Collaborative RPD-Enabled Agents Assisting the Three-Block Challenge in Command and Control in Complex and Urban Terrain

Xiaocong Fan[1], Shuang Sun[1], Bingjun Sun[2], Guruprasad Airy[2], Michael McNeese[1], John Yen[1]

[1]School of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802
{zfan, ssun, mmcneese, jyen}@ist.psu.edu

[2]Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
{bsun, airy}@cse.psu.edu

Timothy Hanratty, John Dumer
[3]US Army Research Laboratory
Aberdeen Proving Ground, Maryland
{hanratty, dumer}@arl.army.mil

Keywords:
Collaborative Agents, Recognition-Primed Decision Making, Human-Centered Teamwork, Contexts

ABSTRACT: *One of the challenging issues in the domain of C2 in complex and urban terrain is how to assist human combat staff to effectively collaborate and make decisions under multiple contexts, and to help people switch their attentions to the most urgent decision tasks. This paper describes a comprehensive approach that uses RPD-enabled agents to support timely decision-making under multiple contexts. We introduce the R-CAST decision support system and report our effort of using the R-CAST agent architecture to assist human combat staff in dealing with information challenges in complex and urban terrain. The simulation demonstrates that by modeling the information requirements of the three block challenge as relevant cues and expectancies, R-CAST agents can effectively share relevant information for complex decision situations.*

1. Introduction

The information age has brought dramatic and challenging changes in contemporary combat operations. The term associated with this change is Network-Centric Warfare (NCW). From an information technology perspective, NCW is defined as an information superiority-enabled concept of operations that generates increased combat power by networking sensors, decision-makers, and shooters to achieve shared awareness, increased speed of command, higher tempo of operations, greater lethality, increased survivability, and a degree of self synchronization. In essence, NCW translates information superiority into combat power by effectively linking knowledgeable entities in the battlespace (Alberts, 2002).

The focus of this research is targeted at coupling software agents and teamwork approaches towards the informational challenges associated with C2 in complex and urban terrain (C2CUT). Urban combat zones are complex and dynamic and there is a critical need for gathering and sharing intelligence information. The informational needs associated with such C2 environments are indistinct, unstructured, and often can cause information overload or deficiencies.

To study the information sharing challenges in C2CUT we have created a simulated scenario called *three block challenge*: within three-block area in a city officers in command must react to a constant flow of intelligence reports and make timely decisions for three kinds of tasks including combat, peacekeeping and humanitarian missions. The idea of conducting peacekeeping, humanitarian, and combat missions in close proximity and simultaneously is a new way of conducting operations. Each mission can have an affect on the other. The ability of staff officers to monitor these situations and react in a timely manner is the key to success.

This issue is challenging because it requires effective team collaborations to establish shared situation awareness, to rapidly link dynamic information from multiple sources for assessing potential threats, for identifying areas of interest, and for choosing optimal corridors for movement in different contexts (e.g., officers should react differently to combat, peacekeeping and humanitarian missions).

To assist human combat staff in dealing with the three-block challenge, we use R-CAST, a collaborative agent architecture that is extended from the CAST architecture (Collaborative Agents for Simulating Teamwork) (Yen, et al., 2005) by incorporating a computational Recognition-Primed Decision (RPD) model (Klein, 1993) for supporting experience-based decision-making. The remainder is organized as follows. Section 2 reviews recent studies on RPD; Section 3 introduces the R-CAST architecture; Section 4 describes the using of R-CAST to assist the combat, peacekeeping and humanitarian missions; comparison and discussion is given in Section 5 and Section 6 summarizes the paper.

2. Towards Collaborative RPD

The RPD model (Klein, 1993) captures how domain experts make decisions based on the recognition of similarity between the current situation and past experiences. The RPD process has two phases: recognition and evaluation. In recognition phase, a decision maker needs to develop situation awareness and recognize what course of actions (COA) worked before in a similar situation. In evaluation phase, a decision maker needs to evaluate each COA by imaging how it will evolve. If a COA does not work for the current situation, the decision maker can either adjust it, or find and examine other COAs until a workable solution is obtained.

The RPD model states that "feature-matching" and "story-building" are two typical strategies used by experts to develop situation awareness. In feature-matching, a decision maker tries to find whether he/she has ever experienced situations similar to the current one by matching the set of observed cues (synthesized from information describing the current situation) with the pattern of cues considered in past experiences. In case that feature-matching cannot produce an adequate account for the current situation due to lack of experience, story-building will be used to construct an explanation, by coherently linking the observed information. A story gives an explanation of how the current situation might have been emerging. When building a story, a decision maker need explore potential hypotheses and evaluate how well each of them may fit what have been observed.

Recognition results in four products: relevant cues (what to pay attention to), plausible goals (which goals make sense), expectancy (what will happen next), and course of actions (what actions worked in this type of situation). An expectancy serves as a gate-condition for continuing working on the current recognition. Due to the dynamic and uncertain nature of the environment, it is important to monitor the status of expectancies because a decision maker may have misinterpreted the current situation but he/she cannot recognize it until some expectancy is invalidated as the situation further evolves. In such cases, the decision maker needs to further diagnose the current situation (e.g., to gather more information).

There have been several attempts in implementing the RPD model. For example, long-term memory structure (Warwick, et al., 2001) and neural networks (Liang, et al., 2001) were used to represent experiences. Fuzzy techniques were used (Robichaud, 2002) to incorporate a fuzzy interpretation of the external environment. While the abovementioned approaches only implemented the feature-matching strategy, the Navy DSS system (Morrison, et al., 1996) did allow a decision maker to build alternative stories. There are also attempts in integrating RPD with agent technologies (Norling, et al., 2000; Sokolowski, 2003). For example, Norling, et al. (2000) explored the ways of using RPD to enhance BDI agents so that the simulations of human societies would be more realistic. These attempts are limited in that the evaluation phase of RPD is ignored (or assumed to be done by human) and the RPD model is considered as an internal process of individual agents. The second limitation actually leaves the most exciting part of RPD as an open research issue: how a team of agents, with a shared computational RPD process, are supposed to work together in collaboratively developing situation awareness, in effectively anticipating others' information needs relevant to cues and expectancies, and in proactively sharing information to make better decisions under time pressure.

The RPD model captures the cognitive activity undergoing in the mind of a decision maker when he/she faces a decision task. In essence, RPD is an individual process because it is within a decision maker's mental state. However, it becomes more interesting when a team of human experts, each making decisions using RPD, needs to work together in distributed dynamic environments. Intuitively, team performance can be considerably enhanced if the team can establish a shared mental model about the dynamic progress of the RPD process being pursued by the decision maker. Emphasizing the individual nature of the RPD process may weaken or ignore the active roles played by the other teammates in the process, especially from the information seeking and sharing perspective.

On the other hand, as domain complexity increases, decision making often involves various kinds of expertise and experiences, which are typically distributed among a group of decision makers. In such cases, the timeliness and quality of decision making highly depend on the effectiveness of team wide collaboration (e.g., anticipating others' information needs, proactive sharing information and expertise).

Thus, in our R-CAST model described below, we consider the situations where a group of people who are experts in different areas, each assisted by one RPD-enabled agent, face the pressure to make better and faster decisions in an environment with high domain complexities. In such a setting, collaboration may exhibit among RPD-agents, between an RPD-agent and its human partner, and among the human experts.

3. R-CAST: RPD-enabled Agent Architecture

Figure 1 shows an abstract view of the R-CAST agent architecture.

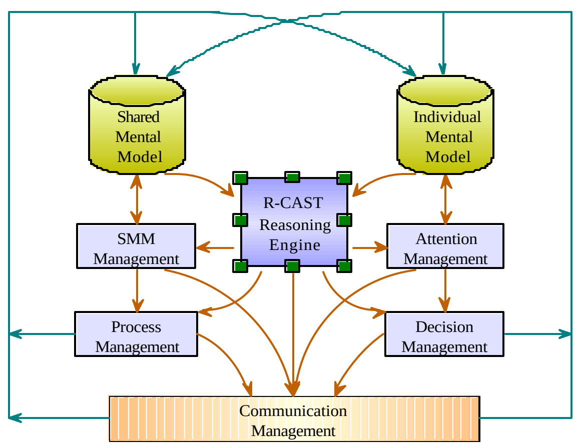

Figure 1. The R-CAST Architecture

Shared Mental Model (SMM): SMM stores the knowledge and information that are shared by all the member of a team. The SMM implemented in R-CAST contains four components: team processes, team structure, shared domain knowledge, and information-needs graphs. The SMM captures two types of information about a team process: the process template--represented as Predicate Transition (PrT) nets, and the process states--represented by token configurations, which track the current progress of an instantiated team process. A *team structure* specifies membership of teams and subteams, role requirements of a team, and the roles each agent can play. The shared *domain knowledge* may include inter-agent conversation protocols and social norms to follow, domain-specific inference knowledge, etc. An information-needs graph maintains a dynamic, progress-sensitive structure of teammates' information-needs, ensuring that only relevant information is delivered to the right entity at the right time. The SMM Management module is responsible for updating and refining the SMM and may entail inter-agent communications to maintain cross-agent consistency of certain critical parts of team members' SMMs.

Individual Mental Model (IMM): IMM stores those mental attitudes privately held by individual agents. It may contain the agent's domain expertise pertinent to its role in the team and its beliefs about the dynamic world and other team members. It is constantly updated by sensor inputs from the environment and communication messages received from other agents.

Attention Management (AM): An agent may have multiple goals to pursue. An R-CAST agent uses the *AM* module to manage the attentions under its concern. For instance, based on the agent's situation assessment and cooperation requests from other agents, the agent may pay more attention to one goal, or suspend the pursuit of one goal and switch to another. More specifically, a team process may involve various kinds of decisions (e.g., working under multiple contexts). Since each decision task will trigger one RPD process, it is the AM's responsibility to effectively and carefully adjust the decision-maker agent's attentions on decision tasks.

Process Management (PM): Once a goal is committed, the *PM* will choose a plan (COA) that can bring about the goal and create a team process. The PM is also responsible for orchestrating team members' behaviors so that they could collaborate smoothly both when everything is progressing as planned and when something goes wrong unexpectedly. More specifically, in normal circumstances, *PM* ensures all the team members behave strictly according to the committed (intended) plans, and synchronize their behaviors whenever necessary. When agents encounter in exceptional circumstances, they use their *PM* modules to collaboratively adapt to changes in the environment (exception handling).

Decision Management (DM): An R-CAST agent may trigger the RPD decision process in one of three modes: 1) based on human's recognition, 2) based on agent's recognition, or 3) based on decision points explicitly specified in a plan. In the first mode, the need for making a decision is first recognized by a human based on his expertise and his assessment of the current situation. He then delegates the identified decision-making task to his assistant agent, who will trigger the RPD process and inform other teammates of the decision-making request so that they can collaborate in making the decision. In the second mode, the need for making a decision is recognized by an agent based on its situation assessment. Running in this mode, an R-CAST agent needs to continually pay attention to the changes of cues or patterns of cues in order to recognize the emerging decision tasks. In the third mode, the needs for making decisions are explicitly specified in a team plan as fixed decision points, and the RPD process is triggered whenever agents reaches a decision point. Generally, an R-CAST agent running in the first mode can evolve into a system in the second mode, after the agent gain enough expertise (e.g., cues to consider) for recognizing decision-making tasks through learning from humans. Similarly, a system operating in the second mode can further evolve to act in the third mode.

R-CAST Reasoning Engine (RRE): RRE is the kernel of the R-CAST architecture. RRE anticipates other team members' relevant information needs based on the progress information of team activities (from SMM); infers tacit information needs based on causal link analysis; tracks teamwork progress, if needed; and initiate information seeking and sharing to achieve shared understanding of situations and team activities.

Communication Management (CM): CM governs inter-agent communications. An agent may either initiate a new conversation context or simply follow existing ones. The manager organizes related messages into conversation sessions, and monitors the development of on-going conversation protocols. CM has algorithms for proactive communication among teammates. Upon acquiring new information from the environment, CM checks whether the new information matches some teammates' future information-needs (from RRE). If there is a match, the agent will consider sending out the new information to the corresponding needers proactively.

3.1 Information requirements for decision making

Decision making is an information-intensive activity. However, information gathering (e.g., reconnaissance about enemy activities) can be expensive due to factors such as information availability and cognitive constraints. It is thus crucial to prioritize information requirements and only collect those that are relevant in making a decision.

"Cue" is a key concept in the RPD model. The term "cue" refers to an agent's internal representation of the decision situation. Cues are higher-level abstractions of the elementary data or synthesization of lower-level information. For example, internally an agent may only care about the fuzzy category (e.g., high, medium, low) rather than the real value of an object's velocity; the "moving pattern" of an approaching unit can be synthesized (fused) from the information regarding the moving directions of all the individuals in the unit. Generally, a cue can be the root of several treelike information-dependence structures, which describe the ways how the cue is abstracted from low-level information. The information requirement reasoning based on information dependency structures is given in (Fan, et al., 2005b).

3.2 Decision spaces and experience retrieval

Each decision-making task places certain expertise (information) requirements on the decision maker. Two decision-making tasks belong to the same *type* if they place the same information requirements (e.g., the same collection of cues) to consider. We thus use the concept of "decision spaces" to organize experiences for complex domain problems, where experiences related to one decision type are maintained in one experience knowledge base (EKB). Upon getting (or identifying) a decision-making task, an agent first needs to decide which decision space applies. The collections of cues considered by different decision spaces may overlap. Thus, choosing decision space itself can be a refinement process. However, the support of overlapping decision spaces is exactly the feature that can be fully leveraged to support multiple-context decision makings. The collection of overlapped cues are the shared information requirements for making the different types of decisions. It is thus very natural for an R-CAST agent who is making a decision of type *A* to share its cue evaluation with another R-CAST agent who is making a decision of type *B* when the same cue is involved in both *A* and *B*.

For each decision-making situation, an R-CAST agent first will retrieve experiences worked before under similar situations. The current situation, which is captured in the synthesized cues, is *matched* (with respect to certain similarity metrics) with past experiences in the relevant EKB. If there is such an acceptable match, the experience will be adapted for the new situation. If there is any ambiguity, the decision-making agent has to diagnose the current situation to find out what kinds of information are still missing and reconsider the situation again. While experience matching in RPD is quite similar to case matching in case-based reasoning (Aamodt and Plaza, 1994), the former emphasizes more on the cycle of matching-evaluate-information gathering. A skeleton algorithm for experience matching can be found in (Yen, et al., 2004).

3.2 Recognition refinement

In complex domains like C2CUT, it is almost impossible to collect complete information about the cues under concern

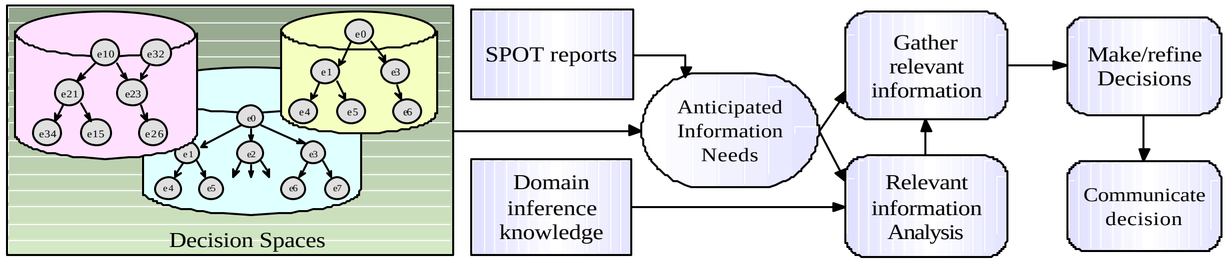

Figure 3. The roles R-CAST agent plays in assisting S2 suit human

all at a time. In such a situation, human decision makers typically consider cues gradually and refine their decisions progressively if necessary. For instance, when decision makers notice an anomaly as the situation evolves, they can adjust their recognition by considering more cues synthesized from the information that just becomes available.

We address this issue by organizing the experiences in a decision space by refinement relations. We say an experience refines another if it considers more cues (information) than the latter. A stronger refinement relation can be defined in terms of additional relations between the other components of experiences, like expectancies, COAs. For instance, for some types of decisions, it may make sense to say that an experience is a refinement of another if it considers more information (cues) in the recognition, both share the same goals, and the course of actions associated with one experience is simply the prefix of what is associated with the other. From such a perspective, experiences in an EKB can be virtually viewed as being partitioned by some experience refinement relations. For the example illustrated in Figure 2, after experience e10 being considered, experience e23 can be selected in the next round of recognition as more information becomes available. However, to elicit a meaningful experience-refinement relation from a decision space is domain-dependent, and the detailed discussion of it is out of the scope of this paper.

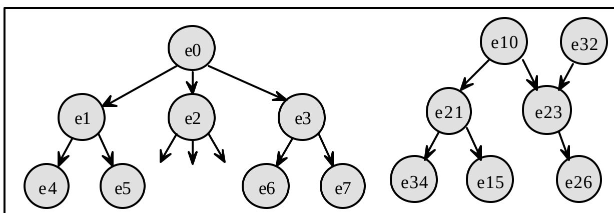

Figure 2. Structure of Decision Spaces

An expected situation can reinforce current recognition, while an anomaly can weaken it. Figure 2 shows a structure of a decision space. Suppose e3 is the current recognition, a decision maker needs to pay attention to cues of e6 and e7 (for further recognition) and the expectancies or anomalies of e0 and e3 (for revising current recognition). The example shows that in a decision space, a decision-maker should pay attention to and collect (ask others or investigate by itself) information about cues of the lower-level experiences.

In sum, by organizing experiences as decision spaces and modeling the information requirements as cues, anomalies, and expectancies, R-CAST agents can efficiently reason about time-sensitive information requirements for complex decision tasks. The algorithm for adaptive decision making can be found in (Fan, et al., 2005a).

4. Assisting C2CUT Using R-CAST

Let's see how a small event like a group forming may cause a ripple effect on everything that is going on. Suppose a large group of locals starts to form in a section of a city. The situation needs to be monitored for the peace keeping forces because they may need to react. This same situation could also cause a problem for the humanitarian effort as well: the supply route may have to be altered to avoid the group. This same group could also cause a problem for the combat forces to capture the key person. The combat forces may have to delay their mission until the streets are clear of locals.

It is thus very important to distribute the correct information to the staff officers and to create for them a situational awareness. In this project, we used R-CAST agents to assist human operators by monitoring multiple types of situations and alerting human whenever critical situations occur. Our scenario involves three battle functional areas (BFAs): the intelligence cell (S2), the operations cell (S3), and the logistics cell (S4), each is assisted by an R-CAST agent. Our discussion below centers on S2's assistant agent (S2AA). Figure 3 illustrates the roles S2AA plays in helping S2 human collect information and make decisions.

S2AA first anticipates the information needs based on the incoming SPOT reports and the cues under its concern (i.e., captured in S2AA's decision spaces). If necessary, it

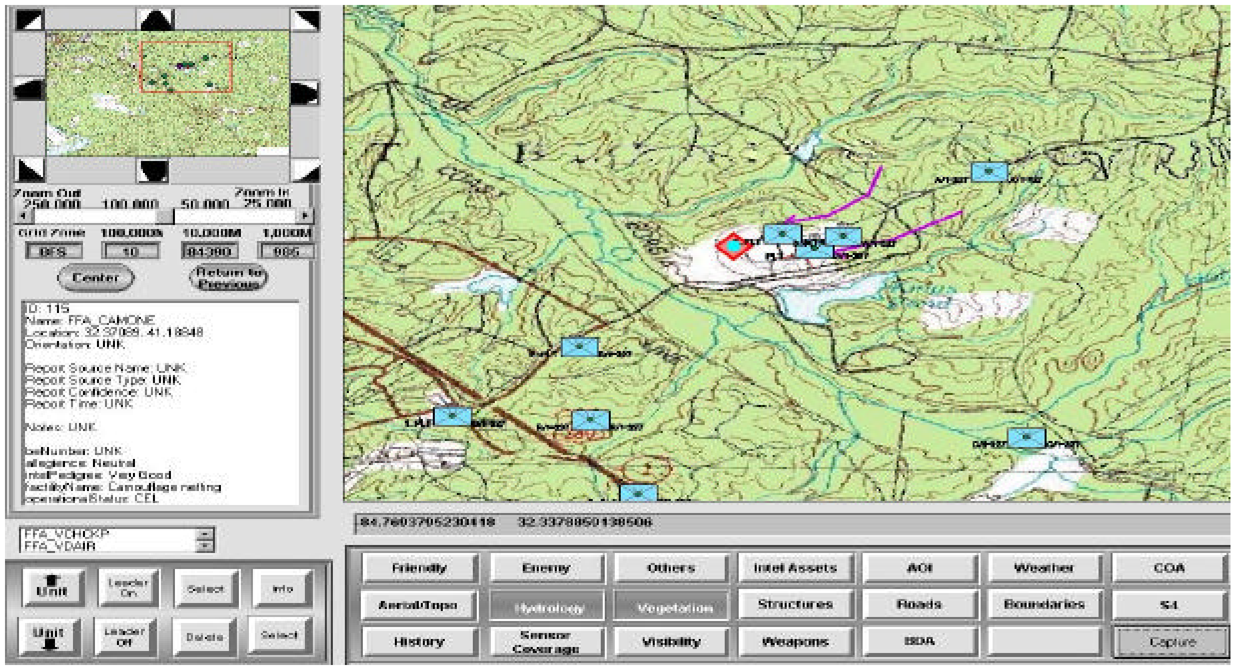

Figure 5. An example screen shot of S2's workspace map

also infers tacit information needs based on its expertise on inference knowledge. The agent then either asks other teammates (e.g., S2 human) or initiates a COA (e.g., launching an UAV) to collect the relevant information. Depending on the level of time stress, the more information S2AA collects, the better decision it can make. In the following, we only focus on decisions about information need analysis and information exchange rather than on decisions about course of physical actions.

4.1 Peacekeeping Operation

The goal of the peacekeeping operation is to keep the tribes from confrontation. SPOT reports will be generated about any crowd growing larger than some threshold levels or appearance of key persons in those crowds in the areas being patrolled. To alleviate S2 suite human's work load and cognitive load, R-CAST agent is used to assist S2 human by processing the incoming SPOT reports and making timely decisions on whether and how to alert S2 human regarding the identified threats based on the relevant experiences. Here is an experience of S2AA:

```
(Experience
   (Cue (somewhat_crowded ?group)
          (not(has_key_person ?group ?person)))
   (Goal  (peace keeping))
   (Action (plan_show_icon_only))
   (Expectancy (not_somewhat_crowded ?group)
          (has_new_SPOT_report yes)
)
```

If the crowd size is so large that a reinforcement call is required to be warranted to avert confrontation, the assistant agent will alert the S2 human. With approval from S2 human, the assistant agent will inform the combat officers about the situation. Then the combat officers can make a decision of ordering a troop in response. If the crowd size is not so large but there is a key person spotted in that crowd, the assistant agent queries the MIDB to seek information about the latest update of the person's threat level. If the threat is high enough, the assistant agent will alert the combat officers to consider launching a capture mission.

Below is a sample session of agent-human interaction in the peacekeeping context:

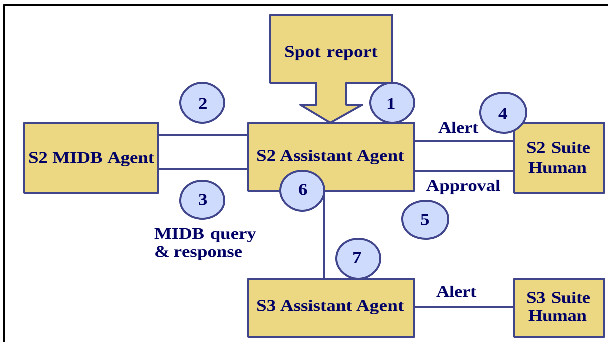

Figure 4. Interaction pattern for peacekeeping

1. S2AA receives a SPOT report containing the following information:

- (Size Group1 15),
- (Has_key_person Group1 AdMir_Erry), and
- (Location Group1 30 60);

2. S2AA sends a query to MIDB Agent:
   - (threat_level AdMir_Erry ?level);
3. S2AA gets a reply from MIDB Agent:
   - (threat_level AdMir_Erry high);
4. S2AA interprets current situation and concludes that the threat from Group1 is high, so it displays an icon on S2's workspace map, and pops up an alert window with all the information about Group1 to S2 human;
5. S2 human confirms the alert (with additional information, if necessary) to the S2AA;
6. S2AA publishes the icon on the general map;
7. S2AA alerts S3 human via S3's assistant agent (which displays an icon on S3's workspace map).

Figure 4 illustrates this interaction pattern. Figure 5 is a screen shot of S2's workspace map.

4.2 Combat Operation

A combat operation involves capturing or killing a key insurgent. Once a key insurgent has been spotted in a building, the combat officers (S3) need all the information about the building itself and its surrounding areas. Thus, as soon as S2AA receives a SPOT report containing information about a key insurgent, it will alert the combat officer about the presence of a key insurgent. In addition, S2AA executes a plan for gathering all related information about the insurgent and the area that the insurgent was spotted in. More specifically, S2AA will query the MIDB and gathers more information about the insurgent, terrain information and floor plans of the building, if any, then proactively share such information with S3AA, which presents the information to S3 human along with an icon on the map displaying the location of the key insurgent.

Below is a sample session of agent-human interaction in the combat context:

1. Spot Report comes in to S2AA:
   - (key_insurgent Laden),
   - (in_building Laden house1 32.36 -84.84);
2. S2AA alerts S3 (through S3AA, which displays an icon on S3's workspace map);
3. S2AA searches its knowledge base about the activity information about Laden;
4. S2AA queries the MIDB agent about the profile information of Laden;
5. S2AA queries the GIS agent about relevant terrain information;
6. S2AA displays an icon on S2's workspace map, and on the general map if S2 approves;
7. S2AA forwards all the acquired information to S3AA, which updates the icon on S3's workspace map.

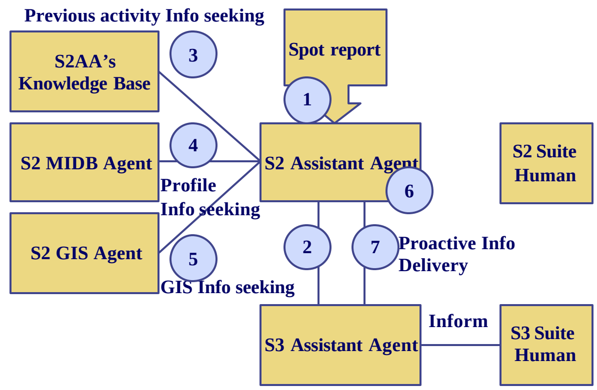

Figure 6. Interaction pattern for combat

4.3 Humanitarian Operation

The humanitarian operation focuses on monitoring routes that supply food and hospital materials throughout the city. If a threat is detected or expected, the route needs to be changed or escorted by forces.

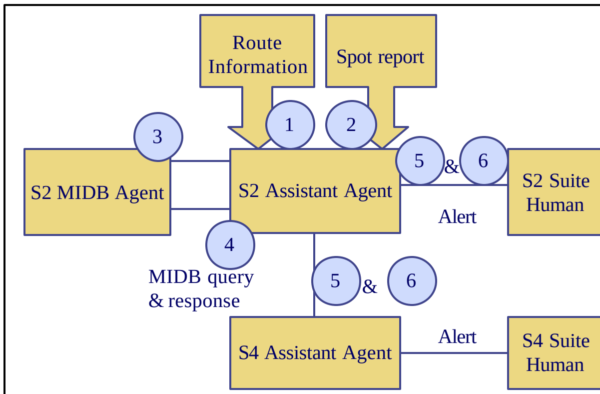

Figure 7. Interaction pattern for humanitarian

The route information is fed to S2's assistant agent (S2AA) for monitoring. S2AA then investigates the route, identifies potential threat areas, and informs the human user, if any. Threat can be either from an IED (Improvised Explosive Device) or from a hostile group. The decision space for the humanitarian operation consists of experiences that deal with different types of situations such as spotting an IED or spotting a hostile group along a route, and S2AA needs to decide whether to alert S4 with hostile information in an area.

Below is a sample session of agent-human interaction in the combat context:

1. S2AA gets updated information about supply routes;

2. S2AA receives a SPOT report about an IED:
   - (IED Bomb1),
   - (Location Bomb1 30 60);
3. S2AA sends a query to MIDB agent;
4. S2AA get replies from MIDB agent;
5. S2AA assesses the situation and concludes that that the IED has a threat to the route, so it displays an alert window with all the relevant information (IED name, location, time, etc.) on both the S2 and S4 machines;
6. In addition, an icon is published with all the relevant information on the general maps of S2 and S4.

5. Comparison and Discussion

Others have been investigating (a) computational approaches to RPD to represent human decision making for concept exploration, analysis, or evaluation (e.g., (Warwick, et al., 2002)); (b) the use of software agents for robust battlefield simulation (e.g., (Allen, et al., 2004)); (c) the use of agents as aids to information filtering in a decision environment (e.g., (Knoblock and Ambite, 1997; Turner, et al., 1997)); (d) shared situation awareness (e.g., (Endsley and Robertson, 2000)), cognitive models of situation awareness (e.g., (Gonzalez, et al., 2004)); and (e) teaming with automation (e.g., (Christofferson and Woods, 2002)). The collaborative-RPD model implemented in R-CAST is linked to but also distinguished from the existing work in important ways. First, R-CAST is the first RPD-enabled agent architecture designed for supporting teamwide collaborations (including human-agent and agent-agent collaborations). With collaboration in mind, we take an intensive view of the recognition phase of the RPD process and focus on the investigation of how proactive information exchange among teammates might affect the performance of a decision making team. Second, R-CAST agents can proactively reason across decision spaces, seek missing information from external intelligence sources, exchange relevant information among teammates, and monitor an on-going decision against potential expectancy. Third, the "cognitively-aware" agents, as teammates or decision aids, each assigned to a specific functional area, can be used to assist human teams (e.g., military staff) in developing shared situation awareness while balancing information requirements against the dynamic and time sensitive decision making process.

Case-based reasoning (CBR) is another psychological theory of human cognition (Slade, 1991), focusing on the process of reminding (experience-guided reasoning) and learning. While there is no clear line between RPD and CBR as far as their process models are concerned (e.g., both cover experience retrieval, solution adaptation and evaluation), they differ in several important aspects. First, RPD originates from studies about how human experts make decisions under time pressure (Klein, 1997; 1998). Experiences in RPD are prior decision making cases, while experiences in CBR can be of any kind. From such a perspective, RPD can be taken as a subfield of CBR. Second, while storage and retrieval are central aspects of CBR, research on RPD is more concerned with the iterative process of recognition refinement (i.e., developing better situation awareness through information gathering). Third, RPD systems ought to be aware of time stress and make as better decisions as time permitted, but this is not a requirement on CBR systems. In addition, the Collaborative-RPD model implemented in R-CAST takes a more extensive view, focusing not only on human-centered teamwork in making decisions, but also seriously addressing related issues such as collaborative situation awareness and expectancy monitoring.

It may be argued that the RPD model does not work *ab initio*; it works on a body of expertise acquired through length engagement in a domain of practice. Then, one critical question is how R-CAST deals with experience acquisition. To answer this question, one has to admit that experiences (cases) are quite different from rules, which are the basic unit of knowledge of traditional expert systems. Knowledge acquisition process becomes a bottleneck of rule-based expert systems, because oftentimes the rules articulated by human experts in fact did not accurately reflect their own problem-solving behavior (Slade, 1991). Psychological studies indicate that people remember their own experience and it is easier for them to articulate knowledge as experience than rules (Slade, 1991). This view favors case-based models and systems. Of course, experience acquisition in RPD is still a challenging task due to human factors. Our first pass at tackling this issue was to rely on Army in-house expertise to populate agents with experiences. In the long run, we will accommodate experience learning into R-CAST, drawing upon research in case-based reasoning (Aamodt and Plaza, 1994). We are also planning to develop a tool to facilitate the KA process for RCAST.

Another interesting question is how time stress is understood by agents. For the purpose of explanation, we simplify the decision process of an R-CAST agent as a sequence (Information-gathering)--(Feature-matching)--(Similarity-evaluation)--(Expectancy-monitoring)--(COA-adjustment), where each step can cycle back to the beginning of the sequence. Among the steps, we can assume the algorithms for Feature-matching and Similarity-evaluation take fixed times (of course, it depends on the size of the experience base and the similarity metrics), and obtain their approximates (T_f, T_s) empirically. Information-gathering, Expectancy-monitoring, and COA-adjustment are anytime algorithms: the more time spent, the better solution obtained. Let T (e.g., 3 minutes from now) be the time constraint associated with the current decision task (T can be figured out dynamically by a human and input to the partner R-CAST agent, or an empirical value preset for a type of decision tasks). Then, an R-CAST agent has to allocate the time ($T-T_f-T_s$) to the activities of Information-gathering, Expectancy-monitoring, and COA-adjustment appropriately. The strategy currently used is to favor the information gathering activity, say, 60% of ($T-T_f-T_s$) is used for collecting missing information that is critical in cue synthesis. Expectancy-monitoring is a parallel process. Whenever an anomaly occurs and the decision process needs to restart, the remaining time will be re-allocated to the activities, and the process continues until either a workable solution is found or time out.

Previously (Fan, et al., 2005a), we conducted experiments in DDD environment to evaluate the adaptive decision making feature of R-CAST agents. While the experimental result reinforced the psychological findings that people are extremely sensitive to time pressure, it also indicated that as a cognitive aid, R-CAST agents can alleviate human's stress caused by time pressure. In this paper, via case study, we demonstrated that R-CAST agents can help collect and share relevant information in the C2CUT domain. This is our first step, however. One of our on-going efforts is to conduct experiments using the C2CUT scenario to understand how R-CAST agents can help people switch their attentions between multiple contexts and evaluate their performance gains.

6. Summary

One of the challenging issues in the domain of C2CUT is how to assist human combat staff to effectively collaborate and make decisions under multiple contexts. The goal of peace keeping is to monitor the streets of a city; this requires timely intelligence information in order to stop problems before they happen. The goal of humanitarian is to safely provide supplies to a local hospital, which needs a clear route to deliver supplies. The goal of combat is to capture a key person in a section of a city. Each mission can have an affect on the other. The ability of staff officers to monitor these situations and react in a timely manner is the key to success.

In this paper, we described a novel approach that uses RPD-enabled agents to support timely decision-making under multiple contexts. The R-CAST architecture has implemented a built-in Recognition-Primed Decision model, leveraging both agent-agent collaborations and agent-human collaborations during the decision-making process. R-CAST agents can proactively reason across decision spaces, seek missing information from external sources, exchange relevant information among teammates, and monitor an on-going decision against potential expectancy. It can be used to develop systems for enhancing the capabilities of anti-terrorist analysts in early detection of potential terrorist threats. R-CAST supports adjustable autonomy, as well. R-CAST agents can collaborate with human partners to monitor expectancies and progressively refine recognitions.

Currently, each R-CAST agent can only work on one type of decisions at a time. Particularly in the C2CUT scenario, the S2 human is assisted by three R-CAST agents which are in charge of peacekeeping, combat, and humanitarian situations, respectively. There are no tight connections among the three R-CAST agents. It would be better if an R-CAST agent can concurrently monitor multiple decision processes so that different views of the current situation can be better connected and human's attentions can be better adjusted. For the future work, as well as extending the existing decision adaptation mechanism by developing tools for supporting experience learning from people, one on-going investigation is to empower R-CAST agents with the capability of helping human operators switch contexts (attentions), and to conduct experiments to evaluate the performance gains of human teams when they are assisted by attention-monitoring R-CAST agents in dynamic, complex domains like C2CUT. The final system is aimed to allow experimentation and demonstration of advanced tactical information exchange, reduced cognitive load, enhanced situation awareness, and positive human-agent collaboration.

7. Acknowledgement

This research has been supported by funds from the US Army Research Laboratory under the auspices of the Command & Control for Complex and Urban Terrain--Army Technology Objective (C2CUT ATO).

8. References

Aamodt, A. and Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, 7(1):39-59.

Alberts, D., Garstka, J., Stein, F. (2002). *Network Centric Warfare*, CCRP.

Allen, J., Barry, K., and McCormick J. (2004). Plan execution monitoring with distributed intelligent agents for battle command. *Proceedings of SPIE*, Vol. #5441.

Cannon-Bowers, J.A., Salas, E. and Converse, S.A. (1990). Cognitive psychology and team training: Training shared mental models and complex systems, *Human Factors Society Bulletin*, 33:1-4.

Christofferson, K., Woods, D.D. (2002). How to make automated systems team players. In E. Salas (Ed.), *Advances in Human Performance and Cognitive Engineering Research*, Vol. 2. JAI Press, Elsevier.

Endsley, M.R., Robertson, M.M. (2000). Training for situation awareness in individuals and teams. In M. R. Endsley & D. J. Garland (Eds.), *Situation awareness analysis and measurement*. Mahwah, NJ: LEA.

Fan, X., Sun, S., McNeese, M., and Yen, J. (2005a). Extending the recognition-primed decision model to support human-agent collaboration. In *Proceedings of AAMAS'05*, July.

Fan, X., Sun, S., and Yen, J. (2005b). On shared situation awareness for supporting human decision-making teams. In *Proceedings of 2005 AAAI Spring Symposium on AI Technologies for Homeland Security*, pages 17–24, March.

Gonzalez, C., Juarez, O., Graham, J. (2004). Cognitive and Computational Models as Tools to Improve Situation Awareness. *Proceedings of the 48th Annual Meeting of the Human Factors and Ergonomics Society*. New Orleans, LA, September.

Hutchins, E. (1995). Cognition in the wild. Cambridge, MA: MIT Press.

Ilgen, D.R., Major, D.A., Hollenbeck, J.R. and Sego, D.J. (1993). Team Research in the 1990's, in M. Chemers, R.A. Ayman, Eds., *Leadership Theory and Research: Perspectives and Directions*, San Diego: Academic Press.

Klein, G (1993). A recognition-primed decision making model of rapid decision making. In G. Klien, J. Orasanu, R. Calderwood and C., Zsambok Eds., *Decision Making In Action: Models and Methods*, pages 138-147.

Klein, G. (1997). The recognition-primed decision (RPD) model: Looking back, looking forward. In C. Zsambok and G. Klein, Eds., *Naturalistic decision making*, pages 285-292.

Klein, G. (1998). Recognition-primed decision making. *Sources of power: How people make decisions*, pages 15-30.

Knoblock, C. and Ambite, J.L. (1997). Agents for Information Gathering. In J. Bradshaw, editor, *Software Agents*. AAAI/MIT Press, Menlo Park, CA.

Liang, Y., Robichaud, F., Fugere, B., Ackles, K (2001). Implementing a Naturalistic Command Agent Design. In *Proceedings of the tenth conference on Computer Generated Forces*, Norfolk, VA, pages: 379-386.

McNeese, M., Salas, E., and Endsley, M. (Eds.). (2001). *New trends in collaborative activities: Understanding system dynamics in complex environments*. Santa Monica, CA: Human Factors and Ergonomics Society.

Morrison, J., Kelly, R., and Hutchins, S. (1996). Impact of naturalistic decision support on tactical situation awareness. In *Proceedings of the 40th Human Factors and Ergonomics Society Annual Meeting*, page 199-203.

Norling, E., Sonenberg, L., and Ronnquist, R. (2000). Enhancing multi-agent based simulation with human-like decision making strategies. In *Proceedings of the second International workshop on Multi-Agent Based Simulation* (Moss, S. and Davidsson, P. Eds.), page 214-228.

Panzarasa, P., Jennings, N., and Norman, T. (2002). Formalizing collaborative decision-making and practical reasoning in multi-agent systems. *Journal of logic and computation*, 12(1):55-117.

Robichaud F. (2002). Implementing a naturalistic command agent design. Presented at the *Workshop on Computerized Representation of RPD*.

Simon, H.A. (1955). A Behavioral Model of Rational Choice. *Quarterly Journal of Economics* 69: 99–118.

Slade, S. (1991). Case-Based Reasoning: A Research Paradigm. *AI Magazine*, 12(1):42-55.

Sokolowski, J. (2003a). Enhanced Military Decision Modeling Using a MultiAgent System Approach. In *Proceedings of the Twelfth Conference on Behavior Representation in Modeling and Simulation*, Scottsdale, AZ., May 12-15, pp. 179-186.

Sokolowski, J. (2003b). Representing Knowledge and Experience in RPDAgent. In *Proceedings of the Twelfth Conference on Behavior Representation in Modeling and Simulation*, Scottsdale, AZ., May 12-15, pp. 419-422.

Sycara, K. and Lewis, M. (1991). Forming shared mental models. In *Proc. of the 13th Annual Meeting of the Cognitive Science Society*, pp. 400-405.

Turner, PJ and Jennings, NR. (1997). On Scalability of Information Management Agents. In *Proceedings of the European IT Conference (EITC-97)*, Brussels, Belgium.

Turner, S.R. (1994). *The Creative Process: A Computer Model of Storytelling and Creativity*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Warwick, W., McIlwaine, S., Hutton, R., McDermott, P. (2001). Developing computational models of recognition-primed decision making. In *Proceedings of the tenth conference on Computer Generated Forces*, Norfolk, VA., page 232-331.

Warwick, W., McIlwaine, S., Hutton, R. (2002). Developing computational models of recognition-primed decisions: Progress and Lessons Learned. In *Proceedings of the 11th conference on Computer*

Generated Forces and Behavior Representations, Orlando, FL.

Woods, D.D., Patterson, E.S. and Roth, E.M. (1998). Aiding the Intelligence Analyst in Situations of Data Overload: A Diagnosis of Data Overload. Institute for Ergonomics/Cognitive Systems Engineering Lab Report, *ERGO-CSEL 98-TR-03*, The Ohio State University.

Yen, J., Fan, X., Sun, S., Hanratty, T., and Dumer, J. (2005). Agents with shared mental models for better tactical decision-makings. *Journal of Decision Support System*.

Yen, J., Fan, X., Sun, S., McNeese, M., and Hall, D. (2004). Supporting Anti-terrorist Analyst Teams Using Agents with Shared RPD Process. In *Proceedings of IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety,* pages 53-60, July.

Author Biographies

XIAOCONG FAN - is a PostDoctoral Researcher at the School of Information Sciences and Technology, the Pennsylvania State University. He received his PhD degree from Nanjing University in 1999. He currently works on the modeling of helping behaviors in multi-agent systems and on the building of agent architectures supporting human-centered teamwork.

SHUANG SUN - is a PhD candidate at School of Information Sciences and Technology (IST), the Pennsylvania State University. His research interests include team-based intelligent agent technology and e-commerce. He is a certified professional on products of many leading IT organizations including SAP, Microsoft, Lotus, and Nortel.

JOHN YEN - is University Professor of Information Sciences and Technology and Professor-in-Charge at the School of Information Sciences and Technology, the Pennsylvania State University. He is the founding director of the Intelligent Agent Laboratory at Penn State. He holds a PhD in computer science from University of California, Berkeley. He is a Fellow of IEEE. His current research focus is developing team-based agent technologies for supporting human decision making teams as well as transferring these technologies into applications related to homeland security, information fusion, and web service composition.

BINGJUN SUN - is a Ph.D. student in the department of Computer Science and Engineering, the Pennsylvania State University. He earned his bachelor's degree from the Tsinghua University in 1997. Currently, he is a research assistant in the Laboratory for Intelligent Agents. He is interested in machine learning in intelligent agent systems.

GURUPRASAD AIRY - is a Ph.D. student in the department of Computer Science and Engineering, the Pennsylvania State University. He received his B.E. degree from the Department of Computer Science and Engineering, Bangalore University, India. His research interests are in the areas of artificial intelligence, trust and security issues in multi-agent systems and computer graphics and animation. He is a student member of SIAM and IEEE.

MICHAEL MCNEESE is an associated professor of information sciences and technology, and an affiliate professor of psychology and learning systems performance at Penn State. He received the B.A. degree in Psychology and the M.A. degree in Experimental-Cognitive Psychology from The University of Dayton, and the Ph.D. in Cognitive Science from Vanderbilt University. He previously directed the Collaborative Systems Technology Laboratory at the Air Force Research Laboratory, and now is the director of the User Science and Engineering (USE) Lab at Penn State. McNeese studies human interaction with information technology in complex environments, particularly collaborative systems that bring together the confluences of cognition, computation, collaboration, and context for given fields of practice. His current research interests include problem-based learning, cognitive systems engineering, cognitive task analysis, and developing naturalistic decision systems applied to the domains of emergency crisis management and intelligence analysis.

TIMOTHY HANRATTY - is a senior computer scientist at the US Army Research Laboratory. His professional interest focus on applied research and development in the areas of knowledge engineering and advance decision support technologies. Hanratty is a doctoral candidate in Applied Information Technology at Towson University and received his MS from Johns Hopkins University.

JOHN DUMER - is a senior computer scientist at the US Army Research Laboratory. His principal research areas include fuzzy logic, reasoning under uncertainty and cooperative expert system technologies. Dumer holds a BS in computer science/communications from Towson University.