Teamwork in Cyberspace: Using TEAMCORE to Make Agents Team-Ready

Milind Tambe, Wei-Min Shen, Maja Mataric, David V. Pynadath
Dani Goldberg, Pragnesh Jay Modi, Zhun Qiu, Behnam Salemi

Information Sciences Institute and Computer Science Department
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
team@isi.edu

Introduction

In complex, dynamic and uncertain environments extending from disaster rescue missions, to future battlefields, to monitoring and surveillance tasks, to virtual training environments, to future robotic space missions, intelligent agents will play a key role in information gathering and filtering, as well as in task planning and execution. Although physically distributed on a variety of platforms, these agents will interact with information sources, network facilities, and other agents via cyberspace, in the form of the Internet, Intranet, the secure defense communication network, or other forms of cyberspace. Indeed, it now appears well accepted that cyberspace will be (if it is not already) populated by a vast number of such distributed, individual agents.

Thus, a new distributed model of agent development has begun to emerge. In particular, when faced with a new task, this model prescribes working with a distributed set of agents rather than building a centralized, large-scale, monolithic individual agent. A centralized approach suffers from problems in robustness (due to a single point of failure), exhibits a lack of modularity (as a single monolithic system), suffers from difficulty in scalability (by not utilizing existing agents as components), and is often a mismatch with the distributed ground reality. The distributed approach addresses these weaknesses of the centralized approach. Our hypothesis is that the key to the success of such a distributed approach is *teamwork in cyberspace*. That is, multiple distributed agents must collaborate in teams in cyberspace so as to scale up to the complexities of the complex and dynamic environments mentioned earlier. For instance, consider an application such as monitoring traffic violators in a city. Ideally, we wish to be able to construct a suitable agent-team quickly, from existing agents that can control UAVs (Unmanned Air Vehicles), an existing 3D route-planning agent, and an agent capable of recognizing traffic violations based on a video input. Furthermore, by suitable substitution, we wish to be able to quickly reconfigure the team to monitor enemy activity on a battlefield or illegal poaching in forests. Such rapid agent-team assembly obviates the need to construct a monolithic agent for each new application from scratch, preserves modularity, and appears better suited for scalability.

Of course, such agent teamwork in cyberspace raises a variety of important challenges. In particular, agents must engage in robust and flexible teamwork to overcome the uncertainties in their environment. They must also adapt by learning from past failures. Unfortunately, currently, constructing robust, flexible and adaptive agent teams is extremely difficult. Current approaches to teamwork suffer from a lack of general-purpose teamwork models, which would enable agents to autonomously reason about teamwork or communication and coordination in teamwork and to improve the team performance by learning at the team level. The absence of such teamwork models gives rise to four types of problems. First, team construction becomes highly labor-intensive. In particular, since agents cannot autonomously reason about coordination, human developers have to provide them with large numbers of domain-specific coordination and communication plans. These domain-specific plans are not reusable, and must be developed anew for each new domain. Second, teams suffer from inflexibility. In real-world domains, teams face a variety of uncertainties, such as a team member's unanticipated failure in fulfilling responsibilities, team members' divergent beliefs about their environment [CL91], and unexpectedly noisy or faulty communication. Without a teamwork model, it is difficult to anticipate and preplan for the vast number of coordination failures possible due to such uncertainties, leading to inflexibility. A third problem arises in team scale-up. Since creating even small-scale teams is difficult, scaling up to larger ones is even harder. Finally, since agents cannot reason about teamwork, learning about teamwork has also proved to be problematic. Thus, even after repeating a failure, teams are often unable to avoid it in the future.

To remedy this situation and to enable rapid development of agent teams, we are developing a novel software system called TEAMCORE that integrates a general-purpose teamwork model and team learning capabilities.

TEAMCORE provides these core teamwork capabilities to individual agents, i.e., it wraps them with TEAMCORE. Here, we call the individual TEAMCORE "wrapper" a *teamcore agent*. A teamcore agent is a pure "social agent", in that it is provided with only core teamwork capabilities. Given an existing agent with domain-level action capabilities (i.e., the domain-level agent), it is made *team-ready* by interfacing with a teamcore agent. Agents made team-ready will be able to rapidly assemble themselves into a team in any given domain. That is, unlike past approaches such as the open-agent-architecture (OAA) that provides a *centralized* blackboard facilitator to integrate a distributed set of agents, TEAMCORE is fundamentally a distributed team-oriented system.

Our goal is a TEAMCORE system capable of generating teams that are:

1. Flexible and robust, able to surmount the uncertainties mentioned above.
2. Capable of scale-up to hundreds of team members
3. Able to improve the team performance by learning at the team level and avoiding past team failures.

An initial version of TEAMCORE system based on the Soar [Newell90] integrated agent architecture is currently up and running. A distributed set of teamcore agents can form teams in cyberspace. The underlying communication infrastructure is currently based on KQML. The rest of this document now briefly describes the TEAMCORE design, architecture and implementation.

TEAMCORE Design and Architecture

Figure 1 illustrates the basics of the TEAMCORE design. It shows existing agents with domain-level expertise, wrapped by teamcore agents. The communication between teamcore agents themselves is based on KQML. KQML is also used in communication with the domain-level agents, but other techniques may also be used. For instance, one domain-level agent may be a route-planner, another a UAV (Unmanned Air Vehicle), and yet another a weather-query service. The teamcore agents wrap these three individual agents to create a team. The teamcore agents themselves do not possess the domain-level capabilities of the individual domain-level agents (such as the capability to take the relevant actions to fly a UAV). However, they provide the necessary teamwork expertise.

Teamcore agents' teamwork expertise can be categorized into two components, a domain-specific and a domain-independent component, the latter forming the central part of teamcore. The domain-specific part consists of domain-specific team reactive plans. These plans define the role of the teamcore agent in the team and are provided by a user or another teamcore agent. While these team reactive plans are much like situated-plans or reactive-plans for individual agents [Firby 87], the key difference is that they explicitly express joint team activities of the relevant team. In the above example, the team reactive plans will refer to the team actions of the UAV, route-planner and weather-query service. These team reactive plans essentially ensure that all teamcore agents know the overall team procedure. This team procedure may be to execute a team activity, to plan a team activity, to collaboratively design an artifact, to collaboratively schedule or to collaboratively monitor and diagnose. Having common knowledge of team procedures is akin to providing the team with the knowledge of "standard-operating-procedures" in a military setting.

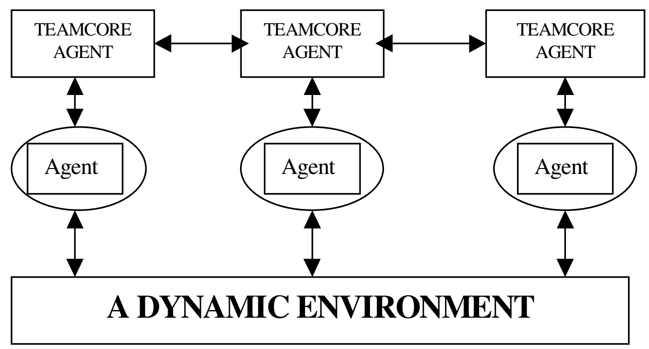

Figure 1: The TEAMCORE Architecture

Common knowledge of team procedures obviously does not protect the team members from incomplete or incorrect beliefs about the environment or other team members, from unexpected failures and successes, unexpected communication failures and so on. The domain-independent component of teamcore is responsible for surmounting such uncertainties of dynamic complex domains and adapting to the current environment. The heart of the domain-independent component is teamwork reasoning based on a general model of teamwork. This teamwork model encodes common-sense teamwork knowledge in the form of domain-independent axioms, that explicitly specify a team member's responsibilities and commitments in teamwork. The model enables individual agents to autonomously reason about coordination and communication in teamwork, and thus provides significant teamwork flexibility and robustness. In building this model, we have built on previous work in theories of teamwork, in particular, the Joint Intentions [LeveCN90] and SharedPlans [GrosK96] theories, as well as our experience in building agent teams for a variety of domains [Tambmany98][Tamb97bjair][Dreamteam98a, b]. This practical experience has led us to integrate novel components into the teamwork model, such as a decision-theoretic component to enable selective inter-agent communication.

The teamwork model in TEAMCORE also includes CONSA (a collaborative negotiation system based on argumentation) [QT98a, QT98b]. Due to agents' incomplete information or different interpretation of information, conflicting beliefs and plans often arise in teamwork. CONSA addresses the problems of negotiation to resolve these conflicts, based on an argumentation pattern proposed in [Toulmin58]. The argumentation process in CONSA involves a proposal/evaluate/counter-proposal cycle between a sender and a receiver, and generation and refutation of these proposals are accomplished by several "argumentation moves" such as *rebut*, *undercut* and *improve-support* (which is required for CONSA's collaborative aspect). CONSA is embedded in TEAMCORE's teamwork model and exploits the latter's domain independent teamwork knowledge to enable generalized and reusable argumentation knowledge that agents can use in negotiation. CONSA is also collaborative, so it requires agents to detect real conflicts where negotiation will improve team performance, which is based on comparing different strengths of two contradicting beliefs between teammates. To address practical issues due to TEAMCORE's dynamic, complex environment, CONSA uses a decision-theoretic approach to evaluate costs and utilities of negotiation before deciding to start, and a *pruning* process before communicating proposals to reduce communication overhead.

The domain-independent component of TEAMCORE also includes a role-based team learning capability based on organizational adaptation for better team performance. In order to accomplish tasks and missions that require collaborative effort in a real-world situation that is dynamic and uncertain, a group of individual agents must adapt and learn as a team. Although the notion of "team learning" clearly implies more than a collection of individual learning agents, the precise difference between team learning and single-agent learning is not clearly defined.

In pursuing our objective to build adaptive agent teams, we have analyzed a number of previous organizational learning architectures in different domains, and proposed a definition of team learning based on the notion of "roles" and "assignments". A role is defined as responsibilities in service of a team plan, and an assignment assigns members of a team to the roles. In the teamcore architecture, the domain-specific component of a teamcore agent represents its role. Each domain-level agent, depending on its capability, is assigned to a teamcore agent and there might be several types of role relationships among teamcore agents. For example, in a team of leader-follower formation flight of two URAVs (Unmanned Reconnaissance Air Vehicles), the role of the leader may be specified as route planning, while the role of the follower is specified as following the leader. With these two roles, there are two possible role assignments for the two URAVs.

Given these concepts, team learning can be performed in at least four aspects to continually improve the team performance: (1) to improve an individual's ability to perform a role; (2) to change role assignment; (3) to change roles themselves (such as create, delete, modify, combine, or split roles); and (4) to modify the team organization by changing the relationship between roles. In our two URAV example, the first two aspects are obvious. For the third aspect, the team may learn to modify the leader's role so as to avoid the foggy areas where the follower cannot see the leader. For the fourth aspect, the team may learn to change the distance maintained between two URAVs. In contrast, non-team learning (or single-agent learning) covers only the first of the four aspects of team learning.

To operationalize this definition, we have developed a new team learning framework called ROBOL (ROle Based Organizational Learning) [SS98] based on our previous work on autonomous learning [Shen93,Shen94,WS97], multi-robot systems [Dreamteam98a, Dreamteam98b], and the concept of a Cooperation Structure proposed by d'Inverno, Luck and Wooldridge [dLW97]. Each teamcore agent maintains and modifies a Partial Organizational Structure (POS). A POS is an acyclic graph where nodes are roles and edges are the relationship between roles. Each node is also labeled with a domain-level agent identifier to specify the role assignment. A POS graph is "partial" in the sense that it may only provide a local view of the team activities. In the context of a team, each teamcore agent uses its own POS to perform its tasks. Based on the feedback at the team level, all teamcore agents will modify their own POS such that the composition of all POSs will converge towards a better performance at the team level. The novel aspect of POS is that it separates roles from domain-level agents and allows richer relationships between roles. Furthermore, it uses team performance as a measurement of the quality of the POS and permits adaptation in a distributed fashion. To address the problem of how a set of POSs can be modified to produce a better global measurement, we are also investigating Pareto Rationality [WS97] as a negotiation strategy.

We are currently implementing this POS framework and testing it on three domains: manufacturing scheduling where agents may dynamically change their roles according to the output of an assembly line, marching-band formation where agents have to learn to establish suitable role assignments and role relationships in order to improve the quality of the formation, and helicopter engage-attack where agents must synchronize their activities. Limited applications are also considered in the RoboCup [MatsNH96] domain, where a team of agents must perform in a highly dynamic, uncertain, and adversarial environment.

Some key teamwork reasoning capabilities provided by the teamwork model include the following:

1. Enable agents to responsibly commit to a given team goal.

2. Enable agents to learn how to reorganize their team to improve the team performance. For example, if an individual team member is unable to perform its role in the team thereby disabling the team from achieving its current goal, then other agents may reorganize the team to compensate for this failure.

3. Enable agents to share relevant information with others, where relevance is determined based on the current joint goal being performed.

4. Enable agents to reason about selective communication with others, where selectivity is determined via a decision-theoretic reasoning component.

In previous work, we have shown applications of an earlier version of this teamwork model called STEAM [Tambe97a, TAAEKMM98]. STEAM was successfully used in several applications --- STEAM-based synthetic pilot-agent teams have successfully participated in DARPA's synthetic theater of war (STOW) demonstrations in 1997 [Tambe97b] and it was also reused in developing our award-winning team for the RoboCup'97 international multi-agent soccer tournament [Tambemany98].

The TEAMCORE project is building on STEAM's earlier successes, extending it in several ways. First, in our earlier implementation, STEAM was integrated within an agent's knowledge base. In contrast, TEAMCORE separates out the teamwork knowledge into a separate teamcore agent. This teamcore agent is currently capable of communicating via KQML (a capability not available within STEAM). Second, novel capabilities such as the negotiation capability (CONSA) have been added to TEAMCORE's teamwork model. Finally, teamcore agents will be provided with novel role based team learning capabilities.

Summary and Current Status

The potential contributions of the TEAMCORE project for agent teamwork in cyberspace, both in the military and commercial arenas, are very significant. The goal of TEAMCORE is to make agents in cyberspace "team-ready", by wrapping them with teamcore agents. Thus, TEAMCORE could rapidly develop flexible, robust and adaptive agent teams for information gathering and filtering, as well as mission planning and execution. For instance, it would enable rapid creation of robust and flexible teams from agents aboard UAVs (Unmanned Air Vehicles), satellites and persistent stores, route-planners and other relevant agents for rapid information gathering and sharing, for applications such as traffic monitoring, or battlefield monitoring. Agent teams could also be assembled for distributed planning or mixed-initiative planning. Teamwork among synthetic forces in battlefield simulations would improve as well, improving training and analysis. Furthermore, TEAMCORE's contributions could extend to a range of agent-based systems, from agent-teams for air-traffic control to teams of spacecraft.

The current Unix implementation of the teamcore agents includes an extended version of the STEAM rules in the Soar integrated architecture, as well as the KQML infrastructure for communication. The teamcore agents form teams, make joint commitments, and maintain coherent group beliefs, all through the teamwork reasoning mentioned above. For instance, when wrapping pilot agents flying helicopters in a ModSAF simulation environment, the teamcore agents decide upon a joint mission and then provide individual task information to each pilot agent. The helicopters can thus perform team tasks with only minimal alteration to the original agent code. In addition, the required alteration concerns only the teamcore communication interface, which builds on domain-independent knowledge of tasking and monitoring. The end system demonstrates TEAMCORE's ability to support flexible team assembly in a dynamic environment.

The implementation of ROBOL is currently underway. In particular, we have built a Java-based system that allows flexible adaptations of roles, role assignments, and role relationships. It will interface with multiple environments such as a Java simulation environment or ModSAF. We will test this implementation in several example domains, including the adaptive marching example and helicopter missions in ModSAF. Furthermore, we are also implementing the learning component of this framework to change Soar's production rules.

Planned extensions to the current TEAMCORE system also include the integration of a novel bottom-up data-driven modeling approach using augmented Markov models (AMMs) [Goldberg & Mataric 1998]. The approach can learn models of the interaction dynamics between an agent and its environment, and may be employed to achieve various levels of coordination among the members of a team. Applications of the model include monitoring individual performance, group performance, and characterizing the experience and abilities of team members. Models are generated on-line and in real-time with little computational and space overhead, thus conforming to limitations that may exist in many practical team-based scenarios. This approach builds on previous work exploring tradeoffs between various robot team structures for performing a task [Goldberg & Mataric 1997].

References

[BarbF95] M.Barbuceanu and M.Fox. The architecture of an agent building shell. In M.Wooldridge, J.Muller, and M.Tambe, editors, Intelligent Agents, Volume II: Lecture Notes in Artificial Intelligence 1037. Springer-Verlag, Heidelberg, Germany, 1996.

[CaldSCMC93] R.B. Calder, J.E. Smith, A.J. Courtemanche, J.M.F. Mar, and A.Z. Ceranowicz. Modsaf behavior simulation and control. In Proceedings of the Conference on Computer Generated Forces and Behavioral Representation, 1993.

[CL91] P.R. Cohen and H.J. Levesque. Teamwork. Nous, 35, 1991.

[dLW97] d'Inverno, Luck and Wooldridge, Cooperation Structure. In the Proceedings of IJCAI, 1997.

[Dreamteam98a] Shen, W.M., J. Adibi, R. Adobbati, B. Cho, A. Erdem. H. Moradi, B. Salemi, and S. Tejada. Building Integrated Mobile Robots for Soccer Competition. In Proceedings of International Conference on Robotics and Automation. Leuven, Belgium. May 1998.

[Dreamteam98b] Shen, W.M., J. Adibi, R. Adobbati, B. Cho, A. Erdem. H. Moradi, B. Salemi, and S. Tejada. Toward Integrated Soccer Robot Team. AI Magazine. Fall, 1998.

[DurfL91] E.Durfee and V.Lesser. Partial global planning: a coordination framework for distributed planning. IEEE transactions on Systems, Man and Cybernetics, 21(5), 1991.

[Firby 87] J.Firby. An investigation into reactive planning in complex domains. I *proceedings of the National Conference on Artificial Intelligence* (AAAI), 1987.

[Goldberg & Mataric 1998] Dani Goldberg and Maja J Mataric, Coordinating Mobile Robot Group Behavior Using a Model of Interaction Dynamics. Submitted to Autonomous Agents '99.

[Goldberg & Mataric 1997] Dani Goldberg and Maja J Mataric, Interference as a Tool for Designing and Evaluating Multi-Robot Controllers, Proceedings, AAAI-97, Providence, Rhode Island, July 27-31, 1997, 637-642.

[Gros96] B.Grosz. Collaborating systems. AI magazine, 17(2), 1996.

[GrosK96] B.Grosz and S.Kraus. Collaborative plans for complex group actions. Artificial Intelligence, 86:269--358, 1996.

[Jenn94] N.Jennings. Commitments and conventions: the foundation of coordination in multi-agent systems. The Knowledge Engineering Review, 8, 1994.

[Jenn95] N.Jennings. Controlling cooperative problem solving in industrial multi-agent systems using joint intentions. Artificial Intelligence, 75, 1995.

[LeveCN90] H.J. Levesque, P.R. Cohen, and J.Nunes. On acting together. In Proceedings of the National Conference on Artificial Intelligence. Menlo Park, Calif.: AAAI press, 1990.

[MatsNH96] H.Matsubara, I.Noda, and K.Hiraki. Learning of cooperative actions in multi-agent systems: a case study of pass play in soccer. In S.Sen, editor, AAAI Spring Symposium on Adaptation, Coevolution and Learning in multi-agent systems, March 1996.

[Newell90] Newell, A. Unified Theories of Cognition. Harvard Univ. Press, 1990.

[QT98a] Qiu, Z. and Tambe, M. Flexible negotiations in teamwork: extended abstract. In M. desJardine (editor), AAAI FALL Symposium on Distributed Continual Planning. October, 1998.

[QT98b] Qiu, Z. and Tambe, M. Towards Argumentation-based Collaborative Negotiation: A preliminary report. In International workshop on multi-agent systems, Mass. Institute of Technology, October, 1998.

[RoseZ94] J.S. Rosenschein and G.Zlotkin. Designing conventions for automated negotiation. AI Magazine, 15, 1994.

[Sen96] S.Sen. Proceedings of the Spring Symposium on Adaptation, Coevolution and Learning. American Association for Artificial Intelligence, Menlo Park, CA, 1996.

[Shen93] W.M. Shen. Discovery as autonomous learning from the environment. Machine Learning, 12:143--165, 1993.

[Shen94] W.M. Shen. Autonomous Learning from the Environment. W. H. Freeman, Computer Science Press, 1994.

[SS98] Salemi, B and Shen, W.M. Role-Based Organizational Learning for Agent Teams. In preparation. November, 1998.

[Tamb96b] M.Tambe. Teamwork in real-world, dynamic environments.

In Proceedings of the International Conference on Multi-agent Systems (ICMAS), December 1996.

[Tamb97a] M.Tambe. Agent architectures for flexible, practical teamwork. In Proceedings of the National

Conference on Artificial Intelligence (AAAI), August 1997.

[Tamb97bjair] M.Tambe. Towards flexible teamwork. Journal of Artificial Intelligence Research (JAIR), 7:83--124, 1997.

[Tambmany98] M.Tambe, J.Adibi, Y.Alonaizon, A.Erdem, G.Kaminka, S.Marsella, I.Muslea, and M.Tallis. Isis: Using an explicit model of teamwork in robocup97. In RoboCup'97: Proceedings of the first robot world cup competition and conferences. Springer-Verlag, Heidelberg, Germany, 1998.

[TambJS97] M.Tambe, W.L. Johnson, and W. Shen. Adaptive agent tracking in real-world multi-agent domains: a preliminary report. International Journal of Human-Computer Studies (IJHCS), 1997.

[TAAEKMM98] Tambe, M., Adibi, J., Alonaizon, Y., Erdem, A., Kaminka, G., Marsella, S. and Muslea, I. Building agent teams using an explicit teamwork model and learning. Artificial Intelligence (to appear), 1998.

[Toulmin58] Toulmin, S. 1958. *The uses of argument*. Cambridge: Cambridge University Press.

[WS97] Wang, X.J and Shen, W.M., Coordination via Negotiation using Pareto Rationality. Poster in the International Joint Conference on Artificial Intelligence, 1997.