Interactive Story Generation for Writers: Lessons Learned from the Wide Ruled Authoring Tool

James Skorupski
University of California, Santa Cruz
1156 High Street
Santa Cruz, CA 95064
jskorups@soe.ucsc.edu

Michael Mateas
University of California, Santa Cruz
1156 High Street
Santa Cruz, CA 95064
michaelm@soe.ucsc.edu

ABSTRACT
The authoring of interactive, generative narrative is a task that typically requires an extensive multi-disciplinary background in computational and narrative theory. Wide Ruled is an authoring tool that aims to address this problem by providing a friendly, intuitive, story-centric interface to an author-goal driven text-based story planner. Over the past two years, this system has been used repeatedly by technical and non-technical users in multiple classroom settings, and evolved into a widely used and publically available story authoring system. In this work, we describe the successes and failures of Wide Ruled, and how it provides a critical evolutionary step in developing a truly usable, writer-friendly, and practical interactive story authoring environment.

1. INTRODUCTION
As interactive entertainment becomes a more pervasive element of our culture, the potential for meaningful narrative experiences will only be realized if we can create tools that open their authoring to a much wider audience. Typically, creating the generative stories that drive these experiences requires technical expertise in computational models of story planning and structure, as well as the knowledge to formulate compelling plot arcs, rich dialog, character conflicts, and other story elements. Unfortunately, the existence of these cross-disciplinary experts with the relevant technical and creative backgrounds is a rare occurrence. Wide Ruled is a freely and publically available[1] story authoring tool that attempts to reduce this required technical expertise and bridge the divide between algorithms and art by providing a non-technical, writer-oriented authoring interface to a text-based interactive story generator. It is based on an existing author-goal driven model of story generation, called UNIVERSE, which models story structure as a set of hierarchical plans that encompass one or more ways to accomplish a story goal for the author. Wide Ruled uses non-technical narrative terminology wherever possible, natural-language descriptions of plan preconditions and actions, and step-by-step guidance in building complex constraints and modifications of the story world [13]. Ultimately, it aims to provide an authoring environment that provokes a feeling of familiarity and relevance to the task of story-telling, and at the same time maintains the flexibility and power of the underlying planning engine.

The development, deployment, and evolution of Wide Ruled over the past two years as a practical tool for story writers has provided abundant and essential feedback on the experiences of student authors of varying technical and non-technical backgrounds. Users both comfortable and unfamiliar with the mathematical and programmatic concepts embodied in the underlying technology have used the tool to create individual generative stories, and the feedback from students, teaching experience of the authors, and the resulting stories made with the tool have in turn driven the continued evolution of the system and motivated further work in story authoring research. This information has given us a unique view of the strengths and weaknesses of Wide Ruled as a usable piece of software for story authoring.

In this paper, we will review the story generation model and capabilities of Wide Ruled and its evolution over the course of its two major versions, and then compare our expectations of its use as an authoring tool to its actual use in classroom settings. We will discuss both quantitative metrics of user-created stories as well as the qualitative experiences of both the students that used the system and the authors that helped these students understand it. Finally, we describe our future work on story authoring tools informed by the lessons learned from Wide Ruled.

2. RELATED WORK
The complexities of authoring interactive stories have been discussed in detail in previous work. Many story authoring tools are built around the creation of story graphs, which require an author to statically represent every potential story path [1, 6, 11, 12, 15]. While these storygraph systems utilize a readily understandable and visualizable model of the space of potential stories, they lack the power of a generative formalism such as our own model, and require the explicit creation of a combinatoric space of story fragments, ultimately limiting the size and variability of the interactive story space to that which can be reasonably specified manually by the author. Wide Ruled, on the other hand, utilizes a plan-based approach that allows reuse of author goals in various points in the story space, as well as dynamic binding of variables in story plans, resulting in varying output depending on the state of the story world during execution. Existing research has explored other plan-like representations of stories that differ from author-goal driven style present in Wide Ruled [4, 14].

Wide Ruled is based on the UNIVERSE author-goal based story generation model developed by Lebowitz [7, 8] (described later in this paper) and is unique in that it is the only deployed and

[1] Wide Ruled is available for download from http://eis.ucsc.edu

Digital Arts and Culture, December 12–15, 2009, Irvine, California, USA.

publically available story generator based on that model, and also among a relatively small population of interactive story authoring systems that have been distributed and widely utilized as a practical tool, beyond the typical bare research implementation useful only as a proof-of-concept. Previous story authoring systems used and evaluated in real-world settings include the hypertext-based StorySpace system [2], and the virtual world authoring tool Bowyer, an extension of the Bowman mixed initiative narrative planning system [3, 14]. StorySpace has proven to be a popular, successful and practical authoring tool, but represents stories in a static graph-like manner that lacks the generative formalisms of our story model. Bowyer, on the other hand, is used to construct graphical virtual worlds and is based on a powerful hierarchical task network planning architecture (similar to our own hierarchical story model), but its evaluation focused on constructing and executing highly technical plan domain specifications and was aimed at planning researchers, instead of non-technical story authors. Our work, in contrast, exposes non-technical as well as technical users to the power of plan-based story generation, and evaluates our system as a complete interactive story creation environment.

3. THE WIDE RULED STORY MODEL

The story generation model in this tool is based on the HTN-style UNIVERSE model of story planning [7, 8] which models story structure as a set of hierarchical plans that encompass one or more ways to accomplish a story goal for the author. This style of story execution was chosen due to its success with students in previous Interactive Narrative classes [13]. A Wide Ruled story contains a set of author-created story objects, represented as "Characters" or "Environments", each with associated attribute-value trait pairs, and relationships to other characters or environments, respectively. The story also contains a set of "Plot Point Types", which define categories of episodic attribute-value data generated and utilized only during the story generation process. "Author Goals", with optional parameter variable inputs, are the primary unit of story planning in this tool, each one relating to one or more "Plot Fragments" that describes a set of actions that fulfill its parent author goal. Plot fragments have ordered precondition constraints that must all be true before execution. These constraints rely on the current state of the story world during generation and can also bind story objects and their attributes to local variables for later use in that same Plot Fragment. Additionally, plot fragments contain a list of sequential "Story Actions" that can modify the story world during execution. These actions can modify story characters, environments, and plot point instances bound in the precondition, create and delete instances of plot point types, calculate new values, print out story text parameterized by bound variables (the actual textual output of the story seen by a reader), and pursue other author goals. A complete description of story actions is detailed in our previous work [13].

Story generation in Wide Ruled begins with a top-level initial author goal, which randomly selects amongst all executable plot fragments with valid preconditions, and then sequentially executes all of its contained story actions to successfully complete a story. If the generator encounters a story action that pursues another author goal, the process repeats and a new random plot fragment is executed. The relationship between author goals and their associated plot fragments describes a potential tree-like space of stories, in which a single generated story is represented as a traversal of the tree, as seen in Figure 1. The original UNIVERSE model was not interactive, so we implemented story interactivity in two different ways for each version of Wide Ruled, one method based on plot fragment selection intervention, and the other based on asynchronous goal execution. Both of these interactivity models are described in the later sections of this paper.

The output of a Wide Ruled story is a string of sequential text generated by any plot fragments that display text as part of their story action list. The variation in each textual story derives from the parameterized nature of each of these blocks of outputted text. Information from characters, environments, and plot points in the story world are captured and stored within preconditions inside each plot fragment, and then modified and printed to the screen in these text blocks. A combination of static hand-written text, interspersed with that dynamically bound variable information, results in story variations dependent on the particular story being generated. This resulting text is displayed in a reading interface that prints the text in real time as it is generated by the system.

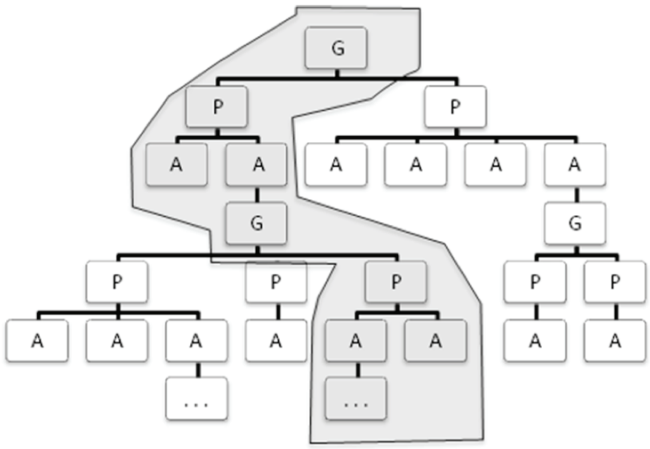

Figure 1. The Wide Ruled Story Hierarchy. "G" nodes represent author goals, "P" nodes represent possibly selected plot fragments, "A" nodes represent sequentially-executed story actions, and the gray overlay represents a single instance of a story, executed top-down, within this potential story space.

4. AUTHORING IN WIDE RULED

The story generation model behind Wide Ruled, and the interface itself, supposes specific techniques for creating an interactive narrative. In this section we will describe the key authoring techniques supported by the system and taught to the students that used the system in the classroom, and provide examples of the techniques in the context of the sample murder mystery story world that is provided to these students.

4.1 Characters and Environments

Characters and environments in Wide Ruled are objects that contain a shared list of "traits", or attribute-value pairs, and "relationships" to other characters or environments, respectively. They hold character or environment-specific numeric, true/false, and textual information, and the common traits and relationships can be added, renamed, and deleted by users. In the murder mystery example story provided to students, characters hold traits such as their name (text), age (number), whether they can

potentially be a victim (true/false), whether they are a detective (true/false), their personal description (text), and other traits. Each murder mystery character also has a set of relationships, including "friend", "enemy", and "co-worker", that connect each character to other characters in the story world. Similarly, the environments in the murder mystery contain traits such as location descriptions (text), the time of day (text), and relationships that include hiding locations for mystery clues in each place, and secret hideouts for potential murderers that are near these environments. In general, characters and environments, along with plot points described in the next section, are designed to hold core pieces of story information and in turn be manipulated by actions within plot fragments..

4.2 Plot Points

Plot points are story objects that contain a list of traits, similar to characters and environments, but no relationship information. They are intended to be used as an information store for important episodic story information that exists only while a single story instance is being generated. For example, in the murder mystery story world, multiple plot points are used to store various murderer, victim, and detective information, which are all key components of the plot of each generated story. The "murderer" plot point, specifically, is used to keep track of the name of the character that is chosen to be the murderer (text), and the name of their hideout locations (text). Because plot points are meant to be a temporary episodic story memory, during authoring, Wide Ruled lets the user specific the type of traits (numeric, true/false, or textual) that are contained within each plot point type, but not the content of any specific plot point instance (which is determined at generation time), in contrast to characters and environments. Any initial story information stored before generation is intended to be stored within character or environment traits or relationships.

4.3 Story Structure

The structure of a Wide Ruled story, as previously mentioned, is contained within the hierarchical arrangement of author goals and plot fragments specified by an author. Author goals are designed to be high level intentions of the author. In the murder mystery story world, these include "Select a victim", "Select a Murderer", "Select a Detective", "Kill the victim", and "Investigate clues". In our murder mystery, we want a static set of author goals executed in order to provide a high level structure of our story. In this story world, we want the following list of author goals to occur sequentially as subgoals within the top level "Create Murder Mystery" author goal: (1)"Select a Victim" – choose a victim from the list of characters, (2)"Select a Murderer" – choose a murderer, who isn't the victim, from the list of characters, (3) "Murder Victim" – have the selected murderer kill the victim in some way, (4) "Select a detective" – choose a detective, who isn't the murderer or victim in this particular story, to investigate the case, (5) "Investigate Murder" – have detective investigate crime scene in some way and interview some people, following any clues he or she finds, and (6) "Solve case" – have detective put clues together, identify murderer, and catch said murderer. As seen in these specific examples, these author goals are very general, and meant to convey high level author intention for the progression of the story plot. Plot fragments, however, are intended to be specific ways a story can enact an author goal. The "Select a Murderer" author goal has two plot fragments, "Random murderer" and "Enemy of the victim", as two different ways of choosing the murderer for an instance of the story. During execution, only one of these options will be chosen, depending on the state of the story world at the time. Ideally, a higher quality story world contains more variation in each story instance, and therefore many plot fragments. The depth and breadth of this story tree should be high, where author goals have many ways of being completed by various plot fragments, and each plot fragment has complex substructure in the form of numerous subgoals. The story structure for the sample murder mystery is displayed in Figure 2.

Author goals contain a list of plot fragments, and a list of parameters. Parameters are pieces of textual, numeric, or true/false information that can be given to an author goal, and passed along to a selected plot fragment during story generation. Similar to parameters to functions in traditional programming languages, they are intended to store and pass along temporary story information that is useful for a specific author goal, but may not necessarily warrant persistent storage in a more global manner, such as within characters, environments, or plot points. In our sample story, the name of the victim's friend, only used once in the story, is passed into a subgoal in order to be included in text describing the investigation of the crime through interviews with family and friends.

4.3.1 Plot Fragment Preconditions

During generation of a story in Wide Ruled, the precondition for each plot fragment is a list of requirements that determines whether it is eligible to be selected as a possible way to complete an author goal. These requirements are a list of constraints on the traits and relationships within characters, environments, and plot points, and every constraint must be true simultaneously for the plot fragment to be valid and ready to use within a story. Preconditions can capture individual traits, relationships, and entire characters or environments, and then perform story actions on that information. Particularly complex precondition statements contain a series of statements that capture, or bind, information to named variables, and then use them within another part of a precondition. For example, in the murder mystery story world, the plot fragment named "Enemy of the victim" first binds the name of the victim, and then uses this information to further find the name of the victim's enemy, using the enemy relationship attribute. This requires two separate constraints, in the following order:

1. `There exists a Victim Plot Point, where trait "victim name" is saved as variable "victimName".`
2. `There exists a Character, where trait "name" = victimName, and relationship "enemy" target name is saved as "enemyName"`

The first constraint locates the plot point that is currently storing the name of the selected victim found earlier in the generation of the story and saves it, while the second constraint uses that victim name to find the victim character and capture the name of that victim's enemy. While not described here, preconditions can also take the form "There does not exist …" which allows an author to require that a story object with certain constraints does NOT exist in the story world. The natural-language description of these constraints is discussed in section 5. In addition to storing traits and relationships, a precondition can save an entire character,

environment, or plot point to a named variable. Parameters from author goals can be used within the precondition constraints, and any named variable created within a constraint can also be used in story actions, described in the next section.

4.3.2 Plot Fragment Story Actions

Story actions are sequentially performed steps that occur when a plot fragment is valid and selected during story generation. The primary story action, responsible for the narrative output of Wide Ruled, is the parameterized text output action. This step is a block of author-specified text, which can be filled with named variables captured in the precondition, or generated by other actions performed before it. During story generation, these variables are bound to numeric, true/false, or textual information from the story world and printed to the screen. For example, in the following snippet of parameterized text from the plot fragment called "Reveal Murderer" in the murder mystery story world, the names between the brackets are relevant story variables that will be filled in at generation time:

```
<enemyName> appears out of the darkness as the
evil murderer, and escapes into the distance to
seek shelter at <hideoutName> from the inevitable
eyes of the police.
```

At story generation time, this text can, depending on the selected enemy and crime location, become one of many variations, such as the following (italics added for emphasis):

```
Gene Franks appears out of the darkness as the
evil murderer, and escapes into the distance to
seek shelter at the abandoned warehouse from the
inevitable eyes of the police.
```

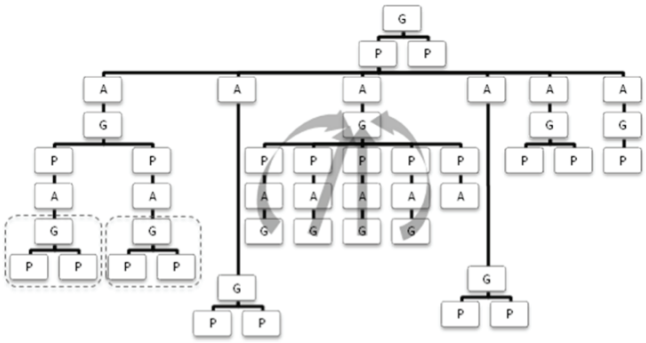

Figure 2. The sample murder mystery story structure provided to students with Wide Ruled 2. Non-subgoal actions are omitted in this diagram for clarity. Arrows represent recursive subgoal story actions, resulting in author goal repetition within the generated stories. The goals surrounded by a dotted border indicate the same author goal being reused multiple times in the story hierarchy.

In addition to parameterized text output, story actions are responsible for modifying the story world, and can therefore insert numbers, true/false values, text, and other named variables into saved characters, environments, and plot points. Here, plot points can be created and deleted as they are needed during story generation. Calculations, where named variables or newly created named variables can add, subtract, multiply, and divide numbers or other named variables, can also be performed as a story action, and saved for later use. Finally, subgoal actions are an essential story action within Wide Ruled. This type of step allows an author to select an author goal to execute, and pass along any desired parameters if so desired. When that subgoal is finished being explored, the story generator will return to the current plot fragment and continue enacting any remaining story actions. This subgoaling power allows for very complex story worlds to be represented hierarchically, and allow each plot fragment to simply point to high level goals, which are authored separately and can be reused in multiple plot fragments elsewhere. Figure 2 shows how subgoals can result in complex story world arrangements. Specifically, the two similar subtrees of the hierarchy on the left side of that story world represent two different plot fragments with different textual output and story actions, that, within them, subgoal the same author goal (highlighted by dotted borders). In addition, the center portion of the tree shows four plot fragments that recursively call a previous author goal, resulting in a looping story construct during generation.

4.4 Common Authoring Techniques

Throughout the development and classroom usage time of Wide Ruled, the authors have utilized some repeated techniques to create story worlds in the system, and this section describes these techniques and their utility in this story generation model.

4.4.1 Top- down story design

The author goal based story model in Wide Ruled encourages a top-down specification of the story during its creation. Because author goals are intended to be high level, self-contained story intentions, they can be created and used before they are fully fleshed out with completed plot fragments. This methodology is familiar to those with a software programming background and especially those with experience in object-oriented design, in which the structure of a computer program can be defined, and is often encouraged to be defined, before its function is ever fully implemented. In the context of Wide Ruled, this involves the creation of author goals and associated plot fragments containing empty preconditions and only subgoal actions. This allows the author to flesh out the high level, or author-goal level structure of a story, before determining how these goals will be accomplished. As the author moves from the highest to the lowest levels in the story hierarchy, he or she must begin to consider the specifics of the story, because these subgoals, by definition, become more specific to the context in which they reside. As a story is fleshed out with preconditions and story actions that print text, calculate, and modify characters, environments, and plot points, these subgoals can stay in place and provide the supporting story skeleton for the creation of the details of a rich dynamic narrative.

4.4.2 Looping

A common construct within Wide Ruled stories is a repeated author goal. Because there is no explicit way to execute a story action multiple times, this story generation model requires recursion to perform repetition. As seen in Figure 2, the murder mystery implements looping using this method. The murder mystery performs an "investigate clues" author goal multiple times before continuing to solve the crime and finishing the story. In order to prevent infinitely deep recursion, an author can specify a plot fragment, which can be randomly selected, that stops this repetition, or, as in the case of the murder mystery story, a plot point can be used to store a counter, which is saved and incremented by a calculations story action, and used to determine

when the looping action is complete. Note that repetition need not be limited to occur within a single step down the story hierarchy, as seen in the example story depicted in Figure 2. One can imagine a loop in which many subgoals occur before a previously used author goal is activated by a plot fragment.

4.4.3 Debugging

Because the creation of dynamic interactive narrative involves generative constructs and often complex constraints, the creation of a Wide Ruled story is not without its own potential problems. When a story world is unruly and acting in unexpected ways, two ways of debugging a story world have been found useful. Inserting parameterized text throughout the story hierarchy that prints out named variables is a tried and true method that finds its roots in the "printf" software debugging methods of the programming world. In addition, inserting an always-will-be-false constraint into a plot fragment allows an author to temporarily disable an entire region of potential story space. The Wide Ruled tool itself provides a verbose output mode that prints the goal and plot fragments that are being executed at any given time, which, when combined with the previous two methods, has allowed authors to track down most story world bugs with relative ease.

5. WIDE RULED 1

Wide Ruled 1, developed and evaluated in 2007, provided a familiar graphical interface to the underlying UNIVERSE-style story planner, utilizing standard interface conventions, including OK/Cancel window actions, editable tables of attribute-value pairs, clickable item lists, and hierarchical and collapsible tree lists. In addition, the interface contained story-centric terminology for each component of the interface, avoiding technical terms where possible to avoid confusion for those without a technical background. This became troublesome when attempting to describe the binding and ordering of variables, passing of parameters, and complex precondition constraints, which are inherently technical in nature. A complete documentation of all the features in Wide Ruled 1 is described in detail in our earlier work [13].

Due to the complexity of creating precondition constraints, Wide Ruled 1 provided a wizard-based interface to create these potentially complex statements [13]. Similarly, wizards were used to generate story actions, which often required referencing a bound variable for modification or printing. These wizards proved to be useful when learning how to create constraints, but cumbersome and repetitive when creating many repeatedly.

In order to simplify complex precondition constraints and story actions, these elements are displayed in the plot fragment editor as natural-language statements. For example, consider this complex constraint in a plot fragment precondition:

```
There exists a character, saved as "myChar", where
trait "alive" = true, and relationship "enemy"
target name is saved as variable "enemyName", and
trait "name" is saved as variable "charName".
```

In this case, any character that is matched must have a value of "true" for trait "alive", the name of the character is bound to the variable "charName", and the name of the enemy of that character is saved to the variable "enemyName". In addition, a reference to the matched character, "myChar" is also saved for later editing in the plot fragment's story actions. The following text is an example of a story action, described in natural language that modifies the "name" trait of the bound character "myChar":

```
Edit Character "myChar": set trait "name" to
"John"
```

These descriptions help users to quickly understand what a plot fragment does without having to decode cryptic symbols.

As mentioned earlier, we implemented a story interactivity model on top of the traditionally non-interactive UNIVERSE-style generator used in this version. The reader could intervene in the story planning process by selecting among possible plot fragments for some author goals. Authors could select a single character as the "active character" in each plot fragment, and a reader would select a character when reading, and select amongst the possible plot fragments available for that particular character, if any were available. This method of interaction proved troublesome and non-intuitive because it attempted to shoehorn the concept of an "active character" for a decidedly non-character-centric story planner [13].

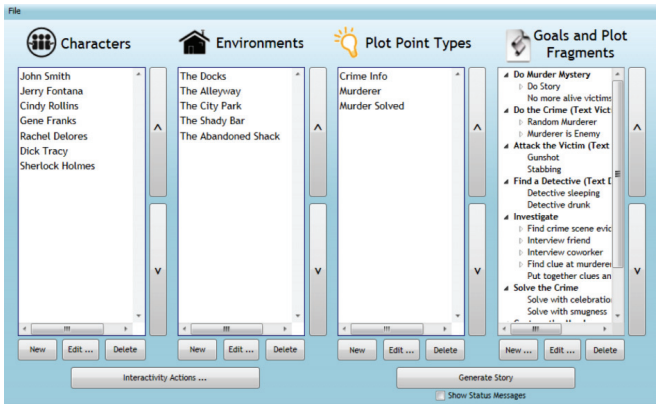

Figure 3. The Wide Ruled 2 Main Window. Here, the characters, environments, plot point types, and story hierarchy are displayed to the author.

6. WIDE RULED 2

Wide Ruled 2 addresses many of the shortcomings of Wide Rule 1, which were made apparent with the feedback received from the initial user evaluations in our previous work, as well as through the continued use of the tool by its authors. It builds off of the same graphical interface style of the original version with a series of improvements described below, and is the publically available version that is currently used in classroom settings and is actively supported. Figure 3 shows the main screen of Wide Ruled 2.

The underlying story generation model of Wide Ruled was modified in two ways for this latest version, in order to simplify the generation model and provide a more intuitive authoring and reading experience. The interactivity model of the previous iteration proved to be unnatural, as described above. To address this, Wide Ruled 2 implements an asynchronous goal execution model, in which story authors specify a set of "Interactive Actions" that a reader can execute at any time during story generation. These actions are separate author goals, which can modify the characters, environments, or plot points in the story and output parameterized text to the screen. This change also required that the story execution be slowed down so that a reader

could choose to activate these actions before the story was completed.

Wide Ruled 2 also introduced an underlying story generator that is driven by the ABL reactive planning architecture used in interactive dramas like Façade [9]. This planner utilizes a similar hierarchical decomposition as that of the UNIVERSE model, and, due to its reactive nature, easily facilitated our new interactivity model. This new reader-driven interactivity model also prompted another change to the underlying story model. In Wide Ruled 1, if an author goal had no executable plot fragment, the generation loop would back-track its execution, erase any outputted text, and choose another valid plot fragment in a previous author goal in the execution stack. This model was not ideal for the online, read-as-you-execute text output model of story generation that is required of the new real-time interactivity model, and would result in the removal of text that was already viewed by a reader. As a result, backtracking was removed from Wide Ruled 2; if the story generator encounters an author goal with no valid plot fragments, story execution halts.

The wizard interfaces used in Wide Ruled 1 to create new precondition constraints and story actions were removed in version 2 in response to user feedback. The slow nature of the step-by-step guidance through the creation process became cumbersome once a user became proficient in plot fragment editing. Wide Ruled 2 implements a more direct, list-based interface that allows the user to edit all components of a constraint simultaneously. The initial learning curve required to use this new interface proved to be minimal and reduced authoring time for most authors.

7. REAL-WORLD WIDE RULED

Since 2007, Wide Ruled 2 was used in three different classroom settings by a total of 91 students with mixed non-technical (digital arts, new media, literature) and technical (computer science, game design) backgrounds. At the University of California, Santa Cruz, we included Wide Ruled 2, like Wide Ruled 1, in two sections of the Interactive Storytelling class hosted by the computer science department. This class is cross-listed as an undergraduate and graduate computer science and digital arts and new media graduate class, allowing a varied (although a majority technical) audience to learn the theory, techniques and technology behind the creation of interactive stories. Students in this class were required to complete an assignment using the Wide Ruled 2 program. In a previous assignment, students were instructed to convert their favorite TV show, book series, or movie into a serial narrative in the form of a story grammar. This story was then suggested as the basis for their Wide Ruled stories. Each section was given tool documentation, a full-class lecture describing the tool and how to use it, full-time email support, and a full-class "clinic", in which students were able to receive in-person help and support on their assignment. In addition, they were provided the sample murder mystery story, and one of the sections was given a tutorial to create a Little Red Riding Hood story world. The third classroom setting for Wide Ruled 2 occurred in the Interactive Storytelling class at the National University of Singapore, in their Communications and New Media program. This class, taught by Alex Mitchell, provided a purely non-technical audience for our system, in contrast to the mostly technical group of students at UCSC, increasing the balance of the group with a total of 41 non-technical users. Students in this class were also assigned to create a story world, were lectured on the story generation model and system, and given documentation, the sample story, and the Little Red Riding Hood tutorial. In the following sections, we describe both quantitative story data metrics as well as a qualitative analysis of our experiences teaching people to create stories using Wide Ruled.

7.1 Quantitative Analysis

In this section we describe story metrics resulting from the batch analysis of student story worlds. Because each classroom usage scenario was slightly different due to changes in assignment requirements, evolving documentation, minor evolution of the tool, and differences in teaching methodologies and styles between universities, this analysis is not the result of a strictly controlled series of studies, but a high level quantitative perspective on the trends and features present in our sample of student story worlds. It is meant to provide the reader a feeling for the quality and content of the stories created with our system, and provide support for the qualitative analysis and lessons learned described later in this paper.

7.1.1 Story Complexity

In general, the overall complexity of story worlds students created in Wide Ruled 2 is slightly lower than expected, but demonstrates usage of all of the key generative features of the story model. With regards to story world objects, students tended to have an average of 7.88 characters, 3.94 environments, 3.12 plot point types, and 9.52 author goals, with a total of 18.34 plot fragments per story world. In general, bigger story worlds, with more available characters, environments, plot fragments and plot points, result in stories with more variation (plot fragment and textual variation as different objects and characters as matched in preconditions during generation) and/or larger overall length (more information to utilize in the story). These numbers are on par or are larger than the provided story world, demonstrating initiative by students to create story worlds more complex than the ones provided to them. Unfortunately, relationship attributes, in general tended to be less common, with almost none appearing in environment objects (0.18 on average per story), and only an average of 1.3 relationship attributes associated with the characters in each story world. Traits for characters and environments were much more common, with 6.2 and 2.2 character and environment traits on average per story world, respectively. The low occurrence of relationships between characters and environments relative to the murder mystery sample story, suggests that there might be better ways to manage this type of information, as discussed in the next section. A small number of relationships in a story world typically indicates that the resulting narrative relies on very little inter-character or inter-environmental dynamics, or that many of the interactions are statically written into the story text. In the former case, the generated stories will have limited complexity, and in the latter, lower variability.

Story hierarchy size metrics demonstrate varying success. For each author goal, there were, on average, more than four plot fragments that implemented that goal, however for each goal there were only an average of 1.02 subgoals per author goal, which is lower than the sample story, implying that subgoal actions were not often used within plot fragments and the depth of the story hierarchy is lower than what is ideal for complex story variation. However, students tended to use 1.6 precondition constraints and

2.34 actions per plot fragment, with an average of 1.7 variable bindings and 2.78 total variable references per plot fragment, both of which are similar and greater than the provided sample story. This implies that plot fragments were not only using multiple constraints, but these constraints properly bound variables and these variables were used in story actions, which is an essential feature of Wide Ruled that captures much of its generative power. An ideal story hierarchy is broad as well as deep, as mentioned in section 4.3, encompassing multiple ways to accomplish each author goal (breadth), and complex plot fragments that have large amounts of deep hierarchical substructure (depth). The stories created by the students had an average explicit story hierarchy depth of 2.18 goals (this does not including recursive repetition, which can deepen an active story tree), and an average of 3.4 author goals in height at the deepest point. These numbers match or exceed that of the sample story, and show that students are decomposing their story structure to an acceptable extent. The number of precondition constraints and variable bindings and references within a plot fragment should scale with the size of the story world, as more plot fragments interact with more characters, environments, and plot points, and in turn print out text containing larger amounts of dynamic story information. If the constraint count is much higher than variable usage on the whole, then one can infer that the plot fragments, in general, are not utilizing information, but only limiting the likelihood of a plot fragment executing. The resulting story will likely contain lower variations between generated stories. On the other hand, if the precondition constraint count is dwarfed by number of variable bindings and references in a plot fragment, then the resulting story will likely contain textual variation, but a high amount of repetition of story information within a single plot fragment.

Story actions consisted primarily of subgoaling, plot point editing, and character editing, By far, subgoaling was the most common story action within plot fragments, happening on average 0.5 times per plot fragment, while story object editing on characters and plot points occurred at a rate of 0.11 and 0.35 edits per plot fragment respectively, revealing that student story structures were character- and plot point- centric, which is confirmed by earlier observations which we comment on in a later section. It is encouraging that subgoal actions were commonly used among students, since subgoaling is a primary component of the UNIVERSE model, and is a primary way to encode story variation. Students also tended to create but not delete plot points, with creation occurring 0.27 times per plot fragment, and deletion happening a much lower 0.005 times per plot fragment, on average. It is clear that environments are a highly unused feature, and were involved in an average of only 0.008 actions per plot fragment, which we comment on in a later section. The lack of plot point deletion is expected, since they are typically used to keep track of plot progression information, and therefore are created constantly referenced throughout the story. In addition, due to the smaller size of story worlds created in these classroom settings, users typically never needed to manage a large number of plot points of the same type simultaneously as a story was generated, therefore reducing the need to delete these story objects during generation.

Between technical and non-technical authors (technical authors have more programming background as determined from an author survey), technical authors have more complex story worlds across the board, with a notably larger number of subgoals per plot point (40% more), and 34% more plot point edit actions per plot fragment. In addition, non-technical authors tend to reference variables less per plot fragment (-29%), especially within precondition constraints (-93%) A majority of the rest of our measures of story complexity were also higher for technical authors, but to a lesser extent. These deficiencies seem to imply that non-technical authors tend to utilize fewer of the generative features of Wide Ruled, especially the dynamic named variable binding components. This highlights the need to make the story model more accessible, which is discussed in a later section.

Notably, technical authors tended to use the calculation story action 79% more than non-technical authors, author goal parameters 92% more, precondition negation ("There does not exist …") 59% more, and variable references 29% more. All of these Wide Ruled features are aspects of our story generation model that most closely resemble software programming constructs. It is not surprising that incrementing values (which made up the majority of calculations), passing parameters, and referencing variables are common operations for authors with a programming background. Finally, non-technical authors had story worlds that were 60% more similar to the sample murder mystery story world provided to students. This similarity value was calculated as a distance between feature vectors of these and other quantitative story world metrics. This implies that non-technical authors may be more hesitant to experiment with this story generation model, and prefer to stick to closely to the sample story when possible. Our future work section discusses ways to make the story model more accessible to non-technical users.

7.1.2 Text Variation

Because Wide Ruled 2 is a text-based generator, analysis of the only method of output, printed parameterized text, should provide insight into the success of the tool as a powerful dynamic story generator. In total, text output actions made up 1.05 actions per plot fragment and were, on average, 294 characters long, with 2 parameterized text blocks per text output action. The parameterized text tended to be single words, and most often nouns, specifically names of people, places, and things. We believe that this amount and kind of dynamic text is a reasonable level of variation, and is close to that of the provided sample story, but we would have liked to see experimentation with more richly parameterized text fragments.

Interestingly, non-technical authors created, on average, text output actions with 478 characters, 179% larger than the technical authors tended to do. There was a smaller (8%) increase in the average number of text output actions for the non-technical audience, but the number of parameters within these text blocks were slightly lower (-5%). This corresponds with lower (-29%) counts of variable references throughout the plot fragments as a whole, suggesting that these authors, while proficient with static narrative creation, are utilizing less the dynamic textual features of the system.

7.2 Qualitative Analysis

The quantitative metrics of the student story worlds does not tell the entire story about the various strengths and weaknesses of Wide Ruled as a story authoring tool. In this section we will cover unusual story structure failures, common complaints, unused features, and general problem areas we and our colleagues encountered as we taught and supported students using Wide Ruled. While this list of deficiencies is daunting, rest assured that

the majority of the students that used the tool successfully created unique generative and interactive narratives. We have learned the most from the failures and complaints encountered along the way, and therefore focus on them as a motivation for future work.

7.2.1 Outlier Stories

With our experiences in the classroom, there are some interesting story worlds that highlight some of the potential conceptual troubles encountered with this story generation framework.

Orphan story hierarchy subtrees. A very common issue students encountered was portions of the story hierarchy that are either never subgoaled, or are blocked by a set of preconditions on a plot fragment that can never be true. This is usually caused by the process of editing a plot fragment, temporarily deleting a subgoal, and never remembering to reattach that subgoal and its subtree of the hierarchy back to the main story hierarchy. This could be solved by providing a useful visualization of the story world, which is addressed in our future work. A precondition blocking a portion of the story space is usually a misunderstanding a complex set of interacting precondition constraints, which is an inherent difficulty in managing these generative constructs.

Plot point driven story. In one story world, a student used plot points as a way to manually drive the progression of the story generator. Instead of arranging the story world as a large hierarchy of goals and letting the story generator execute naturally, he or she had created only a single author goal with many plot fragments. The plot fragments would check for the existence of a certain plot point type, and if it existed, print text and create a new, different plot point type, and subgoal to the same parent author goal, creating a loop which would then trigger a different plot fragment. The result was a story world with many plot point types that replaced the built in sequential ordering in the generator. This was likely a misunderstanding of the purpose of plot points as a way to store important plot information, rather than as a way to drive the progression of the plot.

Static story worlds. In a few student stories, very few or no characters, environments, or plot points were used during story generation within plot fragments (even if they were present in the story file). Most or all of the content of the story was embedded completely in the author goal and plot fragment hierarchy, and stored in almost completely static text. The result was a story world that was nearly functionally identical to a story grammar. The conceptual problems here likely came from difficulties in converting the previously created story grammar assignment to a Wide Ruled story.

Mostly sequential story spaces. Another common issue was story worlds that were mostly sequential. There are two ways that this occurred: either through a very broad tree with many sequential story actions, or a very deep tree with a chained series of subgoals. Both of these situations are depicted in Figure 4. The narrative in these story worlds was mostly linear, with little variation between each story instance, and the text output actions were often primarily static. This kind of problem appears to be the result of difficulty decomposing a story into a hierarchical arrangement. Students would revert to a mostly linear story and attempt to still use the subgoaling or sequential story action features of the generator.

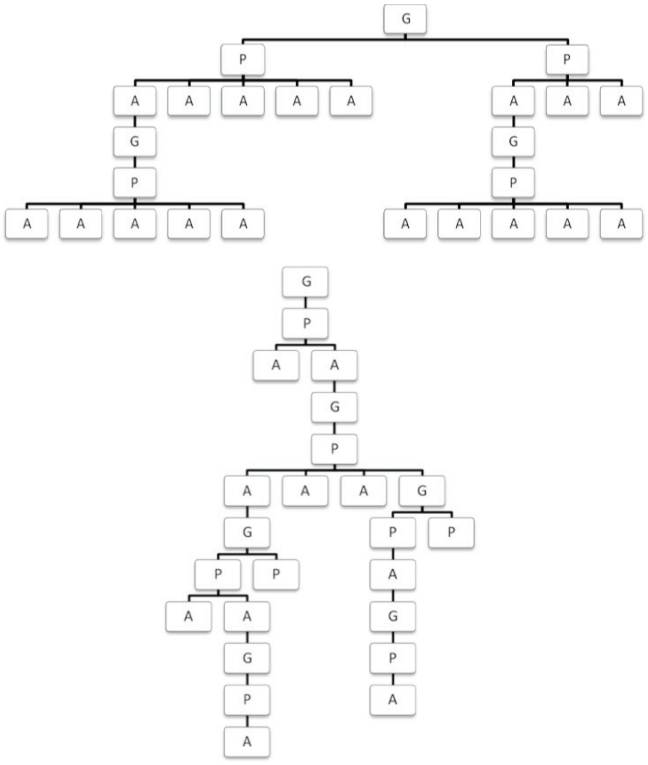

Figure 4. Example problematic story worlds. The top and bottom image depict primarily depth (through subgoaling) and breadth (through sequential story actions) in a story space, respectively, but both result in low variation between story instances. Ideal story worlds exhibit similar breadth and depth for maximum story variation.

7.2.2 Programmers and Writers

As Wide Ruled was being used by both technical and non-technical students, it became clear that the combination of a story-focused system with generative formalisms could cause usability problems with students of either background. Some technical users of Wide Ruled would ask whether a specific feature of a programming language existed within the system, or how that feature could be replicated. The majority of these requests were the result of authors with extensive programming backgrounds struggling with the hierarchical, non-deterministic execution model of Wide Ruled, and attempting to employ traditional programming language constructs in an ad hoc way. For example, some students would create a large Boolean "OR" construct by creating two identical plot fragments with a single constraint differing between the two. Another student created a blank plot fragment in order to provide a way to skip over an author goal. While all of these solutions are valid (and sometimes clever) ways of constructing a Wide Ruled story, they were usually part of a tedious process of attempting to shoehorn a traditional, deterministic program structure into the inherently non-deterministic plan-like story generation model. The result was often a very complex story structure that did not necessarily produce an interesting generative narrative. In the personal experience of one the authors, in a workshop in which Wide Ruled was used with a mixture of undergraduate and graduate students, all with technical backgrounds, the graduate students

seemed to have more trouble creating a new story in the system than the undergraduates at the same seminar. These older, more experienced students didn't appear to "get it" (the story model), because they were so used to traditional programming techniques that the author goal model became a hindrance, while the undergraduate students appeared to be more accepting of the new system and to more easily adapt to it. In contrast, non-technical authors using Wide Ruled often struggled with errors typically encountered by those learning to program, such as unreachable code (plot fragments), endless loops (infinite subgoal recursion), and incorrect or misspelled variable references. It is clear that both of these user backgrounds can result in a problematic learning or unlearning curve for either type of user.

The source of these issues is likely an incorrect balance of computational and literary constructs exposed and taught to the students. There are components of Wide Ruled that still remain inherently very programmatic and therefore complex and unfamiliar to a non-technical author even when they are designed to be as story-centric as possible (described in the next section). At the same time, these technical features closely resemble many traditional programming constructs that can obscure the narrative purpose of these features to someone who has encountered them in other technical contexts. The documentation and in-class lecture describing Wide Ruled focused primarily on the individual features of the model, and the underlying technical capabilities of the system. Even with a provided tutorial and example story world, we believe that some of the trouble that students have with Wide Ruled stems from a lack of proper conceptual decomposition of the system in the context of story creation.

7.2.3 Problem Areas

In addition to the specific story worlds described in section 7.2.1, students using Wide Ruled had troubles with some of the specific features of the tool. In the final section of this paper we discuss some possible solutions to these troubles, as well as for the failures discussed in previous sections.

Constraint programming and variable binding. Complex Boolean constraints, chained precondition variable bindings, and variable reference management are not intuitive concepts. These constructs are considerably harder to grasp for those without a technical background, and pose a large hurdle between the current state of the system, and the ultimate vision of it as a practical tool for non-technical, arts and writing focused students.

Interactivity. The interactivity model present in Wide Ruled 2 was not fully complete for its initial classroom usage, and as a result, only two classes were given an example of it in use, and were able to effectively employ the feature in any meaningful manner. Specifically, the NUS class was required to implement at least two interactive actions, and the second UCSC section was given the option to receive extra credit for implementing an interactive action. While the general feedback from both these classes showed interest in the feature, and a much more positive response than the previous interactivity model, the open-ended nature of it made practical usage difficult. Currently, interactive actions can be activated at any time during story generation. According to student feedback, designing an interactive story action that can be activated any time during generation, and still result in an interesting variation in the story, is generally hard to design for and unintuitive. Multiple students suggested that tying interactive actions to more limited and localized plot fragment or author goal specific contexts would be much more useful for storytelling.

8. CONCLUSIONS AND FUTURE WORK

This work has shown that Wide Ruled is a usable and practical system for technical and non-technical users to harness the power of a plan-based story generation model. It has been used multiple times by students with varying backgrounds to successfully create generative, interactive stories, and it exists as a unique system – the only public implementation of a UNIVERSE-like model of story generation, and one of a very few set of stable, supported, and openly available interactive plan-based story generation tools. Even with these successes, it is clear that there is much work to be done to develop Wide Ruled into a tool that is truly friendly to writers, game designers, and other non-technical authors.

Assistive, visual, intelligent interfaces. While authors, in our experience, are indeed creating dynamic and interactive story worlds with Wide Ruled, the story structure complexity, story world size, and textual variation in these stories falls behind the work of those with a technical background. The main areas of improvement to be made exist in the realm of assisting users in performing, and sometimes automating, complex tasks that are common for technical authors, but totally foreign to those with training in traditional literary methods. Particularly, we must focus on aiding users in the generation and understanding of the potentially complex constraints that may exist within plot fragment preconditions in complex story worlds. These logical constructs are a key part of the Wide Ruled story model, and every author wishing to harness the generative power of the system must use them liberally. In addition, the creation and visualization of relationships and the constraints on these relationships within preconditions could eliminate much of confusion surrounding these unused features, especially since compelling dramatic narrative can often make use of character relationships that break and reform. Adding visual management and navigation of environment relationships would also likely encourage their use in stories. Finally, the management of variable naming, binding and referencing within and amidst preconditions and story actions should be made as intuitive as possible. This could be accomplished with a visual graph-like tool for managing references, removing the explicit naming of variables unless absolutely necessary.

Evaluating creative and expressive benefits. In future studies, an additional, more complete evaluation of Wide Ruled and its successors in the context of enhanced creativity and expressiveness would be useful in determining the tangible gains of the system over traditional authoring techniques. While no existing UNIVERSE-based authoring tools exist, a comparison to a paper-based, reader-selected branching narrative model, such as is demonstrated in the popular "Choose Your Own Adventure" series of books, might lend some insight into the expressive power of the UNIVERSE model and our interface to it. Specifically, by evaluating the authoring process with this traditional "analog" interactive storytelling method alongside the more complex computer-based Wide Ruled system, perhaps we could determine if authors were empowered by our interface and its underlying model, or in fact hindered by its complexity.

Better story-centric feature decomposition and presentation. Although automation and/or assistance in the interface can provide help to non-technical users, there is much potential

benefit in developing a more writer/story focused method of presenting the features of the system, with smaller examples that cover narrative constructs and their interpretation in the Wide Ruled story model. As mentioned in section 7.2.2, students of all backgrounds had occasional trouble in understanding the capabilities of Wide Ruled in the context of a story. Writers were confused by the technical details of the model, while programmers failed to understand how a familiar computational construct was useful for storytelling. While the tutorial and provided example story world demonstrated complete stories, our system did not provide decomposed "story functions", or isolated examples of individual computational features utilized in a narrative context, to connect the technical theory with practical storytelling techniques. To address this weakness, our system itself could provide auto-generated, templated story components, such as a repeating event loop, or a simple rising and falling conflict model, that demonstrate from directly within the system computational constructs suitable for generative narratives. This would provide a straightforward way to guide an author without requiring them to reverse engineer a sample story world. A change of presentation could also help alleviate the problems that technical users tend to have with trying to fit Wide Ruled into a traditional programming language model. If our system can be demonstrated consistently as fundamentally a generative narrative environment, instead of a set of technical features and an execution model that happens to be able to generate stories, then both writers and programmers alike will benefit from this new perspective.

The future of this work lies in all of the above improvements, combined with a vastly more scalable, visual and intelligent programming environment based on storyboard presentation of storytelling. The structural and iconic language of sequential art, of which storyboards and comics are two instances, has been sufficiently analyzed to provide knowledge and heuristics for computational storyboard generation [2, 10]. Sequential art can communicate a large amount of emotional and situational information in a small space, by making careful use of drawing style, coloring, shading, panel ordering and arrangement, and panel composition. The Wide Ruled story model, with its sequential story actions and easily visualized hierarchical story structure, lends itself well to this representation. Furthermore, by replacing traditional lists and scroll bars with visual graphs and other spatial representations of the complex preconditions and variable binding interactions that occur in our story model, Wide Ruled and its successor will handle larger and more complex story worlds (much larger than those created in our classroom evaluations), and thus much more deeply varying and longer generated stories. It is our hope, that by combining the visual metaphors of the storyboard with the Wide Ruled story generation model, we can evolve our story generation system into an authoring tool that fully addresses the weaknesses of Wide Ruled and moves that much closer to the vision of a truly writer-and designer-friendly system for the creation of rich, compelling, dynamic, and interactive narrative.

9. ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grant No. IIS-0747522. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

10. REFERENCES

[1] Barrenho, F., Romao, T., Martins, T., Correia, N. 2006. InAuthoring environment: Interfaces for creating spatial stories and gaming activities. In Proceedings of the 2006 ACM SIGCHI international conference on advances in computer entertainment technology.

[2] Bernstein, M. 2002. Storyspace 1. In Proceedings of the Thirteenth ACM Conference on Hypertext and Hypermedia (College Park, Maryland, USA, June 11 - 15, 2002). J. Blustein, Ed. HYPERTEXT '02. ACM, New York, NY, 172-181.

[3] Cash, S. Patrick. 2007. Bowyer: A Planning Tool for Bridging the gap between Declarative and Procedural Domains. (Master's Thesis). Technical Report. etd-10212007-210447, North Carolina State University.

[4] Donikian, S., Portugal, J. 2004. Writing Interactive Fiction Scenarii with DraMachina. In Proceedings of TIDSE.

[5] Eisner, W. 1985. Comics and Sequential Art. Tamarac, FL: Poorhouse Press.

[6] Gebhard, P., Kipp, M., Klesen, M., Rist, T. 2003. Authoring Scenes for Adaptive, Interactive Performances. In Proceedings of AAMAS-03, pp. 725-732.

[7] Lebowitz, M. 1985. Story Telling as Planning and Learning. Poetics 14, pp. 483-502.

[8] Lebowitz, M. 1984. Creating Characters in a Story-Telling Universe. Poetics 13, pp. 171-194.

[9] Mateas, M. and Stern, A. 2002. A behavior language for story-based believable agents. IEEE Intelligent Systems, July/August 2002, 17 (4), pp. 39-47

[10] McCloud, S. 1993. Understanding Comics. New York, NY: Kitchen Sink Press/Harper Perennial.

[11] Sauer, S., Osswald, K., Wielemans, X., Stifter, M. 2006. U-Create: Creative Authoring Tools for Edutainment Applications. In Proceedings of TIDSE 2005, pp. 163-168.

[12] Silverman, B., Johns, M., Weaver, R., Mosley, J. 2003. Authoring Edutainment Stories for Online Players (AESOP): A Generator for Pedagogically Oriented Interactive Dramas. In Lectures Notes in Computer Science: Virtual Storytelling. Springer

[13] Skorupski, J., Jayapalan, L, Marquez, S., and Mateas, M. 2007. Wide Ruled: A Friendly Interface to Author-Goal Based Story Generation. In Proceedings of ICVS.

[14] Thomas, J., Young, M.R. 2006. Author in the Loop: Using Mixed-Initiative Planning to Improve Interactive Narrative. In the ICAPS 2006 Workshop on AI Planning for Computer Games and Synthetic Characters.

[15] Zagalo, N., Göbel, S., Torres, A., Malkewitz, R. 2006. INSCAPE: Emotion Expression and Experience in an Authoring Environment. In Proceedings of TIDSE.