Towards an Architecture for Intelligent Control of Narrative in Interactive Virtual Worlds

R. Michael Young
Liquid Narrative Group
Department of Computer Science
Box 7535, NC State University
Raleigh, NC 27695
011.919.513.3038
young@csc.ncsu.edu

Mark Riedl
Liquid Narrative Group
Department of Computer Science
Box 7535, NC State University
Raleigh, NC 27695
011.919.513.3038
moriedl@eos.ncsu.edu

ABSTRACT

The creation of novel, engaging and dynamic interactive stories presents a unique challenge to the designers of systems for interactive entertainment, education and training. Unlike conventional narrative media, an interactive narrative-based system may be required to generate its own story structure, determine the appropriate interface elements to use to convey the story's action and manage the effective interaction of a user within the story as it plays out. Here we describe the architecture of the Mimesis system, which integrates a 3D graphical gaming environment with intelligent techniques for generating and controlling interaction with and within a narrative in order to create an engaging and coherent user experience.

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

Interactive narrative is story-telling for purposes of education, training, or entertainment in which a user interacts with a computer system to experience a story as an active participant. In systems that implement this type of interaction, the user is typically immersed in a 3D graphical environment in which a narrative – a structured sequence of events, often referred to as a story – unfolds [2, 5]. Unlike conventional narrative media such as the film or novel, the user takes on the role of a character in the unfolding story and is allowed (or encouraged) to perform actions that substantively change the world in which the story is being played out.

There are two important aspects of any interactive narrative system. The first is the ability to create descriptions of compelling and novel action sequences that will be used to drive the plot experienced by the system's user. In contrast to convention game titles, interactive narrative systems should create their storylines at run time, allowing customization that takes into account a user's ability, interests, previous experience and other contextual factors. Second, an interactive narrative system must be able to effectively manage the run-time dynamics of the user's interaction, balancing the user's control over the environment with the need to preserve the coherence of the unfolding storyline.

The Mimesis system defines an architecture for building intelligent interactive narrative worlds. The goal in developing the Mimesis system is to build a system capable of creating structured interaction within virtual worlds that achieve the same kind of cognitive and affective responses to interactive stories as that seen in the participants of conventional narrative media. The approach taken in the design of the Mimesis architecture is to exploit a well-founded, declarative model of action and intention, in combination with new computational models of narrative structure. In the next section we briefly discuss the Mimesis system architecture and how it can be used to generate and intelligently control the narrative process.

2. OVERVIEW OF THE MIMESIS ARCHITECTURE

The Mimesis system integrates a suite of intelligent control tools with Unreal Tournament (UT), a commercially available 3D graphical game engine. While UT is well-suited as an engine for building conventional 3D interactive game titles, the representation of the environments that it models does not match well with those typically used by AI researchers. Like most game engines, UT's internal representation is procedural – it does not utilize any formal or declarative model of the characters, setting or the actions of the stories that take place within it. Consequently, direct integration of intelligent software components is difficult. To facilitate this integration, Mimesis overrides UT's default mechanisms for controlling its virtual environment, using instead a client/server architecture in which low-level control of the game environment is performed by a customized version of the game engine (called the Mimesis Unreal Tournament Server) and high-level reasoning about narrative structure and user interaction is performed remotely by an intelligent control element (called the Mimesis Controller). The architecture is presented in figure 1 and described below.

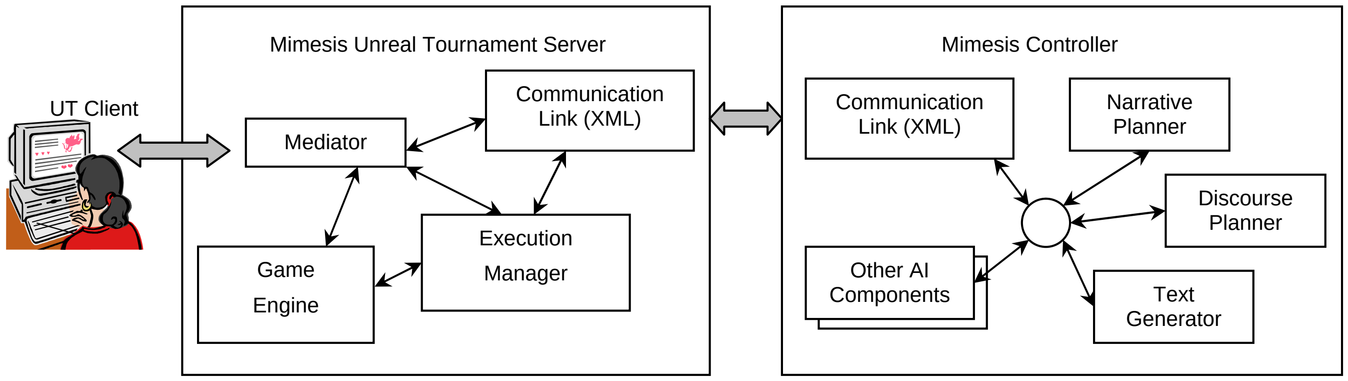

Figure 1. The Mimesis system architecture.

2.1 The Mimesis Controller

Within Mimesis, the Mimesis Controller (MC) acts essentially as a story server, determining the narrative elements of the user's experience within the virtual world. The MC is responsible for both the generation of a story (in the form of a plan characterizing all physical and communicative actions that are to be performed within the environment) and the maintenance of a coherent narrative experience in the face of unanticipated user activity.

The process of constructing a narrative experience involves a number of specialized functions, including reasoning about the actions of individual characters, generating any dialog used by the characters or narration to be provided by the system, creating cinematic camera control directives to convey the action that will unfold in the story, etc. To facilitate the integration of corresponding special-purpose reasoning components, the MC's architecture is highly modularized. Individual components within the MC run as distinct threads and communicate with one another via a well-defined message-passing protocol; developers extending the MC to provide new functionality wrap their code within a message-passing shell that requires only a minimal amount of customization.

While the MC architecture contains a number of intelligent components, two of these are central to the creation of the story's content. One, the narrative planner, is responsible both for determining all actions that will occur within the virtual environment as the story unfolds and for modifying the plan during the story's execution when the user's actions deviate substantially from the story's intended structure. The other central component, the discourse planner, is responsible for selecting the communicative techniques that will be used to convey the unfolding action to the user. We briefly describe these two components here. Additional components, such as a text generation system [3] (used to create custom dialog and narration), a user model and an HTTP server are also included in the MC; space limitations preclude discussion of the design of these elements.

Both the narrative and discourse planner components use the Longbow planning system, a hierarchical partial-order causal link planner that can produce plans both for physical actions [6] as well as communicative ones [7]. The narrative planner takes as input a declarative representation of all the actions that can be performed in the virtual world and a specification of the goals for the end of the story.

The narrative planner searches for a story plan – a sequence of actions to be carried out by the characters in the story (including the character controlled by the user) that will both satisfy the goals of the story and provide an engaging narrative arc. The discourse planner takes as input the story plan and a library of communicative actions that can be used by the game engine to convey the unfolding action of the story. These actions might include directives for a 3D camera controller [1], instructions for narrative voice-overs, the use of background music, etc. The discourse planner creates an action sequence containing directives to be carried out not by characters in the story world but by the game engine's interface resources and intended to be executed concurrently with the story plan itself.

2.2 The Mimesis Unreal Tournament Server

To execute the story and discourse plans, the MC communicates with an extended version of the UT game engine called the Mimesis Unreal Tournament Server (MUTS). Once the story and discourse plans have been created, the MC encodes the relevant plan structure into an XML formatted message and transmits the message to the MUTS via a socket connection. Each discrete, primitive action in the planning representations used by the MC is mirrored by a functional definition in the game engine that implements the action's semantics. Upon receiving the XML message, the Execution Manager, the element within MUTS responsible for driving the story's action, builds a DAG whose nodes represent individual actions in the plans and whose arcs define temporal constraints between actions' orderings. The Execution Manager uses one-to-one mappings from the action types of the nodes in this DAG to game engine functions and from the parameters of each action to instances of game engine objects in order to construct function calls that will drive the appropriate animations and state changes within the virtual world.

Complications to the plan execution process arise in two ways. The first is due to a mismatch in the characterization of actions used by Longbow to produce plans and the code used by the game engine to implement them. The story plan is a sequence of discrete actions whose effects happen instantaneously. In contrast, most of the corresponding function calls made by the game engine require a relatively long period of time to complete. This is especially true when actions must be coordinated with slower graphical character animations. To preserve the temporal semantics of the plan's structure, the Execution Manager launches sentinel threads for each action to be performed. These sentinel threads monitor the environment until the effects of the action have been achieved or until the sentinel determines that the action has failed.

The second complication dealt with by the MUTS during plan execution arises when the user performs actions within the story world that interfere with the structure of the story plan. Because each action that the user performs might potentially change the world state in a manner that would invalidate some as-yet-unexecuted portion of the narrative plan, the MUTS passes all action commands issued by the user to the Mediator prior to executing the corresponding action's code in the game engine. The Mediator monitors the user's action commands and signals an exception whenever the user attempts to perform an action that would change the world in a way that conflicts with the causal constraints of the story plan.

Exceptions are dealt with in one of two ways. The most straightforward is via intervention. Typically, the success or failure of an action within a virtual environment is determined by software that approximates the rules of the underlying story world (e.g., setting a nuclear reactor's control dial to a particular setting may cause the reactor to overload). However, when a user's action would violate one of the story plan's constraints, Mimesis can intervene, causing the action to fail to execute. In the reactor example, this might be achieved by surreptitiously substituting an alternate set of action effects for execution [4], one in which the "natural" outcome is consistent with the existing plan's constraints. A control dial momentarily jamming, for instance, will preserve the apparent consistency of the user's interaction while also maintaining safe energy levels in the story world's reactor system.

The second response to an exception is to adjust the narrative structure of the plan to accommodate the new activity of the user. The resolution of the conflict caused by the exception may involve only minor restructuring of the narrative, for instance, selecting a different but compatible location for an event when the user takes an unexpected turn down a new path. Accommodation may involve more substantive changes to the story plan, however, and these types of modifications can be computationally expensive. For instance, should a user stumble upon the key to a mystery early in a narrative or unintentionally destroy a device required to rescue a narrative's central character, considerable re-planning will be required on the part of the MC's narrative planner.

In order to handle exceptions in an interactive narrative system, the narrative planner analyzes its plans prior to execution, looking for points where enabled user actions (that is, actions whose preconditions for execution are satisfied at that point in the plan) can threaten its plan structure. When potential exceptions are identified, the planner weighs the computational cost of re-planning required by accommodation against the potential cost incurred when intervention breaks the user's sense of agency in the virtual world. Space does not allow a discussion of the interaction between the MUTS and the MC with respect to exception detection and handling.

3. SUMMARY

The Mimesis architecture is an approach for integrating structured interaction into traditional computer game engines, such as that of Unreal Tournament, in order to achieve the kind of cognitive responses to interactive stories as seen in the participants of conventional narrative media. Interactive narrative differs from conventional narrative in that the user is immersed in a 3D graphical environment and is encouraged to take on the role of a character in the unfolding story, requiring the interactive narrative system to be able to create descriptions of compelling and novel action sequences at run-time and to be able to effectively manage the run-time dynamics of the user's interaction. The Mimesis system bridges the gap between declarative representations of narrative and the procedural execution by the game engine and handles user exceptions in a way that is consistent with both the narrative and the user's expectations.

4. ACKNOWLEDGMENTS

The work of the Liquid Narrative group has been supported by National Science Foundation CAREER award 0092586 and by Microsoft Research's University Grants Program.

The Mimesis architecture has been designed and implemented by a number of students, including Mark P. Branly, David Christian, Milind Kulkarni, R. J. Martin, and C. J. Saretto.

5. REFERENCES

[1] Bares, W.H. & Lester, J.C. (1999). Intelligent multi-shot visualization interfaces for dynamic 3D worlds. In *Proceedings of the Third International Conference on Intelligent User Interfaces*, Los Angeles CA, 1999.

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

[3] Elhadad, M. (1993). *Using argumentation to control lexical choice: A functional unification based approach*. Ph.D. Thesis, Columbia University.

[4] Saretto, C.J. & Young, R.M. (2001). Mediation in mimesis liquid narrative. In *Proceedings of ACSME 2001*.

[5] 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.

[6] 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 Artificial Intelligence and Planning Systems*, Chicago IL, 1994.

[7] Young, R.M. (1999). Notes on the use of planning structures in the creation of interactive plot. In M. Mateas & P. Sengers (Eds.) *AAAI Fall Symposium on Narrative Intelligence*.