An Authoring Tool to Derive Valid Interactive Scenarios

Kim Dung Dang, Ronan Champagnat

An Authoring Tool to Derive Valid Interactive Scenarios

Kim Dung Dang and Ronan Champagnat

University of La Rochelle – L3i, Avenue Michel Crépeau, 17042 La Rochelle, France
{kim_dung.dang, ronan.champagnat}@univ-lr.fr

Abstract

In recent years, many tools have been proposed to design interactive scenarios. The aim of these tools is to provide users with a framework in order to write a story with many branches which will be unfolded by an artificial intelligence application. However, the consistency and quality of the generated narratives are not guaranteed (deadlocks, flaws, *etc.*). Previous work has defined the properties of a valid interactive scenario and proposed an approach as well as a tool based on a formal model, Linear Logic (LL), to validate these properties. Nevertheless, the application of this tool requires many special skills and so is not really suitable for normal users. In this paper, we present an authoring tool which allows modeling interactive scenarios and analysing them at the structural level using the deduction rules in LL. Thanks to our tool, normal users are able to create and validate interactive scenarios in comparison with a rich set of predefined properties/criteria of quality.

Introduction

Authoring interactive scenarios for a story requires users to create and manage a huge amount of different evolution possibilities of the story. As a consequence, flaws in this process are unavoidable. Therefore, debugging is considered as one of the most important requirements of an authoring tool (Medler and Magerko 2006).

A lot of authoring tools have been presented in Interactive Storytelling (IS) community such as Scribe (Medler and Magerko 2006), EmoEmma (Pizzi and Cavazza 2008), ENIGMA (Kriegel and Aylett 2010) (Kriegel et al. 2007), DraMachina (Donikian and Portugal 2004), Art-E-Fact (Iurgel 2004), Scenejo (Weiss et al. 2005), StoryTec (Göbel et al. 2008), *etc.* However, only some of them, Scribe, EmoEmma and ENIGMA, enable debugging. Nevertheless, the debugging mechanism of these systems is based on execution traces while unfolding a story. Concretely, they allow users to directly interact with the evolution of a discourse during each test session, provide users with means to observe and evaluate each step (execution trace) of the discourse. Thus, users can detect errors, inconsistency, *etc.* of the progress of each discourse, and correct them thanks to suggestions given by the systems. However, this mechanism is only applicable for scenarios whose discourse number is small. For "big scenarios", these tools only enable a "partial validation" because users cannot test all the possible evolutions of the story. As a consequence, the quality of the authoring process is not guaranteed.

(Collé, Champagnat, and Prigent 2005) (Dang, Champagnat, and Augeraud 2010) (Dang et al. 2011) (Dang, Champagnat, and Augeraud 2013) showed that LL (Girard 1987) allows users to model interactive scenarios and to analyse these scenarios at the structural level (analyse all the possible discourses) according to a set of predefined properties/criteria. Hence this approach (later referred to as "LL approach") enables a "total validation" of interactive scenarios, and so overcomes the foregoing weakness. Nevertheless, the works mentioned in (Collé, Champagnat, and Prigent 2005) (Dang, Champagnat, and Augeraud 2010) (Dang et al. 2011) (Dang, Champagnat, and Augeraud 2013) require users to be specialists of LL. Besides, they do not provide users with any aid (no user interface) to model scenarios. Consequently, these works do not satisfy requirements of an authoring tool (Medler and Magerko 2006).

In this paper, we present an authoring tool using the LL approach thanks to which users, who need not know LL, are able to create valid interactive scenarios. In addition, our authoring tool provides also much more analysis information on the modeled scenarios in comparison with the works in (Collé, Champagnat, and Prigent 2005) (Dang, Champagnat, and Augeraud 2010) (Dang et al. 2011) (Dang, Champagnat, and Augeraud 2013), therefore the validation power of our tool is considerably reinforced.

Before mentioning the principal content of the paper, we define some notions used in our approach:

- A *discourse* is an ordered sequence of events/actions that is a possible unfolding of a story. Therefore a same story can generate various discourses. This

consists in scheduling the events/actions corresponding with the story.

- A *scenario* is a set of all the possible discourses for a story. If we change something in the story, then we will receive a new scenario. In our approach, a scenario is considered as "valid" if it satisfies authors' objectives such as: there is no deadlock, no unused modeling element; the scenario is complex enough; *etc.* (we will present in detail these notions in the next sections of the paper).

Related Work

In the previous section, we have presented some works related to ours. This section deals with other works that focus on scenario validation aspects.

(Dang, Champagnat, and Augeraud 2011) describes an authoring tool using the LL approach, which is similar to our tool. However, this work only enables users to create scenarios without deadlocks. That is not enough for the notion of "valid scenario" in our approach (where such a scenario has to satisfy many more criteria: no deadlock, no unused modeling elements, not too simple/short, structuralized, *etc.*).

KANAL (Kim and Blythe 2003) is an authoring tool in order to produce sound plans. Indeed, it allows users to simulate a plan, to check for a variety of errors and to receive an overview of the steps or of the timelines of objects in the plan. Nevertheless, KANAL was designed only for some specific domains such as process models in biology or plans in military applications, so it is not really suitable for authoring valid interactive scenarios in IS domain.

(Vega and Natkin 2003) (Brom and Abonyi 2006) (Bosser et al. 2011) present approaches or tools based on the scenario analysis at the structural level using Petri nets (Murata 1989) and LL. The weakness of these works is: (i) they do not satisfy requirements of an authoring tool (like (Collé, Champagnat, and Prigent 2005) (Dang, Champagnat, and Augeraud 2010) (Dang et al. 2011) (Dang, Champagnat, and Augeraud 2013)); and (ii) the validity of the created scenarios is not enough for the notion of "valid scenario" in our approach (like (Dang, Champagnat, and Augeraud 2011)).

Thus, the current state of the art shows that an authoring tool, designed for normal users, enabling a scenario analysis at the structural level with much important information, is necessary. In this paper, we present such a tool using the LL approach (this is also the main contribution of the paper).

Authoring Tool and Process of Valid Interactive Scenarios

Our objective is to build an authoring tool thanks to which users, who are not specialists of LL, can apply the deduction rules in LL, to produce valid interactive scenarios. To this purpose, our authoring tool must satisfy the following requirements:

(1) provide users with a expressive graphical language in order to model a scenario of a story even when they do not have any knowledge of LL;
(2) supply users with an appropriate graphical language to setup parameters for the modeled scenario such as: minimum length of the discourses in the scenario, minimum number of discourses in the scenario, *etc.* (they will be used in the scenario analysis process);
(3) allow the automatic execution of two tasks:
    - build the graph representing all the possible discourses in the modeled scenario by means of the mechanism of sequent calculus (Indrzejczak 1998);
    - analyse at the structural level the received graph, and thereby give important information on the scenario such as: unused modeling elements, (un-)successful discourses, events/actions in each discourses, available states of each discourse after each step, possible causes of the detected errors, *etc.* This information and the foregoing graph of discourses will help users evaluate the validity of the current scenario as well as correct/improve it.

In order to satisfy the requirements raised above, we have: (i) proposed a metamodel which enables users, who need not know LL, to model interactive scenarios; (ii) proposed a metamodel allowing users to setup the parameters of the modeled scenario; (iii) developed graphical modules that implement the two foregoing metamodels; and (iv) developed an analysis module which executes automatically the two tasks mentioned in the third requirement.

In the next sections, we present in detail the data models used in our tool, the modules of the tool as well as the authoring process that users can apply to create a valid interactive scenario of a story.

Data Models Used in the Tool

Interactive Scenario Metamodel

Figure 1 gives the metamodel which enables users (non-specialists of LL) to model interactive scenarios. A *Scenario* is composed of three lists of *State*, *EventAction* and *Goal*. Each instance of these elements in the metamodel is distinguished by a *Name* (obligatory). An instance may be described in more details by a *Description*.

- A state represents a certain aspect of a component (characters, objects, *etc.*) of a story. It possesses an attribute to show whether or not it is available at the current moment of the unfolding of the story (*IsAvailableState* = *True/False*, its default value is *False*).

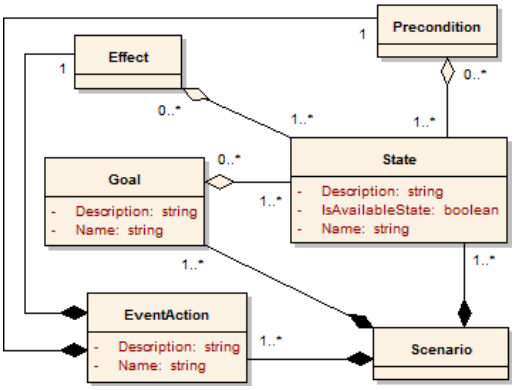

Figure 1. *Interactive Scenario Metamodel.*

- An event/action allows unfolding a story (modifying the state of the components in the story). It includes a *Precondition* and an *Effect* which contain a set of states. An event/action is executable if the states in its precondition are available. After the execution of an event/action, the states in its precondition are replaced by the ones in its effect.
- A scenario of a story may have many *Goals*. A goal is an authors' desired ending, it is composed of one or multiple state(s). An unfolding of a story (a discourse) reaches a goal if the available states of the discourse include the ones of this goal. In our approach, a discourse is considered as "successful" if it reaches one of the goals of the story, on the contrary, it is considered as "unsuccessful" (and so this discourse is a deadlock in the scenario).

Metamodel of Parameters

After modeling a scenario of a story according to the metamodel given in Figure 1, users can setup parameters of the scenario. These parameters describe the constraints that the scenario has to satisfy (they will be verified in the scenario analysis process). A scenario may have the following parameters (see Figure 2):

- Complexity: This parameter represents the complexity of the scenario, which contains two aspects:
  - DiscourseNumberInScenario: this is the minimum number of discourses in the scenario, it guarantees the richness of the scenario;
  - EventActionNumberInDiscourse: this is the minimum number of events/actions in the discourses of the scenario (minimum length of the discourses), it assures that the discourses are not terminated too fast.
- Non-OrderedEventsActions: This is a non-ordered set of key-events/actions that all the discourses in the scenario have to pass.

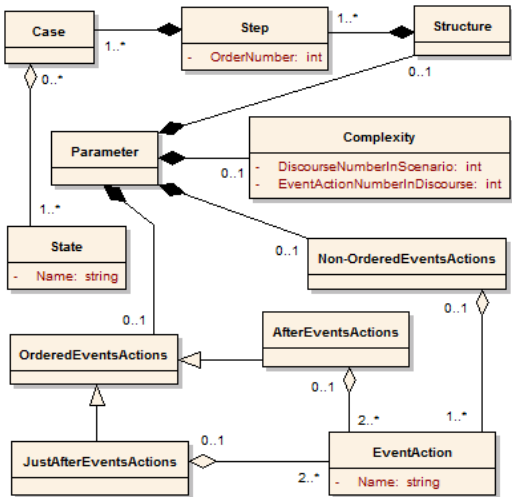

Figure 2. *Metamodel of Parameters.*

- OrderedEventsActions: This is an ordered set of key-events/actions that all the discourses in the scenario have to include (in order to guarantee the coherence of the discourses). There are two types:
  - JustAfterEventsActions: the events/actions in the set have to be consecutively executed;
  - AfterEventsActions: the events/actions in the set need not be consecutively executed (between them, maybe there exist other events/actions).
- Structure: The structure of a scenario is an ordered set of crucial *Steps* in the unfolding of the story where all the discourses in the scenario have to pass. The order of these steps is specified by *OrderNumber*. Each step may consist of some distinct *Case(s)*, a *Case* is composed of one or multiple *State(s)*. A discourse is considered as "pass a step" if it passes a case in this step. A discourse is considered as "pass a case" if the list of available states of the discourse contains all the states of this case. For example, we can define the structure of a scenario of the "Little Red Riding Hood (Little Red Cap - LRC)" story (Ashliman 2011) as an ordered set of the four following steps:
  - Step 1 (*OrderNumber* = 1): This step consists of one state: La - LRC is asked to take a piece of cake and a bottle of wine to the grandmother.
  - Step 2 (*OrderNumber* = 2): This step consists of two distinct cases:
    - Case 1: it is composed of two states: Lm - LRC meets the wolf in the woods, and Ls - LRC says to the wolf where the grandmother's house is;
    - Case 2: it is composed of two states: Lm - LRC meets the wolf in the woods, and Lns -

LRC does not say to the wolf where the grandmother's house is.

- Step 3 (*OrderNumber* = 3): This step consists of two states: Ge - The grandmother is eaten by the wolf, and Le - LRC is eaten by the wolf.
- Step 4 (*OrderNumber* = 4): This step consists of three states: Wd - The wolf is dead, Gal - The grandmother is alive, and Lal - LRC is alive.

Thus, all the discourses in this scenario have to pass in turn the four foregoing steps (for Step 2, the discourses need pass either Case 1 or Case 2).

Modules in the Tool and Authoring Process of a Valid Interactive Scenario

Our authoring tool is composed of the following modules:

- **Editor of scenario:** This module (see Figure 5) allows users to graphically model a story according to the scenario metamodel given in Figure 1. To this purpose, we have built the editor of scenario as a GMF (Graphical Modeling Framework) project (http://www.eclipse.org/modeling/gmp), where the metamodel in Figure 1 is used as its domain model (*ecore* model). As a consequence, we can model a scenario of a story by simple "drag and drop" manipulations (drag and drop the available graphical elements in the editor in order to create the modeling diagrams corresponding to the three lists of *State*, *EventAction* and *Goal*).
- **Setup module of parameters:** This is a graphical interface (see Figure 3) which helps users setup the parameters of the modeled scenario according to the metamodel given in Figure 2.

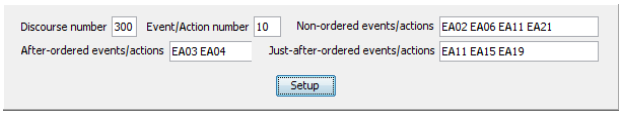

Figure 3. *A Part of the Setup Module of Parameters.*

- **Analysis module:** The aim of this component is to analyse the validity of the modeled scenario (in comparison with authors' objectives) and to suggest possible causes of the detected errors. To that purpose, it executes automatically the following tasks:
  - From the modeling diagrams in the editor, the analysis module creates a LL sequent representing the corresponding scenario. A sequent is composed of two parts (separated by $\vdash$): Left part and Right part.
    - The left part includes the available states and the events/actions in the modeled scenario, these components are connected by the connective $\otimes$. An event/action is represented by a formula $\multimap$: the left (right) side of $\multimap$ corresponds to the precondition (effect) of the event/action, the states in the precondition (effect) of the event/action are connected by the connective $\otimes$.
    - The right part includes the goals of the scenario: the goals are connected by the connective $\oplus$, the states in a goal are connected by the connective $\otimes$.
  - The analysis module proves the received sequent by applying the deduction rules in LL, in order to build the graph representing all the possible discourses of the modeled scenario (where a discourse corresponds to a proof of the sequent – see Figure 6).
  - Finally, the analysis module analyses the graph of discourses (taking into account the setup parameters) and gives users the following analysis information on the scenario:
    - Are infinite loops existing in the scenario?
    - Number, percentage, and list of unused elements (states, events/actions, goals): They allow users to detect errors in the modeling.
    - Number, percentage, and list of (un-)successful discourses: They allow users to find deadlocks in the scenario.
    - Number of discourses in the scenario: This information allows validating the parameter *DiscourseNumberInScenario*.
    - Number, percentage, and list of discourses which satisfy (do not satisfy) a certain length: They allow validating the parameter *EventActionNumberInDiscourse*.
    - Number and list of the longest/shortest (un-)successful discourses: This information and the one on the discourses, which satisfy (do not satisfy) a certain length, provide users with an impression about the possible length of the discourses. Besides, they provide also users with some suggestions for the correction of the scenario:
      - ✓ the shortest successful discourses, which do not satisfy a certain length, are used to look for unintended shortcuts;
      - ✓ the shortest unsuccessful discourses may be a "promising point of departure" to deal with deadlocks;
      - ✓ the longest unsuccessful discourses (and/or the unsuccessful discourses which satisfy a certain length) suggest that the events/actions at the end of these discourses may be the cause of the deadlocks.
    - List of events/actions may be the cause of the unsuccessful discourses: This list is

established from the analysis of the appearance frequency and of the appearance position of the events/actions in the unsuccessful discourses.

- Number, percentage, and list of discourses which contain (do not contain) a non-ordered set of key-events/actions: They allow validating the parameter *Non-OrderedEventsActions*.
- Number, percentage, and list of discourses which respect (do not respect) the structure of the scenario (validate the parameter *Structure*).
- Number, percentage, and list of discourses which contain (do not contain) an ordered set of (in-)consecutive key-events/actions: They allow validating the parameters *After/JustAfter-OrderedEventsActions*.
- Number, percentage, and list of successful discourses for each goal: They allow users to know which discourse leads to which goal and the probability of reachability of each goal (so users can reasonably distribute the probability of success of the goals in the scenario).

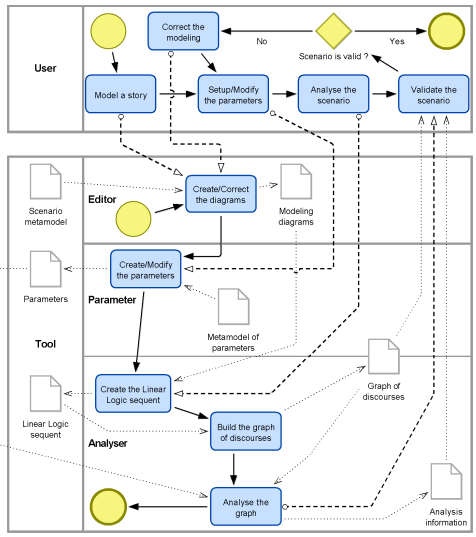

Figure 4. *Schema BPMN (http://www.bpmn.org) Describing the Authoring Process of a Valid Interactive Scenario.*

Thus, the graph of discourses and the information produced by the analysis module enable users to objectively evaluate the validity of the modeled scenario in comparison with their aims. If the scenario is not valid yet, users are able to correct/improve the modeling diagrams (thanks to the suggestions provided by the tool), modify the parameters if necessary, and reanalyse the scenario. These steps are repeated until users obtain a scenario that is the most appropriate for their objectives. Figure 4 describes in detail the authoring process of a valid interactive scenario using our tool.

Example

We give, in this section, an example in order to illustrate the authoring tool and process of a valid interactive scenario. It is an extract of an educational game which warns of domestic electrical accidents whose objective consists in causing an electric shock for the player (Dang, Champagnat, and Augeraud 2010). To this purpose, the game designer (author) anticipates that the player, from her/his initial position, will go to the kitchen or to the bathroom, where the IS controller will start the strategy of causing the electric shock for her/him, via appliances there such as a fridge, a microwave oven, a hair-dryer,…

We model this game by means of the editor of scenario as follows:

- Model the list of states (its corresponding diagram of states is given in Figure 5)

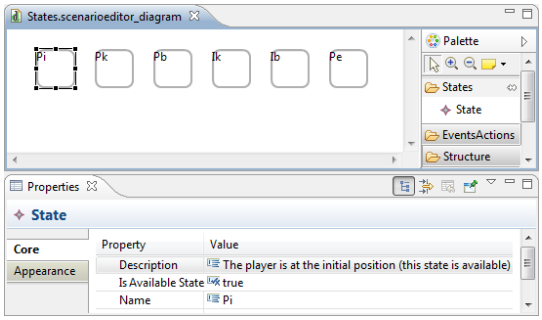

Figure 5. *Diagram of States Built in the Editor of Scenario (the Data in the "Properties" Section Describe the State "Pi").*

Name	Is Available State	Description
Pi	True	The player is at the initial position (this state is available)
Pk	False	The player is in the kitchen
Pb	False	The player is in the bathroom
Ik	False	The IS controller starts the strategy of causing the electric shock for the player in the kitchen
Ib	False	The IS controller starts the strategy of causing the electric shock for the player in the bathroom
Pe	False	The player has got the electric shock

- Model the list of events/actions

Name	Description	Pre condition	Effect
EA1	The player goes to the	Pi	Pk

	kitchen		
EA2	The player goes to the bathroom	Pi	Pb
EA3	The IS controller starts the strategy of causing the electric shock for the player in the kitchen	Pk	Pk, Ik
EA4	The player gets the electric shock in the kitchen	Pk, Ik	Pe
EA5	The player gets the electric shock in the bathroom	Pb, Ib	Pe

- Model the list of goals: This game has only one goal

Name	Description	State
G	The player gets the electric shock	Pe

Because the used example is a very small and simple extract, it does not have any parameter. Thus, after the modeling step thanks to the editor of scenario, the analysis module in our authoring tool executes automatically the following tasks:

- Create the LL sequent corresponding to the modeled scenario: Pi, EA1, EA2, EA3, EA4, EA5 ⊢ Pe, or more concretely: Pi ⊗ (Pi ⊸ Pk) ⊗ (Pi ⊸ Pb) ⊗ (Pk ⊸ Pk ⊗ Ik) ⊗ (Pk ⊗ Ik ⊸ Pe) ⊗ (Pb ⊗ Ib ⊸ Pe) ⊢ Pe.
- Build the graph representing all the possible discourses in the scenario (see Figure 6)

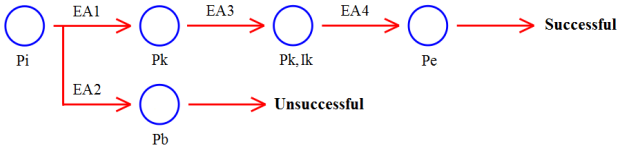

Figure 6. *Graph of Discourses Before the Validation.*

- Analyse the graph of discourses: Thanks to this analysis process, we receive the following information on the current scenario (because of the simpleness of the used example, we do not mention here all the information provided by the analysis module):
  - There is not any loop in the scenario
  - There are 2 discourses
    - Successful (50%): EA1 ⟶ EA3 ⟶ EA4
    - Unsuccessful (50%): EA2
  - There is 1 unused state (17%): Ib
  - There is 1 unused event/action (20%): EA5

Thus, the graph of discourses and the analysis information show that the current scenario is not valid yet: there is one deadlock if the player goes to the bathroom. The unused state (Ib) and the unused event/action (EA5) suggest that there is one flaw in the modeling: lack an event/action allowing the IS controller to start the strategy of causing the electric shock for the player in the bathroom. Therefore, we correct the modeling by adding the following event/action to the list of events/actions:

Name	Description	Pre condition	Effect
EA6	The IS controller starts the strategy of causing the electric shock for the player in the bathroom	Pb	Pb, Ib

The graph of discourses corresponding to the corrected scenario is given in Figure 7. This graph and the information provided by the analysis module show that the scenario after the correction is valid.

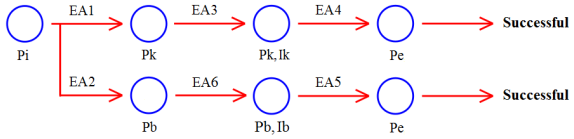

Figure 7. *Graph of Discourses After the Validation.*

Conclusion

In this paper, we have presented an authoring tool, which is designed for non-specialists of LL and enables a scenario analysis at the structural level using the deduction rules in LL. Thanks to our tool, users can create and validate interactive scenarios in comparison with a rich set of predefined properties/criteria. We have also given an example to illustrate the tool as well as the authoring process of valid interactive scenarios in our approach.

Concerning future work to improve the tool, a number of issues should be settled. Firstly, user studies should be scheduled: is our tool useful?; what are the difficulties when applying it?; what points should be changed?; *etc.*

Secondly, we have to deal with the combinatorial explosion problem while looking over all the possible discourses of a scenario. To this purpose, an analysis of the shape of the graph of discourses to reduce processing space is necessary: eliminate symmetric branches, divide the graph into several appropriate parts and only look for deadlocks separately in each part, *etc.*

Thirdly, in addition to the foregoing analysis information provided by our tool, the evaluation and the validation of the "interestingness" of a scenario, especially game scenarios, will be a promising contribution. We have proposed some properties related to this problem such as: relevance of a scenario, difficult level of a scenario, player's emotional states, *etc.* The measurement of these properties will be added to our authoring tool in future.

References

Ashliman, D. L. 2011. Little Red Riding Hood and other tales of Aarne-Thompson-Uther type 333 (last accessed August 20, 2013), http://www.pitt.edu/~dash/type0333.html#grimm.

Bosser, A.-G., Courtieu, P., Forest, J., and Cavazza, M. 2011. Structural Analysis of Narratives with the Coq Proof Assistant. In: Proceedings of ITP11 - Interactive Theorem Proving – Second International Conference, Lecture Notes in Computer Science, Springer.

Brom, C., and Abonyi, A. 2006. Petri Nets for Game Plot. In: Proc. of AISB, Vol. 3. 6–13. AISB press.

Collé, F., Champagnat, R., and Prigent, A. 2005. Scenario Analysis Based on Linear Logic. In: Advances in Computer Entertainment Technology – ACE.

Dang, K. D., Champagnat, R., and Augeraud, M. 2010. Modeling of Interactive Storytelling and Validation of Scenario by Means of Linear Logic. In: Aylett, R. et al. (eds.) ICIDS 2010. LNCS, vol. 6432, pp. 153--164. Springer, Heidelberg.

Dang, K. D., Champagnat, R., and Augeraud, M. 2011. A Methodology to Derive a Valid Scenario of an Interactive Storytelling. In Proceedings of the 8th international conference on Advances in Computer Entertainment technology – ACE 2011. Lisbon, Portugal.

Dang, K. D., Champagnat, R., and Augeraud, M. 2013. A Methodology to Validate Interactive Storytelling Scenarios in Linear Logic. In: Pan, Z. et al. (eds.) Journal *Transactions on Edutainment* – Special Issue on *Interactive Digital Storytelling*. LNCS, vol. 7775, pp. 53--82. Springer, Heidelberg.

Dang, K. D., Hoffmann, S., Champagnat, R., and Spierling, U. 2011. How Authors Benefit from Linear Logic in the Authoring Process of Interactive Storyworlds. In: Si, M. et al. (eds.) ICIDS 2011. LNCS, vol. 7069, 249-260. Springer, Heidelberg.

Donikian, S., and Portugal, J.-N. 2004. Writing Interactive Fiction Scenarii with DraMachina. In: Göbel, S. et al. (Eds.) TIDSE 2004. LNCS 3105, pp. 101-112. Springer, Heidelberg.

Girard, J.-Y. 1987. Linear Logic. Theoretical Computer Science 50(1), 1–101.

Göbel, S., Salvatore, L., Konrad, R. A., and Mehm, F. 2008. StoryTec: A Digital Storytelling Platform for the Authoring and Experiencing of Interactive and Non-linear Stories. In: Spierling, U., Szilas, N. (Eds.) ICIDS 2008. LNCS, vol. 5334, pp. 325-328. Springer, Heidelberg.

Indrzejczak, A. 1998. Jaskowski and Gentzen approaches to natural deduction and related systems. In: Kijania-Placek, K., and Wolensky, J. (Eds.), The Lvov-Warsaw School and Contemporary Philosophy, Kluwer Academic Publishers, 253--264.

Iurgel, I. 2004. From Another Point of View: Art-E-Fact. In: Göbel, S. et al. (Eds.) TIDSE 2004. LNCS 3105, pp. 26–35. Springer, Heidelberg.

Kim, J., and Blythe, J. 2003. Supporting plan authoring and analysis. In: Proceedings of the 8th international conference on Intelligent user interfaces, pp. 109-116.

Kriegel, M., and Aylett, R. 2010. Crowd-Sourced AI Authoring with ENIGMA. In: Aylett, R. et al. (Eds.) ICIDS 2010. LNCS, vol. 6432, pp. 275–278. Springer, Heidelberg.

Kriegel, M., Aylett, R., Dias, J., and Paiva, A. 2007. An Authoring Tool for an Emergent Narrative Storytelling System. In AAAI Fall, Symposium on Intelligent Narrative Technologies.

Medler, B., and Magerko, B. 2006. Scribe: A Tool for Authoring Event Driven Interactive Drama. In: Göbel, S., Malkewitz, R., Iurgel, I. (eds.) TIDSE 2006. LNCS, vol. 4326, pp. 139–150. Springer, Heidelberg.

Murata, T. 1989. Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, volume 77, number 4, pp. 541–580.

Pizzi, D., and Cavazza, M. 2008. From Debugging to Authoring: Adapting Productivity Tools to Narrative Content Description. In: Spierling, U., Szilas, N. (eds.) ICIDS 2008. LNCS, vol. 5334, pp. 285–296. Springer, Heidelberg.

Vega, L., and Natkin, S. 2003. A petri net model for the analysis of the ordering of actions in computer games. In: Proceedings of 4th annual European GAME-ON Conference. London, United Kingdom.

Weiss, S., Müller, W., Spierling, U., and Steimle, F. 2005. Scenejo – An Interactive Storytelling Platform. In: Subsol, G. (Eds.) VS 2005. LNCS, vol. 3805, pp. 77-80. Springer, Heidelberg.