# Agent Support for Mission Planning Under Policy Constraints

Chris Burnett, Daniele Masato, Mairi McCallum, Timothy J. Norman

Computing Science Department
University of Aberdeen, UK

Joseph Giampapa, Martin Kollingbaum, Katia Sycara

Robotics Institute
Carnegie Mellon University, USA

***Abstract*—Mission-critical scenarios, such as military or disaster response missions, often call for the formation of coalitions, made up of people from different countries or organizations and required to adhere to certain *policies*. These policies define the explicit *obligations, permissions* and *prohibitions* governing members of the coalition. While planning for joint action in these scenarios is already a complex problem for human planners, it is made more difficult or even impossible under such policy constraints, especially if policy conflicts exist between them. In this paper we propose that agents could be used to support human planners in coalitions, and present our work in the area of agent support for coalition mission planning under such policy constraints. We define a taxonomy of policies and outline the different types of support that agents can provide to human planners. We describe an experimental framework within which different types of agent support can be empirically evaluated within the context of a human planning problem.**

## I. Introduction

Coalitions are an organizational form whose members engage in collaborative activities. Coalitions are typically motivated by the fact that no single nation or organization has all the necessary capabilities or resources to undertake particular tasks alone. Typical examples are business or military alliances. In recent years, moreover, given the emphasis on business agility in the commercial world, and on rapid response for civilian and military crises, coalitions are formed rapidly and without much lead time or co-training. Although coalition members are engaged in collaboration to fulfill common goals, they differ from teams in many significant respects. First, in teamwork, the assumption is that team members do not have individual goals, but only shared common goals. The team members engage in collaborative planning and execution in pursuit of the common goals. By contrast, coalition members, besides sharing common goals are assumed to have individual goals whose fulfillment they are also pursuing. In other words, coalition members are self-interested, although it is assumed that by being part of the coalition, both the individual utility and also group utility are increased. Second, and partially as a result of the existence of individual goals, there is varying trust among the members of a coalition, whereas models of teamwork do not raise issues of trust. Third, typically coalition members have different policies that range from security policies to policies about how to conduct their missions (e.g. different rules of engagement in military coalitions).

These characteristics of coalitions present a variety of challenges. From the computational point of view, the coalition formation problem, namely forming coalitions that are stable so that no member has an incentive to leave the coalition, and optimize group utility is an NP-complete problem [1]. While the coalition formation problem remains a challenge, there is already substantial literature on approximation algorithms that gives good performance in practice [2,3]. Once a coalition has been formed, in order to fulfill common and individual goals, the members must engage in cooperative planning. The many challenges that coalitional operations face have been well articulated. One of the most crucial is how to construct joint plans in the presence of self interest, individual goals and diverse policies, especially in time stressed situations where there is not much time or previous co-training for the coalition members to recognize and resolve their differences.

One of the ways to address this challenge is to create automated agents that could assist coalitional partners in policy management so that effective coalitional planning can be performed. In this paper, we consider coalitions with a small number of members, each with its own organizational policies, who are cooperating to construct plans to fulfill shared and individual goals. We propose that automated agents could support human planners in coalitional planning and present an experimental framework within which different types of agent support can be empirically evaluated for coalitional mission planning under different policy constraints.

In section II we outline challenges that face coalition planners that operate under different policies. In section III we briefly describe some important types of policies; in section IV, we present different system architectures for agent support. In section V we present the experimental framework, an illustrative scenario and the software infrastructure we have developed to allow experimentations with different agent support strategies. In section VI we present related work and we conclude in section VII.

## II. Coalition Planning Under Policy Differences

Coalition policies are generally established in order to protect coalition members by setting out their expected behavior in a public and unambiguous manner. A coalition

policy may apply to the entire coalition, a subset, or even a single member; however, it is publicly visible to all coalition members (i.e. all coalition members are aware of the policy).

Individual policies, on the other hand, apply to individual members of the coalition, and are established for reasons particular to that individual's home organization. They may be kept private, or shared with other coalition members. However, the coalition authority or the cooperating coalition members may not be aware of the existence of private individual policies until a conflict or violation arises.

Planning within coalitions, guided by policies, presents a particular challenge for human planners. Especially in time stressed situations, this can be potentially mitigated through agent assistance. Challenges in terms of policy-driven planning include:

1. How do the coalition members become aware of policy differences during planning? If not discovered, such differences would be revealed during plan execution with potentially disastrous consequences.
2. How do coalition planners in time stressed situations remember all the different policies of their organization that may pertain to planning a new joint mission?
3. How can coalition planners detect conflicts between policies?
4. How can coalition planners resolve such conflicts in order to achieve an effective joint plan?
5. Typically, the different planners will "inherit" the different policies of their organizations that are generic and not mission specific. This may result in having policies that are unnecessarily rigid with the possible consequence of impeding effective collaborative planning.

Given these challenges, it is easy to imagine that coalition planning could often break down or produce infeasible or very sub-optimal plans. Our goal is to investigate what agent assistance strategies could be effective in allowing the human planners to produce as good a plan as possible considering the policy constraints.

## III. Taxonomy of Policies

As a first step in this investigation, and in order to ground the work in reality, we report on a taxonomy of policies we have developed. In particular, we have identified the following sub-classes of policies: resource policies, information sharing policies, procedural policies (policies defining the procedures to be used under certain conditions), conceptual policies (policies with respect to the interpretation of concepts), action policies (performance of actions and achievement of goals), and background policies.

### *A. Resource policies*

Coalitions often call for the pooling of resources. We assume there is a set of resources available to the coalition and for each resource there is an owner, which is a coalition partner. Consequently, there may be coalition policies that represent coalition-wide resource sharing agreements, and individual policies that represent member-specific resource sharing constraints. For example:

*I am prohibited from allowing coalition partner X to use resource R.*

*I am permitted to use resources of type R from a specific coalition partner X between times T and T'.*

The first example express policies of resource use where there is no prior agreement with the coalition partner concerned. In the second example, a policy is established because of the existence of an agreement between coalition partners.

### *B. Information sharing*

Information sharing policies describe what information can, must or must not be shared. It is likely that coalition members may want to protect certain pieces of information in order to safeguard their own interests, and such cases would be codified by individual policies. Information sharing policies could refer to specific aspects of information, such as:

- Disclosure of objects: e.g. I am permitted to disclose the existence of X, where X is a UAV
- Disclosure of parameters (details of objects): e.g. I am prohibited from disclosing the parameters P of object X where P is {range, sensor configuration, weapons}
- Disclosure of information sources (provenance, pedigree): e.g. I am prohibited to disclose the source of information regarding an imminent attack (because the information was obtained by a highly classified UAV)
- Disclosure of procedures: e.g. I am prohibited from disclosing the normal operating procedures of UAV search patterns
- Disclosure of policies: e.g. I am obliged to disclose information sharing policies with my coalition partner Y

### *C. Procedures*

These are policies that determine how to proceed with certain actions in specific situations (plans of how to accomplish things). For example:

*If my goal is to search area X, then I am obliged to employ search pattern A.*

*If ground troops are to be used for an operation, I am obliged to put in place air cover.*

Some consideration should be given to whether or not all procedural policies can and should be expressed as conditional norms. Some of these "policies" may simply be preferences over possible plans, and hence weaker than a conditional obligation.

### *D. Interpretation of concepts*

For example, two coalition partners may have a policy that obliges them to minimize environmental damage, but the interpretation of what constitutes "minimal environmental damage" may differ between the partners. Hence there is a non-obvious conflict between their respective policies.

### E. Actions

Coalition partners may have policies that oblige, permit or prohibit specific actions, and hence influence the planning of specific actions.

### F. Goals

Policies may also exist that make specific states of affairs obligatory, permitted or prohibited. These will certainly influence the selection of goals/sub-goals, but may also influence the selection of actions due to possible side-effects of performing an action. For example, an action may be obliged, but a known side-effect of that action is prohibited.

### G. Defaults

Default policies express the upfront "normative" position of a coalition partner (or the coalition itself, if there exists an agreement) and determine what "default policy" has to be assumed in case no explicit permission or prohibition is defined. Organizations usually have to elect one of the two possibilities:

- if something is not explicitly allowed, it is prohibited,
- if something is not explicitly prohibited, it is allowed.

For example:

*By default, I am prohibited from using resources owned by other partners.*

*By default, I am permitted to disclose the existence of any object.*

## IV. Agent Support

Agents can support policy-driven collaborative planning in the following ways:

- helping planners to identify and advising on the existence of a policy conflict
- assisting in conflict resolution by proposing solutions that are free of conflict
- detecting and advising of policy violations
- detecting and advising when a policy becomes active or inactive
- actively filtering prohibited information for policy enforcement (*censoring*)

We envision that coalition planners communicate either face to face or through some electronic medium during planning. We identify three different architectural configurations for agent aiding:

- A single filter/reconciler agent supporting the entire coalition, holding the sets of private policies for all coalition partners. With such a 'global' view of the policy space, conflicts can be detected and reconciliation done off line in a centralized manner. Resolutions are pre-computed before the planning begins and the resolutions are presented to the humans at appropriate times during planning, i.e. when conflicting policies become activated. Requirements for this kind of agent are that the agent can be trusted and is robust against manipulation. Depending on the level of trust existing between coalition partners, this may or may not be a desirable solution.
- Each coalition partner has a specific ``proxy'' agent that acts as a policy guide, gives advise on active policies and detects possible violations. The agent proxies themselves may engage in a process of conflict resolution if such conflicts exist and are detected between the partner-specific sets of policies. We may regard this as a decentralized form of conflict detection and reconciliation.
- A trusted agent continuously monitors the communication of the human planners and supports planning ``online''. Such a monitoring and support scheme is of particular interest in ad hoc and time-stressed coalition planning environments.

Our long-term goal is to compare the different architectural approaches and identify tradeoffs among the architectures (e.g. pre-computing the reconciliations is more efficient in terms of providing on-line assistance but not realistic -- we do not expect two organizations to merge their policy databases into a common database).

The condition where the two policy bases are merged can also be run with and without pre-computation of reconciliations (i.e. the agent is looking into its merged database for whether a conflict occurs and using heuristics for reconciliation).

Besides examining different agent based architectures for support of coalition mission planning, we are interested in experimentally comparing different agent aiding strategies. We have identified a variety of agent aiding strategies. First, an agent could act as *censor*, by deleting parts of messages that contain policy violations. Another aiding strategy is a policy critic that recognizes a policy violation during a message exchange, interrupts the message before it reaches the recipient and informs the sender that she has violated a policy, thus giving the sender a chance to mitigate the policy violation.

## V. Human-Agent Experiments

We hypothesize that humans planners, engaged in coalitional planning under the kinds of constraints we have described, will exhibit better performance with intelligent agent assistance than without. We have developed an experimental framework with which to test this hypothesis and evaluate the effectiveness of different agent aiding conditions in supporting human coalitional planning where the coalition partners have their own goals and operate under different policies. To this end, we consider three experimental conditions: humans aided by 'censoring' agents, humans aided by 'critiquing' agents, and humans planning without any agent assistance (the control condition). There are a number of ways in which we can measure performance. Firstly, we may consider the quality of the plans that are produced. While the 'goodness' of a plan is generally a domain specific dimension, we may assume that 'better' plans are those that violate fewer policies, or that maximize some task specific dimension, such as the cost of the plan in terms of resource usage, or the feasibility of the plan in terms of goals achieved. We may also consider dimensions related to the planning process itself, such as time taken to produce a plan, or the number of communication utterances. For example, if aided teams produce plans of similar quality to unaided teams, but do so in less time or with less communication, then we may consider this an improvement of performance. Another

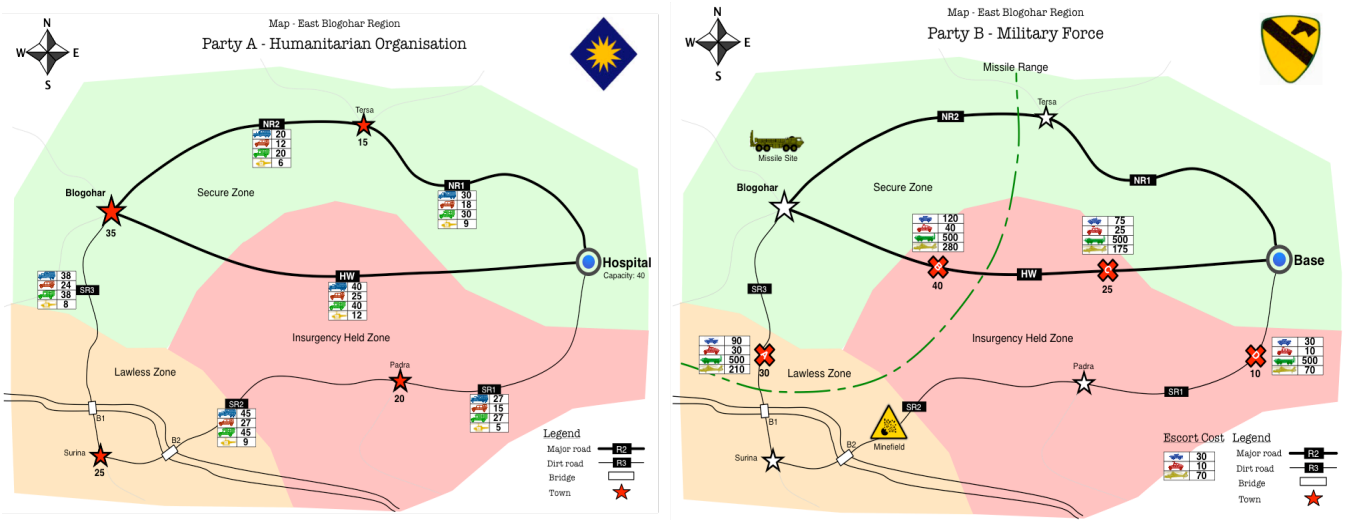


Fig. 1. Player maps

dimension we are interested in investigating is to what extend humans rely on the agent to "catch" policy violations, and thus they are not careful in their adherence to policies.

*A. Experimental Scenario*

We present an illustrative scenario that will serve as a vehicle for testing our experimental hypotheses. Both the scenario and planning problem described here are illustrative of what can be supported in this framework, and are intended to be part of and compatible with the Holistan scenario [4]. The scenario involves two human participants, or 'players'; one representing a military force, the other representing a humanitarian aid agency, both operating in the same region over a period of two days. The region is in the grip of a violent insurgency, and as a result there are injured civilians in the local towns, and insurgent strongholds spread throughout the area.

The goal of the humanitarian player (labeled 'party A') is to plan the medical evacuation of injured civilians from the towns, while the goal of the military player (labeled 'party B') is to plan the deployment of military force to tackle the insurgent strongholds. In order to achieve these goals, the players must collaborate to create plans which describe the deployment of their resources and commitments to each other.

Each player is provided with a map which provides private information relevant to the player's individual goals, in addition to some common information about the region, such as locations of towns and routes between them. For example, party A's map is annotated with the numbers of injured civilians in each town, while party B's map shows the location of the insurgent strongholds (see Fig. 1).

Both players are provided with a detailed briefing which outlines their private *goals*, *resources*, *intelligence*, *capabilities*, and *policies*:

- *Destinations* denote areas where resources must be deployed and are connected by routes: for party A the destinations are towns and for party B they are insurgent strongholds. Destinations also have a numerical requirement, namely the number of injured civilians in towns and military resistance of insurgent strongholds.
- *Resources* are vehicles which when deployed to a destination, satisfies part of its requirement. Resources have associated numerical values, for example, the carrying capacity of vehicles party A's vehicles, or the military 'strength' of party B's vehicles. All vehicles begin at the point on the map marked 'Base' or 'Hospital', and must travel along the routes to reach their destinations. Resources also have associated costs of deployment that vary depending on the routes along which they are deployed. A complete plan specifies a deployment of resources along routes to destinations that together satisfy the requirements of all the destinations on the map (Fig. 1).
- *Intelligence* gives players some additional information about the situation that they can share with each other (irrespective of policies). Each piece of intelligence has an associated source. For example, party B has intelligence about the locations of explosive devices.
- *Capabilities* specify what players can do with respect to their resources and the intelligence they possess. For example, apart from deploying military vehicles to insurgent strongholds, party B can also deploy his vehicles as military escort for party A's operations.
- *Policies* describe the obligations, prohibitions and permissions governing a player regarding the actions she can perform and the information she can share.

Each player is governed by a set of policies inspired by international guidelines for humanitarian/military co-operation [5] (10 for party A, and 16 for party B). Players are allowed 20 minutes in which to familiarize themselves with the policies, and 10 minutes to work through a practice planning problem.

Our test policies are designed in such a way that communication, collaboration and individual sacrifice of utility are necessary if the participants are to produce plans that honor those policies. For example, some policies specify preconditions under which certain deployments may be made. The choices of one player may affect the policy preconditions of the other player. In this way, the players may be brought into conflict; the decisions of one player may affect the allowed capabilities of the other player. If the policies are to be obeyed, then not only must the players recognize that such a conflict exists, they must also evaluate and negotiate alternative plans that do not violate policies.

**Party A:**
*You are only permitted to deploy ground vehicles along a route R on a day D **IF** you first acquire clearance from Party B to use route R on day D **AND** you have intelligence indicating route R is free from explosive device threats.*

**Party B:**
*You are only permitted to conduct military operations along a route R on a day D **IF** Party A has not committed to deploy vehicles along route R on day D **AND** you have not granted Party A permission to use*

Fig. 2. Example policies

The two policies in Fig. 2 demonstrate a kind of conflict between specific parameterizations (in this example, the route and day of deployments) of capabilities, where one player's actions can cause the other's to become prohibited. Our scenario contains one particular road (the 'highway') which is an attractive choice for both players. However, a state in which both players are using the highway at the same time would result in policy violations for one (or possibly both) of the players.

To avoid this, players must negotiate and compromise. However, there may be information sharing policies that can complicate negotiation by prohibiting players from revealing certain information. For example, party B has such a policy forbidding her from revealing intelligence marked as classified. Therefore, party B would be forbidden from using such intelligence when trying to resolve conflicts.

*B. Experimental Conditions and Software*

The framework consists of a coalition planning problem, set within a fictional scenario, and a collaborative interface that allows the insertion of agents to observe and interact with the human participants during the planning process. In order to design an effective framework for our experiments, a number of different issues needed to be considered. Initially, we required an interface that offered the participants an easy way to communicate with one another, build and coordinate their plans, and devise their own courses of action to achieve their objectives. Furthermore, it was a requirement that the interface allow software agents to intercept participants' dialogues and interactions with respect to their policies, and provide feedback to them. Finally, the framework was required to be robust so that the experiments could be run easily in different locations, i.e. that their set up be as simple as possible. To address these requirements, we developed a web-based interface, shown in Fig. 3.

The web interface is supported by a database that stores one or more scenario descriptions in a structured way. From the scenario description, the graphical interface can be dynamically generated according to the participant's role in a given scenario. Each *scenario* is represented in the database in an equivalent way to its textual description. A scenario includes two or more players, each having their own objectives and owning various *assets* with different features that can be used to build a plan and communicate with others. Specifically, assets can be divided in two categories, *intelligences* and *capabilities*: an intelligence is composed by its *content* and *source* (e.g. "there is the high probability of mines along all roads in the Rina (yellow) region [source: Local authorities]"), whereas a capability consists of an *action*, performed by employing a given *resource* at a given *location* at a particular *time* (e.g. "launch missile using Missile Platform 1 along route HW on day 1"). In addition, the database supports the association of one or more *plans* to each player. Plans are represented as an ordered sequence of steps that should achieve the player's objective and at the same time comply with the policies and other participants' plans.

Policies, as detailed in section II, may constrain the sharing of particular pieces of information or the usage of particular assets according to a number of conditions. However, they are not stored in the database, but instead are managed and maintained directly by the agents. In this way, the same scenario can be run in different experimental sessions with alternative agent implementations without affecting the stored scenario description.

The interface is divided into two columns: the left column is modeled as a chat box where the top frame shows the message history while the bottom frame allows players to send new messages by selecting them from a list of predefined phrases, whose content is based on the available assets. Specifically, a phrase is represented as a group of components (e.g. for messages involving a capability the components are *action*, *resource, location* and *time*) that can be either disclosed or not. This structured representation permits the agents to reason about the single parts of a message, for example to check for information disclosure violations, thus avoiding the requirement for natural language processing. Predefined phrases are of three types:

1. General requests to other parties: they include requests for information about intelligences, requests for information on capabilities, requests for permissions, etc;
2. Informative messages, which can be classified into:
   a) information about one's ability to perform an action (a capability): "I can deploy Helicopter 1 along route SR1 on day 2";
   b) information about one's intelligences: "All roads in the Haram (green) region have been cleared of mines".
   c) direct answers that do not imply an action performance, like permission granting, affirmative/negative statements: "I grant you clearance for ground vehicles to travel along road HW on day 1".
3. Commitment messages: these specify that a player is going to (or has already) put a specific step in his plan: "I commit to deploy Field Hospital-2 along road SR2 on day 1"

A free-text input box is also provided for non-standard communications. However, agents would not be able to process this kind of input easily.

The right column is used as a whiteboard when the player's plan can be built as a sequence of steps (i.e. available capabilities), choosing them from the list on the bottom window. Steps already in the top list can be reordered or deleted at will. The planning activity can be divided in two phases: in the first phase, participants are allowed to devise and commit an initial plan independently, and in the second they are asked to revise it, communicating with the other players, so that the final plan is consistent

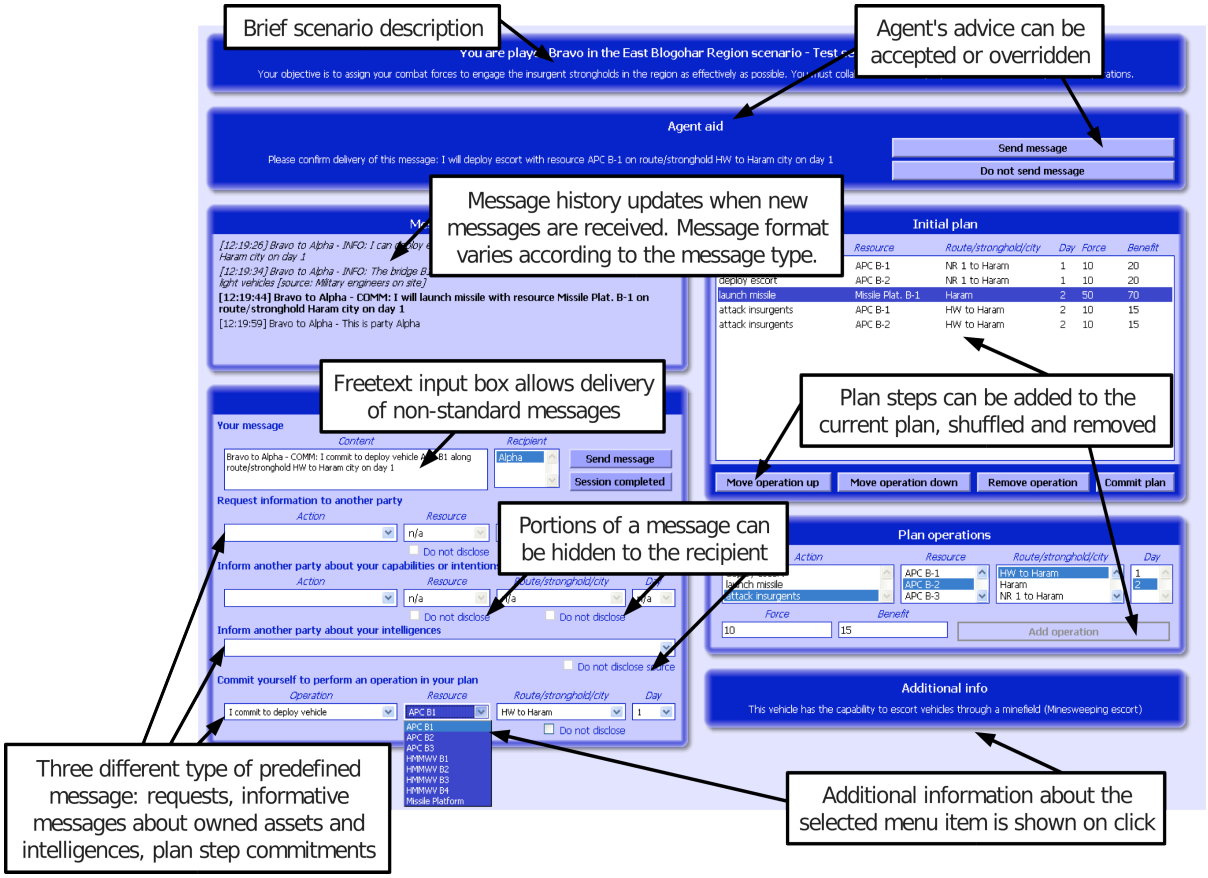


Fig. 3. Experimental Interface

both with the policies and the objectives (although this may not happen if players make mistakes).

Each interaction of the players with the interface (i.e. every exchanged message and modification to the plans) is sent to the agents and logged into the database. This approach offers two advantages: firstly, agents can be notified about participants' interactions with the interface on an event-by-event basis, hence either react immediately or perform more complex reasoning based on multiple interface interactions. We are using this mechanism to monitor policy violations and prevent them immediately, or to check for inconsistencies in the plan steps (e.g. unsatisfied preconditions). Secondly, by accessing the stored logs, it is possible both to perform a post-experiment analysis of players and agents' behavior and to re-run experiment sessions off-line. At present, a separate agent with complete access to the database information is instantiated for each player. Agents can provide feedback to players in the form of text messages by means of the top window (entitled "*agent aid*" in Fig. 3), which also contains two buttons in order to reply to the agent's suggestions. Participants can either accept the agent's behavior or override it. However, the content of the message and the semantics of the buttons are completely customizable according to the desired agent behavior so different types of human-agent interactions can be envisaged. By exploiting this mechanism, it is possible to implement agents that initiate and maintain a dialogue with the players, keeping in account the different choices they perform through the interface and the answers to the agents' suggestions.

*C. Agent Support*

Agent support was designed with three criteria in mind:

1. The correct and appropriate assessment of compliance with or violation of policies
2. Presentation of the agent to the user: degree of visibility, how does the agent make its presence known to the user, striking the right balance between pro-activity and reactivity to user actions, minimizing the possibility of irritating the user
3. User dependence on the agent: we want to design the agent so that its intervention may help the user learn and navigate its policy / goal space more effectively. Rather than blindly rely on the assistance of the agent.

In fulfilling the first design criterion, the agent itself performs the reasoning about a given set of policies. Due to the conditional nature of our form of policies, the Jess expert system shell [6] is used for encoding these policies. Fig. 2 gives an example for a policy of party A. In order to show how such a policy can be expressed in Jess, we first slightly reformulate this policy (see Fig. 4).

This can be directly translated into a Jess rule (see Fig. 5). Assuming that we hold information about messages (e.g. "BC-A1" expresses that party B granted clearance) and intelligence about safety (e.g. an attribute "intel-type" indicates whether intelligence indicates danger).

Based on the fact that party B has sent a message that grants the use of a specific route on a given day and that party A holds intelligence that there is "no danger" on this route on the given day leads to a response of the agent, expressing a permission to use the given route. In the context of the web interface, the reasoning of an agent about these policies takes place in a specific execution cycle, where (a) in the event of sending or receiving messages or

```
IF: you have acquired clearance from party B
to use route R on day D
AND: you have intelligence that there is no
danger on route R on day D
THEN: you are permitted to deploy ground
vehicles on route R on day D
```

**Fig. 4. Example policy**

```
(defrule A-P1 "Party A - Policy 1"
(action-message
    (sender PartyB)(message-id BC-
        A1)(route $?route)(day ?day))
    (intelligence (intel-type
        nodanger)(route $?route)(day? day))
     =>
     (assert (response (response-id A-
        P1)(route ?route)(day ?day)))
```

**Fig. 5. Equivalent Jess rule**

declaring plan steps, the agent has to translate this information into Jess-specific constructs and assert them as facts into the Jess RETE engine; (b) given this new information, the agent will reason about it and produce corresponding responses; and (c) the agent evaluates the responses, which leads to specific advice conveyed to a user, a possible filtering of messages that would otherwise violate policies or suggestions how to constrain the communication between the planning partners.

With respect to design criterion 2, we designed the agent as an unobtrusive monitor of human communication and planning activities. The agent's reaction is based on criteria resulting from multiple information sources and not just a simple and uninformed reaction to user actions. We provide each player with a personal agent that monitors their messages they want to send, the messages they receive and the plan operators they propose. For example, our example policy (see Fig. 5) would be impossible to verify if the agent did not have access to party A's incoming and outgoing messages as well as tentative plans. In this way, the agent follows the human lead, it is aware of information that is arriving to the human and is monitoring the human's intentions by observing the formation of their tentative plans. Autonomously and proactively, the agent either critiques a message or provides suggestions how to change or reformulate outgoing messages so that they are in compliance with policies and goals. The agent does not force the user to a particular action, it merely provides advice and suggestions in the "agent aid" window, which the user can accept or reject.

With respect to criterion 3, we design the agent to assist users in becoming proficient as quickly as possible without actually solving the task for them. We achieve this by designing the critiquing agent as an advisor that proposes options that the user may accept or reject rather completing the task for them. It is important that the problem-solving capabilities of users are strengthened and augmented rather than diminished. The censor agent is also designed so as to avoid over-reliance of the user on the agent. The censor agent simply masks the violated policy part from the message, thus delivering a message that may not be intelligible to the other party.

## VI. Related Work

Policies have been used in disparate fields, ranging from security models of programming languages to the management of resources in distributed IT systems [7]. Explicit policies, as opposed to implicit ones embedded in software, define computational behaviors but allow designers and engineers to easily change them and verify desirable properties in them [8]. Policies are sometimes equated with permissions (e.g., the security model of Java and the UNIX file access system), but this view is rather limited: what is not explicitly permitted is prohibited; obligations cannot be represented. In this paper, we therefore regard policies as describing the three normative concepts "obligation", "permission" and "prohibition". Previous work has investigated methods for detecting and resolving conflicts within sets of policies [9]. This is of particular importance in coalition environments where planners are required to adopt and adhere to policies from different authorities, easily leading to situations where certain actions are simultaneously forbidden and obliged/permitted [10].

## VII. Conclusions

In this paper we have discussed the difficulties facing planners working in coalition environments with respect to policy differences between them, and described agent-based strategies for assisting in resolving these issues. We have outlined an experimental framework within which the effects of these strategies may be empirically evaluated, in the context of a military/humanitarian scenario. We have also discussed some of the design and implementation issues we face in realizing our proposed agents.

## Acknowledgment

Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

## References

[1] H. Moulin. Cooperative Microeconomics: A Game-Theoretic Introduction. *Princeton University Press*, 1995

[2] Yamamoto, J, and Sycara, K. "A Stable and Efficient Buyer Coalition Scheme for e-Marketplaces" Proceedings of the Fifth International Conference on Autonomous Agents, May 28-June 1, Montreal, CA. 2001.

[3] Li, C. and Sycara, K. "Algorithms for Combinatorial Coalition Formation and Payoff Division in e-Marketplace*", Proceedings of the First International*

*Conference on Autonomous Agents and Multiagent Systems (AAMAS 02),* Bologna, Italy, July 15-19, 2002.

[4] D. Roberts, G. Lock, and D. Verma. Holistan: A Futuristic Scenario for International Coalition Operations. *Integration of Knowledge Intensive Multi-Agent Systems, 2007. KIMAS 2007. International Conference on*, pages 423–427, 2007.

[5] V. Wheller and A. Harmer. Resetting the rules of engagement. Trends and issues in military-humanitarian relations. *HPG Research Report*, 2006.

[6] E. Friedman-Hill et al. Jess, the java expert system shell. *Distributed Computing Systems, Sandia National Laboratories, Livermore, CA*, 2000.

[7] J. Moffett and M. Sloman. Policy Conflict Analysis in Distributed Systems Management. *Journal of Organizational Computing*. 1993.

[8] W. W. Vasconcelos. Norm Verification and Analysis of Electronic Institutions. *Volume 3476 of LNAI*. Springer-Verlag, 2004.

[9] W. Vasconcelos, M. Kollingbaum, and T. Norman. Resolving conflict and inconsistency in norm-regulated virtual organizations. *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, 2007.

[10] A. Elagh. On the Formal Analysis of Normative Conflicts. *Information & Communications Technology Law*, 9(3):207–217, 2000.