From DPS to MAS to …: Continuing the Trends

Michael N. Huhns
University of South Carolina
Department of Computer Science and Engineering
Columbia, SC 29208 USA
+1-803-777-5921
huhns@sc.edu

ABSTRACT

The most important and interesting of the computing challenges we are facing are those that involve the problems and opportunities afforded by massive decentralization and disintermediation. The problems and opportunities arise in domains where controlled action is necessary, but centralized control is infeasible. These are the traditional domains of distributed problem solving and multiagent systems, and they include information systems for healthcare, commerce, energy distribution, and traffic control. However, the current incarnations of these domains are scaled well beyond anything envisioned originally. Nevertheless, traditional techniques derived from artificial intelligence are still mostly appropriate. Specifically, representation, reasoning, learning, planning, and situated semantics—when distributed computationally and extended to multiple loci of intelligence—will all be part of potential solutions. They will affect not only the ways systems will be implemented and executed in the future, but also the ways they will be designed, developed, and deployed. This paper focuses on the domains and their challenges. It describes the trends that are observable in our research technologies and shows how they can be used to confront the challenges. It is hoped that new avenues of research will be revealed from following the trends.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence - *Coherence and Coordination*, *Multiagent Systems.*

General Terms

Algorithms, Management, Design, Reliability.

Keywords

MAS application description; distributed and social system inspiration; cross-cutting focus.

1. INTRODUCTION

People and their institutions are becoming increasingly specialized and increasingly interdependent, and such trends are occurring on a global scale. The reason the trends are able to occur is because of advances in computing and telecommunications: computing and communicating technologies enable more complex societies to operate with sufficient efficiency to survive. In fact, computers are the only current way to manage the complexity, and in some areas provide almost complete automation. For example, agriculture and manufacturing, with few assembly line workers, have become largely automated.

In other areas, computers enable people to make their own calls, arrange their own travel reservations, shop at retail outlets around the world, and prepare their own tax returns. However, the increasing complexity of society, with its concomitant massive decentralization and disintermediation, leads to problems, vulnerabilities, and opportunities [3], as well as interesting computing challenges. The problems and opportunities arise in domains where controlled action is necessary, but centralized control is infeasible. These are the traditional and emerging domains of distributed problem solving and multiagent systems, and they include

- Healthcare IT for patients
- Grocery shopping IT for consumers
- Re-architected IT systems for the U.S. Navy
- Individualized traffic control
- Public finances
- IT for jurisprudence
- Next-generation supply chains and supply networks
- RFID for tracking goods
- Consumer control of electricity distribution
- CO_2 monitoring and sequestering
- Microclimate monitoring and assessment
- Edge control of telecom bandwidth
- Consensus knowledge and decisions
- Consensus software

The computing challenges encompass all facets of achieving control globally within the massive scale of world-wide infrastructures, while the agents effecting the control are limited to acting locally.

2. FOUNDATION AND HISTORY

The history of our field encompasses distributed problem solving (DPS), distributed artificial intelligence (DAI), multiagent

Cite as: From DPS to MAS to …: Continuing the Trends, Michael N. Huhns, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 43 – 48

systems (MAS), and service-oriented computing (SOC). Each is centered on a key principle, as follows:

DPS: given a distributed domain and a goal, determine how to distribute tasks across computing nodes

- Decomposition is the key metaphor

DAI: make the nodes smart and cooperative or competitive, i.e., *aware* of each other. DAI emphasizes the run-time autonomy of the nodes and local decision-making by the nodes (intelligence)

- Coordination is the key metaphor

MAS: design the nodes independently, give them their own goals, and design the environment. MAS emphasizes the design-time autonomy of the nodes and the importance of the *environment* in which the nodes act, which itself must often be designed

- Interaction is the key metaphor

SOC: design a node so that it can be part of many applications, both internal and external to an organization

- Encapsulated functionality with a public interface is the key metaphor.

3. SOLUTION THREADS

Three prominent threads have emerged recently within the research being conducted in multiagent systems and presented at AAMAS, as follows:

1. Agents that represent individual preferences and that can interact with very large numbers of other agents.
2. Market mechanisms, which work well for discrete decisions, but need to be developed further for continuous and real-time control.
3. Commitments that govern publically the interactions among a group of agents with their own interests.

Many of the solutions envisioned by my research team exploit some form of consensus, and indeed there is a power in consensus that has not been fully explored or exploited. The forms of consensus might involve opinion, preferences, interpretation, or behavior. Consensus can be determined via some type of voting, such as explored by Rosenschein and his colleagues [8].

We have produced preliminary consensus results by using Google to conduct a vote implicitly. For example, a query such as "population China" yields more than 70 million hits, which an agent can combine by averaging the data from a large number of the sources or by choosing the most common value, which is equivalent to arranging a vote among all of the 70M websites.

In another example, we have determined the "correct" English grammar for prepositional phrases by arranging a "vote" via Google over alternatives. The phrase "in the West Coast" yields 0.6M hits, the phrase "on the West Coast" yields 10.2M hits, and the phrase "at the West Coast" yields 0.2M hits. The phrase that wins the vote, "on the West Coast," is deemed to be the grammatically correct phrase.

It is less clear how to establish consensus functionality. For example, which is the best financial Web service for providing stock quotes? Even more difficult is determining consensus behavior. For example, out of the more than one million recipes on the Web for chocolate chip cookies, which is the best? This is equivalent to determining, for a given function, the best algorithm from a code repository.

Determining and then exploiting consensus remains a continuing challenge, but it appears to be essential for solving some of the problems described next.

4. DECENTRALIZED APPLICATIONS FOR MAS AND SOC

The following subsections provide more details about each of the application areas listed in the introduction and explain why MAS and SOC are the appropriate technologies for addressing them. All of the applications require that control information be obtained *from* the edge of a distributed system to provide desired behaviors *at* the edge of the system.

4.1 Healthcare IT for Patients

The healthcare triad consists of (1) patients, (2) caregivers (doctors and nurses), and (3) institutional facilities (hospitals, HMOs, and insurers). There are a variety of information systems available to support caregivers and facilities, but none to support individual patients, who typically are willing to assist each other. Because patients are naturally distributed, it seems that distributed multiagent systems would be appropriate. Agent-based information systems are needed that, for example, assist patients in locating and recommending doctors to each other, monitor the spread of cold and flu symptoms and their treatments, and provide advice on cost-effective drugs and care.

4.2 Grocery Shopping for Consumers

Aided by information systems for analyzing customer buying data, supermarket chains continually alter the prices of items to maximize their profits. They do this by, in essence, experimenting on their customers. For example, the price of an item might be raised at one store until customers stop buying it. This price is then used at all of the stores in the chain. The customers at the supermarkets, however, do not have any information systems that might aid them in price comparisons and are often at the mercy of the stores. Imagine a system where customers post the prices they paid for their groceries (this could be automated by querying the RFID tags of the items) and where a prospective shopper could enter a grocery list and obtain a pointer to the store with the lowest total price. This would make the customer-to-store interactions fairer and would encourage stores to offer their true prices. To achieve this status, each customer would need an agent to represent his/her interests and to interact with the agents of the other customers.

4.3 Individualized Traffic Control

Traffic control systems are most often designed by traffic engineers, with the system behavior fixed at intersections or modified centrally. There are projects underway (for example [9]) where cell phones with GPS automatically send the coordinates of drivers to a central server that fuses their location data with speed and traffic light sensors to construct a model of the current traffic flow. The result is then returned to the cell phone as a map of traffic, enabling each commuter to plan an expeditious route. The problem is that the provided information is not specialized to the needs and plans of the individual drivers. An alternative system might try to honor the preferences of all drivers in real time as they traveled to their destinations. For example, they would prefer that all traffic lights at the intersections along their route turn green when they approach them. Moreover, the system is reactive and does not consider the previous driving history of the commuter. For example, a commuter that has just been stopped by several red lights should have a higher preference for a green light at the next intersection. The problem that then arises is how to balance such behaviors fairly for all drivers One possibility is by computing consensus at each intersection, but this can be done only with agents representing each commuter and only in a local—not centralized—manner.

4.4 Taxation by Consensus

Most people do not mind paying taxes (Oliver Wendell Holmes, Jr. said, "Taxes are the price we must pay for civilization."), as long as they must pay exactly their fair share. Taxes pay for the "commons," such as infrastructure and physical security, but not everyone uses and benefits from the commons to the same extent. The problem is how to determine each person's "fair share." Tax laws do not do a good job at this, because they are written for the general case, not the specific case; that is why the laws are so voluminous and are patched so frequently.

A form of the solution is that individuals should pay what everyone else in similar circumstances pays. Determining this is a problem for multiagent systems and would involve the calculation of a form of consensus: a majority of the society should agree on what this value is.

In a related problem, how could this form of consensus be applied to spending instead of taxation? That is, are there societal benefits from consumers agreeing to buy goods either from the same companies—perhaps to reward the companies' exemplary employment practices or their use of best-practices—or from different companies in order to maintain a competitive marketplace? In either case, agents representing consumers appear to be the best mechanism for implementing a rational strategy.

4.5 Legal Decisions by Consensus

Legal reasoning is often done by precedence. For a given case, the problem is to find all similar cases and then to form a consensus of the decisions rendered in those cases, possibly weighted by their degree of similarity. The cases might have occurred in a variety of jurisdictions having a variety of laws and regulations. The approach advocated here is to have agents represent each case and then debate and argue among themselves until a vote can be taken and consensus reached.

4.6 Cooperative Information Exchange

The World-Wide Web with its centralized search engines (e.g., Google and Yahoo) has largely solved the problem of information exchange, which is the problem I addressed in some of my original DAI research [6], but it is still a largely passive and client-server solution. Peer-to-peer interactions occur only within the search engines, where privacy concerns and the unavailability of local context prevent the interactions from being truly cooperative. Information retrieval via the major search engines is still not individualized, still largely ignores context, and does not take advantage of the ready availability of cooperative help from other users.

4.7 Microclimate Monitoring

With more than 260 million cell phones in the United States alone, cell phones could be an effective platform for collecting environmental data. By incorporating various sensors into the cell phones, they could provide information on temperature, humidity, pollen, and pollution. However, current projects to exploit the widespread deployment of cell phones perform the data aggregation centrally and the results are intended only to provide a detailed model of the environment, but not to control or affect the environment. This would be the next step and would require a collection of agent-based effectors and robots to be deployed at the network's edge.

There are many plans and trial implementations for location-based services, but they are currently geared toward individual behavior and not aggregate or social behavior. That is, the services are designed to interact with any individuals who are in the vicinity of a given location, but they do not attempt to measure or make use of the behaviors of groups near a location.

4.8 Supply Networks

The efficient and orderly movement of large amounts of supplies and goods world-wide is a massive problem in planning and scheduling. Current solutions to this problem are primarily centralized and top-down, with a strategy based on hierarchical decomposition. Decomposition is a good way of dealing with complexity. However, centralization leads to a relatively static solution that cannot deal with the inevitable contingencies—such as unforeseen delays, breakdowns, and missed assignments—and cannot take advantage of opportunities for synergism that often arise among tasks assigned to different nodes of the hierarchical plan.

An entirely different approach is a completely decentralized one based on negotiating self-interested agents, one for each item bein supplied. That is, each item—each box of parts, each pallet of goods—would be treated as an intelligent entity whose sole objective is to reach its destination. It would decide which mode of transportation it should take, how to contend for limited transportation, storage, and material-handling resources, and how to avoid or resolve conflicts with other intelligent items.

This approach constructs no plan per se, but the system operates as if there were a comprehensive and detailed plan. The operation would occur in two phases: (1) a strategic phase, in which destination and transit goals are assigned to each entity by a central authority, and (2) a tactical phase, in which the entities make local, autonomous decisions about their route and resource usage as they wend their way through a distribution network.

An analogy for understanding this approach is that of how a large number of commuters manage to travel from their homes to their workplaces each day. They each have a goal (e.g., to arrive at their destination by a certain time), share limited resources (e.g., roads and trains), and, most importantly, can make local decisions about when and how to make use of the resources. Imagine instead someone with a large computer attempting to coordinate the travel of all the commuters in a city. They would have to (1) schedule when each commuter should leave their homes, (2) plan all routes, (3) schedule when each should stop or go or turn at every intersection, etc. The planning task would be enormous and, even if an optimally efficient plan could be generated, it would fail when the inevitable exceptions occurred. However, millions of people manage to get to their workplaces each day without any central plan or control. They succeed by autonomously coordinating their actions with those of the other commuters, and making local decisions in the furtherance of their individual goals.

This decentralized solution for supply networks would use this commuter approach: each item would be given a destination (its goal), knowledge about relevant parts of the transportation system (routes and conveyances), and enough intelligence that it can make reasonable decisions when it encounters difficulties or choices, as indicated in Figure 1.

The distributed autonomous logistics agents follow the commuter approach by shifting the detailed decision process to the lowest practical level: the container or even individual items. The agents make dynamic local decisions to:

- Decide the means of conveyance
- Contend for storage, transportation and special handling resources
- Cooperate with other agents to satisfy mutual goals
- Transfer acquired expertise to other agents
- Request services to handle special problems and contingencies
- Respond to real-time queries or changes in objectives and plans.

This approach to supply networks is consistent with the emerging infrastructure of RFID tags and the "Internet of things."

4.9 Consumer Control of Power Distribution

European researchers working on the S-TEN project have developed a framework based on applied semantic Web technologies to make networks self describing [11]. In such a network, each component, such as a volt meter or wind turbine in the case of a power grid, autonomously publishes information on what it is, where it is, and what it does. Instead of storing information in a centralized database, the S-TEN approach is for each node, each sensor or device connected to the network, to have its own intelligence. Enabling energy sources to publish semantic data understandable to both machines and humans should lead to more efficient automated grid management and better decision-support systems for human operators. The data can be accessed through a Web interface to show current status of the power grid, what objects are parts of the grid, and what they are doing. The interface also provides grid monitoring and control to enable preventative maintenance strategies.

The system involves massively distributed intelligence, but only to provide status to central operators, not to accommodate users' preferences.

5. VERY LARGE-SCALE PARTICIPATORY DESIGN

There is a story, maybe apocryphal, about a university that built a new campus. After they built all of the buildings, they planted grass everywhere and did not build any sidewalks. They waited a year, and then paved all of the paths that had been worn through the grass. This is an example of large-scale, distributed (distributed in both space and time) design, although not necessarily cooperative. There are many similar examples where design is typically done in a centralized manner, but could be done much better in a distributed manner. The solution in this particular case requires a form of consensus, where the participants are literally voting with their feet, although they are probably unaware they are voting on a design. Could this be applied to software design?

In a similar vein, SnowMan software [4] helps transportation engineers design roadways that are less likely to be affected by snow drifts and helps maintenance crews more precisely position fences to reduce snow drifts on existing roads. The problem of

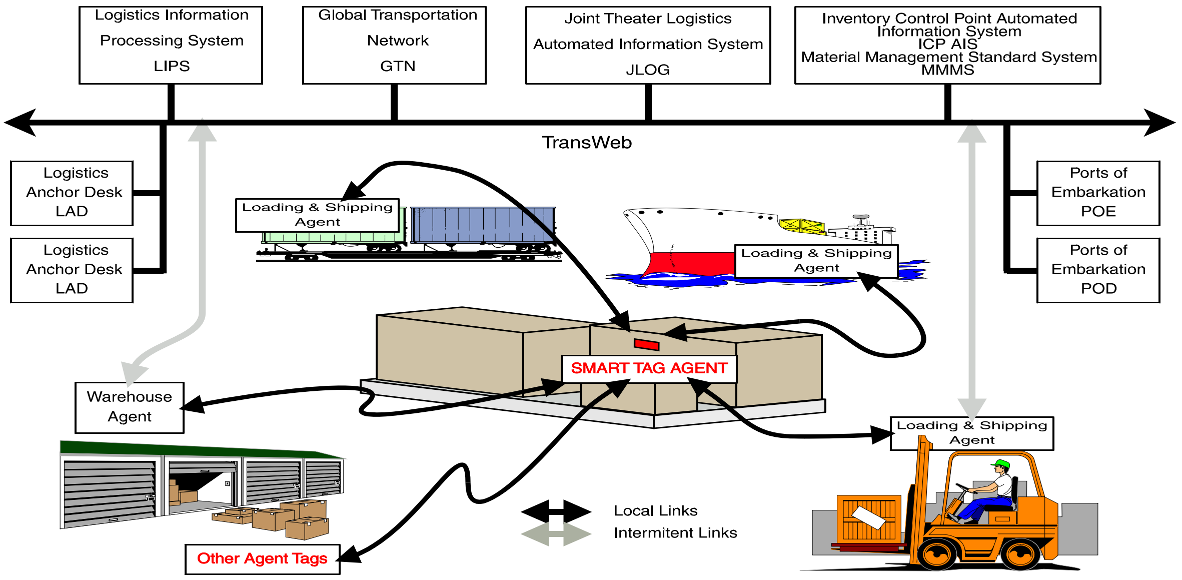
Logistics Information Processing System LIPS
Global Transportation Network GTN
Joint Theater Logistics Automated Information System JLOG
Inventory Control Point Automated Information System ICP AIS
Material Management Standard System MMMS
TransWeb
Logistics Anchor Desk LAD
Logistics Anchor Desk LAD
Ports of Embarkation POE
Ports of Embarkation POD
Loading & Shipping Agent
Loading & Shipping Agent
SMART TAG AGENT
Warehouse Agent
Loading & Shipping Agent
Local Links
Intermitent Links
Other Agent Tags

Figure 1. A supply network can be controlled from its edge by endowing each node at the edge with knowledge about routes, destinations, physical needs of the goods, and means of conveyance

coping with snow drifts on highways is a distributed one, and a distributed design approach provides a natural fit.

For another example, the routes of trains and automobile traffic are designed centrally and statically by transportation engineers, rather than designed in real time by the passengers being transported. Allowing the passengers to be responsible for the transportation system constitutes a form of design that is

- Large-scale, because of the size of the system being designed and the number of designers
- Spatially distributed, in terms of both the system and the designers
- Temporally distributed, because the design of the system evolves over time and the designers interact with it over a long time period
- Participatory, although not necessarily cooperative.

There are a number of similar and important societal problems that resist conventional, i.e., centralized, solutions. The problems affect the design of systems ranging from transportation to infrastructure and economies. Design problems such as these are typified as being distributed and many-faceted, with a large number of interdependent components. How can this sort of large-scale distributed design be supported? I believe it requires the aid of multiagent systems that—mimicking the university pedestrians—produce consensus.

6. HYPERSCALE DEVELOPMENT OF SOFTWARE

The benefits of the open-source (“Bazaar”) approach to software development have not been fully realized, because the number of software developers is still relatively small and orders of magnitude smaller than the number of users. Developers typically are experts in computing, whereas users typically have domain expertise: this produces a disparity in viewpoints, causing a mismatch between the developed software and its desired use. Moreover, the proposition, “given enough eyeballs, all bugs are shallow,” would take on much greater significance, if a larger fraction of the users could also be developers.

An interviewer recently asked Linus Torvalds (the creator of Linux) if Linux’s user base or developer base was more important [ComputerWorld, Vol. 41, No. 43, October 22, 2007]. Torvalds answered that he did not see the need to distinguish between them, because in Linux users can also be developers. In reality, however, there is a large distinction, because the number of developers (there were approximately 800 contributors to the latest release of the kernel) ranks in the hundreds and the number of users ranks in the millions.

In his influential essay “The Cathedral and the Bazaar”, Eric S. Raymond promulgated a view of open-source software development that contrasts two different development models [10]:

1. The Cathedral model, in which source code is available with each software release, but code developed between releases is restricted to an exclusive group of software developers. The development of GNU Emacs and GCC followed this model.
2. The Bazaar model, in which the code is developed over the Internet in view of the public. Linux development has been following this model.

In addition, Raymond posits that "given enough eyeballs, all bugs are shallow," which he terms Linus' law: the more widely available the source code is for public testing, scrutiny, and experimentation, the more rapidly all forms of bugs will be discovered. He claims that much more time and energy must be spent looking for bugs in the Cathedral model, since the code is available only to a few developers. The essay helped convince many open-source and free software projects to adopt Bazaar-style models, including Netscape Communications Corp., which released the code for Netscape Communicator to start the Mozilla and Firefox projects.

However, the superiority of a Bazaar-style model of open-source software development has not been evident, partly because the size of the developer community for both proprietary and open-source software is roughly the same, and partly because it is still orders of magnitude smaller than the size of the user community. There are not “more eyeballs” in open-source projects.

The distinction is not restricted to open-source software: Microsoft has approximately the same number of developers for Windows as there are for Linux, but the resultant operating system is used by more than 100 million people.

In another facet of the distinction, Fred Brooks [1] observed that software tends to resemble the organization that built it: this is often vastly different than the community that uses it, making it more difficult for users to understand it.

Based on this, would it be beneficial for the development of Linux, or any open-source software system, for the distinction between users and developers to be reduced and a much larger fraction of the users also to be developers? If the answer to this is yes, then how might this be made to occur?

Superficially, it would require (1) a straightforward and relatively easy way for users to contribute to the software development process and (2) a way to accommodate and manage the contributions.

A solution to this problem has so far been impractical: end-users often do not have sufficient expertise to contribute software, nor the time to learn how to do so, and there is no way to meld it with existing software until it has been proven correct or from a trusted developer.

What is needed is to broaden dramatically the number of people who contribute to the development of software. If successful, the result would be a more effective software-development process, greatly improved software reliability, and increased end-user satisfaction.

The obvious approach is to make use of—*take advantage of*—all contributed code, components, and designs until they have proven to be of no value. To do this, I suggest that agent-based wrappers could be developed to manage the necessary collaboration and competition, allowing the contributions to be used alongside their existing counterparts until their behavior and features can be assessed. The design and development of software is not a democratic process, however, and voting is often inappropriate for deciding which module might be better than another [12].

7. CONSENSUS AMONG SERVICES

The objective of service-oriented computing (SOC) is to construct software applications out of appropriate services available and executing in place anywhere across the Web. The applications, because they are naturally distributed, could be for any of the domains listed above. To achieve this objective at all requires that techniques for discovering and engaging services be developed. Doing this well requires that additional techniques be developed for ensuring desired quality of service (QoS) metrics.

7.1 Google for Services

Imagine that there are a large number of Web services, each described by a WSDL (Web Service Description Language) file. One might think that these could be indexed in the same way that Web pages are, so that search algorithms could help a developer find the services needed for a service-oriented computing application. That is, services could be clustered by applying text-mining techniques to WSDL files [7]. This works well for Web searches, because keywords describe the partial semantics of Web pages and the PageRank algorithm helps in determining the most appropriate pages. However, this would not be successful for services, because WSDL files describe the inputs and outputs of a service, and provide only minimal information about the semantics of their function and behavior. Moreover, there is no equivalent to the PageRank algorithm for services, because services do not have references or pointers to each other.

All is not lost, however, because services could still be clustered into categories and middle agents could then manage and use the clusters to help locate possible services to satisfy requests from users and developers. A procedure such as unit testing could then be used as a behavioral query tool to test candidate services and select ones that have the desired behavior. A negotiation between the service requestor and providers could then ensue to establish an agreed upon QoS and formalize a contract. The requestors and providers would commit to honor the resultant contracts.

7.2 Social Web

The Social Web (essentially the use of Web 2.0 technologies, such as AJAX) derives content and information organization from large-scale collaboration. Successful examples of this are *Flickr* for photo sharing and indexing, *GenBank* as an annotated collection for biological research, *Wikipedia* for a collaborative encyclopedia of general knowledge, and *del.icio.us* as a folksonomy. Folksonomies have the ability to form stable structures via consensus over large sets of tags. A similar collective categorization scheme could be used as an initial organization and semantic description for software services across the Web. Such social consensus could yield agreement on behavior, but not necessarily robustness and desired quality, for the resultant applications. Robustness requires diversity, which is in opposition to social organization [2].

8. CONCLUSION

A multiagent foundation provides for autonomous active components that can engage in n-party interactions, negotiate with each other, and reconcile their individual, possibly idiosyncratic semantics. They are a *necessary* basis, but not a *sufficient* basis for solving the next generation of massively distributed societal problems: new approaches, possibly exploiting consensus among the large number of participants affecting and being affected by the problems are needed.

Individualized purchasing and manufacturing are becoming routine. Individualized healthcare, transportation, energy distribution, and climate modification are now conceivable. A development I hope is not realized is individualized warfare.

9. ACKNOWLEDGMENTS

My sincere thanks to three decades of researchers and practitioners in distributed artificial intelligence and multiagent systems. The future awaits!

10. REFERENCES

[1] BROOKS, F. 1995. *The Mythical Man Month*, Addison-Wesley, Reading, MA.

[2] FISCHER, G. 2005. Distances and Diversity: Sources for Social Creativity. In *Proceedings of Creativity and Cognition*, London, April 2005, 128-136.

[3] GHENNIWA, H.H. AND HUHNS, M.N. 2004. An Agent-Oriented Marketplace Architecture for Enterprise Integration. In *Encyclopedia of Information Science and Information Technology*, vol. I-III, M. KHOSROW-POUR, Ed. Idea Group Publishers.

[4] GOLDBAUM, E. 2009. "SnowMan" Software Developed at UB Helps Keep Snow Drifts Off the Road. University of Buffalo News, (January 29, 2009), http://www.buffalo.edu/news/fast-execute.cgi/article-page.html?article=98820009

[5] HUHNS, M.N. AND STEPHENS, L.M. 2001. Automating Supply Chains. *IEEE Internet Computing 5*, 90-93.

[6] HUHNS, M.N. 1984. The MINDS Approach to Distributed Artificial Intelligence. In *Fifth Workshop on Distributed Artificial Intelligence*. Ridgefield, CT, October 1984.

[7] LIU, W. AND WONG, W. 2008. Web Service Clustering using Text Mining Techniques. *International Journal of Agent-Oriented Software Engineering (JAOSE)*, Inderscience, in press.

[8] MEIR, R., PROCACCIA, A.D., AND ROSENSCHEIN, J.S. 2008. A Broader Picture of the Complexity of Strategic Behavior in Multi-Winner Elections. In *The Seventh International Joint Conference on Autonomous Agents and Multiagent Systems*, Estoril, Portugal, May 2008, 991-998.

[9] PATEL-PREDD, P. 2009. Cellphones for Science. *IEEE Spectrum*, February 2009.

[10] RAYMOND, E.S. 1998. The Cathedral and the Bazaar. http://www.firstmonday.org/issues/issue3_3/raymond/.

[11] Semantic Web Promises a Smarter Electricity Grid. ICT. http://cordis.europa.eu/ictresults/index.cfm/section/news/tpl/article/id/90399

[12] ZAVALA, L. AND HUHNS, M.N. 2008. Analysis of coincident failing ensembles in multi-version systems. In *Proc. 19th IEEE International Symposium on Software Reliability Engineering: Dependable Software Engineering Workshop*. Seattle, WA, November 2008.