

Using Task-Based Modeling to Generate Scaffolding in Narrative-Guided Exploratory Learning Environments

James M. THOMAS and R. Michael YOUNG
Liquid Narrative Research Group
Department of Computer Science
North Carolina State University, Raleigh, NC USA

Abstract. A description of a novel domain-independent framework that automatically generates and fades scaffolding supports for task-oriented learning within exploratory environments. Both student knowledge and tutorial assistance is modeled through plan-based interactive narrative.

Keywords. Scaffolding, exploratory ITS, interactive narrative, planning.

Introduction

This paper describes a domain-independent framework for a new kind of tutor. This tutor specializes in task-oriented procedural learning within exploratory virtual environments like digital games. Student knowledge, learning goals, and tutorial assistance are all expressed in terms of how actions affect, and are enabled by the state of the virtual world. The most promising domains for this kind of tutor are those rich in inter-related tasks, (e.g., photosynthesis, RNA transcription, or arthroscopic surgery).

This framework is named “Annie”, in honor of Anne Sullivan, who taught the blind and deaf Helen Keller how to communicate through words. An open exploratory environment poses challenges that echo those faced by Anne in dealing with Helen’s restricted communicative bandwidth and high degree of autonomy. Similarly, Annie must intelligently cope with a highly uncertain model of the student’s understanding while continuing to gently guide the student through trial and error learning.

Annie is an extension of a system, called Zócalo [1], that generates interactive narrative through a planning-based knowledge representation. Annie uses plan construction techniques to build and dynamically adapt tutorial plans that guide learners’ actions in a virtual environment. This adaptation takes the form of automatic generation and fading of scaffolding tailored to the immediate knowledge requirements of the student.

The key contribution of this work is to outline an extensible, domain independent framework that automatically generates and fades scaffolding in a consistent manner across a variety of exploratory virtual environments and subtending knowledge domains. By abstracting the environment at the level of individual actions, Annie can work with a wide range of computer-mediated exploratory environments.

1. Related Work

Annie provides scaffolding for exploratory ITSs in the context of interactive narrative. Therefore, the two veins of research most closely related to Annie are the study of scaffolding in ITSs , and narrative centered learning environments.

1.1. Scaffolding in Exploratory ITSs

Although much of ITS research has benefitted from a shared consensus of intelligent scaffolding, [2,3], the group of systems variously described as exploratory, inquiry-based, or scientific discovery share less common ground [4,5]. Whereas much of the work in traditional ITSs exploits a strong thread of verbal discourse, the more exploratory systems have a broader variety of user interfaces [?], and thus fewer shared communicative constructs on which to ground a common approach to scaffolding.

Although Quintana et. al, [5] purport to describe a scaffolding framework for exploratory ITSs, it is not much more prescriptive to a system builder than a catalogue of design choices excerpted from existing systems. More recently, de Jong noted [6] the continuing lack of a general approach to balance instructional scaffolding and exploration “in such a way that learning is supported effectively, but the inquiry process is not reduced to following cookbook instructions.” Annie attempts to provide such a foundation.

1.2. Narrative-Centered Learning

The field of “interactive narrative” studies the automatic generation of stories within virtual worlds in which human users interact with one or more computer controlled agents, [7,8,9]. Mott found that adapting interactive narrative for exploratory learning demands that user autonomy be sufficient to support exploration of specific “hypothesis-generation-testing cycles” [10]. Annie is built on the Zócalo system [11], which builds on the Mimesis [12] approach of balancing narrative and user goals. Mimesis generates plans for actions of agents in a story world based on hierarchical task decompositions and discrete causal requirements. Annie leverages the Zócalo architecture to select the task decompositions that best move the learner toward their particular Zone of Proximal Development (ZPD) [13] at each point in execution of the learning narrative. Narrative provides a broad set of options for intervention, from lighting or sound changes that draw the user’s attention toward a learning opportunity [10], direct or indirect dialogue supplied by non-player controlled (NPC) characters, and omnipotent environmental control that can be used to dynamically adjust world geography, obstacles, and other challenges.

2. The Annie System Architecture

2.1. Overview

Annie uses a planning-based knowledge representation to underly the student model and drive the automated scaffolding, through libraries of diagnostic and remediative templates. This section describes that knowledge representation and describes how the plan

space created in Annie’s initialization can be mined for pedagogically useful information. Next, the full initialization sequence is described, followed by a stage by stage presentation of the execution cycle.

2.2. Knowledge Representation

Annie requires as input a planning-based library of the manipulable actions in the learning environment. The ITS author is free to instrument as many or as few of these actions as are deemed pedagogically relevant.

Given this planning library, Annie can derive conditions that describe the operators and can be used to articulate learning goals. Key among these conditions are “hasPrecondition(*operator*, *condition*)” and “hasEffect(*operator*, *condition*)”. For example, a **deleteFile** operator might be partly described by the following predicates:

- hasPrecondition(**deleteFile**, exists(?fileDescriptor))
- hasPrecondition(**deleteFile**, \neg inUse(?fileDescriptor))
- hasEffect(**deleteFile**, \neg exists(?fileDescriptor))

Annie’s student model contains estimates of the student’s strength of beliefs about each operator in the Zócalo-mediated virtual world (*ZWorld*). As the student makes action choices, Annie updates the strength of beliefs in the student model as indicated through a library of diagnostic templates.

2.2.1. Diagnostic Templates

Figure 1 shows an action attempt **C** that fails due a precondition p that is negated between the time it was established (action **A**) and the time the student attempts **C**.

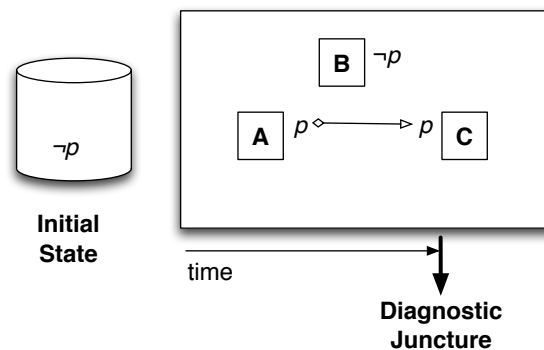


Figure 1. Action Failure - Unresolved Threat

One or more of the diagnoses detailed in Table 1 may be indicated as misconceptions which contributed to the failure. Table 1 is merely a partial specification of just a few of the many domain-independent planning scenarios encoded in the diagnostic template library.

Bug Source	ID	Nickname	Propositional Description
Operator Schema	2A	Unknown Effect	$\neg \text{Bel}(\text{has-effect}(\mathbf{B}, \neg p))$
Plan History	2B	Sequencing Error	$\text{Bel}(\text{time}(\mathbf{A}) \succ \text{time}(\mathbf{B})) \wedge \neg(\text{time}(\mathbf{A}) \succ \text{time}(\mathbf{B}))$
Plan History	2C	Unobserved Threat	$\text{Bel}(\text{time}(\mathbf{B}) \succ \text{time}(\mathbf{C})) \wedge \neg(\text{time}(\mathbf{B}) \succ \text{time}(\mathbf{C}))$
Plan History	2D	Phantom Intervention (action X has not happened)	$\text{Bel}(\text{has-effect}(\mathbf{X}, p)) \wedge (\text{has-effect}(\mathbf{X}, p)) \wedge \text{Bel}(\text{time}(\mathbf{X}) \prec \text{time}(\mathbf{C})) \wedge \text{Bel}(\text{time}(\mathbf{B}) \prec \text{time}(\mathbf{X})) \wedge \neg(\text{time}(\mathbf{X}) \prec \text{time}(\mathbf{C}))$
Operator Schema	1A	Unknown Precondition	$\neg \text{Bel}(\text{has-precondition}(\mathbf{C}, p))$

Table 1. Unresolved Threat - Possible Diagnoses

2.2.2. Remediation Templates

Assume that Annie has nominated the following knowledge gap for remediation:

hasPrecondition(deleteFile, \neg inUse(?fileDescriptor))

This means that Annie wants to increase the student’s strength of belief that the **delete-file** operator has a precondition that the file cannot be *inUse*. Annie chooses a remediation template from the library to apply to the plan. The following example shows one possible instantiation of a **demonstrate** remediation template:

1. *show(inUse(file1, app1))*. Show that file1 is inUse by app1.
2. *deleteFile(Ann, file1)*, An NPC (named Ann) attempts to *deleteFile* on file1.
3. *closeApplication(Ann, app1)*, where the app1 is the current user of file1.
4. *show(\neg inUse(file1, app1))*. Show that file1 is not inUse by app1.
5. *Ann, deleteFile(file1)*. Ann successfully deletes file1.

This fairly complex example was chosen to highlight that the remediation templates are not simple atomic activations, but partially-specified plan structures that Annie must dynamically weave into the particular state of the particular session to achieve defined pedagogical goals. The template is “partially-specified” in that it contains placeholders that allow Annie to find the right combination of operators and ground terms to bring about the intended changes.

2.3. Plan Space Exploration

As the student progresses, Annie gains more and more information about the state of the student’s knowledge, but has less and less time remaining to act on these inferences. Annie leverages the richness of the plan space to estimate the student’s proximity to world goals. This is used to help determine the urgency with which guidance should be provided.

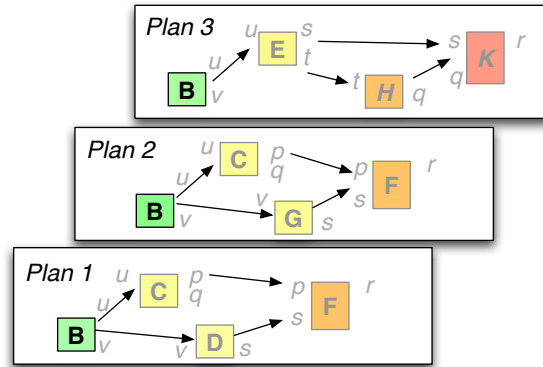


Figure 2. Alternative Plans From the Plan Space

For example, figure 2 depicts a situation where action B has completed successfully. The upper case letters represent actions and lower case letters in the figure represent preconditions and effects. The most proximal actions to consider are those shaded yellow {C, D, E, and G}. Any of those actions could be executed next, as their preconditions are currently satisfied, i.e., zero intermediate steps are required before they can execute. This group of actions can be called “Tier 0”. The next most proximal tier of actions (Tier 1 = {H, F}: shaded orange) are those whose preconditions can be satisfied by a causal chain of length one. Followed by action K is in tier 2. Thus, we have a numerical method to rank goal proximity across the plan space and guide diagnosis.

2.4. System Inputs

Annie is a data-driven system. External libraries of domain-independent plan-based diagnostic and remediation templates encode all the particulars of Annie’s scaffolding. At initialization, these libraries are applied to the session-specific Learning Problem Description (*LPD*), which defines a set of domain-specific learning goals for Annie to achieve in that particular session. The first component of the *LPD* is a *ZLib*, which describes the plan operators, objects, conditions, initial and goal states of the *ZWorld*, without reference to any learning states or goals. The *ZLib* is written in a restricted form of the STRIPS-like language used for all interactive narratives in *Zócalo*.¹

Because the *ZLib* necessarily contains a planning problem description, it would be possible to embed pedagogical goals into particular operators and states of objects in the world to create a tutorial that does not exercise any of Annie’s student modeling. Indeed, most commercial games implement teaching by creating actions for the student to perform that signify to the system that the student knows some particular thing. However, one of the prime motivations for Annie is that it is often not enough for the student to simply “do” some action. It is possible for a student to perform an action without fully understanding it, either because the action was a “lucky” guess, or the student was distracted by other game elements when the action was performed. For these situations, the second component of the *LPD* allows the author to specify learning goals as a set

¹The plan-based interactive narrative representation used by *Zócalo* is described more fully in [?]

of explicit meta-knowledge of the operators in the *ZLib*, using derived predicates such as “hasPrecondition(*operator*, condition)” and “hasEffect(*operator*, condition)”. These predicates also form the basis of the student model,

At initialization, Annie uses the external template libraries, the *ZLib*, and the rest of the *LPD* to compile its knowledge base, called *ABase*. The *ABase* incorporates the entire *ZLib*, and expands it with pedagogically-focused elaborations. Thus, Annie represents the student’s state of belief for each precondition and effect of each operator in the *ZLib*. Included in this construction phase is the compilation of all the remedial and diagnostic templates to represent beliefs specific to the individual operators of the domain.

The final step of Annie’s initialization is creating a tutorial plan consisting of a plausible partially-ordered sequence of student and system-initiated actions that is guaranteed to bring about a specific goal state for the world. Annie uses this plan to initiate the tutorial session within the *ZWorld*. Because the planner is only concerned with the states of the world, the tutorial plan does not *guarantee* that the goal beliefs for the student will be achieved. Rather, Annie monitors student behavior and optimizes the frequency and extent of its tutorial interventions to increase the likelihood that the goal beliefs are acquired.

2.5. Execution Cycle: Overview

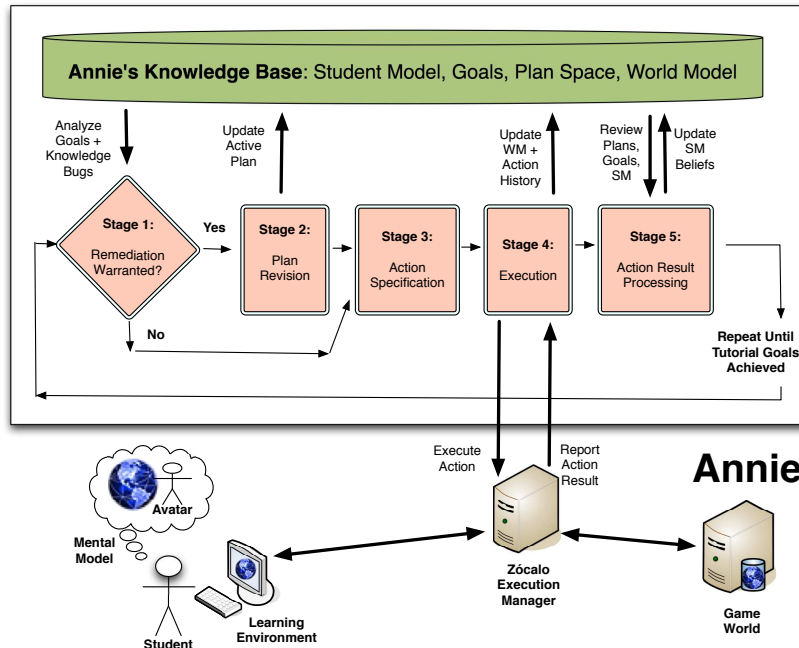


Figure 3. Execution Data Flows

Annie’s run-time behavior consists of a loop that iterates each time the student or the system executes an action in the world. As van Lehn observed, many seemingly dissimilar tutoring systems [2] employ cyclic execution models. In fact, most of these can

be characterized as a pair of nested loops, where the outer loop iterates over “tasks” and the inner loop iterates over each *step* in a task. Because Annie’s plans contain hierarchical decompositions, its single loop sometimes iterates over tasks and sometimes over steps in a task. As depicted in Figure 3, Annie’s execution cycle is divided into five *stages*.

2.5.1. Stage 1: Remediation Consideration

First, Annie reviews the student model for unrealized pedagogical goals. It then compares the beliefs about upcoming tasks in the tutorial plan space to identify knowledge gaps that may hinder the student’s immediate progress. Urgency thresholds for repairing these knowledge gaps are derived based on the proportion of steps remaining in the possible successful plans. If the threshold is met, the most critical gap is chosen for remediation in Stage 2. Otherwise, if no candidate remediation is deemed sufficiently urgent, the plan is left unchanged and Annie proceeds directly to Stage 3.

2.5.2. Stage 2: Plan Revision

Annie consults a library of domain-independent plan templates to choose a “remediation” for the knowledge gap indicated in Stage 1. These templates incorporate narrative-based analogues of common ITS interventions including **prompt**, **hint**, **demonstrate**, **teach**, **do**. Beyond these, the library contains more complex, multi-step sub-plans particularly particularly attuned to common domain-independent misconceptions, as well as hand-authored sub-plans tailored to a particular domain. At this point, Annie may consider switching from the currently active tutorial plan in favor of another potentially successful plan in the plan space, if it offers a better fit for the current plan history and proposed remediation. For example, if the student’s actions show a pattern of increasing divergence from the current plan, coupled with increasing similarity to an alternative plan, Annie defers to the student’s initiative and switches to the plan that seems to better match what the student is doing. Otherwise, if a remedy is selected, a sub-plan particular to the current plan context is generated, and the existing tutorial plan is repaired to include this new sub-plan.

2.5.3. Stage 3: Action Selection

In this stage, Annie selects the action of the current plan to be performed on this iteration through the execution loop. Because Annie’s plans are partially ordered, there may be more than one potential action that could be executed next, and these actions may be either student or system-initiated. Annie selects the next action based on its projected pedagogical utility.

2.5.4. Stage 4: Execution

First, Annie checks its message buffer to see if it has been notified of any student actions since the last iteration. If not, Annie sends a message to the execution manager telling it the next action to be executed. The execution manager acknowledges this message and the loop continues. It might seem reasonable that in the case of student-initiated actions, Annie should wait for an acknowledgement that the student has in fact taken the action. Instead, we allow those messages to arrive asynchronously and buffer them. If, at the beginning of this stage, such a message is in the buffer, Annie discards the action that had been selected and simply carries the student-initiated action result message into stage 5.

2.5.5. Stage 5: Action Result Processing

The plan history, current state of the *ZWorld* and the student model are updated based on the action result. An extensive catalogue of action result scenarios guides Annie in deciding how to update each component of the student model.

3. Conclusions

This paper described a novel framework, named Annie, that automatically generates and fades scaffolding supports for task-oriented learning within exploratory environments. Annie builds on a plan-based approach to the generation of interactive narrative in order to provide narrative-based student guidance. A key contribution of this work is that it provides an extensible, domain-independent approach to guidance in exploratory ITSs. A test domain of a virtualized operating system vulnerable to various malware infections has been created as a “mod” using the commercial game Unreal Tournament 3. This game “mod” will be used to empirically evaluate of Annie’s teaching effectiveness in the domain of malware infection removal.

References

- [1] T.M. Vernieri. A Web Services Approach to Generating and Using Plans in Configurable Execution Environments. Master’s thesis, North Carolina State University, Raleigh, North Carolina, January 2006.
- [2] K. VanLehn. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.
- [3] S. Puntambekar and R. Hubscher. Tools for Scaffolding Students in a Complex Learning Environment: What Have We Gained and What Have We Missed? *Educational Psychologist*, 40(1):1–12, 2005.
- [4] T. de Jong and W.R. van Joolingen. Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research*, 68(2):179–201, 1998.
- [5] C. Quintana, B.J. Reiser, E.A. Davis, J. Krajcik, E. Fretz, R.G. Duncan, E. Kyza, D. Edelson, and E. Soloway. A Scaffolding Design Framework for Software to Support Science Inquiry. *The Journal of the Learning Sciences*, 13(3):337–386, 2004.
- [6] Ruth Aylett, Sandy Louchart, Joao Dias, Ana Paiva, Marco Vala, Sarah Woods, and Lynne Hall. Unscripted narrative for affectively driven characters. *IEEE Comput. Graph. Appl.*, 26(3):42–52, 2006.
- [7] T. de Jong. Technological advances in inquiry learning. *Science(Washington, D. C.)*, 312(5773):532–533, 2006.
- [8] M. Mateas and A. Stern. Structuring Content in the Façade Interactive Drama Architecture. *AIIDE*, pages 93–98, 2005.
- [9] B. Magerko, J.E. Laird, M. Assanie, A. Kerfoot, and D. Stokes. AI Characters and Directors for Interactive Computer Games. *Proceedings of the 2004 Innovative Applications of Artificial Intelligence Conference*, 1001:48109–2110, 2004.
- [10] W. Swartout et al. Toward the Holodeck: Integrating Graphics, Sound, Character and Story. *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 409–416, 2001.
- [11] B.W. Mott and J.C. Lester. U-director: a decision-theoretic narrative planning architecture for storytelling environments. *AAMAS*, pages 977–984, 2006.
- [12] Thomas Vernieri and R. Michael Young. Web services for interactive narrative. In *Second GAMEON North America Conference*, pages 13–17, Monterey, CA, USA, September 2006.
- [13] Mark Riedl, C.J. Saretto, and R. Michael Young. Managing interaction between users and agents in a multi-agent storytelling environment. In *AAMAS*, 2003.
- [14] T. Murray and I. Arroyo. Towards Measuring and Maintaining the Zone of Proximal Development in Adaptive Instructional Systems. *ITS*, 2002.