Conciliating Coherence and High Responsiveness in Interactive Storytelling

Marcelo M. Camanho§  Angelo E. M. Ciarlini§  Antonio L. Furtado‡  Cesar T. Pozzer¥  Bruno Feijó‡

§UNIRIO
Dep. de Informática Aplicada
Av Pasteur, 458 – Térreo
Rio de Janeiro, Brazil
marcelo.camanho@uniriotec.br
angelo.ciarlini@uniriotec.br

‡Puc-Rio
Dep. de Informática
Rua Marquês de São Vicente, 225
Rio de Janeiro, Brazil
furtado@inf.puc-rio.br
bruno@inf.puc-rio.br

¥Universidade Federal de Santa Maria
Dep. de Eletrônica e Computação
Campus Universitário – pr. 07, 3º and.
Santa Maria, RS, Brazil
pozzer@inf.ufsm.br

ABSTRACT
Interactive storytelling is a new form of digital entertainment that brings together techniques and tools for the creation, visualization and control of interactive stories through electronic means. One of the main challenges of interactive storytelling is the generation of stories that are both coherent and interesting, while providing means to allow the user to interact with the story. On the other hand, user's intervention should not violate the rules of the chosen genre, leading to unlikely developments. Another problem arises when storytelling environments require high responsiveness, as is the case in interactive TV. The fact that storytelling systems are usually very CPU intensive should not affect responsiveness. This paper describes a model to deal with these requirements, which has been implemented over a previous version of Logtell, a logic-based tool for interactive generation and dramatization of stories.

1. INTRODUCTION
In recent years, different approaches for interactive storytelling systems have been proposed. Some of them are oriented to filmmaking, others to games and others to authoring literary texts. Techniques and strategies adopted in interactive storytelling systems depend essentially on their purpose: systems might actually create stories or just tell them in different ways; stories could be told with a first- or a third-person viewpoint; and the focus of the narrative could be either on the characters or on the plot, a well-understood design dilemma.

In interactive storytelling systems, keeping the consistence of the narrative and a good level of interactivity poses a constant conflict: the more the user can interact with the story, the higher is the chance of inconsistencies. There are interactive storytelling models that try to avoid losing track of the story by using more solid structures, which demands more manual effort and closer attention to details on the part of the author. They offer restricted options in specific points of the narrative, thus guaranteeing consistency and direction, but reducing the number of variations, the effect of the user's interaction, and hence the potential to entertain and surprise. Other models use logic structures to define how the characters and/or the plot work, specifying rules to model their behaviour. They offer many more opportunities of interaction at the expense of higher difficulty to keep coherence and higher computational effort.

There are many possible applications for interactive storytelling systems. They can serve not only as a way of entertainment, but also for training and decision support. For entertainment, the generated stories serve to guide the dramatization of events inside games, and also for replacing hand-coded pre-established scripts, with a better chance to introduce interesting and surprising variations. Another use is the application of interactive storytelling in the context of digital TV, where users would play, in principle, a more passive role than in games, but may still want to interact with the story in diverse ways.

The exploration of the possible forms of interaction in digital TV is a very recent research area. With the beginning of the
dissemination of this technology around the world, many possibilities are appearing to take advantage of this as yet poorly explored capacity of direct interaction with thousands of spectators. This new level of interaction offers much promise for education, cultural dissemination and entertainment. Interactive storytelling techniques will certainly be very important in this context.

Besides the tension between coherence and interactivity, when we consider the use of interactive storytelling systems in media like TV, we have to take into account that such media demand high responsiveness. The user must not be kept waiting while the story is being created, although the amount of interactivity may vary depending on the user’s preference. Even a lazy interactivity may be appropriate, in case the user just wants to watch the story unfold by itself. Computational power of set-top boxes is also a factor, since it tends to be more limited than that of a PC. Finally, simplicity and comfort of the interaction is essential, particularly in stories to be watched for a long time by users who are not required to be expert game-players.

In this paper, we describe an extension of the interactive storytelling system Logtell [5]. We have been developing this extension to cope with media that demand high responsiveness, such as interactive digital TV. The approach adopted in Logtell is attractive for TV, because it is essentially committed to the coherence of stories and, at the same time, it enables the user to interact with the story at different levels. The user can choose between not intervening in the story, or merely ask for alternatives, but may on the contrary have a strong participation, establishing that certain events and situations should occur. Virtually, any intervention that is logically compatible with the model specified for the genre is allowed. The main problem to be faced is that, in order to guarantee the coherence of stories, time-consuming simulations would be needed, making it hard to conciliate coherence and high responsiveness. The paper outlines the new general architecture and the main strategies adopted to achieve a suitable compromise.

Section 2 discusses interactive storytelling models and techniques. Section 3 gives an overview of the Logtell approach for interactive storytelling. Section 4 presents the new architecture, designed to meet higher responsiveness requirements. Section 5 examines in detail the strategies for conciliating coherence, interactivity and responsiveness. Section 6 contains concluding remarks.

2. STORY MODELS AND TECHNIQUES
In character-based storytelling systems [3,4,22], the story emerges from the real time interaction between autonomous agents, each one with its own objectives. The main advantage is to facilitate user intervention, since any action of any character in the story can be influenced, so that the plot may take different directions. In such cases, however, the level of interference may lead into implausible turns, or cause the loss of essential events, making it hard to guarantee that the emerging situations will be interesting enough to create a good experience for the user.

Some other systems have a plot-based approach [11,16], wherein a series of rigid rules, built into the plot, guide the narrative, making user intervention far more limited. In such approach, there is a stronger control over the story being presented, preventing the user to stray from the context defined by the author. The general given structure usually includes beginning, middle and ending points previously fixed by the author, and user interaction affects only the way the story will reach these predefined points. One of the main inspirations for this kind of model has been the seminal literary work of Vladimir Propp [19], at the beginning of the twentieth century. Propp examined a large number of Russian fairy tales, and showed that they could all be described by specializations of 31 typical functions, such as villainy, hero’s departure, reward, etc. In purely plot-based models, the intervention of the user is more limited, but it is much easier to guarantee coherence together with a measure of dramatic power.

There are also alternatives that integrate characteristics of both plot-based and character-based approaches. The interactive system Façade [14], for instance, has a drama manager that keeps the characters largely autonomous most of the time, but changes their behaviour to move the plot forward, conciliating higher-level goals, which are essential to the story, with lower-level goals, specific to the autonomous behaviour of the characters. Erasmatron [7], in contrast, starts from the notion of verbs and sentences. Actions are represented by verbs with roles assigned to characters to form sentences.

Automated planning algorithms are important parts of some storytelling systems, because they can be used to create a logical chaining of events. In [3], for instance, hierarchical task network (HTN) planning is used to control the way characters achieve their goals in accordance with user intervention. HTN planning tends to be efficient but less general, demanding the previous construction of a task hierarchy and methods to perform each task. More flexible planning algorithms have also been adopted, as in [20] for example. Such algorithms do not limit solutions to previously defined alternatives; instead of that, they combine events conciliating different objectives with pre-conditions and effects of each event. The computational effort, however, becomes even greater, due to the complex nature of automated planning.

In the next section, the Logtell approach for interactive storytelling is detailed. In an effort to guarantee coherence while providing a good level of interactivity, it tries to balance character-based and plot-based characteristics, and is basically supported by planning algorithms.

3. LOGTELL APPROACH
Logtell focuses on exploring the generation and dramatization of an ample variety of stories compatible with a formal description of a chosen genre. The main idea is to enable the user to intervene in the story up to the limit that the story still makes sense and stays within the conventions of the genre. Plot generation is performed by a module called IPG (Interactive Plot Generator) [6] in multiple steps of inference of goals, planning and user interference. Plots generated by simulation, with a varying degree of user participation, can then be dramatized by virtual actors in a 3D virtual environment.

Inspired by Vladimir Propp’s ideas, Logtell tries to go one step further and combine the plot-based and character-based approaches, extending the informal notion of functions defined by Propp. The typical events are described through parameterized operations with pre- and post-conditions, such that planning
different modules, in charge of the generation, interaction and visualization of stories, as shown in Figure 1.

The context for story generation is stored in the context database and accessed via the Context Control Module (CCM). The context contains the description of the genre according to which stories are generated and the initial state, describing characters and the environment at the beginning of the story. The genre is basically described by:

- a set of parameterized operations, with pre- and post-conditions, logically specifying the events that might occur; and
- a set of goal-inference rules, specified with a temporal modal logic, which specify situations that lead characters to pursue the achievement of goals.

The main interface of the system is the Plot Manager. The story generation, which is done by the Interactive Plot Generator (IPG), is controlled by the Plot Manager. The Plot Manager receives the current partially generated stories, and then allows the user to interfere with the generation process. To visualize the dramatization of a story (partially or completely generated), the user must first choose a total order of the events, compatible with the partial order requirements identified by IPG.

As mentioned earlier, the Interactive Plot Generator module performs simulations by inferring goals and using automated planning for the achievement these goals. In order to do that, the context of the stories is consulted. Restrictions on partial stories, informed by the user to the Plot Manager during the interaction phases, are taken into consideration while the simulation process continues.

The Drama Manager, responsible for the dramatization of the story, controls actors to play each character in a 3d environment. During the dramatization, the Drama Manager consults IPG to keep the logic and the graphical representations coherent.

The main focus of the system regarding interaction is to allow the user to explore the coherent alternatives that are possible for the story being composed, in tune with the typical events and goal inference rules modelling the characters, conforming to the chosen story genre. The user's control is exerted only via the Plot Manager, which in turn communicates with IPG to create the story and maintain its coherence, and with the Drama Manager to control the story's visualization. Created in Java, the Plot Manager thus functions as a user interface, enabling the user to choose the events that will go into the plot as well as their order (Figure 2). Each event is represented by a box with a color indicative of its state.

It should therefore be stressed that interaction is always mediated, since the user cannot act directly over the scene or the characters, and every interaction must be validated by IPG before being incorporated to the story. The fact that generation and dramatization are done separately represents one difference between Logtell and most of the character-based approaches. Only during the simulation phase the user has the real opportunity to affect the plot's outcome.

The plots result from the objectives supplied by the goal-inference rules. On each step of the simulation, new goals are added and events are planned in order to reach the objectives. The inserted events are sent to the Plot Manager and submitted to the user. In the Plot Manager, there are 2 main controls for the automatic generation of the story: another and continue. The another command asks IPG to backtrack so as to offer an alternative solution to reach the same goal of the recently finished (but still not committed) step. The continue command asks IPG to infer new goals and go along with the stepwise simulation process.
The user's interventions can be classified as weak or strong. In the weak intervention, the user applies only the previously explained controls, and puts the events in a preferred order-compatible sequence. This kind of interaction, admittedly, does not go much farther from the basic version of the story conceived by the author.

Besides weak intervention, strong intervention is permitted in two ways. One is the insert situation command, which allows the specification of goals to be reached — whereas how they will be reached is up to IPG, in a supplementary planning task. Note that IPG may signal a failure if no valid plan is possible for the indicated goal, or if the computational effort exceeds the configured search limits for the planner. As in the automatic generation, the user can confirm the presented solution by replying continue, or ask for an alternative, via the another command. A second way to do a strong interaction is the explicit insertion of events, by using the insert event command. Similarly to the previous case, the continue command must be applied next for validation with respect to the current plot. If the preconditions of the inserted event are not satisfied, IPG will propose the insertion of additional events. Naturally the user has the option to undo the intervention, by removing events not yet incorporated, which is the normal thing to do for those that were rejected.

Recall that before calling for dramatization, at the end of any generation step, a fully ordered sequence must be selected for the current plan. If the simulation process is resumed afterwards, this selection, which implies the addition of temporal restrictions, must also be regarded as an intervention, since it affects the inference of new goals. To determine the sequence, the user draws edges connecting the event boxes in an order of his liking, respecting the temporal restrictions enforced by IPG. The events already connected are shown in yellow, the events that can be immediately connected become green, and the ones that cannot be connected remain red. The initial node is blue and the event that is currently being dramatized becomes cyan. Any time the plot (or part of it) is thus manually connected, the linked sequence can be dramatized by using the render command.

Figure 3: The dramatization of a scene

Even though Logtell already allows the creation of a variety of plots with different degrees of user intervention, while preserving logical coherence and faithfulness to the specified model, one can notice that Logtell's style of interaction still looks somewhat unnatural, and is perhaps too complex for an interactive television environment. Also, carrying simulation separately from dramatization does not seem convenient in such environments, because the user is kept waiting for the completion of the simulation steps before watching the dramatization. Therefore, establishing a coordination model where plot generation can occur parallel to dramatization should be regarded as a fundamental concern.

Opportunities for both weak and strong interactions in the story's composition should be offered to the user in a comfortable and understandable way. In summary, it is desirable that the new coordination model make it possible to orchestrate simulation, interaction with the user, and dramatization, attaining the high responsiveness that is expected in media like interactive TV, and keeping, or even expanding, the forms of interaction allowed by Logtell.

4. NEW ARCHITECTURE

In order to deal with the demand for high responsiveness, the initial architecture of Logtell was modified as presented in Figure 4. The main modification is that a client-server model was adopted. The client-side is responsible for user interaction and dramatization of stories. The application server side is responsible for simulation. All processes run now in parallel and are coordinated with each other. For each story being told there is an application running in the server, while one or many applications are kept running in different clients. This takes care of the case wherein multiple users are simultaneously sharing the same story.

If clients are set-top boxes for interactive TV, their computational resources are limited, making it hard to perform CPU-intensive tasks such as automated planning. By concentrating simulation tasks in application servers, it is easier to achieve higher scalability. In addition, it is also easier to exert control when a single story is shared by many users.

The Plot Manager figuring in the initial architecture was responsible for controlling the interaction with the user and for the activation of the simulation (IPG) and dramatization modules (Drama Manager). In the new architecture, the Plot Manager was split into three modules: the Simulation Controller, the Interface Controller and the User Interface. The Simulation Controller and the Interface Controller run on the server and the Graphical User Interface runs on the client.

On the client side, the user interacts with the system via the User Interface, which informs the desired interactions to the Interface Controller placed at the server side. The Drama Manager requests the next event to be dramatized from the Drama Manager, and controls actor instances for each character in a 3D environment running on our Graphical Engine.
On the server side, the Interface Controller centralizes suggestions made by the various clients. When multiple users share the same story, interactions are selected according to the number of clients that requested them. When there is only one client, suggestions are automatically sent to the Simulation Controller. The Simulation Controller is responsible for:

- informing the Drama Manager at the client side the next events to be dramatized;
- receiving interaction requests and incorporating them in the story;
- selecting viable and hopefully interesting strong interactions to be suggested to the users;
- controlling a number of instances of the Interactive Plot Generator, in order to obtain the next events to be dramatized; and
- controlling the time spent during simulation.

In the new architecture, there can be various instances of IPG running on the server. Besides the instance corresponding to the current flow of the story, others are used to avoid interruption in the dramatization. The simulation has to be some cycles ahead of the dramatization to keep responsiveness. When there is no user intervention, goals are inferred and events are planned continuously. When users interact with the system, however, they interact in accordance with the events that are currently being dramatized. The Simulation Controller keeps snapshots of the state of the simulation after each cycle, so that simulation can be resumed from the correct point after an intervention. Logical coherence of a requested intervention is always checked before being incorporated, or either discarded if inconsistent. When an intervention is incorporated, simulation has to discard simulation cycles that were previously planned without taking the intervention into account. In order to be prepared for interventions, the system creates other instances of IPG, simulating the incorporation of strong interventions to be suggested to the users. But the suggestions are only communicated if the IPG instance confirms that they are consistent. And if they are accepted, the next events are already planned and there is little risk of interruption.

The time spent for simulation is constantly monitored by the Simulation Controller. When there is risk of interruption in the dramatization because there are not enough events planned, a message is sent to the Drama Manager, so that strategies can be used to extend the dramatization of the current events until the situation is normalized.

The access of all modules to the context of the stories, specified in the Context Database, is always performed via the Context Control Module, which runs in the server.

5. STRATEGIES FOR HIGH RESPONSIVENESS

The strategies for incorporating demands of high responsiveness in Logtell address three issues: the adoption of interaction alternatives at various levels such that the system is able to conciliate the focus on the coherence of the stories with the new requirements; the definition of an adequate load balance between tasks performed on the server side and on the client side; and methods for enhancing planning capabilities, in order to speed up simulation. In this section, we describe how these strategies can be configured in accordance with the architecture presented in Section 4.

5.1 Interaction Alternatives

Considering that our storytelling system is designed to run on an environment such as digital television, viable alternatives of interaction should not hinder the experience of watching the dramatization. Interaction cannot demand a high level of attention, unless the user opts to halt the dramatization in order to interact. The interruption of dramatization to allow the user to interact, which was mandatory in the first version of Logtell, is still allowed, but it is expected to be an exceptional case, being replaced by other more expedient kinds of interaction.

The Simulation Controller plays an important role in the implementation of the new forms of interaction. One of its most important tasks is to generate interaction suggestions for the user while the dramatization plays on the client side. Suggestions can correspond to weak and strong interactions.

Suggestions of weak interactions work basically around the “normal” flow of the story, as one would have with the “continue” and “another” commands in the first version of Logtell. The Simulation Controller directs the flow of the story by automatically selecting alternatives and total order of the events to be dramatized. Such selections can be done either in a random way or based on user satisfaction models. The idea is that stories are worth of telling even if the user only watches the dramatization, with no intervention. Even in this case, we keep the possibility of watching different but still coherent stories based on the same initial configuration.

Other kind of weak interaction is to return to a previous point in the narrative, so that alternative directions for the plot can be chosen. In the first version of Logtell, at the end of a simulation phase, the user could examine the new events not yet incorporated, and decide whether or not to consider them interesting; if not, the user would ask for the generation of an alternative. With parallel dramatization, it becomes highly desirable to extend the backtracking range, to allow the user to undo the narrative up to any previous stage, and have a chance to find how it would develop if different alternatives were chosen at such point. For this objective, different snapshots of the story are kept in memory by the Simulation Controller, corresponding to the end of each simulation cycle.

Suggestions of strong interactions correspond either to the insertion of specific events in the plot or to the directive that a certain situation should occur at a specific time. These suggestions can be made based on the events already inserted in the story and on an analysis of the context of the genre. We shall describe two methods for their creation. Independently of the method utilized, the Simulation Controller checks the consistency of the suggestion with the current narrative, before sending it to the user.

The first method to obtain a suggestion of strong interaction uses a library of typical plans, organized as a hierarchy of basic and complex events. Typical plans usually consist of certain combinations of events whereby the various characters pursue their goals, but they can also correspond to *motifs*, i.e. recurring
structures compiled in the course of critical studies on the genre [1]. IPG contains a procedure for the recognition of plans, based on an algorithm specified by Kautz [13]. The procedure is able to discover that some given events are compatible with a motif for which we have a typical plan, enabling the Simulation Controller to suggest the inclusion of additional events contained in the plan.

The second method to obtain a suggestion of strong interaction is based on the application of the goal-inference rules against the current situation of the narrative. A suggestion, in this case, is triggered by the occurrence of a combination of facts leading to the inference of a cogent new goal.

Besides the strategies for incorporating the weak and strong interactions that we had in the first version of Logtell, further kinds of interactions are under consideration:

- **Use of abstract concepts** – The library of typical plans admits two types of relationships between events. Events can be part of other events, but they can also be specializations of a more abstract event. Moving up along these is-a links, the insertion of events could be done at a more abstract level. The user would, in the context of fairy tales for example, indicate that the princess must suffer villainy, but leave the choice of the type of villainy to the automatic simulation process. Analogously, if a hierarchy of predicates is defined, the user could establish that the princess be in danger at a certain time, but without specifying how. A suitable level of abstraction encourages the user to intervene in the story without having to specify details that may be unattractive or not well understood. Conversely, this leaves more flexibility to the simulation process to deal with user requests.

- **Tuning the narrative tensions** – Numeric scales referring to levels of violence, romantic turns, as well as the speed of the narrative, etc. could be manipulated by the user while watching the dramatization. These settings would help guide the simulation in the inference of new goals and in the planning process. Of course this requires a considerable investment when modelling the genre [17]; for instance, certain events would be tagged as violent (e.g. abduction), as opposed to others of a similar effect (e.g. elopement).

- **Multi-user modes** – Having in mind the massive aspect of interactive television, it is worthwhile to analyze ways to allow more than one user to have influence on the evolution of a shared story. Such mechanisms would include polls, whereby the most voted options would be chosen at goal planning crucial moments of story creation. In the new architecture, the Interface Controller organizes the interaction among users and interacts with the Simulation Controller as if a single user were present. As a rule, any suggestions should be sent by the Interface Controller to the Simulation Controller only when they receive a prescribed level of support in a poll.

- **Natural language suggestions** – The user should have the option to communicate with the system by entering phrases in (a restricted subset of) natural language. After being parsed, these would be treated just like all forms of interaction (already available or under development): that is, the logical consistency of the suggestions would be checked as always. In a multi-user model, suggestions expressed in natural language might more easily be discussed among the users.

### 5.2 Load Balance

In the first version of Logtell, processes of simulation, interaction and dramatization ran on a single machine. For an environment like interactive digital television, this would be hardly a good choice, given the highly distributed nature of the environment and the high processing demand.

In the new architecture, we leave the dramatization to be done by the client, and the simulation on a server machine, making use of a distributed architecture like Enterprise Java Beans, which offers many advantages such as its almost transparent scalability. Simulation demands a considerable amount of memory and CPU resources. Taking into account that clients will probably be set-top boxes, with more limited computational power, assigning simulation tasks to clients does not sound reasonable. On the other hand, concentrating all simulations in only one server could be a problem, especially if each user is interacting with a different story. Our solution is to have a pool of application servers working in parallel, each one controlling one story. As different application servers will access the same structures in the context database, replicas of the database will eventually be necessary to avoid creating a bottleneck.

Depending on the amount of resources in the client, different alternatives might be compared. If computational resources are very limited, the client side could have only the user interface, delegating the dramatization to the server. In this case, however, problems due to the huge amount of multimedia data being transmitted may arise. If the computational power of the set-top boxes is enough to cope with the simulation, the server side could be much simpler, being responsible only for controlling the context of the stories. But then the sharing of the same story by multiple players might become much more complicated.

### 5.3 Efficient Planning

In the first version of Logtell, we used a non-linear planner implemented in Prolog, which was adapted from [21], with a number of extensions. In a non-linear planner, a least-commitment strategy is adopted: the order of events is partial and constraints on the order of events and on the values of variables are established only when it is strictly necessary. In this kind of planner, it is easier to find solutions conciliating various goals. Features to permit the abandonment of goals were included, and also constraint programming techniques for dealing with numerical pre-conditions. The planner has great flexibility to deal with different goals inferred for different characters and to check compatibility and consistency of user strong interventions in the plot. IPG contains also, as mentioned earlier, a procedure for the recognition of plans including a given set of observed events. We employ it as a powerful complement to plan generation, inside the plot composition process. Specifically, a plot can be generated by informing some events, recognizing a plan in which these desired events take part, and then starting a simulation run to validly incorporate the recognized plan, fully or partly, in the plot being generated.

To meet the requirement of high responsiveness, efficiency of the planning algorithm becomes essential, and possible enhancements to the planner deserve special attention. Most of the successful cases in real-world applications of automated planning are based on algorithms that use hierarchical task networks (HTNs), such as
These algorithms tend to be more efficient because they reduce the search space. They depend however on the previous definition of a hierarchy of tasks and methods to perform the tasks. The task network has to be built for each domain, and all generated solutions are limited to combinations of previously defined ones, giving less flexibility to the creation of alternatives. Algorithms based on HTNs can work with a partial order of events, and HTNs are compatible with the way we specify a hierarchy of typical plans. Taking advantage of that, we intend to enhance our planner, producing a hybrid planner capable of working with our type of simulation, mixing non-linear planning with HTN planning. The idea, on the one hand, is to use HTN planning whenever possible, applying least-commitment-strategies to combine parts of HTN plans. But, on the other hand, the ability to, occasionally, generate plans from scratch would not be lost. By doing this, we hope to enhance performance and, at the same time, keep the generality and flexibility of our original simulation tool.

In recent years, some neoclassical planners that work in the space of states, with planning-graph techniques [2,10], have given good results. The use of heuristics and control strategies to prune the search space, as in [8, 12], are also promising strategies. We are also studying the possible adoption of these techniques.

6. CONCLUDING REMARKS

In this paper, we presented a new architecture for the interactive storytelling system Logtell, together with a set of strategies we have been implementing in the new version of the system. The new architecture and strategies are intended to conciliate the strong emphasis on the coherence of stories and the various alternatives of user interaction, present in Logtell, with demands of high responsiveness of environments such as interactive digital TV. In spite of being mainly inspired by demands of digital TV, the proposed solutions are compatible with other environments, such as the Web.

Besides the effort to coordinate simulation and dramatization in real time, we are currently looking into ways to provide more variety for the dramatization of events, and to allow the user to directly interact with the story while the events are being represented. An ample variety of possibilities for dramatization enriches the storytelling experience. In addition, the Drama Manager is given opportunity to apply policies for controlling the duration of each event, according to the needs of simulation control. The direct interference of the user in the course of an event can be useful to enhance the feeling of immersion. The main difficulty to be overcome is that any change in the dramatization, with or without user interference, has to be consistent with the overall plot that is being represented.

In order to achieve realism and good quality in the narration of stories by Logtell, several other research topics are being investigated, including models for representing beliefs and emotions of characters, alternative ways to control the camera, and automatic text generation for supporting dialogs and the explicit narration of stories.

Finally, we should stress that the techniques already incorporated in Logtell, as well as those still under development, have applications that are not limited to entertainment purposes. They are useful, in particular, for e-learning and decision support, since our simulation runs allow the users to analyze what may or may not occur within a specific context. The effort to cope with requirements of high responsiveness is crucial to make the interactive storytelling experience more pleasant when the purpose is simply entertainment, and in addition more productive when directed to learning or decision support.

7. REFERENCES


