A Survey of Agent Designs for TAC SCM

Wolfgang Ketter
Dept of DIS
RSM Erasmus University
Rotterdam, NL

John Collins
Dept of CSE
University of Minnesota
Minneapolis, MN

Maria Gini[*]
Dept of CSE
University of Minnesota
Minneapolis, MN

Abstract

An autonomous trading agent is a complex piece of software that must operate in a competitive economic environment. We report results of an informal survey of agent design approaches among the competitors in the Trading Agent Competition for Supply Chain Management (TAC SCM). We identify the problem of decision coordination as a crucial element in the design of an agent for TAC SCM, and we review the published literature on agent design to discover a wide variety of approaches to this problem. We believe that the existence of such variety is an indication that much is yet to be learned about designing such agents.

Introduction

Organized competitions can be an effective way to drive research and understanding in complex domains, free of the complexities and risks of operating in open, real-world environments. Artificial economic environments typically abstract certain interesting features of the real world, such as markets and competitors, demand-based prices and cost of capital, and omit others, such as human resources, secondary markets, taxes, and seasonal demand. The Trading Agent Competition for Supply-Chain Management (Collins *et al.* 2005) (TAC SCM) is based on an economic simulation in which competing autonomous agents operate in a simple supply-chain scenario, purchasing components, managing a factory and warehouse, and selling finished products to customers.

TAC SCM has been an active competition since 2003, and the design of the game has been stable since 2005. More than 50 different teams have participated, and a number of papers have been published that describe agent designs, agent and game analyses, and specific methods for modeling the markets and decision processes in the simulation.

TAC SCM is an interesting challenge for a number of reasons, and different groups have approached the problem from a variety of perspectives, depending on the individual interests and backgrounds of the participants. For example, a team that is primarily interested in developing and testing machine-learning techniques will have a very different approach to the problem than a team that is primarily interested in developing methods to solve constrained optimization problems under uncertainty. To better understand this variety, we conducted an informal survey of many of the active teams in 2007. In this paper, we report on the results of that survey, and we explore in some depth and attempt to classify the variety of approaches we have observed to one of the special challenges in designing a successful agent for TAC SCM, the problem of coordinating the various decision processes.

Overview of the TAC SCM game

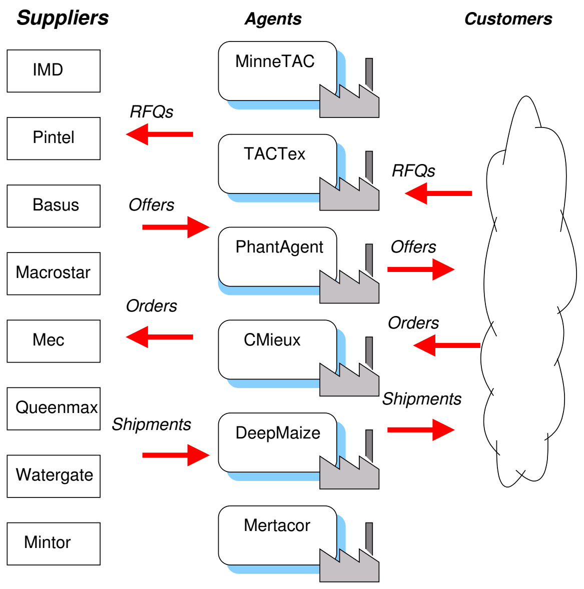

Figure 1: TAC SCM game scenario.

In a TAC SCM game, each of the competing agents plays the part of a manufacturer of personal computers. Figure 1 gives a schematic overview of the TAC SCM game. Agents compete with each other in a procurement market for computer components, and in a sales market for customers. A game runs for 220 simulated days over about an hour of real time. Each agent starts with no inventory and an empty bank account. The agent with the largest bank account at the end

[*]Supported in part by the National Science Foundation under award NSF/IIS-0414466.

of the game is the winner.

Customers express demand each day by issuing a set of Request for Quotes (RFQs) for finished computers. Each RFQ specifies the type of computer, a quantity, a due date, a reserve price, and a penalty for late delivery. Each agent may choose to bid on any subset of the day's RFQs. For each RFQ, the bid with the lowest price will be accepted, as long as that price is at or below the customer's reserve price. Once a bid is accepted, the agent is obligated to ship the requested products by the due date, or it must pay the stated penalty for each day the shipment is late. Agents do not see the bids of other agents, but aggregate market statistics are supplied to the agents periodically. The customer market is segmented into a low-cost segment with five products, a mid-range segment with six products, and a premium segment with five products. Customer demand in each segment varies independently through the course of the game by a random walk with a superimposed Poisson distribution.

Agents assemble computers from parts, which must be purchased from suppliers. When agents wish to procure parts, they issue RFQs to individual suppliers, and suppliers respond with bids that specify price and availability. If the agent decides to accept a supplier's offer, then the supplier will ship the ordered parts on or after the due date. Late shipments are possible because supplier capacity varies from day to day by a mean-reverting random walk. Supplier prices are based on the ratio of demand to current uncommitted capacity.

Once an agent has the necessary parts to assemble computers, it must schedule production in its finite-capacity production facility. Each computer model requires a specific set of parts, and a specified number of assembly cycles. Assembled computers are added to the agent's finished-goods inventory, and may be shipped to customers to satisfy outstanding orders.

Agent decision processes

An agent for the TAC SCM scenario must make the following four basic decisions during each simulated "day" in a competition:

1. decide what parts to purchase, from whom, and when to have them delivered (Procurement).
2. schedule its manufacturing facility (Production).
3. decide which customer RFQs to respond to, and set bid prices (Sales).
4. ship completed orders to customers (Fulfillment).

These decisions are supported by models of the sales and procurement markets, and by models of the agent's own production facility and inventory situation. The details of these models and decision processes are the primary subjects of research for participants in TAC SCM. Many factors, such as current capacity and outstanding commitments of suppliers, and sales volumes and price distributions in the customer market, are not visible to the agents.

Game balance

The design of TAC SCM was carefully tuned over the first three years to make the competition interesting and challenging. The most "obvious" opportunities for strategic manipulation (Ketter *et al.* 2004; Wellman *et al.* 2005) have been eliminated. Agents must manage their reputations with respect to each supplier, to discourage agents from making large requests and then turning down the resulting offers. Suppliers reserve approximately half of their total capacity at the beginning of the game for future demand, which makes it very difficult to "corner" the market for some component type.

The parameters of the game scenario are set to ensure that decision coordination among procurement and sales is at least somewhat challenging. Figure 2 shows the overall balance between supply and demand. It is a histogram of the daily customer RFQ count over 200 games, for 40,000 observations. Superimposed on the histogram are the mean customer demand, the aggregate capacity of six agent factories, and the expected supplier capacity. The key message from this balance is that an agent can expect to buy enough parts to keep its factory busy, but a strategy that simply tries to keep the factory busy all the time is likely to result in a large unsold inventory at the end of an average game[1]. On the other hand, there are some games in which the agents cannot supply all the demand, and the variability of the game can lead to serious imbalances between customer demand for specific products and the availability of parts to build them.

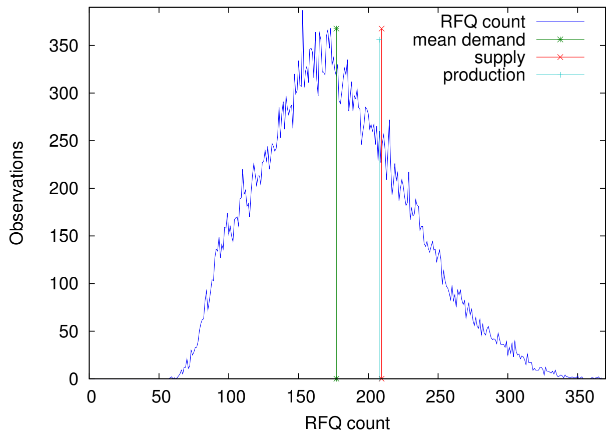

Figure 2: Game balance. Mean customer demand is below the production capacity of all the agents, and below the expected availability of parts in the supplier market.

Designs of agents for TAC SCM

We report findings from an informal survey which we sent to the TAC SCM community via the TAC SCM discussion

[1]This balance was first introduced in the 2005 competition. Price wars were a large problem in the early rounds of that competition until the full-production agents were eliminated

email list in May 2007. The questionnaire was closed by September 2007 and was completed by the best teams in previous tournaments. For the reader's convenience we include the survey in an appendix. Table 1 lists the teams who responded to the questionnaire, and Table 2 lists teams that have published unique TAC SCM agent designs in the literature, but did not take part in our survey. We categorize the survey and literature review results according to our understanding of the research agendas of the teams, and by the specific architectural emphases the teams identified in their agent designs (see Table 3). After we compiled the results into a working paper, we sent it to the participating teams for a peer review process to ensure the quality of our survey. In that way we could be sure that we classified their work correctly. Two teams replied with minor modifications, and the rest approved our findings regarding the research agenda categorization.

The DeepMaize team (Kiekintveld *et al.* 2006) identifies three key issues that a successful TAC SCM agent must address: dealing with substantial *uncertainty* in a highly *dynamic* economic environment, in competition with other self-interested agents whose behavior is naturally *strategic*. We observe a convergence between these issues and our independent findings from the TAC SCM questionnaire. Analysis of the questionnaire results shows how these issues, in conjunction with a variety of general research agendas, has driven the architectural styles adopted by the various teams.

The remainder of this section is organized by primary research agenda. The aim is to show how these various research focus areas have led to specific agent design choices.

Constraint optimization

A supply-chain trading agent must make its decisions subject to a number of internal and external constraints. These constraints apply to parts of the supply-chain, such as procurement (availability of supplies, reputation), production (limited production capacity), sales (limited demand, variable pricing), and fulfillment (shipments limited by finished goods inventory). Nearly all the teams who answered our questionnaire used some form of constraint optimization, so we list here the ones who highlighted it. The teams who focus on realtime optimization, Botticelli (Benisch *et al.* 2004), DeepMaize (Kiekintveld *et al.* 2006), Foreseer (Burke *et al.* 2006), MinneTAC (Ketter *et al.* 2007) use mainly third party optimization packages, including CPLEX[2], Ilog OPL[3], and lp_solve[4]. An exception is CMieux (Benisch *et al.* 2006) which uses an internally-developed algorithm to solve a continuous knapsack problem for pricing customer offers.

Machine learning

Many teams use machine learning algorithms to learn from historical market data and many agents have some ability to learn during operation to adapt to changing situations. CMieux (Benisch *et al.* 2006), DeepMaize (Kiekintveld *et al.* 2007), MinneTAC (Ketter *et al.* 2007), and TacTex (Pardoe & Stone 2007) identified the need to support learning and adaptation as primary concerns in the design of their agents. CMieux, DeepMaize, and TacTex use the Weka (Witten & Frank 2005)[5] machine learning tool set. MinneTAC uses Matlab[6] in combination with the Netlab[7] neural network toolbox to develop and train market models (economic regimes), and to bootstrap the agent with the resulting models. At runtime, MinneTAC updates and adjusts those models using feedback and machine learning algorithms.

Management of dynamic supply chains

Traditionally, supply chains have been created and maintained through the interactions of human representatives of the various enterprises (component suppliers, manufactures, wholesalers/distributors, retailer and customers) involved. However, the recent advent of autonomous trading agents opens new possibilities for automating and coordinating the decision making processes between the various parties involved. A good overview of multiagent based supply chain management outside of TAC SCM is given by Chaib-draa and Müller (2006). The TAC SCM simulation is an abstract model of a highly dynamic direct sales environment (Chopra & Meindl 2004), as exemplified by Dell Inc.[8], for procurement, inventory management, production, and sales.

Several teams have a strong research focus on dynamic supply-chain behavior. The best example is probably CMieux (Benisch *et al.* 2006), developed by the e-Supply Chain Management Lab at Carnegie-Mellon University. Others who identified this focus area include Foreseer (Burke *et al.* 2006), Mertacor (Kontogounis *et al.* 2006), MinneTAC (Ketter *et al.* 2007), and Tiancalli (Galindo, Ayala, & Lopez 2006). These teams strive for high flexibility in their agent design, so that they can easily accommodate changes.

Scalable autonomous agents

The CrocodileAgent (Podobnik, Petric, & Jezic 2006) group is part of a larger group that focuses on autonomous agents for management of large-scale telecommunication networks. They view TAC SCM as a challenge in building an agent that can operate in a dynamic environment, but they are also concerned with scalability and other issues that go far beyond TAC SCM. They base their design on the JADE (Bellifemine, Caire, & Greenwood 2007)[9] framework, which is well-proven for large-scale multi-agent systems with a huge number of entities on both the consumer, the business, or both sides of a supply chain (such as a telecom environment).

[2]`http://www.ilog.com/products/cplex/`
[3]`http://www.ilog.com/products/oplstudio/`
[4]`http://sourceforge.net/projects/lpsolve`
[5]`http://www.cs.waikato.ac.nz/ml/weka/`
[6]`http://www.mathworks.com/`
[7]`http://www.ncrg.aston.ac.uk/netlab/`
[8]`http://www.dell.com/`
[9]`http://jade.tilab.com/`

Team	University	Team contact
Botticelli (B)	Brown University (USA)	Amy Greenwald Victor Naroditskiy
CMieux (CM)	Carnegie Mellon University (USA)	Michael Benisch Norman Sadeh
CrocodileAgent (CA)	University of Zagreb (Croatia)	Ana Petric Vedran Podobnik
DeepMaize (DM)	University of Michigan (USA)	Chris Kiekintveld Michael Wellman
Foreseer (F)	Cork Constraint Computation Centre (Ireland)	Kenneth Brown David Burke
Mertacor (M)	Aristotle University of Thessaloniki (Greece)	Pericles Mitkas Andreas Symeonidis
MinneTAC (MT)	University of Minnesota (USA)	John Collins Wolfgang Ketter
Southampton (S)	University of Southampton (UK)	Minghua He Nick Jennings
TacTex (TT)	University of Texas (USA)	David Pardoe Peter Stone
Tiancalli (T)	Benemerita Universidad Autonoma de Puebla (Mexico)	Darnes Ayala Daniel Galindo

Table 1: Teams participating in the TAC SCM architecture design survey.

Team	University	Authors
RedAgent (R)	McGill University	Philipp Keller, Felix-Olivier Duguay, Doina Precup
PSUTAC (P)	Pennsylvania State University (USA)	Shuang Sun, Viswanath Avasarala, Tracy Mullen, John Yen
PhantAgent (PA)	Politehnica University of Bucharest (RO)	Mihai Stan, Bogdan Stan, Adina Magda Florea

Table 2: Unique TAC SCM agent designs from the literature apart from our survey.

Architecture

Some teams identified a specific focus on software architecture for autonomous agents. CrocodileAgent (Podobnik, Petric, & Jezic 2006) and Southampton SCM (He *et al.* 2006) have structured their agent decision processes around the the IKB (Information, Knowledge, and Behavior) model (Vytelingam *et al.* 2006), a three layered agent-based framework. The first layer contains data gathered from the environment, the second knowledge extracted from the data, and the third encapsulates the reasoning and decision-making components that drive agent behavior. An advantage of using JADE is that the separation of I, K & B layers enables physical distribution of layers on multiple computers. Information layer agents parse out information from the TAC SCM server messages, while information and knowledge flows are implemented as JADE agent communication-based messages. The CrocodileAgent team reported that the separation of I, K & B layers and the introduction of JADE agent platform to the TAC SCM domain causes a much more complex system with lots of intercommunication. CrocodileAgent deals with this overhead in TAC SCM, since their main agenda is to use JADE agents in their research regarding the next generation of telecommunication networks.

The MinneTAC team has developed an architecture that separates the four major decision processes into separate components, and attempts to minimize the learning curve for a researcher who wishes to work on a specific subproblem.

PSUTAC (Sun *et al.* 2004) employs an expert system for decision making. The agent has two functional modules: a database that keeps track of all transactional data and a knowledge base centered kernel. The database records daily transactions and decisions. As opposed to the machine learning agents described above, in PSUTAC human designers analyze data and code knowledge for the agent offline and then transfer it into the knowledge base, which is implemented in Jess[10].

Empirical game theory

The DeepMaize (Jordan & Wellman 2007) team is developing methods for empirical game-theoretic analysis as a major research emphasis. They employ an experimental methodology for explicit game-theoretic treatment of multiagent systems simulation studies. For example, they have

[10]`http://herzberg.ca.sandia.gov/`

Research Agenda	Team	Architectural Emphasis
Constraint optimization	B, CM, F, MT	3rd party packages
	CM, DM	Internal optimization methods
Machine learning	CM, DM, MT, TT	External analysis framework, 3rd party packages
Dynamic supply-chain	CM, F, M, T	Flexibility
Scalability	CA	Distributed Computation
Architecture	CA	IKB model for physical distribution
	MT	Blackboard architecture with evaluator chain
	P	Knowledge-based architecture with central repository
	R	Sequence of internal market places
Empirical game theory	DM	External analysis framework
Decision coordination	CM, DM, M, PA, R, S	Modularity
Dealing with uncertainty	B, F, S, MT	Modularity
	PA	Simple heuristics

Table 3: Identified research agendas of teams and their architectural emphases.

developed a bootstrap method to determine the best configuration of the agent behavior in the presence of adversary agents (Jordan, Kiekintveld, & Wellman 2007). They also use game-theoretic analysis to assess the robustness of tournament rankings to strategic interactions. Many of their experiments require hundreds to thousands of simulations with a variety of competing agents. To support their work they have developed an extensive framework for setting up and running experiments, and for gathering and analyzing the resulting data[11].

Dealing with uncertainty

TAC SCM is designed to force agents to deal with uncertainty in many dimensions. Sodomka et al. (Sodomka, Collins, & Gini 2007) provide a good overview of the sources of uncertainty in the context of an empirical study of agent performance. The Botticelli group clearly identifies the problem of dealing with uncertainty as one of their main research goals in (Benisch *et al.* 2004). Southampton-SCM (He *et al.* 2006) employs a bidding strategy that uses fuzzy logic to adapt prices according to the uncertain market situation.

The key architectural decision in Foreseer (Burke *et al.* 2006) is that all constraint optimization models used in the agent are subject to uncertainty. In Foreseer both the customer bidding model and the component procurement model are subject to uncertainty. Both are parameterized, such that probability distributions representing the current *belief* of the state of the market can be passed into and used by the models. Given the uncertainty of the market, these beliefs allow Foreseer to represent different possible market states with different probabilities.

MinneTAC (Ketter *et al.* 2007; 2008) observes conditions in the customer market to characterize the microeconomic situation of the market, economic regimes, and predict future market regimes. This approach represents the uncertainty of the market as a probability distribution across economic regimes. The agent can use this information to make both tactical decisions, such as pricing, and strategic decisions, such as product mix and production planning.

[11]P. Jordan, private communication

Decision coordination

The supply-chain scenario places a premium on effective coordination of decisions affecting multiple markets and internal resources. There are many different ways to coordinate these decisions, and the next section is dedicated to examining a variety of these methods in more detail.

In this kind of setting it is advantageous to be able to replace individual decision-oriented components of an agent and compare their performances, e.g. two different sales modules compared on final profit. Many teams mentioned "modularity" as a specific goal for their designs, but we think that this is really a precondition that allows this sort of experimental flexibility.

Decision coordination is an explicit emphasis for the CMieux (Benisch *et al.* 2006), DeepMaize (Kiekintveld *et al.* 2006), Mertacor (Chatzidimitriou *et al.* 2007), and Southampton (He *et al.* 2006) teams. This problem is commonly viewed as one of enabling independent decision processes to coordinate their actions while minimizing the need to share representation and implementation details.

Southampton uses the hierarchical IKB approach, in which the Knowledge layer can be viewed as a type of blackboard. DeepMaize (Kiekintveld *et al.* 2006) treats the combined decisions as a large optimization problem, decomposed into subproblems using a "value-based" approach. The result is that much of the coordination among decision processes is effectively managed by assigning values to finished goods, factory capacity, and individual components over an extended time horizon.

We now focus in some detail on the decision coordination methods used by a variety of agent designs.

Agent coordination mechanisms

A successful agent design must provide a way to coordinate the agent's procurement, sales, production, and fulfillment decisions effectively, whether or not the problem of decision coordination is an explicit element of a group's research agenda. This is a fundamental challenge of the TAC SCM scenario. Two of the four agent decision problems, procurement and sales, are dominated by the variability of the game scenario and are strongly affected by the actions of other agents, while the production-scheduling and fulfillment decisions are internal to the agent and less affected by the inherent variability in the game. Because of this, some agent designs simply fold fulfillment into the sales problem, and production scheduling is sometimes also bundled into sales, especially for agents that use a make-to-order production strategy.

Simply stated, a solution to coordination problem will maximize (expected) profit over an entire game, subject to availability of individual part types in the supplier market, demand in the customer market, and capacity of the agent's factory. Of course, prices and availability in the supplier market are at least partly determined by the behavior of other agents in the simulation. In addition, prices in the customer market are almost always determined by the behavior of the other agents, since competition almost always keeps prices well below customer reserve prices.

As we shall see, many approaches to the coordination problem have been tried, and there is little evidence from tournament standings that any of these approaches dominates the others. In fact, a study by Jordan et al. (2007) has shown that no single dominant strategy has yet been found, and our analysis shows that the top three agents in the Jordan study, namely TacTex, DeepMaize, and PhantAgent, use different coordination mechanisms. We do know that the "push" strategy that was popular in the 2003 and 2004 competitions (for example, Benisch et al. (2004)) is not effective, because the factory can produce more than can be sold at a profit, at least in expectation. This approach attempts to purchase enough parts early in the game to keep factory utilization high for the entire game, thereby eliminating procurement from the coordination problem.

In the following sections, we explore the variety of coordination approaches that we have observed among published agent designs. We note that none of them have tried to solve the general problem in its entirety, presumably because the variability inherent in the simulation and the difficulty of predicting the behaviors of other agents have so far defeated all attempts to do so. Therefore, what we see is that each design has chosen a more manageable approach, one that simplifies the problem through approximations, through heuristics, and through focus on much shorter time horizons than the entire game.

Predicted sales volume

Because the balance of supply, demand, and production capacity in the simulation design has defeated a simple "push" approach to coordination, the next obvious choice would seem to be adoption of a "pull" approach, in which sales activities pull finished goods through the factory, which in turn pulls in components through the procurement market.

A good example of this approach is SouthamptonSCM (He *et al.* 2006). This agent was a finalist in the 2004 competition, and placed second in 2005. Figure 3 is a schematic representation of this design. In SouthamptonSCM, a Customer Agent uses fuzzy reasoning to compute offer prices, based on inventory level, customer demand, and time in the game[12]. Priority is given to the products with the highest expected per-unit profit. The Component Agent buys a portion of its components with long lead-times, because prices tend to be lower with longer lead times. The remaining component inventory is purchased with shorter lead times, in response to observed customer demand and to depletion of inventory by sales to customers. The Factory Agent primarily builds outstanding customer orders, and if it has spare capacity and available parts, it builds up a modest inventory of finished goods.

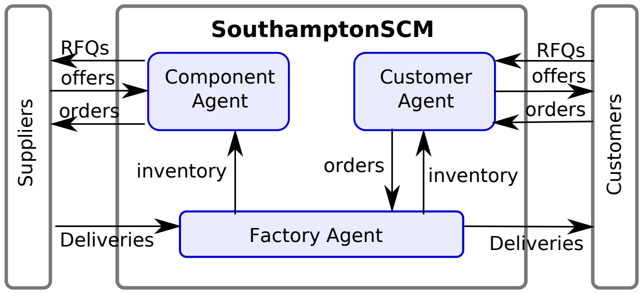

Figure 3: SouthamptonSCM: Coordination primarily by the Customer Agent.

The MinneTAC agent (Collins, Ketter, & Gini 2008), shown in Figure 4, can be configured in a number of different ways, but the configuration used in the 2007 competition solved a linear program each day to maximize expected profit over a 20-day horizon, subject to constraints on production capacity, customer demand, and inventory. The current-day "sales quota" was used by the Sales Manager to set prices in the customer market, and future-day quotas were used by the Supplier Manager to drive procurement. MinneTAC was a finalist in the 2005 and 2006 competitions.

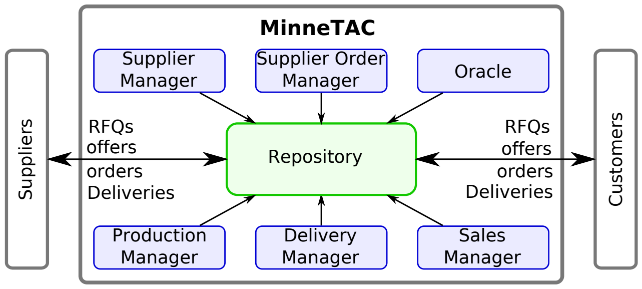

Figure 4: MinneTAC: Coordination through the Repository, details depend on configuration.

As we can see from Figure 4, MinneTAC uses a very different design approach from the other agents we examine

[12]A separate rule set is used near the end of the game, because of the need to exhaust inventory and because prices tend to be much more volatile late in the game

here. The Repository acts as a "blackboard", and the various components interact only through the Repository. The Oracle component is a wrapper for a large number of small modules, called "Evaluators", that can be strung together as specified in a configuration file to do the necessary analysis and prediction tasks requested by the decision components. The actual coordination among decision components happens because they share some of those Evaluators. Specifically, both the Sales Manager and the Supplier Manager use the sales quotas produced by one of the Evaluators.

Future production schedule

DeepMaize (Kiekintveld *et al.* 2006) coordinates its decisions through a principled approach called "value-based decomposition". In this approach, a long-term production schedule is constructed by incrementally adding the products that are expected to return the highest profit. The general scheme is shown schematically in Figure 5.

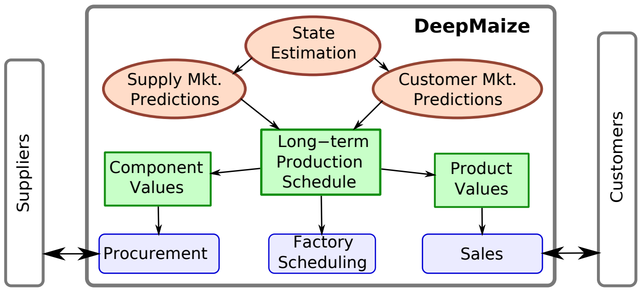

Figure 5: DeepMaize: Coordination through a long-term production schedule, using value-based decomposition.

This approach depends on pricing models in both the customer and supplier markets that effectively capture price-quantity tradeoffs. The two prediction components shown in the diagram, along with an off-line machine-learning process, are responsible for producing those models. Given a long-term production schedule, the Procurement module attempts to provide the necessary components to fill it, and Sales uses it to set prices in the customer market. DeepMaize has been a finalist in all of the TAC SCM tournaments. It placed third in 2006 and 2007.

Inventory management

Three published agent designs appear to focus on an inventory model to coordinate decisions. Mertacor (Chatzidimitriou *et al.* 2007; Kontogounis *et al.* 2006) is the clearest example. As we see in Figure 6, an "Inventory Manager" component is the central element in this design. Mertacor uses an "Assemble to Order" approach, which is recommended in the literature on inventory management for situations where assembly times are significantly shorter than procurement lead times. The Inventory Manager attempts to maintain component stocks above a minimum threshold, subject to committed and expected sales, and to committed deliveries from suppliers. Mertacor placed third in the 2005 competition.

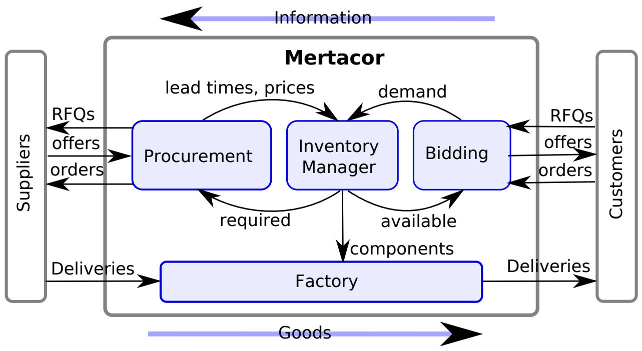

Figure 6: Mertacor: Coordination through the Inventory Manager.

PhantAgent (Stan, Stan, & Florea 2006) is another design that appears to focus on inventory management, although as we see in Figure 7, the inventory management function is conceptually combined with the procurement function in a Component Module. The goal of the Component Module is to maintain expected stocks of each component type within narrow bounds throughout the game. It computes expected stocks for each component for each day until the end of the game, and formulates new supplier orders to make up any deficits. PhantAgent placed second in 2006, and first in the 2007 competition.

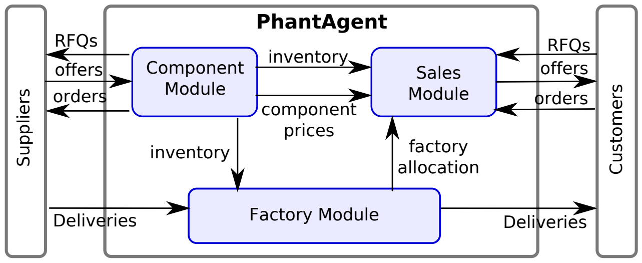

Figure 7: PhantAgent: Coordination is by inventory control, originating in the Procurement module.

PhantAgent is interesting in another way. They deal with the inherent complexity and uncertainty of the TAC SCM environment, and the resulting strategic inter-dependencies between their different agent modules, using heuristic approximations rather than optimization algorithms. Their assessment is that finding optimal solutions to the different sub-problems does not always lead to the best overall performance.

Another agent whose decision coordination mechanism seems focused on inventory control is CrocodileAgent (Podobnik, Petric, & Jezic 2006; Petric, Podobnik, & Jezic 2007). The agent drives procurement to maintain expected component inventory stocks within defined minimum and maximum bounds. Similarly, Production operates to maintain a finished goods inventory within pre-defined bounds. Sales then bids on customer requests using a simple pricing algorithm, in an attempt to sell products, profitably, as fast as they are being produced. When demand is low, the profitability constraint causes inventory to back up, and production and procurement to slow down.

Central strategy module

An agent that has very clearly separated the decision coordination issues from the details of procurement, sales, and production scheduling is CMieux (Benisch *et al.* 2006), a finalist in the 2007 competition. A schematic diagram of the CMieux design is shown in Figure 8.

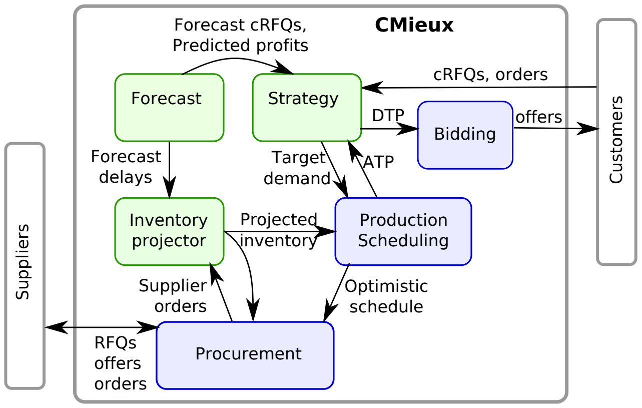

Figure 8: CMieux: Coordination by a separate Strategy module.

The Strategy module sets overall goals for the remainder of the system, such as the portion of expected demand to target, and the portion of the production schedule (ATP, the products Available to Promise) that should be sold to customers (DTP, products Desired to Promise). The Forecast module observes the markets and makes predictions about demand, prices, and delays in supplier shipments. The Inventory Projector combines that with current inventories and expected supplier deliveries to generate inventory projections over time. Procurement uses the projected inventory along with an optimistic version of the production schedule (what Production would expect to build if there were no inventory constraints) to decide what to order from suppliers, and supplies Inventory Projector with actual supplier orders.

Separate supply and demand models

The design of TacTex (Pardoe & Stone 2007) is quite different from the others we have reviewed, in the sense that it does not try to centralize decision coordination at all. Instead, it employs a Supply Manager that interacts with suppliers and models the supply market, and a Demand manager that interacts with customers and models the customer market. Coordination is achieved by communication between these two models. TacTex has been a very strong competitor, placing first in 2005 and 2006, and second in 2007.

In this design, the Supply Manager attempts to minimize the cost of procuring the components requested by the Demand Manager, and provides in return an inventory projection including current inventory and expected future deliveries, along with replacement cost estimates for each compo-

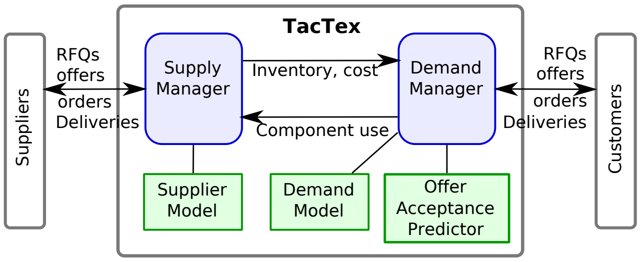

Figure 9: TacTex: Coordination is by communication of inventory, cost, and projected usage data between the Supply Manager and the Demand Manager.

nent type. The Demand Manager, in turn, seeks to maximize the agent's profits from sales, subject to constraints from the customer market, its own production capacity, and the information provided by the Supply Manager.

Internal markets

RedAgent (Keller, Duguay, & Precup 2004) is a unique approach to agent design. It won the first year's competition in 2003, but did not do well in 2004 and was never updated after the rule change in 2005.

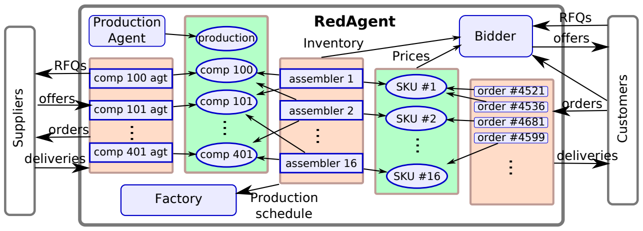

Figure 10: RedAgent: Coordination by a sequence of internal markets.

As we can see from Figure 10, RedAgent manages the flow of components from suppliers through production and into sales and fulfillment of customer orders through a series of internal markets. The Bidder observes its inventory status and the current prices in its internal finished-goods markets, and makes offers to customers. Customer orders then compete for products in the internal product markets, which are supplied by assemblers for each product type. Those assemblers in turn compete for components in internal component markets, which are supplied by individual component agents. The component agents then interact with suppliers in order to set prices and supply their markets. RedAgent used loosely-coupled "sub-agents" competing with each other in internal auction-based markets for finished goods, production capacity, and components. This achieved a radical decoupling of the various components, but proved to be uncompetitive after the game design was adjusted in 2005 to defeat some of the simplest approaches that lacked adequate coordination among decisions. Specifically, agents that focused procurement only on keeping the factory in full production found themselves overproducing when the balance between factory capacity and expected customer demand was adjusted.

Conclusions and Future Work

We have presented a brief overview of agent design ideas and architectures for TAC SCM, using information both from a survey of agent development teams and from their published results. The overall survey outcome shows that there are common themes emerging from the different research groups on how to design a successful agent architecture. These include common software engineering quality criteria such as modularity, low coupling, and separation of concerns, in addition to more problem-specific approaches such as coordination of sales and procurement through internal models of inventory and prices, and assigning current and future value to inventory and production resources. There are also some strong differences such as how to organize the communication between the different modules and which modules should own the data for specific tasks. These findings, and the fact that after several years of competition there is still much to be learned, suggest that the recipe for a full competent supply-chain trading agent is still an unsolved problem, even for an abstract, constrained environment like TAC SCM.

We are planning to execute another agent design survey in a couple of years, and compare the results to the material presented here. We will include in the next survey more detailed questions about the specific research agendas of the different teams, and we will provide as a guideline the current categorization. Of course, research agendas might change and new teams might have different agendas than we found in this survey. Furthermore, we will include a question about agent coordination mechanisms and provide our findings from this research as a reference. It will be interesting to see the extent to which variations on the designs we have reported here will spread to other agent teams. This might indicate that some basic building blocks have been discovered.

Appendix: TAC SCM Design Questionnaire

We sent out to the TAC SCM community the following survey questions:

1. Which team do you represent? What has been your role on the team?
2. What are the main goals of your design apart from winning the game?
3. What are your organizing design principles (architectural style, major modules and responsibilities)?
4. What are the strengths and weaknesses of your design? In other words, what is easy and what is hard to do given your design? To what extent do you feel your design has met your goals?
5. If you have been in the competition for more than two years, have you made significant changes in your design and why?
6. Does your design represent a significant departure from the Agentware package?
7. Which significant 3rd party packages have you used, e.g. Weka, CPlex, Apache Excalibur, Jade, etc.?
8. Have you based your design on a publicly-available agent design, like TacTex, GeminiJK, or MinneTAC?
9. Have you published information about your agent design? If yes, where?

References

Bellifemine, F. L.; Caire, G.; and Greenwood, D. 2007. *Developing Multi-Agent Systems with JADE.* Wiley.

Benisch, M.; Greenwald, A.; Grypari, I.; Lederman, R.; Naroditskiy, V.; and Tschantz, M. 2004. Botticelli: A supply chain management agent designed to optimize under uncertainty. *ACM Trans. on Comp. Logic* 4(3):29–37.

Benisch, M.; Sardinha, A.; Andrews, J.; and Sadeh, N. 2006. CMieux: adaptive strategies for competitive supply chain trading. In *Proc. of 8th Int'l Conf. on Electronic Commerce*, 47–58. ACM Press.

Burke, D. A.; Brown, K. N.; Tarim, S. A.; and Hnich, B. 2006. Learning market prices for a real-time supply chain management trading agent. In *AAMAS 2006 Workshop on Trading Agent Design and Analysis*.

Chaib-draa, B., and Müller, J. 2006. *Multiagent based Supply Chain Management (Studies in Computational Intelligence).* Springer-Verlag New York, Inc. Secaucus, NJ, USA.

Chatzidimitriou, K. C.; Symeonidis, A. L.; Kontogounis, I.; and Mitkas, P. A. 2007. Agent Mertacor: A robust design for dealing with uncertainty and variation in SCM environments. *Expert Systems with Applications* Article in Press.

Chopra, S., and Meindl, P. 2004. *Supply Chain Management.* New Jersey: Pearon Prentice Hall.

Collins, J.; Arunachalam, R.; Sadeh, N.; Ericsson, J.; Finne, N.; and Janson, S. 2005. The Supply Chain Management Game for the 2006 Trading Agent Competition. Technical Report CMU-ISRI-05-132, Carnegie Mellon University, Pittsburgh, PA.

Collins, J.; Ketter, W.; and Gini, M. 2008. Flexible decision support in a dynamic business network. In van Heck et al., E., ed., *Edited Volume of the 3rd Smart Business Network Conference*. Springer Verlag.

Galindo, D.; Ayala, D.; and Lopez, F. L. Y. 2006. Statistic analysis for the Tiancalli agents on TAC SCM 2005 and 2006. In *Proc. 15th Int'l Conf. on Computing*, 161–166.

He, M.; Rogers, A.; Luo, X.; and Jennings, N. R. 2006. Designing a successful trading agent for supply chain management. In *Proc. of the Fifth Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, 1159–1166.

Jordan, P. R., and Wellman, M. P. 2007. Best-first search for approximate equilibria in empirical games. In *Workshop on Trading Agent Design and Analysis (TADA)*, 11–18. AAAI.

Jordan, P. R.; Kiekintveld, C.; and Wellman, M. P. 2007. Empirical game-theoretic analysis of the tac supply chain game. In *Proc. of the Sixth Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, 1188–1195.

Keller, P. W.; Duguay, F.-O.; and Precup, D. 2004. Redagent - winner of the TAC SCM 2003. *SIGecom Exchanges* 4(3):1–8.

Ketter, W.; Kryzhnyaya, E.; Damer, S.; McMillen, C.; Agovic, A.; Collins, J.; and Gini, M. 2004. Analysis and design of supply-driven strategies in TAC-SCM. In *Workshop: Trading Agent Design and Analysis at the Third Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, 44–51.

Ketter, W.; Collins, J.; Gini, M.; Gupta, A.; and Schrater, P. 2007. A predictive empirical model for pricing and resource allocation decisions. In *Proc. of 9th Int'l Conf. on Electronic Commerce*, 449–458.

Ketter, W.; Collins, J.; Gini, M.; Gupta, A.; and Schrater, P. 2008. Detecting and Forecasting Economic Regimes in Multi-Agent Automated Exchanges. *Decision Support Systems* in publication.

Kiekintveld, C.; Miller, J.; Jordan, P. R.; and Wellman, M. P. 2006. Controlling a Supply Chain Agent Using Value-Based Decomposition. In *Proc. of 7th ACM Conf. on Electronic Commerce*, 208–217.

Kiekintveld, C.; Miller, J.; Jordan, P.; and Wellman, M. 2007. Forecasting market prices in a supply chain game. In *Proc. of the Sixth Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, 1318–1325.

Kontogounis, I.; Chatzidimitriou, K.; Symeonidis, A.; and Mitkas, P. 2006. A Robust Agent Design for Dynamic SCM Environments. *Proceedings of the 4th Hellenic Joint Conference on Artificial Intelligence (SETN), Heraklion, Greece* 127–136.

Pardoe, D., and Stone, P. 2007. An autonomous agent for supply chain management. In Adomavicius, G., and Gupta, A., eds., *Handbooks in Information Systems Series: Business Computing*. Elsevier.

Petric, A.; Podobnik, V.; and Jezic, G. 2007. The CrocodileAgent: Designing a robust trading agent for volatile e-market conditions. In Nguyen, N. T.; Grzech, A.; Howlett, R. J.; and Jain, L. C., eds., *Agent and Multi-Agent Systems: Technologies and Applications*, volume 4496 of *LNAI*. Springer-Verlag. 597–606.

Podobnik, V.; Petric, A.; and Jezic, G. 2006. The crocodileagent: Research for efficient agent-based cross-enterprise processes. In Meersman, R.; Tari, Z.; and Herrero, P., eds., *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, volume 4277. Springer-Verlag. 752–762.

Sodomka, E.; Collins, J.; and Gini, M. 2007. Efficient statistical methods for evaluating trading agent performance. In *Proc. of the Twenty-Second National Conference on Artificial Intelligence*, 770–775.

Stan, M.; Stan, B.; and Florea, A. M. 2006. A Dynamic Strategy Agent for Supply Chain Management. *Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing* 0:227–232.

Sun, S.; Avasarala, V.; Mullen, T.; and Yen, J. 2004. PSUTAC: A Trading Agent Designed from Heuristics to Knowledge. In *Workshop on Trading Agent Design and Analysis at AAMAS*, 15–20.

Vytelingam, P.; Dash, R. K.; He, M.; and Jennings, N. R. 2006. Trading strategies for markets: A design framework and its application. In Poutré, H. L.; Sadeh, N. M.; and Janson, S., eds., *Agent-Mediated Electronic Commerce: Designing Trading Agents and Mechanisms*, volume 3937 of *Lecture Noted in Artificial Intelligence*. Springer-Verlag. 171–186.

Wellman, M. P.; Estelle, J.; Singh, S.; Vorobeychik, Y.; Kiekintveld, C.; and Soni, V. 2005. Strategic interactions in a supply chain game. *Computational Intelligence* 21(1):1–26.

Witten, I. H., and Frank, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition*. San Francisco, CA: Morgan Kaufmann.