September 2001

Decentralized Economic Coordination in Multi-Agent Systems

Torsten Eymann
Albert-Ludwigs-Universität Freiburg, eymann@iig.uni-freiburg.de

In: Buhl, Hans Ulrich, u.a. (Hg.) 2001. *Information Age Economy*; 5. Internationale Tagung Wirtschaftsinformatik 2001. Heidelberg: Physica-Verlag

ISBN: 3-7908-1427-X

Decentralized Economic Coordination in Multi-Agent Systems

Torsten Eymann
Albert-Ludwigs-Universität Freiburg

Abstract: This article envisions pro-active software components on personalized smart devices with wireless communication, e.g. PDAs, smart clothing or embedded automobile devices that move with the user. In changing electronic commerce environments, these agents might be able to buy and sell autonomously on behalf of their human owner. From a system perspective, a decentralized and continuously changing multi-agent system is created with the need for coordination of supply and demand. In this article we show how such a multi-agent system may be decentrally coordinated using a bottom-up MAS approach, where software agents bargain with each other under the constraints of incomplete information, non-equilibrium and time pressure. It can be shown that the multi-agent system as a whole shows emergent coordination in absence of a centralized coordination institution.

Keywords: Automated Negotiation, Agent-Mediated Electronic Commerce, Digital Business Agents, Market Coordination, Multi-Agent Systems

1 Using Software Agents on Smart Devices

Tomorrow's software agents might be the main software component of personalized smart devices with wireless communication capabilities. Such smart devices in the context of "ubiquitous computing" [Weis91] or "pervasive computing" are envisioned to embed the Internet in mobile phones, smart sensors, smart clothes, building materials, car electronics and so on [EsGo00]. These devices are mostly considered to passively collect information from the environment or centralized web servers and display this information to the human user. But there is just a small step to have them actively pursuing a goal, according to given preferences and an adaptive strategy. These autonomous devices can be either stationary, such as a refrigerator which buys food [Kahn99], or a water heater which buys electricity [Ygge98]. They can also be mobile, such as a PDA-based shopping agent [YoMo00], or a car-based electronic toll payment device [DeAr00].

Instead of constant interaction between human owners of such devices, responsibility to conduct these economic transactions may be handed over to software agents: software that acts autonomously in some environment to fulfill its design goals [Wool99]. These agents will monitor other agents and the environment continuously, watching for potential opportunities to fulfill their design goals. They will be able to enter into negotiation with many potential trade partners at once, reaching an acceptable deal and setting up a contract in a matter of milliseconds [GuMa98; Prei98]. In the remainder of this article, we focus on negotiating software agents in business environments, regardless of them being Internet- or PDA-based, which we call "Digital Business Agents" (DBAs). DBAs will assist human buyers and sellers in digital business processes and environments, which are characterized by trading and money transactions.

The research question pursued in this article is how different adaptive negotiation strategies of DBAs compete against each other in such an unregulated environment, and the impact on the coordination of the environment as a whole. In the next chapter, we describe the technical requirements and the setup of a multi-agent system (MAS) that coordinates a model supply chain process using decentralized economic coordination. Some experiments with different bargaining strategies in the DBAs are described in chapter 3. The article concludes in chapter 4 with an outlook into possible application areas and the problems of linking economic concepts and technical realization of market coordination.

2 An Experimental Market Coordination Environment

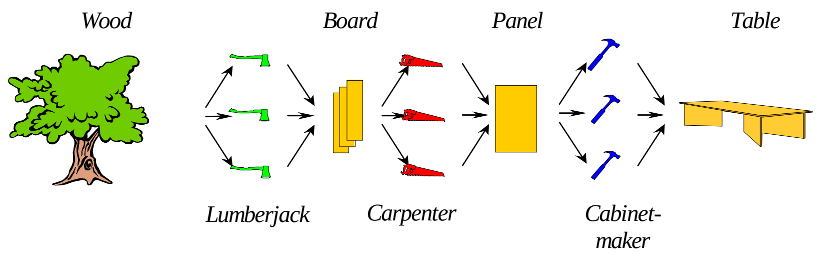

Figure 1: A simple supply chain scenario.

In order to evaluate the future impact and functionality of agent-driven coordination issues, a prototypical market-coordinated MAS named AVALANCHE has been designed. Here, DBAs represent human users intending to buy, produce and sell different goods or services as "miniature automated businesses" [KeHa00]. A simple and arbitrary supply chain is modeled in which three different types of DBAs produce a consumer good: “lumberjacks”, that buy trees to produce and sell boards; “carpenters”, that buy the boards and fix them together to sell as a panel; and “cabinetmakers”, that buy the panels, build tables out of them and sell them. Two additional types, producers and consumers, which only define a seller or buyer strategy, respectively, provide the tight ends of the supply chain. This scenario serves as a blueprint for a specific class of typical information system application environments – it is not considered to be a realistic model of a furniture-making supply chain.

2.1 The Technical Setup

AVALANCHE is realized in JAVA 1.2 and uses the ORB class library VOYAGER 3.0 [Obje99]. All agents are independent JAVA threads. The system architecture consists of three basic classes, the marketplaces, trader agents, and one experiment control object. Both marketplaces and trader agents are initialized simultaneously in our experiments, but due to the openness of the architecture, trader agents could enter marketplaces at any point in time. Detailed information about AVALANCHE can be found in [Eyma00].

The *agent class* defines communication abilities and negotiation protocols (in the remainder of this paper, we use the terms “agent”, “software agent” and “DBA” synonymously). Each agent communicates with every other object on the marketplace in direct, bilateral and unmediated fashion. All agents are based on standardized concepts of JAVA, and distinguishable by a unique identity.

The *electronic marketplaces* are based on Java Virtual Machines (JVM), which run as a server process in a TCP/IP-based network. In order not to interfere with the negotiation process of the software agents, the functions and services of the object are kept to a minimum. The marketplace neither actively influences the software agents’ strategies nor the communication – in particular, it does not explicitly synchronize or schedule the agents’ activities. It only offers a single “white pages” directory service for any software agent to register as seller or buyer, and to inquire about other registered agents. Price bids are never publicly posted (as in catalogs), since the price information might partly reveal the agent’s strategy to competitors and bidders, and prevents the negotiation of individual prices per bidder.

The *experiment control object* contains technical functions for logging the transaction data, which is not expected to have any effect on the market coordination.

2.2 The Negotiation Protocol

The key difference of AVALANCHE compared to many other agent-based marketplace projects is the absence of a central arbitrator or auctioneer, and the implementation of a strictly decentralized automated negotiation (bargaining) protocol. Comparable automated negotiation efforts in multi-agent systems can be found in the research context of agent-mediated electronic commerce [GuMa98; Sier00] and market-oriented programming [Well96].

All agents in AVALANCHE follow the economic goal of profit maximization. They try to buy input goods for less and to sell output goods for more. Depending on the actual market situation, its equity, and stock, the agent decides autonomously which action to take next - whether to buy, sell, produce, move or self-terminate. The agent lifecycle “follows the money”: if the agent has finished goods in stock, it tries to sell. If the agent has no goods, but input factors in stock, it simulates the production of output goods (by waiting an appropriate length of time). If the agent has no input factors in stock, it tries to buy some. If the market situation is not satisfying at all, e.g. if there are no offers or demands within a certain time span, the agent tries to move to another marketplace. If the agent has spent its entire budget or all marketplaces are shut down, it has to terminate – every few milliseconds the agent has to pay utilization fees to the market anyway, so doing nothing is never a dominating strategy.

In the case of buying or selling, the DBA goes through the three stages of a market transaction: information, agreement, and settlement [ScLi98]. In the information phase of any transaction, a buyer or seller has to identify its potential trading partners. If a seller agent wants to sell its output, it currently advertises its offer in the directory, and waits until a prospective buyer agent demands a negotiation. Simultaneously, the agent will actively search in the directory for potential transaction partners. Buyer agents apply a mirror procedure.

The buyer agent then initiates the agreement phase by communicating with any supplier from the list. The software agents negotiate using a monotonic concession protocol [RoZl94], where propose and counter-propose messages with subsequent price concessions are exchanged. If the negotiation process is successful, both agents will reach a compromise price agreement; otherwise, someone will sooner or later drop out of the negotiation. In this event, the agents will restart with other partners obtained from the directory.

In the final settlement phase, the transaction is carried out and monitored. Sellers and buyers exchange goods and money, respectively. If any party misbehaves in this phase (by not keeping commitments), the resulting reputation information can be used to modify strategies and behavior to be used in future encounters with that agent as in [PaSa01], effectively repulsing those agents from the market community.

2.3 The Agents' Negotiation Strategy

The goal of the agents, and the reason to engage in negotiation at all, is to maximize the spread between materials cost and sales revenue. The agent thus follows a certain negotiation strategy, derived from human negotiation behavior, which uses parameters such as *demand level*, *concession*, and *concession rate*:

> A bargainer's *demand level* can be thought of as the level of benefit to the self associated with the current offer or demand. A *concession* is a change of offer in the supposed direction of the other party's interests that reduces the level of benefit sought. *Concession rate* is the speed at which demand level declines over time. […] These definitions […are] unproblematic, when only one issue or underlying value is being considered, as in a simple wage or price negotiation. [Prui81, p. 19]

The AVALANCHE agents use an adaptive heuristic strategy based on a non-deterministic finite state automaton (non-det FSM), in which action paths are taken depending on stochastic probes against certain internal parameters. A non-det FSM is equivalent to a complex deterministic rulework [HoUl94], but easier to model because it has less parameters. In AVALANCHE, a combination of six parameters, collectively called the *Genotype*, describes the strategy in the non-det FSM (see below). When an opponent's offer is received by any agent, the following process is carried out.

The goods that are traded are defined as commodities, so the only negotiation variable is the price. The opponent's price will be extracted and put in relation to the subjective market price level, stored in the variable *Memory*. If the opponent's price is not in the deal range (e.g. below twice the *Memory* value), the offer will be rejected as excessive. Anyway, the offer price will modify *Memory* by using a weighted exponential average with weight *w_memory (w_mem)*. The *Satisfaction* probability parameter (*p_sat*) is probed next: it determines if an agent will drop out from an ongoing negotiation instead of making a counter-proposal. Effectively, this parameter creates time pressure - the more steps the negotiation takes, or the more excessive the opponent's offers are, the more likely it is that the negotiation will be discontinued.

If the opponent's offer passes this test, the agent decides to make a counter-proposal. In the next step, the agent has to decide whether to make a concession, and if so, what the new offer price should be. Whether an agent concedes in an actual situation is subject to a stochastic probe against the reciprocal value of the concession making probability, called *Acquisitiveness (p_acq)*. The lower the acquisitiveness value, the higher the average concession rate.

If the agent concedes, the *In-negotiation delta price change (del_change)* parameter calculates the amount of the price concession between two negotiation steps. Both partners calculate a percentage from the price difference of their original offers. If the buyer has *del_change* = 0.5 and the supplier *del_change* = 0.0, the agents will reach agreement after two negotiation steps at the initial price of the supplier, if no agent drops out. In the next step, the agent sends back his new offer and waits for a counter-proposal. This routine will alternatively be carried out by the opponents, until compromise is reached or one of the agents leaves the negotiation process.

During the subsequent settlement phase, *Reputation (p_rep)* affects the agent’s probability of behaving cooperatively by exchanging the committed amount of goods or money. In this article, all agents are assumed to cooperate (*p_rep* = 1); for other cases, see [PaSa01]. Finally, after each successful negotiation, the agents will try to change their initial demand level in their favor, lowering or raising it depending on their role as seller or buyer. The percentual value of that change is encoded in *Pre-negotiation delta price change (del_jump)*.

By varying the setting of the *Genotype* parameters in subsequent experimental runs, we can show how dominant parameter combinations either succeed or are counterbalanced. Heterogeneity is achieved by a uniform random distribution of the initial strategy parameter values, which effectively assigns a different strategy to each agent. The "fitness" denominator for the individual agent's strategy is obviously currency: an agent, which is faster and/or fitter in trading than another, will have a relatively higher income.

3 Experimental Results

To test the behavior of the AVALANCHE system we conduct several test series. Every series consists of several experiment runs, which under similar conditions lead to comparable outcome patterns. In the current implementation we run a generator program, an experimental control object, a single marketplace and 50 trading agents of each type (for a total of 250 agents) in parallel on a Pentium PC under Windows NT for about 10 minutes. The following presentations all show similar patterns when repeatedly started with the same initial values. However, absolute price levels or points in time have no meaning since they change in every run.

3.1 Markets without Intelligence

We start with some simple parameter combinations and subsequently move on to more complex experiments. The first experiment shows homogenous strategies, which means that all agents are equipped with an identical *Genotype* and there is no variation of parameter values in the population. The agents hold no stock, but some initial equity money. Every 10 milliseconds a new “wood” is produced by any producer agent. The consumer agents never run out of money. Initial goods valuations are preprogrammed equally for every agent type (e.g. 25 money units for boards).

Genotype	P_acq	del_change	del_jump	p_sat	w_mem	p_rep
Value	0.5	0.25	0.15	0.75	0.2	1.0
Variance	0.0	0.0	0.0	0.0	0.0	0.0

Table 1: Parameter settings for introductory experiment

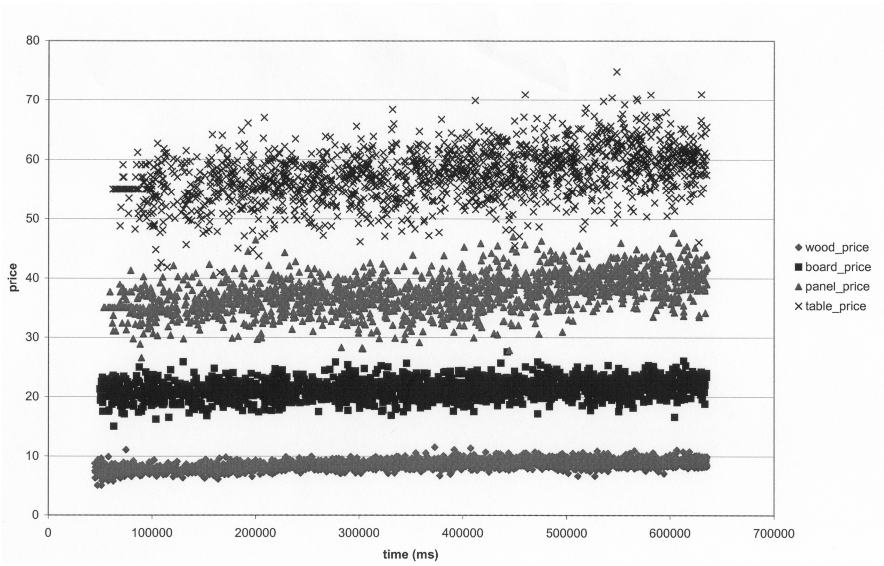

Figure 2: Price dynamics using homogeneous strategies

The presentation charts are explained using the simple picture of Figure 2. The horizontal axis measures the time in milliseconds, the vertical axis shows the price level. Every dot in the chart points out time and agreement price of a transaction, e.g. the x-value of a triangle marks the time when some carpenter software agent has sold a table to a cabinetmaker software agent for y-value price. The type of goods is represented by different gray scales and symbols as indicated by the legend, which shows the supply chain from top to bottom. In the chart, this succession has been turned upside down since woods are sold for the lowest price and tables for the highest. To make the graphic comparable there is a price ceiling for tables at 120 money units, which can be seen in later experiments. The time lag at the beginning is caused by software initialization and initial negotiation until the first goods are moved up the supply chain.

In Figure 2 it is not possible for a single agent to gain an advantage, since all agents possess an identical static negotiation strategy. The price levels do not substantially change during the course of the experiment. The growing variance of the goods prices with the supply chain steps seems to be a variant of the bullwhip effect, common in supply chain management [LePa97]. In the next experiment, the carpenter agents are initialized with a "greedier" negotiation strategy than all other types.

Genotype	P_acq	del_change	del_jump	p_sat	w_mem	p_rep
Value	0.6 for carpenters 0.5 for other types	0.25	0.15	0.75	0.2	1.0
Variance	0.0	0.0	0.0	0.0	0.0	0.0

Table 2: Parameter settings for greedy carpenters

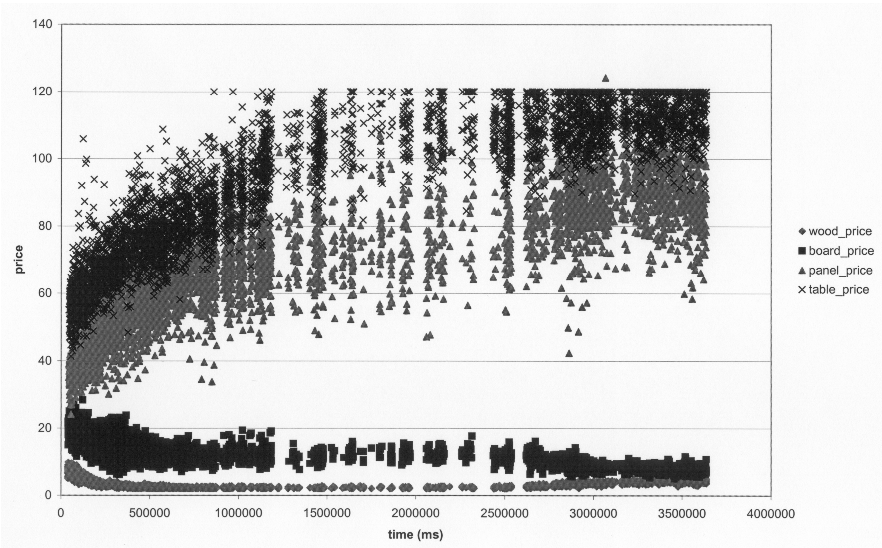

Figure 3: Different concession rates lead to different profits[1]

The greediness of only one agent type (carpenters) leads to a picture of winners and losers (see Figure 3). Since the other agent types concede faster than the carpenters do, the latter are able to increase their income (the spread between board prices and panel prices) at the expense of both lumberjacks and cabinetmak-

1 The vertical white stripes in the figure are experimental artifacts caused by the Windows NT environment and have no effect on the outcome.

ers. It appears to be a dominating strategy to set the agent's acquisitiveness as high as possible, at least higher than any other agent it might trade with, and in fact this is where the gains of the carpenters come from. In an environment without evolutionary learning and open markets, the higher acquisitiveness of the carpenters is a dominating strategy. However, most of these experiments abort early, because the lumberjacks and cabinetmakers run out of money and are thus not able to continue the supply chain – in this case, the victory was short-lived.

3.2 Introducing Evolutionary Learning in the Market

The negotiation profit gained from a specific *Genotype* is correlated to the relative setting of other agent's strategies, not the absolute values. "Learning" the right values relative to the other agents is thus essential to gain a higher profit in the future. In Figure 3, the inability of the agents to change their strategy affects both winners and losers – the other agents should raise their own *acquisitiveness* setting, while the carpenters could lower their setting to conduct more transactions within the same time.

If we move towards a more realistic and heterogeneous setting, we can show how the application of a feedback learning mechanism can emergently control the system. Our approach uses a decentralized evolutionary algorithm taken from [SmTa98], which is based on gossip information about the success of transactions.

In AVALANCHE, every agent sends a "plumage" object after a successful transaction, advertising its average income and its genes to one random agent of the same type. This can be interpreted as someone receiving gossip or insider information about an otherwise private deal information of someone else. Upon having received a fixed number of plumages, every agent ranks them and selects the plumage with the highest average income attached. This plumage will then be crossed over with the agent's own strategy genes, and "planted" into a new agent which then enters the marketplace. That way the evolutionary algorithm relegates unsuccessful agents/strategies and promotes successful agents/strategies, changing the composition of the population over time.

Figure 3 has shown how heterogeneous strategies of the agents affected the overall behavior of the system, in case there is no evolution. In contrast, Figure 4 shows the results of an experiment where economic effects are combined with EC-like interactions between agents. The carpenters were, as before, equipped with a lesser probability to make price concessions, and this is again evident in the first half of the experiment run: the spread between the second (boards) and third (panels) price curve increases. But this time, the lumberjack and cabinetmaker agents are (after some time) not only able to withstand the pressure, but to turn the tables and to dominate the carpenter agents. This is graphically displayed in Figure 5, where every curve represents the average *acquisitiveness* values of a specific agent type over time. The highest curve at the beginning represents the carpenters, and one can see that their initially high average of the *acquisitiveness* parameter soon declines too much.

After the first half of the experiment, the carpenters end up with the lowest setting of all three inner agent types. At the same time, the lumberjack and cabinetmaker agents are able to co-evolve and raise their *acquisitiveness* setting. At around 1500 seconds in the earlier Figure 4, the spread between board and panel prices had finally vanished and with it the profits of the carpenter agents. In Figure 5 at that time, the average *acquisitiveness* setting of the carpenter agents reaches the lowest value compared to the other agent types.

Genotype	P_acq	del_change	del_jump	p_sat	w_mem	p_rep
Value	0.6 for carpenters 0.5 for other types	0.25	0.15	0.75	0.2	1.0
Variance	**0.5**	0.0	0.0	0.0	0.0	0.0

Table 3: Parameter settings for co-evolutionary greediness

This picture changes again in the second half of both charts, when the cabinetmaker agents learn and raise their *acquisitiveness* again. Similar to the beginning, now the cabinetmaker, and to a lesser extent, the lumberjack agents reduce their *acquisitiveness* values as shown in Figure 5. The result can be immediately seen in Figure 4: the carpenter agents gain ground, the lumberjack agents are able to withstand the pressure and keep their profit situation, and the cabinetmaker agents lose everything gained in the first half of the experiment run. In summary, a highly dynamic pattern of co-evolution arises, where learning successful parameter settings allows other participants to survive and prosper early while dominating strategies are counterbalanced. Because the size and strategies of the population constantly change, this result will probably never reach a static endpoint, like an evolutionary function optimizer does.

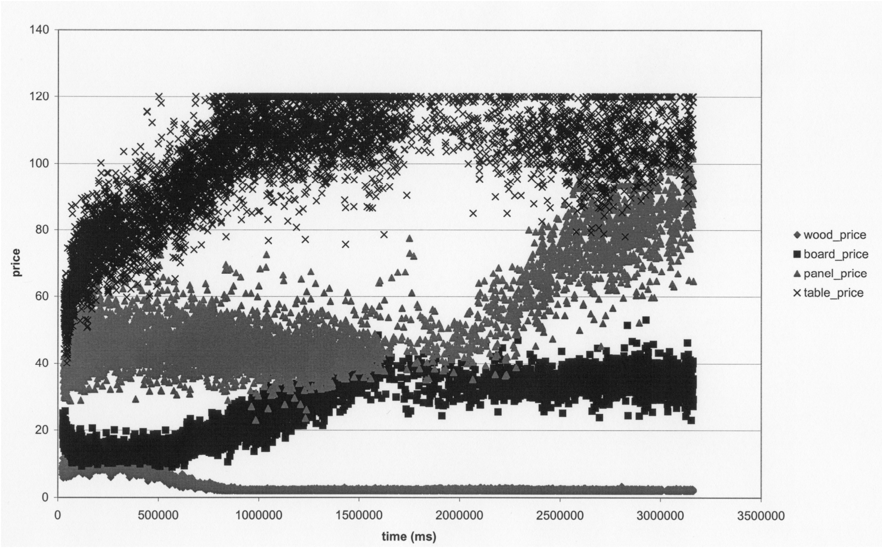

Figure 4: Price Level Development with Evolution.

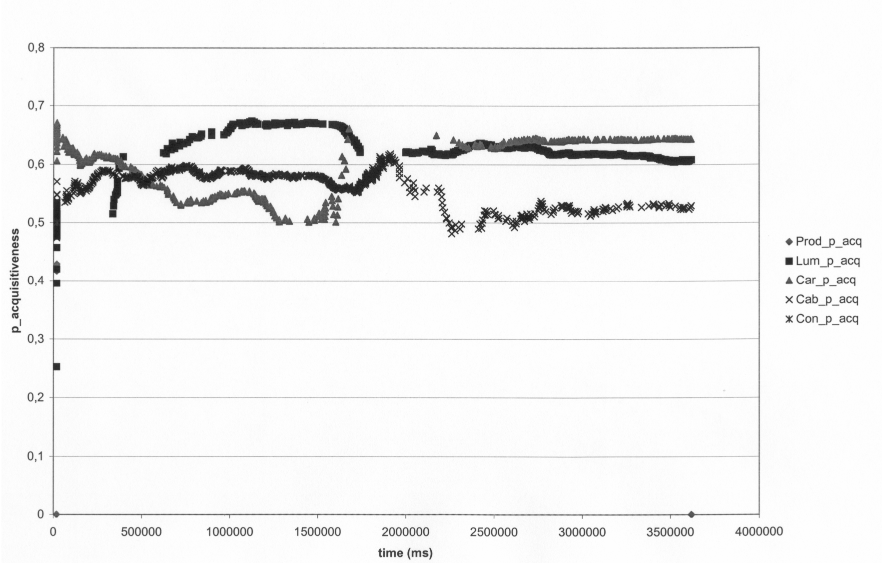

Figure 5: Evolutionary development of the acquisitiveness parameter

4 Market Coordination between Economic Concepts and Technical Realization

Most future visions of ubiquitous computing are mainly hardware-centric and focus on humans interacting through otherwise non-intelligent mobile devices. But to reduce transaction costs, human users will rely on pro-active, device-based software agents to conduct a majority of economic transactions in the background, without human intervention (cf. “silent commerce” [Sche00]). In a electronic commerce context, device-based comparison shoppers allow automated price comparisons between different suppliers [YoMo00]. As counter-strategy, sellers are expected to begin to dynamically post prices to negotiate with individual buyers [KeHa00]. As a result, a complex, decentralized network of selfish trading software agents emerges, which needs coordination to prevent chaos.

This article introduced a coordination mechanism for systems of autonomous decentralized devices, based on constant negotiation and price signaling. Related work can be found in both agent technology and economics, notably agent-based computational economics [Tesf97], to develop new technical possibilities of coordinating decentralized information systems consisting of autonomous software agents. This article shows, how a self-regulating market approach exhibits certain coordination patterns and leads to a co-evolution of strategies. If the agents are able to learn and revise their strategies, a stabilization of price curves can be achieved. As a result, adaptivity of strategies is regarded as a key requirement for software entities in the context of ubiquitous computing, especially in e-commerce scenarios – in a broader sense, adaptive strategies are required for all multi-agent systems consisting of selfish, economic-goal driven agents.

The findings of this article are only the beginning and clearly need further discussion, especially concerning the link between economic understanding and technological realization of market coordination. Some research topics still remain unsolved. It is desired to design mechanisms for self-interested agents such that if agents maximize their local utility, the overall system behavior will at least be acceptable [Binm92]. Among the problems which such a complex system faces are the over-usage of a shared resource known as “tragedy of the commons” [Hard68], chaotic behavior [HuHo88; ThSy98], and the appearance of malevolent agents, which may be countered by the embodiment of centralized and decentralized reputation mechanisms in the system [PaSa01]. These problems have to be watched closely, because they will determine appearance and acceptance of decentralized systems consisting of autonomous devices.

However, a successful implementation of the distributed resource allocation mechanism introduced here has the advantage of a flexible structure and inherent parallel processing. From an application viewpoint, an appealing aspect lies in the inherently distributed structure, which mirrors the division of work in companies and the network of businesses and markets. By using the same coordination rules that govern the real world, decentralized market mechanisms may allow the creation of innovative fast, flexible, parallel working, and self-regulating IT systems.

References

[Binm92] Binmore, Ken: Fun and games - a text on game theory. D.C. Heath, Lexington, Mass 1992.

[DeAr00] Dean, Bob; Arnold, Dave; Harrop, Peter: Smart Cards in Transport. IDTechEx Ltd., Cambridge 2000.

[EsGo00] Estrin, Deborah; Govindan, Ramesh; Heidemann, John: Embedding the Internet. In: Communications of the ACM 43 (2000) 5, p. 39-41.

[Eyma00] Eymann, Torsten: Avalanche - ein agentenbasierter dezentraler Koordinationsmechanismus für elektronische Märkte. Ph.D. Thesis. Albert-Ludwigs-Universität Freiburg, 2000.

[GuMa98] Guttman, Robert; Maes, Pattie: Agent-mediated Integrative Negotiation for Retail Electronic Commerce. In: Proceedings of the Workshop on Agent Mediated Electronic Trading (AMET'98). Minneapolis, Minnesota 1998.

[Hard68] Hardin, Garrett: The Tragedy of the Commons. In: Science 162(1968) 1243-1248.

[HoUl94] Hopcroft, John E.; Ullman, Jeffrey D.: Introduction to automata theory, languages, and computation. Addison-Wesley, Reading, Mass 1979.

[HuHo88] Huberman, Bernardo A.; Hogg, Tad: The behavior of computational ecologies. In: Huberman, Bernardo A.(ed.): The Ecology of Computation.Elsevier (North Holland), Amsterdam 1988.

[Kahn99] Kahney, Leander: The coolest Internet appliance. WIRED Technology News. http://www.wired.com/news/technology/0,1282,17894,00.html, 1999-2-12, access date 2001-3-15.

[KeHa00] Kephart, Jeffrey O.; Hanson, James E.; Greenwald, Amy R.: Dynamic Pricing by Software Agents. IBM Research. http://www.research.ibm.com/infoecon/paps/html/rudin/rudin.html, 2001, access date

[LePa97] Lee, H. L.; Padmanabhan, V.; Whang, S.: Information Distortion in a Supply Chain: The Bullwhip Effect. In: Management Science 43 (1997) 4, p. 546-558.

[Obje99] Objectspace, Inc.: Voyager Documentation. Objectspace Website. http://www.objectspace.com/products/voyager/, 1999, access date 2000-3-1.

[PaSa01] Padovan, Boris; Sackmann, Stefan; Eymann, Torsten; Pippow, Ingo: A Prototype for an Agent-based Secure Electronic Marketplace including Reputation Tracking Mechanisms. In: Proceedings of the 34th Hawaiian International Conference on Systems Sciences. IEEE Computer Society, Outrigger Wailea Resort, Maui 2001.

[Prei98] Preist, Chris: Economic Agents for Automated Trading. Hewlett Packard Laboratories, Bristol 1998.

[Prui81] Pruitt, Dean G.: Negotiation behavior. Academic Press, New York 1981.

[RoZl94] Rosenschein, Jeffrey S.; Zlotkin, Gilad: Rules of encounter - designing conventions for automated negotiation among computers. MIT Press, Cambridge, Mass 1994.

[Sche00] Schenker, Jennifer L.: The Trillion-Dollar Secret. TIME Magazine. http://www.time.com/time/europe/magazine/2000/228/trading.html, 2000-2-28, access date 2001-3-13.

[ScLi98] Schmid, Beat; Lindemann, Markus: Elements of a Reference Model for Electronic Markets. In: Proceedings of the 31st Hawaiian International Conference on Systems Sciences. IEEE Press, Los Alamitos, CA 1998.

[Sier00] Sierra, Carles: A roadmap for Agent-Mediated Electronic Commerce. In: Sierra, Carles and Dignum, Frank(ed.): Agent Mediated Electronic Commerce - The European AgentLink Perspective.Springer, Heidelberg 2001.

[SmTa98] Smith, Robert E.; Taylor, Nick: A Framework for Evolutionary Computation in Agent-Based Systems. In: Langton, Christopher, Taylor, C., Farmer, Jose D., and Rasmussen, S.(ed.): Articial Life II.Addison-Wesley, Reading, Mass. 1998.

[Tesf97] Tesfatsion, Leigh: How economists can get alife. In: Arthur, W. Brian, Durlauf, S., and Lane, D.(ed.): The Economy as a Evolving Complex System II.Addison Wesley, Redwood City, CA 1997.

[ThSy98] Thomas, J.; Sycara, Katia: Stability and heterogeneity in multi agent systems. In: Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS-98). Paris, France 1998.

[Weis91] Weiser, Mark: The Computer for the Twenty-First Century. In: Scientific American 265 (1991) 3, p. 94-104.

[Well96] Wellman, Michael P.: Market-Oriented Programming: Some Early Lessons. In: Clearwater, Scott H.(ed.): Market-Based Control: A Paradigm for Distributed Resource Allocation.World Scientific, Singapore 1996.

[Wool99] Wooldridge, Michael J.: Intelligent Agents. In: Weiss, Gerhard (ed.): Multiagent Systems.MIT Press, 1999.

[Ygge98] Ygge, Fredrik: Market-Oriented Programming and its Application to Power Load Management. Ph.D. Thesis. Lund University, Sweden, 1998.

[YoMo00] Youll, Jim; Morris, Joan; Krikorian, Raffi; Maes, Pattie: Impulse: Location-based Agent Assistance. In: Proceedings of the Fourth International Conference on Autonomous Agents (Agents 2000). Barcelona 2000.