# Managing Interaction Between Users and Agents in a Multi-agent Storytelling Environment

Mark Riedl
Department of Computer Science
North Carolina State University
Raleigh, NC, 27695
+1.919.513.3038
moriedl@eos.ncsu.edu

C.J. Saretto
Microsoft Corporation
1 Microsoft Way
Redmond, WA 98052
+1.425.707.8048
cjsar@microsoft.com

R. Michael Young
Department of Computer Science
North Carolina State University
Raleigh, NC, 27695
+1.919.513.3038
young@csc.ncsu.edu

## ABSTRACT

This paper describes an approach for managing the interaction of human users with computer-controlled agents in an interactive narrative-oriented virtual environment. In these kinds of systems, the freedom of the user to perform whatever action she desires must be balanced with the preservation of the storyline used to control the system's characters. We describe a technique, narrative mediation, that exploits a plan-based model of narrative structure to manage and respond to users' actions inside a virtual world. We define two general classes of response to situations where users execute actions that interfere with story structure: accommodation and intervention. Finally, we specify an architecture that uses these definitions to monitor and automatically characterize user actions, and to compute and implement responses to unanticipated activity. The approach effectively integrates user action and system response into the unfolding narrative, providing for the balance between a user's sense of control within the story world and the user's sense of coherence of the overall narrative.

## Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem solving, control methods, and search – *plan execution, formation and generation.*

## General Terms

Algorithms, Design, Theory.

## Keywords

Interactive narrative, planning, computer games, embodied agents

## 1. INTRODUCTION

Increasingly, interactive, narrative-oriented systems (or INSs) ranging from conventional computer games to intelligent tutoring systems are being designed in which human users interact with one or more embodied, animated agents inside a virtual world. In these systems, users and agents interact to carry out actions as characters in furtherance of a storyline. In these systems, the overall experience of the user is dependent to a great extent upon the coordination of the actions performed by the user and the agents operating within the story world according to some system-determined narrative structure. To ensure coherence, specific attention is typically paid to the coordination of the agents operating within the world, for instance, through the use of a global planning system [4, 15] or negotiation techniques between autonomous characters [11] to create action sequences that form engaging storylines.


*AAMAS '03*, July 14-18, 2003, Melbourne, Australia.


When the user is also in control of a character within the same environment, management of the coherence of the unfolding storyline takes on an additional complexity. The interaction between the human agent and the other autonomous agents must be managed carefully to ensure that the human agent does not disrupt the activities of the other agents to the point of failure. However, while the user is encouraged to play the role of a story-world character, she typically only has partial knowledge of the narrative. In fact, this partial knowledge is often central to the user's experience (e.g., in the creation of suspense in entertainment applications); consequently, approaches for coordinating actions between a user and a collection of autonomous agents via explicit communication (e.g., [10]) are not always appropriate in story-based environments. When a lack of knowledge of the unfolding narrative is combined with the ability to interact with the story world in a relatively uncontrolled manner, the potential arises for the user, through her character, to perform actions that are not only contrary to the plan shared by the other agents but even harmful to the coherence of the narrative.

In this paper, we describe the process of *narrative mediation*, a technique for detecting and responding to unanticipated user activity within Mimesis [21, 22], an architecture for building intelligent interactive narrative worlds.

## 2. RELATED WORK

One approach to coordinating agent activity within environments where change is common and unanticipated is to adopt reactive techniques. In these approaches, the selection and execution of individual tasks is closely integrated with the state of the environment, and changes in the environment are immediately reflected in the behavior of the agents within the system.

Firby's [6] Reactive Action Packages (RAP) is one such system designed to handle agent execution in contexts in which agent actions might fail. Given a set of tasks for an agent to perform, the RAP system defines a set of alternative procedural methods for accomplishing each task. While RAP has been demonstrated to be highly effective in handling agent execution in uncertain environments, the task model is limited in its ability to model the structure of unfolding narratives. For example, in RAP, each task is unordered with respect to all other tasks. Furthermore, an agent performs the most relevant task at any given time in an opportunistic fashion. This type of opportunistic execution of agents' actions may directly violate an INS's narrative structure.

Blumberg and Galyean [1] introduce a technique for controlling believable agents that interact with the user in a shared virtual environment. Their approach uses sets of behaviors arranged in a hierarchical fashion, with action selection performed via a competition mechanism between behaviors. In their approach, behaviors are ranked based on mathematical descriptions of sensory input. Because each agent in this approach pursues its own goals without regard for cooperation with other agents or with the user, the type of coordination needed for narrative coherence is not directly controllable. However, Blumberg and Galyean provide the possibility of external control mechanisms that are capable of guiding agent behavior by inserting behavioral modifications directly into the agents. Galyean has further demonstrated that an interactive narrative system can provide external control [7] for the architecture that they define. This narrative control, however, operates similarly to reactive agent behavior selection. While their approach provides for some ordering of tasks within a narrative, control of the methods for performing these tasks lies primarily with individual agents.

Weyhrauch [23] defines an architecture, named MOE, used for structuring actions within an interactive narrative. MOE models both user and system actions, using a form of adversarial search to select actions for system-controlled agents based on a complex set of heuristics. In systems built using MOE, users are unconstrained in the actions that they chose to execute; MOE manipulates the story line (either by initiating the actions of system-controlled characters or by directly altering the state of the story world) in order to subtly guide the user along a sequence of actions that will provide the most entertaining experience. At each point in a story, MOE searches through all potential sequences of subsequent actions that could be taken by the user, by the characters within the story and by the system itself. A ranking function is used to select the best sequence to execute based on a number of factors including: the user's sense that a storyline is logically connected, the user's perceived freedom of action over the sequence's execution, the excitement a user will feel as the storyline progresses and the user's sense that her interaction within the story is being manipulated by the system.

Steve [15] is a pedagogical animated agent that interacts with a human user within a virtual training environment. Steve demonstrates skills to the user, monitors the user's activity within the environment and responds appropriately when she interferes with Steve's activities or attempts the tasks herself. The Steve architecture uses a planning algorithm and a hierarchical task model to determine Steve's behavior and the SOAR cognitive model [12] to select actions to perform. The SOAR cognitive model handles changes to the environment made by unanticipated user activity by monitoring the environment and opportunistically selecting behaviors that can establish unachieved goals and deselecting behaviors when the system observes that their goals have been achieved. Recently, Steve has been extended for use within an interactive narrative system, the Mission Rehearsal Exercise (MRE) [18], and responsiveness to unexpected user actions within the MRE system is managed by the same approach used in Steve.

# 3. MEDIATION IN INTERACTIVE NARRATIVES

## 3.1 Balancing Control and Coherence

A central issue in the development of effective and engaging interactive narrative environments is the balance between coherence and control. The understandability of any narrative is determined, in part, by it's *coherence*, that is, by the user's ability to comprehend the relationships between the events in the story, both within the story world (e.g., the causal or temporal relations between actions) and in the story's telling (e.g., the selection of camera sequences used to convey the action to the user). Dramatists often refer to narrative as having a premise or point [5]; stories are told for a reason and much of our comprehension of a story involves the construction of cognitive models that predict or explain these relationships [8, 9]. Systems that construct actions for telling a story should respect the story's coherence by clearly linking each action in the story world to its overall structure.

The degree of engagement by a user within an interactive narrative lies, to a great extent, with the user's perceived degree of *control* over her character as she operates within the environment. The greater the user's sense of control over her character, the greater will be her sense of presence [13], that is, the sense that she is a part of the story world and free to pursue her own goals and desires.

Unfortunately, control and coherence are often in direct conflict in an interactive narrative system. To present a coherent narrative, the actions within an INS's story are carefully structured (either at design time by human designers or at run time by narrative generation systems) so that actions at one point in the story lead clearly to state changes necessitated by actions occurring at subsequent points in the story. When users exercise a high degree of control within the environment, it is likely that their actions will change the state of the world in ways that may interfere with the causal dependencies between actions as intended within a storyline.

Conventional forms of narrative (e.g. film and novel) resolve the issue of coherence versus control by completely eliminating control; the audience is a passive observer. Computer game developers, in contrast to film makers, introduce interactivity in their systems, but carefully limit the control exercised by the user by designing the environment so that the user's choices for action at any point reduce to a small set of options moving the user through a pre-defined branching structure [2]. In the remainder of this paper, we discuss a technique called narrative mediation which allows a degree of control and coherence that lies between that of computer games and conventional narrative media. This technique is implemented within an INS named Mimesis, described briefly below.

## 3.2 The Mimesis Architecture

The Mimesis system defines an architecture for building and coordinating interactive adaptive narratives. The system utilizes a

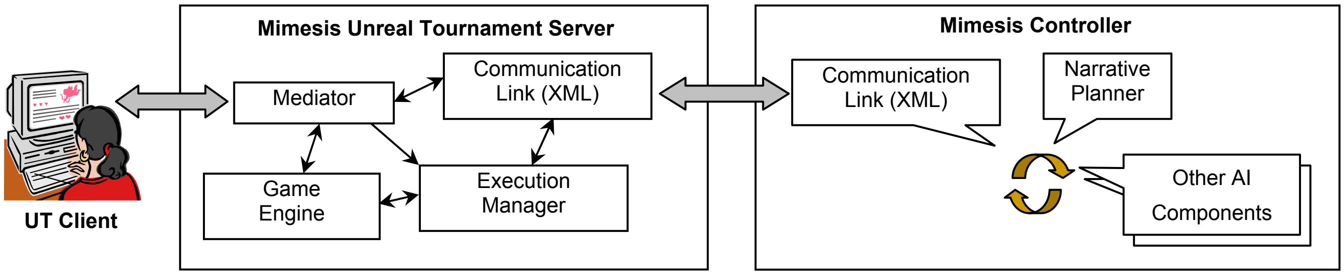


**Figure 1. The Mimesis system architecture.**

two-tiered architecture where responsibility for generating narrative and effective management of user interaction is divided respectively [21, 22]. The upper tier of the architecture implements a suite of intelligent tools for high-level reasoning about narrative structure and user interaction called the Mimesis Controller (MC). In Mimesis, narrative action is modeled as a sequence of character actions represented declaratively as a plan structure generated by a narrative planner. Before an interactive session begins, the narrative planner, based on the Longbow hierarchical partial-order causal link planning system [19], builds a story plan which represents the actions of all the agents in the story world, including those of the user. To build the story plan, the planner relies on a declarative representation of all actions that are available to characters in the story world (including those actions available to the character controlled by the user). The approach used by the planning system to create plans with appropriate narrative structure is beyond the scope of this paper. The plan structures themselves are, however, similar to those used in partial-order, causal link and HTN-style planning systems [14, 16]. The plans contain annotations that explicitly mark the temporal relationships between all actions in the story plan, defining a partial order indicating the steps' order of execution. Other annotations, called *causal links,* are used to mark all causal relationships between the actions in the plan as well. A causal link connects two plan steps $s_1$ and $s_2$ via condition $e$, written $s_1 \xrightarrow{e} s_2$ when $s_1$ establishes the condition $e$ in the story world needed by subsequent action $s_2$ in order for step $s_2$ to execute.

The hierarchical, partial-order, causal link representation of plans used by the narrative planner has several advantages for use by the Mimesis system. First, the plan structures have effectively served as models of human users' representations of complex tasks [20] and are similar in structure to cognitive models of narrative [9]. Second (and more directly related to this discussion), the plans' temporal and causal structure can be put to use to determine when a user's action might interfere with the soundness of the plan's future actions.

The second tier of the Mimesis architecture is built upon Unreal Tournament (UT), a commercially available 3D graphical game engine. This tier is responsible for the low-level execution of the story plan within the virtual environment of the story world. While UT is well-suited as an engine for building conventional 3D interactive game titles, the internal representation of the environments that it models are procedural [4]; user input results directly in the execution of specific code that modifies the game environment. Consequently, direct integration of the UT game engine with intelligent software components that utilize declarative models is not straightforward. To facilitate this integration, Mimesis replaces UT's mechanisms for controlling its virtual environment with a customized version of the game engine called the Mimesis Unreal Tournament Server (MUTS). The MUTS employs procedural representations of actions that mirror the declarative plan operators used by the MC's planning system. This parallel representation approach is similar to the relationship between task and method laid out in [6]. A more-detailed description of the Mimesis architecture can be found in [22].

## 3.3 Detecting and Managing Unanticipated User Activity

As described above, Mimesis drives the action within its story world based on the structure of a plan produced by a narrative planner. As users issue commands for their characters to perform actions within the story world, these actions must be checked against the narrative plan to determine how they fit with the plan's structure. Within Mimesis, each action $\alpha$ performed by the user can be characterized in one of three ways with respect to the unexecuted portion of the plan. One possibility is that $\alpha$ is *constituent* to the plan – $\alpha$ matches an action prescribed by the narrative plan for execution by the user, in which case the user is doing exactly the action that the system desires her to do in order to perform that portion of the storyline.

The second possibility is that $\alpha$ is *consistent* with the plan – $\alpha$ is not constituent and none of the effects of $\alpha$ interact with any of the plan's remaining structure. For example, it may be consistent if the user rotates her character in a circle in order to orient herself spatially before walking out of a room, as long as her act of walking out of the room is part of the narrative and is successfully performed during the appropriate timeframe. The third possibility is that $\alpha$ is *exceptional* – $\alpha$ is not constituent and one or more of $\alpha$'s effects threaten the conditions in the world required by future agent actions. Specifically, an exception occurs whenever a user attempts to perform some action $\alpha$, where some effect $\neg e$ of $\alpha$ threatens to undo some causal link $s_1 \xrightarrow{e} s_2$ between two steps, $s_1$ and $s_2$, with condition $e$, where $s_1$ has occurred prior to $\alpha$ and $s_2$ has yet to occur. [1]

If a user performs an exceptional action, the effects of the exception on the virtual world undoes the condition of at least one causal link in the plan, invalidating some or all of the plan's

[1] This discussion uses a propositional language for simplicity when discussing the interaction between causal links and action effects. In practice, a restricted first-order language is used to represent action, and unification is used to determine when a user's action undoes a condition associated with a causal link.

subsequent structure. It is the responsibility of the system to detect exceptions when they arise and to respond accordingly in a manner that balances the need to preserve the coherence of the narrative with the need to preserve the user's sense of control. Within the Mimesis system, response to exceptions occurs in one of two ways. Either the system allows the exception to occur and restructures the narrative plan mid-story, or it prevents the exception from actually executing, in effect coercing the user into compliance with the existing plan structure. We refer to this process of exception detection and response as *narrative mediation*.

### 3.3.1 Determining Responses to Exceptions

There are two mediation strategies that the Mimesis system uses for responding to the user's command to execute an exceptional action. The first strategy is *accommodation*. Accommodation involves executing the user's exceptional action and then restructuring the unexecuted portion of the narrative plan to re-establish any threatened causal connections. Accommodation often requires only small changes to the narrative plan, such as selecting a different but compatible location for an event when the user takes an unexpected turn down a new path. However, accommodation may also involve more substantive changes which can be computationally expensive. For instance, should the user stumble upon the key to a mystery early in the narrative or unintentionally destroy a device required to rescue a central character, considerable re-planning might be required on the part of the MC's narrative planner. A more complete discussion of the process by which efficient re-planning occurs in the context of narrative mediation is beyond the scope of this paper.

The second mediation strategy used by the Mimesis system is *intervention*. Intervention involves altering the user's exceptional action by surreptitiously substituting an alternate set of effects, one in which the "natural" outcome is consistent with both the plan's existing causal constraints as well as the user's model of reasonable outcomes [17]. Intervention is handled by replacing the user's exceptional action with an instance of an action called a *failure mode* [17]. The nature of the failure mode action is similar to the exception action (e.g., both actions might use the same animations within the virtual world, thus appearing identical to the user), but the effects of the failure mode produce results that do not conflict with any of the causal structure in the plan.

For example, suppose the system defines an operation, buy, in which the user purchases a drink from a vending machine for the cost of a one dollar coin. The system developers have defined two all-too-familiar failure modes for the buy operation, broken-buy and refund. The broken-buy failure mode represents the situation where the vending machine takes the coin but does not dispense a beverage. The refund failure mode represents the situation where the coin passes through the vending machine and is returned to the user without the beverage.

Suppose that there is a character, Pat, which is being controlled by the user. The narrative plan requires Pat to possess a coin in order to satisfy the conditions of some future action. If the user decides, instead, that she should use the coin to buy a drink from a vending machine, an exception arises. Because there is only one coin in our world, the system cannot accommodate this exception. When performing intervention in this case, the two failure modes for the buy operation are considered. Broken-buy is discarded because Pat will lose her coin, a condition which itself violates the requirement that Pat have a coin at some later time. Refund, however, ensures that Pat does not lose her coin to the vending machine, and so the Refund action is substituted in place of the user's attempt to buy. The user is disappointed, but a vending machine that refunds a coin is not outside the realm of reasonable experience.

### 3.3.2 Planning for Exceptions

Using planning structures to model narrative is advantageous because the narrative plan lays out the sequence of agent actions over the entire duration of the narrative. Even though the plan that the system creates initially may not be the one that survives to the end of execution, having the entire sequence specified prior to the start of the story affords the INS the ability to predict exceptions. At the beginning of a session, Mimesis generates a narrative plan specifying all the action of the story world. Before the story begins execution, the causal structure of the narrative plan is analyzed for opportunities for possible exceptions to arise. For every causal link, $s_1 \xrightarrow{e} s_2$, the Mimesis Controller identifies every action α such that 1) α could be performed by the user during the interval spanned by the link and 2) $\neg e$ is an effect of α. For every causal link/user-performed action pair, the MC creates an entry in a *mediation policy* – a table whose entries describe user actions and the method by which they are to be mediated if they occur during execution. If a possible exception is to be accommodated, the system preemptively constructs a new narrative plan for use subsequent to the point at which user action would occur, and links the plan to the relevant entry in the mediation policy. Alternatively, if a possible exception is to be handled through intervention, the possible failure modes are linked to the policy's relevant entry.

### 3.3.3 Low-Level Control of Exception Management

Once the MC has built a narrative plan and a mediation policy to accompany it, the plan and its policy are sent to the MUTS, which is responsible for execution of the steps of the narrative plan and for detecting and responding to exceptions. This process is managed by the *Execution Manager*. The Execution Manager receives the narrative plan from the Mimesis Controller at startup and creates from it a directed acyclical graph (DAG) of execution order. Nodes in this DAG represent the operations that all agents are to perform throughout the entire duration of the narrative, while arcs from one node to another represent a restriction that the source node must successfully complete its execution before the destination node's operation can begin. The execution manager schedules execution based on the DAG, executing actions with no unexecuted predecessors in the DAG and updating the DAG to reflect the completion status of all actions that have ended their execution. The Execution Manager, however, cannot execute operations that are designated for execution by user-controlled characters. When scheduling a user action for execution, the Execution Manager creates a placeholder operation whose execution suspends until the Execution Manager detects that the user has achieved the intended operation's success conditions.

Of course, there is no guarantee that the user will perform all and only the operations that the system desires her to. To detect exceptions at run-time, the system monitors user actions and characterizes each user command as either constituent, consistent or exceptional. To do this, the Execution Manager passes

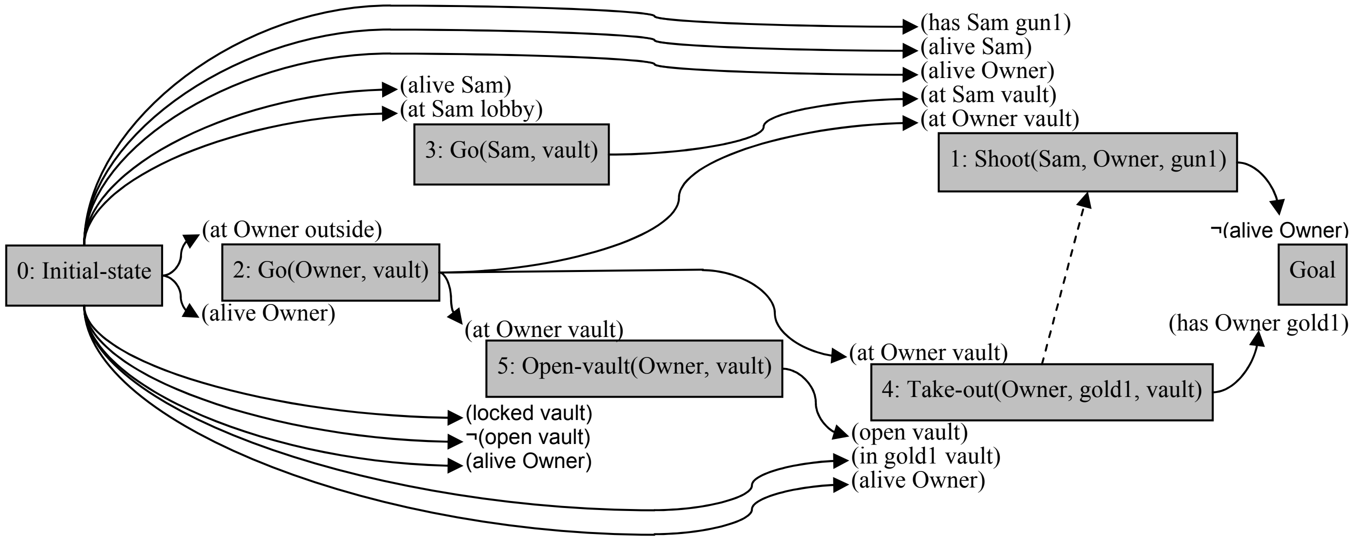


**Figure 2. A narrative plan about a bank robbery.**

commands for actions received from the user to a module named the *Mediator* before the commands are executed. The Mediator compares each user action to the mediation policy for the current plan. User actions that do not match the action of any entry in the policy whose interval covers the current time point are deemed consistent or constituent and are relayed back to the Execution Manager, those actions that are constituent are added to the execution DAG, and the operations are allowed to execute accordingly. User actions that match an entry in the current mediation policy are characterized as exceptions and are then handled through accommodation or intervention, as discussed in the previous section.

Once the response to an exception has been determined and is being implemented, the Mediator also informs the MC of the nature of the exception and its response. When the Mediator accommodates an exception, the MC produces a mediation policy for the new narrative plan put in place as a response to the user's action. When the Mediator intervenes, the MC may also decide to produce a new policy, since, should the user decide to repeat the action that gave rise to the current exception, alternate intervention responses may be appropriate. Section 4 discusses the issues involved in selecting appropriate responses to exceptions in more detail.

## 3.4 Examples

The following is an example of narrative mediation in a simplified story plan. The story is as follows. Sam, the character controlled by the user, is a nighttime guard at a bank. The bank has a vault in which there is a large amount of gold. One night that Sam is on duty, the bank owner comes into the bank, opens the vault and begins removing the gold. Sam's suspicious are aroused and Sam takes action to stop the bank owner from removing the gold.

The narrative plan is shown in Figure 2. In this figure, actions are indicated by gray rectangles and are given unique numbers for reference. Solid arcs between actions indicate causal links with text labels showing the conditions associated with each link. Temporal ordering is indicated in a rough left-to-right order. When temporal ordering is not determined solely by the order imposed by causal links between actions, it is indicated by dashed arrows. Notice that step 3 – Sam goes to the vault – is unordered with respect to steps 2, 4, and 5. The user, controlling the Sam character, can move to the vault at any time during which the bank owner himself is moving to the vault, opening the vault, and taking the gold. The mediation policy associated with the narrative plan is shown in Table 1. This policy is computed by the MC from the plan and a library of operator definitions describing all actions that are available to the user in the story environment. Each entry in this table indicates a potential exception that might arise from the user's actions; as described above, every potential exception for the narrative plan is included in this table. Each entry indicates the exceptional action, the interval during which the action is to be considered exceptional, and the system's response (for brevity, only an indication of intervention or accommodation is shown for the response to each exception. We discuss examples below and indicate further details for each type of response). For example, in Table 1, entry B states that if any agent were to attempt to open the vault in the interval between the start of the narrative and the time the bank owner arrives at the vault, the system should accommodate that agent's action. Policy entry F states that if any agent was to attempt to shoot the bank owner with any weapon between the time that the owner arrives at the vault and the time that the owner had completed opening the vault, then the system should intervene. Note that, even though an exception will occur if any agent performs the shooting, Sam is the only user-controlled character and is therefore the only character subject to the mediation process.

### 3.4.1 Accommodation

A closer inspection of policy entry B shows that if the user were to open the vault before the bank owner were to open the vault himself, then that exceptional user action should be accommodated. Let us suppose that Sam has a key that opens the vault. Further, assume that Sam goes to the vault and opens the vault before the bank owner arrives. If this exception were not detected by the Mediator, the narrative plan would fail: the bank owner would attempt to open the vault even thought the vault door was open and unlocked. However, by analyzing the plan for

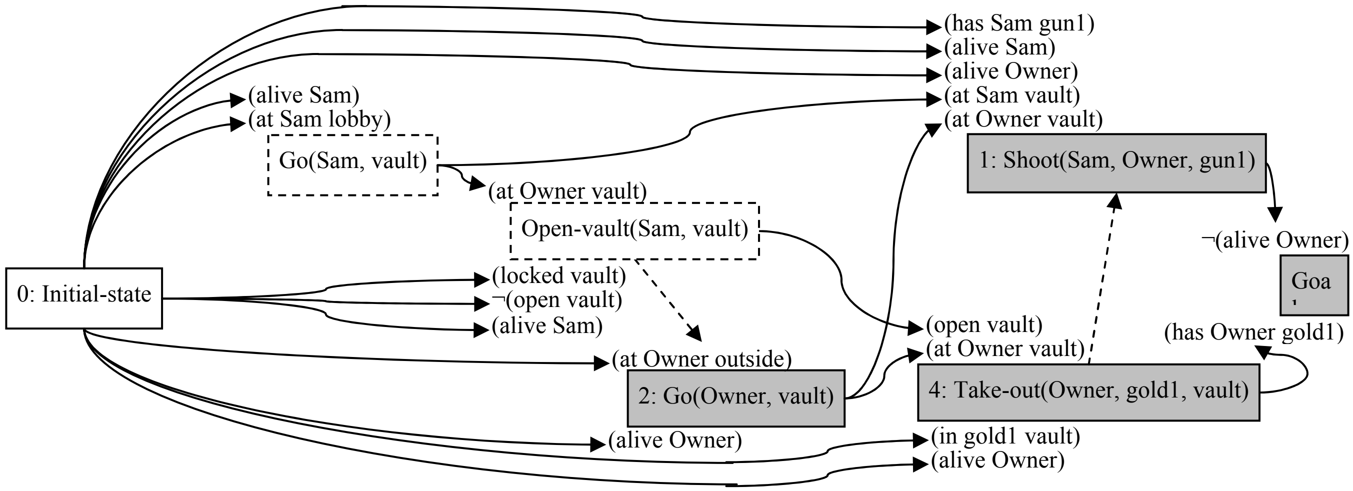


**Figure 3. The revised narrative plan after accommodation (policy entry B).**

potential exceptions before execution of the plan begins, Mimesis can identify this possibility, determine that accommodation is appropriate and construct a revised narrative plan to put to use should the potential exception actually arise. The revised narrative plan is shown in Figure 3. For readability, this figure indicates actions previously executed by the user with dashed rectangles. The plan steps differ from those in Figure 2 in that Sam, rather than the owner, is responsible for opening the bank vault (indicated by the causal link connecting Sam's open action to the owner's take-out action), and the owner no longer performs an open-vault act.

### 3.4.2 *Intervention*

Entry F in the mediation policy shown in Table 1 calls for intervention; should a user's character attempt to shoot the bank owner between the time the owner moves to the vault and the time the owner finishes opening the vault, the resulting death of the owner would invalidate the portion of the plan dependent on the owner opening the vault and removing the gold.

For each exception whose response involves intervention, the mediation policy lists a set of failure modes (not shown in Table 1) whose execution in place of the exception action would maintain the causal constraints of the narrative plan. In our example domain, the system developers have defined two failure modes for the shoot operation: jamming-shoot, in which the gun jams when the trigger is pulled such that no shot is taken and the gun becomes inoperable, and shoot-and-miss, in which the agent fires but the shot goes wide, missing the intended target. The former failure mode is not usable as a response to a potential shoot action performed by Sam because it would render the gun inoperable, making the future step 1 impossible to execute. Therefore, the mediation policy only lists a single failure-mode: shoot-and-miss. In cases where more than one failure mode is applicable, system developers are free to specify any strategy for selecting among all those failure modes that are applicable.

| | Action | Interval | Response |
|---|---|---|---|
| A | Shoot(*?char*, Owner, *?gun*) | <0, 2> | Intervention |
| B | Open-vault(*?char*, vault, *?key*) | <0, 2> | Accommodation |
| C | Take-out(*?char*, gold1, vault) | <0, 2> | Intervention |
| D | Shoot(*?char*, Sam, *?gun*) | <0, 3> | Intervention |
| E | Go(Sam, *?there*, lobby) | <0, 3> | Accommodation |
| F | Shoot(*?char*, Owner, *?gun*) | <2, 5> | Intervention |
| G | Open-vault(*?char*, vault, *?key*) | <2, 5> | Accommodation |
| H | Take-out(*?char*, gold1, vault) | <2, 5> | Intervention |
| I | Shoot(*?char*, Owner, *?gun*) | <5, 4> | Intervention |
| J | Take-out(*?char*, gold1, vault) | <5, 4> | Intervention |
| K | Shoot(*?char*, Sam, *?gun*) | <3, 1> | Intervention |
| L | Go(Sam, *?there*, vault) | <3, 1> | Accommodation |

**Table 1. Mediation policy for the narrative plan.**

## 4. TOWARDS SELECTING AND MANAGING RESPONSES

The previous example demonstrates how the Mimesis system behaves with a particular mediation policy. However, the example does not discuss how the system chooses between accommodation and intervention for each possible exception in the policy. A number of factors go into this decision. When the user performs an exceptional action, one or more casual links leading from actions that have already executed are threatened, that is, the conditions associated with those links will be undone if the exceptional action executes in the story world. While accommodation removes the possible threats in a narrative plan, the changes to a plan's structure imposed by accommodation may remove the only portions of the plan that required the actions whose causal links were threatened; consequently, the execution of those actions will have established conditions that will no longer play a role in the revised plan. Because users understand the narrative in part by drawing causal connections between past, current and anticipated future actions, actions that are left playing no causal role in the plan may lead the user to make unwarranted conclusions about the direction of the story. In this regard, the system must balance the benefits of re-planning with the potential to damage the overall coherence of the story.

On the other hand, when the system intervenes, the user may become aware that a failure mode is being substituted for her command and that her control is being restricted. This is especially true when the user repeatedly attempts to perform the same exceptional action.

The choice between accommodation and intervention is not simply qualitative, however. Some user actions, if executed, may result in the system being unable to generate any plan to achieve the story world's goals. For example, when the user attempts to destroy some resource that is essential to achieving the story's final goal, no alternate plan involving other resources may be available. The following sections summarize the process of constructing mediation policies, taking into consideration the qualitative and quantitative properties of narrative mediation.

## 4.1 Constructing Mediation Policies

While a complete description of the process for constructing a narrative plan's policy is beyond the scope of this paper, the process can be briefly characterized in terms of three tasks. First, the narrative planner generates a plan to drive the action of the story world. Next, the intervals of the plan are analyzed, as described in Section 3.3, to determine all the exceptions that might occur during the plan's execution. Finally, each possible exception is dealt with individually to determine the appropriate mediation response. For each possible exception, the re-planning component of the narrative planner is called to produce a revised plan to be used should the system choose to accommodate the action. Once this new plan is computed, a heuristic function is called to rank the narrative structure of the story world that would result from using the new plan against the narrative structure of the story worlds that would result from using the original narrative plan with an intervention. Each possible failure mode for the exceptional action is considered in turn.

This heuristic function takes three arguments: a *history*, or the plan fragment that has executed in the story world leading up to the potential exception, an *action*, either the proposed failure mode (in the case of intervention) or the exceptional action (in the case of accommodation), and a *future,* a plan fragment specifying the action in the story yet to occur. In the case of intervention, the future contains that portion of the original narrative plan that will execute after the exception. In the case of accommodation, the future contains the new plan created during re-planning. The role of the heuristic function is to provide an overall qualitative evaluation of the story structure of a narrative plan; the response (intervention or exception) associated with the highest ranked action sequence is added as the response entry in the policy for the exception being analyzed.

New policies are sent to the MUTS at several points during an interactive session with Mimesis. As mentioned above, a policy is produced at start-up time, after the initial narrative plan is generated and before the user begins interacting with the system. But new policies are also sent to the MUTS whenever an exception is raised by the user. When the system responds to an exception by accommodation, the MC transmits to the MUTS a policy for the revised plan at the time the response occurs. When the system responds to an exception by intervention, a new policy is also transmitted even though the remaining narrative plan is not modified. A new policy is adopted after intervention because the intervention itself changes the context used to compute the entries for subsequent exceptions in the original policy. By updating the policy at every intervention, the system avoids situations where repeated exceptions give rise to the same intervention. For instance, unless the policy is revised at every intervention, a user playing the Sam character from the example in Section 3.4 may quickly become frustrated as she repeatedly fires her weapon, only to see herself shoot-and-miss every time.

## 4.2 Run-Time Management of Mediation Policy Construction

The process of transferring mediation policy from the MC to the MUTS is on-demand; a policy is sent to the MUTS at start up and one is transferred again as the response to every exception is carried out. In practice, however, the MC builds policies in an opportunistic fashion, taking advantage of times when processing demands on the MC are low, for instance, when users execute sequences of actions that are all consistent or constituent with regards to the narrative plan, avoiding the need for exception handling. At these times, the MC pro-actively computes policies for plans other than the one that is currently executing. By pre-computing policies during times of low-processor demand, Mimesis is less likely to show processing delays in times of high demand, for instance, when exceptions are raised in rapid succession.

In general, the process of constructing policies involves building a *policy tree*; a node in the tree pairs a narrative plan and its mediation policy, and an arc from one node to another connects an exception in the source node's policy to a new plan/policy pair that is to be used as the exception's response. A range of ordering strategies can be imposed on the MC as it constructs policy trees, and each strategy may effect both the quality of the system's response to exceptions and the ability to have responses ready for use at any time exceptions might arise. For instance, the MC can build a policy tree in a breadth-first manner, constructing new policies in the tree first for those exceptions in the current plan that might occur soonest. Alternatively, the MC can build a policy tree using a best-first approach, where new policies are constructed first for those exceptions that are deemed most likely to occur. The current implementation allows for empirical experimentation to determine which combination of search strategies for constructing policy trees leads to situations where the MC has the highest quality policy ready each time an exception arises.

# 5. CONCLUSIONS

In interactive narrative systems, a human agent coordinates her activities with those of other agents in order to bring about a coherent narrative experience. While the user is encouraged to play the part of a story world character, she typically only has partial knowledge of the narrative and is prone to perform actions that disrupt the plan execution of other agents. Narrative mediation is a technique for detecting and responding to exceptional user actions in order to preserve the coherence of the narrative. In this paper, we present two forms of narrative mediation: accommodation and intervention. Accommodation integrates exceptional user actions into the narrative fold through the use of re-planning. We show that a narrative plan can be analyzed for the purpose of anticipating exceptions and that the narrative plan can be preemptively revised in order to accommodate exceptions. Intervention is the substitution of a failure mode – a similar action with different effects that do not jeopardize narrative coherence – for an exceptional user action.

Both accommodation and intervention, however, have implications for the overall quality of the narrative experience, particularly with respect to the user's perceived levels of control and narrative coherence.

Narrative mediation is an effective method for managing the interaction between human and computer-controlled agents in a narrative setting. However, there are several limitations to the approach we have defined. Two central limits are listed below:

- Locality of *exceptions*. As described here, an exception is raised when a user issues a command for a single action whose effects, if executed, would violate the structure of the narrative plan. Often, however, a user might execute a *sequence* of actions in preparation for the exceptional act. For instance, a user might steal money from a bank teller by first stealing a car, writing a note demanding money from the teller, driving to the bank, etc. A system that made effective predictions about the plan being executed by the user might find opportunities for intervention or accommodation before potentially extreme responses to exceptions are required. (see [3] for an overview of related work on *plan recognition*).
- Locality of *intervention* [17]. As described here, intervention involves the substitution of a *single*, alternate user action in place of the action that a user intends to perform. However, it is possible that intervention be assisted, meaning that a sequence of actions, some carried out by other agents, could be executed in order to intervene. For example, another character can move to jostle a user's character just as she is firing a weapon, causing the user to miss her target.

Our current work addresses these limitations as well as (a) the development of heuristics for selecting between intervention and accommodation in the construction of mediation policies (see Section 4) and (b) a more complete characterization of the computational requirements for these mediation algorithms.

## 6. ACKNOWLEDGMENTS

The authors wish to thank the members of the Liquid Narrative Group at NC State University for their assistance in developing the system described here. The work has been supported by National Science Foundation CAREER award 0092586.

## 7. REFERENCES

[1] Blumberg, B.M. & Galyean, T.A. (1995). Multi-level direction of autonomous creatures for real-time virtual environments. In *Proceedings of SIGGRAPH '95*, 47-54.

[2] Bruckman, A. (1990). The combinatorics of storytelling: Mystery train revisited. Unpublished manuscript.

[3] Carberry, S. (2001). Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2), 31-48.

[4] Cavazza, M., Charles, F., & Mead, S.J. (2002). Character-based interactive story-telling. *IEEE Intelligent Systems*, 17(4), 156-162.

[5] Egri, L. (1946). The Art of Dramatic Writing: It's Basis in the Creative Interpretation of Human Motives. New York: Simon & Schuster.

[6] Firby, R.J. (1994). *Adaptive Execution in Complex Dynamic Domains*. Ph.D. Thesis, Yale University Technical Report YALEU/CSD/RR #672.

[7] Galyean, T.A. (1995). *Narrative Guidance of Interactivity*. Ph.D. Dissertation, MIT Media Arts and Sciences Program.

[8] Gerrig, R.J. (1993). Experiencing Narrative Worlds: On the Psychological Activities of Reading. Cambridge, MA: Yale University Press.

[9] Graesser, A.C., Lang, K.L., Roberts, R.M. (1991). Question answering in the context of stories. *Journal of Experimental Psychology: General*, 120(3).

[10] Grosz, B. & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence*, 86(2), 269-357.

[11] Hayes-Roth, B., Brownston, L., & van Gent, R. (1995). Multiagent collaboration in directed improvisation. In *Proceedings of the 1st International Conference on Multi-Agent Systems*, San Francisco, CA, 1995.

[12] Laird, J.E., Newell, A., & Rosenbloom, P.S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*, 33(1), 1-64.

[13] Lombard, M. & Ditton, T. (1997). At the heart of it all: the concept of presence. *Journal of Computer-Mediated Communication*, 3(2).

[14] Penberthy, J. & Weld, D. UCPOP: A sound, complete, partial order planner for ADL. In *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, pages 103-114, Cambridge, MA, 1992.

[15] Rickel, J. & Johnson, W.L. (1999). Animated agents for procedural training in virtual reality: perception, cognition, and motor control. *Applied Artificial Intelligence*, 13: 343-382.

[16] Sacerdoti, E. D. *A Structure for Plans and Behavior*, American Elsevier, New York, 1977

[17] Saretto, C.J. (2001). *Mediating User Interaction in Narrative-Structured Virtual Environments*. Masters Thesis. Department of Computer Science, NC State University.

[18] Swartout, W., Hill, R., Gratch, J., Johnson, W.L. et al. (2001). Toward the holodeck: Integrating graphics, sound, character and story. In *Proceedings of the Fifth International Conference on Autonomous Agents*, May 2001.

[19] Young, R.M., Pollack, M.E., & Moore, J.D. (1994). Decomposition and causality in partial-order planning. In *Proceedings of the Second International Conference on AI and Planning Systems*, 188-193, Chicago, IL, 1994.

[20] Young, R.M. (1999). Using Grice's maxim of quantity to select the content of plan descriptions. *Artificial Intelligence*, 115, 215-256.

[21] Young, R.M. (2001). An overview of the Mimesis architecture: integrating intelligent narrative control into an existing gaming environment. *The Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment*, 78-81, Stanford, CA, March 2001.

[22] Young, R.M. & Riedl, M.O. (2003). Towards an architecture for intelligent control of narrative in interactive virtual worlds. In *Proceedings of the 2003 International Conference on Intelligent User Interfaces*, 310-313.

[23] Weyhrauch, P. (1997). *Guiding Interactive Fiction*. Ph.D. Dissertation, Carnegie Mellon University, Pittsburgh, PA.