner is the highest bidder. However, the winner only pays the amount of the second-highest bid.
- Double sided auctions
    - The Continuous Double Auction
      Both sellers and buyers submit bids, which are then ranked highest to lowest to generate demand and supply profiles. Trades are initiated as soon as matches are detected. On detection of a match, the auction clears at the price of the existing bid, and generates a new price quote.
    - The Periodic Double Auction
      The call market collects bids over a specified interval of time, then clears the market at the expiration of the bidding interval. The market maker sets a clearing price, where highest buy price equals the lowest sell price.
- Combinatory auctions: bids are made on combinations of goods and services [Benyoucef & Keller, 2000].

Marketplace properties:
- Fixed price: the price is fixed, there is no negotiation. (with a possibility for merchant brokering to check who offers the product cheapest)
- Dynamic price: a proposal is followed by a counter proposal until accepted.
- Bilateral bargaining: two parties are involved in the negotiation process.
- Multi-lateral bargaining: multiple parties are involved in the negotiation process.

Because of the diversity of auction types, it is hard to pinpoint exactly what is an auction and what is a marketplace, especially since the definition of an auction contains the words *market institution*. The main difference between an auction and a marketplace is that an auction has one seller, selling one product or group of products at once, while a marketplace consists of a number of sellers each selling multiple, different products.

2.1.3 Brokering vs. Searching

Table 2.1 mentions product and merchant brokering. Clurman, Foley, Guttman and Kupres [1997] define *brokering* as the matching of buyers and sellers for the purpose of getting to a deal. Matching two parties is a process performed by a third party, the broker or matchmaker.

This thesis lets buyers find sellers and visa versa. They have to *search* for themselves. This resembles a real marketplace, where buyers wander around in search of a seller that fits their preference. In this thesis there is no brokering, but consumers and merchants have to find each other by sending and reading advertisements.

2.1.4 Buying vs. Selling, Consumer vs. Merchant

The members of each of these two pairs are different enough. One important assumption is made in this thesis, however. Therefore all four definitions are described before the particularity is described.

Buying: acquiring a product in exchange for money.

Selling: acquiring money in exchange of a product.

Consumer: a person (or agent) that needs to acquire products in order to fulfill a wish list (In the context of this thesis). A consumer already has a number of products to start with.

Merchant: a person (or agent) that has as goal to earn as much money as possible, by selling products.

The particularity is that both consumers and merchants may buy and/or sell. The consumer gets more money to buy a product on its wish list, when a product is sold which is considered to be obsolete. A merchant may want to buy products cheap to sell them expensive.

2.1.5 Agent

A large number of definitions of agents are available. For this thesis, the definition of Wooldridge & Jennings [1995] is used:
An agent is a hardware or (more usually) software-based computer system that enjoys the following properties:

- autonomy: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- social ability: agents interact with other agents (and possibly humans) via some kind of agent-communication language;
- reactivity: agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the internet, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;
- pro-activeness: agents do not simply act in response to their environment, they are able to exhibit goal-directed behavior by taking the initiative.

2.1.6 Strategy

When trading products on a marketplace, decisions have to be made. A *strategy* is seen as the manner in which a situation is handled in which a decision has to be made.

Trading strategies are about:

- Deciding what product to buy or sell based on price, personal interest, and utility.
- Deciding about the negotiation party based on price, history with this party, trust, friendship.
- Deciding about opening bid: the outer value of the price interval, a price just next to the outer value, or an even more reasonable price.
- Deciding about number of negotiations on one product. Messages cost money, but a good deal is also important.
- Keeping the message costs in mind.
- Making a big plan at the start or take it one product at a time.
- Negotiation:
  - Play an easy, medium, hard, or impossible negotiation. A huge number of strategies is available for negotiation.
  - Torment others by promising thing you don't have (keep message cost and reputation in mind).
  - Changing the product in the middle of a negotiation.
- Saving message cost by saving up messages, while closing a negotiation as soon as possible.

All these decision have consequences. There may be a strategy about how soon the overall trading strategy is adapted to try to optimize the results in the future, what strategy is to be

used when it is not certain it will improve the situation. Choices have to be made at each level of the trading task.

2.2 Basis for Trading task model

The task model described in section 3 is based on the six stages in Consumer Buying Behavior summarized in figure 2.1, combined with the general commerce framework [Clurman, Foley, Guttman & Kupres, 1997] shown in table 2.2.

Seller		**Buyer**
Identify potential customers	—	Identify potential sellers
Identify customer needs	←	Identify own needs
Inform potential customer	→	Evaluate alternatives
Obtain order	←	Place order
Deliver thing	→	Receive thing
Receive payment	←	Make payment
Provide customer service	→	Use thing

—	These symbols indicate the information flow between buyer and seller (none, to seller, to buyer)
←	
→	

Table 2.2. The General Commerce Framework [Clurman, Foley, Guttman & Kupres, 1997].

The aspects mentioned in both models are named, followed by an explanation why they are used or rejected.

Identifying sellers and customers is a process done automatically in the Agent Mediated Marketplace. When an agent registers at the manager, a list of other agents present in the marketplace is received where each agent is marked as being a consumer or a merchant.

Need identification is done by giving each agent their budget, and their products in stock, and by giving a wish list to each consumer agent. Merchant agent have to make as much money as they can. Merchant agents have of determine the needs of the consumers by advertising and reading advertisements of consumers. This leads immediately to the aspect of product and merchant brokering, discussed in section 2.1.3. Agents have to search for themselves, by advertising and reading advertisements. Based on information retrieved in this way, they select what product to trade (evaluate alternatives) and with whom they trade.

The marketplace used for this thesis does not use fixed prices. Placing and receiving orders, as described in table 2.2, does not happen before a negotiation has taken place. The reason for incorporating negotiation in the task model is that negotiations are very interesting and numerous strategies may be tested.

When a deal is made, the product and money change hands. Delivery, service and usage are not modeled in the trading task of this thesis. Though interesting extensions may be made by adding aspects such as customer service or delivery time and cost, they are beyond the scope of this thesis.

The trading process needs to be managed, and a history of the whole process has to be maintained by the agent. Subtracting and adding the aspects mentioned above from and to the models one may arrive at a task model described in section 3.

It is important to note that the *negotiation* described in this thesis is *bilateral*. Other agents than the two involved in a specific negotiation are not even aware of this negotiation, because they do not receive any of the messages send regarding this negotiation. The *trading* described in this thesis is *multi-lateral*. Each agent can start a negotiation with any other agent present on the marketplace.

2.3 Interesting concepts

This section describes some interesting concepts found in literature, that

According to [Clurman, Foley, Guttman & Kupres, 1997] there are also 3 concepts that support the brokering & negotiation:

- ontology: the rules of the deal
- personalization: the information about your preferences
- reputation & trust: the information that you have about the other party in the deal

These concept are all mentioned in this thesis: appendix A describes the structure, language, rules, and punishments of the market, and all possible problems and solutions and extra possibilities, buyer and seller receive clear instructions at the start of the market, and agents have to find out for themselves whether the other party is a good trading partner.

3 Trading as a Task

A perspective on the activity of trading is to model trading as a task, resulting in a task model. A task may be refined in a number of subtasks, also called task composition. By composing tasks into subtasks, process hiding is achieved, and strategies may be added at each level of the task hierarchy. There may, for example, be different strategies for selecting a suitable negotiation partner. In order to adjust only the strategy for this subtask in a straightforward manner, the strategies presented in the subtask responsible of selecting negotiation partners are adapted. To facilitate this approach the task Trading is refined as deep as possible.

Section 4 describes in more detail how strategies are integrated in the task model. This section describes the task model of the trading task. Section 3.1 describes the requirements and the basis of the task model, section 3.2 describes the main subtasks of the trading task. Sections 3.3 through 3.8 describe all subtasks in more detail.

3.1 Requirements

The task model is constructed to form the basis of both the merchant and the consumer agent; both types of agents are able to buy and sell in a marketplace. Each subtask in the task model is a decision point at which a number of (strategic) choices are made. The task model is constructed in such a way that strategies are explicitly modeled.

The basis of the task model is formed by the consumer behavior model [Guttman & Maes, 1998] and the General Commerce Framework [Clurman, Foley, Guttman & Kupres, 1997]. In both models trader's needs and the potential trading party have to be identified. The trading task model is constructed with the assumption that the trader has to search a trading partner, and therefore this task has to be modeled. No broker is present on the market to couple traders.

Finding a trading partner is done by looking for potential trading partners in advertisements and, when no partner is found, sending advertisements. The latter process is a pro-active process and is therefore an independent task. Neither of the aforementioned models includes this task, because they do use broker.

A number of aspects may be open for negotiation. Mostly this is the price of the product, but there are more negotiable aspects such as delivery time and costs, extra features, extra service, more products with a group discount and many more. This thesis describes the price aspect, but the task model makes other aspects also possible. The whole process has to be managed and all steps have to be recorded, so the trader may learn from its mistakes. Delivery, product service and evaluation are not explicitly modeled. In real life, a merchant is judged on these are aspects, but in the simulation of the market, they are not present. They do, of course, make a nice expansion possibility, but are, for now, beyond the scope of the thesis.

Keeping track of all events and forming and changing strategies are tasks not mentioned in literature, because they're not special for trading only. Since they have to be performed for and during trading, they are modeled in the task model, as History Maintenance and Trading Process Coordination. There are no separate tasks for handling messages. These tasks are trivial and not a part of the actual trading, and are therefore not modeled. They are only

necessary for the type of communication used in this thesis. Figure 3.1 shows the task hierarchy in detail.

Multi-Object-Multi-Lateral-Trading
1. Product Selection
    1.1. Determination Of Possible Products
    1.2. Select Products
2. Party Selection
    2.1. Determination Of Potential Parties
    2.2. Select Parties
        2.2.1. Price Comparison
        2.2.2. Past Negotiation Evaluation
3. Advertisement Determination
4. Negotiation
    4.1. Negotiation Start
        4.1.1. Determination Of Opening Proposal
        4.1.2. Send Opening Proposal
    4.2. Bid Determination
        4.2.1. Determination Of Next Bid
        4.2.2. Send Bid
    4.3. Deal Closure
        4.3.1. Send Deal Confirmations
        4.3.2. Process Deal Consequences
    4.4. Negotiation Management
        4.4.1. Negotiation Analysis
            4.4.1.1. Estimation Of Strategy Other Party
            4.4.1.2. Estimation Of End Price
            4.4.1.3. Continuity Analysis
            4.4.1.4. Formulation Of Analysis Of Negotiation
        4.4.2. Determination Of Own Negotiation Strategy
5. History Maintenance
6. Trading Process Coordination
    6.1. Own Characteristics
    6.2. Trading Process Analysis
        6.2.1. Evaluation Of Trading Strategy
        6.2.2. Evaluation Of Inventory
        6.2.3. Evaluation Of Goals
        6.2.4. Evaluation Of Negotiations
        6.2.5. Evaluation Of Deals
        6.2.6. Evaluation Of Wishlist
        6.2.7. Evaluation Of Total Budget
        6.2.8. Evaluation Of Price Intervals
        6.2.9. Evaluation Of History
        6.2.10. Evaluation Of Incoming Advertisements
        6.2.11. Evaluation Of Participants
        6.2.12. Formulation Of Analysis Of Trading Process
    6.3. Market Price Estimation
    6.4. Price Limit Determination
    6.5. Determination Of Trading Strategy

Figure 3.1. Task Hierarchy, also available as unfold in appendix C..

The task model of the trading task must satisfy the following requirements

- The task model shall be a realistic view on trading.
- The task model shall support trading done by both consumers and merchants.
- The task model shall support both buying and selling behavior of an agent.

- The task model shall model trading with multiple trading parties.
- The task model shall model trading with multiple products to trade.
- The task model shall support the search for trading partners that want to trade a specific product. An subtask dealing with advertisement sending has to be present for this purpose.
- The trading task shall be constructed in such a way that strategies for all aspects of the trading task are explicitly modeled.
- The task model shall have a sub-process to evaluate strategies in order to learn whether a strategy functions good or bad.

Sections 3.2 through 3.8 describe all tasks in this hierarchy in detail.

3.2 Multi-Object-Multi-Lateral-Trading

The task model is designed to support trading of more than one object at the same time (multi-object) with more than one trading partner (multi-lateral). It may also be used for trading of only one object, or bi-lateral trading. The tasks Product Selection or Party Selection are than simply skipped, respectively.

This section describes the main subtasks of the trading process in section 3.2.1, the task control in section 3.2.2, and the information flow in section 3.2.3.

3.2.1 Composition of Trading Task

The trading task is decomposed in a number of subtasks, i.e. Product Selection, Party Selection, Advertisement Determination, Negotiation, History Maintenance, and Trading Process Coordination. Figure 3.2 shows these subtasks and the information flow between them.

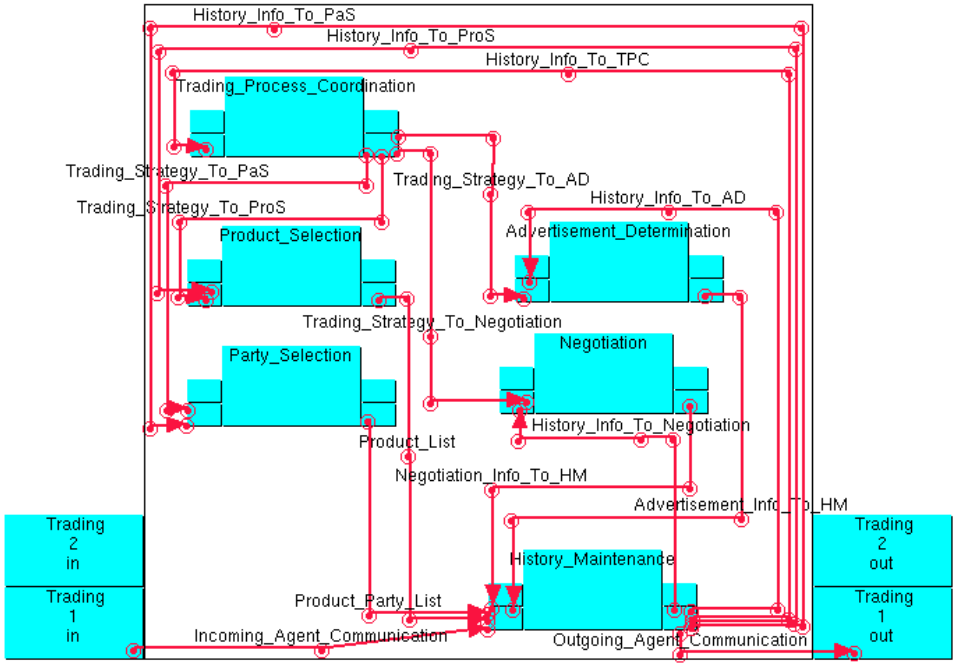

Figure 3.2. The Trading task with information flow.

1. Product Selection

Product Selection includes making a selection of all products that are to be traded. Trading all products at the same time may not always be an option due to message costs or limitations. In such a case a selection of a limited number of products has to be made. The Trading Strategy decides how the selection is made, and the decisions are made on the basis of History Info. The result is a list of a number of products that may be traded this turn. Since both consumers and merchants may buy and sell, the Product List has both “to buy” and “to sell” products.

The refining of this task is described in more detail in section 3.3.

In: Trading Strategy, History Info.

Out: Product List.

2. Party Selection

The task Product Selection determines per product which parties are suitable for trading the product. Restricting the number of trading partners saves time and money. History Info and Trading Strategy are needed for selecting appropriated trading partners from the incoming advertisements and the product list. For each product a party list is created. If the agent cannot select a suitable party with the information it has this turn, the agent may send advertisements to get this information. These advertisements are created in Advertisement Determination. The result of this task is a list of products, with a list of trading parties for each product.

The refining of this task is described in more detail in section 3.4.

In: Trading Strategy, History Info.

Out: Product-Party List.

3. Advertisement Determination

Advertisement Determination determines which advertisements are to be sent. The agent may not have sufficient information about the other agents. This task gives other agents information about your wishes and intentions by sending advertisements. Other agents may use this information to help you achieve your goals. The History Info has information about what product advertisements are sent, and the Trading Strategy determines which agents receive advertisements. The results of this task are advertisements to be sent.

This task has no subtasks.

In: Trading Strategy, History Info.

Out: Advertisement Info.

4. Negotiation

The task Negotiation handles everything regarding a negotiation, from the first message to deal closure. The task is activated only once per turn for every negotiation that received a message or is started this turn. The negotiation task involves the start, the bidding, or the closure of a negotiation. Negotiation means in this thesis haggling about the price of a product, but it is also possible to negotiate about more attributes. History Info is needed to follow the history of a negotiation and use information about a negotiation partner. The Trading Strategy is needed to form a negotiation strategy, or end the negotiation. The Negotiation Info contains outgoing messages and information about the progress of the negotiation.

The refining of this task is described in more detail in section 3.5.

In: Trading Strategy, History Info.

Out: Negotiation Info.

5. History Maintenance

The history of the observed information inside and outside the agent is stored selectively in order to use the information and to learn from it. This task is put central in the task model. Output information of one task is input information for another tasks, information about the incoming and outgoing messages is also needed by some tasks. As History Maintenance records all this information, it might as well be the central task of the task model and deal with the Incoming and Outgoing Agent Communication.

The Product List, Product-Party List, Advertisement Info, and Negotiation Info produced by the basic trading tasks are recorded in History Maintenance. Messages produced by these tasks are sent to the agent concerned. The trading management task, Trading Process Coordination, produces the Trading Strategy, which is also stored. The Trading Strategy also has a strategy for the History Maintenance task in which is described how information has to be stored, and how long it has to be stored.

The agent may use the History Info to see consequences of strategies used, compare these with the expected results, and adapt strategies. Everything is stored with a precise timestamp, not only the turn on the market, but also sub-time steps in the agent. Turns and time steps are described in section 3.2.2.

This task has no subtasks.

In: Incoming Agent Communication, Trading Strategy, Negotiation Info, Product List, Product-Party List, Advertisement Info.

Out: Outgoing Agent Communication, History Info.

6. Trading Process Coordination

This task analyses the situation and determines the Trading Strategy to be followed. It uses History Info to determine what went right and wrong and uses this information to alter the trading strategy or leave it as it was. Since everything that happens is first stored in the History Maintenance, it is always up to date and can thus be used for determining the trading strategy.

This task is described in more detail in section3.6

In: History Info.

Out: Trading Strategy.

3.2.2 Task Control

The market is *turn* based. Each turn, the messages send in the previous turn are delivered, deals are executed (products and money change hands), and agents are made awake.

The trading task is divided into *time steps*. One time step is comparable with one turn on the marketplace. When the agent is made awake, the agent's time step starts. In one time step, the agent receives a number of messages, it reasons about these messages and creates new messages that may help the agent to reach its goal, i.e. selling as much products as possible with a high profit, or buying pre-specified computer parts. At the end of the time step, after all messages are sent, the agent informs the market manager that it is done reasoning for this turn. The agent's time step ends, but the market turn continues until all agents are done reasoning. The market clock tells what turn it is. The agent's time step counter corresponds with the clock.

A time step is divided into sub-time steps. A sub-time step starts with History Maintenance and Trading Process Coordination. At the beginning of each sub-time step, Trading Process Coordination determines what has to be done in the upcoming sub-time step. Only one task other than History Maintenance and Trading Process Coordination is performed during each sub-time step.

History Maintenance records everything that is done in the previous sub-time step, or if it is the first sub-time step of the current time step, stores all incoming messages. This includes previously used trading- and negotiation strategies, all incoming- and outgoing messages and all conclusions that have been drawn by all tasks. All this information is stored with the precise timestamp (including sub-step).

Trading Process Coordination determines which other tasks the agent has to perform, how these tasks have to be performed and in what order these tasks have to be performed this time step and this sub-time step.

This results in the task control loop shown in figure 3.2:

```
new time step
        History Maintenance
        // new sub time step starts
        while (Trading Process Coordination != end time step)
                Product Selection OR
                Party Selection OR
                Advertisement Determination OR
                Negotiation

                History Maintenance
        end while
end time step
```

Figure 3.1. Task Control Loop described in pseudo code.

3.2.3 Information Types and Information Flow

This section first describes the information types used by and sent between the main tasks of trading. This is followed by a description of the information flow.

- Trading Strategy: Trading Strategy holds the trading strategy. This information type determines which task is done at a certain time, and how the task is done. For each task, this information type has a sub type. For the main tasks, described in section 3.2, these are Product Selection Strategy, Party Selection Strategy, Advertisement Determination Strategy, Negotiation Guideline, and History Maintenance Strategy. History Maintenance does not only have to store all information, including strategies, but receives a strategy on how to maintain all information.
    - Product Selection Strategy: This is the strategy used to select a suitable list of products to trade.
    - Party Selection Strategy: This is the strategy used to select a suitable list of parties for products to trade.
    - Advertisement Determination Strategy: This is the strategy used to determine advertisements.
    - Negotiation Guideline: This is the strategy used for the formation of the negotiation strategy.

  - History Maintenance Strategy: This is the strategy used for the maintenance of the information types and the messages.

- Product List: The products the agent wants to trade are specified in this information type. It is a limited list that is created in Product Selection, so not all possible products are in it, only the ones the agent wants to get started on this turn.
- Product-Party List: Product-Party List is a list of products, and each product has a list of potential negotiation parties. Here also the list is limited. The list is created in Party Selection.
- Advertisement Info: Advertisement Info consists of advertisements constructed in Advertisement Determination. This information is transported to History Maintenance.
- Agent Communication: Agent Communication consists of all messages that are sent and received.
  - Incoming Agent Communication: Incoming Agent Communication consists of all messages from all other agents that are sent to this agent. It consists of as many sub information types as there are messages to receive. Transaction messages from the manager are also in this information type. Other messages from the manager are about the world and are handled by world info.
  - Outgoing Agent Communication: Outgoing Agent Communication consists of all messages to all other agents that are send to these agents. It consists of as many sub-information types as there are messages to sent. Transaction messages to the manager are also in this information type
- Negotiation Info: Negotiation Info consists of all information concerning the negotiations. Everything that is determined and all the results of that are in this information.
- History Info: the observed information in the marketplace, and in the agent, is stored with a precise timestamp in History Info. Negotiations are not only stored as a sequence of messages, but also in a more structured manner, in a Negotiation Path. In the structure of such a Negotiation Path the following information is stored:
  1. The identification of a specific negotiation
  2. The product (described in detail)
  3. The negotiation partner
  4. The start time
  5. All bids with the exact time and the maker of the bid

Stored like this, it is easier to draw conclusions about a negotiation.

History info also keeps track of what strategy has been used, how this strategy was formed, and what the consequences were of using this strategy. This way the agent may learn from previous actions by using this information in Trading Process Coordination.

The information types described above are sent from one task the another. Incoming Agent Communication is stored in History Maintenance and is processed into History Info. This is how the agent communication is received by all other tasks. History Info also keeps records about all negotiations, current and closed. This is described in more detail in section 3.9 (History Info). All components use ate least a part of History Info and their output is transferred back to History Maintenance. Trading Process Coordination uses History Info to create the Trading Strategy. This is used by all components, even History Maintenance, for example for

strategies on how and how long certain information is stored. History Maintenance not only uses this information, but also stores it. History Maintenance also stores the internal information of Trading Process Coordination. This information is used to learn from previous strategies and their consequences. One possibility for example is that Trading Process Coordination determines a strategy and guesses the possible consequences. When after a number of turns the real consequences are known, these two pieces of data can be compared, and the agent can learn to predict better. All other components use the Trading Strategy for obvious reasons. Messages created by Advertisement Determination and Negotiation are stored in before they are send to other agents.

3.3 Product Selection

The task Product Selection consists of a number of subtasks. Section 3.3.1 describes the composition of the task Product Selection , section 3.3.2 describes the task control, and section 3.3.3 describes the local information types.

3.3.1 Composition of Product Selection

This section describes the composition of the task Product Selection. This task has the subtasks Determination of Possible Products and Select Products, as shown in figure 3.5. The subtasks are described below in more detail.

1. Product Selection

Product Selection is refined into Determination of Possible Products and Select Products because these subtasks can have different strategies. Splitting them up makes it easier to change the strategy of one of them, and not affect the other.

In: Trading Strategy, History Info.

Out: Product List.

1. Product Selection
    1.1. Determination Of Possible Products
    1.2. Select Products

Figure 3.5. Subtasks of Product Selection.

1.1. Determination Of Possible Products

Determination Of Possible Products generated a list of possible products that may be traded, using Trading Strategy and History Info.

In: Product Selection Strategy, History Info.

Out: Possible Product List.

1.2. Select Products

Select Products has to pick a number of products it wants to negotiate about from the Possible Product List. For this the task needs the Product Selection Strategy and the Possible Product List.

In: Product Selection Strategy, Possible Product List.

Out: Product List.

3.3.2 Task Control of Product Selection

The task control in this task is fairly straightforward. Determination of Possible Products determines a list of products you can buy or sell, depending on your goals. This list is reduced by Select Products to create a list of products that have priority, thus resulting in the product list that is used by other tasks.

3.3.3 Local Information Types of Product Selection

This section describes the local information types of the task Product Selection.

- Possible Product List: The information type Possible Product List expresses knowledge about the possible products that may be traded.

3.4 Party Selection

The task Party Selection consists of a number of subtasks. Section 3.4.1 describes the composition of the task Party Selection , section 3.4.2 describes the task control, and section 3.4.3 describes the local information types.

3.4.1 Composition of Party Selection

This section describes the composition of the task Party Selection. This task has the subtasks Determination Of Potential Parties and Select Parties, as shown in figure 3.6. These subtasks are described below in more detail.

2. Party Selection

Party Selection is refined in Determination Of Potential Parties and Select Parties because these subtasks require a different strategy. Changing the strategy of one task does not affect the other. The History Info includes the product list from Product Selection, because this task is done in another sub-time step. Between two sub-time steps, the History Info is updated.

In: Trading Strategy, History Info.

Out: Product-Party List.

2. Party Selection
    2.1. Determination Of Potential Parties
    2.2. Select Parties
        2.2.1. Price Comparison
        2.2.2. Past Negotiation Evaluation

Figure 3.6. Subtasks of Party Selection.

2.1. Determination of potential parties

First an agent has to search the market for potential trading parties, i.e. parties that may want to sell a product the agent wants to buy, or parties that may want to buy a product the agent wants to sell. This is done with the aid of History Info and Trading Strategy. This results in a list of potential parties. If no suitable party can be determined, the agent has to create advertisements to get the appropriate information or wait for advertisements of other agents.

In: Party Selection Strategy, History Info.

Out: Potential Product-Party List.

2.2. Select Parties

The list of potential parties created in Determination Of Potential Parties is reduced in this task. The Trading Strategy is used to select a number of good trading parties from the Potential Product-Party List. History Info is needed for information about the product, prices, and agents evaluated.

The task is refined in the tasks Price Comparison and Past Negotiation Evaluation. Both these tasks are a means to evaluate potential negotiation parties. Depending on the strategy, only one is used, or both. These subtasks are describe below in more detail.

In: Party Selection Strategy, Potential Product-Party List, History Info.

Out: Product-Party List.

2.2.1. Price Comparison

One way of selecting a party above another is to compare the amount of money they ask or offer for the product. The ones with the best prices for the agent's perspective are selected.

In: Party Selection Strategy, Potential Product-Party List, History Info.

Out: Product-Party List.

2.2.2. Past Negotiation Evaluation

Another way to compare parties is to see how previous negotiations went with each of them. Even with a price far away from yours, they could come out close to your price in the end.

In: Party Selection Strategy, Potential Product-Party List, History Info.

Out: Product-Party List.

3.4.2 Task Control of Party Selection

The task control of Party Selection is straightforward. Determination Of Potential Parties determines all possible parties that have or need the product you need or sell, respectively. Select Parties reduces this list by comparing prices, evaluating past negotiation with these parties, or both. The resulting list is used by other tasks.

3.4.3 Local Information Types of Party Selection

This section describes the local information types of the task Party Selection.

- Potential Product-Party List: The information type Potential Product-Party List expresses knowledge about the potential negotiation parties for a specific product.

3.5 Negotiation

The task Negotiation consists of a number of subtasks. Section 3.5.1 describes the composition of the task Negotiation , section 3.5.2 describes the task control, and section 3.5.3 describes the local information types.

3.5.1 Composition of Negotiation

This section describes the composition of the task Negotiation. This task has the subtasks Negotiation Start, Bid Determination, Deal Closure, and Negotiation Management, as shown in figure 3.7.

4. Negotiation

This task is refined into the subtasks Negotiation Start, Bid Determination, Deal Closure, and Negotiation Management because these tasks are used in different phases during negotiation process. A negotiation has a start, in which a first bid is determined and send, and the consequences of the start of the negotiation are stored. This is followed by a series of bids by both parties, leading to the closure of a deal, ideally. When a deal is closed, the consequences have to be processed, because the result is not always directly observable. This process has to managed, in order to compare the results of this negotiation with that of another, to check whether to product is not already bough or sold in another negotiation, etc.

In: Trading Strategy, History Info.

Out: Negotiation Info.

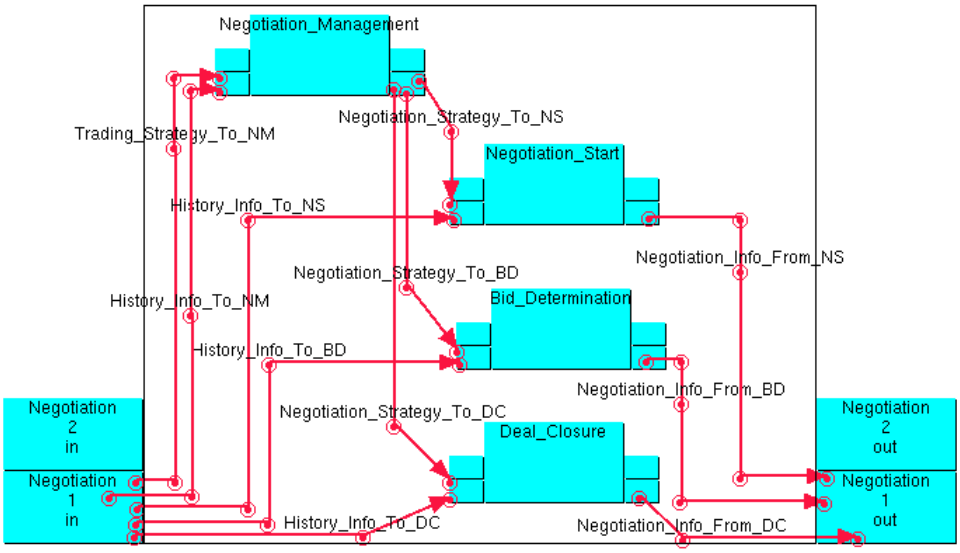

Figure 3.7. The Negotiation task with information flow.

4.1. Negotiation Start

This task is responsible for the start of the negotiation. An opening bid is determined with the aid of the Negotiation Strategy and is sent to the other party. The state of the negotiation is sent to other tasks in the form of Negotiation Info. This reason for not integrating this task into Bid Determination is the fact that the strategy needed for determining the first bid is different than the strategy used for determining the next bid in an ongoing negotiation.

This task has the subtasks Determination Of Opening Proposal and Send Opening Proposal. Determination Of Opening Proposal determines the first bid that is done. This knowledge is expressed in the information type Opening Proposal that is transferred to Send Opening Proposal. Send Opening Proposal constructs and sends a message expressed in the information type Negotiation Info.

In: History Info, Negotiation Strategy.

Local Information Type: Opening Proposal: Opening Proposal expresses knowledge about the opening proposal.

Out: Negotiation Info.

4.2. Bid Determination

This task determines a bid on the basis of the Negotiation Strategy. This is sent to the other party. The state of the negotiation is sent to other tasks in the form of Negotiation Info. This task has the subtasks Determination Of Next Bid and Send Bid. Determination Of Next Bid determines the next bid that is done. This information is expressed in the information type Next Bid that is transferred to Send Bid. Send Bid constructs and sends a message expressed in the information type Negotiation Info.

In: History Info, Negotiation Strategy.

Local Information Type: Next Bid: Next Bid expresses knowledge about the next bid.

Out: Negotiation Info.

4.3. Deal Closure

In this task a deal is closed using the Negotiation Strategy. This information is sent to the other party. The state of the negotiation is sent to other tasks in the form of Negotiation Info. This task has the subtasks Send Deal Confirmation and Process Deal Consequences. Deal confirmations do *not* have to be *determined* in the way that a bid has to be determined. They *follow* from the negotiation. When the next bid is the same as the last bid of the negotiation party, than an agreement is reached, and the specification of the deal are already known. They have to be send to the other party or the market manager, which is done by Send Deal Confirmation. Process Deal Consequences processes the consequences of the deal, because they cannot yet be observed. No local information types are necessary, because both tasks use the incoming History Info and Negotiation Strategy and put their information in Negotiation Info.

In: History Info, Negotiation Strategy.

Out: Negotiation Info.

4.4. Negotiation Management

This task does the same for Negotiation as Trading Process Coordination does for the trading process. It analyses the situation and determines the Negotiation Strategy to be followed. For this it uses the Trading Strategy and it needs the History Info to compare results of other negotiations. This task is described in more detail in section 3.7.

In: Trading Strategy, History Info.

Out: Negotiation Strategy.

3.5.2 Task Control of Negotiation

The first subtask that is handled is Negotiation Management. This task determines what other subtask of Negotiation is done this sub-time step and how it is done. This information is send to that subtask in Negotiation Strategy. Only one subtask, other than Negotiation Management, is done at once. Negotiation can of course be conducted more than once per time step, but only one of its subtasks are handled at a time.

3.5.3 Local Information types of Negotiation

This section describes the local information type of the task Negotiation.

- Negotiation Strategy: Negotiation Strategy contains the negotiation strategy used for this negotiation.

3.6 Trading Process Coordination

The task Trading Process Coordination consists of a number of subtasks. Section 3.6.1 describes the composition of the task Trading Process Coordination, section 3.6.2 describes the task control, and section 3.6.3 describes the local information types.

3.6.1 Composition of Trading Process Coordination

This section describes the composition of the task Trading Process Coordination. This task has the subtasks Own Characteristics, Trading Process Analysis, Market Price Estimation, Price Limit Determination, and Determination Of Trading Strategy, as shown in figure 3.9. These tasks are described below in more detail.

6. Trading Process Coordination

Trading Process Coordination is refined into the subtasks Own Characteristics, Trading Process Analysis, Market Price Estimation, Price Limit Determination, and Determination Of Trading Strategy. The manner in which the trading process is coordinated, depends on the characteristics of the trader. The trading process needs to analyzed in order to determine, or adjust the strategy to be used. The market price is to be estimated before a price limit is determined. Both of the latter tasks need the strategy to be executed, and are needed to determine the strategy for other tasks.

In: History Info.

Out: Trading Strategy.

6 Trading Process Coordination
- 6.1. Own Characteristics
- 6.2. Trading Process Analysis
- 6.3. Market Price Estimation
- 6.4. Price Limit Determination
- 6.5. Determination Of Trading Strategy

Figure 3.9. Subtasks of Trading Process Coordination.

6.1. Own Characteristics

Own Characteristics gives the own characteristics of an agent. This is information about whether the agent is a consumer or not, if it is eager to sell or buy, and more characteristics that may influence the agent's behavior.

In: -

Out: Own Characteristics.

6.2. Trading Process Analysis

This task analyses the trading process, using History Info and Own Characteristics. The task produces the results in Trading Process Analysis. The Trading Strategy determines the order in which the subtasks are done. The Trading Strategy is a sub-information type of History Info.

In: Own Characteristics, History Info, Trading Strategy.

Out: Trading Process Analysis.

6.3. Market Price Estimation

In this task the agent estimates the market price of each item in the inventory and on the wishlist. Each time step this market price can be revised (or refined) to a better estimation. The incoming Trading Strategy is from Determination Of Trading Strategy and is constructed this sub turn. More information about the task control and information flow in Trading Process Coordination is found in section 3.6.2.

In: Trading Strategy.

Out: Estimated Market Prices.

6.4. Price Limit Determination

For each product a price limit is determined. This has to be done in Trading Process Coordination and not in Negotiation Management. Trading Process Coordination has the whole picture of the trading process and this is needed to determine the price limits, for spending more or less money on one product has direct consequences on available money for other products. The incoming Trading Strategy is from Determination Of Trading Strategy and is constructed this sub turn. More information about the task control and information flow in Trading Process Coordination is found in section 3.6.2.

In: Trading Strategy.

Out: Price Limits.

6.5. Determination Of Trading Strategy

Determination Of Trading Strategy determines the trading strategy.

In: Trading Process Analysis, Estimated Market Prices, Price Limits.

Out: Trading Strategy.

3.6.2 Task Control of Trading Process Coordination

Before a strategy can be determined, the progress of the overall trading task has to be analyzed and the agents own characteristics have to be determined. This is done in Trading Process Analysis and in Own Characteristics. After this the Trading Strategy is determined in Determination Of Trading Strategy. This information is used in Market Price Estimation and in Price Limit Determination where the market price for each product of the wishlist and in the inventory is estimated and a price limit for each of these products is determined. This information is fed back into Determination Of Trading Strategy because this information is needed this turn for other tasks.

The control flow is summarized in figure 3.10.

```
Start Trading Process Coordination
        Own Characteristics
        Trading Process Analysis
        Determination Of Trading Strategy
        Market Price Estimation
        Determination Of Trading Strategy
        Price Limit Determination
        Determination Of Trading Strategy
End Trading Process Coordination
```

Figure 3.10. Control Flow of Trading Process Coordination depicted in pseudo code.

3.6.3 Local Information Types of Trading Process Coordination

This section describes the local information types of the task Product Selection.

- Own Characteristics: This is information about whether the agent is a consumer or not, or is the agent eager to buy or sell, does it get anxious at the end of the market, etc. Every possible characteristic may be put in it.
- Trading Process Analysis: A complete analysis of all factors that have something to do with the trading process.
- Estimated Market Prices: An estimation of the current market price of each product that is on the wishlist and in stock. This estimation is based on information the agent has gathered so far.
- Price Limits: Depending on the total budget, the personal utility, and maybe other info, a maximum (or minimum) price of all products on the wishlist and in stock is set. This may change as other products cost less or more, or have been sold for less or for more money, or as the market comes to an end and the agent needs to buy or sell certain products in a hurry.

3.7 Negotiation Management

The task Negotiation Management consists of a number of subtasks. Section 3.7.1 describes the composition of the task Negotiation Management, section 3.7.2 describes the task control, and section 3.7.3 describes the local information types.

3.7.1 Composition of Negotiation Management

This section describes the composition of the task Negotiation Management. The task has the subtasks Negotiation Analysis and Determination Of Own Negotiation Strategy, as shown in figure 3.11. This section also describes the subtasks of Negotiation Analysis. All these tasks are described below in more detail.

4.4 Negotiation Management
- 4.4.1. Negotiation Analysis
  - 4.4.1.1. Estimation Of Strategy Other Party
  - 4.4.1.2. Estimation Of End Price
  - 4.4.1.3. Continuity Analysis
  - 4.4.1.4. Formulation Of Analysis Of Negotiation
- 4.4.2. Determination Of Own Negotiation Strategy

Figure 3.11. Subtasks of Negotiation Management.

4.4 Negotiation Management

The task Negotiation Management is refined into Negotiation Analysis and Determination Of Own Negotiation Strategy.

In: Trading Strategy, History Info.

Out: Negotiation Strategy.

4.4.1 Negotiation Analysis

Negotiation Analysis analyses the negotiation process. In order to determine a negotiation strategy, the negotiation has to be analyzed. The Trading Strategy is needed for the subtasks of Negotiation Analysis and History Info expresses knowledge about the history of this

negotiation. Negotiation Analysis is composed of the subtasks Estimation Of Strategy Other Party, Estimation Of End Price, Continuity Analysis, and Formulation Of Analysis Of Negotiation. These subtasks are described below in this section, in more detail.

In: Trading Strategy, History Info.

Out: Negotiation Analysis.

4.4.1.1 Estimation Of Strategy Other Party

Estimation Of Strategy Other Party uses the History Info to estimate the negotiation strategy a negotiation partner is using. Estimating a strategy may be done in different ways, so Trading Strategy is needed to determine in what manner this task is executed.

In: History Info, Trading Strategy.

Out: Estimated Strategies Of Other Party.

4.4.1.2 Estimation Of End Price

Estimation Of End Price estimates the possible end price of the negotiation. The strategy of the other party (Estimated Strategies Of Other Party) combined with the own negotiation strategy (Trading Strategy), and the last bid (History Info) is used to estimate the end price of the negotiation.

In: Trading Strategy, History Info, Estimated Strategies Of Other Party.

Out: Estimated End Price.

4.4.1.3 Continuity Analysis

Continuity Analysis determines if it is necessary, wise or useful to continue with a negotiation. The Trading Strategy may have information about deals made or negotiations close to an end, which may make this negotiation obsolete. The end price that is estimation may be to high compared to the price limit. The strategy of the other party that is estimation may be judged to be unfair. All these factors considered this task gives the sign to end the negotiation or go on with it.

In: Trading Strategy, Estimated End Price, History Info, Estimated Strategies Of Other Party.

Out: Continuity Analysis Info.

4.4.1.4 Formulation Of Analysis Of Negotiation

Formulation Of Analysis Of Negotiation formulates the result of the analysis. All information transferred to this task is put together in the Negotiation Analysis.

In: Trading Strategy, Estimated Strategies Of Other Party, Estimated End Price, Continuity Analysis Info.

Out: Negotiation Analysis.

4.4.2 Determination Of Own Negotiation Strategy

Using the Negotiation Analysis, the Negotiation Strategy can be determined with the aid of Historical Info and the information type Negotiation Guidelines, which are a part of the Trading Strategy. These Negotiation Guidelines are needed to consider the big picture (i.e., the trading task) in the determination of the negotiation strategy.

In: Negotiation Analysis Info, Historical Info, Trading Strategy.

Out: Negotiation Strategy.

3.7.2 Task Control of Negotiation Management

The task control of Negotiation Management is straightforward. First a negotiation is analyzed, then the negotiation strategy is altered or it stays as it was in Determination Of Own Negotiation Strategy. The task control of Negotiation Analysis is somewhat more complex. First the strategy of the negotiation partner is estimated by looking at the history of the negotiation. With this information, the end price of this negotiation is estimated. In Continuity Analysis this is compared with the price limits that are made. Other criteria may be checked here to make sure this negotiation may be continued. In Formulation Of Analysis Of Negotiation, the Negotiation Strategy is formulized.

3.7.3 Local Information types of Negotiation Management

- Negotiation Analysis
  - Estimated Strategies Of Other Party: Information about the possible strategies of other agents, estimated by this agent.
  - Estimated End Price: Information about the possible end price of the current negotiation, estimated by this agent
  - Continuity Analysis: Advise about whether to continue with this negotiation or not.

3.8 Trading Process Analysis

The task Trading Process Analysis consists of a number of subtasks. Section 3.8.1 describes the composition of the task Trading Process Analysis, section 3.8.2 describes the task control, and section 3.8.3 describes the local information types.

3.8.1 Composition of Trading Process Analysis

This section describes the composition of the task Trading Process Analysis. The subtasks are shown in figure 3.12. The subtasks are described below in more detail.

6.2. Trading Process Analysis

Trading Process Analysis is composed of the subtasks Evaluation Of Trading Strategy, Evaluation Of Inventory, Evaluation Of Goals, Evaluation Of Negotiations, Evaluation Of Deals, Evaluation Of Wishlist, Evaluation Of Total Budget, Evaluation Of Price Intervals, Evaluation Of History, Evaluation Of Incoming Advertisements, Evaluation Of Participants, and Formulation Of Trading Process Analysis. Each of these subtasks analyses a part of the history of the trading process, except Formulation Of Trading Process Analysis. Formulation Of Trading Process Analysis combines the information produced by the other tasks in the information type Trading Process Analysis.

In: Own Characteristics, History Info, Trading Strategy.

Out: Trading Process Analysis.

6.2. Trading Process Analysis
- 6.2.1. Evaluation Of Trading Strategy
- 6.2.2. Evaluation Of Inventory
- 6.2.3. Evaluation Of Goals
- 6.2.4. Evaluation Of Negotiations
- 6.2.5. Evaluation Of Deals
- 6.2.6. Evaluation Of Wishlist
- 6.2.7. Evaluation Of Total Budget

6.2.8. Evaluation Of Price Intervals
6.2.9. Evaluation Of History
6.2.10. Evaluation Of Incoming Advertisements
6.2.11. Evaluation Of Participants
6.2.12. Formulation Of Trading Process Analysis

Figure 3.12. Subtasks of Trading Process Analysis.

6.2.1. Evaluation Of Trading Strategy

Evaluation Of Trading Strategy compares the expected results of the used trading strategy with the actual results. The outcome of the comparison is expressed in Trading Strategy Analysis.

In: History Info.

Out: Trading Strategy Analysis.

6.2.2. Evaluation Of Inventory

Evaluation Of Inventory checks the stock of the agent to see what products it (already) owns, in order to determine what has to be sold or bought.

In: History Info.

Out: Inventory.

6.2.3. Evaluation Of Goals

Evaluation Of Goals checks what goals there are, whether they are met or not, and what has to be done to meet them.

In: Own Characteristics.

Out: Goals.

6.2.4. Evaluation Of Negotiations

Evaluation Of Negotiations evaluates past and ongoing negotiations. It does not need Negotiation Info as input, because this information is already present in History Info.

In: History Info.

Out: Negotiations Evaluation.

6.2.5. Evaluation Of Deals

Evaluation Of Deals checks what objects are acquired or sold in ongoing deals. The products are not yet in stock, so they cannot be observed as a “product in possession”, but there is no need to start a negotiation about them, because they will be in possession in a couple of turns.

In: History Info.

Out: Deals.

6.2.6. Evaluation Of Wishlist

Evaluation Of Wishlist checks the wishlist to see if all the products on the wishlist are acquired. This task is only executed by consumers, since merchants don’t have a wishlist.

In: Own Characteristics.

Out: WishList.

6.2.7. Evaluation Of Total Budget

Evaluation Of Total Budget checks the bank account. A consumer has to divide its budget over all or the most important products of its wishlist. A merchant has to check its budget to see if it has made enough money. All agents have to take into account the money that is needed for communication.

In: History Info

Out: Budget.

6.2.8. Evaluation Of Price Intervals

Evaluation Of Price Intervals checks the price intervals given at the start of the market. In order to make a reasonable estimation on the amount of money to ask or bid for a products, a starting point is needed. Humans have world knowledge to cope with this problem, computer programs do not.

In: History Info.

Out: Price Intervals.

6.2.9. Evaluation Of History

Evaluation Of History evaluates the entire history of the trading process in order to learn from mistakes made, or strategies well chosen.

In: History Info.

Out: History Evaluation.

6.2.10. Evaluation Of Incoming Advertisements

Evaluation Of Incoming Advertisements evaluates incoming advertisements to see if there is a product that can be bought or sold, for a good price.

In: History Info.

Out: Incoming Advertisements Evaluation.

6.2.11. Evaluation Of Participants

Evaluation Of Participants determines the group of agents to whom certain advertisements are send.

In: History Info.

Out: Participants Info.

6.2.12. Formulation Of Analysis Of Trading Process

Formulation Of Trading Process Analysis combines all evaluations to form the Trading Process Analysis. Trading Process Analysis contains more information than all incoming information types, because it also includes links between and conclusion about the incoming information.

In: Trading Strategy Analysis, Inventory, Goals, Negotiations Evaluation, Deals, WishList, Budget, Price Intervals, History Evaluation, Incoming Advertisements Evaluation, Participants Info.

Out: Trading Process Analysis.

3.8.2 Task Control of Trading Process Analysis

All subtasks of Trading Process Analysis are handled in no particular order, except the Formulation Of Analysis Of Trading Process. This task is done last.

3.8.3 Local information types of Trading Process Analysis

- Trading Process Analysis: A complete analysis of all factors that have something to do with the trading process.
    - Trading Strategy Analysis: An analysis of consequences of the used trading strategy.
    - Inventory: Information about the possessions of the agent.
    - Goals: The goals the agent has set. This may be to buy a pre-specified computer or sell the complete stock with a high profit. More detailed goals are possible.
    - Negotiations Evaluation: An evaluation of the used negotiation strategies.
    - Deals: Information about the deals that are made. Necessary for knowing what the agent needs to buy or sell.
    - WishList: Information about the computer parts an agent has to buy. This is only necessary for consumer agents.
    - Budget: An agent's total budget.
    - Price Intervals: Information about price intervals as they are given at the start of the market.
    - History Evaluation: A complete evaluation of the History Info.
    - Incoming Advertisements Evaluation: Information about advertisements sent and received.
    - Participants Info: Information about which agent is a consumer or a merchant. This is used for determining advertisements and selecting parties.

3.9 Requirement Assessment

This section discusses all requirements of the trading task model summed up in section 3.2 and how the task model satisfies them.

- *The task model shall be a realistic view on trading*. The task model is based on the consumer behavior model [Guttman & Maes, 1998] and the General Commerce Framework [Clurman, Foley, Guttman & Kupres, 1997]. It has a scientific foundation. There is one aspect of the task model that is not realistic, however. It is made to work in a specific market that is turn based. No market in real life is turn based.
- *The task model shall support trading done by both consumer and merchant*. Trading performed by consumers is different than trading performed by merchants. Both start with an amount of money, and a number of products, but consumers wish to acquire more or other products and sellers wish to make money by selling products. Both determine which product to trade, and whom to trade with. Both send advertisements and negotiate. The tasks are the same, but the goals and the strategies are different. The tasks are modeled in the task model. The task model can thus be used by both consumers and merchants.

- *The task model shall support both buying and selling*. For buying and selling the same tasks are used. The strategies and goals are different, but those are not modeled in the trading task model.
- *The task model shall be able to handle multiple trading parties*. The task Party Selection selects a small group of parties to start a negotiation with about a pre-selected product. Multiple negotiation may be employed, each negotiation with one negotiation partner.
- *The task model shall be able to handle multiple products to trade*. The task Product Selection selects a list of products about which a negotiation can be started. Multiple negotiation may be employed, each negotiation about one product.
- *The task model shall support the search for trading partners that want to trade a specific product. An subtask dealing with advertisement sending has to be present for this purpose*. The task Advertisement Determination determines advertisements based on the list of products that were selected for trading, and a list of trading parties that are thought to be suitable trading partners for these products.
- *The trading task shall be constructed in such a way that strategies for all aspects of the trading task may be explicitly modeled*. The task Trading Process Coordination analyses all aspects of the trading task and makes strategies for all subtasks of the trading task.
- *The task model shall have a sub-process to evaluate strategies in order to learn whether a strategy functions good or bad*. Trading Process Analysis analyses what the results were of a used strategy. In Determination Of Trading Strategy this information is used to alter strategy if necessary.

4 Trading Strategies

Strategies described in literature mostly concern negotiation. Not only that but it concerns usually negotiation about one product or negotiation with one negotiation partner. Strategies concerning altering strategies in the middle of a negotiation process are completely unheard of.

Yet humans do this all the time. They set goals, and change them when they seem too low or too high. When selling a computer a mouse is thrown in a for free, just to make the consumer think the deal just got better. Modeling the thought patterns to come to these actions is possible using the task model from section 3.

Section 4.1 discusses why all subtasks of trading need good strategies, section 4.2 describes a number of trading strategies from literature, and section 4.3 describes some additional strategies. Section 4.4 describes how strategies are plugged in and section 4.5 describes a simple example.

4.1 More than negotiation

Actions prior to negotiation need strategies just as much as the negotiation. Decisions made outside the negotiation have more influence on the outcome of trading than the use of complicated negotiation strategies. The following aspects, small, easy and insignificant as they may seem, may affect the trading process in a big way:

- The order in which products are traded. The first products that are acquired influence the amount of money available for the last products to be acquired.
- The selection of a negotiation partner. For one product, multiple potential negotiation partners may be available. Since message sending costs money, the number of negotiation parties should be kept low. Selecting the right party may save money on message sending and may result in a better price than starting a negotiation with a random negotiation partner.
- The selection of advertisements to be send. Advertisements are expensive to send, so a good strategy on advertising is needed to keep the costs low.
- Changes made in the strategies used for trading. Changing strategy during trading or negotiation may lead to better results and makes it harder for other agents to predict the next move. Making the decision to change strategy is based on information about the way you conduct your trading.

4.2 Example strategies from literature

This section sums up a number of example strategies from literature and shows where and how the strategy fits in the task model of section 3.

4.2.1 Simple price functions

Chavez and Maes [1996] describe the web-based system Kasbah, where users create autonomous agents that buy and sell goods on their behalf. One agent buys or sells one product, so no product selection in needed. Selling agents start by offering their product for the desired price (i.e., the highest price). They lower the price when the price is not accepted. When the desired selling date arrives, the price should be the lowest acceptable. Selling agents have a choice of three strategies of lowering their price: linear, quadratic, or cubic. Buying agents do exactly the opposite, with the opposite functions for price raising.

The strategies used for negotiation are simple functions, for price lowering or raising, which are chosen by the user. In the task model, Determination Of Next Bid (T 4.2.1) could make use of functions such as these to determine the next bid.

4.2.2 Utility function

Tete-a-tete [Guttman, Moukas & Maes, 1998] negotiates about more than price alone: warranty length and options, shipping time and cost, service contract, return policy, quantity, accessories / bundles, credits / loan options, payment option. Unfortunately, this is beyond the scope of this thesis.

The strategy used captures the constraints given by the user in a utility function. This utility function is used to negotiate an optimal deal.

4.2.3 Changing tactics

Matos, Sierra and Jennings [1998] mention a number of reasons for changing strategy while negotiating to buy a product. These strategies are time-dependent, resource-dependent, or behavior-dependent tactics. A combination of these tactics is a weighted-counter proposal.

- The time-dependent strategy models the fact that a consumer is likely to concede more rapidly as the negotiation deadline approaches.
- The resource-dependent strategy makes consumers willing to concede earlier when the quantity of the resource diminishes.
- The behavior-dependent strategy is based on imitation of the negotiation partner.

When a consumer is attracted to more than one of these strategies, a weighted counter proposal may be made by making a weighted combination of the three strategies.

Changing strategies during a negotiation is a process that requires meta-reasoning. In the task model the trading strategy, and thereby the negotiation strategy, is determined in Determination Of Trading Strategy (T 6.5)

4.3 Additional strategies

The strategy for selecting a trading or negotiation partner may depend on more aspects than choosing a partner on the aspect of starting price alone. In previous negotiations with this party, his strategy can be estimated from the bids he did. With this extra information, the end result of the negotiation can be guessed. The outcome may be that a party with a

higher start bid than another party reaches a lower end price because he has a bad strategy. The task model has a separate task for selection a negotiation partner on the basis of past negotiations (T 2.2.2).

A strategy for misleading other agents is advertising a product to sell, when you want to buy it. Agents reacting to this advertisement are now recognized as competition and share with you the prize they are willing to offer for the product. Desperate as you are to acquire the product, you can top these offers at an agent that really sells this product (or so you think). While you are misleading the competition and keep them busy with a negotiation that will go nowhere, in the hope that they will not continue the negotiation with the agent that does have the product, you are in a good position to acquire the product. There are a few drawbacks, however. You are spending money on messages, while it is not certain that this tactic is working. Furthermore, all agents start bidding with the lowest price of the price intervals, which are given at the start of the market. This strategy is in the task model of course determined in Determination Of Trading Strategy (T 6.5) and the advertisement determination is done in Advertisement Determination (T 3).

This last point leads to another strategy, which is always start bidding just below the upper bound when selling, or just over the under bound when buying. Agents using a simple selection of trading partners will pick you, because you are closer to their price than all agents taking the outer bound. When advertising to buy a product with a high performance index, a more realistic bid can be done, to gain a better chance of getting selected by merchants.

4.4 Strategy Plugging

This section describes how strategies in the task model are combined and how this fits in the task model. Described is what happens in the task model when a new sub task is added, or when a strategy is added that does not fit in the task model.

All strategies combined form the Trading Strategy. Each subtask has a strategy that is part of the Trading Strategy. The subtasks of Trading Process Control receive History Info in which the Trading Strategy is present, so they too receive Trading Strategy.

Trading Strategy is composed of Product Selection Strategy, Party Selection Strategy, Advertisement Determination Strategy, Negotiation Guidelines, and History Maintenance Strategy. With the aid of the Negotiation Guidelines, the Negotiation Strategy is determined in Negotiation Management. Negotiation Strategy is composed of Negotiation Start Strategy, Bid Determination Strategy, and Deal Closure Strategy.

The connection between the strategies is shown in figure 4.1.

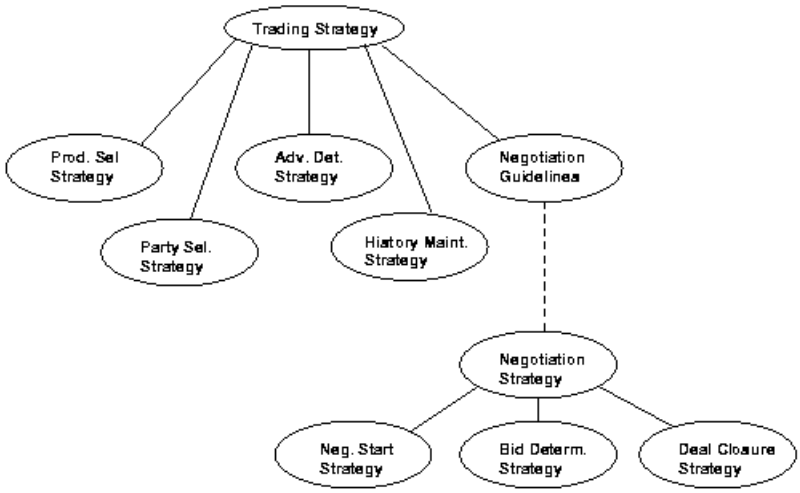

Figure 4.1. The connection between the strategies in the task model.

Strategies are (pre)formed in Trading Process Coordination. The strategies for negotiation are "preformed". This means that they are not completely formed in Trading Process Coordination, but that guidelines for forming them are constructed. Each negotiation has to take the whole trading process into account, and these guidelines transfer important information about the trading process to Negotiation Management, where the negotiation strategies are formed.

Subtasks receive the Trading Strategy and use the sub strategies meant for them. In the agent model, an improvement is made. In the task model the Trading Strategy is transferred to the deepest level. In the agent model components with subcomponents that each use a different part of the Trading Strategy split the Trading Strategy up into the sub-strategies needed by its subcomponent. This sub-strategy is then transferred to the each subtask. An example is described in section 5.8.1.

When a new subtask is inserted, a strategy has to be made for this subtask. The subtask may also influence other tasks. Therefore an extra evaluation task has to be inserted in Trading Process Analysis. The information provided by this evaluation task is used to form strategies used by the other tasks.

Inserting a new strategy that involves new tasks, e.g., a strategy where merchant cooperate to form a cartel, is possible. A new task has to be inserted, Cooperation Management, which might have subtasks. More importantly, this task needs to be analyzed in Trading Process Analysis, where also a new task is inserted, e.g., Evaluation Of Cooperation. With the information produced by the evaluation task, all strategies are formed, and these strategies are all influenced by this information, if they have to be.

4.5 Simple Agent Example

A simple example.

Let's consider a consumer agent with the following specifications:

- Money: 3000.

- Products:
    - Monitor, PI: 200.
    - Mouse, PI: 10.
    - Hard disk, PI: 100.
    - … and more to make a working computer.
- Wish list:
    - Monitor, PI: 500.
    - Mouse, PI: 60.
    - … and more to make a working computer.

The consumer agent has to have a working computer at the end of the market. It may sell computer parts to raise money and it may acquire parts with a Performance Index equal or higher than given in the wish list.

The costs per message are known at the start of the market, but it is unknown how much messages the agent needs to send in order to succeed. Therefore the total amount of money necessary for message sending is unknown. The total amount of money available for the products to be acquired is also unknown. It may be less than the starting amount because of the message sending, but maybe the old monitor and mouse can be sold to gain money.

It is clear from the start of the market that a lot of information is unknown, will change during the market, or may never be known. With the information that is known, all other information has to be deduced as good as possible. Decisions that are made depend much on the characteristics of the agent. The agent can take many risks, may drive a hard bargain or not.

At the start of the first turn, after receiving its messages, the agent analyses the trading process (T 6.2). The strategy is determined. This Trading Strategy determines what the next process is that the agent does.

A product has to be selected to be acquired. To agent may take a risk and start negotiations about the mouse *and* the monitor. The agent may want to see how trading develops with the small mouse and learn in order to make a better deal with the monitor. Or the agent wants to buy the big monitor, and see if there is enough money left to buy the relatively cheap mouse. The last option is to try and make money by selling the monitor and mouse, which will become superfluous when the new stuff is bought.

As is seen, one little choice can be made in a lot of different ways. There is no right choice. There are only choices that may turn out right or wrong.

The choice between buying and selling is made in Determination of Trading Strategy (T 6.5). The strategy for choosing what product to buy or sell is also made here. The actual selection of products is done in Select Products (T 1.2) after the possible products are determined in Determination Of Possible Products (T1.1).

The rest of the task model is handled the same way.

5 Agent Model

The task model of trading, described in section 3, was used in the design of a Trading Agent. In this section the agent model of the trading agent is described. Section 5.1 describes the generic agent model of Brazier, Jonker and Treur [2000] which is used as the basis for the model of the Trading Agent. Section 5.2 describes the mapping from the task model to the agent model and section 5.3 describes the agent model of the trading agent. Section 5.4 describes the information types of the trading agent.

5.1 Generic agent model

The agent model of the trading agent is based on the generic agent model proposed by Brazier, Jonker and Treur [2000]. This agent model is shown in figure 5.1. In this model, the agent has six main processes. For interaction with the world and other agents, the agent uses World Interaction Management and Agent Interaction Management. The agent maintains a history about the world and other agents in Maintenance of World Information and Maintenance of Agent Information respectively . The agent may reason about its own processes in Own Process Control and all tasks that are specific for the agent are handled in the component Agent Specific Task.

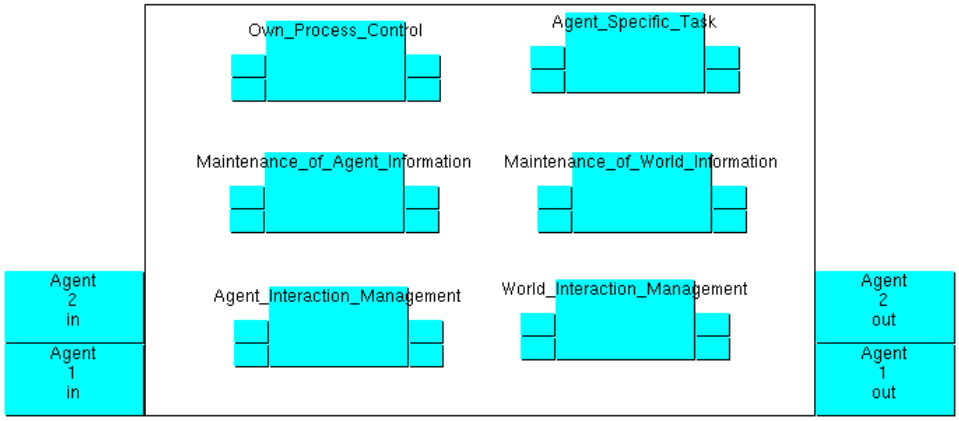

Figure 5.1. The Generic Agent Model and its six components.

This generic agent model forms the basis of the the agent model of the Trading Agent, which is described in section 5.3. First the mapping from the task model to the agent model is described.

5.2 Mapping of Task model to Agent Model

This section describes the mapping of the task model to the multi-agent model. In section 5.2.1 the tasks are mapped to the components and in section 5.2.2 the information types are mapped from the task model to the agent model.

5.2.1 Tasks to Components

This section describes the mapping of the task identified in section 3 to processes in the agent model. The subtasks Product Selection, Party Selection, Advertisement Determination, and Negotiation are mapped to components of the same name. These components are subcomponents of the component Trading. The Trading task is put into the component Agent Specific Task, since this is a task, which is specific for a trading agent. Trading Process Coordination and History Maintenance do not belong in this component. Trading Process Coordination reasons about the trading process in Trading and is therefore put into the component Own Process Control. History Maintenance is not a task specific for trading, but a process all agents need to perform, so it is a top level process.

Table 5.1 summarizes the mapping of the task model to the agent model.

Task	Sub-component	Component
Product Selection	Product Selection	Trading (AST)
Party Selection	Party Selection	
Advertisement Determination	Advertisement Determination	
Negotiation	Negotiation	
History Maintenance	Maintenance Of Agent Information	History Maintenance
	Maintenance Of World Information	
	Maintenance Of Own Information	
Trading Process Coordination	Trading Process Coordination	Own Process Characteristics
Own Characteristics	Own Characteristics	

Table 5.1. Mapping of the tasks in the task model to the components of the agent model.

5.2.2 Information Types

This section describes the mapping of the information types of the trading task described in section 3 to the information types of the Trading Agent. Trading Strategy is mapped to Trading Strategy, which has the same information, but in the agent model it has more sub-information types. This is described in more detail in section 5.8.1. Product List, Product-Party List, and Negotiation Info are put into Trading info. The components that have these information types as output, are subcomponents of Trading. Trading combines these outputs in Trading Info. Incoming Agent Communication is mapped to Communicated Info, and Outgoing Communication is mapped to Info To Be Communicated. Table 5.2 summarizes the mapping of information types.

Task information types	Agent sub information types	Agent information types
Trading Strategy		Trading Strategy
Product List	Product List	Trading Info
Product-Party List	Product-Party List	
Negotiation Info	Negotiation Info	
Incoming Agent Comminication		Communicated Info
Outgoing Communication Info		Info To Be Communicated
History Info		Historical Info

Table 5.1. Mapping of the information types from the task model to the agent model.

5.3 Agent Architecture

This section describes the main components of the Trading Agent. Section 5.3.1 compares the Trading Agent with the Generic Agent Model described in section 5.1. Section 5.3.2 gives an

overview of all components of the Trading Agent, section 5.3.3 describes the main components of the Trading Agent in more detail. Section 5.3.4 describes the task control, and section 5.3.5 describes the information types.

5.3.1 Comparison with Generic Agent Model

This section compares the Trading Agent model with the generic agent model described in section 5.2. In the Trading Agent model no World Interaction Management is used. Communication is handled through messages and no direct world observations are made. When the agent wishes to know something about the world, a message is send to the manager, who will send a message back with observation results. Messages are handled by Agent Interaction Management, so World Interaction Managemant is obsolete.

Maintenance of Agent Information and Maintenance of World Information are not in the top level of the Trading Agent, but in History Maintenance. This is done because there is also a Maintenance of Own Information that records all internal steps that are taken. Three history components at top level is too much, so they are placed in History Maintenance.

Brazier, Jonker and Treur [1996] propose an extra component for cooperation between agents, Cooperation Management. Trading, and especially negotiation, may be seen as a form of cooperation between two or several agents, trying to come to a mutual agreement about the price of an object. Another view, however, is that negotiation is a battle between agents, where each agent is trying to get the best possible deal. In this view, there is no cooperation at all. In this thesis the latter view is adopted. Cooperation Management is not used for the task of negotiation in this model.

The model may be expanded, however, to include cartel formation of merchants, boycotting consumers, or any other kind of cooperation. For these extensions the component Cooperation Management is suggested and will have to be reintroduced in the model. Consequences of adding this component to the agent model are discussed in section 7.4.

This results in the agent model shown in figure 5.2, with Own Process Control, Agent Interaction Management, History Maintenance, and Agent Specific Task. The main components are described in section 5.3.3.

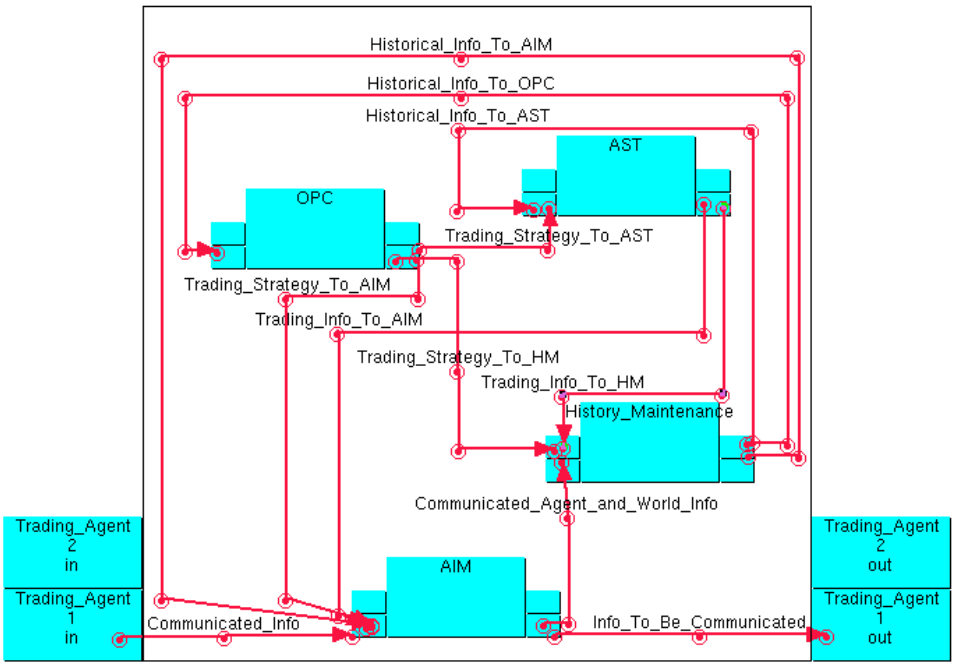

Figure 5.2. The top level of the trading agent.

5.3.2 Process Hierarchy

This section describes the hierarchy of components of the Trading Agent. The main components of the Trading Agent are Own Process Control, Agent Interaction Management, History Maintenance, and Agent Specific Task. The rationale for these components is given in section 5.3.1. The main differences between the tasks in the task model and the components in the agent model are highlighted in section 5.2.

In the task model, Own Characteristics is a subtask of Trading Process Coordination. Own Characteristics are not specific for trading or the evaluation of trading. They may infuence more than these processes alone, so in the agent model it is not a component of Trading Process Coordination, but a component of Own Process Control.

Tasks directly related to trading, Product Selection, Party Selection, Advertisement Determination, and Negotiation are components in the component Trading in Agent Specific Task. From there, the hierarchy is the same as the task model.

Figure 5.3 shows the components of the trading agent. A number of subcomponents are omitted from this figure for clarity reasons, but they do exist in the agent model, and are described in the task hierachy in figure 3.1 in section 3. The task model is also found in appendix C, as an unfold.

Trading Agent
A 1. Own Process Control
A 1.1. Own Characteristics (T 6.1)
A 1.2. Trading Process Control (T 6)
A 1.2.1. Trading Process Analysis « (T 6.2)
A 1.2.2. Market Price Estimation (T 6.3)
A 1.2.3. Price Limit Determination (T 6.4)
A 1.2.4. Determination Of Trading Strategy (T 6.5)
A 2. Agent Interaction Management
A 3. History Maintenance (T 5)
A 3.1. Maintenance of Agent Information
A 3.2. Maintenance of World Information
A 3.3. Maintenance of Own Information
A 4. Agent Specific Task
A 4.1. Trading
A 4.1.1. Product Selection « (T 1)
A 4.1.2. Party Selection « (T 2)
A 4.1.3. Advertisement Determination (T 3)
A 4.1.4. Negotiation (T 4)
A 4.1.4.1. Negotiation Start « (T 4.1)
A 4.1.4.2. Bid Determination « (T 4.2)
A 4.1.4.3. Deal Closure « (T 4.3)
A 4.1.4.4. Negotiation Management (T 4.4)
A 4.1.4.4.1. Negotiation Analysis « (T 4.4.1)
A 4.1.4.4.2. Determination Of Own Negotiation Strategy (T 4.4.2)

« : denotes subtasks. (see section 3)
A: denotes that it concerns the Agent model
T: denotes that it concerns the Task model

Figure 5.3. Components of the Trading Agent.

5.3.3 Process Composition of Trading Agent

This section describes the main components of the agent in detail. These components are Own Process Control, Agent Interaction Management, History Maintenance, and Agent Specific Task.

A 1. Own Process Control

Own Process Control is responsible for all processes in an agent, including the trading task, the formulation of a trading strategy and own characteristics of the agent. The Trading Strategy is determined after analyzing the Historical Info. The components of Own Process Control are described in section 5.4

In: Historical Info.

Out: Trading Strategy.

A 2. Agent Interaction Management

Agent Interaction Management manages the interaction with other agents. It receives messages from other agents in Communicated Info, and sends messages in Info To Be Communicated. Received messages are processed and this information is moved to History Maintenance as Communicated Agents and World Info. The Trading Strategy may contain strategic information about how to save costs on message sending. Trading Info contains

messages created by the subtasks of Agent Specific Task. Historical Info may assist this process with knowledge about the history.

In: Communicated Info, Trading Strategy, Trading Info, Historical Info.

Out: Info To Be Communicated, Communicated Agents and World Info.

A 3. History Maintenance (T 5)

History Maintenance records everything that happens, with a precise time stamp. The Communicated Agents Info, the Communicated World Info, and all decisions and outcomes of these decision (Trading Strategy and Trading Info) are recorded. The components of History Maintenance are described in section 5.5.

In: Communicated Agents and World Info, Trading Strategy, Trading Info.

Out: Historical Info.

A 4. Agent Specific Task

Agent Specific Task does the tasks that are specific for the agent, in this case trading. For trading, Historical Info is needed to determine in what stage certain processes are. Trading Strategy is needed to determine the next step. The components of Agent Specific Task are described in section 5.6.

In: Historical Info, Trading Strategy.

Out: Trading Info.

5.3.4 Process Control of Trading Agent

The Process Control of the Trading Agent is based on the task control described in the task model in section 3.2.2, with some slight modifications. The control is divides into steps and sub steps.

At the start of each turn, the mailbox comes in. The first thing the agent does is processing its mail. The mailbox is processed in Agent Interaction Management and the information is stored in History Maintenance. This new information is processed by Own Process Control, and Own Process Control stores its findings in History Maintenance. Own Process Control also determined what the agent should do next. This may be an agent specific task, or the end of the turn. In case of an agent specific task, this task is handled like Own Process Control determined. Own Process Control has information about how tasks may be processed and should learn what the concequences are of a specific way of doing a task. When a task is completed, all information is stored in History Maintenance and Own Process Control may choose another task. When a task produces a messages, this message is transported to Agent Interaction Management, which constructs a message and sends it to an agent, or the manager.

```
start of new time step
receive and process mailbox in Agent Interaction Management
store info in History Maintenance
        while (process info in Own Process Control != end turn)
                Agent Specific Task

                if necessary: send message in Agent Interaction Management
                store info in History Maintenance
        end while
store info in History Maintenance
end turn
```

Figure 5.4. Control loop of the agent.

5.3.5 Information Types and Links of Trading Agent

This section describes the information types that are used in the agent.

- Communicated Info: Communicated Info consists of all incoming mail. The mail received from other agents and the market manager is transported to Agent Interaction Management.
- Info to be communicated: Info to be Communicated consists of all outgoing mail. The messages are constructed in Agent Interaction Management and transported to other agents.
- Communicated Agent and World Info: Communicated Agent and World Info consists of information about other agents and the world. Communicated Agent and World Info has two subinformation types: Communicated Agent Info, and Communicated World Info. These subinformation types are described in section 5.5.3. Communicated Agent and World Info is moved to History Maintenance, where it is added to the Historical Info.
- Historical Info: Historical Info consists of all history of the other agents, the world and the agent itself. This information is transported from History Maintenance to Agent Specific Task and Own Process Control. It has subinformation types Agent Info, World Info, and Own State Info. These subinformation types are described in 5.5.3. Trading Strategy determines how long certain information is stored.
- Trading Strategy: Trading Strategy contains of information about the strategy to be followed. It has subinformation types that contain strategies for the subcomponents of the Trading Agent. The sub-information types of Trading Strategy are described in section 3.
- Trading Info: Trading Info contains information about messages to be send, decisions made at each level during trading (bids, products, parties, changes in trading strategy). Trading Info has sub information types Product List, Product-Party List, Advertisement Info, and Negotiation Info. These sub information types are described in section 5.8.3.

The links at top level and the information types are summarized in table 5.3.

Information link	From		To	
	Process	Information type	Process	Information type
Communicated Info	Agent	Communicated Info	AIM	Communicated Info
Info to be communicated	AIM	Info to be Communicated	agent	Info to be Communicated
Communicated agent + world info	AIM	Communicated agent + world info	MH	Communicated agent + world info
Historical Info to AIM	MH	Historical Info	AIM	Historical Info

Historical Info to OPC	MH	Historical Info	OPC	Historical Info
Historical Info to AST	MH	Historical Info	AST	Historical Info
Trading Strategy to aim	OPC	Trading Strategy	AIM	Own Characteristics + Trading Strategy Info
Trading Strategy to mh	OPC	Trading Strategy	MH	Trading Strategy
Trading Strategy to ast	OPC	Trading Strategy	AST	Trading Strategy
Trading info to aim	AST	Trading Info	AIM	Trading Info
Trading info to mh	AST	Trading Info	MH	Trading Info

Table 5.1. Specification of information exchange.

5.4 Own Process Control

The component Own Process Control consists of a number of subsomponents. Section 5.4.1 describes the composition of the components Own Process Control, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.4.1 Process Composition of Own Process Control

This section describes the process composition of the component Own Process Control. This component has the subcomponents Own Characteristics and Trading Process Coordination, as shown in figure 5.4. The subcomponents are described below in detail.

A 1. Own Process Control

Own Process Control is responsible for the analysis of the trading task, the formalization of the trading strategy and the own characteristics of the agent. This is done with the Historical Info. The components and information flow of Own Process Control is shown in figure 5.4.

In: Historical Info.

Out: Trading Strategy.

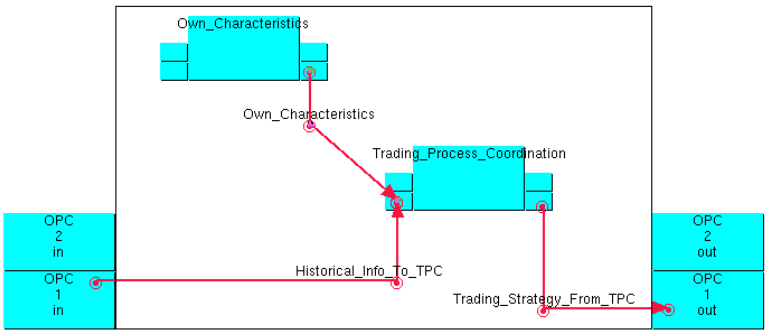

Figure 5.5. Subcomponent and information flow of Trading Process Coordination.

A 1.1. Own Characteristics (T 6.1)

In the agent model the own characteristics are separated from the rest of Trading Process Coordination. The reason for this is that the agents own characteristics may encompass more than trading behavior alone.

In: -

Out: Own Characteristics.

A 1.2. Trading Process Coordination (T 6)

Apart from the Own Characteristics, the rest of Trading Process Coordination is the same as in the task model. This component has subcomponents which are described in section 5.7.

In: Historical Info, Own Charcteristics.

Out: Trading Strategy.

5.4.2 Process Control of Own Process Control

The order in which the processes in Own Process Control are activated is as follows: first Own Characteristics determines the characteristics of the agent. These are used by Trading Process Coordination, which produces the Trading Strategy.

5.4.3 Information Types and Links of Own Process Control

- Own Characteristics: Own Characteristics contains the characteristics of the agent.

Table 5.4 summarizes which link transferres which information type.

Information Link	From		To	
	Process	Information Type	Process	Information Type
Own Characteristics	Own Characteristics	Own Characteristics	Trading Process Coordination	Own Characteristics
Historical Info to TPC	Own Process Control	Historical Info	Trading Process Coordination	Historical Info
Trading Strategy from TPC	Trading Process Coordination	Trading Strategy	Own Process Control	Trading Strategy

Table 5.1. Information link in Own Process Control.

5.5 History Maintenance

The component History Maintenance consists of a number of subsomponents. Section 5.4.1 describes the composition of the components History Maintenance, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.5.1 Process Composition of History Maintenance

This section describes the process composition of the component History Maintenance. This component has the subcomponents Maintenance of Agent Information, Maintenance of World Information, and Maintenance of Own Information, as shown in figure 5.4. The subcomponents are described below in detail.

A 3. History Maintenance (T 5)

This component record everything that happens, with a precise time stamp. The Communicated Agents Info is recorded by Maintenance of Agent Information. The Communicated World Info is recorded by Maintenance of World Information. All decisions and outcomes of these decision (Trading Strategy and Trading Info) are recorded by Maintenance of Own Information. The component is shown in figure 5.6.

In: Communicated Agents and World Info, Trading Strategy, Trading Info.

Out: Historical Info.

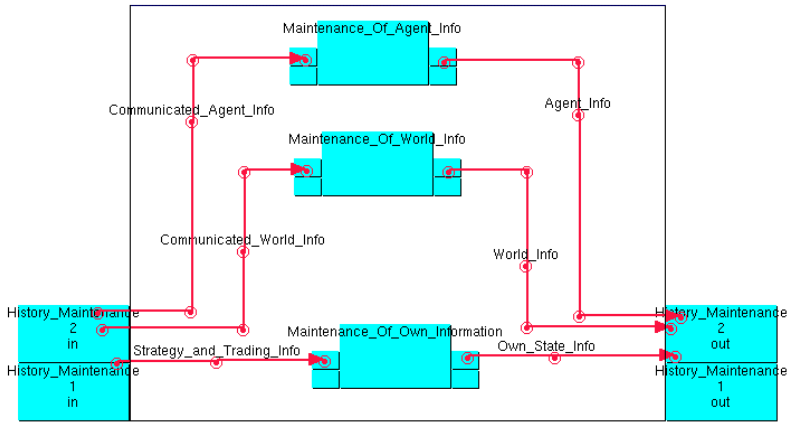

Figure 5.6. Subcomponent and information flow of History Maintenance.

A 3.1. Maintenance of Agent Information

Maintenance of Agent Information records all information about other agents.

A 3.2. Maintenance of World Information

Maintenance of World Information records all information about the world, ie. inventory, budget, and clock.

A 3.3. Maintenance of Own Information

Maintenance of Own Information records all decisions, results, and strategies of the other components.

5.5.2 Process Control of History Maintenance

The components of History Maintenance may theoretically be activated at the simultaniously, since they all process and produce different information and do not use information produced by one of the other components in History Maintenance.

5.5.3 Information Types and Links of History Maintenance

- Communicated Agent and World Info: Communicated Agent and World Info consists of information about other agents and the world. Communicated Agent and World Info has two subinformation types: Communicated Agent Info and Communicated World Info.
  - Communicated Agent Info: Communicated Agent Info consists of information about other agents. This is a sub information type of Communicated Agent and World Info.
  - Communicated World Info: Communicated World Info consists of information about the world. This is a sub information type of Communicated Agent and World Info.
- Historical Info: Historical Info consists of all history of the other agents, the world and the agent itself. This information type has the subinformation types Agent Info, World Info, and Own State Info.
  - Agent Info: Agent Info consists of all information known about other agents. This means all agents and what is known about each agent. This includes a pointer to the messages send by the agent, its possible negotiation strategy, and products it wants or needs.
  - World Info: World Info consists of all information known about the world. This means the time past and remaining, money in the bank, products in storage, and the normal

prices. Estimated market prices are also stored here, althought they are not observed, but estimated by the agent.

- Own State Info: Own State Info contains all history of the agent itself. This includes history of decision made in Trading Process Coordination. This is explicitly stored so the agent may learn from the things it has decided.

Table 5.5 summarizes which link transferres which information type.

Information Link	From		To	
	Process	Information Type	Process	Information Type
Communicated Agent Info	History Maintenance	Communicated Agent Info	Maintenance of Agent Info	Communicated Agent Info
Communicated World Info	History Maintenance	Communicated World Info	Maintenance of World Info	Communicated World Info
Strategy and Trading Info	History Maintenance	Trading Strategy, Trading Info	Maintenance of Own Information	Trading Strategy, Trading Info
Agent Info	Maintenance of Agent Info	Agent Info	History Maintenance	Agent Info
World Info	Maintenance of World Info	World Info	History Maintenance	World Info
Own State Info	Maintenance of Own Information	Own State Info	History Maintenance	Own State Info

Table 5.1. Information links in History Maintenance.

5.6 Agent Specific Task

The component Agent Specific Task consists of a number of subsomponents. Section 5.4.1 describes the composition of the components Agent Specific Task, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.6.1 Process Composition of Agent Specific Task

This section describes the process composition of the component Agent Specific Task. This component has one subcomponent, Trading, as shown in figure 5.7. This subcomponent is described below in detail. No sections for task control and information types are necessary because the compent is very simple.

A 4. Agent Specific Task

Agent Specific Task does the tasks that are specific for the agent, in this case Trading. The agent could be expanded to have more tasks than Trading alone. The trading task is therefore put into a component of its own, in stead of being immediately split up in Agent Specific Task. The component is shown in figure 5.8.

In: Historical Info, Trading Strategy.

Out: Trading Info.

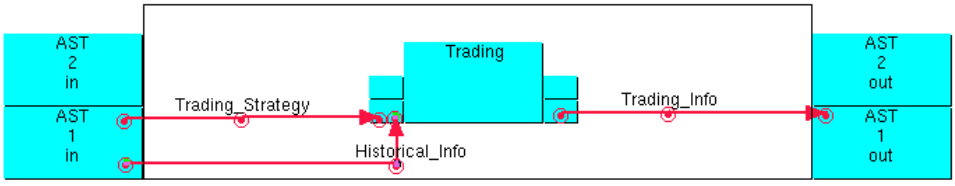

Figure 5.7. Subcomponent and information flow of Agent Specific Task.

A 4.1. Trading

As seen in section 5.2, the tasks Product Selection, Party Selection, Advertisement Determination and Negotiation are mapped onto components of the same name and are brought together in the component Trading. Information type Trading Strategy is split up in Product Selection Strategy, Party Selection Strategy, Advertisement Determination Strategy, and Negotiation Strategy. The component is shown in figure 5.8.

In: Historical Info, Trading Strategy.

Out: Trading Info.

5.7 Trading Process Coordination

The component Trading Process Coordination consists of a number of subsomponents. Section 5.4.1 describes the composition of the components Trading Process Coordination, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.7.1 Process Composition of Trading Process Coordination

This section describes the process composition of the component Trading Process Coordination. This component has the subcomponents Trading Process Analysis, Price Limit Determination, Market Price Estimation, and Determination of Trading Strategy, as shown in figure 5.4. The subcomponents are described below in detail.

A 1.2. Trading Process Coordination (T 6)

Trading Process Coordination is composed because the coordination process consists of subprocesses which make use of different strategies.

In: Historical Info, Own Charcteristics.

Out: Trading Strategy.

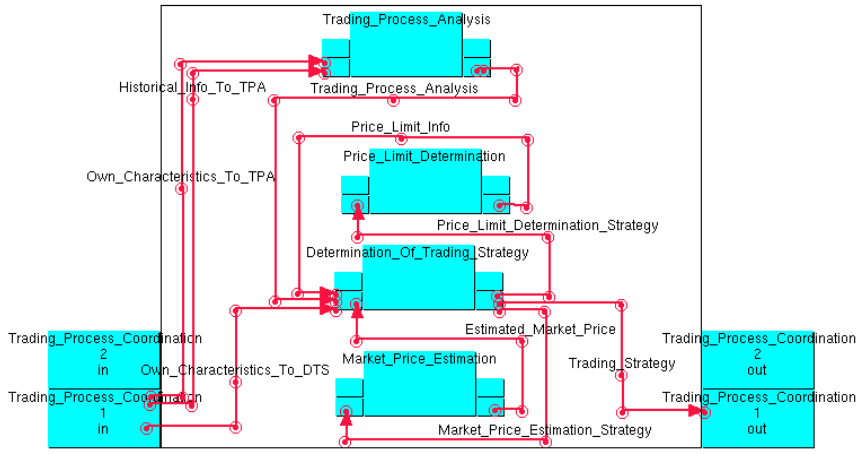

Figure 5.8. Subcomponent and information flow of Trading Process Coordination**.**

A 1.2.1 Trading Process Analysis (T 6.2)

The taskTrading Process Analysis is composed of a number of subtasks which are described in section 3.8.

A 1.2.2 Price Limit Determination (T 6.3)

Price Limit Deterination determines the maximum or minimum price of each product the agent has in the storage and on its wishlist, depending on whether the agent wishes to buy the produxt or sell it.

A 1.2.3 Market Price Estimation (T 6.4)

Market Price Estimation estimates the market price of each product the agent has in the storage and on its wishlist, depending on whether the agent wishes to buy the produxt or sell it.

A 1.2.4 Determination Of Trading Strategy (T 6.5)

Determination of Trading Strategy determines the trading strategy with the aim of the trading process analysis.

5.7.2 Process Control of Trading Process Coordination

The first process of Trading Process Coordination to act is Trading Process Analysis. The analysis made in Trading Process Analysis is used by Determination of Trading Strategy. Determination of Trading Strategy produces a strategy for Market Price Estimation. The information produced by Market Price Estimation is used by Determination of Trading Strategy to produce a strategy for Price Limit Determination. The information produced by Price Limit Determination is used by Determination of Trading Strategy to produce to Trading Strategy used by the other components of the Trading Agent.

5.7.3 Information Types and Links of Trading Process Coordination

- Trading Process Analysis: Trading Process Analysis contains the analysis of the trading process made in the component Trading Process Analysis. Trading Process Analysis consists of subinformation types, specified in section 3.8.3.
- Price Limit Determination Strategy: Price Limit Determination Strategy expresses knowledge about the strategy used to determine the price limit of products that are of interest.
- Market Price Estimation Strategy: Market Price Estimation Strategy expresses knowledge about the strategy used to estimate the market price of products that are of interest.
- Price Limit Info: Price Limit Info expresses knowledge about the price limit of products that are of interest.
- Estimated Market Price: Estimated Market Price expresses knowledge about the estimated market price of products that are of interest.

Table 5.6 summarizes which link transferres which information type.

Information Link	From		To	
	Process	**Information Type**	**Process**	**Information Type**
Historical Info to TPA	Trading Process Coordination	Historical Info	Trading Process Ananlysis	Historical Info
Own Characteristics to TPA	Trading Process Coordination	Own Characteristics	Trading Process Ananlysis	Own Characteristics
Own Characteristics to DTS	Trading Process Coordination	Own Characteristics	Determination of Trading Strategy	Own Characteristics
Trading Process Ananlysis	Trading Process Ananlysis	Trading Process Ananlysis	Determination of Trading Strategy	Trading Process Ananlysis
Market Price Estimation Strategy	Determination of Trading Strategy	Market Price Estimation Strategy	Market Price Estimation Strategy	Market Price Estimation
Price Limit Determination Strategy	Determination of Trading Strategy	Price Limit Determination Strategy	Price Limit Determination	Price Limit Determination Strategy
Trading Strategy	Determination of Trading Strategy	Trading Strategy	Trading Process Coordination	Trading Strategy
Estimated Market Price	Market Price Estimation Strategy	Estimated Market Price	Determination of Trading Strategy	Estimated Market Price
Price Limit Info	Price Limit Determination	Price Limit Info	Determination of Trading Strategy	Price Limit Info

Table 5.1. Information links in Trading Process Coordination.

5.8 Trading

The component Trading Process Coordination consists of a number of subcomponents. Section 5.4.1 describes the composition of the components Trading Process Coordination, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.8.1 Process Composition of Trading

This section describes the process composition of the component Trading. This component has the subcomponents Product Selection, Party Selection, Advertisement Determination, and Negotiation, as shown in figure 5.4. The subcomponents are described below in detail.

A 4.1. Trading

As seen in section 5.2, the tasks Product Selection, Party Selection, Advertisement Determination and Negotiation are mapped onto components of the same name and are brought together in the component Trading. Information type Trading Strategy is split up in Product Selection Strategy, Party Selection Strategy, Advertisement Determination Strategy, and Negotiation Strategy.

At this point an optimalization is made compared to the task model. In the task model the information type that is transferred to the components mentioned above is Trading Strategy. The tasks take the part of this information type they need and ignore the rest. In the task model Trading Process Coordination produces Trading Strategy and this infromation type is transferred to the tasks that use it. This way it is clear that it concerns a strategy for trading.

In the agent model, the determining of the trading strategy is done in Own Process Control. This information is tranferred to other components. Agent Interaction Management and History Maintenance use the part they need and ignore the rest, but in the component Trading in Agent Specific Task the information type is split up. Trading Strategy enters the

component, is split up and each subcomponent receives the subinformation type it needs. In the agent model, Product Selection, Party Selection, Advertisement Determination, and Negotiation don't receive Trading Strategy as in the task model, but a subinformation type thereof. Likewise they produce subinformation types of Trading Info. The component is shown in figure 5.8.

In: Historical Info, Trading Strategy.

Out: Trading Info.

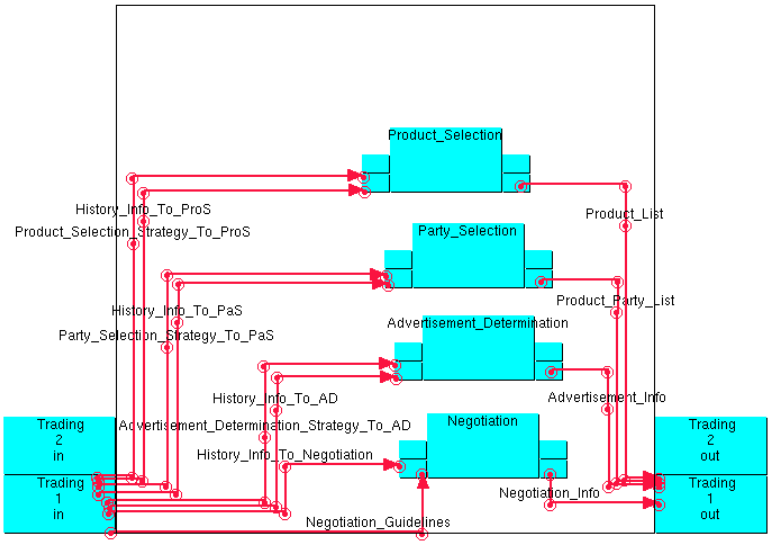

Figure 5.9. Subcomponents and information flow of Trading.

A 4.1.1 Product Selection (T 1)

Product Selection Strategy splits up to Determine Possible Products Strategy and Select Products Strategy. A local information type Possible Products move information from Determination Of Possible Products to Select Products. History Info will be used for determining what product is needed, are to be sold. The component Product Selection is mapped from the task model without any change. A description of this component is found in the task model in section 3.3.

In: Product Selection Strategy, Historical Info.

Out: Product List.

A 4 1.2 Party Selection (T 2)

Party Selection Strategy splits up in Determine Possible Parties Strategy and Select Party Setrategy. The latter information type splits up in Price Comparison Strategy and Past Negotiation Evaluation Strategy. It also contains a strategy on which of these components will be used, or weighs stronger in the selection process. The component Party Selection is mapped from the task model without any change. A description of this component is found in the task model in section 3.4.

In: Party Selection Strategy, Historical Info.

Out: Product-Party List.

A 4.1.3 Advertisement Determination (T 3)

In this component outgoing advertisements are composed using Historical Info to determine what products needs adertising, and Advertisement Determination Strategy to determine what the advertisement should look like and which agents should receive the advertisement. This results in an advertisement passed though in Advertisement Info.

In: Advertisement Determination Strategy, Historical Info.

Out: Advertisement Info.

A 4.1.4 Negotation (T 4)

This component handles one negotiation per sub-turn. The information type Negotation Guidelines is used in Negotiation Management to determine the negotiation strategy for one of the other components in Negotiation. Negotiation Management picks the right component based on the Negotation Guidelines and the Historical Info. The component picked to handle the current negotiation state uses the Historical Info and the Negotiation Strategy to handle it. The information types used in the component Negotiation are slightly different from the ones described in the task model in section 3.5. The subcomponents and information types are described in section 5.9.

In: Negotation Guidelines, Historical Info.

Out: Negotiation Info.

5.8.2 Process Control of Trading

Only one of the processes in Trading is done in on subturn. The Trading Strategy determines which process it and how the process is done.

5.8.3 Information Types and Links of Trading

- Trading Strategy: Trading Strategy contains of information about the strategy to be followed. Trading Strategy has subinformation types that contain strategies for the subcomponents of the Trading Agent. These subinformation types are Product Selection Strategy, Party Selection Strategy, Advertisement Determination Strategy, and Negotiation Strategy. These information types are also described in the task model in section 3.2.3. They are described in this section because of the optimatization described in section 5.8.1.
  - Product Selection Strategy: Product Selection Strategy contains information about the strategy to be followed concering product selection. This is a subinformation type of Trading Strategy.
  - Party Selection Strategy: Party Selection Strategy contains information about the strategy to be followed concering party selection. This is a subinformation type of Trading Strategy.
  - Advertisement Determination Strategy: Advertisement Determination Strategy contains information about the strategy to be followed concering advertisement determination. This is a subinformation type of Trading Strategy.
  - Negotiation Guidelines: Negotiation Guidelines constains information determine the Negotiation Startegy in Negotiation Management. This is a subinformation type of Trading Strategy. The Negotiation Strategy has sub-information types that are described in section 5.9.3.

- Trading Info: Trading Info constains information about messages to be send, decisions made at each level during trading (bids, products, parties, changes in trading strategy). Trading Info has subinformation types Product List, Product-Party List, Advertisement Info, and Negotiation Info.
  - Product List: Product List contains a list of products about which the agent may start a negotiation this turn, when a suitable trading partner is found.
  - Product-Party List: Product-Party List contains a list of products, each with a list of negotiation partners fit for trading the product. Negotiations may be started with all these agents.
  - Advertisement Info: Advertisement Info contains advertisements to be send.
  - Negotiation Info: Negotiation Info constains information about messages to be send, decisions made during negotiation (bids, changes in negotiation strategy).

Table 5.7 summarizes which link transfers which information type.

Information Link	From		To	
	Process	Information Type	Process	Information Type
History Info to Pros	Trading	History Info	Product Selection	History Info
History Info to PaS	Trading	History Info	Party Selection	History Info
History Info to AD	Trading	History Info	Advertisement Determination	History Info
History Info to Negotiation	Trading	History Info	Negotiation	History Info
Product Selection Strategy to Pros	Trading	Product Selection Strategy	Party Selection	Product Selection Strategy
Party Selection Strategy to PaS	Trading	Party Selection Strategy	Product Selection	Party Selection Strategy
Advertisement Determination Strategy to AD	Trading	Advertisement Determination Strategy	Advertisement Determination	Advertisement Determination Strategy
Negotiation Strategy to Negotiation	Trading	Negotiation Strategy	Negotiation	Negotiation Strategy
Product List	Party Selection	Product List	Trading	Product List
Product Party List	Product Selection	Product Party List	Trading	Product Party List
Advertisement Info	Advertisement Determination	Advertisement Info	Trading	Advertisement Info
Negotiation Info	Negotiation	Negotiation Info	Trading	Negotiation Info

Table 5.1. Information links in Trading.

5.9 Negotiation

The component Negotiation consists of a number of sub-components. Section 5.4.1 describes the composition of the components Negotiation, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.9.1 Process Composition of Negotiation

This section describes the process composition of the component Negotiation. This component has the subcomponents Negotiation Start, Bid Determination, Deal Closure, and Negotiation Management, as shown in figure 5.4. The subcomponents are described below in detail.

A 4.1.4 Negotiation (T 4)

This component handles one negotiation per sub-timestep. Negotation Strategy is refined in Negotiation Management to one of the sub-information types used for one of the other components in Negotiation. Negotiation Management picks the right component based on the Negotation Strategy and the Historical Info. The component picked to handle the current negotiation state uses the Historical Info and its Sub-Negotiation Strategy to handle it. The component is shown in figure 5.11.

In: Negotation Strategy, Historical Info.

Out: Negotiation Info.

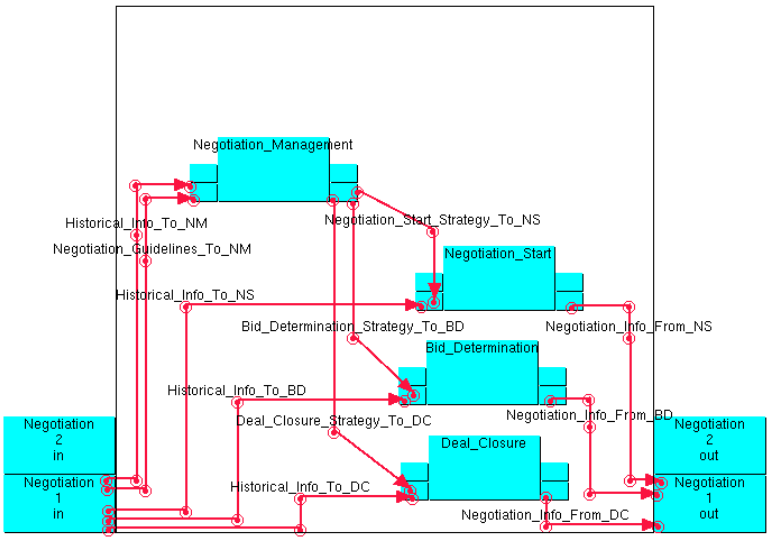

Figure 5.12. Subcomponents and information flow of Negotiation.

A 4.1.4.1 Negotiation Start (T 4.1)

Negotiation Start determines that and how a particular negotiation is started. Determining the opening bid for a product requires a completely different strategy than making a bid halfway a negotiation. An opening bid depends on normal prices, performance index, expectations of the other party, and the strategy that is followed. Determining a next bid depends on previous bids of the negotiation partner, own previous bids, price limits, and the strategy that is followed. The task is composed of the subtasks Determination Of Opening Proposal and Send Opening Proposal. These subtaks are not described in this chapter, but in section 3.5.1.

In: Negotiation Start Strategy, Historical Info.

Out: Negotiation Info.

A 4.1.4.2 Bid Determination (T 4.2)

Bid Detemination determines what the next bid will be. As pointed out at Negotiation Start, this depends on previous bid of the negotiation partner, the own previous bid, the price limits, and the strategy that is followed. This task is composed of the subtasks Determine Next Bid and Send Bid. These subtaks are not described in this chapter, but in section 3.5.1.

In: Bid Determination Strategy, Historical Info.

Out: Negotiation Info.

A 4.1.4.3 Deal Closure (T 4.3)

Deal Closure is responsible for the closing of the deal. All consequences of this deal are recorded here. A product may have been sold are bought, but is not yet in or out of the store. Somehow the agent has to know that this situation is about to change. This task is composed of the subtasks Send Deal Confirmations and Process Deal Condequences. These subtaks are not described in this chapter, but in section 3.5.1.

In: Deal Closure Strategy, Historical Info.

Out: Negotiation Info.

A 4.1.4.4 Negotiation Management (T 4.4)

Negotiation Management determines what sub task of negotiation is done and how it is done. Though the main strategic decision are made in Trading Process Coordination, decision regarding the course of the negotiation are made here. This does not have to be the case, even when using this agent model. It depends on what knowledge will be stored in which components. Since knowledge bases and process components are separated in this model, the choise of the exact decision of certain components does not have to made here. The Trading Agent that is going to be implemented for this thesis, however, will probably have most of the strategic knowledge at Trading Process Coordination. In the task model this task is described in section 3.7. The component is described in section 5.10 to show the small differences from the task model regarding the information types that are transferred this level of the agent model (i.e., Trading Strategy is split up and its sub-information type Negotiation Guidelines is transferred).

In: Negotation Guidelines, Historical Info.

Out: Negotiation Sub-Strategy (Negotiation Start Strategy, Bid Determination Strategy, Deal Closure Strategy).

5.9.2 Process Control of Negotiation

First Negotiation Management determines which of the other three components is activated and what strategy is used by this component. Depending on the phase of the negotiation, the negotiation is started, a bid is done, are a deal is closed. Ending the negotiation without closing a deal is done by Bid Determination.

5.9.3 Information Types and Links of Negotiation

- Negotiation Strategy: Negotiation Strategy constains information about the strategy to be followed concering the negotiation. This is a subinformation type of Trading Strategy. Negotiation Strategy has subinformation types. These are described in section 5.9.3.
  - Negotiation Start Strategy: Negotiation Start Strategy constains information about the strategy to be followed concering the start of a negotiation. This is a subinformation type of Negotiation Strategy.
  - Bid Determination Strategy: Bid Determination Strategy constains information about the strategy to be followed concering the bidding. This is a subinformation type of Negotiation Strategy.
  - Deal Closure Strategy: Deal Closure Strategy constains information about the strategy to be followed concering the closure of a deal. This is a subinformation type of Negotiation Strategy.

Table 5.8 summarizes which link transferres which information type.

Information Link	From		To	
	Process	Information Type	Process	Information Type
Historical Info to NM	Negotiation	Historical Info	Negotiation Management	Historical Info
Historical Info to NS	Negotiation	Historical Info	Negotiation Start	Historical Info
Historical Info to BD	Negotiation	Historical Info	Bid Determination	Historical Info
Historical Info to DC	Negotiation	Historical Info	Deal Closure	Historical Info
Negotiation Strategy to NM	Negotiation			
Negotiation Start Strategy	Negotiation Management	Negotiation Start Strategy	Negotiation Start	Negotiation Start Strategy
Bid Determination Strategy	Negotiation Management	Bid Determination Strategy	Bid Determination	Bid Determination Strategy
Deal Closure Strategy	Negotiation Management	Deal Closure Strategy	Deal Closure	Deal Closure Strategy
Negotiation Info from NS	Negotiation Start	Negotiation Info	Negotiation	Negotiation Info
Negotiation Info from BD	Bid Determination	Negotiation Info	Negotiation	Negotiation Info
Negotiation Info from DC	Deal Closure	Negotiation Info	Negotiation	Negotiation Info

Table 5.1. Information links in Negotiation.

5.10 Negotiation Management

The component Negotiation Management consists of the subsomponents Negotiation Analysis and Determination Of Own Negotiation Strategy. The component is slightly different from the task Negotiation Management described in the task model in section 3.7. The information type Trading Strategy is split up in its sub-information type Negotiation Guidelines. Section 5.4.1 describes the composition of the components Negotiation Management, section 5.4.2 describes the process control, and section 5.4.3 describes the information types and the links of this component.

5.10.1 Process Composition of Negotiation Management

This section describes the process composition of the component Negotiation Management. This component has the subcomponents Negotiation Analysis, and Determination of Own negotiation Strategy, as shown in figure 5.4. The subcomponents are described below in detail.

A 4.1.4.4 Negotiation Management (T 4.4)

The course of the negotiation is set here. Though the main strategic decision are made in Trading Process Coordination, decisions regarding the course of the negotiation are made here. The general picture has to be kept in mind when determining the strategy for a negotiation. This is done by transferring the Negotiation Guidelines from Trading Process Coordination to the Negotiation Analysis. The component is shown in figure 5.12.

In: Negotation Guidelines, Historical Info.

Out: Negotiation Strategy (Negotiation Start Strategy, Bid Determination Strategy, Deal Closure Strategy).

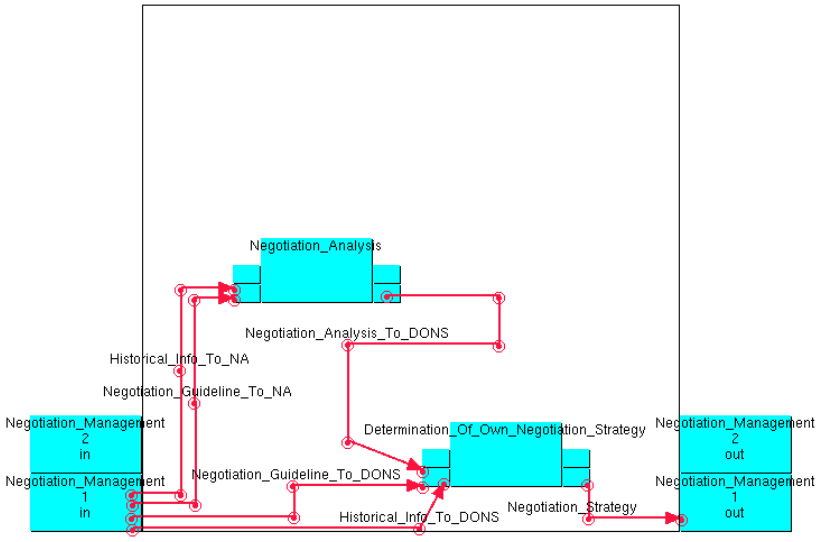

Figure 5.13. Subcomponents and information flow of Negotiation Management.

A 4.1.4.4.1 Negotiation Analysis (T 4.4.1)

This component analysis the negotiation with the aid of the Historical Info and the Negotiation Strategy. First the strategy of the other party is estimated in Estimation Of Strategy Other Party using Historical Info and Negotiation Strategy. With this information and the agents own strategy found in Historical Info, the end price of the negotiation is estimated in Estimation Of End Price. These two pieces of information and Negotiation Strategy the and the Historical Info help to make the choise if continuing with this negotiation is the best thing to do. This choise depends on whether:

- the end price is within limits
- the other parties strategy is acceptable
- the item is still needed or already sold or
- other reasons thought up in Trading Process Coordination for ending the negotiation.

The component is shown in figure 5.13. More detail about information flow and task control are found in section 3.7

In: Negotation Guidelines, Historical Info.

Out:. Negotiation Analysis.

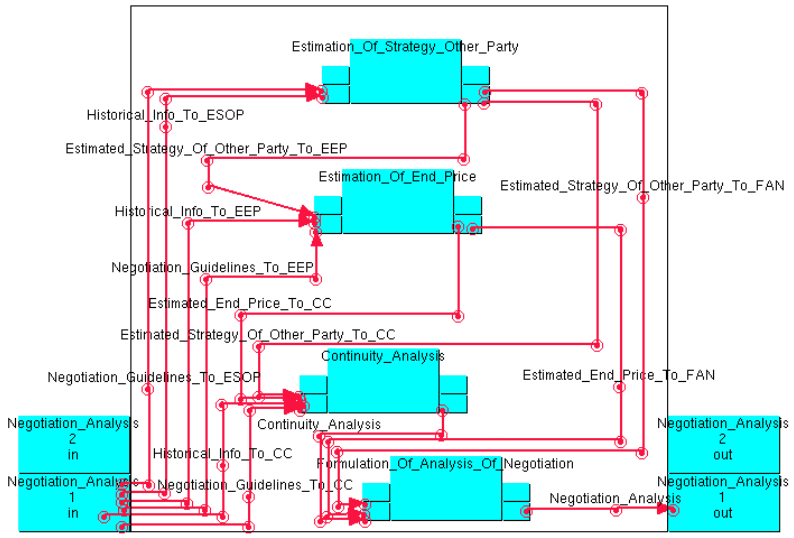

Figure 5.14. Subcomponents and information flow of Negotiation Analysis.

A 4.1.4.4.2 Determination Of Own Negotiation Strategy (T 4.4.2)

This component determines which component in Negotiation should be used, and the strategy to be followed by that component.

In: Negotiation Analysis, Negotation Guidelines, Historical Info.

Out: Negotiation Sub-Strategy.

5.10.2 Process Control of Negotiation Management

First the negotiation is analysed and then a negotiation strategy is determined.

5.10.3 Information Types and Links of Negotiation Management

- Negotiation Analysis: Negotiation Analysis expresses knowledge about the analysis made of the negotiation.

Table 5.9 summarizes which link transferres which information type.

Information Link	From		To	
	Process	Information Type	Process	Information Type
Historical Info to NA	Negotiation Management	Historical Info	Negetiation Analysis	Historical Info
Historical Info to DONS	Negotiation Management	Historical Info	Determination Of Own Negotiation Strategy	Historical Info
Negotiation Strategy to NA	Negotiation Management	Negotiation Strategy	Negetiation Analysis	Negotiation Strategy
Negotiation Strategy to DONS	Negotiation Management	Negotiation Strategy	Determination Of Own Negotiation Strategy	Negotiation Strategy
Negotiation Analysis to DONS	Negetiation Analysis	Negotiation Analysis	Determination Of Own Negotiation Strategy	Negotiation Analysis
Negotiation Sub Strategy	Determination Of Own Negotiation Strategy	Negotiation Start Strategy OR Bidding Strategy OR Deal Closure Strategy	Negotiation Management	Negotiation Start Strategy OR Bidding Strategy OR Deal Closure Strategy

Table 5.1. Information Links in Negotiation Management.

6 Detailed Design

In this section the detailed design of the agent is described. Section 6.1 describes the approach taken, section 6.2 describes the architecture of the agent. Section 6.3 describes the structure of the Java classes. Section 6.4 describes the order of control of the different components. Section 6.5 describes how strategies are changed, and section 6.6 evaluates the detailed Design.

6.1 Approach

This section describes the approach taken for the implementation of the trading agent. The marketplace and everything on it (Clock, Storage, Bank, Market Manager, and Messages) is already present. For this thesis a prototype of a Trading Agent that can function on this market has to be built. The agent has to handle the incoming messages and send understandable messages in such a way that it can buy and sell goods in order to achieve its goals.

A number of possible implementation possibilities exist. First, the links could be explicitly implemented as well as the input and output interface buffers of all components. This approach seemed to be inefficient. Deep copies need to be made at each information transfer point: from a component output interface to a link, from the link to a component input interface, etc.

The second approach, the use of blackboards [Erman, Hayes-Roth, Lesser & Reddy, 1980; Reddy, 1976] is chosen instead. A blackboard is a central global database for the communication of independent asynchronous knowledge sources focussing on related aspects of a particular problem. This thesis uses an adapted version of the blackboard. The processes using the knowledge on the blackboard depend on the order in which the processes are executed. Therefore the processes are executed sequentially by a process control. The structure used in this thesis is described in more detail in section 6.2. This approach was used before during the college Multi-agent systems in Complex Domains for programming an agent and seemed promising.

Since the marketplace is programmed in Java (URL: http://java.sun.org), this language was also chosen for implementation of the Trading Agent.

The requirements for the detail design are:

- The detailed design should have good documentation about how to plug in strategies.
- A clear transition between the agent model and the detailed design should be present. Knowledge types and flow, and components from the agent model should be recognizable.

6.2 Agent Architecture

This section specifies the blackboard structure of the Trading Agent. A super class of the agent is available for the basic functions needed in the marketplace, as collecting and sending mail. Classes are made of all the information types and components described in section 5. All components have their own blackboard and classes of them are also made. Super classes are made for components, information types, and blackboards.

Section 6.2.1 describes the basic structure of a blackboard, section 6.2.2 describes the information flow, section 6.2.3 describes how CPU time can be saved by using pointers, and section 6.2.4 describes an example of how the architecture works.

6.2.1 Structure of a Blackboard

Basically a blackboard consists of a number of information types, and an information type can consist of a number of sub-information types. A sub-component has limited access to the information present on the blackboard of its super-component. Figure 6.1 shows how a blackboard is situated in a component. A small readable part (R) for one of its sub-components to read is shown. This readable part is different for each sub-component, as each component needs different information.

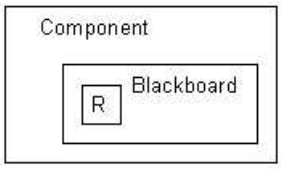

Figure 6.1. Component with blackboard. R is the readable part.

Figure 6.2 shows the DESIRE view of a super-component with 2 sub-components, and the information flow between them. All visible information types A, B, C, and D are present on the blackboard of the super-component. Information type A is readable for sub-component X. Information types B and D are the output of component X and information type B is input for component Y. Information type C is output of component Y. Component X takes information type A and puts this on its blackboard. It reasons with this information and delivers output: information types B and D. Component Y does the same, but with information types B as input and C as output. In the course of section 6 this view is adapted, to support the optimalization described in section 6.2.3.

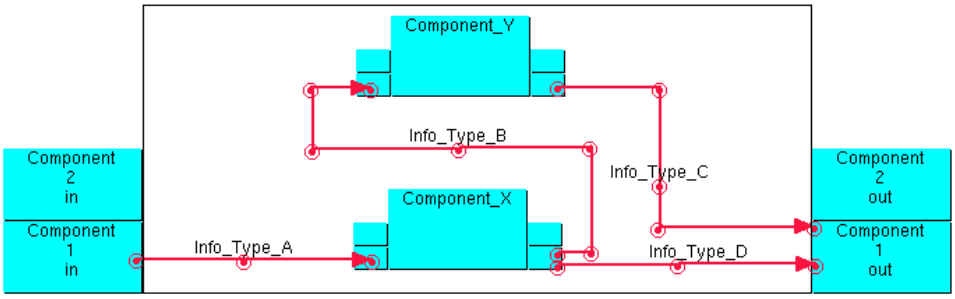

Figure 6.2. DESIRE view of information flow within and between components.

6.2.2 Information Flow

The information flow in the agent is arranged in the following manner: suppose component X has a blackboard X. Component X reads information that is important for it or its sub-components from the blackboard of its super component, and puts that information on to its own blackboard X (figure 6.3). Component X can only read information from the blackboard of the SuperComponent that is meant for component X.

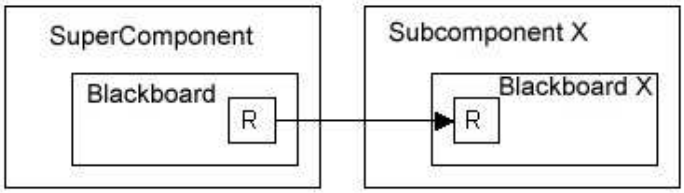

Figure 6.3. Component X can only read a specific part (R) of blackboard of the SuperComponent and put it onto blackboard X.

Now component X and X's sub-components can read info from blackboard X, process this info and write their output back on blackboard X. When there are no sub-components, component X reasons with the information it read from the blackboard of SuperComponent (figure 6.4). After this is done, component X can write the output back to the blackboard of SuperComponent (figure 6.5).

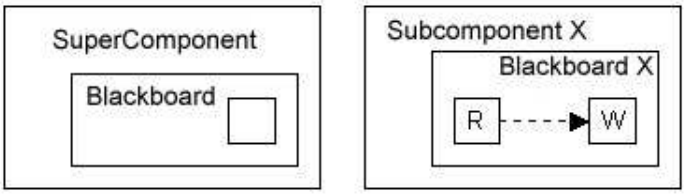

Figure 6.4. Component X reasons with the information received from the blackboard of SuperComponent (R) and creates new output (W).

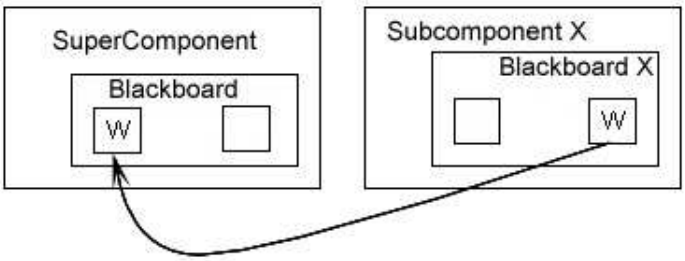

Figure 6.5. Component X puts the created output (W) back on the blackboard of SuperComponent.

With this approach a lot of CPU time is lost in copying information from one blackboard to another and back. Section 6.2.2 describes how this can avoided, while still maintaining a well-structured agent. Section 6.2.3 explains the structure with an example.

6.2.3 Info State

Instead of copying a part of a blackboard to another blackboard, a pointer to the information to be used can be copied to another blackboard. The component will work directly with the information present in its super component, but it will look like it is working locally with only the information it can see. The consequence of this approach is that more components could be working on the same information at once. This should be avoided, so components should work sequentially.

All information types together are called the *Info State* of an agent. All components receive a part of the Info State to work with, and put their output information back on it. History Maintenance has to make deep copies, because it has to keep immutable records.

This approach has as consequence that a component needs a copy of the pointer to its output information type as input. This output is present in the Info State, before it is filled by the

component. The component receives the copy of the pointer and fills the information state with its output. There is no need to return the pointer, because it is a copy. This way, no output has to be actively transferred, and for the input, only pointer copies have to be transferred. An example is presented in the next section.

This has the following consequences for the way the DESIRE view is mapped to the detailed design. Figure 6.6 is the same as figure 6.2. Using the view described in this section, the large component now has pointers to the information types A, C, and D on its blackboard. These information types are physically present on the blackboard of the agent. Sub-components of the agents use pointers that point to these information types.

Information type B is a local information type that is 'physically' present on this blackboard. This local information type is not part of the Info State of the agent. Only sub-components can use this info, which is exactly what is wanted.

Component X has pointers to information type A, *and* B and D, which are empty, on its blackboard. Component X reasons with information type A, and its output is written directly to information types B and D. Component X writes its output directly in the information types, which are physically present at other locations than the blackboard of component X, that only has pointers. As the blackboard of component X only has a pointer to information type B and D, no output is transferred to other components. Components that need the output information of component X already have a pointer to it.

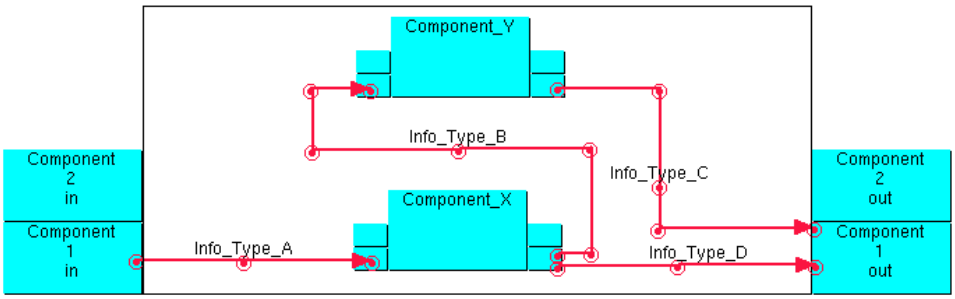

Figure 6.6. DESIRE view of information flow within and between components.

Components share information in the Info State of the agent. A result of this is that component have to work in sequence. When component would work parallel, components may use information that is rewritten at the moment they use it. This may cause problems.

6.2.4 Example

In this section the blackboard structure is illustrated with an example.

At the top level of the agent the information types Communicated Info, Info to be Communicated, Communicated Agent and World Info, Historical Info, Trading Info, and Trading Strategy are present. These information types form the Info State of the agent. Own Process Control has a local blackboard (figure 6.4).

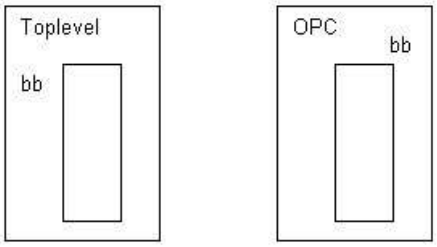

Figure 6.7. The top level of the agent and Own Process Control, each with a blackboard.

Own Process Control receives a copy of a pointer to part of the blackboard at top level, Historical Info and Trading Strategy (Figure 6.5).

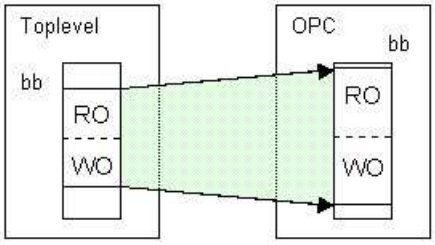

Figure 6.8. Own Process Control receives a pointer to the information it needs.

The latter of these two is to be filled by Own Process Control. This pointer becomes a part of the blackboard of Own Process Control. The local information type, Own Characteristics, is already present on the blackboard.

Trading Process Coordination now receives pointers to the information types Historical Info and Own Characteristics and Trading Strategy. This component reads the incoming information and fills the outgoing information, the Trading Strategy. In figure 6.5 the arrow shows that Own Process Control somehow writes this information (because of information and process hiding other component do not know how Own Process Control does this).

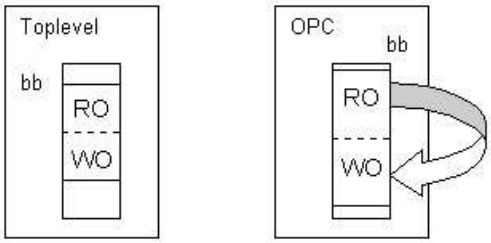

Figure 6.9. Own Process Control somehow writes information on the blackboard.

Though it looks like Own Process Control writes this information locally, it really writes directly onto the Info State of the Trading Agent as shown in figure 6.6.

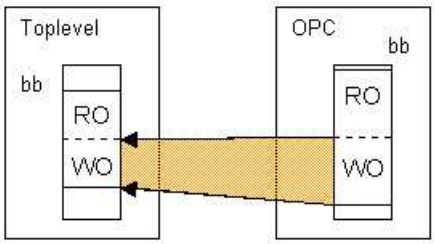

Figure 6.10. Own Process Control writes information onto the Info State of the Trading Agent.

Information used only locally in a component is not part of the Info State of the Trading Agent. This information is only present on the blackboard where it is used.

6.3 Structure of the classes

This section describes the structure of the Java classes that are used to implement the agent. Classes are made of all components, information types, and blackboards. Super classes are also made of these three types.

A component has one blackboard. A blackboard has a number of copies of pointers to information types that are used by the component. Local information types of a component are directly pointed to the information type; no copy of a pointer has to be made. A component can only use the information types present in its blackboard.

The Trading Agent is an extended class of the agent class, which was made for the college Multi-agent systems in Complex Domains. The agent class has methods for acting in the Agent Mediated Marketplace.

The Trading Agent creates its blackboard. This blackboard creates pointers to the information types present in the Trading Agent. Each information type creates pointers to its sub-information types.

The Trading Agent also class creates its components, and each component creates its sub-components. Each component creates its blackboard, and each blackboard creates it local information types, when it has these. Each component is created with a copy of the pointers to the information types used by the component. The blackboard created by each component is created with these copies.

6.4 Control

This section describes the control flow in the implementation of the Trading Agent.

The control flow shown in section 5 is also used in the implementation, with a few slight modifications and additions.

First all components and sub-components and all of their blackboards and the information types have to be created. Then the agent has to locate the manager and register at the market. Then the ‘market loop’ starts. One go through the ‘market loop’ is equal to one turn. The loop is exited when the market ends or when the agent stops participating in the market.

Within the ‘market loop’ is another loop, the ‘turn loop’, that is exited only when the agent wants to end the turn. Before the ‘turn loop’ starts, the agent has to get its mailbox. In the turn loop the following happens: first the history is maintained, then TPC determines what has to be done and how it has to be done. When, for example, in the previous pass through the loop AST decided that a message had to be send, this pass AIM has to created and send that message with the aid of the information present on the black-board. For an overview of the control flow, see figure 6.11.

```
Start-up
create all components and their black-boards
locate and register at manager
while (! end of market)
receive mailbox
        while (! end of turn)
                MH
                OPC
                continue turn check
                MH
                case
                        AST
                        AIM
                end case
        end while
        MH
end while
```

Figure 6.11. Control Flow in the Trading Agent in pseudo code.

6.5 Strategy Plug in

This section describes how strategies can be changed in the implementation of the Trading Agent.

Each component class has a standard method called StartReasoning. This method reasons with the strategy that is given in the Trading Strategy. The method consists of a number of standard decisions that have to be taken. The information in the trading strategy determines which choice is made.

When a strategy of a component has to be changed, the following actions have to be undertaken:

The strategy used for this component has to be adapted. The strategy is determined and constructed in the component Determine Trading Strategy. This is the place where the strategy has to be adapted.

When a strategy is wishes in which decision are made that are not described in the StartReasoning method of the component, then this method has to be adapted. This is done by creating a new method that deals with this decision. When a trading strategy has no information for the decision, a standard decision has to be made.

6.6 Evaluation

This section discusses the requirement described in section 6.1 and how they are satisfied:

- *The detailed design should have good documentation about how to plug in strategies.* Section 6.5 describes the changes that have to be made when a strategy is changed.
- *A clear transition between the agent model and the detailed design should be present. Knowledge types and flow, and components from the agent model should be recognizable.* Section 6.2 describes the use of blackboards and how components use these blackboards for the information flow.

7 Conclusions

This section discusses the conclusions of this thesis. This thesis describes a task model of the trading task, the design and implementation of a prototype of a trading agent. The formulation of trading strategies play an important role in the aspects described.

Section 7.1 discusses the Agent Mediated marketplace, section 7.2 discusses conclusion about the task model, section 7.3 discusses conclusions about the strategies, section 7.4 discusses conclusions about the agent model, and section 7.5 discusses conclusion about the detailed design. Section 7.6 discusses possible alternatives in the design and concluding remarks.

7.1 Agent Mediated Marketplace Discussed

The Agent Mediated Marketplace (AMMP) as described in Appendix A forms the platform for the Trading Agent described in this thesis. This platform is originally constructed for the course Multi-agent systems in Complex Domains. Because it is made with great speed, not every problem encountered was solved in a neat way. It is constructed to do its work, which is making it possible for agents to trade, and this works.

However, the message flow is deeply integrated in the market structure, which is not how it should be. Messages need to be sent between agents, and should be monitored by the manager. Observing the stock or bank account should be done by sending a message to the manager of the stock or the bank, and a message should be received with the result.

These changes, and more, would make the AMMP much better, but who has the time when it already works?

The market environment may be changed in the following ways:

- The turn-based market may be changed to a market where the time is continuous.
- The protocol described in appendix A may be changed in order to make cheating more controlled. Adding more message types are altering message types increases the possibilities in trading:
    - In the existing market products are selected based on their performance index and a negotiation is started about the price of this product. A better alternative is to select a product category and negotiate about the attributes concerning the product, delivery, and service.
    - The language or ontologies may be changed.
    - Negotiations may become multi-lateral.

Appendix B proposes a number of changes in the message types used for trading products. Using these message types, deal making is more secure.

The sections concerning the task model, the agent model, and the detailed design describe how the changes described in this section influence them.

7.2 Task model Discussed

The task model described in section 3 is based on the General Commerce Framework [Clurman, Foley, Guttman & Kupres, 1997] and the six stages identified in Consumer Buying Behavior [Guttman & Maes, 1998; Guttman, Moukas & Maes, 1998].

The task model of the trading task satisfies the following requirements:

- The task model is a realistic view on trading.
- The task model supports trading done by both consumer and merchant.
- The task model supports both buying and selling.
- The task model is able to handle multiple trading parties.
- The task model is able to handle multiple products to trade.
- The task model supports the search for trading partners that want to trade a specified product. A subtask dealing with advertisement sending has to be present for this purpose.
- The trading task is constructed in such a way that strategies for all aspects of the trading task can be explicitly modeled.
- The task model has a sub-process to evaluate strategies in order to learn whether a strategy functions good or bad.

The task model is affected by the changes mentioned in section 7.1 in the following way:

- The task model can be used in a market where time is continuous. The control flow may have to be adapted, to cope with the irregular arriving of messages. New information has to be processed in time to react in an adequate manner. The implementation of the task model may be slow, because a lot of information is processed.
- The message types are converted to information types. The information types are used by all tasks, the message types are only used in History Maintenance, where they are converted. A small adaptation in the conversion makes it possible to deal with other message types
    - The task model is open for multi-attribute negotiation. The bids determined at different places in the task model may be bids concerning multiple attributes. In trading process Control, however, the main focus is put on price negotiation, but tasks may be added to deal with other attributes. Details on multi-attribute trading or negotiation are not discussed in this thesis.
    - Changes in the ontology can not be handled by the task model described in section 3. Changes have to be made in the Trading Process Analysis, where evaluations are made of in detail named parts of the ontology used.
    - The task model can handle multi-lateral negotiations. The reason why multi-lateral negotiation is not possible at the moment is the limitation of the protocol.

The task model is extensible to support more strategies, and it does not depend on specific language or protocol. One possible way to extend to task model is by adding tasks to deal with cooperation.

The task model was not created over night. Numerous alternatives were rejected. Advertisement Determination was originally a part of Party Selection. This was done because sending advertisements is a consequence of failing to find a negotiation party. Later was

decided that determining advertisements is a separate task, because there may be other reasons the send advertisements, such as gaining information about other agents, or giving information to other agents.

7.3 Strategies Discussed

Trading requires more strategies than negotiation strategies alone. Sample strategies are described in section 4 and discussed is how they fit in the task model. Strategies have to be adapted when the market is not turn based. More emphasis has to be put on reacting quickly to changes.

7.4 Trading Agent Model Discussed

The task model is mapped to an agent model. It forms the basis for the detailed design.

The agent model is affected by the changes mentioned in section 7.1 in the following way:

- The agent model can function in an environment that is not turn based. The agent model is based on turns and sub-turns, but these turns do not have to occur in a market. Own Process Control determines what tasks are done each turn in the agent model. The agent and world knowledge has to be kept up to date. This is done by emptying the mailbox every few (internal) turns. This way it is not always the case that the agent reasons with the most recent knowledge, but this should not be a problem. Another way to deal with this problem is to update the agent and world knowledge as soon as a message is send. Negotiation is usually a process in which to parties each send a proposal in turn. At deal making time, the agent should be cautious to have the overall picture in mind.
- Changes in message types only affects the Agent Interaction Component. This component converts the messages in the information types used by the other components. Other components use only the information types.
    - The agent model is open for multi-attribute negotiation. The bids determined at different places in the task model may be bids concerning multiple attributes. In trading process Control, however, the main focus is put on price negotiation, but tasks may be added to deal with other attributes.
    - Changes in the ontology can not be handled by the agent model described in section 3. Changes have to be made in the Trading Process Analysis, where evaluations are made of in detail named parts of the ontology used.
    - Multi-lateral negotiations can be handled by the agent model.

The agent model is independent of the agent platform.

The agent model is extensible, e.g., with the Cooperation management component to model cooperative behavior of agents.

An optimalization took place in the agent model compared to trading task model: sub-strategies transferred to subtasks instead of the complete trading strategy.

7.5 Trading Agent Architecture

The detailed design makes use of blackboards instead of using the standard Input / Output Interface Buffers used in DESIRE.

7.6 Prototype

The prototype is not finished, but it will be operational medio september. No advanced strategies will be implemented, but places will be left open to plug in strategies.

7.7 Concluding Remarks

The thesis has answered the question:

> *How to model an agent in such a way that trading strategies can be tested?*

The agent model of the Trading Agent is described in section 5. Strategies can be plugged in.

7.8 Future Research

The following research is interesting and may be done by other researchers:

- Development evaluation criteria for the AMMP and trading agents.
- Implementation of Strategies in the Trading Agent.
- Introduction of Cooperation and Cooperation management to form cartels.
- Instead of using a performance index, negotiations could become multi-attribute.
- The agent may trade on a market that is not turn based.
- The model is expandable in a number of ways, as described in section 8.
- The trading agent described in this thesis may be used for the successor of the course Multi-agent systems in Complex Domains: Intelligent Interactive Distributed Systems.
- The trading agent may be used for the design and modification of trading agents e.g., in a Multi Agent Factory [Brazier & Wijngaards, 2001]. Such a factory builds agents on request for a user.

7.9 Personal Evaluation

This thesis made me aware that bright ideas don't get you anywhere unless you can formulate them in such a way that other people know what you mean.

For this I like to thank Niek, who almost never understood my great ideas, which he did on purpose to learn me the important skill of 'one step at a time'.

I would also like to thank Loes for her support in good times and in bad.

Modeling is fun, but implementing your own model is great. Too bad that I didn't get to do this, yet.

I would like to conclude with my moto:

> *Everything is gonna be just fine.*

8 Recommendations & Further Research

This section discusses recommendations in section 8.1 and some future research in section 8.2. The style of writing in this section is less formal than in the rest of the thesis. In this section all ideas available are blurted on the paper, without much structure. Real recommendations about the protocol are found in Appendix B.

8.1 Recommendations

This section describes recommendations for the protocol as described in appendix A, and for the structure of the implementation of the Agent Mediated Marketplace. Section 8.1.1 discusses the necessity of the attribute Product Name of a product, section 8.1.2 discusses the fact that it is possibility to start a negotiation without first advertising, and section 8.1.3 describes the difficulty of checking for cheaters in the deal making, and how a change in the protocol can change this. This is further described in appendix B.

8.1.1 Necessity of PN

This section discusses whether the attribute Product Name of a product is really necessary. First the list of attributes from the protocol described in appendix A, used to identify an object is given.

The following attributes are needed, for identifying a product:

- Product Category (PC): what kind of product is it?
- Performance Index (PI): how good is the product?
- Price (P): how expensive is it?
- Product Name (PN): what is the name of this particular product?

The PI is an index that stands for the combined attributes of a product. A cordless mouse may have the same PI as a mouse with a scroll button, yet both have different attributes.

The PN is used to identify one product, i.e. “The mouse I hold in my hand is called ‘mouse14’. The mouse in my other hand is of the same PC and has the same PI, yet it is called ‘mouse15’.”

In the protocol described in appendix A, the PN was required while negotiating for a price or requesting whether an object was for sale or not. The problem with this is that sellers have to name their products while advertising, or else buyers cannot start a negotiation. Sellers on the other hand can start a negotiation with everyone, without the need to advertise first.

The biggest problem with these weak rules is that there is a big possibility that in the first round a huge number of messages is sent, more than the manager could handle. This is solved by letting people pay for their messages and restricting the amount of messages that may be send by each agent in each turn.

It would be nice however that a neat order in which message are send during trading is enforced: first advertising, followed by negotiating and finished with deal making. I will expand on this in *Neat order of messages*.

This section not only questions whether the attribute Product Name is necessary, but whether it hinders trading.

Suppose a seller has two products with the same PC and PI. The seller starts negotiations about these products with three buyers. The seller is negotiation with one buyer about one of these two products, and with the two other buyers about the other product. After a while it sells the product to one of the two buyers. Does this imply that the seller has to start over again with the negotiation it had with the remaining buyer of these two, or can he simply change the product name to the product it negotiates about with the first buyer? Isn't it possible to have a negotiation without making use of a product name? Can't we just delete the attribute Product Name?

The attribute Product Name cannot be deleted, because a product instance needs an identity in order to make it impossible for agents to create a product. When negotiating, the product name causes trouble as shown above. When exchanging a product between two parties, the manager could just take one of the two. It really doesn't matter which one it takes, because they are the same. So during negotiation and exchanging agents need not know the product name.

One drawback is that now not only the seller can initiate a negotiation without first advertising but the buyer can do this as well.

8.1.2 Neat order of messages

It would be nice that a neat order in which message are send during trading is enforced: first advertising, followed by negotiating and finished with deal making.

A possibility to enforce a neat order of messages is to make a rule that states that only buyers can initiate a negotiation could be added. Sellers now know the PN, but cannot sell anything without making the PN known to the buyers. Now advertising is almost compulsory.

Another possibility to enforce a neat order of messages is to enforce that all messages, except adverts, have to be a response to another message. This implies that one cannot start a negotiation without first having heard about an object through advertisements. This also implies that seller may start a negotiation after having heard that a buyer is looking for a specific object. The problem is that one can reply to a message and talk about something completely different. "I want to buy a mouse". "In reply to your message about the mouse: do you want to buy this monitor for $500."

This is not an option, except if you make the protocol so tight, that a reply-to message has the same content as the original, which is also checked by the manager. "want-to-buy: PC-X PI-Y". "reply-want-to-buy: PC-X PI-Y, want-to-sell, P-A" "reply-want-to-sell: PC-X PI-Y, want-to-buy: P-B" etc. This is hard to enforce. It also makes it impossible to handle package deals.

A slight modification of this might work: the first proposal has to be this tight reply message: "i-want-to-sell PC-X PI-Y" "reply-i-want-to-sell PC-X PC-Y, buy first-proposal P". Or even: "i-want-to-buy PC-X PI-Y" "reply-i-want-to-buy PC-X PC-Y, sell first-proposal P". (the seller may also do the first proposal.) After this first proposal, both agents can say and do anything they like.

Enforcing strict rules in this stage is not necessary, because it is not wise to mess around while negotiating. First of all message sending costs money, so sending messages without knowing whether the party is even remotely interested in what you have to say is expensive. Further more the other agents will not trust you anymore.

In fact, it's wise not to mess around at all, because chances are, that other agents don't take you seriously any longer. So I propose to make a neat formal communication that agents have to follow. No enforcing is necessary, because agents will punish each other by not negotiating with trouble makers.

8.1.3 Cheating in deal making

The protocol described in appendix A leaves room for agents to cheat. After the exchange of transaction messages between the agents, both agents have to send a message to the manager (page 10 of Appendix A). The message to the manager may be entirely different than the one send to the other agent.

The protocol described in appendix B closes this hole. When sending a transaction proposal, a copy of the transactions proposal is send to the manager. Since the manager acts between the turns when the agents are put on hold, it can check the transactions messages and send a confirmation that he has received the same transaction proposal as the agent who receives it does. The receiving agent will thus receive 2 transaction proposals in the turn after the sending agent sent it: one from the agent who sent it, and one confirmation from the manager.

The same will happen with the confirmation send by the receiving agent. The complete protocol is described in appendix B.

8.2 Future Research

One possible expansion of the market is breaking up the performance index, to make the trading multi-attribute. Not only the physical attributes of a product, but also other properties like delivery time can be simulated.

Shipping costs could be simulated in the protocol. When negotiation a price with more than one negotiation partner, the one with the highest shipping costs has to try to make a more appealing deal then the others. Combining products at one source may lower the combined shipping costs.

A great challenge is to design a marketplace where all participants are situated at a specific location. Consumers can wander around, while merchants have to stay close to their store. Negotiations can only be conducted while standing close to the merchant you are negotiation with. The only means of advertisement is yelling, or mouth to mouth advertisements of buyers. While wandering around on the marketplace, prices may change at the places you have been, without you knowing it. This means that world knowledge becomes outdated very quickly. This environment could result in very nice strategies.

A smaller step from the Agent Mediated Marketplace is negotiating about package deals. Merchants are able to "throw in a mouse for free".

References

Benyoucef, M. and Keller, R. K. (2000). A conceptual architecture for a combined negotiation support system. *Tech. Rep. GELO-118*, Montreal, Canada, February 2000.

Brazier, F. M. T., Dunin-Keplicz B. M., Jennings, N.R. and Treur, J. (1995,1997). Formal specification of Multi-Agent Systems: a real-world case, in: V. Lesser (ed), *Proceedings of the First International Conference on Multi-Agent Systems, ICMAS'95*. Cambridge MA: MIT Press, pp. 25-32. Extended version: 1997, *in*: M. Huhns and M. Singh (eds), *International Journal of Co-operative Information Systems*, special issue on Formal Methods in Co-operative Information Systems: Multi-Agent Systems, **6**, 67-94.

Brazier, F. M. T. and Wijngaards, N. J. E.: (2001). Automated servicing of agents. *D. Kudenko & E. Alonso (eds), Proceedings of the AISB-01 Symposium on Adaptive Agents and Multi-agent systems, at the Agents & Cognition AISB-01 conference, the society for the study of artificial intelligence and the simulation of behaviour*, ISBN 1.902956.17.0, pp. 54 - 64.

Brazier, F. M. T., Jonker, C. M. and Treur J. (1996). Modelling project coordination in a multi-agent framework, in *Proceedings Fifth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WET ICE'96*, Los Alamitos: IEEE Computer Society Press, pp. 148-155.

Brazier, F. M. T., Jonker, C. M. and Treur J. (1998). Principles of Compositional Multi-agent System Development, *in* J. Cuena (ed), *Proceedings of the 15th IFIP World Computer Congress, WCC'98, Conference on Information Technology and Knowledge Systems, IT&KNOWS'98*, pp. 347-360.

Brazier, F.M.T., Jonker, C.M., and Treur, J. (2000). Compositional Design and Reuse of a Generic Agent Model . *Applied Artificial Intelligence Journal.*, 2000, Volume 14, number 5, p 491-538.

Chavez, A., Maes, P. (1996). Kasbah: An Agent Marketplace for Buying and Selling Goods. Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, London.

Clurman, W., Foley, T., Guttman, R., Kupres, K. (1997). Electronic commerce with software Agents, *Electronic Commerce and Marketing on the Internet, Professors Tom Malone & John Little*

Erman, L.D., Hayes-Roth, F., Lesser, V., and Reddy, D. (1980). The HEARSAY II speech understanding system: Integrating knowledge to resolve uncertainty. *Computin Surveys* 12 (2): 213-253.

Guttman, R.H. and Maes, P. (1998). Agent-mediated Integrative Negotiation for Retail Electronic Commerce, *paper presented at the Workshop on Agent Mediated Electronic Trading, (Minneapolis/St Paul, May 10), a workshop of the 2nd International Conference on Autonomous Agents (Agents'98); Lecture Notes in Artificial Intelligence* 1571, pag. 70-90, Springer-Verlag. URL: http://www.iiia.csic.es/amet98/paper5.pdf

Guttman, R.H., Moukas A.G., and Maes, P. (1998). Agent-mediated Electronic Commerce: A Survey., *Knowledge Engineering Review*, 13, 147-161.

Matos, N., Sierra, C. and Jennings, N. R. (1998). Determining successful negotiation strategies: an evolutionary approach. *Proc. 3rd Int. Conf. on Multi-Agent Systems (ICMAS-98)*, Paris, France 182-189.

McAfee, R. P., McMillan, J. (1987). Auctions and Bidding, *Journal of Economic Literature*, June 1987, pp. 699 – 738.

Wooldridge, Micheal, and Jennings, Nicolas R. (1995). Agent Theories, Architectures, and Languages: a Survey. *Woolridge and Jennings Eds., Intelligent Agents*, Berlin: Springer-Verlag, 1-22.

Agent Mediated Market Place

Communication Protocol

Version 0.4; April 18th, 2001

This document describes the communication protocol as used by the agents in the Agent Mediated Market Place (AMMP), developed for the course Multi-Agent Systems in Complex Domains. The message layout is described, the protocol (i.e., sensible sequences of messages) and definitions of the contents of types of messages.

Contents

1 Basics

First the generic message layout is described, then some basic terminology is described, after which assumptions on the marketplace are described. The messages described in this document are presented in the ACL format even though an agent receives a Java object from a specific class. The `toString()` method may be used to obtain an ACL format of a message.

1.1 Generic Message Layout

The messages used in AMMP use the ACL format. In the protocol (and message types) it is described which communicative acts are used. The message parameters described in Table 1 are *always* used within the AMMP. At the moment the interactions with the manager, bank, store and clock are modelled as world interactions (except for performing a transaction).

Message Parameter	**Definition**	**Usage in AMMP**
:sender	identity of sending agent	Is automatically added to a message when an agent sends a message. The following special cases exist: "Manager", i.e., the AMMP-manager "Bank", i.e., the AMMP-bank "Store", i.e., the AMMP-store "Clock", i.e., the AMMP-clock.
:receiver	identity of recipient agents	Is specified by the sending agent. • The identity has to be a valid ID of an agent in the marketplace otherwise the message is not posted. • A single recipient may be used, or a list of recipients. • The following special case exists: "ALL" denotes *all* participants in the marketplace
:content	content of the message (or the subject of the action)	Here products, prices, bids, deals, etc. are described.
:reply-by	Elapsed time until content of message 'expires'	The time until the message becomes obsolete. Note that the parameter is given a slightly different use: an integer. A number of elapsed turns is expected as the value (so this is not an official date format). Note: this is not a delta, this is not an increment. This is the number of elapsed turns since the beginning of the marketplace, as registered by the Clock.
:language	encoding scheme of content	Is automatically set by the message transport methods of an agent.
:ontology	ontology to be used for the content	This is automatically set by the message transport methods of an agent to be the "micd-ammp-computer-products-v1.0" ontology (see another document).
:protocol	the protocol with which messages are sent	This is automatically set by the message transport methods of an agent to be the "micd-ammp-v1.0" protocol.

Table 1. Message parameters that are always used within AMMP messages.

The parameters :envelope and :conversation-id are not used. The remaining message parameters (:reply-with, :in-reply-to) may be used in specific messages (see below).

1.2 Terminology

Table 2 describes the terminology for computer components.

Constant	**Term**	**Definition**
category	Product Category	These are the categories of products which can be traded (e.g. processors, mouses)
name	Product Name	This is a unique name of a unique product (i.e., an instance of a product category). E.g., mouse-14, processor-5.

perf-ind	Performance Index	A positive integer value (e.g., 20, 500, …)

Table 2. Computer components terminology.

These terms can be combined together to form objects (=specific computer components). For example:

- Mouse_14 is an instance of category Mouse with performance-index 20.
  (category Mouse) (perf-ind 20) (name Mouse-14)
- Processor_5 is an instance of Processor with performance-index 300.
  (category Processor) (perf-ind 300) (name Processor-5)

Table 3 describes the terminology for agents in the market place

Term	**Definition**
agent-ID (also named: public-ID)	A unique (in the context of this marketplace) identifier with which agents can be named (= login name). This agent-ID is used to, e.g., address messages.
Secret-ID	The secret ID (i.e., the password) an agent needs to be able to perform actions and observations in the world. E.g., an agent needs to provide its public-ID and secret-ID to observe its current bank account.
Price	A price is an integer value, *larger than zero* with as upper limit the JAVA-limit.
Object	This is a specific computer component, i.e. a combination of Product-Category, Performance-Index, and Product-Name; e.g., object(Mouse, 20, mouse-14)
Acknowledgement	'yes' or 'no'.
Role	'seller' or 'buyer'
…	…

Table 3. AMMP terminology.

The identity of an agent is guaranteed to be unique. Similarly, it is guaranteed by the Manager that a message sent by agent X, has indeed been sent by agent X. That is, it is not possible to send a message using someone else's identity.

1.3 Market assumptions

The protocol for the Agent-Mediated Market Place is based on a number of assumptions:

- The market is a closed market, no outside influences are needed, nor allowed.
- Wishlists do not change.
- Wishlists, initial objects, and money are allocated when an agent enters the marketplace.
- For consumers, the initial objects form a configuration of a computer which at least contains objects form all essential product categories.
- The progress of an agent is evaluated at the end (i.e., closing of) the marketplace.
- Agents may obtain information on other agents: public-IDs, and an indication whether each agent is a merchant or a consumer.
- When a message contains a :reply-with field with value RW (a String), and you wish to reply to this message, you *have to* use a :in-reply-to field with the value RW. This will enable you to trace message dependencies. This implies that you should always provide your own :reply-with field in your messages.
- *Envelopes* are not discussed in this document, as they are used to combine multiple messages to the same receivers. This is not relevant for the protocol, as discussed in this document.
- Strings are often used in the protocol (and implementation). You have to ignore their capitalisation. So, the string "XyZ" == "xyZ", etc.

2 Agent Communication Protocol

A number of aspects of the protocol are discussed. First of all the logical clock / turns is discussed. Then protocols for various interactions among agents are described. You do not have to use the

:reply-with, :in-reply-to, and :reply-by fields (the manager will not check for compliance), but it will greatly facilitate your ability to keep track of your interactions in the marketplace.

Relating messages.

You can tell whether a message you receive is in response to a message you have previously sent, by looking at the :in-reply-to field. This should be the same as the :reply-with field in your previous message. The implication is that you should always 'obey' the :reply-with field! If you receive a message with particular information in the :reply-with field, and you wish to reply or follow up to that message, than use a :in-reply-to field with that particular information.

On Error messages.

Whenever a protocol is not followed (i.e., an 'odd' message arrives) *no* Error message is sent. So we do not have an explicit error message, such as 'I don't understand message X'.

On ":reply-by" parameters.

Some offers or requests (etc.) are valid for a limited time only. Whenever such a time constraint is violated, the assumption is that the 'negotiation' has ended: the other party is apparently not interested (it may have been killed). Note: if you send a message in turn 212, with a :reply-by 213, then no agent is able to respond.

On message flow

For subprotocols the message flow is described. This shows a message from one agent to another, and how to interpret / react to such a message.

On naming products

In advertisements and bids it is not necessary to use the name of a specific object -- just using the product category and performance index may be sufficient for a negotiation. At transaction time, you always have to specify the precise name of the product to be sold/bought.

2.1 Logical clock / turns

The marketplace operates in 'turns', or 'rounds'. When all agents have indicated that they have finished their communication, a new turn will commence. The clock accurately counts the number of turns that have passed. The clock also gives an inaccurate indication of the number of turns that remain until the market is closed.
On the implications of having a turn-based marketplace:

- in turn X: when agent A sends message M1 to agent B, the message M1 is delivered but may not be read yet by agent B (i.e. is not included in the *current* mailbox).
- in turn X+1: agent B retrieves its new mailbox in which message M1 resides.

In short: in turn X an agent retrieves its mailbox which contains messages received in turn X-1. The flow / sequences of messages described below are influenced by this decision.

2.2 Basic agent state

When an agent has registered and is allowed to communicate, it is in its 'basic agent state'. Whenever it decides to do something, it enters another 'state'. The possible state transitions departing from the basic agent state are interal to the agent, and serve to distinguish a number of protocols. The transitions are listed below:

- basic agent state → advertising state
  The agent has decided to do advertising (see the advertising protocol)
- basic agent state → negotiation state
  The agent has decided to start negotiations (see the negotiation protocol)
- basic agent state → transaction agreement state
  The agent has decided to agree on a transaction (see the transaction agreement protocol)

- basic agent state → finish communication
  The agent has decided to end its communication in this turn (see the world interactions).

Note that a specific agent is able to execute several protocols in parallel.

2.3 Advertising protocol

Explanation. Both buyers and sellers can advertise. An advertisement is used to inform other agents; they are not questions which need a reply. Example advertisements:

- I want to sell something.
- I want to buy something.

In an advertisment (both sell and buy) a list of offers may be included. An offer may be only a product category (possibly with a price), or a combination of a product category and a performance index (possibly with a price), or by referring to a specific object (i.e. product category, performance index, and product name) possibly by mentioning a price.

Message flow. No particular message flow.

- advertising state → advertising state
  by sending any advertisment
- advertising state → basic agent state
  The agent has decided to finish sending advertisements.

2.4 Negotiation protocol

The negotiation protocol consists of:

- negotiating about the price of that object.

By proposing a price for a specific object, or an as-yet-unknown object, of a specific performance index and product category, negotiations are started.

Negotiation about prices includes that both parties, the Seller and the Buyer, propose prices for the object. Any party can also state that it is no longer interested in the object. Also, a party can try to propose a transaction (i.e., 'close the deal').

- Seller proposes a price (i.e. a bid by the seller)
- Buyer proposes a price (i.e., a bid by the buyer)
- Seller indicates that object is no longer for sale.
- Buyer indicates that object is no longer wanted.

Message flow. The following message flow is defined:

step	**agent → agent**	**message**
1	Buyer → Seller	Possible Messages: 1. Proposal for a price for a specific object. → go to step 2. 2. statement that object is no longer wanted from Seller → end of negotiation, go to 'basic state'. * Buyer already ended negotiation * Seller also ends negotiation 3. statement for 'transaction making' → go to that subprotocol. * Buyer is in 'transaction making' mood. * Seller may still propose prices (see section 2.4) 4. any other message is sent to the Seller → Error. Seller assumes that negotiation has ended. * Seller ends negotiation. * Buyer is assumed to have ended negotiation as well.

2	Seller→Buyer	Possible messages: 1. Proposal for a price for a specific object → go to step 1. 2. statement that object is not for sale anymore to Buyer. → end of negotiation, go to 'basic agent state * Seller already ended negotiation * Buyer also ends negotiation 3. statement for 'transaction making' → go to that subprotocol. * Seller is in 'transaction making' mood. * Buyer may still propose prices (see section 2.4) 4. any other message is sent to the Buyer → Error. Buyer assumes that negotiaton has ended. * Buyer ends negotiation. * Seller is assumed to have ended negotiation as well.

2.5 Transaction Agreement protocol

Here we abstract from which agent has first stated 'lets' make a transaction' (i.e., it doesn't matter whether the Buyer or Seller initiated the transaction making). Names to be used in the following message flow: Initiator, Follower.

Penalty

If one, or both, of the agents involved in the transaction do not have their goods or money at time t+3 (see below), then the following penalties are given, depending on the situation they are in:

- Selling agent does *not* have the goods; Buying agent does have the money.
  The selling agent is 'killed' (i.e., made deaf & mute). Buying agent is not changed at all, it didn't get the good and it didn't pay any money.
- Selling agent does have the goods; Buying agent does *not* have the money.
  The selling agent is not changed at all, it didn't give away the good and it didn't get any money.
  The buying agent is 'killed' (i.e., made deaf & mute).
- Selling agent does *not* have the goods; Buying agent does *not* have the money.
  This is a combination of the above: both agents are killed.

The objective is to make sure your agents *survive*.

The situation may arise that Seller and Buyer perform a transaction but disagree on specifics, e.g. the name of the object to be transferred or the price to be paid. In that case the transaction does *not* take place, no penalty is given. And a transaction-failure message is sent to both parties by the Manager. Likewise, if only one agent sends a 'perform transaction X' to the Manager, it just receives a failure message and the transaction does not take place.

Situations

Two situations can be distinguished: sequential or parallel initiation of the transaction agreement. Below for both cases the successful transaction agreement is shown. In the description below, 't+3' is again a time, as registered by the Clock (so not a delta or an increment).

For the *sequential transaction* agreement:

S-time t Initiator → do transact X?
S-time t+1 Follower → Yes, do transact X at time t+3! /* is committing */
S-time t+2 Initiator → Yes, do transact X at time t+3! /* both have committed */
S-time t+3 /* now both agents have to perform the transaction! */
Initiator → perform transact X
Follower → perform transact X

For the *parallel transaction* agreement:

P-time t	Initiator → do transact X?
	Follower → do transact X?
P-time t+1	Initiator → Yes, do transact X at time t+3!
	Follower → Yes, do transact X at time t+3! /* both have committed */
P-time t+2	/* no messaging about the transaction */
P-time t+3	/* now both agents have to perform the transaction! */
	Initiator → perform transact X
	Follower → perform transact X

In both situations it holds that if you respond to a "do transact X?" question, you can still bail out by saying "No." (or you mail a proposal for a new price --- that is the same as 'not agreeing to the last transaction'). When an agent says "Yes, do transact X at time t+3" it has committed. But: both agents need to commit themselves on doing the transaction at time t+3. This means that both agents have to send the messages at time S-t+3 and P-t+3. That is when the actual transfer of money and goods is realised. The Manager always sends a message to the agents who initiated the transaction, describing the result of the transaction (success, killed, failure, or inconsistent; see the messsage description later in this document).

If an agent sends multiple transactions to the Manager at the *same time* (i.e. within the same time), there is no guarantee on the order in which transactions are processed. Don't gamble on selling an object, and thus having enough money to buy another object.

At transaction time, you have to use the specific name of a product. However, at transaction announcement time, the buyer may omit the specific name of the product to be transacted (e.g., when it has not been announced during bidding by the seller). The *seller* always has to provide the specific name of the product.

Message flows

Message flow for sequential trace.

step	**agent → agent**	**message**
t	Initiator → Follower	shall we do a transaction X? → go to step t+1
t+1	Follower → Initiator	Possible messages: 1. Confirmation: yes do transaction X at time t+3 → go to step t+2 2. Proposal for a price for a specific object → Go back to Price Negotiation (an implicit NO to the transaction making, but a YES to keeping the negotiations alive). (The 'Follower' is not finished with negotiations yet.) 3. statement that object is not wanted by/for sale to Initiator. → end of negotiation. → go to 'basic agent state' 4. any other message that is sent to the Initiator → Error → end of negotiation → go to 'basic agent state'
t+2	Initiator → Follower	Possible messages: 1. Confirmation: yes, do transaction X at time t+3. → go to step t+3 2. any other message → Error → end of negotiation → go to 'basic agent state'
t+3	Initiator → Manager	Possible message: 1. perform transaction X
also t+3	Follower → Manager	Possible message: 1. perform transaction X

Message flow for parallel case

step	agent → agent	message
t	Initiator → Follower	shall we do a transaction X? → go to step t+1
also t	Follower → Initiator	shall we do a transaction X? → go to step t+1
t+1	Follower → Initiator	Possible messages: 1. Confirmation: yes do transaction X at time t+3 → go to step t+2 2. any other message that is sent to the Initiator → Error → end of negotiation → go to 'basic agent state'
also t+1	Initiator → Follower	Possible Messages: 1. Confirmation: yes do transaction X at time t+3 → go to step t+2 2. any other message that is sent to the Follower → Error → end of negotiation → go to 'basic agent state'
t+2	<none>	No messaging relevant to this agreement making part. 1. any message received: → Error → end of negotiation. → go back to 'basic agent state'.
t+3	Initiator → Manager	Possible message: 1. perform transaction X
also t+3	Follower → Manager	Possible message: 1. perform transaction X

3 Message definitions

In this section the messages are described in more detail. The row 'content format' describes the information which is in the actual :content parameter of the message. Note that whitespace can be as long as you wish(space, tab, newline, etc.). A number of abbreviations are used, which are variables. These variables need to be instantiated.
Variables employed are:

PC product category instance
PI performance index instance
PN product name instance
P price instance
R role instance
T time instance (i.e. an integer denoting a turn)
XXX agent public ID instance
YYY agent public ID instance

The brackets '[' and ']' indicate that something in between is optional; the brackets '[' and ']' are not part of the message content!

3.1 Advertising: selling something

Kind of message	Advertisement: informing the recipient that the sender wants to sell or buy something.
Communicative Act	cfp /* call for participation */
Content	I am in role R and advertise goods as described by the following list: Product Category PC [with performance Index PI] | [Performance Index PI and Product-Name PN] [for price P] /* Content 1 contains PC, with possible PI, and possibly a Price */ /* Content 2 contains PC and PI, possibly PN, and possibly a Price */

Content format 1	((role R) (((category PC) [(perf-ind PI)] [(price P)]), … ((category PC) [(perf-ind PI)] [(price P)])))
Content format 2	((role R) (((category PC) (perf-ind PI) (name PN) [(price P)]), … ((category PC) (perf-ind PI) (name PN) [(price P)])))
Message Parameters	:reply-with, :reply-by

3.2 Proposal for a price (both seller & buyer)

Kind of message	An agent proposes a price for a specific object. Note that the role the agent has is encoded in the content (Role = 'buyer' or 'seller'). The specific name (PN) of the object does not need to be known when bidding; you're bidding for *a* mouse of performance index 20.
Communicative Act	propose
Content	I have role R, and set the price P for Product Category PC with Performance Index PI [and Product Name PN]
Content format	(bid (role R) (price P) (category PC) (perf-ind PI) [(name PN)])
Message Parameters	:reply-with, :reply-by, :in-reply-to

3.3 End of negotiation

Kind of message	The Buyer or Seller states that the negotiation has ended with respect to an object.
Communicative Act	reject-proposal
Content	I have role R, End of negotiation Product Category PC with Performance Index PI [and Product Name PN]
Content format	(end (role R) (category PC) (perf-ind PI) [(name PN)])
Message Parameters	:in-reply-to

3.4 Transaction announcement

Kind of message	The Sender indicates that he/she wishes to make the following transaction with the Receiver (XXX and YYY are the names of the agents). This is a question: the Receiver may refuse to commit to the proposed transaction. Buyer may omit PN.
Communicative Act	Propose-deal
Content	The following object Product Category PC with Performance Index PI [and Product Name PN] will be moved from XXX to YYY and in return the following money price P will be moved from YYY to XXX. I, as the Sender agent, declare that I have the good/price in my possession.
Content format	(transact-proposal (object (category PC) (perf-ind PI) [(name PN)]) (price P) (move-object (from XXX) (to YYY)) (move-money (from YYY) (to XXX)))
Message Parameters	:in-reply-to, :reply-with, :reply-by

3.5 Transaction response

Kind of message	The Sender indicates to the receiver that the Sender agrees with the transaction making and is willing to commit to the transaction. (XXX and YYY are the names of the agents). Here the entire, full, information on the product is used.
Communicative Act	accept-proposal
Content	At time T the following object Product Category PC with Performance Index PI and Product Name PN will be moved from XXX to YYY and in return the following money price P will be moved from YYY to XXX. I, as the Sender agent, declare that I have the good/price in my possession.
Content format	(transact (time T) (object (category PC) (perf-ind PI) (name PN)) (price P) (move-object (from XXX) (to YYY)) (move-money (from YYY) (to XXX)))
Message Parameters	:in-reply-to, :reply-with, :reply-by

3.6 Transaction making statement

Kind of message	The Sender indicates that he/she wishes to make the following transaction with the Receiver and is also committed to handing over the object or the money. (XXX and YYY are the names of the agents). This message is sent to the "Manager" (both parties will send this message to the Manager).
Communicative Act	agree

Content	The following object Product Category PC with Performance Index PI and Product Name PN will be moved from XXX to YYY and in return the following money price P will be moved from YYY to XXX. I, as the Sender agent, declare that I have the good/price in my possession.
Content format	(transact (object (category PC) (perf-ind PI) (name PN)) (price P) (move-object (from XXX) (to YYY)) (move-money (from YYY) (to XXX)))
Message Parameters	:in-reply-to,

3.7 Transaction result message

Kind of message	The Manager indicates to both the Seller and the Buyer what the result of their transaction was. The variable TR indicates four possible results: (1) "success", the good and the money have been transferred. (2) "killed", you (the agent who receives this) could *not* produce the object or the money, and has been killed. (3) "failed", the other agent could not produce the object or the money, and was killed; the transaction did not take place. (4) "inconsistent", the two parties have send a message with different objects and /or amounts of money, the Manager could not resolve the issue. No transaction took place.
Communicative Act	reject
Content	The transaction has result TR Product Category PC with Performance Index PI and Product Name PN which was supposed to be moved from XXX to YYY in return for the following money price P
Content format	((result TR) (object (category PC) (perf-ind PI) (name PN)) (price P) (move-object (from XXX) (to YYY)) (move-money (from YYY) (to XXX)))
Message Parameters	:reply-with

3.9 Wishlist

The wishlist that each consumer agent obtains, is a message.

Kind of message	The wishlist is sent by an 'outside' user to the agent, via the Manager.
Communicative Act	inform
Content	A list of tuples: < Product Category, Performance Index, Subjective Utility >
Content format	((category PC), (perf-ind PI), (subj-util SU)), … (category PC), (perf-ind PI), (subj-util SU)))
Message Parameters	

3.10 Extra

To allow agents to communicate with each other outside, and beyond, the protocol and messages described above, a specific message type is introduced. It's communicative act is named 'extra', to distinguish it from the messages described above.

Kind of message	The Extra messages is sent to an agent.
Communicative Act	extra
Content	undefined.
Content format	/* any string */
Message Parameters	undefined

4 Instantaneous messages

The interactions described in this section all refer to methods present in the Agent class. These interactions are 'instantaneous': they do not depend on the logical clock, and return as fast as possible results. The result of a method call are usually a Message object. The message content is described below. A value 'null' or 'false' indicates an error in connecting to the Manager.

4.1 Registration to the marketplace

An agent registers by telling the Manager who he is. In return an agent obtains its public-ID (or agent-ID) and its secret-ID. Each agent that has registered then signals 'end of turn' to the Manager. Agents may register simultaneously. When all agents have registered, the marketmanager starts the marketplace by starting the first turn (turn number 1). The secret-ID is managed by the internals of your agent (by the class Agent), you don't need to use it.

4.2 Acquiring list of participants

The list of participants can be acquired at any time. The list contains the public information on the participants in the marketplace. The format of the list is as follows:

Kind of message	List of participants in the market place and an indication whether they are a consumer. The Sender is the "Manager".
Communicative Act	inform
Content	A list of tuples: <agent public ID, Are the consumer y/n>
Content format	(((publicID ID) (consumer BOOLEAN)), … ((publicID ID) (consumer BOOLEAN)))
Message Parameters	

4.3 Observing the bank account

When observing one's bank account the agent has to provide both its public-ID and its secret-ID. In return it obtains the amount of money it has.

Kind of message	Status of the bank account of an agent. The Sender is the "Bank".
Communicative Act	inform
Content	The agent has at the beginning of this turn amount A money on the bank.
Content format	(your-wealth (time T) (amount A))
Message Parameters	

4.4 Observing the store account

When observing one's objects in storage, the agent has to provide both its public-ID and its secret-ID. In return the agent obtains a list of the objects it owns.

Kind of message	Status of the storage account of an agent. The Sender is the "Store".
Communicative Act	inform
Content	The agent has at the beginning of this turn a number of objects in storage.
Content format	(your-goods (time T) (((category PC) (perf-ind PI) (name PN)), … ((category PC) (perf-ind PI) (name PN))))
Message Parameters	

4.5 Observing the norms in the marketplace

To acquire the general knowledge on prices for Product Categories, the agent may obtain norms from the Manager. For each product category, a price interval is given (which does not indicate absolute lower or upper bounds).

Kind of message	The domain knowledge on average products and their average price interval (which does not include the current market situation of supply & demand). The Sender is the "Manager".
Communicative Act	inform
Content	A list of all product categories and low and high prices for each product category.
Content format	(((category PC) (low-price P1) (high-price P2)), … ((category PC) (low-price P1) (high-price P2)))
Message Parameters	

4.6 Observing the (logical) Clock

Two observations on the logical clock can be made.

- *Elapsed time* The time elapsed since the start of the marketplace. In turn 0 all agents register; so when all agents have registered, the logical clock is incremented to 1, and all agents are allowed to retrieve their mailbox (which contains messages received in turn 0: e.g. the wishlist).
- *Remaining time* An estimate is given of the time remaining in the marketplace.

Kind of message	The current time and estimate of the remaining time. The remaining time is a delta: you add it to the current time to calculate the estimated final turn time. E.g.: current time = 34. remaining time is 12. This means: 12 turns left (approximately). Possible ending time: 34 +12 = 46.
Communicative Act	inform
Content	elapsed time is T1, remaining time is T2.
Content format	((elapsed-time T1) (remaining-time T2))
Message Parameters	

4.7 Sending a message

When an agent wishes to send a message, it contacts the Manager and provides the Manager with the message to be sent. Both the public-ID and the secret-ID are needed for this procedure. If the Agent is not 'blocked' (i.e., has not stated 'finished communication' yet in this turn), the message will be delivered to the Manager, which will further process the Message. If the recipient cannot be determined, then the message is not delivered to an agent (return value = false).
If the sender of the message sends an advertisement, and does not have sufficient money to have the advertisement delivered to the recipient (if recipient == ALL, then you need enough money to send advertisment to all), then the message is not delivered (return value = false). The agent is not 'killed'.

4.8 Retrieving one's mailbox

An agent can, as many times at it wants, retrieve its mailbox. In the same turn it will always obtain the same mailbox, which is a list of messages it has been sent in the previous turn. Note that the messages are not ordered in any interesting fashion, the agent itself has to interpret the mailbox and messages itself. To retrieve its mailbox, the agent has to provide both its public-ID and its secret-ID.

4.9 Retrieving one's wishlist

In case an agent has forgotten its wishlist, it can retrieve it from the Manager. The wishlist is always sent to an agent after registration at the MarketPlace.

4.10 Ending communication

The agent uses this world interaction, which is an action: setting a flag, to indicate to the manager that this agent has finished its communication for this turn. This agent will not perform any more world interactions (including sending messages and retrieving one's mailbox). The agent can test whether it is allowed to communicate again.

4.11 Leaving the marketplace

An agent may indicate to the manager that it wishes to leave the marketplace. It is then removed from any subsequent actions, and may not participate in the market anymore (not send messages). Hereby you can lower the costs of sending messages to 'all' for the remaining agents.

Last modification: april 18th, 2000 by Niek Wijngaards

Appendix B

Section B.1 describes problems and solution for this protocol that were not implemented. The protocol used was voted on in the college MICD 2001, and the author had no influence there. The messages proposed for a new protocol are described in section B 2.

B 1. Transaction Message Fraud

The protocol described in appendix A leaves room for agents to cheat. After the exchange of transaction messages between the agents, both agents have to send a message to the manager (page 10 of Appendix A). The message to the manager may be entirely different than the one send to the other agent.

The protocol described in section B 2 closes this hole. When sending a transaction proposal, a copy of the transactions proposal is send to the manager. Since the manager acts between the turns when the agents are put on hold, it can check the transactions messages and send a confirmation that he has received the same transaction proposal as the agent who receives it does. The receiving agent will thus receive 2 transaction proposals in the turn after the sending agent sent it: one from the agent who sent it, and one confirmation from the manager.

B 2.7 is the transaction proposal from the sending agent. The manager processes this before it is send to the receiving agent. The manager makes a copy of this message and sends this copy (message type B 2.9) together with the original to the receiving agent. The same goes for the response (message type B 2.8).

An agent commits itself to the deal the moment the proposal or the response is send. When no response or another proposal concerning the particular deal is send, the agent is no longer committed to the proposal. Transaction confirmations that differ from the proposals they are connected to are punished.

When using this protocol, agents are much earlier certain of a successful closure of a deal. In the protocol of appendix A agents could have an agreement with each other, but they could still send a wrong message to the manager, which would result in an unclosed deal. When an agent receives a transaction proposal in the protocol described here, it knows that the agent who sent it can no longer back out.

B 2. Message types for the new protocol

This section describes the message types used in the protocol. Section B 2.1 describes the notation used, the other sections describe one type each.

B 2.1 Notation:

Notation for format:

[X]: optional X

{X}: one or more X

else: mandatory X

Name notation:

PC: Product Category

PI: Performance Index

P: Price

PN: Product Name

N: Negotiation Identification.

Form of N: A-B-n

where:

- A is the agent that does the first bid
- B is the agent that doesn't
- N is the total number of negotiations that have been started between these two agents.

This way, all Negotiation ID's are unique.

B 2.2 Advertisement for selling

The agent advertises some product it has for sale.

i-want-to-sell

{object ((category PC) (perf-ind PI) [(name PN)]) [(price P)]}

B 2.3 Advertisement for buying

The agent advertises some product it wants to buy.

i-want-to-buy

{object ((category PC) (perf-ind PI) [(name PN)]) [(price P)]}

B 2.4 Error message

A message was not understood properly.

not-understood

(COPY OF MESSAGE)

B 2.5 Proposal (both buyer and seller)

The agent bids a certain price in a negotiation N where it has role R, and for the object X price P [and for the object X_2 price P_2] is being proposed. Note that with this protocol, multiple objects can be negotiated.

bid

(negotiation-ID N)

(role R)

{object ((category PC) (perf-ind PI) [(name PN)]) (price P)}

B 2.6 End of negotiation

The agent ends the negotiation N [which is about object X {and about object X_2}]

end-of-negotiation

(negotiation-ID N)

[{object ((category PC) (perf-ind PI) [(name PN)])}]

Comment: negotiation ID alone would be enough, but one can choose to keep the object in the message, for sake of completeness.

B 2.7 Transaction announcement

The agent proposes the following deal in negotiation N: at time T about object X for price P the object is moved from agent A to agent B (and the money the other way) [objects may also be moved the other way and money back]

transaction-proposal

(negotiation-ID N)

(time T)

{object ((category PC) (perf-ind PI) [(name PN)]) (price P)}

move-object (from-agent A) (to-agent B)

[{object ((category PC) (perf-ind PI) [(name PN)]) (price P)}

move-object (from-agent A) (to-agent B)]

This message has to be send to both the negotiation party and the manager.

B 2.8 Transaction confirmation

The agent confirms the following deal in negotiation N: at time T about object X for price P the object is moved from agent A to agent B (and the money the other way) [objects may also be moved the other way and money back]

transaction-proposal

(negotiation-ID N)

(time T)

{object ((category PC) (perf-ind PI) [(name PN)]) (price P)}

move-object (from-agent A) (to-agent B)

[{object ((category PC) (perf-ind PI) [(name PN)]) (price P)}

move-object (from-agent A) (to-agent B)]

This message has to be send to both the negotiation party and the manager.

B 2.9 Transaction message confirmation by manager

The manager confirms that agent A has send the following transaction announcement to the manager.

Confirmation

(COPY OF MESSAGE)

This message is sent to the negotiation party of the agent that sent the message.

Task Model

Multi-Object-Multi-Lateral-Trading
1. Product Selection
    1.1. Determination Of Possible Products
    1.2. Select Products
2. Party Selection
    2.1. Determination Of Potential Parties
    2.2. Select Parties
        2.2.1. Price Comparison
        2.2.2. Past Negotiation Evaluation
3. Advertisement Determination
4. Negotiation
    4.1. Negotiation Start
        4.1.1. Determination Of Opening Proposal
        4.1.2. Send Opening Proposal
    4.2. Bid Determination
        4.2.1. Determination Of Next Bid
        4.2.2. Send Bid
    4.3. Deal Closure
        4.3.1. Send Deal Confirmations
        4.3.2. Process Deal Consequences
    4.4. Negotiation Management
        4.4.1. Negotiation Analysis
            4.4.1.1. Estimation Of Strategy Other Party
            4.4.1.2. Estimation Of End Price
            4.4.1.3. Continuity Analysis
            4.4.1.4. Formulation Of Analysis Of Negotiation
        4.4.2. Determination Of Own Negotiation Strategy
5. History Maintenance
6. Trading Process Coordination
    6.1. Own Characteristics
    6.2. Trading Process Analysis
        6.2.1. Evaluation Of Trading Strategy
        6.2.2. Evaluation Of Inventory
        6.2.3. Evaluation Of Goals
        6.2.4. Evaluation Of Negotiations
        6.2.5. Evaluation Of Deals
        6.2.6. Evaluation Of Wishlist
        6.2.7. Evaluation Of Total Budget
        6.2.8. Evaluation Of Price Intervals
        6.2.9. Evaluation Of History
        6.2.10. Evaluation Of Incoming Advertisements
        6.2.11. Evaluation Of Participants
        6.2.12. Formulation Of Analysis Of Trading Process
    6.3. Market Price Estimation
    6.4. Price Limit Determination
    6.5. Determination Of Trading Strategy

Appendix C

Task hierarchy flapout.