15. Flexible Decision Support in a DynamicBusiness Network

John Collins[1], Wolfgang Ketter[2] and Maria Gini[3]

[1]Computer Science and Engineering, University of Minnesota, United States of America, jcollins@cs.umn.edu

[2]Decision and Information Sciences, RSM Erasmus University, The Netherlands, wketter@rsm.nl

[3]Computer Science and Engineering, University of Minnesota, United States of America, gini@cs.umn.edu

Abstract

We present the design of a service oriented architecture which facilitates flexible managerial decision making in dynamic business networks. We have implemented and tested this architecture in the MinneTAC trading agent, which is designed to compete in the Supply Chain Trading Agent Competition (Collins et al., 1998). Our design enables managers to break out decision behaviors into separate, configurable components, and allows dynamic construction of analysis and modeling tools from small, single-purpose "evaluator" services. The result of our design is that the network can easily be configured to test a new theory and analyze the impact of various approaches to different aspects of the agent's decision processes, such as procurement, sales, production, and inventory management. Additionally we describe visualizers that allow managers to see and manipulate the configuration of the network, and to construct economic dashboards that can display the current and historical state of any node in the network.

Introduction

Organizations in business networks have a growing need for intelligent software that can assist managers by gathering and analyzing information, making recommendations, and supporting business decisions. Advanced decision support systems and autonomous software agents promise to address this need by acting rationally on behalf of humans in numerous application domains. Examples include procurement (Sandholm, 2007; CombineNet, 2006), scheduling and resource management (I2, 2006; Collins, Bilot, Gini, & Mobasher, 2001), and personal information management (Berry et al., 2006). The recent advent of *Smart Business Networks* (SBN) (Vervest, Preiss, Heck, Pau, 2004; Vervest, van Heck, Preiss, & Pau, 2005; van Heck, & Vervest, 2007) extends the area of traditional business processes and gives rise to new challenges, especially in the area of dynamic and modular

P.H.M. Vervest et al. (eds.) *The Network Experience*

business process management, by enabling integration of legacy systems and by providing advanced tools to facilitate human managerial decision making.

We make four major contributions to the SBN literature. One of the major theoretical tenets of SBNs is the ability of actors to quickly connect to other actors to achieve specific business objectives and then disconnect when a task is finished. Our first contribution in this paper extends the SBN literature through the design and implementation of a highly configurable and flexible decision support system that dynamically connects to different nodes of a business network and disconnects them when no longer needed. Our second contribution is the vision of goal directed service composition. This allows business services with formal semantic descriptions to be composed and validated. Thirdly, we are developing a tool to enable managers to visualize, understand, and validate the theoretically designed decision chain with a graphical representation of the actual network chain. Finally, we have developed a flexible economic dashboard architecture that can be dynamically connected to selected nodes to visualize their real-time status, current parts and finished goods inventory positions, risk and reward management, and the like. This architecture can greatly empower business network managers in their understanding of the overall business network structure and facilitate real-time managerial decision making. Currently, we are working on an even more interactive version of this dash-board which allows the human decision maker to interact with the business network to make structural changes.

Since operating on real world business networks has high risks, and might cause serious business problems when not done properly, we tested our architecture and algorithms on a supply-chain testbed, the Trading Agent Competition for Supply Chain Management (Collins et al., 2005) (TAC SCM). We describe the implementation of our flexible decision support system and demonstrate its value using as an example MinneTAC (Collins, Ketter, & Gini, 2008), an autonomous agent that performs coordinated buying, selling, production scheduling, and inventory management in the context of TAC SCM. In addition, we present results of our network visualizer toolbox, where a manager is able to see the current configuration of the network as well as the state of the different nodes. We review the relevant related literature, and finish with conclusions and future work. In the future work section we describe the Dutch flower auction network as an example of a complex, strategic, and uncertain business network on which we are currently working to integrate our architecture and algorithms.

A Business Network Testbed: The Trading Agent Competition for Supply Chain Management

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. The Trading Agent Competition for Supply Chain Management (TAC SCM) is an abstract model of a highly dynamic direct sales (Chopra & Meindl, 2004) environment, as exemplified by Dell Inc.,[1] for procurement, inventory management, production, and sales.

TAC SCM simulates a product life-cycle for a manufacturing organization. In the simulation scenario, each of six competing agents plays the part of a manufacturer of personal computers. Agents compete with each other in a procurement market for computer components, and in an auction-based sales market to sell computers to customers, as shown in Fig. 15.1. The scenario models a market situation where products have limited market life, and the major components used to manufacture those products have little or no residual value at the end of that market life. A typical simulation runs for 220 simulated days over about an hour of real time. Each agent starts with no inventory, an empty bank account, and a finite-capacity production facility. Agents must borrow (and pay interest) to build up inventory of computer components before they can begin assembling and shipping computers.

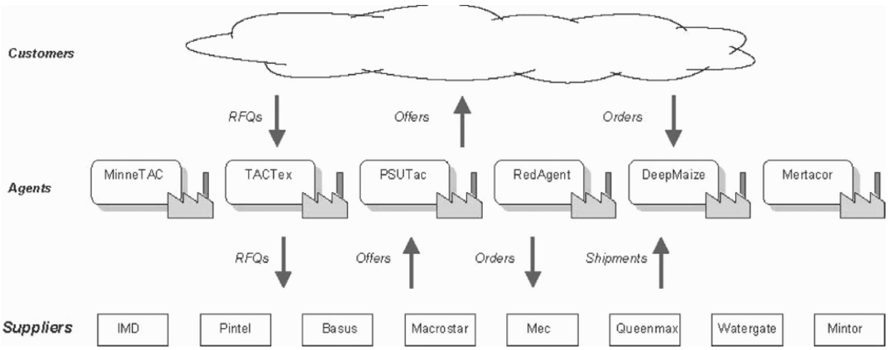

Fig. 15.1 Schematic overview of a typical TAC SCM game scenario. Agents submit daily Request for Quotes (RFQ) to suppliers to buy component parts, and customers request finished computers

Agents have very limited visibility of the actions of other agents, and must deal with significant variability in customer demand, supplier capacities, and other factors. The primary performance criterion is profitability, so the agent with the largest bank account at the end of the simulated year is the winner.

Organized competitions can be an effective way to drive research and improve understanding in complex domains, free of the complexities and risk of operating in open, real-world environments. Artificial economic environments typically abstract certain interesting features of the real world, such as markets, competitors, demand-based prices, and cost of capital, and omit others, such as personalities, taxes, and seasonal demand. Examples related to electronic commerce, besides TAC SCM, include the Penn-Lehman Automated Trading Project (Kearns & Ortiz, 2003), the TAC travel competition (Wellman et al., 2001), and the CAT competition (Niu et al., 2008).

[1] http://www.dell.com

Designing an Intelligent Trading Agent for Dynamic Business Networks

Since the inception of TAC SCM in 2002, more than 50 teams have built agents to play in the competition. These agents represent a variety of approaches to solving the various modeling and decision problems presented by the simulation scenario. We wanted our agent to be a flexible research tool, to enable easy testing of hypotheses and comparison of approaches. We intend to use MinneTAC as a teaching tool, to teach concepts in supply-chain management, economic decision making, machine learning, and software design. To address the twin challenges of simulating a business organization and supporting a research agenda, the design of MinneTAC (Collins et al., 2008) models a flexible organization using a service-oriented approach. There are a few top-level decision elements (Procurement, Manufacturing, Sales) and a large number of services that act as analysis modules, supported by a common database. We call these modules *evaluators*. A high-level schematic representation of this design is shown in Fig.15.2.

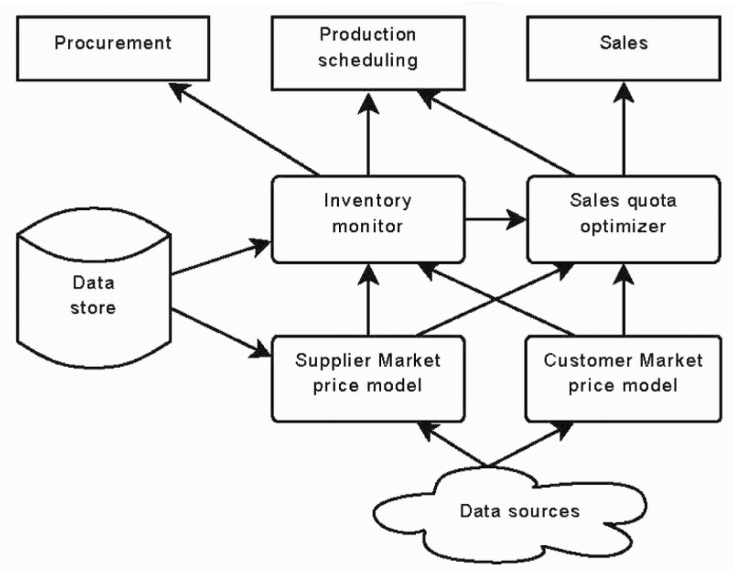

Fig. 15.2 MinneTAC trading agent architecture. Arrows show data flow, not dependencies

Decision components operate by retrieving data from the database, and using evaluation results from evaluators. Evaluators share a common service-oriented design, and they may be composed into chains and feedback loops to perform arbitrarily complex analyses. They may request inputs from other evaluators, from the database, and from external sources. They transform that data in various ways, for example by updating price models, estimating demand trends, or running optimization algorithms to produce sales quotas or procurement recommendations. Results are provided in a common, self-describing format so they can be used by other evaluators or decision components. Connections among decision components and evaluators are entirely configurable and modifiable at runtime; the only real dependency in this design is on the database, and on external data sources such as market data and user inputs. This allows individual researchers to encapsulate modeling and decision problems within the bounds of components and services that have minimal, well-defined interactions among themselves.

In Fig. 15.2, the primary decision components are shown across the top. The Procurement component deals with suppliers, attempting to buy the parts needed by Manufacturing at the lowest possible cost. Manufacturing schedules the production facility with assembly tasks that maximize the expected value of its available inventory and production capacity. Sales sets prices and makes customer offers that are expected to maximize profit, given its available resources. These three decision components are in turn supported by a common data store, and by a large set of evaluators that perform various modeling, analysis, and prediction tasks. These are represented schematically here as the interconnected blocks in the center of the diagram, the "Sales Quota Optimizer," the "Customer Market Price Model," etc. The evaluators, in turn, have access to each other and to various internal and external data sources, primarily in the form of periodic market reports that are issued by the simulation, and a large body of historical data that has been "digested" by machine learning models, such as the "economic regime" model described by Ketter, Collins, Gini, Gupta, & Schrater (2007, 2008).

The radical separation of the MinneTAC agent design into separate decision processes and evaluator services addresses the needs of researchers, who need short learning curves and low risk of interfering with each other. Does it serve the needs of the agent itself, which must effectively coordinate its decisions? The most obvious coordination methods are the "push" approach, in which Procurement tries to keep the factory busy and Sales works to maximize profits on the resulting finished goods, and the "pull" approach, in which Manufacturing and Procurement work to maintain target inventory levels at minimum cost as Sales finds profitable opportunities to sell the available inventory. Another possible approach to the coordination problem is the one used by the RedAgent team at McGill University (Keller, Duguay, & Precup, 2004), in which the primary decision components communicate through internal auction-based markets. The DeepMaize team at Michigan (Kiekintveld, Miller, Jordan, & Wellman, 2006) uses a projected production schedule as the primary coordination structure. Slots in the schedule are filled with products that are expected to return the highest marginal profit at some point in the future. Procurement then works to provide sufficient inventory to run the projected schedule, and sales works to sell what is produced.

In MinneTAC, the database holds a record of all transactions made in the past, as well as inventory data, current customer requests, and supplier offers. The evaluators use this data, along with their own data sources, to produce analyses and recommendations that drive decisions. The version of MinneTAC that ran in the 2007 Trading Agent Competition used a modified "pull" method to coordinate its decisions. It was configured to use current and projected sales quotas over an extended time horizon as the primary coordination mechanism, to drive not only sales, but also production and short-term procurement. Long-term procurement was based on estimates of future customer demand, which is produced by another evaluator, and also used as an input for generating sales quotas.

Evaluators can be composed into arbitrarily complex structures, through a back-chaining process. They do this by requesting the outputs of other Evaluator services in the process of producing their results. Such Evaluation requests are made by name rather than by direct reference, and these names are configurable, either through XML configuration files, or through a user interface. This approach preserves independence among Evaluator services, and makes visible the high-level structure of the agent's decision processes. The result is that complex chains and feedback loops can be constructed from relatively simple services using metadata.

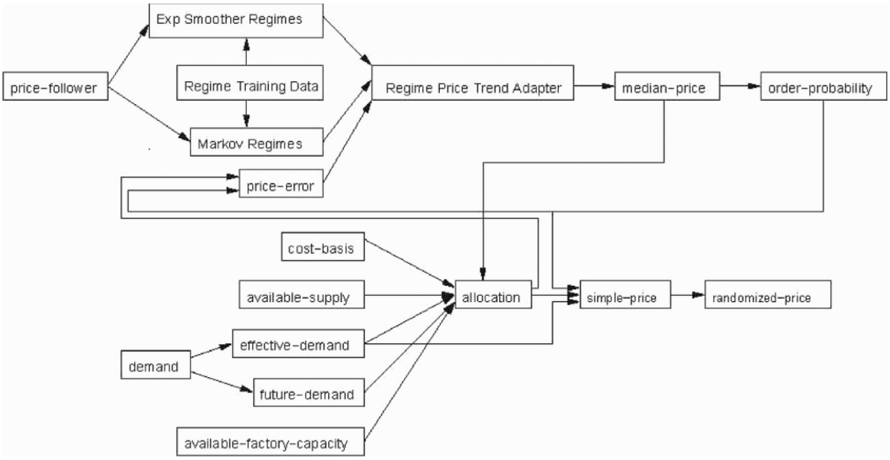

Fig. 15.3 Evaluator chain for a sales manager that uses sales quota and information provided by regimes to determine prices, price trends, and order probability

To illustrate the power of evaluators, in Fig.15.3 we show the evaluation chain that is used to produce sales quotas and set prices in the MinneTAC configuration that ran in the 2007 competition. Each of the cells in this diagram is an Evaluator. Across the top of the diagram is a set of evaluators that estimate current market prices, future price trends, and the shape of the customer order-probability function, based on the method of "economic regimes" developed by Ketter (2001).

We have implemented three different economic regime identification and prediction methods, namely Markov prediction (MP), Markov correction-prediction (MCP), and an exponential smoother lookup (ExpS) process, with the help of evaluators. We also designed a training data evaluator, which is shared by the individual regime evaluators. The training data evaluator uses an external data source that contains an analysis of a large number of past simulations. The analysis was developed using machine learning methods, as described in (Ketter, 2007). These evaluators can dynamically select the most appropriate portions of the training data for a given market situation. In a real business network setting we would train the system on historical transaction data, and update it in regular intervals, e.g. after closing of a set of Dutch flower auctions.

The Sales component used with the evaluator chain shown in Fig.15.3 is conceptually simple – it places bids on each customer RFQ for which the randomized-price evaluator returns a non-zero value. The core of this chain is the allocation evaluator, which composes and solves a linear program each day of the simulation. The problem represents a combined product-mix and resource-allocation problem that computes daily sales quotas that maximize expected profit. The objective function is

$$\Phi = \sum_{d=0}^{h} \sum_{g \in \mathcal{G}} \Phi_{d,g} A_{d,g} \tag{15.1}$$

where Φ is the total profit over a time horizon h, $\mathcal{G}$ is the set of goods or products that can be produced by the agent, $\Phi_{d,g}$ is the (projected) profit for good g on day d, and $A_{d,g}$ is the allocation or "sales quota" for good g on day d. The constraints are given by the evaluators *available-factory-capacity*, the current day's *effective-demand*, projected *future-demand*, and by Repository data, such as existing and projected inventories of parts and finished products, and outstanding customer and supplier orders. Predicted profit per unit for each product type is the difference between *median-price* and *cost-basis* for those products.

Managers need not only to understand and control their decision processes, but also to visualize the data that are being used and produced by the elements of that process. This is very easy to do when decision processes are broken up into a set of discrete, single-purpose services. Figure 15.4 is a screen shot of an early prototype of the user interface.

Figure 15.5 displays the history of daily demand (the output of the "demand" evaluator) along with daily sales quotas (the output of the "allocation" evaluator). This information can be displayed for the overall market, or for individual products or market segments.

Figure 15.6 shows current sales commitments that have not yet been scheduled for production. The MinneTAC design allows a user to dynamically compose such "dashboard" displays by connecting a variety of graphing and plotting widgets to the outputs of the various evaluators. This can be done "on the fly", while the system is running, because the composition of services (Sinn, Hendler, & Parsia, 2002; Wu, Parsia, Sirin, Hendler, & Nau, 2003) and visualizations is entirely dynamic.

Related Literature

This work draws from several fields. In Computer Science, it is related to Software Engineering, Artificial Intelligence, autonomous agents, and multi-agent systems, especially agent architectures, machine learning, and reasoning under uncertainty. In Economics and Information Decision Sciences, it draws from the framework of

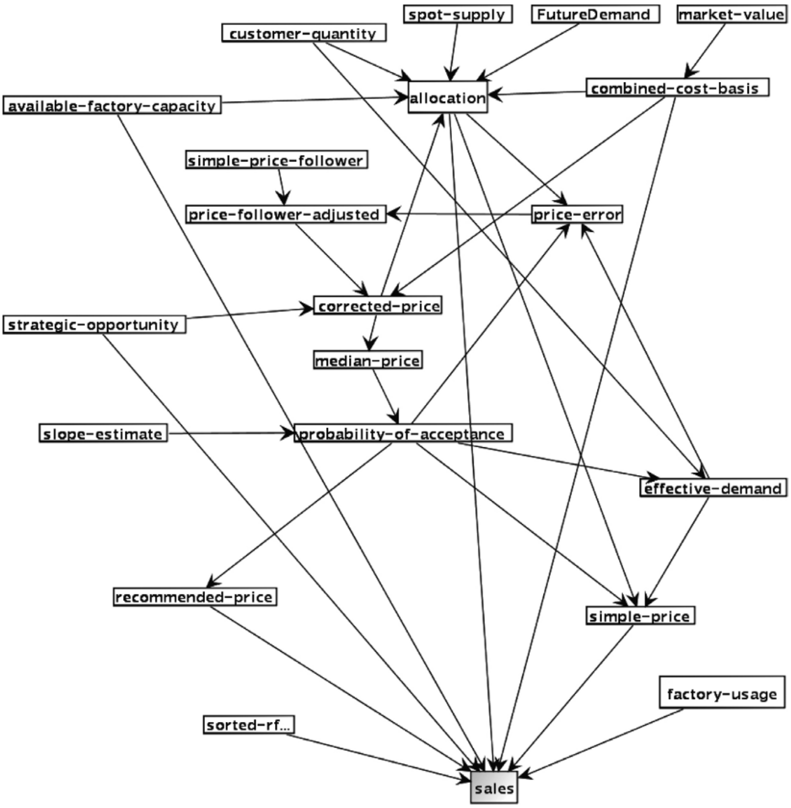

Fig. 15.4 Detail view from evaluator business network visualization tool

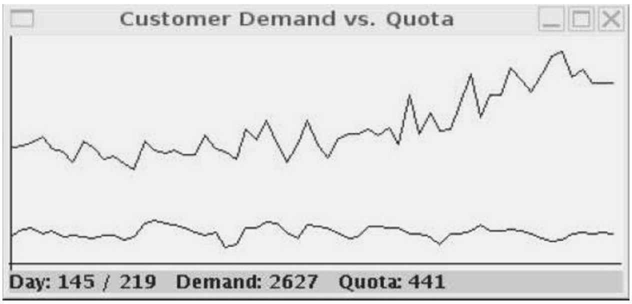

Fig. 15.5 Dynamic network status visualization: daily demand and sales quotas

smart business networks and decision theory. From Operations Research, it draws from work in combinatorial optimization and supply-chain management.

Fig. 15.6 Dynamic network status visualization: outstanding customer orders

Multi-Agent Systems

Most agent design efforts have focused on either the autonomous behavior aspects of agency, or on interactions among agents. Norman, Jennings, Faratin, and Mamdani (1997) describe *agent societies* that model organizational structures and automate business processes. These ADEPT agents negotiate over service agreements that can involve many parties and many dimensions. JADE (Moraitis, Petraki, & Spanoudakis, 2003) is an agent framework that has been used to build trading agents. Its primary emphasis is on building multi-agent systems that comply with FIPA specifications for inter-agent communications, and with flexible deployment in a network environment. These features are not necessary for the TAC SCM domain.

Vetsikas and Selman (2003) describe a method for studying design tradeoffs in a trading agent. This approach could be used effectively in MinneTAC, but the issues addressed by their method are orthogonal to the component/evaluator scheme underlying MinneTAC. Vytelingam, Dash, He, and Jennings (2006) describe the IKB approach to the design of trading agents, consisting of an Information layer, a Knowledge layer, and a Behavioral layer. Podobnik, Petric, and Jezic (2006) have applied this approach to the TAC SCM scenario in CrocodileAgent. The MinneTAC design could be roughly mapped to this scheme, with the database as the Information layer, the set of evaluators as the Knowledge layer, and the decision components as the Behavioral layer.

He, Rogers, Luo, & Jennings (2006) have adopted a design consisting of three internal "agents" to handle Sales, Procurement, and Production/Shipping. Sales decisions use a fuzzy logic module. Some algorithmic aspects are given, but there is little further detail on the architecture of the agent. TacTex05, the winner of the 2005 competition (Pardoe & Stone, 2006) is based on two major modules, a Supply

Manager that handles procurement, and a Demand Manager that handles sales, production, and shipping. These modules are supported by a supplier model, a customer demand model, and a pricing model that estimates sales order probability.

Smart Business Networks

During the mid-nineties Goldman, Nagel, and Preiss (1995) and Sanchez (1995) stressed that in highly dynamic business networks the capability of a quick connect of network actors (businesses) is essential to enable fast response times and greater variety when presented with new product opportunities. The concept of "quick connect" includes a search and select behavior by the different businesses. Goldman et al. (1995) further argue the need for a "quick disconnect" when the business transaction is over, otherwise open network connections can create unwanted information flows that make create unwanted side effects. At the time those articles were published no such network existed. Our architecture offers a unique way of automatically connecting, disconnecting and communicating with the appropriate actors in the network.

One has to pay special attention to the interfaces of the different network actors. Establishing a temporary connection between actors needs to be grounded on a good and matching interface design. This interoperability can be facilitated by modularity. Garud, Kumaraswamy, & Langlois (2002) define modularity as decomposability of a system by grouping elements into a smaller number of subsystems. Modularity is further a very well known concept in the software engineering field, which refers to the extent to which software is divided into components, called modules, which have high internal cohesion,[2] low coupling[3] between each other, and simple interfaces. Our architecture exhibits high cohesion and low coupling.

Hoogeweegen, van Liere, Vervest, van der Meijden, & de Lepper (2006) and van Liere (2007) argue that knowledge of the network structure empowers the decision maker, and leads to better business decisions. With our approach we are able to visualize the network structure, and even drill down on particular network actors to get a detailed picture of specific decision chains. Kambil & Short (1994) already argued in 1994 that there is a strong need to construct software tools for business network representation, visualization, and analysis. These tools can help researchers and managers to visualize the different network actors, or roles, and linkage-based strategies of different organizations enabling the systematic repre-sentation and analysis of changes in emerging organizational forms. Our architecture offers unique capabilities for network visualization, role-and linkage analysis.

Creating performance and information dashboards (Eckerson, 2005) is part of the new emerging field of Business Intelligence (BI) (Shmueli, Patel, & Bruce,

[2] A measure of the extent to which related aspects of a system are kept together in the same module, and unrelated aspects are kept out. High cohesion is better than low cohesion.

[3] A measure of the extent to which interdependencies exist between software modules. Low coupling is better than high coupling.

2006). BI is a very powerful tool, as it provides functionalities such as real-time monitoring, performance reporting, support for exploring solution space with normative models, statistical techniques, and visualization. Business intelligence softwarecan crawl the web, mine data, and come back with a report customized to user preferences. Our architecture fully supports BI and our dashboards are customizable for individual managers. According to Adam and Pomerol (2002) the layout of an economic dashboard has a direct impact on the understanding derived by managers. We believe that our customizable design will facilitate managerial decision making. They argue that a graphical user interface (GUI) of a dashboard should be leveraged to maximize the visual impact of the dashboard.

Furthermore dashboards (a) provide users with functions to find more detailed information of a certain metric or indicator (drill-down capabilities), (b) provide users an interactive way of communicating with different actors (agents) in the network, (c) allow customizing the appearance of how information is delivered and its granularity (days vs. weeks vs. months views), and (d) provide search queries which help agents to learn from a user. A complete and extensive work on the visual design of dashboards has been presented by Few (2006). According to Few many software companies have developed and sold dashboard applications since 2001. That year was characterized by the Enron scandal which increased awareness throughout companies of the importance of monitoring closely their most important business processes. Software companies from all kinds of sizes, such as Microsoft and Oracle, have developed dashboards.[4]

Conclusions and Future Work

Experimental work with multi-agent systems in business networks requires an implementation. Often, the design qualities that best support experimental work are different from those normally considered "ideal" in industry. In complex economic scenarios such as TAC SCM, the desired design qualities include clean separation of infrastructure from decision processes, ease of implementation of multiple decision processes, clean separation of different decision processes from each other, and controllable generation of experimental data. The ability to compose agents with different combinations of decision processes enables testing the effectiveness of the competing decision models.

We have presented one way to construct such an agent, using a readily-available component framework[5] and a facility that allows metadata-driven composition of analysis and modeling tools using evaluators. Additionally we presented tools to visualize the network structure, and economic dashboards to present the current state of each business unit.

There are many possible extensions to the basic design we presented here. One that we are currently pursuing is to add an "executive" component to allocate "resources" to competing implementations of basic decision processes within a

[4] http://www.enterprise-dashboard.com

[5] We used the Apache Excalibur component framework, see http://excalibur.apache.org/.

single agent. This would allow a high degree of adaptability in the game environment, where the level of demand can fluctuate greatly, and where the actions of other agents can have a significant impact on the markets.

As implementation of business intelligence requires a lot of time, money and effort, managers need to know when to consider business intelligence and when not. We implemented our approach in TAC SCM, an abstraction of a real world supply-chain scenario. The next step is to create a web service wrapper around the evaluators, and integrate it in a real business network, such as the Dutch Flower auction (Kambil & van Heck, 1998; Kambil & van Heck, 2002).

We plan to implement automated web services (Sirin et al., 2002; Wu et al., 2003) to better connect to unknown network actors. This will guarantee a smooth run of the network as suggested by (van Hillegersberg, Boeke, & van den Heuvel, 2004).

Acknowledgements

Partial support for Maria Gini is gratefully acknowledged from the National Science Foundation under award NSF/IIS-0414466.

References

Adam, F., and Pomerol, J. Critical Factors in the Development of Executive Systems – Leveraging the Dashboard Approach. *Decision making support Systems: Achievements and challenges for the new decade* (pp. 305–330). PA, USA: IGI Publishing

Berry, P., Conley, K., Gervasio, M., Peintner, B., Uribe, T., Yorke-Smith, N.: Deploying a personalized time management agent. *In Proceedings of the Fifth international conference on autonomous agents and multi-agent systems*. Hakodate, Japan (2006)

Chopra, S., Meindl, P. *Supply Chain Management*. NJ, USA: Pearon Prentice Hall, (2004)

Collins, J., Arunachalam, R., Sadeh, N., Ericsson, J., Finne, N., Janson, S.: *The Supply Chain Management Game for the 2006 Trading Agent Competition* (Tech. Rep. No. CMU-ISRI-05-132) Carnegie Mellon University, Pittsburgh, PA, USA (2005)

Collins, J., Bilot, C., Gini, M., Mobasher, B. Decision processes in agent-based automated contracting. *IEEE Internet Computing*, *5*(2), 61–72 (2001)

Collins, J., Ketter, W., Gini, M. Architectures for Agents in TAC SCM. In AAAI Spring Symposium on Architectures for Intelligent Theory-Based Agents (pp. 7–12). Stanford University, Palo Alto, CA, USA (2008)

CombineNet: *Sourcing solutions*. Retrieved from http://www.combinenet.com/sourcing solutions/ (2006)

Eckerson, W.W. *Performance Dashboards: Measuring, Monitoring, and Managing Your Business*. NY, USA: Wiley (2005)

Few, S. *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Media (2006)

Garud, R., Kumaraswamy, A., Langlois, R. *Managing in the Modular Age: Architectures, Networks, and Organizations*. Oxford, England: Blackwell (2002)

Goldman, S., Nagel, R., Preiss, K. *Agile competitors and virtual organizations*. Van Nostrand Reinhold, NY, USA (1995)

He, M., Rogers, A., Luo, X., Jennings, N. R. Designing a successful trading agent for supply chain management. In Proceedings of the fifth international conference on autonomous agents and multi-agent systems (pp. 159–1166) Hakodate, Japan (2006)

van Heck, E., Vervest, P. Smart business networks: How the network wins. Communications of the ACM, *50*(6), 28–37 (2007). DOI http://doi.acm.org/10.1145/1247001.1247002

van Hillegersberg, J., Boeke, R., van den Heuvel, W. Potential of Webservices to enable smart business networks. Journal of Information Technology, *19*(4), 281–287 (2004)

Hoogeweegen, M., van Liere, D., Vervest, P., van der Meijden, L., de Lepper, I. Strategizing for mass customization by playing the business networking game. Decision Support Systems *42*(3), 1402–1412 (2006)

I2 *Next-generation planning*. Retrieved from http://i2.com/solution library/ng planning.cfm (2006)

Kambil, A., van Heck, E. Reengineering the Dutch Flower Auctions: A Framework for Analyzing Exchange Organizations. *Information Systems Research*, *9*(1), 1–19 (1998)

Kambil, A., van Heck, *E. Making markets: How firms can design and profit from online auctions and exchanges*. Boston, MA, USA: Harvard Business School Press (2002)

Kambil, A., Short, J. Electronic integration and business network redesign: A roles-linkage perspective. *Journal of Management Information Systems*, *10*(4), 59–83 (1994)

Kearns, M., Ortiz, L. The Penn-Lehman Automated Trading Project. IEEE Intelligent Systems *18*(6) 22–31 (2003)

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

Ketter, W. *Identification and prediction of economic regimes to guide decision making in multi-agent marketplaces* (Ph.D. thesis, University of Minnesota, Twin-Cities, USA). (2007)

Ketter, W., Collins, J., Gini, M., Gupta, A., Schrater, P.: A predictive empirical model for pricing and resource allocation decisions. In Proceedings of Ninth international conference on Electronic Commerce (pp. 449–458). Minneapolis, MN, USA (2007)

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

Kiekintveld, C., Miller, J., Jordan, P.R., Wellman, M.P. Controlling a Supply Chain Agent Using Value-Based Decomposition. In Proceedings of Seventh ACM conference on Electronic Commerce (pp. 208–217). Ann Arbor, USA (2006)

Liere, D. *Network Horizon and the Dynamics of Network Positions: A Multi-Method Multi-Level Longitudinal Study of Interfirm Networks*, Ph.d. thesis RSM Erasmus University, Rotterdam, Netherlands) (2007)

Moraitis, P., Petraki, E., Spanoudakis, N.: Engineering JADE agents with the Gaia methodology. In R. Kowalszyk, J. Miller, H. Tianfield, and R. Unland (Eds.), *Lecture Notes in Computer Science*: Vol. 2592, Agent-Mediated Electronic Commerce: Designing Trading Agents and Mechanisms (pp. 77–91). Berlin: Springer (2003)

Niu, J., Cai, K., Parsons, S., Gerding, E., McBurney, P., Moyaux, T., et al. JCAT: A Platform for the TAC Market Design Competition. In Proceedings of the Seventh International Conference on autonomous agents and multi-agent systems (AAMAS 2008). Estoril, Portugal (2008)

Norman, T.J., Jennings, N.R., Faratin, P., Mamdani, E.H.: Designing and implementing a multi-agent architecture for business process management. In M.J. Wooldridge, J.P. Müller, and N.R. Jennings (eds.), *Lecture Notes in Artificial Intelligence.* Vol. 1193. Intelligent agents III (pp. 261–275). Berlin: Springer, (1997)

Pardoe, D., Stone, P.: Tactex-05: A champion supply chain management agent. In Proceedings of the Twenty-First National Conference on Artificial Intelligence (pp. 1389–1394). AAAI, Boston, MA, USA (2006)

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

Sanchez, R. Strategic Flexibility in Product Competition. Strategic Management Journal, 16, 135–159 (1995)

Sandholm, T. Expressive commerce and its application to sourcing: How we conducted $35 billion of generalized combinatorial auctions. AI Magazine, *28*(3), 45–58 (2007)

Shmueli, G., Patel, N., Bruce, P. Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner. NY, USA: Wiley-Interscience (2006)

Sirin, E., Hendler, J., Parsia, B. Semi-automatic composition of web services using semantic descriptions. In Web Services: Modeling, Architecture and Infrastructure workshop in conjunction with 5th International Conference of Enterprise Information System 2003

Vervest, P., van Heck, E., Preiss, K., Pau, L.F. *Smart Business Networks*. Berlin: Springer, (2005)

Vervest, P., Preiss, K., Heck, E., Pau, L. The emergence of smart business networks. *Journal of Information Technology,* 19(4), 228–233 (2004)

Vetsikas, I.A., Selman, B. A principled study of the design tradeoffs for autonomous trading agents. In Proceedings of the Second international Conference on autonomous agents and multi-agent systems. Melbourne, Australia. (2003)

Vytelingam, P., Dash, R.K., He, M., Jennings, N.R.: Trading strategies for markets: A design framework and its application. In H.L. Poutr´e, N.M. Sadeh, and S. Janson (Eds.), *Lecture Noted in Artificial Intelligence*, Vol. 3937 (pp. 171–186) Agent-Mediated Electronic Commerce: Designing Trading Agents and Mechanisms. Berlin: Springer (2006)

Wellman, M.P., Wurman, P.R., O'Malley, K., Bangera, R., Lin, S., Reeves, D., et al. Designing the market game for a trading agent competition. IEEE Internet Computing, *5*(2), 43–51 (2001)

Wu, D., Parsia, B., Sirin, E., Hendler, J., Nau, D.: Automating DAML-S web services composition using SHOP2. In Proceedings of Second International Semantic Web Conference (ISWC2003), Sanibel Island, FL, USA.

Review of “Flexible Decision Support in a Dynamic Business Network”

Don’t Forget the Bounded Rationality of the Agent Designer

Kul Bhushan C. Saxena

Management Development Institute, Gurgaon, India, bsaxena@mdi.ac.in

This paper (Collins, Ketter, & Gini, 2008) addresses the managerial needs for intelligent decision support in a Smart Business Network (SBN) environment and recommends autonomous software agent technology to build such systems. Furthermore, it demonstrates this with an example of a business network test-bed of Trading Agent Competition for Supply Chain Management and an example architecture for such a decision support system (DSS) (called Minne TAC trading agent), which can be used both as a flexible research tool and a teaching tool. The flexibility of this tool is demonstrated by its capability of dynamically connecting and disconnecting various nodes of a SBN, comprising of ‘decision elements’ nodes and ‘evaluator’ (decision modelling services) nodes.

Flexibility as a design criteria has been at the heart of the concept of a DSS right from the days of traditional architectures proposed for a DSS (Saxena & Kaul, 1986; Sprague & Carlson, 1982). These architectures provided decision support flexibility through a model-base comprising of a number of models and the choice of a model was made by the decision-maker by actuating a model management subsystem. Intelligent agent technology embeds intelligence to automatically invoke the required model as deemed fit for the decision environment, and thus frees the system from the bounded rationality constraints of the decision maker. However, the intelligence embedded in most agents is generally limited to structured routine decisions which are largely deterministic rather than judgemental or experiential tacit-knowledge based. From a practical real-world perspective, this may limit the application of this technology to relatively simple and narrow rule-based decision situations, which may not be the case in the contemporary complex business environments where SBN applications may be more appropriate.

Another type of flexibility required in a DSS is in its user interface which needs to be designed differently for a novice versus an expert DSS user as well as for a frequent versus an infrequent user (Saxena & Kaul, 1986). However, the paper does not address this issue.

As for the autonomous nature of software agents, it frees the decision making process from the bounded rationality of decision-maker, but the autonomy of the software also constrains the decision-making process by the bounded rationality of the software agent designer(s)! This can be handled through exception-handling routines providing a 'manual override' disabling the automated decision process. The more complex the decision situation, the more may be the need for such exception handling procedures, unless the software agent has an experiential learning capability.

In spite of these limitations, the proposed architecture demonstrates a goal-oriented service composition in a SBN environment, provides a visualisation tool which may help decision-makers in understanding the active network architecture at any time, and supports building a dashboard to facilitate monitoring of critical business performance parameters. Thus, the proposed DSS tool can be used as a powerful teaching tool supporting action learning, and provides a valuable contribution to software engineering and multi-agent systems technology.

References

Collins, J., Ketter, W., & Gini, M. (2008). Flexible decision support in a dynamic business network. In P. H. M. Vervest, D. W. van Liere, & L. Zheng (Eds.), *The Network Experience – New value from smart business network*. Berlin: Springer.

Saxena, K. B. C., & Kaul, M. (1986). A conceptual architecture for DSS generators. *Information & Management, 10*(3), 149–157.

Sprague, R. H., & Carlson, E. D. (1982). Building effective decision support systems. Englewood Cliffs, NJ: Prentice-Hall