14th ICCRTS
"C2 and Agility"

Title: Symbiotic Planning: Cognitive-Level Collaboration Between Users and Automated Planners

Topic 3: Information Sharing and Collaboration Processes and Behaviors

Ron Scott, PhD
BBN Technologies
8778 Danton Way
Eden Prairie, MN 55347
952-974-3756
rscott@bbn.com

Emilie M. Roth, PhD
Roth Cognitive Engineering
89 Rawson Road, Brookline, MA 02445
617-277-4824
emroth@mindspring.com

Jeffrey L. Wampler
Air Force Research Laboratory
Human Effectiveness Directorate
711HPW/RHCS
2255 H Street
Wright Patterson, AFB 45433-7604
(937) 255-7773
jeff.wampler@wpafb.af.mil

Elizabeth Kean
Air Force Research Laboratory
Information Directorate
AFRL/RISA
525 Brooks Rd
Rome, NY 13441-4505
(315)330-2601
Elizabeth.Kean@rl.af.mil

POC: Ron Scott

88 ABW PA Cleared 3/4/2009; 88 ABW-2009-0851

Abstract

As the Department of Defense (DoD) moves into a more agile C2 environment, added requirements will be laid on operators and systems – integration with outside systems and organizations, operation in an environment where rules change daily, and severely limited resources. We foresee an increased reliance on automated support tools – optimizers, schedulers, evaluators – to meet these requirements. These automated tools need to be closely coupled with an operator interface, allowing the operator to:

- Task and constrain the automated tool,
- View and understand what the automated tool has done, and
- Override, correct, and retask the automated tool.

This paper describes a mission planning/execution work aid coupled with an automated scheduler. The prototype has been designed using a cognitive approach which concisely presents critical information to allow the operator to rapidly understand and assess the planner's recommendations. The operator can finely control the scheduling tool, setting constraints and bounds on what the scheduler may affect. The operator can evaluate multiple options suggested by the scheduler, and can retask the scheduler to modify its work. This symbiotic prototype allows the operator and automated tool to collaborate at a cognitive level to arrive at a better solution than either alone could find.

Introduction

This paper describes applied research being sponsored by the Air Force Research Laboratory (AFRL) on Air Mobility Command C2 under two separate advanced technology demonstration programs, Work-Centered Interface Distributed Environment (WIDE) and Global Response Synchronization (GRS). The paper will first briefly discuss our cognitive based approach to developing work aids called Work Centered Support Solutions (WCSS), which has previously been described in (Wampler, et al., 2005). In the current research project, we have extended the WCD methodology, applying it to designing a prototype system including interaction between a human user and an automated scheduling support tool. We describe in some detail both the problem we've chosen to attack and the design of the prototype system we've implemented. We conclude with a discussion of the characteristics of C2 problems amenable to solution by variations on this design concept.

Work Centered Support Solutions

Work-centered support solutions (WCSS) have been a focus of AFRL research and development projects for nearly a decade. WCSS facilitate C2 decision making performance by exploiting advanced visualization and automation to keep users on topic, focused, and well-informed. More specifically, WCSS enhance visibility of mission-critical decision factors, depict relationships between mission plan elements and constraints, and highlight problematical conditions.

A WCSS is a client that "plugs" into net-centric data services, links to the right information, and "knows" how to effectively present it (Young *et al.*, 2000). A WCSS affords its

user(s) an optimal 'window' onto their work subject matter, in the sense that it is tailored to provide effective information support for the task(s) at hand plus the functions necessary to accomplish the users' work activities. WCSS visualizations depict the problem space from a user's perspective, including constraints and possibilities, so as to aid user memory and "suggest" solutions (Eggleston & Whitaker, 2002). We fit automation into the natural flow of work activity by using intelligent agents to find, fuse and embed relevant information in these visualizations. In a good WCSS design, the user can address essential elements of work on the same terms as he / she cognitively engages them. By facilitating attention to the work and not the system or tool itself, a WCSS affords the user improved situation awareness on his/her overall workstream, the facts of particular 'cases' within that workstream, and the state of the task(s) at hand in processing each 'case'. In other words, WCSS supply the right data for the task at hand, minimize overhead associated with finding and fusing this data, and effectively convert this data into actionable information.

WCD's goal is to generate effective WCSS designs by crafting data visualization, navigation, and manipulation to reflect and accommodate actual work practices. This is the optimum approach for effectively supporting the key components of information-intensive C2 tasks - decision-making, collaboration, work management and work product generation (Eggleston, & Whitaker, 2002). Our WCD methodology builds on work in cognitive engineering (e.g., Rasmussen, 1986; Woods and Roth, 1988) and cognitive work analysis (e.g., Vicente, 1999) to overcome prior limitations in applying cognitive research to practical settings (Eggleston, 2002, 2003). We conduct WCD as a team incorporating 3 expert roles - cognitive engineer, WCSS designer, and software engineer. All 3 roles actively participate throughout the WCD process.

An important product of the work-centered design process is a set of system software requirements. While, of course, there are numerous other requirements analysis processes, the requirements produced as output of work-centered design tend to be quite different. Our work-centered design process in some ways considers the human operator and software together as a combined system. The requirements we produce concentrate on the ability of the human operator to use the software to effectively perform the high-level tasks required by his day-to-day duties, as opposed to more traditional software system requirements which generally treat the software systems as black boxes – maybe dependent on user input, but not in any way in collaboration with the user.

Problem Domain Background

Air Mobility Command (AMC) conducts centralized command and control of airlift and air refueling assets in an air operations center (AOC) called the Tanker Airlift Control Center (TACC). The TACC schedules and tracks strategic tanker and airlift resources worldwide. Air Force and Department of Defense support taskings are channeled through this state-of-the-art mobility C2 hub. It is a global AOC with several hundred people planning and executing hundreds of air missions per day.

Mission planning is an activity undertaken by AMC in response to United States Transportation Command (USTC) requests to move military equipment and personnel by air. Mission planning is a complicated activity that must take into account issues such as matching loads to currently available aircraft, landings in and over-flights of foreign nations, competing airlift demands, airfield constraints, air refueling requirements, and aircrew constraints.

There is a basically linear process path leading from mission planning through detailed planning of mission factors (e.g., routes, permissions) to mission execution. The focus of this work is an operational domain that is both complex and dynamic. Changes may occur at any time, and the range of actionable events or conditions is essentially unconstrained. Relevant decision making factors are numerous, situation-specific, and interrelated in complicated ways. Though planning may have begun months in advance, it is not until a few hours prior to launch that a plan can be finally evaluated for adequacy and feasibility. Once the mission is in progress, C2 becomes an extremely time-sensitive and time-critical function.

These operations are conducted by a staff of specialists. Mission Planners conduct initial mission planning in response to incoming USTC requirements. As time goes on, other specialized roles contribute detailed specifications to this plan. For example, Barrelmasters assign aircraft, Aerial Port staff work out cargo logistics, diplomatic clearance (DIP) specialists obtain overflight permissions, and Flight Planners lay out details of routing and scheduling. As launch time approaches Flight Managers finalize flight plans and assemble aircrew documentation, including weather forecasts generated by meteorological specialists.

Twenty-four hours prior to planned mission launch responsibility for the mission is transferred to the Execution Cell that handles any last minute changes and deals with problems during mission execution. The Execution Cell is manned by Duty Officers (DO) and enlisted controllers that are responsible for identifying, tracking and resolving problems. Typically the enlisted controllers receive calls from pilots or the wings alerting them to problems (e.g., aircraft maintenance problems; delays in loading cargo; unanticipated changes in cargo; airfield closures). Difficult cases are escalated to a DO who is responsible for identifying potential repercussions for the current mission as well as follow-on missions and modifying mission plans to resolve any problems identified.

Legacy computer systems support management and execution of TACC airlift and air refueling missions. They are intended to provide accurate, near-real-time data required for making decisions on effective use of AMC resources. This data is presented to TACC users via Microsoft Windows compliant user interfaces. However, simple presentation of all available data does not ensure effective support for TACC users' work activities. Critical information is not always available at the right time or in the right format to support decision making and other cognitive work activities.

In this paper we will describe a particular set of tasks to be performed by duty officers in the TACC which are not particularly well-supported by the existing C2 system. We describe our initial design for a WCSS to better support these tasks. We then broach the main topic of this paper – how extending this WCSS to have the support of an automated scheduling tool led us to a particular augmented WCSS design which includes the interaction of our human operator with the automated tool. Finally, we describe the unique features of our problem that drove us to this symbiotic design as an attempt to generalize the interaction and visualization techniques into design principles which can be applied to other domains.

Initial Problem Description

Duty Officers (DOs) working on the TACC floor are tasked with monitoring and correcting issues that arise with air missions currently (or soon to be) in execution. The primary tool to support their work is the Global Decision Support System (GDSS) – AMC's main C2 system. GDSS, in various versions, has been operational for well over a decade. It consists of a large distributed database, containing information about air missions, airfields, aircrews, and

various other domain objects. User interfaces are largely tabular, although there are some graphical elements.

Air missions are complicated objects. Each air mission has multiple planning factors that interact with each other and constrain each other. These factors include type of aircraft, type of cargo, countries to be flown over, and airfields where intermediate stops will occur. Any change in one factor can have repercussions for other factors. For example an unexpected change in cargo (e.g., the presence of unplanned hazardous material) can influence the viability of the planned route because some countries do not allow flights with such cargo. Similarly, a delay in the flight (e.g., due to an aircraft maintenance requirement) can have further repercussions for a mission. For example, it may cause a flight to reach an airfield after it has closed or result in a violation of air crew duty day limitations.

The initial mission plan generated by the mission planner takes these multiple factors into account. However, unanticipated changes can arise between the time the mission is initially planned and when it is executed. The DO has to be able to understand the goals and constraints in the mission plan initially developed by the mission planner and assess the implications of these last minute changes for the viability of the flight. GDSS does not effectively support the DO in assessing the impact of changes on the viability of a flight plan and revising the plan appropriately. Although all the relevant data is available, it is presented in a tabular form that makes it difficult to get a 'holistic' understanding of the elements of the mission plan and the objectives and constraints that underlie it.

Let us illustrate this situation with regard to the concrete example of a mission delay. It is quite common for a DO to receive a telephone call from a pilot in the field, informing him that his mission is going to incur a delay. It may be that there is an aircraft maintenance problem to be addressed. It may be that cargo or passengers had been delayed, so takeoff is going to have to be delayed as well. It is the DO's responsibility to essentially replan this mission on the fly – to reschedule the current leg and all further legs of this mission in such a way that the schedule will be legal with respect to all the various planning factors and constraints. Crew duty day limitations, airfield operating hours and quiet hours, airfield parking capacities (referred to as MOG – maximum on-ground), any relevant permissions and clearances, required delivery dates of any cargo – all these must be taken into account and their constraints satisfied, if possible. And if not possible, certain of these elements can be renegotiated, and again it is the responsibility of the DO to make this happen.

GDSS offers the DO the opportunity to view any of these data elements. A mission summary screen will show the scheduled times of departure and arrival at each airfield. Each airfield can be brought up, in a separate window, to view the airfield operating hours. A diplomatic clearance summary window can be brought up to show each of the diplomatic clearances allowing overflight of countries, each with its own valid time window. The DO will need to perform mental arithmetic to adjust for any potential delay to decide which clearances might be violated. A separate tool, in a separate window, can give a summary of how many planes are on the ground at any time at any particular airfield. An analysis of when and where airfield MOG capacities might be exceeded can be done using this tool, but again, it will have to be done multiple times, with multiple windows on the screen for multiple airfields. All of this is to be handled, of course, in an environment of constant interruption by phones, by co-workers, by superior officers all drawing attention to other, higher priority, problems.

While we have not seen the requirements specification for GDSS, our experience with such documents leads us to believe the requirements are of the form: *The system shall be able to*

display planned departure and arrival time for each leg of each mission. The system shall be able to display operating hours for each airfield. The system shall be able to display the valid time window for each diplomatic clearance for each leg.

Our contention is that these requirements miss the mark. Each of these is a valid, and necessary, requirement. But a system that meets all these requirements is not necessarily a system that supports its user in replanning missions on the fly. These requirements need to be augmented with the higher-level cognitive requirements that we produced in our work-centered design process:

- The user must be able to use the system to simultaneously view all the basic planning factors and constraints for all the legs of a mission on a common referential context (e.g. time), and be alerted to any constraint violations within that context.
- The user must be able to use the system to quickly do a what-if – i.e., reschedule a mission and immediately see the effects on planning factors and constraints within the referential context.

As we have performed knowledge acquisition activities within the TACC, conducting many hours of interviews and observations covering a wide range of user subjects, one additional critical factor has been made clear to us. There are effectively two kinds of information flows within the TACC. There is the formal path, in which information is captured, represented, and displayed in GDSS, the existing C2 system. A DO will look at the representation of a particular mission in GDSS and be able to find all the basic mission information.

There is also an informal path – mechanisms by which extra information may be made available to the DO. There may be information about the cargo carried on this mission, and its relative priority. There may be special requirements attached to a mission, or maintenance information about the airplane assigned to the mission. Any of this information might come from telephone conversations, from shared notes, from several home-grown software tools. This extra information may override information from GDSS. This extra information may even repeat information from GDSS (in this case it might be considered that the extra information is really the meta-information that this fact here is a critical bit of information about this mission, and its role is really that of a cue to draw the user's attention). Frequently, the extra information may be stored in GDSS in unstructured textual form, in "mission remarks". In this case, the DO might easily not notice an important bit of information lost in a sea of standard remarks.

We have seen numerous artifacts in use in the TACC for users to save, keep track of, and share this extra information. We've seen users use sticky notes attached to the edges of their terminals. We've seen users use a formally organized text document shared among cooperating users. We've even seen a graphic mission display with allowance for graphical and textual annotations to manage this information. But a central truth about how work is done and decisions are made in the TACC is that this "extra" information is often the deciding factor when it comes to evaluating replanning options. Any decision support tool that hopes to help improve replanning decisions must explicitly provide support for entering, viewing, and drawing the user's attention to these critical decision factors.

Initial Design

Based on the requirements listed above, an initial prototype of the WIDE timeline GUI was designed and implemented. A series of empirical evaluations of the prototype were conducted to compare targeted users' replanning performance on multi-mission synchronization

problems using the new tool vs. using a comparison tool representative of their current computer systems. The results revealed statistically significant improvements in solution times (three times faster), quality of solution (a third as many errors), situation awareness and workload (Roth, et, al. 2006) (Roth et.al in press).

The WIDE timeline GUI was designed to serve as an alternative viewing mechanism for air missions described in the GDSS database. The intent of the design is to let multiple air missions be viewed simultaneously, with each air mission showing indication of any planning constraint violations. There are mechanisms for drilling down into details of mission data to better understand constraint violations. A key functionality of the design is an edit mode allowing a dynamic 'what-if' capability. Users can click and drag on any number of sorties to "visually" reschedule them within constraints while the effects of these changes on constraint violations are dynamically updated on the screen. The user basically drags the sortie(s) until the display is violation-free. This approach drastically reduces the user's cognitive burden (mental math) associated with calculating multiple constraint violations and mentally "fusing" violations from multiple displays as schedule repairs are made.

The timeline graphical user interface (GUI) consists of three linked displays–the multi-aircraft timeline view, a multi-airfield timeline view, and a compact timeline view. These displays are linked in the sense that they are different views of the same data. Any modifications to the data in one display will be reflected in the other displays.

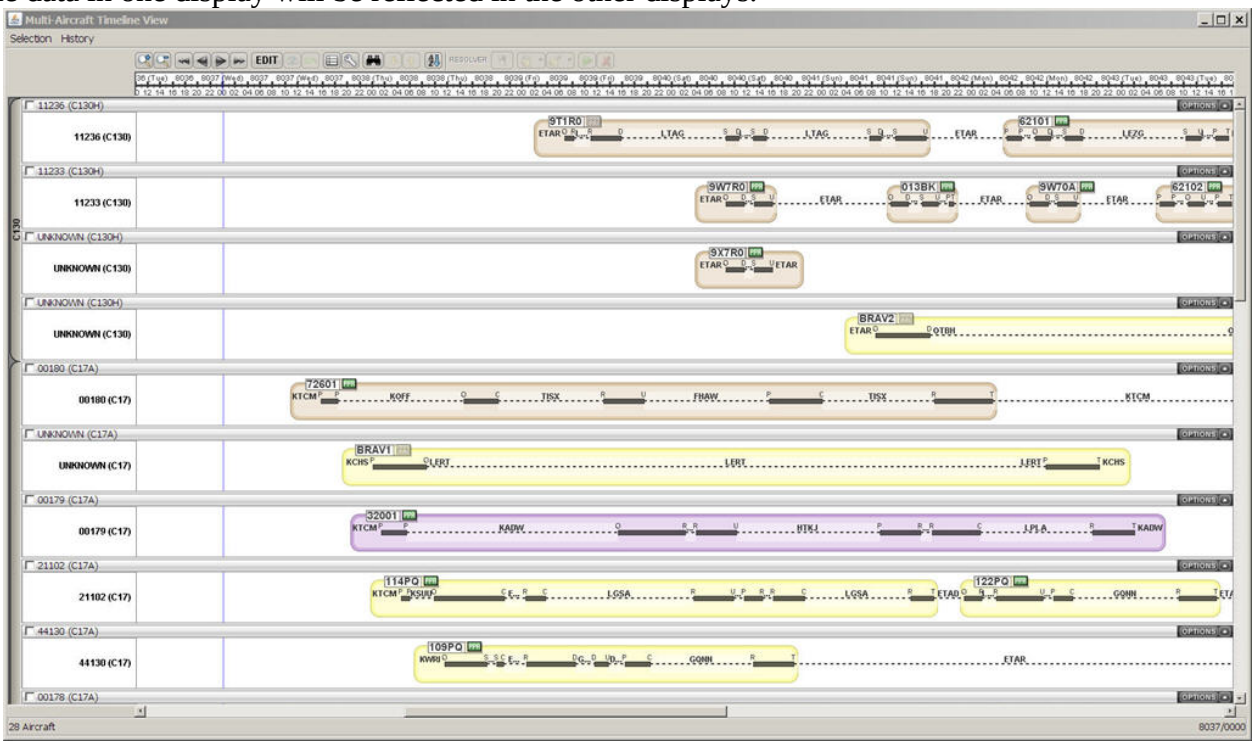

Figure 1

The multi-aircraft timeline view (see Figure 1) is organized around aircraft. Each main horizontal slice of the display represents a single aircraft showing all the missions that aircraft is scheduled to fly. Each mission is enclosed in a colored envelope, color-coded by a user-specified characterization of the mission's type. Within the envelope, each sortie is represented as a dark gray bar, reflecting the planned takeoff and landing times for that sortie. Ground legs–

periods during which the tail is on the ground at an airfield are represented as dotted lines, with the ICAO symbol for the airport labeled. Also available in the default mode of the multi-aircraft timeline view are the purpose codes for each takeoff and landing, giving an indication not only of the purpose of the mission, but of any necessary ground activity.

By clicking on the "Options" button at the right-hand side of each single aircraft timeline users can access a menu that will let him expand the data available for that timeline. Additional data clusters include aircrew data, airspace data, cargo requirements data, airfield parking MOG, and airfield operating and quiet hours. In Figure 2 we've expanded data clusters for airspace, aircrew, and cargo requirements, and have zoomed in in timescale to see more detail for a particular mission. We can see which aircrew is tasked with flying this mission, when they must report for duty and when they are expected to complete their duty day. We can see the expected pick-up and delivery times for cargo, and the fact that the current plan affords a buffer of about eight hours before the cargo must be delivered. We can also see windows of times in which the mission is projected to be overflying which countries, and in some cases the windows of diplomatic clearances covering those overflights. Finally, we can see that the current plan takes this mission over the country of Mali about eight hours earlier than the diplomatic clearance calls for.

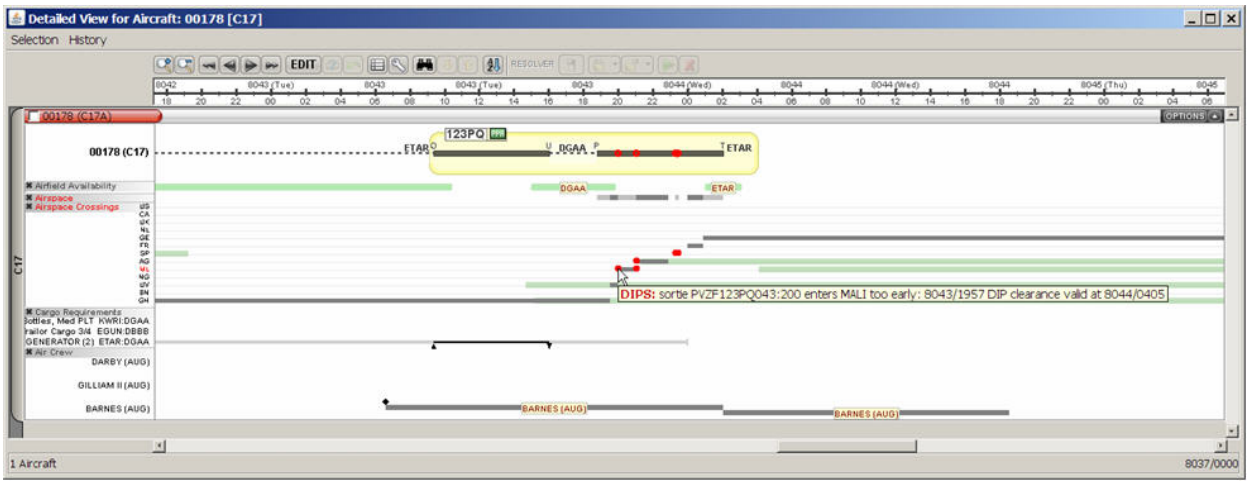

Figure 2

The multi-airfield view gives a different view of the same air missions (see Figure 3). This display is organized around the airfields. Each airfield there has a summary line showing the parking MOG status for that airfield. MOG data comes to us from another TACC system, known as the Station Coordinator tool. If the airfield is one that is managed by the Station Coordinator, we have information about both narrow-body and wide-body MOG capacities (as well as what tail types are considered to be narrow-body and wide-body for this particular airfield). In that case, the MOG status of the airfield will be summarized by two lines–one showing the narrow-body MOG status and one showing the wide-body MOG status. At any particular time, the line will be colored gray (at least one parking spot available), yellow (zero parking spots available), or red-brown (meaning the airfield is over MOG capacity for the body category). For airfields not managed through the Station Coordinator tool, we have no such information for either body category or capacity. For these airfields we have a single line,

displaying various shades of gray (together with a number) that depicts directly how many tails are on the ground at that airfield at that time.

By clicking on the appropriate MOG bar of a Station Coordinator-managed airfield, users can drill down to see the missions being flown for each tail that is on the ground during the period depicted in this view. The ground time for this tail at this airfield is highlighted. In this way, users get an immediate view of which tails are on the ground. Just as in the multi-aircraft view, users can click and drag sorties and missions, with the effects on the MOG situation, as well as any other constraint violations, being immediately visible. Using this facility, users can try to reschedule to avoid MOG conflicts while staying aware of any other problems this rescheduling introduces into the overall plan.

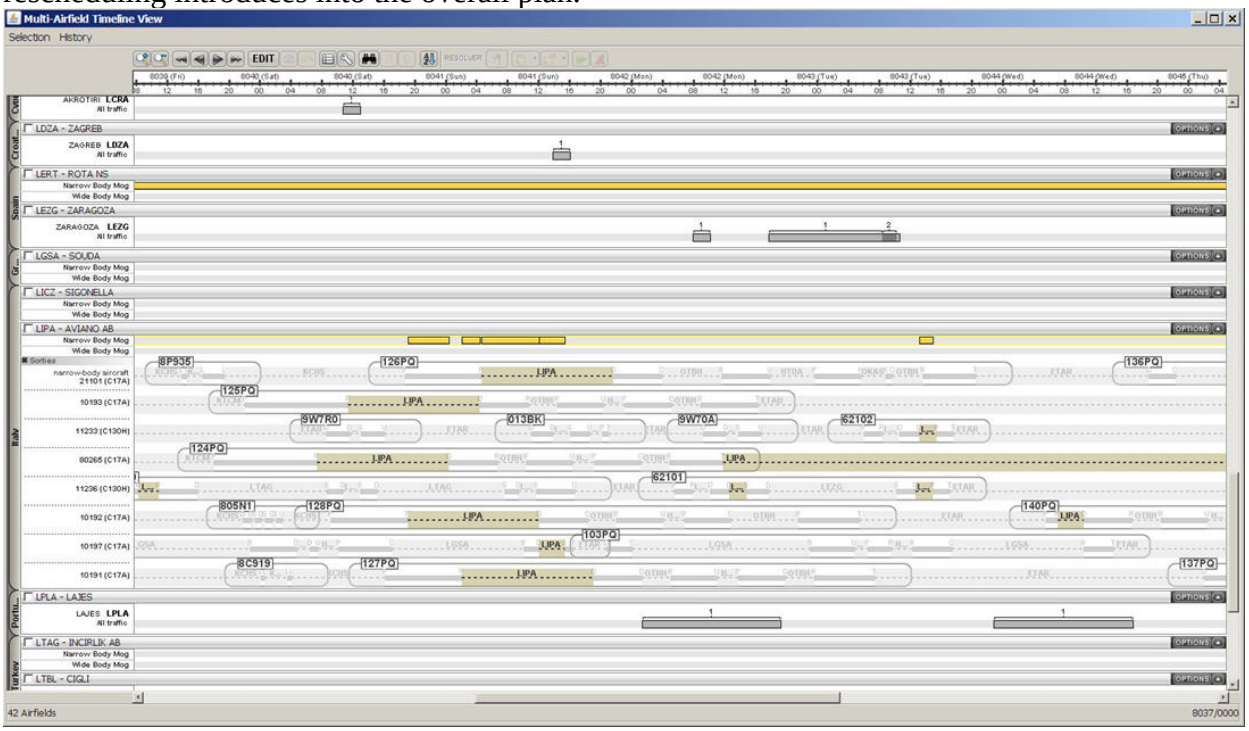

Figure 3

The compact timeline view (see Figure 4) is a schematic view of the same missions displayed in the multi-aircraft view. As in the multi-aircraft view, the display is organized around tails. Each horizontal segment depicts one tail and the missions it is scheduled to fly. Displaying only missions, and not individual sorties, frees up the space to include useful data. The sequence of airfield ICAOs the tail will be transiting is displayed within each mission envelope. There is also a space allocated for description of this mission's cargo.

This compact timeline view, since it has less detail (and hence more available pixel space), is also our primary view in which extra annotation information may be entered and viewed. With a right mouse-click, a user may choose to add an annotation, either to an individual mission, or to an individual aircraft. The annotation is entered by the user in a rich text format (allowing the user to use color and/or font to emphasize certain information). There is also a facility to add *annotation templates*, new annotations that are likely to be useful for multiple missions or tails.

Once an annotation template has been added, the user can add an annotation of this type to a particular mission using a drop-down menu choice.

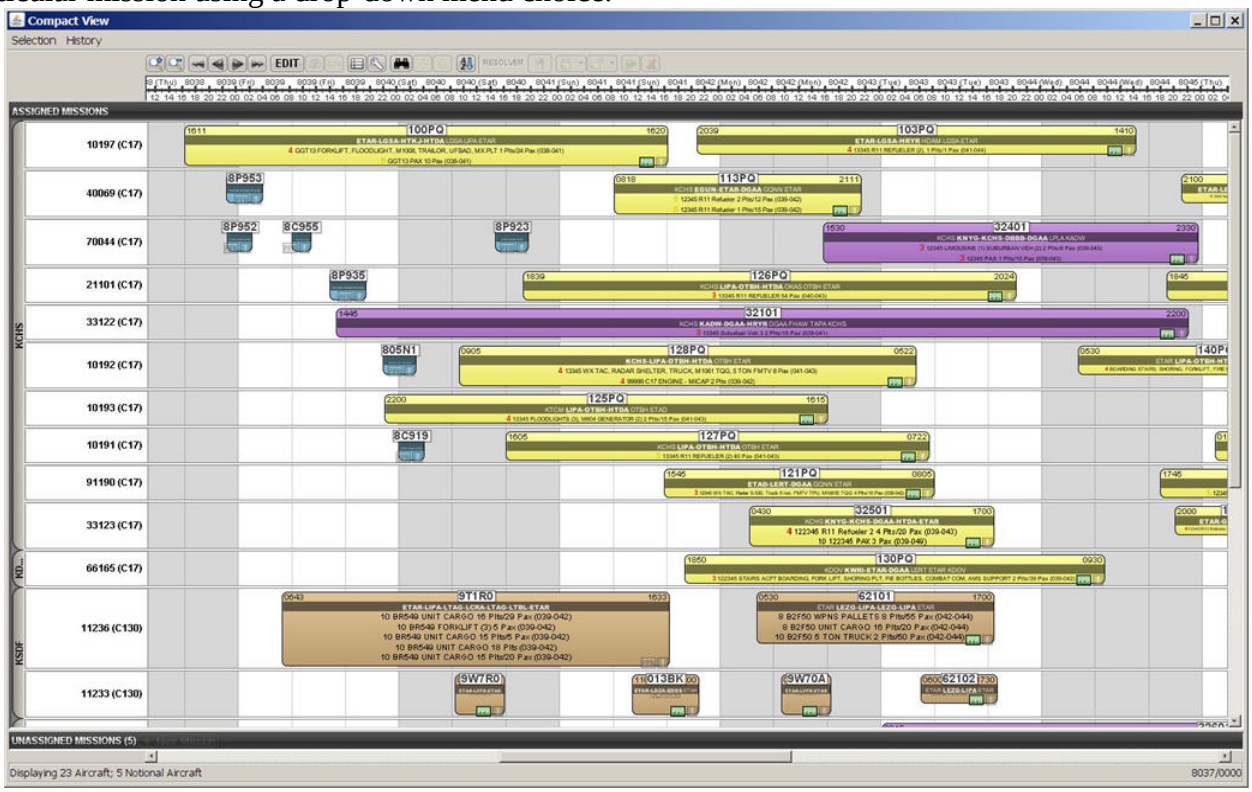

Figure 4

The compact timeline view, while the only view in which annotations can be entered, is not the only place annotations can be viewed. While there is not enough space to include full annotations in the multi-aircraft and multi-airfield views, the presence of an annotation will be noted by an icon near the annotated object. With a mouse-click, the user can bring up the full annotation from such an icon. The critical factor from the cognitive point of view, though, is that even in these other views, the user is still cued to the fact that there is an item of interest about this mission (and can take further action to get details about what that item of interest is).

To summarize, and illustrate the use of this GUI design, we will refer back to the previous example of a DO receiving a telephone call from a pilot in the field informing him of a mission delay. The DO, using the multi-aircraft view, would click and drag the appropriate sorties of the mission under consideration to reflect the new proposed schedule. Any constraint violations (the aircrew flight duty period being too long, or sortie trying to land at an airfield outside of its operating hours, for example) would immediately be marked in red on the multi-aircraft timeline view. The DO could continue to move sorties around until a conflict-free solution is found. If there are MOG constraint violations, the same procedure may be followed, but typically the situation is a little more difficult – MOG constraint violations involve multiple aircraft, so the appropriate solution might need to affect missions on any or all of those aircraft. In this case, the DO might switch to the multi-airfield view, looking in detail at the missions involving aircraft on the ground at any airfields where the MOG constraint has been exceeded. In the multi-airfield view, the DO can again click and drag to move any or all of the missions he sees to try to deconflict the situation.

Augmented Problem Statement

As we evaluated the effectiveness of operators using our initial prototype design it became clear that, while operator effectiveness was greatly improved using this prototype, there were still opportunities for further advances. As operators were attempting to solve replanning problems like the one described in the previous paragraph, involving MOG constraints, situations often arose in which operators were reduced to manually hunting a large search space for rescheduling solutions. Our initial prototype at least made this manual search feasible – the what-if mode allowed the operator to graphically move missions on the timeline and immediately see the consequences. But an automated planner that could automatically find solutions to such constraint problems would certainly provide for both quicker and better replanning solutions.

On Time Systems (OTS) had previously demonstrated the Distributed Worldwide Aeronautic Route Planner (DWARP) as an extension to its existing optimized aircraft-routing technology. We began a collaboration with OTS to marry their DWARP system with our initial prototype, using DWARP's optimization technology as an automated rescheduling tool to solve our replanning problems.

Optimization technologies had previously been applied to applications within the TACC, with mixed success. During knowledge acquisition trips we had heard many stories from TACC users of research programs applying optimization techniques within the TACC, producing tools that users were unwilling or unable to use. In the summer of 2006, just before the GRS program got started, members of the BBN knowledge acquisition team were present at a briefing where an experience AMC user listed his difficulties with the "plan-set generator" in AMC's main mission planning system Computer-Aided Mission Planning System (CAMPS).

1. There is no way for users to compare the new plan set (output of the optimization tool) with the old plan set.
2. There is no way for users to constrain the operation of the optimizer–to say, for example, leave this particular set of mission plans untouched.
3. There is no way for users to evaluate the operation of the optimization tool–is the new plan set better in some way than the old one? The user just has to have faith that the tool is operating as designed, and is improving the plan set.

This feedback from a CAMPS user, together with our understanding of the cognitive environment of the TACC floor led us to lay out a number of design goals for this project combining DWARP and WIDE technology. These goals were seen as the key constraints in our design that would let us produce a capability that would not only be effective for our users, but would be able to be trusted by our users.

Design Goals

1. Our primary goal is to provide an automated planner to assist users in solving complicated multiple-mission rescheduling problems. The tool will deal with critical planning factors including MOG constraints, crew-duty-day issues, airfield operating hours, diplomatic clearances, and minimum ground times.
2. Users must be able to understand the solution(s) provided by the automated support tool. It must be readily apparent what missions have been rescheduled in what ways, what

constraint violations have been corrected as a result of the rescheduling, and what constraint violations still remain.

3. Users must be able to compare schedules provided by the optimizer with the original mission schedules. This helps users understand, and ultimately trust, the solutions that the tool generates.
4. Users must be able to override particular decisions made by the automated support tool, for example designating some individual constraint violations as unimportant (not to be taken into account by the tool) or some individual constraints as waiverable. The user is the authority, and is always likely to have information relevant to the solution that the system does not. This extra information might take the form of understanding the relative priority of cargo or passengers on the planes; it might be an order dependency between two missions; it might just be duty-officer-level information on airfields for which a MOG waiver or quiet hours waiver might be obtainable. [Note: while this was one of our initial design goals, we have not fully addressed this in our prototype solution. It does, however, remain as a primary design goal in any continuation of this work.]
5. Users must be able to receive, visualize, understand, and compare multiple possible solutions returned by the automated support tool. Any single solution may turn out to be unworkable or otherwise undesirable, for reasons that a duty officer might know (but the automated planner would not). Giving users the ability to see and compare multiple possible solutions increases the chances of finding a truly desirable solution.

Augmented Design

In our augmented timeline GUI design, we have made a number of modifications to enable the user to collaboratively solve a problem with the DWARP automated support tool. To outline the added functionalities, we'll pick up our canonical use case where we left it off. A DO is working on a mission replanning problem, initially cued by a telephone call from the field informing him of a new mission delay. The DO uses our initial design, in the steps we've already described, to try to find a good solution. But (in the kind of situation that prompted us to begin on this augmented design) there are MOG constraint violations – possibly at multiple airfields, and certainly involving multiple missions. Even with the WIDE timeline GUI click-and-drag capability, it is not easy to find a violation-free solution. This is exactly the case that our augmented GRS timeline was designed for.

There are two major steps to using the DWARP automated planner to help solve this problem. The first is to describe (and appropriately constrain) the problem to DWARP. The second is to evaluate the possibly multiple solutions that DWARP proposes.

Our first design task was to find a way for the user to appropriately constrain the problem being sent to DWARP. The problem needs to be constrained in two distinct ways. The first is to limit the set of air missions that are to be "in play" – those missions that can be altered at all by DWARP. The first observation to be made is that in order to produce a correct solution, DWARP needs to see all the missions currently in operation – i.e., it would not suffice to just tell DWARP to operate on the missions currently on the user's screen. DWARP is designed to reschedule missions to fix constraint violations. Since MOG constraints, are essentially resource constraints (the resource being parking space at an airfield), to not tell DWARP about some of the resource's consumers (i.e., some of the missions) would increase the likelihood that DWARP would find incorrect and unworkable solutions.

But even though DWARP will need to see all the current missions, it still needs to have limits on which missions it can consider for rescheduling. The organizational structure of the TACC demands this. A single DO is responsible for a subset of current missions. A DO would like to find a solution that requires changes to only missions that he is responsible for. If no such solution exists, he may widen the scope of missions and try again, but would not want to unnecessarily reschedule missions out of his control. It is also true that rescheduling missions is not free – each rescheduled mission requires re-coordination among various parties, both inside and outside the TACC. As a rule, the DO would like to find a solution that minimizes the number of missions to be rescheduled.

There is a second way in which a rescheduling problem can be constrained. For each mission that is considered to be in-play (moveable by DWARP), there may still be individual time constraints that need to be honored by DWARP. An air mission is a sequence of sorties (a sortie being a flight from a takeoff to a landing). Each takeoff and each landing might have some real time constraint. In our canonical example with a pilot calling in from the field, he may be telling the DO that the takeoff from his airfield cannot be any earlier that 1400 today. The DO needs to be able to enter such time constraints on individual mission events, view them on the timeline GUI, and send them to DWARP with a problem definition.

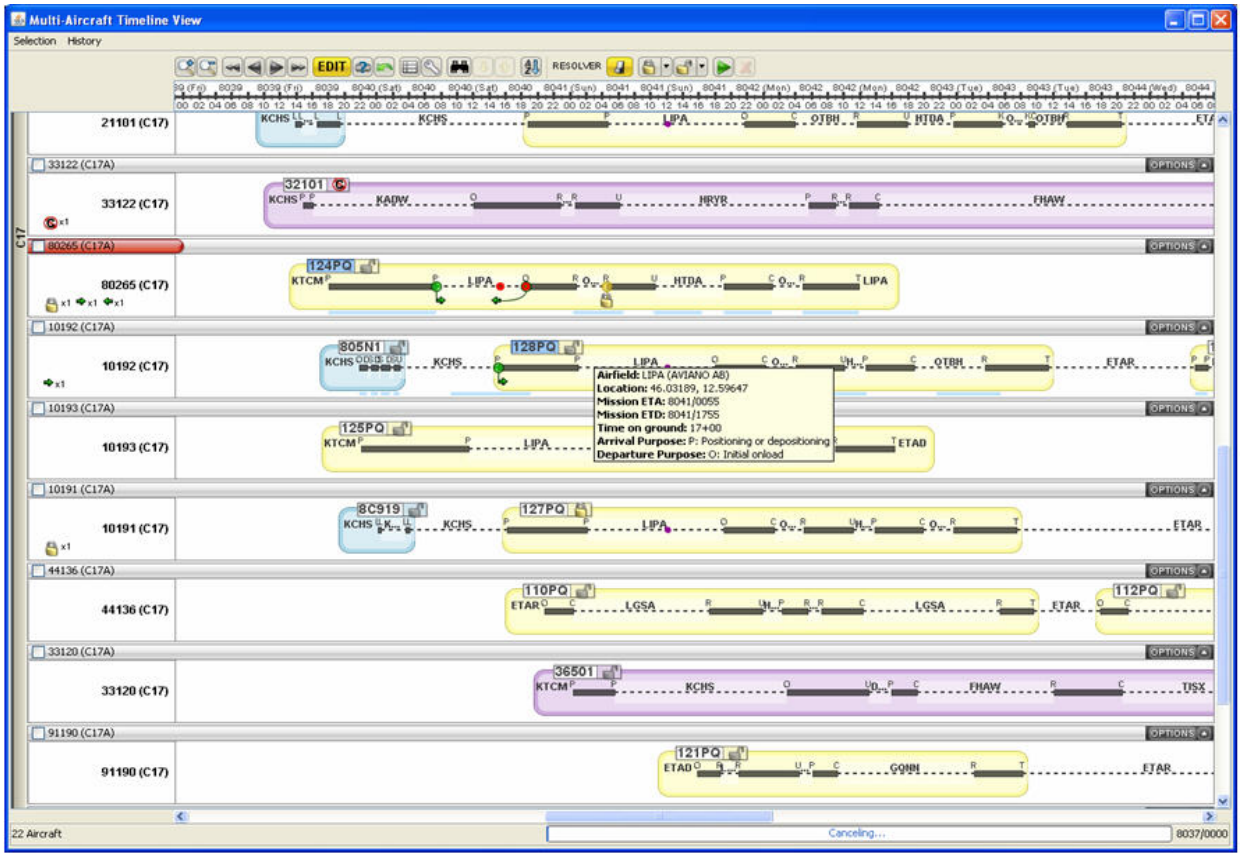

Figure 5

Constraining the Problem - Resolver Mode

The WIDE timeline's *Edit Mode* lets users alter the mission schedules for each aircraft. For the augmented GRS timeline, we have added a new *Resolver Mode*, which is the user's entry to interaction with DWARP. With *Resolver Mode* enabled, the multi-aircraft timeline view shifts from an interface where users can reschedule sorties and missions to one where they can add or remove additional constraints to DWARP. DWARP is already constrained by each aircraft's schedule. *Resolver Mode* provides locks and bounds so that users can add their knowledge about the scenario not represented by the data to further constrain DWARP (**Error! Reference source not found.**).

When *Resolver Mode* is enabled, each aircraft's timeline is repainted with an unlocked padlock icon next to the mission number. The unlocked icon indicates that the mission as a whole has no additional constraints. Right-clicking on the unlocked icon displays a menu with three options: disallow cancellation, lock an entire mission, unlock mission. Disallowing cancellation of a mission tells the *Resolver* that it can make changes to the mission but must not cancel it. This locking capability corresponds to the first type of constraint mentioned above.

Each sortie departure and arrival can be locked or given upper or lower bounds. If a sortie departure or arrival is locked, the *Resolver* must not change the departure or arrival time at all. If there is a lower bound on a sortie departure (green arrow pointing to the right), the *Resolver* can only consider schedules where that sortie departs at or after the time where the green arrow is displayed. If there is an upper bound on a sortie departure (green arrow pointing to the left), then the *Resolver* can only consider schedules where that sortie departs at or before the time where the green arrow is displayed.

Locks or bounds for a sortie event are added by right-clicking near the event (departure or arrival). A menu will appear with the options to lock the event, clear locks/bounds, add an upper bound, and add a lower bound **(**Figure 6). Locks or bounds can be changed or removed by right clicking on the green-circle around the event to which they are attached.

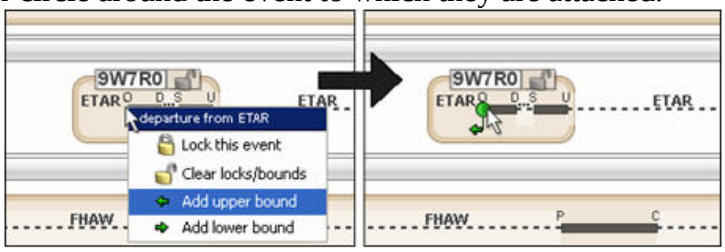

Figure 6

Users can assert more control and tightly manage the Resolver by adding locks and bounds in Resolver Mode. To quickly add or remove locks to all missions based on Mission Category or JCS Priority (a mission priority scheme in use in the TACC), users can use the lock buttons on the Resolver toolbar, which is less time consuming to adding locks to individual sorties and missions. The *Mission Category* button will let users lock or unlock all missions of a certain type. If all missions of a certain category are locked, then the "Lock by category" menu will show lock icons next to that category. The *JCS Priority* button lets users lock or unlock all missions above, at, or below a given priority. Again, if all missions of a certain priority are locked, then the "Lock by JCS" menu will show lock icons next to that priority. In Figure 7, a lock icon appears next to 1B3 because all missions with priority 1B3 are currently locked. If there are no missions in the dataset that match a certain Mission Category or JCS Priority that

appears in these lists, then a lock icon will never appear next to that option. For this reason, none of 2A1, 2A2, 2B1, 2B2, and 3A1 priorities have lock icons.

Finally, the problem can be sent to DWARP by clicking on the green arrow button in the *Resolver* toolbar. WIDE will then send the schedule and locks to DWARP and wait for its response before displaying the *Resolver Results.* If the user wishes to cancel this process, he may click on the *cancel* button (the red X in the toolbar).

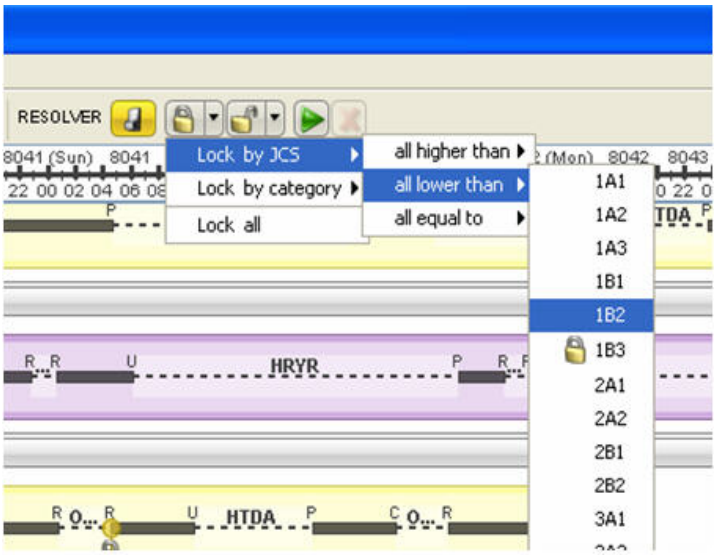

Figure 7

Evaluating Proposed Solutions

The second step to using the DWARP automated planner to solve a problem is to evaluate the proposed solutions returned from DWARP. As stated above, it is often true that the key bits of information that let a DO decide between possible solutions are not contained in the GDSS C2 system (or are in GDSS only embedded in long textual remarks). In either case, these important tidbits of information are not amenable to automated processing, and are not evaluated by DWARP. This has two important implications for the design of our system.

Firstly, we need to have DWARP return multiple potential solutions to us, instead of performing a traditional optimization relative to a scoring function and returning only the best schedule to us. The reason is simple. Any particular solution, no matter how high scoring, might have a feature that makes it unworkable. The DO might have the extra information about priority of one particular mission that makes a solution that delays that mission unpalatable. For example, a DO will not likely accept a solution that delays a mission carrying a 3-star general. He will reject that solution in favor of one that may be suboptimal in other ways (and consider himself lucky that he noticed the pitfall before he made it happen).

Secondly, our design must be geared to let the user quickly browse the potential solutions, understand the effects of each, compare the solutions to each other. To the extent possible, we need to make it easy for the user to see which missions are rescheduled by which options, and which of those missions have factors that make those changes unworkable. To that end, we have designed a tabular format for presenting DWARP's proposed solutions to the DO.

After the user queries DWARP for solutions, the WIDE interface will respond with a table summarizing all of the resulting options (see Figure 8). This display contains two major parts: the tabular summary of solution options (this consists of the white tabular component, and its two gray headers); and summary information about the mission set and solution options with buttons that invoke more-detailed visualizations of these options.

The tabular component of the *Resolver Results* display is broken into three parts: the white summary grid with icons representing schedule changes and the two gray headers. The gray header that appears above the white summary grid displays summary information about each individual option (including the number of small changes - reschedules by less than 24 hours, large changes – changes by more than 24 hours, cancellations, RONs – changes in where an aircrew remains overnight, and number of constraint violations unable to be resolved by DWARP in this solution). The gray header to the left of the white summary grid contains summary information about each aircraft and mission that is affected by a delay or cancellation in one of the resolver options.

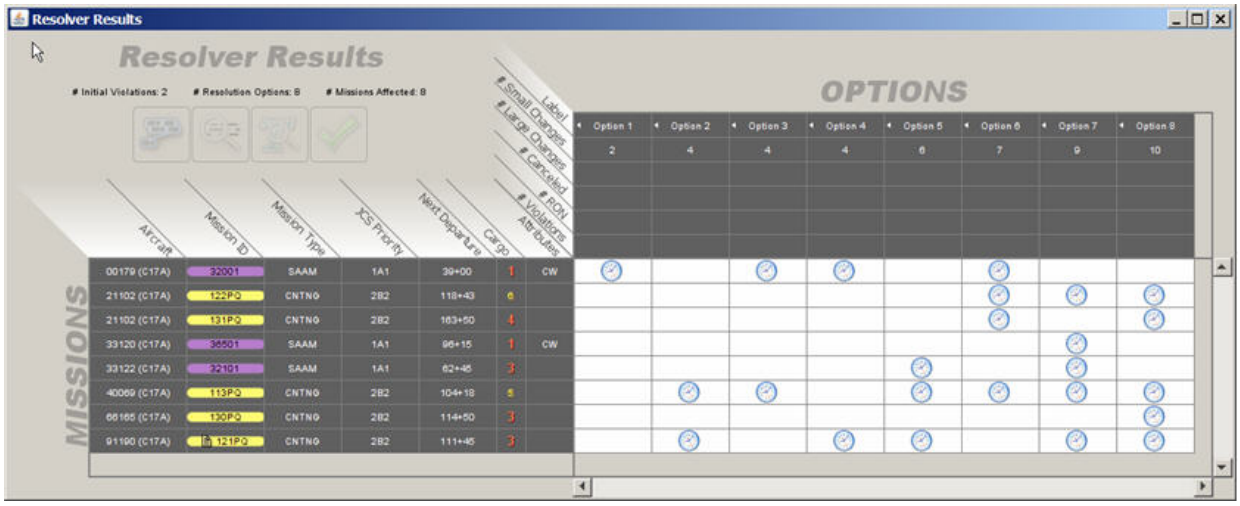

Figure 8

The *Mission ID* column stands out from other columns as it can have a background color and an annotation icon. The background color represents the *Mission Category* of the mission. The *Mission Category* is consistent with other views of the WIDE software. If an annotation icon (a yellow paper icon) exists next to the mission ID, then at least one user-entered annotation exists for that mission. Right-clicking on the icon will invoke a menu that allows users to access each annotation.

The *Next Departure* column represents the time offset (relative to the current time) of the next departing sortie for a particular mission. A mission with the value "8+30" means that its next departure is eight and one half hours from the current time. This information is useful as it may be a crucial factor in deciding whether a delay to that mission is feasible.

The *Cargo* column contains a number representing the cargo priority. Priorities are from 1-10, with 1 being the highest. The cargo priority numbering and coloring scheme is consistent

with WIDE's compact timeline view. Mousing-over the cargo requirement column will invoke a description of the cargo.

Finally, the *Attributes* column contains a list of abbreviated attributes that further qualify a particular aircraft or mission. Some of the available attributes for display in this column are:

DV = Distinguished visitor onboard
CW = Close-watch
MC = MICAP – carrying parts/personnel to fix a broken airplane
AE = Performing an air evacuation mission, and so unlikely to be rescheduled

The white summary grid contains cells that indicate how a particular resolver option affected a mission. If the cell is blank, then no change to a particular mission was made. If the cell contains a clock icon with a plus sign, there is at least one sortie of the mission represented by that row that was changed by 24 hours or more. If the cell contains a clock icon without a plus sign, there is at least one sortie that was changed by less than 24 hours. Mousing-over the clock icons will give details about which sorties were changed and by how much. If a cell contains a red X icon, then the mission represented by that row was canceled.

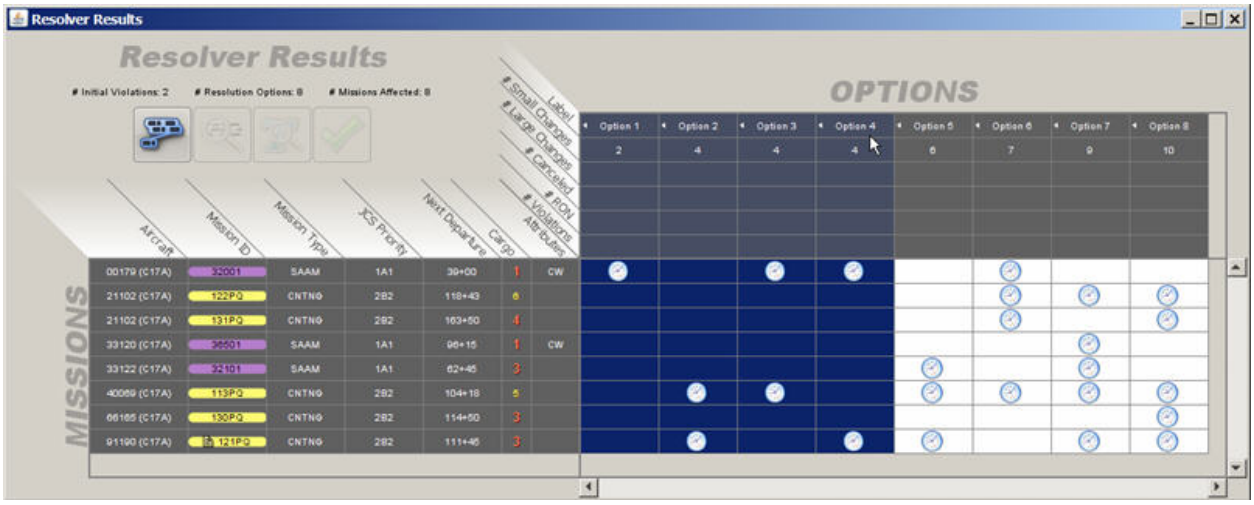

Figure 9

The *Resolver Results* display supports various forms of interaction. Users can hide and unhide options and drill down into different views of the options. The button panel on the top-left of the *Resolver Results* window provides controls for launching additional views and accepting an option

Each option's column has a left or right-pointing arrow next to its title, which will hide or unhide the option when clicked. A hidden column cannot be selected nor accepted as the new schedule.

The intent is that a DO will be able to scan the list of options, compare to missions that have features which make them unsuitable for delays, and fairly quickly limit his scope down to a few possible solutions. At this point, a more direct method of comparing a few options can be employed. When one or more options are selected in the table, clicking on the *Comparison View* button launches a new *Comparison View* window, bringing up a view that allows the user to

quickly get a sense of the difference between multiple options (and the original mission schedules).

The *Comparison View* is designed to visualize the differences between multiple options at a high level (see Figure 10). It contains elements that are common to WIDE views: sorties represented by a black bar, colored envelopes to visually group sorties by their mission and mission category, and violations. Each row of black bars represents a different option's schedule. The base state is always shown in this display for reference.

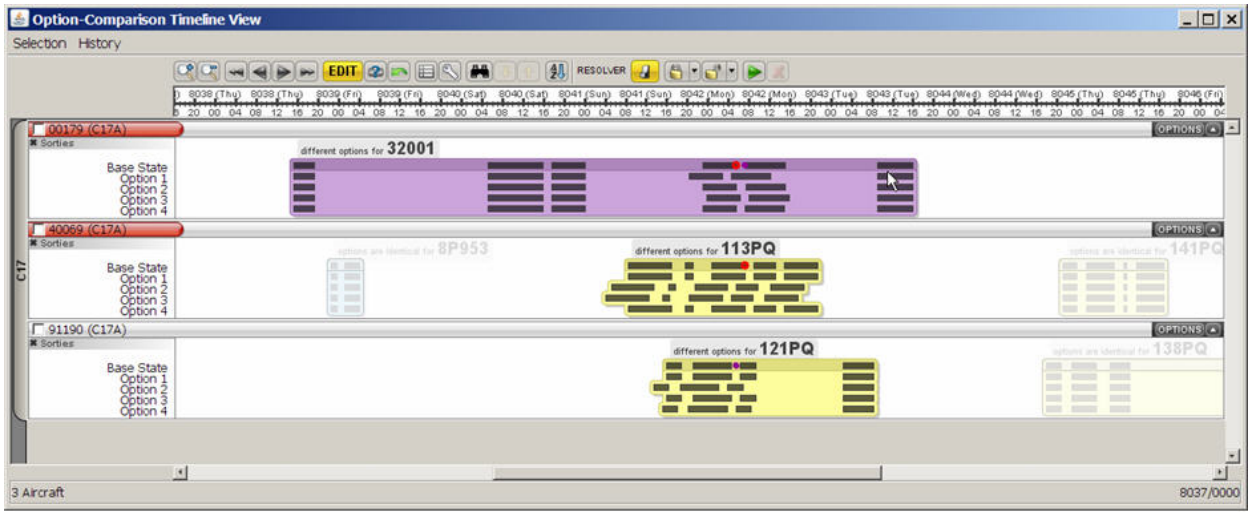

Figure 10

WIDE's usual timeline representation of a sortie is used in this display. Each mission is surrounded by a mission envelope. The mission envelope, like the Multi-Tail view and the compact timeline view, is colored based on its *Mission Category*. The base state is visually distinguished from resolver options with a slightly darker (or lighter depending on the *Mission Category* color) background color.

Since the *Comparison View* was designed to highlight the differences between multiple options' schedules, only missions that have changes are displayed with full opacity by default. Missions that have identical schedules across all options are displayed with transparency for de-emphasis. To view these options in full opacity, mouse-over the mission number that appears above the schedule. Mousing-over a particular mission's label will cause all other missions to become transparent and deemphasized.

Canceled missions will appear with red X's. Since a canceled mission has no particular attachment in time, the red X's are placed at the midpoint between two options.

Constraint violations are also shown on the *Comparison View*. Normally, the resolver generates solutions with no violations, so usually they only appear on the top line (Base State) of sorties.

To compare a different set of resolver options, the user simply opens the *Resolver Results display* again, changes column selections, and clicks on the "Open Comparison View" button.

Generalizing Our Design Solution

As we near the end of this research effort, it seems appropriate to take a more abstract look at the problem space we've been working in, and the particular design solution we've found. We would like to characterize the properties of our problem that drove us to our particular solution. This would help us identify the set of human-automation interaction problems for which our solution would be useful, and bring us closer to our ultimate goal of applying our technology to a wider range of more generic problems.

The basic problem is dynamic replanning of AMC air missions, although we can make some claim to also assisting in the problem of initially planning air missions. This is a very complex planning domain. Each air mission must be planned to make use of at least three classes of scarce resources–airplanes, crews, and airfield space. While each mission, as a rule, is planned to use only one physical tail, a single mission may use multiple crews and will likely use multiple airfields. In addition to the scarce resources and their effects on planning multiple simultaneous missions, each mission is subject to a number of individual constraints ranging from availability of cargo to be loaded, to diplomatic clearances required to cross certain national airspaces, to availability of fuel at intermediate airfields.

This effort was conceived to investigate applying optimization technology to assist human operators in a dynamic replanning problem. We have designed software user interfaces to help users task DWARP (the optimization software), and display optimization results in such way as to assist users in understanding, evaluating and trusting those results.

The particular task given to DWARP is, as a rule, something even an expert user would have a difficult time performing in a reasonable amount of time. This is not due to the conceptual complexity of the problem, but is due to the computational complexity – the sheer number of possible schedules and constraints.

Characteristics of our problem space

1. We are searching for solutions in a high-dimensional space.
2. We operate in an environment with missing critical information. The information about relative priorities of air missions and their cargo or passengers is generally not available in GDSS, although the user may have other sources for this information.
3. The user is the authority, knowing more about each individual problem element than the system. While users may not be able to optimally solve the problem, they can evaluate potential solutions, and can identify problems with them.
4. Schedules will not be executed precisely according to plan. Environmental factors and maintenance issues, among other factors, will perturb the plans. So just because a plan (a set of mission schedules) is legal doesn't mean it is a good plan–robustness in the face of further perturbations is a factor.
5. As a rule, replanning decisions are not time-critical, at least not in the sense that solutions need to be found in seconds or small numbers of minutes. Taking an hour or even two to find and implement a replanning solution is generally acceptable. This, of course, depends on the problem–there are cases when decisions need to be made quickly.
6. Thrashing (constantly changing solutions) is very bad. Replanning missions takes coordination between multiple parties–air crews, ground crews, flight managers, and DIPS planners are just some of the people that may have to act on changes to mission plans.

7. Our rescheduling problems generally allow for multiple solutions. There are multiple measures of goodness for a plan–completing all the missions as quickly as possible, having a robust schedule, getting air crews home at appropriate times, getting high-priority cargo delivered first, minimizing flight hours or fuel usage, and minimizing time spent doing time-consuming human re-coordinating as schedules change are just some of them. The user community has no consensus on a single scoring function that accurately reflects their preference for one solution over another.

Symbiotic Planning Design Guidelines

Abstracting from our design, we put forth the following design guidelines that we claim are suitable for human-automation interaction problems which match the complex replanning characteristics described earlier in the paper. In further work we expect to expand upon and refine these guidelines as well as compare and contrast our design solution and our guidelines with work described in (Hanson, 2004 and Horvitz, 1999).

1. The automated planner operates only as directed by the user–it does not operate autonomously. The rationale for this is as follows: The user is the authority in the problem space, adding value to a solution found by the automated tool by interpreting it in light of domain information he possesses that is unknown to the automated tool. Having the automated planner operate autonomously would essentially interrupt the human operator to evaluate the work of a support tool.
2. The automated planner is tasked by the user to replan some set of missions–not to replan the entire universe of missions. The set of missions to be considered for replanning is entirely under the control of the user. This matches the organization of duty officers within the TACC–each duty officer is responsible for a discrete set of missions. If he observes conflicts between his missions and missions outside his responsibility, he may be able, in some cases, to affect those other missions, but coordination would be needed to make that happen.
3. Even once the scope of the problem under consideration by the automated planner has been established, the user should be able to add more information to constrain the action of the automated tool to each individual problem element. In our prototype design, users may lock individual events (takeoffs or landings) of a mission, specifying that they must occur at specific times, or that they must occur before or after specific times. Empowering users to add this additional constraint information is critical–it allows users to enter real-world constraints that are not in the execution system (GDSS). For example, a pilot may call in to the TACC floor with the information that his plane is in maintenance and needs three more hours before it can take off. While this information may eventually make its way into GDSS, the duty officer needs to consider that information in his replanning decisions. In the prototype design, he can label the next takeoff for this tail as to occur no sooner than three hours from now, and have the solutions generated by the automated planner appropriately reflect the situation.
4. The automated planner generates multiple possible solutions to each problem to be evaluated by the user. There are two characteristics of the problem space that make this necessary. First, the fact that there is no globally agreed upon scoring function means that there may be multiple solutions worthy of the user's time. Secondly, there are often

unrepresented critical information items such as high priority cargo, or an impending airfield closure, that only the user may understand and appreciate. This additional information can be used by the human operator to further distinguish between possible solutions.

5. Users need to understand the solutions generated by the automated tool. To aid users' understanding and reduce the cognitive burden of interpreting large data sets, the tool needs to include a mechanism to graphically compare each proposed solution with the original plan as well as a mechanism to graphically compare proposed solutions.
6. Users need the ability to incrementally modify a particular solution generated by the automated support tool. This support needs to include a mechanism to accept a particular solution and then manually modify individual elements of it. It would be useful if users had the ability to accept a particular solution, modify it by altering the constraints, and using the result as a new input to the automated support tool.

Conclusion

This paper describes a problem domain in which the close integration of an automated support tool – in this case, an automated scheduler – with a user-centered decision support tool is necessary to help human operators find solutions to problems they would otherwise be unlikely to correct. The need for automated support is certainly not unique to this particular problem. There are many other current instances of C2 systems which would benefit from close integration with automated support tools. Evolving C2 system requirements will inevitably lead to many more systems in which effective integration with automated support tools will be not just helpful, but will be mission-critical.

Past attempts at integrating automated support tools into C2 systems have not always been effective. On several occasions, automation has been rejected by the very operators it was supposed to help most. The reason for the lack of acceptance varies, but is largely due to the lack of control and insight of the "automation" and its output (i.e. the format of the interaction to work with the automated planner and the complexity of the generated options or solutions).

Our methodology of work-centered design has led us to a very non-traditional design for this particular problem, in which the role of the system was to present multiple possible solutions to the operator and help the operator evaluate which (if any) solution to use. Specifically, extensions to the timeline visualization provide controls for the user to collaborate with the planner on a cognitive level. This dialogue is unique because it is not only visual in nature, but has concise abstract representations of the work domain from a cognitive perspective which afford rapid understanding, interaction and assessment, particularly with large data sets. Further, by allowing the operator to collaborate with the scheduler through constraint/bound controls and the option development process, the user becomes in integral part of the generation of the solution. This, in turn, engages the user in the option generation process and instills understanding and trust in the final solution.

We have, in this paper, identified some of the characteristics of our problem domain that led us to this design. We expect to, in future work, further explore how to most effectively integrate automated support tools into C2 decision support systems as well as generalize these principles to be applied to other relevant domains.

References

Eggleston, R. G. (2002). Cognitive systems engineering at 20-something: Where do we stand? In M. D. McNeese & M. Vidulich (Eds.), *Cognitive Systems Engineering in Military Aviation Environments: Avoiding Cogminutia Fragmentosa*. Wright-Patterson Air Force Base, OH: HSIAC Press, 15 - 78.

Eggleston, R. G., and Whitaker, R. D. (2002). Work Centered Support System design: Using organizing frames to reduce work complexity. *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting* (Baltimore MD: Sept. 30 - Oct. 4 2002), Santa Monica CA: Human Factors and Ergonomics Society, 265 - 269.

Eggleston, R. G. (2003). Work-centered design: A cognitive engineering approach to system design. *In Proceedings of the Human Factors and Ergonomics Society 47th Annual Meeting* Denver, CO: Human Factors and Ergonomics Society.

Hanson, Roth, Hopkins, Mancuso 2004, Developing Mixed-Initiative Interaction with Intelligent Systems: Lessons Learned from Supervising Multiple UAVs

Horvitz, 1999, Principles of Mixed-Initiative User Interfaces

Rasmussen, J. (1986). *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*. New York: North-Holland.

Roth, E. M., Stilson, M., Scott, R., Whitaker, R., Kazmierczak, T., Thomas-Meyers, G. and Wampler, J. (2006). Work-centered Design and Evaluation of a C2 Visualization Aid. *Proceedings of the Human Factors and Ergonomics Society 50th Annual Meeting.* (pp. 255-259). Santa Monica, CA: Human Factors and Ergonomics Society.

Vicente, K.J. (1999). *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer- Based Work*. New Jersey: Lawrence Erlbaum Associates.

Wampler, J., Whitaker, R., Roth, E., Scott, R., Stilson, M. and Thomas-Meyers, G. (2005). Cognitive Work Aids for C2 Planning: Actionable Information to Support Operational Decision Making. In *Proceedings of the 10th International Command and Control Research and Technology Symposium* (June, 2005).

Woods, D. D. & Roth, E. M. Cognitive Engineering: Human Problem Solving with Tools. *Human Factors*, 1988, *30* (4), 415-430.

Young, M.J., Eggleston, R.G., and Whitaker, R.D. (2000). Direct manipulation interface techniques for interaction with software agents. Paper presented at the NATO/RTO symposium on Usability of Information in Battle Management Operations, Oslo, Norway, April 2000.