A Framework for Supporting Teamwork between Humans and Autonomous Systems

Elyon A.M. DeKoven and Anne K.G. Murphy
Soar Technology, Inc.
3600 Green Court, Ste. 600, Ann Arbor, MI, 48105
dekoven@soartech.com; akgmurphy@soartech.com

ABSTRACT

The US Army's vision of future warfare includes command and control (C2) of multiple manned and uninhabited assets in parallel. Central to this vision are human-robotic teams, in which uninhabited assets and human warfighters operate in a coordinated fashion toward shared objectives. Effective C2 will require user interface controls that allow an operator to integrate all types of elements in these heterogeneous teams in support of effective coordinated tactics and procedures.

The Intelligent Control Framework (ICF) project at Soar Technology is exploring issues related to the design and development of operator interfaces for C2 of manned and uninhabited assets. We are presently focused on aspects of teamwork related to collaborative planning. In this context, the ICF architecture forms a communicative substrate for human-system negotiation about task responsibilities and levels of autonomy among assets.

This paper describes three tiers of collaboration that need to be supported in such C2 interfaces and the system intelligence required to support those tiers. We describe our implementation of an Adjustable Autonomy Module (AAM) as a partial fulfillment of these reasoning requirements within the ICF system, and use the three tiers to discuss lessons we have learned concerning interaction design to support operator-system communication about plans and asset autonomy.

Conference Topics

C2 Concepts and Organizations, Social Domain Issues, C2 Architecture

1. INTRODUCTION

Modern warfare tactics specify key roles for uninhabited vehicles and other robotic elements. For example, currently the US Army is using robotic assets in the removal of roadside improvised explosive devices (IED) protects human soldiers from both the direct danger of the explosives, as well as possible sniper fire from enemy forces watching the IED emplacements. In the future, these roles will include mixed human-robotic teamwork in a high-tempo situation, such as in disaster relief or breaching an enemy compound.

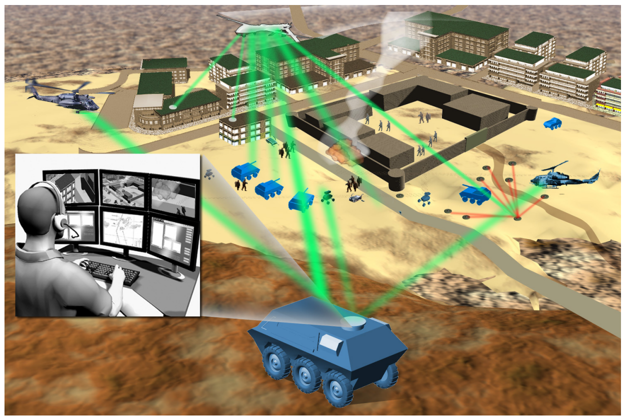

Figure 1: Future combat scenario involving single operator control of multiple uninhabited assets

Two goals for future force development (FCS ORD 2003) are to increase the number of robotic elements that can be controlled by a single human operator (as depicted in Figure 1), and increase the effectiveness of those robotic elements when deployed. However, real-time control of multiple, uninhabited battlefield robots and other semi-autonomous systems is an extremely difficult problem that has not been adequately addressed.

Simply making the UVs more autonomous will not necessarily lead to increased operator effectiveness and can actually increase operator cognitive workload. When cognitive workload is increased, especially in high stake situations where the operator must know what the system is doing (and why), operator trust is diminished, and the system may not be used as a result. While increased system autonomy can reduce operator task workload, there have been numerous studies on automation being ignored, switched off or removed, often for reasons that could be corrected simply by improving the user interface design (Parasuraman, 2000;Parasuraman and Riley, 1997).

The user interface is the key to helping the operator communicate goals and preferences to the system, in order to offload tasks to system automation, while maintaining an appropriate level of human supervisory control over system activity. Human-automation interaction designers and researchers have repeatedly stressed the value of looking at the human and machine as two (or more) agents working collaboratively toward achieving the human's goals. In fact, the U.S. Army's requirements for future forces (FCS ORD 2003) explicitly takes a teamwork-oriented perspective, using different forms of the term 'collaboration' more than eighty times. In order to achieve that vision, system intelligence must be designed, not as a replacement for human capabilities, but rather as part of a connected cognitive unit with the human's intelligence. As Woods (2001) writes:

> "To create successful human-automation ensembles we need data, models and innovation on distributed cognition. Agents, be they human or machine, are always partially autonomous with differential capabilities. Robust performance derives from how different roles are coordinated given the inherent variability of the world, given irreducible uncertainty about the situation and future, given finite resources of any agent and set of agents tasked to achieve control in the pursuit of goals, and given that there are always multiple but potentially conflicting goals. This means that modeling, studying and designing how cognitive work is distributed is as fundamental as increasing machine autonomy and that balance across these lines of work is necessary if we are to avoid repeating 'classic' design errors."

Soar Technology is addressing these challenges by developing an Intelligent Control Framework (ICF) from which to create adaptive C2 interfaces that augment warfighter abilities by automating many of the tasks and data transformations that now must be performed manually, while maintaining a useful level of supervisory control over the automation. ICF is a multi-agent system in which teams of cooperative intelligent software agents form the basis for a warfighter's assistant. Like a commander's support staff, ICF is designed to take requests from the operator at a high level, anticipate operator needs and situational requirements, and request input from the operator at appropriate times.

We initiated this research with the development of two systems for intelligent human-system interaction: one for task automation, called CIANC[3] (Beard et al., 2003; Wood, 2003), and one for enhanced information delivery, called BINAH (Zaientz et al., 2005). While either of these systems could be useful as statically-configured interaction tools, their full benefit will be realized when their behavior can be adapted to the user's immediate task needs. As such, we needed a system for determining in which tasks the user is actively engaged (we are assuming that users in inherently complex domains are rarely engaged in only one task at a time), the various goals the user is trying to achieve, and preferences and constraints the user imposes on how to achieve them. This system must be able to reason about the relative importance of user task types, environmental events, the user's cognitive abilities, and the capabilities of the UVs available to support the user's task. The goals of the system are to automate those tasks that are mundane or tedious, provide information only when it is useful and in the form that is most usable, and control tasks that the user is unable or unwilling to accomplish. If successful, such a system would form a collaborative relationship with the user.

The rest of this paper describes the results of our initial efforts at building this "missing layer" for user task adaptation, called the Adjustable Autonomy Module (AAM), and integrating it with a plan execution system (PES) and a C2 user interface (UI) similar to those currently envisioned by the US Army. We first describe three tiers of collaboration that need to be supported and the system intelligence required to support those tiers. We describe our implementation of an Adjustable Autonomy Module (AAM) as a partial fulfillment of these reasoning requirements within the ICF system. We then discuss lessons we have learned concerning interaction design for each of the three tiers, focused on ways to support communication between the operator and the system about plans and asset autonomy. The paper concludes with a summary of key research, design and development areas requiring further work.

2. SYSTEM REQUIREMENTS

The specific technical design challenge we adopted for ICF is to develop a *computational model of collaborative behavior* that supports human-automation collaboration commensurate with the U.S. Army's requirements for future forces operations (FCS ORD 2003). While keeping in mind that certain functions should not be completely automated, such as a decision to open fire, the system needs the ability to take over certain tasks from the user, either autonomously or as directed by the user. At the same time, the system must take direction from the user, and keep the user usefully in the loop in any autonomous decisions. Determining what is a "useful" system in any domain is primarily a design challenge, not strictly a technical one.

Just as in human-human collaboration, human-automation collaboration requires extended conversations aimed at making plans and coordinating activities. However, although human-human collaboration can occur in many ways, human-automation collaboration takes place in an application's user interface. In order to support effective collaboration, the user interface must be designed to appropriately minimize the human's *metal effort*, and to support *communication* aimed at ensuring that the human's and machine's *actions are coordinated*. As stated in (Rich, Sidner and Lesh, 2001):

> "Participants in a collaboration derive benefit by pooling their talents and resources to achieve common goals. However, collaboration also has its costs. When people collaborate, they must usually communicate and expend mental effort to ensure that their actions are coordinated."

Our hypothesis is that a system designed according to norms of human interaction will tend to be a better assistant for the human than ad-hoc designs. The norms we support come from existing knowledge in cognitive engineering, collaborative discourse and dialogue design. In particular, Woods (2001) identifies seven human-automation design challenges, from a cognitive engineering perspective, each placing a number of requirements on system interaction design; Rich, Sidner and Lesh (2001) discuss human-automation issues from a collaborative discourse perspective, specifically regarding building and maintaining shared plans and common focus of attention; and DeKoven (2004) discusses designing user interfaces to support human-automation collaborative dialogue aimed at achieving the user's goals. Combining the requirements from these sets of literature, applications built from ICF should support the following three tiers of collaboration:

- *Expression of user intention and negotiation on plans and preferences*. The user must be able to express what is to be done and how, and the system must be able to give constructive feedback and suggest useful alternatives.
- *Human-in-the-loop automated re-planning and synchronization of actions*. Plans built under collaboration are not always complete, leaving some details to be filled in over time. This is crucial to remaining resilient to surprises or conflicts, within existing plan constraints. The user and the system must be able to try out different options to filling in plan details, working together toward local optimization of resources. Collaborators must know what other participants are doing, believe that each can fulfill its responsibilities, and be able to renegotiate responsibilities when the situation warrants (Grosz and Sidner, 1990).

- *Team situational awareness and shared focus of attention in a multi-threaded situation*. Complex real-time situations, such as modern warfare, involve multiple simultaneous threats and high-priority decisions. Collaborators need to be aware of what the other participants are doing, and make sure their activities are appropriately observable and interpretable. Collaborators need to be able to quickly shift their focus of attention as a team, making sure they understand what is happening and what needs to be done, and able to resume their previous focus as necessary. Moreover, each participant needs to make its activities observable to other participants in a way that helps the other participants properly interpret their focus of attention and eventual intentions. At the same time, participants must know what not to make observable, so as not to overwhelm their collaborators.

A user interface that's usable *and* useful must support all three tiers of collaboration through 1) user situational awareness, 2) inspection and manual control of the system's autonomous decisions and actions, and 3) user-system communication about tasks and delegation of responsibility. In turn, these user-facing capabilities require an intelligent system capable of handling some of the task load, and of performing autonomously when given permission by the user or when the situation warrants it.

Previous work on autonomous agents focused on providing a fixed set of actions that the agent could perform without human intervention. Everyday examples of this type of system include home thermostats that automatically turn the heating on or off as the temperature in the house changes, or a programmable thermostat that might modify its programs in response to historic patterns in the house's thermodynamics. While this level of agency is predictable (a key factor of usability), it is ill equipped to handle surprise or flexible human-automation collaboration (a key factor of usefulness according to (Woods, 2001)).

Today, researchers are building agents that can adjust their own level of autonomy. Such agents typically reason over the current (task-relevant) situation using heuristic rules of action utility, task priority or operator workload. The results of applying the rules are used to shift the agent across a spectrum between passively and aggressively helping the user achieve or maintain a certain state. Rather than fixed automation states, adjustable autonomy can provide flexibility with respect to changing situations and operational requirements from the user.

While adjustable autonomy is a key technical enabler of effective human-automation collaboration, the precise user requirements from adjustably autonomous collaborative systems are still unknown, and there is presently little agreement as to the best source of information or algorithms to determine autonomy shifting. Nevertheless, in order to respond to current user needs, an adjustably autonomous collaborative system must generally follow a process of perception, reasoning and acting, including:

1) Perceiving (input): Sensing external, internal and user stimuli

2) Reasoning: There are many types of reasoning an autonomous system must perform. In the context of ICF we have focused on the following:

    a) Recognizing user intentions, plans and progression through existing tasks – key aspects of user modeling

    b) Calculating high-level situational factors – also called situation reasoning

    c) Using calculated situational factors to determine automation levels – delegating responsibilities across controlled assets and (requests to) the user

3) Acting (output): Updating control instructions to automated processes and assets, and sending status reports across the network and command chain

As noted above, there is no single method for building such systems. One goal of the ICF project is to provide tools to more rapidly create adjustably autonomous collaborative systems for specific user needs. The tools will need to be expressive of both the general collaborative capabilities described earlier in this section, as well as the perception, reasoning and acting faculties described above.

3. ICF ARCHITECTURE OVERVIEW

The architecture of our Intelligent Control Framework (ICF) is based loosely on the reference architecture presented by Wood (2003), but organized in abstraction layers from which specific interfaces and applications can be created (see Figure 2).

ICF is an *n*-tiered, service-based architecture in which agent-based technologies and other analytical services are combined with domain-specific user interfaces to form specific battlefield applications. It can be combined with other battle command and robotic control systems to enhance user performance, and can communicate with underlying control and simulation systems via standard protocols. Underlying ICF is an interface to various requisite knowledge bases — including standard, high-performance database implementations — as well as ontological knowledge representations that include richer semantic relationships. Collectively, the applications, agents, and services represent a cooperative technology that integrates various forms of user assistance in a holistic manner. The result is a general framework for system intelligence that can be applied to any echelon of command and control, from individual warfighters on up.

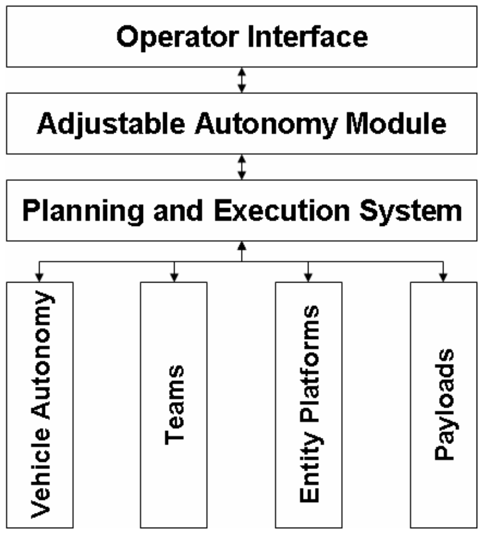

Figure 2: Intelligence and control abstraction layers for ICF

We designed the core capabilities of ICF around three main modules corresponding to the three tiers of human-automation collaboration overviewed in the previous section (see Figure 2):

- The adjustable autonomy module (AAM) contains the reasoning necessary for modeling user plans and intentions, situation reasoning, and determining new automation settings. This module covers most of the reasoning requirements labeled (2a-c) in the previous section.
- The planning and execution system (PES) contains an abstraction layer for controlling automated processes, as well as providing the rest of the system access to information generated by the automated processes, such as sensor readings and asset status information. This module covers most of the perception (1) and action (3) requirements described above.

- The user interface (UI) supports operator-system communication and provides the operator with appropriate levels of situational awareness and direct control of autonomous assets. While aspects of the system intelligence necessary to support adjustably autonomous collaboration with the user are found in the other two components as well, the user's only awareness and utilization of system intelligence is through the UI. This module covers all three tiers described in the previous section.

Aspects of the work behind the UI efforts in ICF can be found in (Zaientz et al., 2005; DeKoven, 2004); the PES has previously been discussed in (Beard et al., 2003; Wood, 2003), with the addition in ICF of more complete implementation of the deontic reasoning (Lisse et al., 2006); and details of the AAM implementation can be found in (DeKoven and Wood, 2005). Rather than repeating these results, this paper discusses the lessons we have learned from integrating the AAM with the PES and a domain-specific UI.

Figure 3 shows a flow diagram describing the critical path processes for ICF. The AAM takes as input user actions, as well as system status information and external events related to the user's mission, function, and current tasks from the PES. As output, the AAM produces a model that reflects its understanding of the user's current situation, and a table of adjustably autonomous sub-systems and current tasks with associated autonomy levels. The autonomy levels indicate to these sub-systems which tasks they should be engaged in at any time, and to what degree they should be performing or monitoring those tasks, along with links back to information in the user model. The PES translates this model into control statements relevant to the particular types of assets under the user's control, and sends the command sequences to the assets in order to carry out the necessary actions.

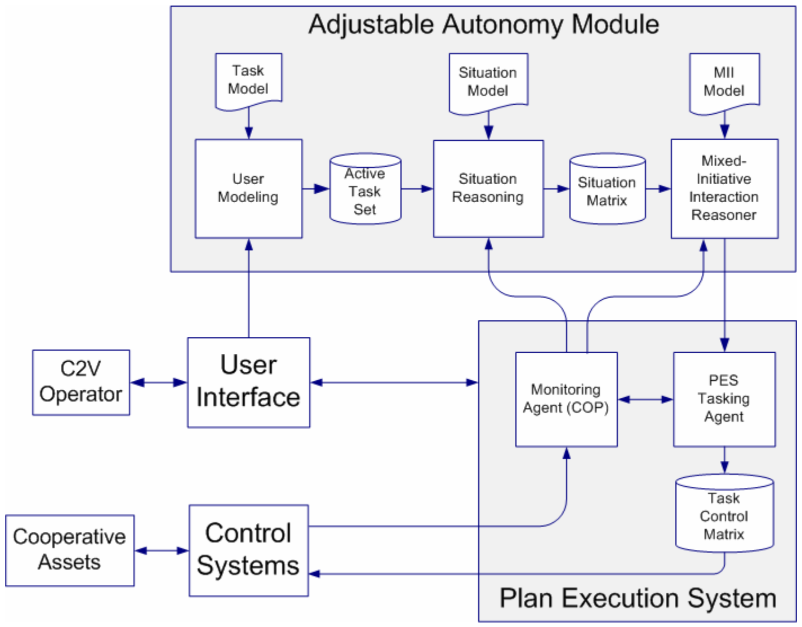

Figure 3: Flow diagram for ICF, incorporating the Adjustable Autonomy Module, Plan Execution System, User Interface and Robotic Control Systems

The following sections describe the ICF operations with regard to the perceiving, reasoning and acting processes listed in Section 2.

3.1 Perceiving: Sensing external, internal and user stimuli

In the sensing process in ICF, input is collected from three main sources: user actions, such as keystrokes, spoken commands, and other forms of human-system interaction; world and environmental events, such as new threats, targets, or commands from higher echelons; and system actions, such as changes in system status, completion of system- or user- requested functions, and other information updates from the system that may affect the user. Information sensed about the user is used as input to the task reasoning process in the user modeling component in the AAM. Information from external sources is combined into a world model in the monitoring agent in the PES. There is currently no process for modeling internal events in ICF.

3.2a Reasoning: User modeling

The user modeling component in the AAM builds a picture of user goals and plans by interpreting actions in the user interface against a pre-designed task model. The process of recognizing the plans and goals of others based upon observations of their actions and context is called *plan recognition*. *Plan reasoning* is a deliberative process in which the system considers or determines meta-information about the tasks, such as when a task is active or abandoned, or whether a task is difficult for the user or demands a high amount of user workload.

Plan recognition and plan reasoning are common tasks performed by humans in everything from daily activities (e.g., detecting/anticipating another driver's turns, even if the other driver does not use directional signals) to warfare (e.g., anticipating the intentions of hostile forces). However, getting software agents to perform these tasks with regard to users is quite difficult. There are a large number of issues associated with plan recognition. As described in (DeKoven and Wood, 2005), these include aspects of *observability*, *complexity*, and *ambiguity*.

This is by no means an exhaustive list. Nevertheless, it indicates that understanding what another is doing in a complex situation is rarely completely accurate, even for human observers. All these issues stem from a mismatch between the user's and the agent's understanding of what is happening. As in human-human collaboration, this mismatch can be addressed, though not completely solved, through extended human-machine interaction sequences for grounding, clarification, and specification. The three tiers of collaborative interaction mentioned in Section 2 are key elements in this process.

3.2a.1 Task models

The design of the algorithms for both plan recognition and plan reasoning is dependent upon the form of task (or "plan") modeling in the system. Task models are used by adaptive systems to collect information about the current user state and are used to determine how and when to help the user given current task constraints, current and past performance, and user preferences and styles. In the present implementation of ICF, task models are comprised of a form of GOMS hierarchical task decomposition that can account for all user functions (Card, Moran, and Newell, 1983). These models compose low-level physical (e.g. pressing a button), cognitive (e.g. remembering a name) or perceptual (e.g. reading a label) tasks in different methods for accomplishing high-level goals (e.g. completing a mission or collecting information to support making a decision).

GOMS models represent the procedural knowledge required to perform tasks from the user's perspective. That is, they represent the "how to do it" information about a task, rather than "how it works" information about a system (Kieras and Polson, 1985). For this reason, GOMS models can even be used early in the design process, before an interface is built, to rapidly explore a design space.

3.2a.2 Probabilistic plan recognition

While GOMS is used as a generative description of user interaction with a system, GOMS was not designed for use in plan recognition. Part of the challenge here is that GOMS models generally make strong use of mental operators, such as remembering something or visually perceiving something. Since the machine has no direct connection to the user's brain, the only evidence it can have of a user's mental operations is inferred from the user's interaction with the system, such as mouse events or spoken commands. Nevertheless, with careful design, and with the ASPRN plan recognition system, GOMS models can provide a potentially rich base for merging design-time knowledge with run-time event analysis.

As mentioned above, plan recognition is fraught with challenges. In a military domain, much of the information available to the system comes from sources that may not always be completely reliable. For example, data obtained from remote sensors may be out of date by the time it is received (such as for targets that move), or may simply be inaccurate due to smoke, noise, or other obstructions.

Due to the uncertain nature of the information available to our plan recognition system, we have chosen to take a probabilistic approach. The contextual modeling (CM) system in ICF uses ASPRN (Huber, 1996) as its core recognition engine. ASPRN first translates GOMS task models into a form of Bayesian belief network called a plan recognition network (PRN). Most Bayesian networks, like the PRN, generally require significant attention to defining an accurate initial set of probabilities (priors) for each node of the network. Through the use of a number of predefined generic patterns, ASPRN is able to generate an initial set of probabilities (priors) for each element (node) of the PRN. This makes it easier to make progress on the design and development of our general framework without being overly concerned about expected frequencies of specific tasks in particular situations. Moreover, unlike many current plan recognition systems, ASPRN does not make the assumption that its task models are complete or accurate. Using ASPRN, we are thus able to rapidly iterate the task model design without worrying about brittleness.

ASPRN provides a state space for each element of the task model, such as Inactive / Active / Achieved statuses for methods and Observed / Not Observed statuses for primitive operators. Each element of the state is merely a number between 0 and 1. Interpretation of the numbers requires a separate layer of plan reasoning.

3.2a.3 Plan reasoning

Interpreting Bayesian networks, such as the network created by ASPRN, requires significant domain knowledge, such as when a number is "high" or "low". In addition to being very domain-specific, PRN reasoning algorithms need to be robust to asymptotes, evidence decay, and cross-node influences. At present in ICF, the user modeling component only contains a few heuristic rules for thresholding the output of ASPRN and some minimal temporal reasoning. While encoding the "right" task model and determining the "right" thresholds will require significant iterative design with subject matter experts (SMEs), the plan reasoning algorithms we currently have implemented should scale accordingly.

3.2b Reasoning: Situation assessment

Situation models are used by the system to understand how current world events affect survivability, mission success, and other factors of interest (e.g., Commander's Critical Information Requirements). Ultimately, situation models provide the information that allows the system to reason about how the user is contributing (or failing to contribute) to various success criteria, to which tasks the user is attending or should be attending, and with what the system could assist the user.

The situation reasoning (SR) component is responsible for characterizing the user's current situation based on a combination of user modeling and world modeling. This represents the system's assessment of how the user is performing, whether the user is working on the highest priority task, and whether the user is likely to need assistance based on the current situation. Situation modeling can be viewed as an aggregate value composed of a variety of measures including cognitive workload, the user's overall task load, the speed with which tasks need to be accomplished (versus when they need to be completed), all of the METT-TC factors, and other measures. The SR takes all these measures into account in order to derive a higher-level characterization of the situation.

Rather than developing a single measure of "goodness" for the situation, perhaps akin to the US Homeland Advisory System[1] on a smaller scale, the SR in ICF characterizes goodness as a set of situational factors, each represented by a state vector. This allows downstream components to reason in more detail about how to provide specific assistance to the user. For instance, if there is an incoming missile but the mobility situation is not good, then suggesting moving the vehicle is not a sufficient mitigation solution.

In the current implementation of ICF, the SR uses two sets of tunable heuristic rules to compute high-level state interpretations about each user task in the task model. The state information is translated into threat and urgency characterizations related to each asset capability. For example, a UGV asset may have mobility, sensor, and payload (sensor or camera) capabilities. Certain situations may imply a threat to its sensors (e.g., smoke) but not to its mobility. Each set of characteristics by asset capability constitutes a single situation vector. The set of vectors constitutes the situation matrix shown in Table 1.

3.2c Reasoning: Autonomy delegation

The mixed-initiative interaction reasoner (MIIR) is responsible for determining new autonomy adjustments. In effect it controls how tasks are delegated between the human and the system. It must consider which tasks are most critical, which most timely, which most contribute to the user's cognitive workload, which currently have the user's attention, and which need attention. It must also consider which portions of each task are best suited to human control and which to computer control, based on task and human performance characteristics, user preference and proficiencies, doctrine, rules of engagement, and policy (e.g., legal issues). While there are some tasks that will always be human- or machine-driven, a great many will vary according to the task type and situation. For those tasks where mixed-initiative interaction is appropriate, the reasoning system will help determine who has control and, in the case of the system, the level of autonomy for that task.

The MIIR takes as input the set of situation vectors from the SR and world state information from the PES monitoring agent. The MIIR combines this information with ontological descriptions of the domain and a set of tunable heuristic rules for determining how autonomy should be delegated based on specific situational factors. In each system cycle, the MIIR looks at the level of automation currently applied to each task and determines whether that level should remain as is, or be raised or lowered. The MIIR uses a combination of rule sets, including doctrinal (e.g., firing requires human approval), human performance (e.g., people do not perform well under high working-memory loads), task-specific (e.g., some steps must be performed before others), environmental (e.g., vehicles have different levels of automation available), and situational (e.g., some systems may not be functional) rules.

Whereas the SR takes a decidedly human-centric viewpoint, the MIIR takes more of a task-centric viewpoint. Similar to the distinction between performance and effectiveness, the MIIR focuses on the overall effectiveness of the warfighter/automation/platform team, while the situation reasoning component focuses on the performance of the individual. For each task that can be automated, the MIIR determines the level of automation (e.g. Parasuraman, Sheridan and Wickens, 2000) that is most appropriate during specific situations (using guidelines from Parasuraman, 2000).

---

[1] See http://www.whitehouse.gov/news/releases/2002/03/20020312-5.html for a description of this classification system.

3.3 Acting: AAM output

The output of the AAM is a task-control matrix that can be accessed by performance aids and other automation systems to determine their tasks, user interaction characteristics, and the level of autonomy at which they should be operating. Table 1 illustrates an autonomy delegation matrix for a small set of tasks.

Asset Capabilities

Auto-matable Tasks	**UAV Mobility**	**UAV Payload**	**UGV Mobility**	**UGV Lethality**	**UGV Sensors**
Evade	Medium	Low	Low	Medium	High
Attack	Low	Low	Medium	Low	Medium
Report	Low	High	Low	Medium	High

Table 1: An example Autonomy Delegation Matrix indicating autonomy levels for different asset capabilities for each of different automatable tasks at a given moment

At the end of each cycle of the AAM, control tables and system models are updated with the results of the reasoning processes and other system inputs, and given to the PES. The PES performs some validation on the new orders and translates them into command sequences for the controlled assets. In addition, the PES notifies the user, via the UI, of any planning, autonomy or initiative changes, and other critical information.

4. RESULTS

Our initial attempts at implementing the architecture outlined in the previous section helped shed light on the user interaction and adjustable autonomy requirements discussed in Section 2. This section describes lessons we have learned according to those requirements. It should be noted that these results are based on our own experiences, and not on a formal user study. In the discussion section, we identify some metrics that should be particularly useful in future controlled studies.

Reasoning requirements for adjustable autonomy

During this initial implementation of ICF (and described in more detail in DeKoven and Wood, 2005), we identified a number of problems with regard to system reasoning for adjustable autonomy, as well as about perceiving and acting.

1) Perceiving (input):

- Situational factors change quickly, particularly when there are multiple UVs being controlled. Compared to the current ICF user interface, there should be more feedback and direct user control of asset autonomy levels to support just-in-time, human-in-the-loop re-planning. There should also be better support for extended, multi-threaded, user-system planning dialogues.

2) Reasoning:

- User plan recognition and reasoning, situation reasoning, and autonomy delegation (the three primary pieces of the AAM) are crucial components for ICF, but are not sufficient to support the collaborative capabilities described in Section 2. In particular, the system needs to have significant domain and task knowledge informing the reasoning processes.
- The AAM in ICF was designed to be modular with respect to sets of heuristic rules. These rules require significant tuning, such as for timing of events and priority of tasks, and design and tuning principles should be developed. Domain experts should be able to inspect the rules.

- It is dangerous for the system to interpret the situation and make autonomous decisions without confirmation and discussion with the user. In particular, the user must be informed of autonomy adjustments and may need to be consulted in adjustment decisions. This results in the need for structured dialogue management, and user interface support for user-system negotiation.
- The user task model is the foundation for the system's assistance to the user. As a deeper understanding of user needs results from iterative design and development, this understanding should be fed back into the system's heuristics for reasoning over the task model. Moreover, changing the task model might require changing the plan reasoning algorithms in the user model.
- The ASPRN system as it stands is not designed for a work domain where user interactions will be repeated many times as part of different tasks. Once a particular action has been done, it is registered as done, and no further evidence may be collected about it. There is no support for "instancing" or reasoning about time periods. There is no "reset" function once a task has been completed, other than what we have implemented for the purpose of the year one demo.
- Accurate plan recognition requires an expanded form of contextual modeling. User modeling and task modeling should be improved, and the models should also incorporate world events, in particular to help disambiguate user actions, but also to increase the system's ability to interpret user events. This would lead to an increased ability to make predictions about the user's need for assistance, and better threat interpretation. Incorporating other predictive capabilities, such as found in systems like GLEAN, would further improve accuracy.
- Teamwork modeling and deontic implementation must be extended. Adjustable autonomy and mixed-initiative interaction appear to place new requirements on deontic reasoning, or at least new types of deontic relationship.

3) Acting (output)

- It is not always useful for the system to take control. There is a resulting need for testing with target users and other domain experts in order to identify useful automated behaviors, and to properly tune the algorithms used by the AAM and PES. User testing is also needed to identify the right times and means for user-system dialogue about autonomy and situation interpretation.

User requirements for collaborative interaction

While our current ICF efforts are not aimed at developing an optimal user interface for a specific domain, our prototype interface has made apparent a number of issues regarding the three characteristics of human-automation collaboration described in Section 2.

Expression of user intention and negotiation on plans and preferences

- The user needs to be able to request that a task be done with a certain level of autonomy. This could involve specifying report structures and requirements as to timing.
- The user needs to the able to request coordination between assets, which means that what one asset does may depend on what another asset does or senses.
- The user needs to express high-level goals and preferences that can be used to guide the planner in making decisions.
- The UI should support structured dialogue between the user and the planning system to clarify the user's intentions if priorities or details are unclear. This will facilitate the user's ability to express intentions successfully.

Human-in-the-loop automated re-planning and synchronization of actions

- The user may need to be consulted in autonomy adjustment decisions. This results in the need for user interface support for user-system negotiation of autonomy.

- The user may need to be consulted by the planning system due to unexpected events or changes in the world. The user needs notifications to be salient according to level of priority and to be understandable in context.
- The user may need access to more of the context behind a message from the system. This should be available to them if possible.
- The user needs to be able to either accept or reject decision-making tasks assigned to them by the system. The user should be able to offload decision-making to the planning system, as well as more detailed specification.
- If the system makes autonomous decisions, the user needs to be able to override them. The user needs to be able to clarify whether this decision means that the system should just change its immediate behavior, or if it should change the rule that led to the autonomous decision.
- The user needs to be able to take detail-level control of an asset, maneuver it, and later return it to the plan or task that existed before taking control. The user will not always be able to express intent to the system and may know things that the system doesn't know. Having canned, semi-autonomous maneuvers available to the user streamlines this process.
- The user needs to be able to request that any of its assets hold their positions until further notice.

Maintaining team situational awareness and shared focus of attention in a multi-threaded situation

- The user needs to be informed of events in the world that impact the goals and constraints of the plan.
- Information pertaining to the history of a single element in the situation should be collected for the user to support evaluation of the situation and prediction of the near future.
- The user needs to be informed of autonomy adjustments.
- The user needs to be able to express thoughts and observations to the system in the form of labels, NAIs, waypoints, and other types of annotation that will then be used for planning and decision-making. This helps support the user's SA by creating a digital Common Operating Picture (COP) that reduces the user's dependency on short-term memory.

5. DISCUSSION AND FUTURE WORK

Agent-based collaborative systems, such as ICF, offer much promise for reducing the complexities involved in single-operator control of multiple uninhabited vehicles in a military context. However, system intelligence does not automatically posit an effective warfighter. There remain many critical and pervasive problems that must be addressed before such systems will be truly useful. In particular, automation naturally reduces operator awareness; the goal is to make the system take things off our hands so we don't have to think about it. The result it that careful attention must be paid to designing the operator-system interaction. The operator needs to be aware *at the right level and at the right time* of the system's autonomy settings and actions taken, and the operator and system need to negotiate over the best ways to delegate authority across all assets.

The goal of ICF is to provide a layer of intelligence to support user interaction with autonomous systems. This paper described the AAM within ICF and offered some initial insights we had while integrating the AAM with the UI and PES. These efforts were only a first step towards creating the tools and mechanics necessary to support all three tiers of human-automation collaboration. The UI provides a form of perception for the autonomous systems; the AAM provides the adaptation and customization necessary to facilitate mixed-initiative interaction between an operator and autonomous assets; and the PES translates user intent into control sequences for the assets.

Given our current experience with implementing ICF, we intend to embark on the following next steps:

- *Agent reasoning explanation*. The current implementation of ICF relies heavily on Soar agent reasoning. As the situation and autonomy heuristics are extended, design and development of agents will require more robust debugging tools. It is feasible that some of the information presented to the programmer in these tools should be migrated to the operator user interface.
- *Integrated planner*. The ability of the PES to make more useful decisions is limited by its lack of ability to replan. To some degree, this capability can be improved through communication with the operator. Communication, in turn, depends to a large degree on the usability of the user interface design along the three tiers described in this paper.
- *Dialogue management*. Given the complex nature of future combat, it is apparent that there is a need for extended mixed-initiative interaction ("dialogues") between the user and the system. Moreover, there can be several simultaneous dialogues as threats and assets change status, and as the user switches between tasks, such as setting flight patterns for a UAV to evade a threat while sending reports to higher echelons or other networked entities. ICF needs a way to build a structured dialogue model within the system. Creating and utilizing this model in interaction with the user is what is known as "dialogue management".
- *Validation and evaluation*. There is a considerable amount of design work involved in the development of ICF. This work includes numerous aspects of the agent reasoning rules, as well as scenario development, task modeling, interface design, and dialogue design. These designs need to be tested with target users and domain experts in order to ensure completeness, consistency, and usefulness.

This last point is crucial in developing systems that are both *usable* and *useful*. While this paper has described three key areas for user interface design to support human-machine teamwork, we have not yet performed any rigorous user studies. Any future study evaluating a user interface for a collaborative planning system should be focused on examining the following key metrics:

- *Workload reduction and human resource conflict resolution:* Does the system effectively reduce the workload experienced by users, especially in high-workload problem areas? This can be measured in a comparative study, or evaluated through analysis and use of the system. Evaluating the task model can also highlight points of high workload and suggest strategies for workload reduction. At the same time, it is important to discover if the interface and task model assist in minimizing resource conflicts as far as users are concerned. Here is one place where an adaptive user interface has a particular advantage over current C2 products in terms of directing focus and making sure a user's hands and eyes are not required to be in two places at once.
- *Planning effectiveness:* Is the end result an effective plan? Was a plan achieved with a minimum of errors and re-planning? How many times users have to enter a course correction, otherwise clarify their intent, reverse an autonomous decision, or go back and re-do something they have already done are telling signs about how effective the user interaction and planning are. In the area of battlefield C2, this might also be evaluated in terms of effective sensor coverage, enemy detection and engagement percentages, given operational parameters, and world events. In addition, there are metrics such as how many losses are caused by the enemy in a simulated exercise, plus how much fratricide or collateral damage was incurred. Efficiency of resource assignment is a good one, but the collaboration problem is not merely an optimization challenge. Domain experts other than the user can be used to evaluate this as well. Objectively, it should
- *Task Completion Time:* How long did it take to complete the mission? How long does it take users to attend to a situation on screen? To understand it and take action? Times to complete specific goals within a mission can be good parameters for measuring planning effectiveness. If these times are long, it might be an indication that the UI is failing to support SA and decision-making.

- *Situation Awareness:* Do users know what's going on and why? Can they identify key and important elements in the situation at any one time? Do they know where they are and what they are doing there? It is especially difficult to maintain user awareness of the situation when automation is involved, especially mixed-initiative autonomy such as we've described. A usability assessment for a collaborative assistant UI should consider how accessible to the user is information about autonomy levels, autonomous actions, and reasons for autonomous decisions.

Future forces, like smart homes, intelligent transportation, and civil aviation, represent complex situations requiring adaptive interfaces for effective user supervisory control. While the optimal form for these interfaces is still not clear, ICF is creating the tools, technology and underlying scientific principles that will enable design exploration toward the creation of highly effective human-machine teams.

6. REFERENCES

1. Beard, J., Frederiksen, R., Huber, M., Wood, S. and Zaientz, J. (2003). "CIANC^3: An Agent-Based Intelligent Interface for Future Combat Systems Command and Control," *Proceedings of the 2003 Conference on Behavior Representation in Modeling and Simulation (BRIMS)*, Scottsdale, Arizona.
2. Card, S.K., Moran, T.P., and Newell, A. (1983). The psychology of human-computer Interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.
3. DeKoven, E. (2004). "Help Me Help You: Designing Support for Person-Product Collaboration", Ph.D. thesis, Delft University of Technology, Delft, The Netherlands.
4. DeKoven, E. and Wood, S. (2005), "Integrating adjustable autonomy in an intelligent control framework", Proceedings of SPIE 2005 Session on Unmanned/Unattended Sensors and Sensor Networks II, Brugge, Belgium.
5. Grosz, B. and Sidner, C. (1990) "Plans for discourse", Intentions in Communication, Cohen, Morgan, Pollack (Eds), Bradford Books, pp. 417-444.
6. Huber, M.J. (1996). Plan-Based Plan Recognition Models for the Effective Coordination of Agents Through Observation, PhD thesis, The University of Michigan.
7. Kieras, D.E., and Polson, P.G. (1985). An Approach to the Formal Analysis of User Complexity. *International Journal of Man-Machine Studies*, 22(4): 365-394.
8. Parasuraman, R. (2000). Designing automation for human use: empirical studies and quantitative models, *Ergonomics*, 43(7): 931 – 951.
9. Parasuraman, R., and Riley, V. (1997) Humans and Automation: Use, Misuse, Disuse, Abuse, *Human Factors*, *39*, pp. 230-253.
10. Parasuraman, R., Sheridan, T., and Wickens, C. D. (2000). "A Model for Types and Levels of Human Interaction with Automation," IEEE Transactions on Systems, Man, and Cybernetics.
11. Rich, C., Sidner, C., and Lesh, N. (2001). "COLLAGEN: Applying Collaborative Discourse Theory to Human-Computer Interaction". AI Magazine 22(4): 15-26.
12. US Army (2003). "Operational Requirements Document for the Future Combat Systems", Change 3, April 14.
13. Wood, S.D. (2003). "Cooperative Interface Agents for Networked Command, Control, and Communications (CIANC^3): Phase I Final Report," United States Army Research Institute for the Behavioral and Social Sciences.
14. Woods, D (2001). "Human-Centered Design of Automated Agents and Human-Automation Team Play". Cognitive-Systems Engineering Laboratory, Ohio State University.
15. Zaientz, J., Wood, S. and Hawkins, R. (2005). "Improving Information Assimilation by Modeling Warfighter Context," *Proceedings of SPIE Defense and Security Symposium 2005*, 5805, Orlando, FL: SPIE Defense and Security Symposium.