Planning-Integrated Story Graph for Interactive Narratives

Wook-Hee Min, Eok-Soo Shim, Yeo-Jin Kim, and Yun-Gyung Cheong
Graphics&OS Group, Samsung Advanced Institute of Technology
Giheung-Gu, Gyeonggi-Do, South Korea
+82-31-280-9686
{ wookhee.min, esp.shim, yeoj.kim, yuna.cheong }@samsung.com

ABSTRACT
The advances in the interactive contents enable users to have a variety of experiences on diverse devices. In particular, two main approaches have been researched to construct digital interactive contents: a) conditional branch techniques and b) planning techniques. Each approach offers its own benefits; the conditional branch techniques allow the user to create tightly-plotted interactive contents; the planning techniques reduce the author's burden to specify every possible connection between contents considering the user input. As an attempt to combine these advantages provided by each technique, this paper discusses an interactive story structure incorporating the planning technique into the conditional branch techniques. Also, we briefly describe PRISM, a framework capable of creating and playing our story structure. We expect that the author can compose well-woven stories which can respond to a wide range of user interaction.

Categories and Subject Descriptors
E.1 [**DATA**]: Data Structures – Graphs and networks; I.3.6 [**COMPUTER GRAPHICS**]: Methodology and Techniques – Interaction techniques; H.5.1 [**INFORMATION INTERFACES AND PRESENTATION (e.g., HCI)**]: Multimedia Information Systems – Animations

General Terms
Design

Keywords
Interactive Narrative Representation, Story Structure, Interactive Content Structure, Interactive Narrative Platform

1. INTRODUCTION
While a variety of approaches have been introduced to create interactive contents, two approaches have been commonly used: a) conditional branch techniques with which users experience a story that flows according to conditional branches as specified at design time, and b) planning techniques with which sequences of story units are constructed toward story goals.

SRMC'08, October 31, 2008, Vancouver, British Columbia, Canada.

The conditional branch technique has been recognized as an effective tool for the author to create well-woven interactive contents [2; 3; 4; 8; 9]. A branching narrative is typically illustrated as a story graph as in Figure 1, which consists of nodes and conditional branches; a node denotes a series of scripted scenes, and a conditional branch denotes transition from one node to another depending on the user input. This approach is adequate to tightly-plotted interactive stories. As making use of such an approach, INSCAPE [2], a software tool for the user to plan, build and experience interactive stories utilizing graphical representation, splits a story into logical containers known as stages and situations and it constructs a story graph using branching techniques. Although this approach gives the author intuitive illustration of an interactive story, it has little flexibility to handle the user interaction; the plot only flows as the author's explicit design. Therefore, a highly interactive story system confronts explosion of conditional branches as recognized by a number of researchers [1; 3; 5]. Thus, the author is burdened with constructing the plot taking into accounts all possible user input at the design time; otherwise, the user's interaction would be ignored.

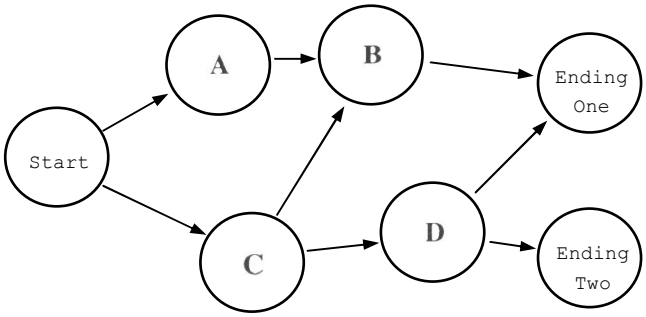

Figure 1. Story graph used in the OZ project [4]

In an interactive story system employing the planning techniques, on the other hand, the author is not responsible for linking every single node to node; instead she needs to create story units and encode their preconditions and postconditions with which a planning algorithm searches for the next content as the user interacts with the system. A precondition is a logical statement which describes the state for the story unit to take place; an postcondition is a logical statement which describes a state that is altered during the unit execution. With this planning formalism annotated units and a goal state along with an initial state, the planning algorithm constructs a story plot as a sequence of units.

When the system receives user input that requires changes to the current story, the system re-plans it to deal with the interaction. For instance, MIMESIS [11] uses a partial-order plan for a fully structured plot where it either incorporates user's arbitrary actions into the story or tries to avoid them if the actions conflicts with the plot. In the real-time interactive drama Façade [5], its drama sequencer selects the next story unit (called story beat) from the ones that their preconditions are fulfilled in the current state. As an attempt to combine both approaches (branching narratives and planning techniques) Magerko [6] presents an authoring tool that constructs a complete story structure from a partially specified structure using a planning technique.

This paper presents an interactive content structure, called PRISM container, which incorporates the planning formalism into branching narratives. As a result, the PRISM container can describe tightly-plotted contents, which are developed at the design time, with the conditional branch techniques and, at the same time, provide the creation of highly interactive story contents through a planning technique.

In the next section, we discuss the PRISM container in detail. Then, we introduce a sample scenario for an interactive content based on a short story and represent it with PRISM container. Next, we briefly describe our interactive narrative system named PRISM which utilizes the PRISM container. Finally, we conclude this paper by discussing extensibility issues for this approach.

2. PRISM Container: Interactive Content Structure

A PRISM container is an XML document that represents an interactive content. To encode a PRISM container, we define an xml schema that contains one Storymap and a series of Actions; a storymap element describes the story space that the user would explore; it consists of Situation, Condition, State, StateUpdate and Browsing elements. Each element except for the browsing has its own ID and thus can be referenced by other elements through the ID, so that there is no need to define the same element redundantly. On the other hand, actions indicate immediate response to the user input, such as a character performing repetitive gestures responding to a mouse click. In this section, we describe all elements contained in the PRISM container that can cover any interactive contents fully.

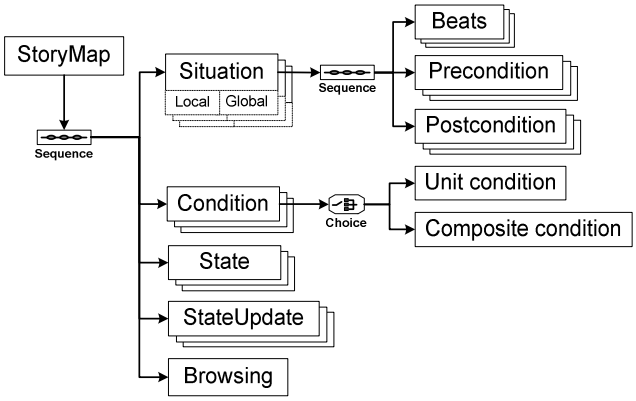

Figure 2. Storymap structure of the PRISM container

2.1 Situation

Situation is the same term as a node in a typical story graph as in Figure 1. An author constitutes a situation from a sequence of beats—an atomic unit of dramatic action, written in XML. Situations are classified into two types: local and global situations. While the former is related to the main story plot, the latter is not directly related to the main story but serves as supplements to the story. Global situations can be reached from any situations when their conditions are met. In addition, once a global situation is activated, the story returns to the previous story point—where the global situation is called—when the global situation is completed. An example of this is described in section 4. One of the most important characteristics of situation is that a situation has optional elements of preconditions and postconditions, which enable the use of planning. More details on planning using our structure are discussed in the following sections.

2.2 Condition

A condition is a logical statement that describes user input or the story state for the interactive system to determine the execution of conditional branches, stateupdates and planning in a given situation. Conditions can be classified into two different types: unit condition and composite condition. The unit condition is a single condition that represents a specific event or a story state. For example, the condition C1:LeftClickOnCharacterA denotes an event when the user clicks on the character A, and the condition C2:CharacterAHasSword represents the story state that the character B is having a sword. A composite condition is constituted by a compound of multiple conditions and logic operators such as AND and OR. For example, the conjunction of "LeftClickOnCharacterA" and "CharacterAHasSword" can be represented as (C1 AND C2). By employing this composite condition, complicate conditions can be described in a simple way.

Although conditions can be referenced by browsing or situations, this section describes only conditions referenced by situations; the conditions referenced by browsing is explained in the section 2.4. When conditions are referenced by a situation, they serve as its preconditions—logical statements that should be fulfilled prior to be reached from other situations.

2.3 State and StateUpdate

The state element denotes information about the story world that should be maintained during the play of an interactive content, thereby influencing the story development. States are categorized into two types: the generic state and the content-specific state. The generic states describe information that is commonly maintained regardless of interactive contents such as the total play time, the total number of user interactions and so on. Such states are always managed by the system without the author's specification. On the other hand, the content-specific state is collected and updated only when the author specifies its associated stateupdate elements in the PRISM container. For instance, ST1:CharacterAWeapon describes the type of weapon that the character A possesses.

In addition to the state, the PRISM container contains StateUpdate; as its name implies, StateUpdate is a functional description that specifies the alternation of a state. For example, if the character A, mentioned above in ST1, acquires a shotgun for his weapon, the stateupdate STU1 can be defined as in Figure 3.

```
<StateUpdate stateUpdateID=STU1>
    <RefStateID> ST1 </RefStateID>
    <Update> Set </Update>
    <Value> shotgun </Value>
</StateUpdate>
```

Figure 3. Example for Stateupdate

2.4 Browsing

The browsing element plays a significant role in structuring the multi-story line of an interactive content by integrating situations, conditions and stateupdates. The role of browsing corresponds to the set of arcs in conventional branching narrative. Browsing consists of a set of SituationCheck elements, each of which has its own ID referring to a situation contained in the storymap. Each SituationCheck, which corresponds to the source situation, contains a sequence of BranchCheck elements, each of which consists of RefSituationaID (i.e., destination situation), RefConditionID (i.e., a condition that triggers the activation of the destination situation), and CaseValue. The casevalue can be either true or false. If the casevalue is set true, branching to the destination situation is executed when the condition is fulfilled; otherwise, the branch would be executed when the applied conditions are not fulfilled.

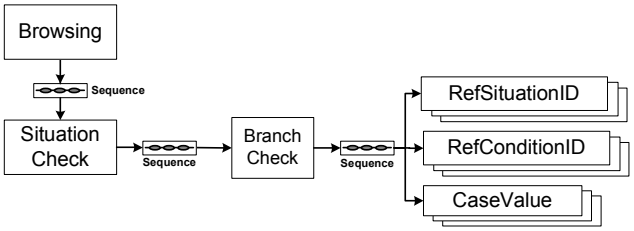

Figure 4. Browsing structure in Storymap

Further, it is possible to use combination of multiple conditions. If an author wants to make the branch B1 which is reachable only when the condition C1 and the condition C2 are satisfied, then she should assign true on the both casevalue of C1 and C2. If the author wants to make branch B2 which is reachable when C1 is satisfied but C2 is not fulfilled, she should assign true on the casevalue of C1 and false on that of C2.

2.5 Actions

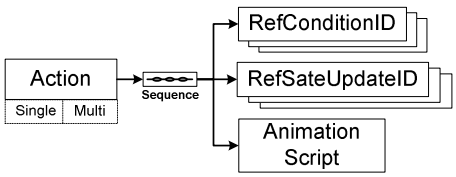

Figure 5. Action Structure for PRISM Container

The action element refers to a temporary and immediate animation which is irrelevant to the story flow but presents pleasing animations to the users while she interacts with the system. Figure 5 represents the schema structure of the action element. As described in the figure, the action element is classified into two types: the single-object action and the multi-object action. The single-object action is the animation that involves a single object such as a character making a somersault when a user clicks on it. The multi-object action, on the other hand, is the animation that involves multiple objects, such as an animation of pouring water into a cup when the user drags a jar onto the cup.

3. Sample Scenario: One Thousand Dollars

The sample is based on the short story "One Thousand Dollars" by O. Henry. The story begins with a situation in which Mr. Gillian was told that he inherited one thousand dollars from his late uncle, who had been supporting him, by a lawyer. According to the will, however, he was required to account for every penny he spent. There was another heir, Miss Hayden, a ward of his late uncle, who was left only a ring and the $10. Mr. Gillian consulted with his friends and people he met on the street about the proper usage of his money. He finally visited Miss Hayden, whom he had loved alone, and gave her the one thousand dollars. Back at the lawyer's office, he was told that fifty grand for him will be paid to Miss Hayden if he had squandered his money. He thereon told the lawyers that he lost the money at the race track and left with whistling [7].

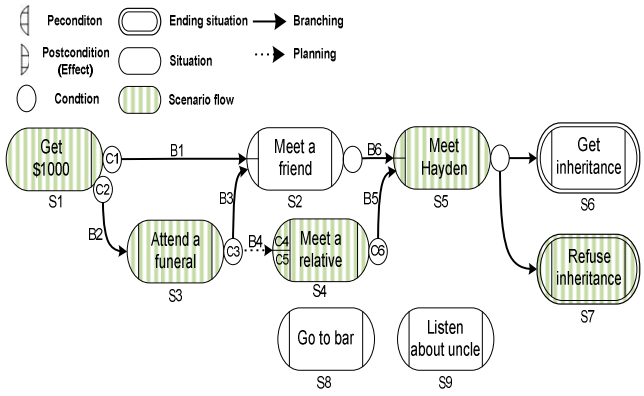

Figure 6. Sample interactive content story map: "One Thousand Dollars"

We adapted the original content to an interactive content as described in Figure 6. In our scenario, a user plays the role of Mr. Gillian's friend who can influence his decision by expressing opinions through interaction such as typing text or clicking mouse buttons. The scenario begins with the initial situation S1:*Get $1000* in which Mr. Gillian has been just given one thousand dollars from the lawyer. Mr. Gillian is then asked to make a choice between whether he goes to meet his friend or to attend his uncle's funeral. If he makes a selection of the latter, the story progresses to the situation S3:*AttendAFuneral*, which is not included in the original novel. In this *AttendAFuneral* situation, he goes to the funeral where he encounters lots of people including a female relative. At this point, the user may encourage Mr. Gillian to have conversation with the relative, which drives the story into the S4:*MeetARelative* situation. While talking with the female relative in the *MeetARelative* situation, he becomes

aware that his uncle assigned a huge amount of money to be given to him in case his expenditure of $1,000 was desirable. Knowing this fact, Mr. Gillian departs to meet Hayden and have a talk with her (S5:*Meet Hayden*), and finally, he visits the lawyer back to refuse the inheritance (S6:*RefuseInheritance*).

4. PRISM Container: One Thousand Dollars

This chapter describes a PRISM container (see Figure 7 and Figure 8) that represents the sample scenario "One Thousand Dollars." Due to space limits, we focus only on two situations, S1:*Get $1000* and S4:*Meet a relative*, to explain how the conditional branch technique and the planning technique are integrated in the PRISM Container.

We first describe the branching technique and the stateupdate element used in our approach with an example of the initial situation S1. We then illustrate the use of planning technique with an example of the situation S4. For the reader's convenience, the PRISM container is separated into browsing and non-browsing parts in the following figures. Both Figure 7 and Figure 8 are not describing the whole PRISM container, but a portion of it to demonstrate the expressiveness of the PRISM container.

As an illustration of the branching technique, the initial situation S1:*Get $1000* contains two single conditions C1:*Typed(MeetFriend)* and C2:*Typed(AttendFuneral)*. These conditions have their associated conditional branches B1 and B2 respectively such that B1 advances the story to the situation *S2:Meet a friend* and B2 advances the story to the situation *S3:Attend a funeral*. Thus, if the user types "*Attend a funeral*," she will be presented with the situation S3, as specified by the browsing B2. In addition, the situation S1 has a postcondition that is associated with the stateupdate STU_MONEY_SET which assigns 1000 to the state *ST_MONEY* that keeps track of the amount of Mr. Gillian's money. Thus, once the situation S1 is completed, the state is set one thousand.

```
<StoryMap>
  <SituationGroup>
    <LocalSituationList>
      <Situation situationID="S1">
        <Postcondition>
          <RefStateUpdateID>
            STU_MONEY_SET
          </RefStateUpdateID>
        </Postcondition>
        <Description>Get $1000</Description>
      </Situation>
      <Situation situationID="S3">
        <Postcondition>
          <RefStateUpdateID>
            STU_LOCATION_FUNERAL
          </RefStateUpdateID>
        </Postcondition>
        <Description>Attend a funeral</Description>
       </Situation>
      <Situation situationID="S4">
        <Precondition>
          <RefConditionID>C4</RefConditionID>
          <RefConditionID>C5</RefConditionID>
        </Precondition>
        <Description>Meet a relative</Description>
      </Situation>
    </LocalSituationList>
  </SituationGroup>

  <ConditionList>
    <Condition conditionID="C1">
      <UnitCondition>
        <SystemDefinedType>Typed</SystemDefinedType>
      </UnitCondition>
      <Value>
        <StringValue>MeetFriend</StringValue>
      </Value>
    </Condition>
    <Condition conditionID="C2">
      <UnitCondition>
        <SystemDefinedType>Typed</SystemDefinedType>
      </UnitCondition>
      <Value>
        <StringValue>AttendFuneral</StringValue>
      </Value>
    </Condition>
    <Condition conditionID="C4">
      <UnitCondition>
        <SystemDefinedType>IstSet</SystemDefinedType>
      </UnitCondition>
      <StatusID>ST_LOCATION</StatusID>
      <Value>
        <StringValue>funeral</StringValue>
      </Value>
    </Condition>
    <Condition conditionID="C5">
      <UnitCondition>
        <SystemDefinedType>Clicked</SystemDefinedType>
      </UnitCondition>
      <Object>
        <ObjectID>OBJ_RELATIVE1</ObjectID>
      </Object>
    </Condition>
  </ConditionList>

  <StateList>
    <State>
      <StateID>ST_MONEY</StateID>
      <StateType>Integer</StateType>
    </State>
    <State>
      <StateID>ST_LOCATION</StateID>
      <StateType>String</StateType>
    </State>
  </StateList>
  <StateUpdateList>
    <StateUpdate stateUpdateID=STU_MONEY_SET>
      <RefStateID>ST_MONEY</RefStateID>
      <Update>Set</Update>
      <Value>1000</Value>
    </StateUpdate>
    <StateUpdate stateUpdateID=STU_LOCATION_FUNERAL>
      <RefStateID>ST_LOCATION</RefStateID>
      <Update>Change</Update>
      <Value>funeral</Value>
    </StateUpdate>
  </StateUpdateList>

  <Browsing>
    …
  </Browsing>
</StoryMap>
```

Figure 7. Sample PRISM container without Browsing part

```
<Browsing>
  <SituationCheck refSituationID="S1">
    <BranchGroup>
      <ConditionalBranch>
        <RefConditionIDList>
          <RefConditionID>C1</RefConditionID>
        </RefConditionIDList>
        <Branch branchID="B1">
          <CaseValueList>
            <CaseValue>True</CaseValue>
          </CaseValueList>
          <DestinationSituationID>S2</DestinationSituationID>
        </Branch>
      </ConditionalBranch>
      <ConditionalBranch>
        <RefConditionIDList>
          <RefConditionID>C2</RefConditionID>
        </RefConditionIDList>
        <Branch branchID="B2">
          <CaseValueList>
            <CaseValue>True</CaseValue>
          </CaseValueList>
          <DestinationSituationID>S3</DestinationSituationID>
        </Branch>
      </ConditionalBranch>
    </BranchGroup>
  </SituationCheck>
  <SituationCheck refSituationID="S4">
    <BranchGroup>
      <ConditionalBranch>
        <RefConditionIDList>
          <RefConditionID>C6</RefConditionID>
        </RefConditionIDList>
        <Branch branchID="B5">
          <CaseValueList>
            <CaseValue>True</CaseValue>
          </CaseValueList>
          <DestinationSituationID>S5</DestinationSituationID>
        </Branch>
      </ConditionalBranch>
    </BranchGroup>
  </SituationCheck>
</Browsing>
```

Figure 8. Sample PRISM container: Browsing part

As an illustration of planning in our approach, the situation S4:*Meet a relative* has two preconditions of C5:*Clicked(OBJ_RELATIVE1)* and C4:*IsSet(ST_LOCATION, funeral)*, where ST_LOCATION refers to the state about the Mr. Gillian's location and OBJ_RELATIVE1 refers to the specific character that plays the role of a Gillian's relative. Because this situation has no incoming branches, it is only reachable when its preconditions are achieved—i.e., when the user clicks the relative character in the funeral; since S3:*Attend a funeral* has the postcondition *Set(ST_LOCATION, funeral*), the situation S4 is reachable from S3 through planning.

As mentioned in the section 2.1, situations are classified into local situations and global situations. In Figure 6, the situations from S1 through S8 are the local conditions while S9 is a global situation. These local situations account for the main story of the interactive content. The global situation, on the other hand, typically provides background knowledge for users. For example, the situation S9 supplies the user with additional information about Gillian's uncle such as his career and occupation.

5. A SYSTEM OVERVIEW: PRISM

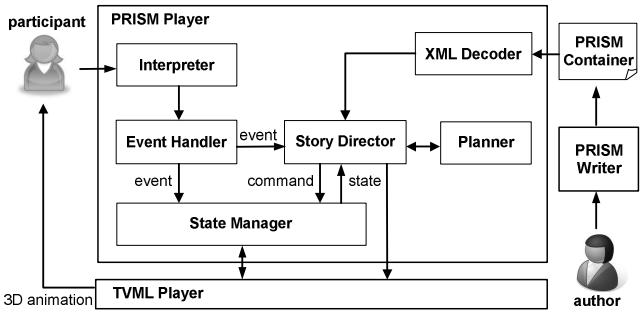

Figure 9. PRISM container processing diagram

This section presents a platform in which a PRISM container is used to represent interactive narratives. PRISM is an integrated system for interactive story creation and presentation; the system takes in a PRISM container and unfolds the story based on the interaction with the participant as if a prism breaks light up into different spectral colors.

PRISM consists of three components: the PRISM writer, the PRISM player, and the TVML (TV program Making Language) [10] player. The PRISM writer is an interactive story authoring tool which interleaves the participant's decision points with story material (i.e. 3D objects, images, sounds, movie clips, etc.) following the PRISM container schema. The PRISM player takes as input the container and determines the next situation to be presented to the user as the interaction arises. The TVML player takes in text descriptions (i.e. TVML scripts) and realizes them into 3D animation.

Figure 9 shows how the PRISM player processes a PRISM container. Once the player takes in the PRISM container, the XML decoder parses it to build the story map structure for the story director. Thereon the story director loads the story map on the memory and requests the TVML player to render the initial situation of the story map. Once the TVML player completes rendering the situation, it requests the state manager to update related states (e.g., situation completion). If the user interaction arises while rendering the content, the event handler dispatches the user input to the story director and the state manager.

On receiving the event, the story director generates a proper response to the user input. If the event corresponds to one of the conditions attached to the current situation, the story director advances the story to the destination situation of the branch. If the event relates to an action, the director gives the TVML player to renders the animation script for the user input. Lastly, if the story director finds no matching conditions from conditional branches and actions, it requests the planner to retrieve the next situation. If the planner returns a situation whose preconditions are satisfied in the current state, the story director progresses the story into the situation. If such situations are not found, the story director takes no response to the user input.

Although conditional branching and situation planning are both available, the conditional branches are more prioritized than the planning in the current design. As an exception, the planning can take precedence over branching when the situation contains only a

single blank conditional branch, which connects the current situation to a destination situation without checking conditions. For example, the situation S2:*Meet a friend* in figure 6 contains a blank condition that leads the story to S5:*Meet Hayden* without the need of user interaction. In this case, the story can advance to the situation S8: *Go to bar* by the planning technique if the user types "Go to bar" which achieve the precondition of S8. However, the decision when to use the branching and planning can be customized by the system designer using a policy table.

6. Conclusions

The conditional branch technique has been extensively used as an effective tool for the author to create tightly-plotted interactive contents [2; 3; 4; 8; 9]. However, since the plot only flows as the author explicitly specifies, a highly interactive story system confronts explosion of conditional branches [1; 3; 5].

To address this issue, this paper presents a hybrid approach to structuring interactive contents by augmenting the planning formalism (i.e., preconditions and postconditions) into the conditional branch technique. As a result, the PRISM container can describe well-woven contents, which are developed at the design time with the conditional branch technique, and provide the creation of highly interactive story contents through a planning technique. In addition, as an effort to enhance reusability of elements in the interactive content structure, we present the composite condition and the casevalue to the story structure, which can express complicated conditions simply by compounding a variety of single conditions via the reference of their IDs. This paper also describes an example adapted from a short original story using our structure and presents an interactive narrative framework that utilizes the PRISM container.

Although the sample example has shown the expressiveness of our structure, we plan formal experiments to evaluate its effectiveness in representing tightly plotted and highly interactive story content in interactive narrative systems as our future work.

7. REFERENCES

[1] Crawford, C. 1989. Indirection. *Journal of Computer Game Design, Volume 3*. http://www.erasmatazz.com/library/JCGD_Volume_3/JCGD_Volume_3_Index.html

[2] INSCAPE. 2008. DOI=http://www.inscapers.com

[3] Iurgel, I.: From Another Point of View: ArtEFact. In: 2nd international conference on Technologies for Interactive Digital Storytelling and Entertainment, pp. 26--35, Darmstadt, Germany (2004)

[4] Kelso, M. T., Weyhrauch, P., and Bates, J. (1993). Dramatic Presence. *Presence: The Journal of Teleoperators and Virtual Environments*, 2(1), 1-15.

[5] Mateas, M. and A. Stern. 2005. Structuring Content in the Façade Interactive Drama Architecture. In Proceedings of the 1st Annual Conference on Artificial Intelligence and Interactive Digital Entertainment (Marina del Rey, USA, June 1-3, 2005), AAAI Press, 93-98.

[6] Magerko, B. 2005. Story Representation and Interactive Drama. In Proceedings of the 1st Annual Conference on Artificial Intelligence and Interactive Digital Entertainment (Marina del Rey, USA, June 1-3, 2005), AAAI Press.

[7] O. Henry, 2001, One Thousand Dollars and Other Plays DOI=http://www.searchlit.org/stories/7687.php

[8] Spierling, U., Weiß, S.A., Müller, W.: Towards Accessible Authoring Tools for Interactive Storytelling. In: 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment (2006)

[9] Swartout, W., Hill, R., Gratch, j., Johnson, W. L., Kyriakakis, C., LaBore, C., Lindheim R., Marsella, S., Miraglia, D., Moore, B., Morie, J., Rickel, J., Thiébaux, M., Tuch, L., Whitney, R., Douglas, J.: Toward the holodeck: integrating graphics, sound, character and story. In: 5th international conference on Autonomous agents, pp.409--416. Montreal, Quebec, Canada (2001)

[10] TVML. 2008. DOI=http://www.nhk.or.jp/strl/tvml/english/what/index.html

[11] Young, R. M., Riedl, M. O., Branly, M. Jhala, A., Martin, R.J., Saretto, C.J. 2004. An Architecture for Integrating Plan-based Behavior Generation with Interactive Game Environments. Journal of Game Development 1(1): 51-70.