Achieving Global Coherence in Multi-Agent Caregiver Systems: Centralized versus Distributed Response Coordination in I.L.S.A.*

Thomas A. Wagner Valerie Guralnik John Phelps
Honeywell Laboratories
3660 Technology Drive, MN65-2600
Minneapolis, MN 55418
http://www.drtomwagner.com/
Tom.Wagner@htc.honeywell.com

Abstract

In order for multi-agent systems to exhibit global coherence the agents must coordinate their activities or be limited to a problem space in which activities are highly independent. In the elder care domain, global coherence across the agent system responsible for giving the care requires coordination if only because the different agents in the system interact over a set of shared resources, namely the individual for whom the system provides care and the fallback human caregivers. In this paper we explore the general coordination issue in the elder care problem space and discuss the *response planning and coordination* portions of the Independent LifeStyle Assistant™ agent system being developed at Honeywell.

Introduction

One key aspect of multi-agent systems (MAS) on which different researchers can agree is that the MAS model means distributed processing and distributed control. As with other distributed processing models, one important problem of MAS research is how to obtain globally coherent behavior from the system when the agents operate autonomously and asynchronously. In general, when the agents share resources or the tasks being performed by the agents interact, the agents must explicitly work to coordinate their activities.[1] Consider a simple physical example. Let two maintenance robots, R1 and R2, be assigned the joint task of moving a long table from one room to another. Let both robots also have an assortment of other independent activities that must be performed, e.g., sweeping the floor. Assume that neither robot can lift the table by him/herself. In order for the robots to move the table together they must coordinate their activities by 1) communicating to determine when each of the robots will be able to schedule the table moving activity, 2) possibly negotiating over the time at which they should move the table together, 3) agreeing on a time, 4) showing up at the table at the specified time, 5) lifting the table together, and so forth. This is an example of communication-based coordination that produces a temporal sequencing of activities enabling the robots to interact and carry out the joint task (over a shared resource – the table). Without the coordination process, it is unlikely that the table would ever be moved as desired unless the robots randomly decided to move the table at the same moment in time. Note that if the robots are designed to "watch" each other and "guess" when the other is going to move the table that this is an instance of coordination by plan inference and still counts as a coordination episode. In general, achieving global coherence in a MAS where tasks interact requires coordination.

In the robot/table example, the coordination episode is peer to peer. Imagine now a room full of maintenance robots, each having multiple joint tasks with other agents and all sharing physical resources such as tools and floorspace or X/Y coordinates. Without coordination said room full of robots would have much in common with a preschool "free play session" with robots moving about, unable to perform tasks due to obstacle avoidance systems always diverting them from their desired directions or due to the lack of a required tool. There are two primary ways to coordinate this room full of robots – either in a distributed peer to peer (or group to group) fashion or in a centralized fashion. When coordination is distributed each agent is responsible for determining when to interact with another agent and then having a dialog to determine how they should sequence their activities to achieve coherence. When coordination is centralized generally one agent plans for the others or manages a shared resource. Note that in the example above coordination focuses on *when* to perform a given task. Coordination can also be about *which* tasks to perform, *what resources* to use, *how* to perform a task, and so forth.

While the robot domain is good for illustrating conceptually the coordination problem, the need for coordination is not limited to robots. Consider the Independent LifeStyle Assistant™ (I.L.S.A.) system being developed at Honeywell. I.L.S.A. is a multi-agent system that monitors an individual in his/her home and automates aspects of the care giving process. For instance, I.L.S.A. may notify a caregiver if the individual being monitored should go too long without eating. I.L.S.A. is a multi-agent system where dif-

*This work was performed under the support of the U.S. Department of Commerce, National Institute of Standards and Technology, Advanced Technology Program, Cooperative Agreement Number 70NANB0H3020.

[1]For application domains in which the activities of the agents are mostly independent, or the actions carried out by the agents are particularly lightweight (so the implications of an intersection are slight), explicit coordination may not be necessary.

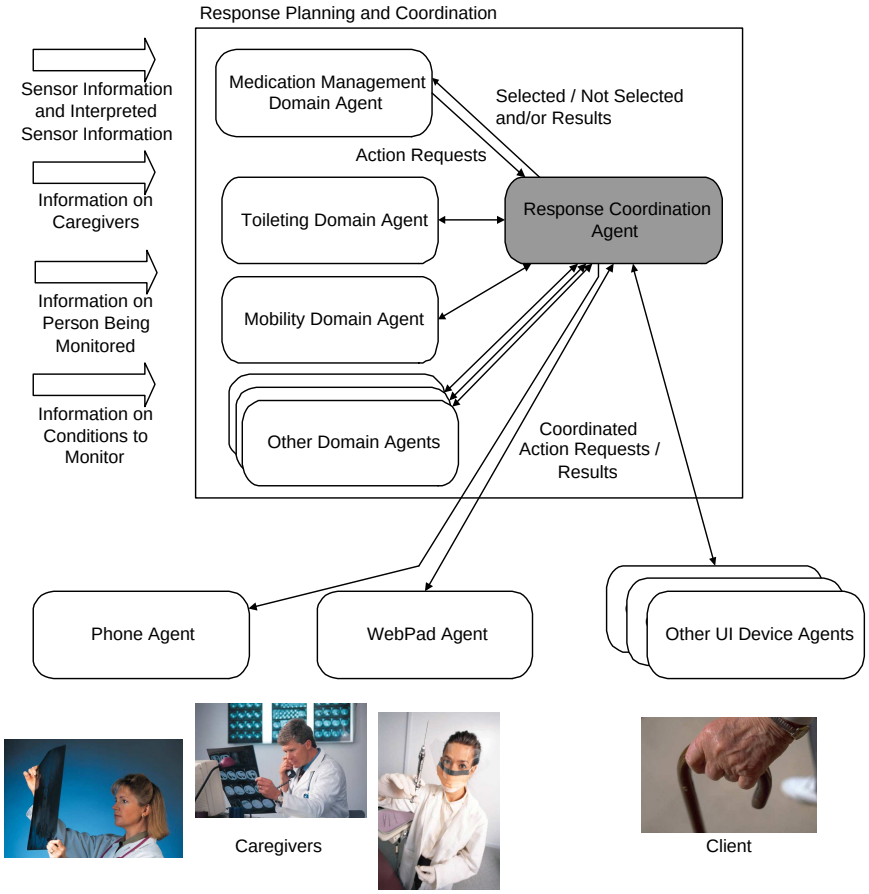

Figure 1: The Response Planning Component of I.L.S.A.

ferent agents are responsible for sensing, reasoning about the sensor data, deciding on a course of action, and interacting with the caregivers and the individual for which the system is giving care. Without coordination in I.L.S.A. it would be possible for the system to exhibit undesirable characteristics. For instance, the agents might overburden the individual for which care is being given by issuing a series of separate reminders all within a few minutes of one another, e.g., "it is time to take your medication," "it is time for lunch," "you haven't been moving as much today," "I believe the oven is on." Imagine the impact of having a telephone ring or a beeper go off every few minutes with a different reminder function. Worse still, imagine if the client were to fall and the emergency notification being sent to a caregiver was delayed so that the client could be reminded to eat his/her lunch. Or imagine that same client, who is unable to get up, listening to the phone ring where the ring is caused by a reminder that it is time to take a nap. I.L.S.A.'s MAS requires coordination. Through coordination we can organize the responses produced by the different agents and achieve globally coherent behavior for the system as a whole.

I.L.S.A.'s overall architecture and the system's design goals are more fully documented in other papers (Haigh, Phelps, & Geib 2002). In this paper we focus on the coordination problem of the I.L.S.A. system and compare it to the coordination techniques used in the UMASS IHome project, which is another multi-agent system for intelligent facilities management. While IHome's goals are different there are strong parallels in the projects.

For the purposes of this paper we will use the moniker "Lois" to denote the individual of which I.L.S.A. is taking care and the term "caregiver" to denote some health care professional that may assist I.L.S.A. in dealing with certain situations.

In the following sections we provide more detail about the agent coordination problem in I.L.S.A., discuss the process of selecting between distributed and centralized coordination, discuss the coordination technique used in I.L.S.A. and identify future coordination issues.

Coordination Evaluation in I.L.S.A. is Part of a Larger Effort

The study of coordination in MAS has a long history that goes back to the early work in distributed artificial intelligence (Lesser & Erman 1980; Lesser & Corkill 1981). Recent efforts include generalized approaches to coordination (Decker & Li 1998), team centered coordination (Tambe & Zhang 1998), formalisms of joint problem solving (Grosz & Kraus 1996; Cohen & Levesque 1990; Wooldridge & Jennings 1994), research in coordination protocols (Singh 1998) and others (Jennings, Sycara, & Wooldridge 1998). Despite progress in these areas, the community as a whole

lacks a science or methodology that one can apply to determine which type of coordination is appropriate for a given application. The underlying science that is needed is 1) an understanding of the important domain characteristics (e.g., dynamic and distributed, size and scope), and 2) how these map to the strengths of the different coordination technologies and approaches that are used. For instance, in a dynamic supply chain management problem (Wagner, Guralnik, & Phelps 2002) where agents are situated at the sites of all involved parties and all products are built to order, there is no centralized control and there is no steady market state. In this case, agents must coordinate by reasoning about the value of candidate options as well as order deadlines and interactions between different options (e.g., a manufacturer needs materials from multiple suppliers to fill an order and must thus negotiate with them over delivery times to meet its deadline). This is in contrast to the I.L.S.A. caregiver domain where responses can be centrally coordinated and the coordination process does not involve interactions between certain classes of activities. The contribution of this particular paper is in the high-level analysis of the I.L.S.A. coordination space and in the centralized coordination approach used. The community is working to develop a science in this area, e.g.,(Durfee 2001) and (Wagner, Smith, & Vouros 2002).

Response Planning and Coordination in I.L.S.A.

In the current implementation of I.L.S.A. the coordination problem pertains to organizing activities over shared resources – namely the interfaces to the caregivers and Lois. As discussed in the conclusion other aspects of I.L.S.A. are candidates for coordination, e.g., the coordination of multiple I.L.S.A. instances and the coordination of agents within I.L.S.A. over task relationships.

I.L.S.A.'s overall architecture is documented in (Haigh, Phelps, & Geib 2002). Here we focus on the subset of the architecture that deals with 1) deciding how to respond to a given situation that exists with Lois, and 2) coordinating the different responses that may be on going at any given moment in time. This functionality is loosely called *response planning and coordination* and the subset of the architecture is shown in Figure 1. Response planning in I.L.S.A. is carried out by multiple domain agents, each of which has a particular area of expertise. For instance, there is a domain agent that monitors Lois' medication and issues reminders to her if she forgets to take her medication on schedule and issues notifications and alerts to caregivers if Lois does not correct her medication situation. Other domain agents specialize in toileting, eating, falls, mobility, and sleeping, to name a few. The domain agents receive multiple different types of input from other agents in the system. Specifically:

- The intent recognition agent provides information on the plans Lois is likely to be performing. For instance, Lois might be cooking dinner with some specified probability.
- The sensor clustering agent provides filtered sensor data, e.g., toilet flushes or information from motion detectors. This agent also provides the clock pulse.
- The database agent provides the domain agents with information on conditions for which they are to monitor (e.g., track the consumption of medication X) and how they should respond if a particular circumstance arises. For instance, if Lois fails to take her medication, first remind her, then notify caregiver X using device Y, then if she remains inappropriately unmedicated issue an alert to caregiver Z using device K.
- Domain agents also receive feedback from a *response coordinator* (below) agent and results back from the device agents when appropriate (e.g., indicating the caregiver X has accepted an alert and will handle the condition).

From this data the domain agents monitor the environment, monitor Lois' condition, and respond when necessary. Domain agents respond to problems by interacting with Lois or one or more of the caregivers. Responses fall into four categories: *reminders* (to Lois), *notifications* (to caregivers of some condition), *alerts* (to caregivers of some serious condition), and *alarms* (to caregivers of some life threatening condition). The domain agent responses are the coordination focal point. The response coordination problem in I.L.S.A. has the following properties:

- The domain agents operate asynchronously and autonomously and generally do not interact to solve problems. The implication is that any domain agent may generate an *action request* at any time. For our purposes it is sufficient to view action requests as a request to interact with a caregiver or Lois via one of the UI devices.
- The agents share several resources, namely the UI devices and indirectly the caregivers and Lois.
- In addition to the resource interaction, action requests potentially intersect on a temporal basis. In this case for there to be a resource coordination problem the requests must occur within a certain temporal proximity to one another or within a system state window (below).
- The response space for a particular individual caregiver or a particular client (Lois) is relatively sparse, i.e., the number of interactions that the system needs to have with any given individual is small over time. We can make this characterization because the endpoints of these action requests are human and bounded by human capacity. For instance, Lois cannot process 10,000 interactions a minute nor can a given care provider.
- In contrast, the load of a given device may be relatively high when compared to the load on an individual caregiver or Lois. Consider an I.L.S.A. installation where the only interaction medium for Lois and all caregivers is the telephone. The telephone itself may be a bottleneck unless usage is coordinated accordingly.
- In addition to interacting over shared resources, action requests interact with each other through priority. For instance, if the falls domain agent issues an alert because it has determined that Lois has fallen, the alert condition should take precedence over a previously issued, but not yet fulfilled, request to notify a caregiver that Lois is behind on her medication.

- Action requests also interact through system state. For instance, if the panic agent issues an alarm because Lois has pressed her panic button, reminders from the other domain agents (to Lois) must be suppressed.
- Different responses have different interaction protocols that I.L.S.A. must follow. For instance, reminders go to Lois. Notifications are sent to caregivers but the caregivers do not need to explicitly acknowledge them. Alerts are sent to caregivers and require explicit acceptance – plus alerts cause I.L.S.A. to cycle over a list of caregivers until one is found who will accept responsibility for handling the alert. Alarms are sent to a list of selected caregivers all at once, in parallel. Each of these responses is further conditioned by device, e.g., caregiver Bob might always want to be notified via cell phone rather than beeper.
- One of the design goals of I.L.S.A. is to keep the architecture open so that 3rd parties can add custom or enhanced functionality.
- The domain agents will execute in close proximity to one another or will have a reliable network connection between them. This is necessary for them to obtain information about the environment and to monitor Lois.

On the surface, the response coordination problem space in I.L.S.A. is akin to that found in the UMASS IHome (Lesser *et al.* 1999) project. In IHome agents manage different appliances such as the hot water heater, dish washer, washing machine, coffee maker, and coordinate their activities to improve the quality of life for the occupants, e.g., making sure that there is sufficient hot water for morning showers while still getting the laundry done and the dishes washed. In IHome coordination centers on shared resources like hot water, electricity, noise, and a shared mobile robot (simulated) that can perform selected tasks within the environment.

In both IHome and I.L.S.A. two primary approaches for solving the coordination problems exist – centralized and distributed. In IHome both centralized and distributed approaches are used and the motivation for each selection is more well understood today than it was at the time the decision was made. For resources that are characterized as 1) not centralized and 2) not heavily contested, a decentralized approach is used. In the decentralized approach agents are responsible for coordinating on their own behalf, i.e., there is no single agent that serves as the moderator or controller for that resource. The noise resource is an example of a resource over which the agents coordinate in a distributed fashion. For resources that are centralized, such as hot water which is produced locally by the hot water heater, an agent is assigned the task of controlling and coordinating usage of the centralized resource. In the case of hot water, a hot water heater agent handles the allocation of hot water to individual agents for their use.

What are the important differences between the hot water resource and the noise resource in IHome? How do these differences relate to I.L.S.A.? The hot water resource is inherently centralized. The noise resource is inherently distributed (spatially). This is an important distinction because not all agents that make noise actually need to coordinate in IHome. For instance, the television agent would not need to coordinate with the vacuum cleaning agent if they are located in different rooms of the home. In this case, a centralized coordination mechanism is unnecessary and undesirable (assuming that the process of determining whether or not the agents need to coordinate is low cost). In contrast, the hot water used by the dishwasher always comes from the same source as the hot water used by the shower which always comes from the same source as the hot water used by the washing machine and so forth. In this case, these agents always need to coordinate their activities and their coordination will often span more than an individual pair of agents. Additionally, in IHome, hot water proves to be a fairly contentious resource so centralized coordination serves to reduce message traffic and coordination overhead.

To summarize, axis along which to evaluate a coordination approach suggested by IHome include 1) how many agents use a given resource, 2) whether the resource is inherently centralized or distributed, 3) how contentious the agents are for the resource. Another issue that arises in IHome is when multiple resources are required by a given agent before any processing can be performed, e.g., the washing machine needs noise, electricity, and hot water resources to function. This class of resource issues does not occur in I.L.S.A. (yet) though the first three criteria are useful metrics in I.L.S.A..

Now consider I.L.S.A.'s coordination problem as specified above. If we distill that information and correlate related items we can enumerate a subset of important issues to consider when evaluating I.L.S.A.'s coordination needs:

- All of the domain agent action requests interact over the set of UI devices (directly) and the set of caregivers $\cup$ Lois (indirectly).
- Action requests also interact with each other through system state and priority. Note that system state is a global condition (e.g., system is in alarm state so suppress all reminders to Lois).
- The number of action requests expected to be issued by the system is computationally bounded on the upside by the sum of the processing capabilities of the caregivers plus the capability of Lois, i.e., the number of requests the system will generate and process will be small by computer standards.
- The action requests are lightweight activities that do not require large amounts of processing to evaluate.
- The agents do not need to engage in negotiation activities or perform complex cross agent task sequencing activities.
- Because I.L.S.A. must adhere to a set of protocols for issuing notifications, alarms, alerts, and reminders, if domain agents are given direct access to the devices they must each implement portions of the protocols (some of the protocol requirements are implemented by the device agents).
- I.L.S.A. should support the addition of 3rd party / after market agents to the system.

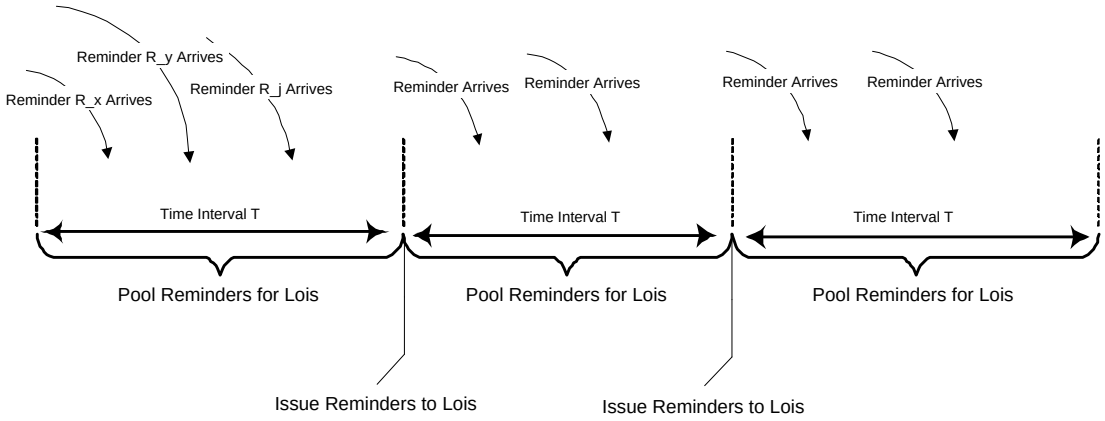

Figure 2: Centralized Coordination Pacing Reminders

The end result? We selected a centralized approach to response coordination in I.L.S.A.. This is due in part to the way in which the lightweight action requests may interact and due to the ratio of requests to caregivers that the system will exhibit in order to be useful. Centralization provides an efficient path for response coordination that we believe will scale with the system in the case of I.L.S.A.. Centralization is also particularly appropriate for I.L.S.A. because the action requests are lightweight and not strongly situated temporally, i.e., the domain agents do not need real-time time or performance guarantees thus consequences of an action request being delayed by another are slight (if any). Centralized coordination is still required (versus no coordination) to ensure that the agents do not overwhelm the caregivers or Lois and to ensure that important messages are sent in a timely fashion. Centralization also enables us to encapsulate certain trusted behaviors in an agent of our own design. If coordination were performed in a peer to peer fashion, a poorly written or malicious 3rd party agent could wreak havoc with the system by not coordinating well (or at all).

The centralized approach to coordination is implemented by the response coordinator, shown in Figure 1. This agent is responsible for accepting all action requests from the domain agents and coordinating the requests as follows:

- If multiple reminders for Lois occur within some period of time $\mathcal{T}$ the reminders are multiplexed into a single message so that Lois only receives one phone call or one email message that contains the set of reminders. Space precludes presentation of the algorithm but it is illustrated in Figure 2. The general idea is to create temporal buckets into which all requests that occur within their boundaries are tossed. When the edge of the temporal window arises, the reminders contained in the bucket are multiplexed and sent. The implications are that some reminders will be delayed slightly, however, this is also the only way to ensure that Lois is bothered by a reminder at most every $\mathcal{T}$ time units.
- If multiple notifications for the same caregiver occur within the same interval of time, they are treated as the reminders are for Lois (above).
- When alarms and alerts are generated, they are immediately dispatched to the appropriate devices according to the appropriate protocol (which the response coordinator implements.)
- If the system is in a state such that reminders to Lois are to be suppressed (e.g., an alarm has been issued), the action requests are returned to the domain agents instead of being issued to Lois. The domain agents are told why the requests were refused and are able to reissue the request at some point in the future. If the condition has cleared in the meantime, the new requests are granted. Note that the task of tracking the system state and knowing when it is cleared is also centralized with the response coordinator.
- Note that device load is moderated by the binning algorithm shown in Figure 2. If device load becomes a performance issue we can adjust the algorithm or implement other protocols, e.g., store low priority messages for longer periods of time, easily by changing the response coordinator's control algorithm. (Currently aspects of this class of concerns are also carried forward into the implementation by using priority queues at the device agent level so that alarms and alerts are always issued before reminders and notifications.)

It is important to note that centralization in the *general case* may not always be desirable. Centralization can lead to a localized performance bottleneck and creates a system with a single point of failure. If the agents are spatially distributed centralization also introduces network uncertainty, latency, and connectivity issues so that even if the centralized coordinator is still online and functioning agents who are unable to connect are unable to coordinate. Other issues with centralization include scale and computational limitations – if coordination requires deep temporal analysis it is unlikely that any single agent could handle the coordination problem of even a fairly modest MAS. Other issues that are particularly important for agent-based ecommerce are localized control and information hiding or privacy. In general, we strongly advocate a distributed approach like that used in our supply chain research (Wagner, Guralnik, & Phelps 2002) and other research that is based on TÆMS (Decker & Li 1998), DTC agent scheduling (Wagner, Garvey, & Lesser 1998), and GPGP agent coordination (Decker & Li 1998). However, as enumerated above, I.L.S.A.'s requirements are different and the characteristics of the current and near term

problem space lead us to a centralized approach.

Conclusions and Future Work

Coordination within I.L.S.A. currently centers on coordination over shared resources, i.e., the interface to the caregivers and to Lois. In the future it may be necessary to coordinate the activities of the other agents in the system. For instance, if the computational tasks being performed by the agents become heavyweight enough that they over burden the processor, performance of certain tasks may have to be coordinated to avoid unacceptable system slow downs. Another example is if the sensor interpretation agents were to only perform certain analysis tasks if the output were needed by one or more of the agents in the system. In the current model, the agents within the system do not explicitly reason about the interactions in their activities and do not coordinate over said interactions. If agents were to reason about such interactions, it is highly probable that distributed temporal task centered coordination like that used in our other work, based on TAEMS/DTC/GPGP, will be appropriate.

Another area in which coordination will play a role in the I.L.S.A. project is when the I.L.S.A. system is deployed in a multi-unit care facility. In this setting, one I.L.S.A. system will take care of one client but the pool of human caregivers will be common across the set of I.L.S.A. systems (or at least subsets may be common). The individual I.L.S.A. systems will thus need to coordinate over non-emergency tasks to optimize caregiver time and to prevent thrashing behavior (caregivers being overbooked, not having sufficient time with each client, etc.).

Acknowledgments

The I.L.S.A. project is a large effort at Honeywell Laboratories and many individuals have contributed to the intellectual underpinnings of the project. Some of the contributors are: Chris Miller, Karen Haigh, David Toms, Wendy Dewing, Steve Harp, Christopher Geib, John Phelps, Peggy Wu, Valerie Guralnik, Ryan Van Riper, Thomas Wagner, Steven Hickman, and others.

References

Cohen, P., and Levesque, H. 1990. Rational interactions as the basis for communication. In Cohen, P.; Morgan, J.; and Pollack, M., eds., *Intentions in Communication*. MIT Press.

Decker, K., and Li, J. 1998. Coordinated hospital patient scheduling. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, 104–111.

Durfee, E. H. 2001. Scaling up agent coordination strategies. *IEEE Computer* 39–46.

Grosz, B. J., and Kraus, S. 1996. Collaborative plans for complex group action. *Artificial Intelligence* 86:269–357.

Haigh, K.; Phelps, J.; and Geib, C. 2002. An open agent architecture for assisting elder independence. In *Proceedings of the First Intl. Joint Conference on Autonomous Agents and Multi-Agent Systems*. To appear.

Jennings, N. R.; Sycara, K.; and Wooldridge, M. 1998. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems* 1(1):8–38.

Lesser, V. R., and Corkill, D. D. 1981. Functionally accurate, cooperative distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics* 11(1):81–96.

Lesser, V. R., and Erman, L. D. 1980. Distributed interpretation: A model and an experiment. *IEEE Transactions on Computers* C-29(12):1144–1163.

Lesser, V.; Atighetchi, M.; Horling, B.; Benyo, B.; Raja, A.; Vincent, R.; Wagner, T.; Xuan, P.; and Zhang, S. X. 1999. A Multi-Agent System for Intelligent Environment Control. In *Proceedings of the Third International Conference on Autonomous Agents (Agents99)*.

Singh, M. P. 1998. Developing formal specifications to coordinate heterogenous autonomous agents. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, 261–268.

Tambe, M., and Zhang, W. 1998. Towards Flexible Teamwork in Persistent Teams. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*.

Wagner, T.; Garvey, A.; and Lesser, V. 1998. Criteria-Directed Heuristic Task Scheduling. *International Journal of Approximate Reasoning, Special Issue on Scheduling* 19(1-2):91–118. A version also available as UMASS CS TR-97-59.

Wagner, T.; Guralnik, V.; and Phelps, J. 2002. Software agents: Enabling dynamic supply chain management for a build to order product line. In *Proceedings of Agents for Business Automation*. A version also appears in the Proceedings of the AAAI02 Workshop on Agents for Business Automation.

Wagner, T. A.; Smith, S.; and Vouros, G. 2002. Toward an Application Science: MAS Problem Spaces and Their Implications to Achieving Globally Coherent Behavior. Workshop held at the 1st International Conference on Autonomous Agents and Multi-Agent Systems.

Wooldridge, M., and Jennings, N. 1994. Formalizing the cooperative problem solving process. In *Thirteenth International Workshop on Distributed Artificial Intelligence*, 403–417.