DON'T CANCEL MY BARCELONA TRIP*: Adjusting autonomy of agent proxies in human organizations

Paul Scerri, Milind Tambe, Haeyoung Lee, David Pynadath
Information Sciences Institute and Computer Science Department
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
{scerri, tambe, hlee, pynadath}@isi.edu

March 28, 2000

1 Introduction

Teams of heterogeneous agents working within and alongside human organizations offer exciting possibilities for streamlining processes in ways not possible with conventional software[4, 6]. For example, personal software assistants and information gathering and scheduling agents can coordinate with each other to achieve a variety of coordination and organizational tasks, e.g. facilitating teaming of experts in an organization for crisis response and aiding in execution and monitoring of such a response[5].

Inevitably, due to the complexity of the environment, the unpredictability of human beings and the range of situation with which the multi-agent systems must deal, there will be times when the system does not produce the results it's users desire. In such cases human intervention is required. Sometimes simple tweaks are required due to system failures. In other cases, perhaps because a particular user has more experience than the system, the user will want to "steer" the entire multi-agent system on a different course. For example, some researchers at USC/ISI, including ourselves, are currently focused on the Electric Elves project (http://www.isi.edu/agents-united). In this project humans will be agentified by providing agent proxies to act on their behalf, while entities such as meeting schedulers will be active agents that can communicate with the proxies to achieve a variety of scheduling and rescheduling tasks. In this domain at an individual level a user will sometimes want to override decisions of their proxy. At a team level a human will want to fix undesirable properties of overall team behavior, such as large breaks in a visitor's schedule.

However, to require a human to completely take control of an entire multi-agent system, or even a single agent, defeats the purpose for which the agents were deployed. Thus, while it is desirable that the multi-agent system should not assume full autonomy neither should it be a zero autonomy system. Rather, some form of *Adjustable Autonomy* (AA) is desired. A system supporting AA is able to dynamically change the autonomy it has to make and carry out decisions, i.e. the system can continuously vary its autonomy from being completely dependent on humans to being completely in control. An AA tool needs to support user interaction with such a system.

To support effective user interaction with complex multi-agent system we are developing a layered Adjustable Autonomy approach that allows users to intervene either with a single agent or with a team of agents. Previous work has in AA has looked at either individual agents or whole teams but not, to our knowledge, a layered approach to AA. The layering of the AA parallels the levels of autonomy existing in human organizations. Technically, the layered approach separates out issues relevant at different levels of abstraction, making it easier to provide users with the information and tools they need to effectively interact with a complex multi-agent system.

*Location of Agents 2000 conference

The single agent, or local, layer of the AA system provides a spectrum of options regarding the autonomy the agent has in making a decision on the user's behalf. In our current prototype, agent proxies use learning to adapt their behavior to the user's preference for which decisions the user likes to make and which ones the agent should make autonomously. The local layer can exploit detailed information about a particular user to be very specialized for that user.

The team layer of the AA system abstracts away details of individual agents and uses it's background knowledge of team behavior to focus the user on important team features and problems. A user is provided tools to monitor the coordination and team plans of the team, and then modify assignments to roles, team plans etc. as required. Without adjusting the local autonomy of individual agents the user can guide the behavior of the entire team.

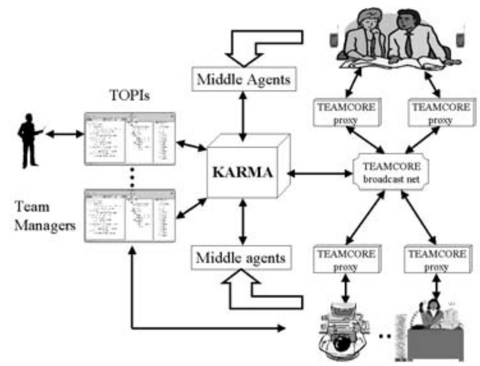

Figure 1: Overview of the Teamcore framework.

2 Electric Elves

The USC/ISI "Electric Elves" project is an integrated effort focused on dynamic team formation and crisis response in complex human organizations. In highly agentified future organizations — which agentify all active entities, including humans — dynamic teaming of agents will enable the organizations to act coherently and robustly toward their mission goals, react swiftly to crises and adapt dynamically to events. In the Electric Elves project, humans will be agentified by providing them with proxies to act on their behalf, while entities such as meeting schedulers will be active agents that can communicate with the proxies. For instance, the Electric Elves should enable us to schedule demonstrations outside our institute, determining which project should be demonstrated, which person in the project should travel to give the demonstration, which project members should provide support to the demonstration, schedule the shipping and packaging of equipment, etc.

As an agent integration architecture we are using the TEAMCORE framework [9]. Human developers specify a team-oriented program in a tool called TOPI. A team program consists of a high-level specification of a hierarchical team plan. High-level team plans typically decompose into other team plans and, ultimately, into leaf-level plans that are executed by individuals. For example, the team plan for a meeting is decomposed into a team plan to find appropriate participants for the meeting, e.g. presenters and attendees, and individual plans for attending the meeting. Each TEAMCORE proxy represents a single human user or active system. Proxies form teams that execute the team plans autonomously, coordinating amongst themselves as required.

3 Layered Adjustable Autonomy

Our layered approach to AA parallels the layers of autonomy that occur within human organizations. The local layer of the AA system reflects an individual's ability to make their own decisions while the team level reflects the social basis of team autonomy. It appears that the layering leads to benefits in terms of information presentation and the ability to provide effective mechanisms for user control when it comes to implementing the AA system.

Local autonomy is the autonomy an agent has to make local decisions. For example, in the Electric Elves domain a decision to volunteer for some role is a local decision that may be made by either the human or their agent proxy. The degree of autonomy the proxy has is the decisions which it can make on its own and those for which it's human counterpart need be consulted. Note that the agents local decision may be overridden by decisions of the agent's team – e.g. simply because a proxy volunteers does not guarantee they will be assigned the role.

Team autonomy is the autonomy an agent team has to make it's own decisions. A fully autonomous team could negotiate a team decision for, say a visitor schedule, based on the preferences of the agents in the team without being dependent on human authorization. In a less autonomous agent team the human team would have the opportunity to veto decisions made by an agent team before a commitment is made to the decision.

The team level AA is required in addition to the individual layer because a team decision may be undesirable despite the individual decisions of all its members being correct. For example, consider the case of only two proxies, A and B, volunteering for two available roles. The team may now decide that the users A and B will be giving a demonstration based on the wishes of the team members proxies. However, both A and B may be junior project members and a head of department may wish to have someone more senior participate. Hence the head of department will override the team decision but not necessarily that of A and B to volunteer. Indeed, that A and B volunteered is to their credit. Notice that the team autonomy is completely independent of the individuals autonomy, i.e. the autonomy the team has to make a decision is independent of whether any individuals decisions came autonomously from the proxies or directly from the human users. Furthermore it is interesting to note that the reasons for the team decision are independent of the reasons for the decisions of any of the team members.

While inspiration for the layering of adjustable autonomy comes from the layering of autonomy in human organizations it turns out that a variety of technical advantages are gained by layering an AA tool. In a large complex multi-agent system there is far too much going on for a human to take everything that the multi-agent system knows into account when making a decision. Hence, an issue for AA tools for multi-agent systems is how to find and present the information that is relevant to a particular situation. The layering approach simplifies this task. Because local AA systems take no account of the team perspective local AA tools need only be concerned with information about the local agent. Conversely, the team AA tools can focus on presenting more abstract, team-oriented information, such as the important relationships between agents and the status of team plans.

The task of providing mechanisms for enforcing a human's decision is also simplified by the layered approach. The individual AA tool only needs to interact with the local proxy and the team AA tool only needs to interact with the team structure, roles and relationships. Hence the mechanisms can be more focused and, therefore, easier to build.

3.1 Local Adjustable Autonomy

Figure 2: A snapshot of a user-agent's dialogue box. At the top of the window is a description of the reason for the repair. The middle of the window shows the available repair options. Finally, at the bottom is a check-box for the user to indicate whether the proxy should ask for confirmation is situations like this.

A key challenge in integrating heterogeneous (human) agents is that humans may have differing requirements with respect to the autonomy their local proxies have to make decisions on their behalf. In the Electric Elves domain, proxies for humans can suggest coordination and repair decisions autonomously to aid in team activities such as meetings. For example, an agent proxy will suggest repairs for the team plan for a scheduled meeting when it notices that its human counterpart will be late. What to do about the failing plan differs from person to person and situation to situation, hence the human may or may not want the proxy to autonomously make the decision. Conversely, restricting the proxy to always confirm it's decision with the user is also undesirable, since it would end up overwhelming the user with confirmation requests for trivial decisions.

By watching the user the proxy can learn to predict which repair option the human usually selects and also learn whether the human allows the proxy to act autonomously in a particular situation. I.e. after a proxy selects a repair (e.g. delay 5 minutes) it should also decide whether to confirm its selection with the user. The pair of choice of repair and whether to confirm the choice may vary person to person. To learn these two decisions dimensions we use two learning processes for each proxy: one to predict a choice of repair option and one to predict which decisions to take autonomously.

We use a supervised learning approach(C4.5) using user feedback for each decision. Eleven attributes are used in learning which repair to use and twelve attributes are used for learning whether to consult the user before making the decision. Attributes include personal information (e.g. office location, seniority), team plan information (e.g. meeting location, time, number of attendees), monitored information about user's current location and the proxies predicted decision. These attributes are extracted from the user's schedule, organizational charts, etc. In the training phase, a proxy suggests a repair and queries whether the user would wish it to make such a decision autonomously (see the snapshot of the proxy's dialogue box in Figure 2).

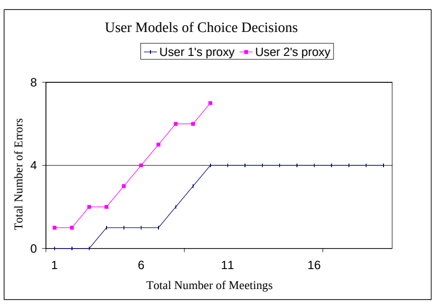

Figure 3: This graph shows cumulative number of errors the proxy made in selecting a repair option. Notice that after 10 meetings there were no more errors for User 1 while User 2's proxy continued to make errors.

We have experimented with the local AA system in a real environment, using real meeting schedules, with 5 agentified humans, 2 palm pilots and a GPS systems for 2 weeks. Figure 3 and Figure 4 are the test results from two of the more experienced users (20 data points for User 1, 10 data points for User 2). C4.5 is continuously learning from new data and the learned rules are applied immediately to new predictions.

Figure 3 shows that User 1's proxy is learning to predict choices very well after only 10 meetings. User 2's proxy is continuing to make errors. The reason seems to be that User 2 tends to choose different repairs for each meeting (i.e., ”delay 5 minutes” for one meeting, ”cancel it” for another, ”user is attending” or ”user will not attend” for yet others). Given the results in Figure 3 it is not surprising that Figure 4 shows User 1 giving full autonomy to its agent while User 2 stays in complete control. User 1 seems satisfied with its proxy's learning process so gives complete autonomy to it's proxy, while User 2 seems suspicious of the predictions it's proxy made hence gave no autonomy to the proxy.

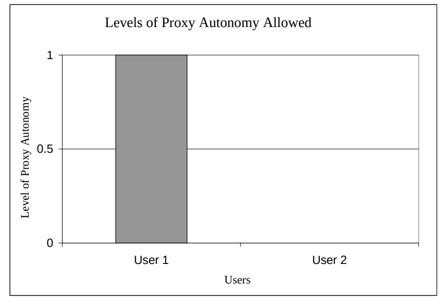

Figure 4: This graph shows the amount of autonomy each user gave to it's proxy.

3.2 Adjustable Autonomy for Teams

Team level AA refers to the ability to change the autonomy that a team has to make decisions without being dependent on human authorization. The team autonomy is independent of the the autonomy any individual agent has to make its own decisions. A tool for team AA, therefore, must allow monitoring and interacting with the team as a whole. Initially our aim is to provide functionality that allows a user to change agent role assignments of agents online. Role changes may be required either because of failures by particular agents or simply because a user decides that a different agent should be assigned a particular role. The team programming tool TOPI has been extended with support for the AA capabilities (see Figure 5).

For a user to respond effectively to agent failures the AA system needs to provide information not only about the agent that failed but the overall team state, any team response to that failure, implications of the failure, agents available to fulfill the failing agents role, etc. This type of information is generally not explicitly available. To understand the current state of the multi-agent system, TOPI relies on plan recognition which infers the state of the team members from the coordination messages normally transmitted during execution. The plan-recognition-based method is non-intrusive, avoiding the overhead of the proxies having to continually communicate their state to the AA tool. TOPI reasons about the current state of the system and, using its knowledge of the team plan, knowledge of agent abilities acquired from an agent resource manager and built-in knowledge of teamwork, explains to the user the implications of failures. In the future TOPI will be able to use similar mechanisms to explain to a user the implications of role changes or changes to team plans.

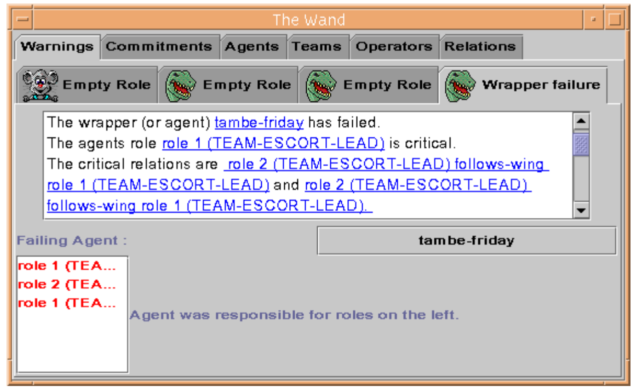

Figure 5: A screen shot of the AA aspect of TOPI. The figure shows warnings that have been brought to the user's attention. Each of the tabs along the top of the window link the user to general information about the current status of the team. The severity of the warning, as determined by TOPI, is reflected in the size of the (real) animal displayed on the warning tab. The hypertext explanation in the middle of the screen allows the user to quickly jump to related information about the failure. The bottom of the screen supplies more detail about the reason for the warning.

4 Conclusion

When agents work within a human organization human intervention will sometimes be required. Layering is a novel approach to the task of designing effective AA tools for multi-agent systems. Barber [1] also looks at AA from with respect to agents working together. However the AA in their system is peer to peer not from a human to a team. Deters [3] describes a AA system for multi-agent systems which relies on agents reporting their state to a central database. Dellarocas [2] looks at automated failure detection and repair for multi-agent systems. Both Deters and Dellarocas focus solely on the team level without providing mechanisms for individual agent AA. Our layered approach to Adjustable Autonomy, which parallels the layers of autonomy found in human organizations, allows users to intervene either with an individual agent or with the team as a whole. The layered approach separates out the important issues that are relevant at each layer of abstraction allowing development of effective tools for users.

References

[1] K. S. Barber and C. E. Martin. Agent autonomy: Specification, measurement, and dynamic adjustment. In *Autonomy Control Software Workshop, Autonomous Agents 99*, 1999.

[2] C. Dellarocas and M. Klein. An experimental evaluation of domain-independant fault handling services in open multi-agent systems. In *Proceedings of the International Conference on Multi-Agent Systems (ICMAS'2000)*, 2000.

[3] R. Deters. Adjustable autonomy in a large distributed agent-based system. In *Autonomy Control Software Workshop, Autonomous Agents 99*, 1999.

[4] J. Hendler and R. Metzeger. Putting it all together – the control of agent-based systems program. *IEEE Intelligent Systems and their applications*, 14, March 1999.

[5] M. N. Huhns and M. P. Singh. All agents are not created equal. *IEEE Internet Computing*, 2:94–96, 1998.

[6] N. Jennings. Agent-based computing: Promise and perils. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99)*, August 1999.

[7] J. R. Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[8] P. Langley S. Rogers, C Fiechter. An adaptive interactive agent for route advice. 1999.

[9] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research (JAIR)*, 7:83–124, 1997.