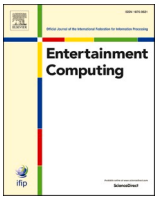
Entertainment Computing 47 (2023) 100590

Contents lists available at ScienceDirect

Entertainment Computing

journal homepage: www.elsevier.com/locate/entcom



Managing the plot structure of character-based interactive narratives in games

Edirlei Soares de Lima [a,b,*], Bruno Feijó [c], Antonio L. Furtado [c]

[a] Universidade Europeia, Faculty of Design, Technology, and Communication, Lisbon, Portugal
[b] UNIDCOM/IADE - Design and Communication Research Unit, Lisbon, Portugal
[c] Pontifical Catholic University of Rio de Janeiro, Department of Informatics, Rio de Janeiro, Brazil

ARTICLE INFO

Keywords:
Interactive storytelling
Narrative generation
Drama management
Plot structure
Automated planning

ABSTRACT

The use of narrative generation methods in games is a complex challenge that involves multiple problems of plot-based processes integrated with character-based methods. Examples of these problems are the high computational complexity of many story generation algorithms, the difficulties associated with the generation of interactive narratives that are compelling and emotionally impactful, the complex interactions among characters, and the need for tools and methods to support story writers in the process of creating and managing the narrative structure of interactive stories. In this work, we present and evaluate a new approach to generate and manage the plot structure of character-based interactive narratives in games, which combines multi-agent planning with a drama management strategy based on narrative structures. The proposed method is supported by an authoring tool that allows authors to create and test interactive narratives using graphical interfaces and intuitive diagrams. The results of our study suggest the effectiveness of our approach in generating interactive narratives for highly interactive game environments. In addition, a user study of the proposed authoring tool indicates that it can successfully support the development of character-based interactive narratives without requiring programming knowledge.

1. Introduction

Interactive storytelling is a fundamental element in games that allows players to control narratives through their own actions. Although interactive narratives are known for providing immersive and engaging experiences for players [1,2], they also represent a challenge for story writers. While linear stories allow authors to have full control over the narrative, non-linear plots require extra planning to guarantee that the intended narrative structure is achieved independently of the player's choices. In this context, most games with non-linear plots use branching structures to allow writers to manually craft every possible storyline for the game. This approach has been successful in recent years, with games such as the Mass Effect trilogy (BioWare, 2007–2012) and The Witcher trilogy (CD Projekt RED, 2007–2015) being acclaimed for their immersive and engaging interactive narratives. However, predefined branching structures can impose limitations to the story variety and can reduce the player's sense of agency [3,4].

Over the past years, research on interactive storytelling has introduced various alternatives to hand-crafted branching structures, which can be classified into plot-based, character-based, and hybrid approaches.[1] The majority of previous works on the implementation of interactive storytelling techniques in games are based on plot-based approaches [5,6,7]. However, for certain game genres, such as Role-Playing Games (RPGs), modeling the world as a simulation is more appropriate and offers some benefits over strict plot-based models. These benefits include the ability to support multiple parallel storylines that can intersect with each other, increasing the number of potential stories and providing players with more possibilities for interaction [8].

In character-based approaches, narratives emerge from the behaviors of autonomous agents that inhabit a virtual world. Although this model favors story diversity, it can also lead the plot to unexpected situations that violate the narrative structure, or to stories that are not complex enough to create an interesting drama. In this context, managing the ongoing plot to ensure a certain narrative structure is a crucial step to lead players through a dramatically engaging sequence of events. We use the term "narrative structure" to refer to the order and manner

* Corresponding author at: Universidade Europeia, Faculty of Design, Technology, and Communication, Lisbon, Portugal.
E-mail address: edirlei.lima@universidadeeuropeia.pt (E.S. de Lima).

[1] Detailed surveys on narrative generation methods can be found in [12][13].

https://doi.org/10.1016/j.entcom.2023.100590
Received 10 April 2023; Received in revised form 16 June 2023; Accepted 19 June 2023
Available online 26 June 2023

along which a plot evolves. Narrative structures, such as the Hero's Journey [9] and the Grail Hero [10], work as templates for the development of compelling narratives, which have been identified in the plot of many successful games [11].

This article presents a new approach to assisting story writers in the process of managing the plot structure of character-based interactive narratives. The human author starts by defining the narrative structure (e.g., a story arc based on the Hero's Journey pattern) and creating rules to guide the actions of the story characters. By using our method to detect situations that compromise the narrative, the author can create new rules and facts to solve the conflicts and maintain the narrative structure. The proposed method uses event traces extracted from pre-generated plots to identify event patterns that can interfere with the structure of the narrative. These patterns are then used in real-time to identify situations that violate the very structure of the ongoing plot. In this process, the system triggers additional characters' goals and performs all the necessary modifications of the world state to guide the story towards the expected narrative structure. In addition, to support the authoring process, this article also proposes and evaluates an authoring tool that enables users to design and test interactive narratives using graphical interfaces and intuitive diagrams.

This work is an extension of our paper "A Character-based Model for Interactive Storytelling in Games", published in the 21st Brazilian Symposium on Computer Games and Digital Entertainment (SBGames 2022), where we proposed the original character-based model for interactive storytelling in games [14]. In the present work, we extend the original model to support the use of narrative structures to guide the narrative generation process. Our new approach resulted in a novel method to solve inconsistencies in the plot structure of ongoing stories (see Section 4) and a novel authoring tool to support the process of creating character-based interactive narratives (see Section 5). This article also presents the results of new technical tests to evaluate the performance of our method and the results of a user study conducted to evaluate the usability of the authoring tool (see Section 6).

The article is organized as follows. Section 2 reviews related work. Section 3 presents the architecture of our character-based model. Section 4 describes the proposed method to manage the plot structure of character-based narratives. Section 5 presents the authoring tool developed to assist authors in the process of creating interactive narratives using our model. Section 6 describes the experiments conducted to evaluate the proposed method. Finally, concluding remarks are presented in Section 7.

2. Related work

There are multiple strategies to handle plot structures in character-based narratives. One of the earliest systems to include a module to manage the structure of the narrative was proposed by Riedl and Stern [15], using a drama manager to give plot-based guidance to autonomous characters. Their method to ensure a basic structure for the narrative is embedded into the planning specification of the story, where all possible inconsistencies that can arise during the simulation are precalculated and solved using alternative plans. A similar approach is explored by Aylett et al. [16] using concepts borrowed from tabletop RPGs, such as the game master role. Another solution is explored by Cai et al. [17], who presented a system with multiple agents that share the responsibility of generating interactive stories: a scriptwriter agent generates non-linear plots using a Goal Net model; a director agent selects the plot scenes to tell using Fuzzy Cognitive Maps; and virtual actor agents perform the actions requested by the director. Their approach to model the plot structure is based on the use of an agent modeling tool, where the author can manually define temporal causal relationships between scenes and characters' behaviors.

Recent research on character-based interactive storytelling has been focusing on different strategies to handle plot generation. Porteous and Lindsay [18] present a non-cooperative multi-agent planning system for narrative generation where an antagonist obstructs the protagonist in recoverable ways, allowing the protagonist to eventually achieve his/her goal. As a general counter planning approach, their method combines plan generation, goal recognition, landmark identification, and replanning. The automated interactions between protagonist and antagonist remove the need for authored narrative structuring information, but it is limited to narratives that revolve around the antagonist's obstructions to the protagonist's plans. In another research work, Porteous et al. [19] propose a narrative generation approach based on the concept of point of view, which uses a character's perspective on the story to create narrative variants. A particular feature of interest of their system is its ability to use constraints to help control the content of the generated plots. The constraints are defined as partially ordered predicates that establish key components that are expected in the plot of well-formed narratives. Although the concept of constraints resembles the way we handle narrative structures in our system, Porteous et al. [19] apply constraints directly in the planning algorithm as goals of subproblems. Our solution operates on a higher level by validating the narrative structure according to the events that occur in the plot, providing an accessible way for the author to define the desired narrative structure and alternative solutions.

Interactive storytelling and narrative generation techniques also have been explored in the context of games. Goal Oriented Action Planning (GOAP) [20] and the Mimesis architecture [21], which apply concepts commonly used in planning-based narrative generation systems, have been applied to implement the artificial intelligence of non-player characters in games [22]. A recent work by Boeda [23] describes a multi-agent cooperation system for games based on GOAP and a messaging system, which allows non-player characters to communicate and cooperate with other agents to achieve goals. Although agents' actions and goals can be modeled to represent narratives, the Boeda [23] system does not take into account narrative structures or any form of drama management.

Most of the recent research on the application of interactive storytelling techniques in games has been focused on pure plot-based approaches. Breault et al. [24] propose a quest generation engine for games that uses a deterministic planning algorithm to generate linear quests that are consistent with a given world description and characters' preferences. A more dynamic approach is explored by Lima et al. [6,25], which combines planning, execution, and monitoring to handle nondeterministic events and alternative outcomes using a hierarchical structure of alternative goals. A similar hierarchical formalism is explored in the work presented by Yacoub et al. [26], which applies Discrete-Event System Specifications (DEVS) to model alternative subplots that can be used to replace equivalent parts of the story when a certain path cannot be reached. These approaches can generate coherent quests, but no procedure to ensure the narrative structure is employed.

In a more recent work, Lima et al. [7,27] present a quest generation method based on planning and genetic algorithms, which uses story arcs to guide the quest generation process according to a specific narrative structure. Their work was also extended to support the adaptation of existing quests according to user-specified story arcs [28]. Story arcs are useful to guide computational emergent narratives towards a general dramatic structure, but they are limited to determining the expected tension for each narrative phase. When authors need more control over the generated stories, a more expressive way to establish the narrative structure is required.

Petri Nets are also commonly applied to simulate and control the plot of interactive stories. In this context, Riedl et al. [29] employs a distinct variant of Petri net, known as a colored Petri net, to allow authors to manually construct interactive stories. Their Petri nets are executed by an algorithm that continuously monitors the current state of the network to identify situations where player actions violate the integrity of the plot. The violations are then solved by applying a planning algorithm to generate new sequences of events that reinstate the coherence of the Petri net. Another example of work that employs manually authored

Petri nets to represent stories is described by Balas et al. [30], which uses hierarchical Petri nets to define branching narratives for games. A similar approach is explored by El-Sattar [31,32], who uses Petri nets as a state-based model to design interactive narratives using linear logic and reachability analysis, which is combined with a character engine that enhances the modeling of character behavior based on filmmaking theory. A different strategy is explored by Lima et al. [33], who present a method to derive Petri nets from story domains specified as situation calculus schemas, which allows the use of planning algorithm to check if the specification covers the desired story situations. However, these approaches rely solely on the modeling abilities of the author to establish a coherent narrative structure, without considering variations that can occur as results of user interactions.

3. Character-based model for games

3.1. Basic definitions

In this article, the *narrative* of a game is defined as a sequence of *events* that include actions carried out by non-player characters, such as giving items, communicating, and kidnapping, as well as tasks assigned to the player, such as collecting items, rescuing characters, and fighting enemies. While non-player character's actions are executed by the system, player-assigned tasks are performed by the player. However, both types of events are treated equally for the narrative generation process. As a result, the term *action* is also used in this article to describe the tasks assigned to the player. When an action is performed by a non-player character or by the player, the action is added to plot as an *event* (i.e., we used the term *event* to refer to executed actions).

The narrative of the game is modeled as a simulation that includes a set of characters $\{CH_1, CH_2, \ldots CH_n\}$. Every *character* is represented by a triple $CH_i = (m_i, G_i, P_i)$, where m_i is the name of the character, G_i refers to set of *goals* $\{g_1, g_2, \ldots, g_n\}$, and P_i represents a sequence of actions $\langle a_1, a_2, \ldots, a_n\rangle$ called *plan*, which leads CH_i to achieve all goals of G_i. Both g_i and a_i are represented by an atomic formula of the form $T(t_1, t_2, \ldots, t_n)$, where T defines the type of goal (e.g., *free*, *has*, *dead*) or action (e.g., *kill*, *rescue*, *get*) and the terms t_i (also called parameters) represent the elements involved in the action or goal (e.g., items, locations, characters). For example, *has*(*luke*, *master-sword*) denotes a goal that implies that *luke* (a character) must have the *master-sword* (an item). Likewise, *get*(*luke*, *master-sword*, *mystic-forest*) represents the action of *luke* obtaining the *master-sword* at *mystic-forest* (a location).

The game world is logically represented by a state, which we call *world state*. The *world state* comprises a set of positive literals (atoms) defining all characters, locations, items, and their relations. An example of *world state* is:

> *character(alice), character(crymson), location(white-castle), location (dark-tower), victim(alice), villain(crymson), free(alice), free(crymson), alive(alice), alive(crymson), at(alice, white-castle), at(crymson, dark-tower), protection(white-castle, 3), protection(dark-tower, 3), relation (alice, crymson, negative), relation(crymson, alice, negative),*

which defines two characters (*alice* and *crymson*) and two locations (*white-castle* and *dark-tower*). While *crymson* plays the role a villain, *alice* acts as the victim. Both *crymson* and *alice* are free and alive. *crymson* is at the *dark-tower* and *alice* is at the *white-castle*. The protection level of both *dark-tower* and *white-castle* is 3. The relation between *crymson* and *alice* is mutually *negative*.

An *action* is an instance of an operator, which contains restrictions for the action to occur (preconditions) and the effects of the action on the world state (postconditions). An *operator* is defined by a triple $O_i = (act_i, PRE_i, POS_i)$, where act_i is an atomic formula with variables that defines the structure of the action O_i, PRE_i is a set of positive literals that establishes the preconditions for the occurrence of O_i,[2] and POS_i is a set of positive or negative literals that defines the effects of O_i. For example, the operator for the *kidnap* action is defined by:

$O_1 = ($
$act_1 = kidnap(C1, C2, L),$
$PRE_1 = \{protagonist(C1), character(C1), character(C2),$
$location(L), villain(C1), alive(C1), alive(C2),$
$free(C2), at(C1, L), at(C2, L),$
$relation(C1, C2, negative),$
$know\text{-}unprotected(C1, C2), protection(L) < 1\},$
$POS_1 = \{\neg free(C2), kidnaped\text{-}by(C2, C1)\}$
$),$

where *C1*, *C2* and *L* are variables, *negative* is a constant, and ¬ is a negation symbol. The literal *protagonist*(*C1*) is a special type of predicate used to recognize the character expected to perform actions, which changes dynamically in the world state based on the character that is requesting a new plan.

When a character CH_i has a goal ($G \in CH_i \neq \emptyset$), a planning algorithm is used to generate a sequence of actions $P \in CH_i$ to achieve the character's goal. As characters perform their actions in the game, the narrative plot is gradually created. The simulation of multiple characters in the game adds additional dimensions to the plot structure, where each character gains its own timeline of events (Fig. 1). This dynamic structure also supports the occurrence of parallel events (two examples of possible parallel actions are illustrated in Fig. 1: a_2 in $P_1 \in CH_1$ and a_1 in $P_1 \in CH_3$; and a_1 in $P_1 \in CH_2$ and a_4 in $P_1 \in CH_3$).

The goals of characters are generated using production rules, which are triggered when specific situations are observed in the world state. Each *production rule* is represented by a pair $PRULE_i = (COND_i, GOAL_i)$,

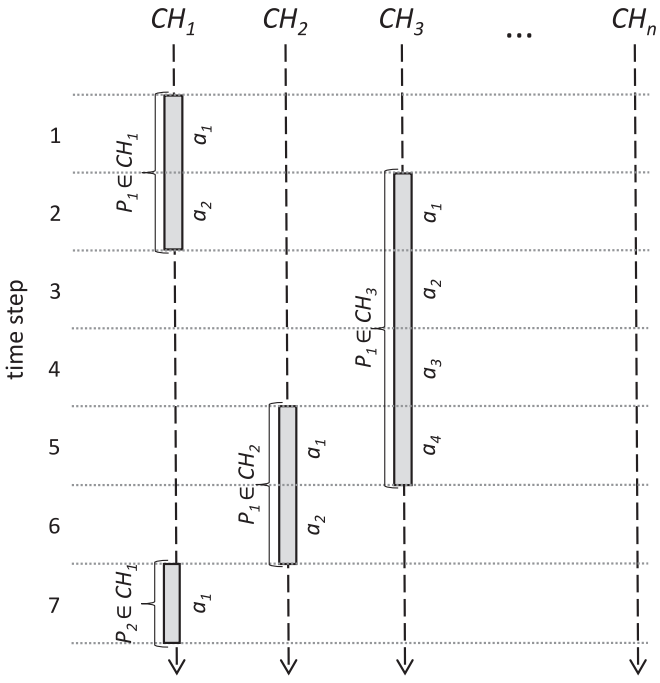


Fig. 1. The dynamic structure of the character-based plot.

[2] We make the assumption that operators' preconditions consist solely of positive literals, which is intended to enhance the computational efficiency of the planning process [34].

where $COND_i$ is a set of positive literals that defines the conditions (i.e., literals that must hold in the current world state) for the activation of $PRULE_i$, and $GOAL_i$ denotes a set of positive literals that defines the goals generated by the activation of $PRULE_i$. The author of the story defines production rules to reflect the expected behaviors of characters when particular situations occur in the game world. For example, the behavior of a character willing to inform the villain that a victim is unprotected can be defined as:

$PRULE_1$ = (
  $COND_1$ = {*protagonist*(C1), *victim*(C2), *villain*(C3),
    at(C1, L), *free*(C2), *at*(C2, L), *protection*(L) < 1,
    relation(C1, C2, *negative*),
    relation(C1, C3, *positive*),
    relation(C3, C2, *negative*), *alive*(C3)},
  $GOAL_1$ = {*know-unprotected*(C3, C2), *at*(C1, L)}
).

The game world is also controlled by general rules, which define how the world changes when certain situations occur. A *general rule* is expressed as a pair $GRULE_i$ = ($COND_i$, $EFFECT_i$), where $COND_i$ defines a set of positive literals representing the conditions for the activation of $GRULE_i$, and $EFFECT_i$ establishes a set of positive or negative literals defining how the world state is affected by $GRULE_i$ (i.e., literals to be added or removed from the current world state). While production rules operate by generating goals for specific characters, general rules affect the world state directly and independently of the characters' actions. For example, a general rule to define that any character that is not at a protected location is unprotected can be described as:

$GRULE_1$ = (
  $COND_1$ = {*at*(C, L), *protection*(L) < 1},
  $EFFECT_1$ = {¬*protected*(C), *unprotected*(C)}
).

3.2. The architecture of the character-based system

Fig. 2 illustrates the architecture of our character-based interactive storytelling system for games, which is constituted of two primary modules: (1) the *Character-based Simulation* module, responsible for controlling the simulation of character agents during the narrative generation process; and (2) the *Game Manager* module, responsible for managing the execution of characters' actions within the game, while handling the actions that are requested to the player.

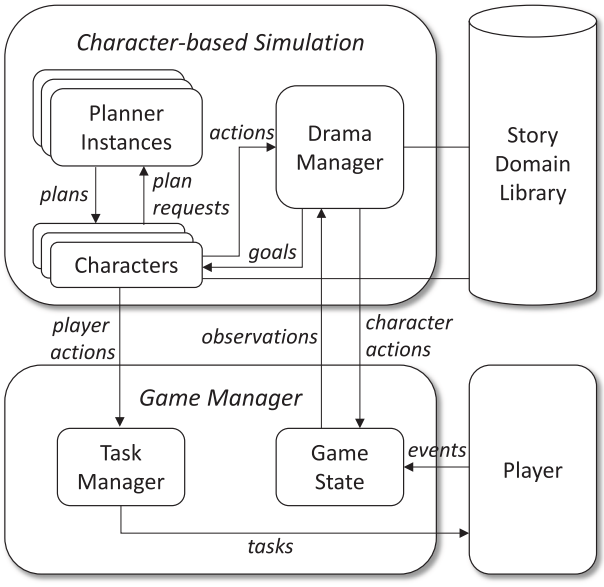


Fig. 2. Architecture of the character-based interactive storytelling system for games.

As an integral component of the *Character-based Simulation* module, a collection of *Characters* represents the simulated agents, taking on the responsibility of devising their individual plans of actions to accomplish their objectives through the aid of *Planner Instances*. In our implementation, we utilized Coles et al.'s POPF planner [35], which uses a forward-chaining state-based search strategy and is in line with our STRIPS formalism [36].

As part of the narrative generation process, the *Drama Manager* generates characters' goals by observing the world state for situations that can activate production and general rules. Although the characters are responsible for independently planning their actions to achieve their goals, their actions are executed in the same game world, which can cause inconsistencies in the plans of other characters. For example, if the plan of character CH_1 involves getting an item from a specific location but it is collected by another character before CH_1 can reach it, the plan of CH_1 will fail when executing the action *get* (the item will not be at the expected location). To ensure plot consistency, the *Drama Manager* validates the characters' actions by verifying their preconditions based on the current world state before executing them. In case of an inconsistency, a replanning procedure is performed to find an alternative plan. If no alternative plan is found, the agent aborts the current goal.

In our system, the *Drama Manager* is also responsible for making decisions about the order in which actions occur or allowing characters to perform certain actions in parallel. These decisions are made based on the actions' relevance for the narrative to avoid simultaneous occurrences of important actions, which are defined by the author of the story. Additionally, the *Drama Manager* can incorporate player choices, allowing the player to select how the events of the story unfold. While validating and handling the actions of all characters, the *Drama Manager* also maintains the ongoing list of events for the narrative plot, which allows the system to identify situations that can violate the expected structure for the narrative. The process of handling and managing the plot structure of the narrative is described in Section 4.

Both the *Drama Manager* and *Characters* have access to the *Story Domain*, which comprises the definition of operators, the initial world state, production and general rules, and the expected narrative structure. More details about the *Story Domain* are presented in Section 5.

In the *Game Manager* module of our system, the actions generated for the player character are processed by the *Task Manager*, which transforms them into game objectives presented to the player as the story progresses. As the player interacts with the game, the *Game State* is updated to reflect these actions, and is then used by the *Drama Manager* to keep track of the current world state.

4. Drama management

Although the dynamic nature of the character-based simulation adds more possibilities for story variation and user interaction when compared to a plot-based approach, the freedom provided to autonomous characters can also produce narratives that are not well-structured or complex enough to create an interesting drama. The proposed method to guide the composition of the plot is based on the use of a narrative structure defined by the author, which is employed by the system to ensure that all generated stories follow the intended structure. Because we are integrating character-based methods with plot-based processes, we strive to ensure that the plot arc (the course of the overall story) is reflected by each character arc (the path of a specific character).

The term "narrative structure" refers to the order and manner along which a plot develops. The concept was initial shaped by Gustav Freytag, a 19th century German critic and playwright that proposed a basic plot pattern inspired by the ideas of Aristotle about epic and tragedy (Freytag's Pyramid) [37]. Over the years, normative notions of dramatic structure were adopted by story writers as recipes for the development of successful stories [38,39]. A remarkable example is the three-act

structure commonly used by the film industry, which divides the story into Setup, Confrontation, and Resolution. While some narrative structures focus on the definition of general acts, such as the three-act structure, others go further and establish specific *stages* or *episodes* to describe the narrative, such as the Hero's Journey [9] and the Grail Hero [10].

The main challenges associated with the use of narrative structures in character-based interactive storytelling systems are related to the difficulty in estimating the final plot for a narrative that unfolds in response to real-time user interactions. If the full plot was known, a simple pattern matching algorithm could be applied to analyze and identify the current narrative structure. Although analyzing all possible outcomes while the narrative is being generated is a possibility, it can affect the real-time performance of the system when complex story domains are considered. Therefore, our strategy to deal with this problem consists in automatically analyzing event traces extracted from plots generated in a pre-processing phase to identify event patterns that can interfere in the structure of the narrative. In this way, the author of the story domain can specify creative solutions to recover and guide the plot structure when such event patterns are identified in the plot of an ongoing story.

4.1. Basic definitions

We define a *narrative structure* as a set of episodes $\{EP_1, EP_2, \ldots, EP_n\}$ that are expected to occur in the plot of a narrative. Each *episode* is a triple $EP_i = (name_i, PATTERNS_i, RESOLUTION_i)$, where $name_i$ is the episode's name, $PATTERNS_i$ is a set of event patterns $\{EVP_1, EVP_2, \ldots, EVP_m\}$ that prevents the occurrence of EP_i in a plot, and $RESOLUTION_i$ is a set of conflict resolution statements (world state changes and additional goals) used to recover the plot structure of a narrative that lacks EP_i.

An *event pattern* is a pair $EVP_i = (type_i, EVENTS_i)$, where $type_i$ identifies the type of pattern (single event, consecutive events, or non-consecutive events), and $EVENTS_i$ is a set of partially instantiated narrative events $\{ev_1, ev_2, \ldots, ev_k\}$. Each event ev_i is denoted by an atomic formula $T(t_1, t_2, \ldots, t_n)$, where T defines the type of event and the terms t_j represent the elements involved in the event. A term t_j can also be represented by the symbol "_", which indicates a term that matches any other term when e_i is compared to another event e_j (identical to the anonymous variable of Prolog).

A *conflict resolution statement* is a pair $RESOLUTION_i = (RSTATE_i, RGOAL_i)$, where $RSTATE_i$ is a set of positive or negative literals that define how the world state is affected by $RESOLUTION_i$, and $RGOAL_i$ is a set of pairs (ch_i, g_i), in which g_i is a set of positive literals that establish additional goals to be activated for a character ch_i when EVP_i is detected in a narrative.

For example, the episode "Failed Mediation" of the grail-based narrative structure [10], where the hero fails to complete his goal for lacking knowledge or skills, can be represented as:

```
EP1 = (
  name1 = "Failed Mediation",
  PATTERN1 = {
    EVP1 = (
      type1 = "single",
      EVENTS1 = {train(luke,_)}
  )
  },
  RESOLUTION1 = (
    RSTATE1 = {has-trap(dark-tower), ¬safe(dark-tower), reckless(luke),
      can-get-hurt(luke)},
    RGOAL1 = {(chi = "luke", gi = {stronger-than(crymson, luke)})}
  )
).
```

According to the example above, the occurrence of the "Failed Mediation" episode is directly compromised by the inclusion of a single event *train*(*luke*,_) in the plot (i.e., if *luke* trains, he will be strong enough to defeat the villain *crymson* on their first encounter; therefore, the "Failed Mediation" episode will not occur). The conflict resolution statement of this episode indicates a solution that involves adding three new facts to the world state (*has-trap*(*dark-tower*), *reckless*(*luke*), *can-get-hurt*(*luke*)) and removing the fact *safe*(*dark-tower*) (i.e., a trap is added to the *dark-tower*, which can hurt *luke* and reduce his strength). In addition, a new goal to ensure that *luke* will not be able to defeat the villain on their first encounter will be generated: *stronger-than*(*crymson*, *luke*). It is important to notice that the goal used in this example is fairly generic and does not indicate the exact solution for the situation. The dynamic nature of the simulation allows the story to develop in different ways depending on user choices or the time characters spend performing their actions. While the world state modifications create a viable solution to ensure the occurrence of the "Failed Mediation" episode, they do not indicate the exact sequence of events that will lead to the episode.

4.2. Identifying event patterns that interfere in the narrative structure

Our method combines automatic steps with human intervention to find creative solutions to problems that compromise the narrative structure of interactive stories. The first step of the proposed strategy to manage the narrative structure of an ongoing story consists in analyzing the events of pre-generated plots to identify event patterns that can interfere in the narrative structure. The event patterns identified in this step are then used by the author of the story domain to describe the episodes that are considered essential for the narrative structure, which also include the definition of conflict resolution statements to recover the plot of narratives that lack the expected episodes.

The process of identifying event patterns that can interfere in the narrative structure starts with a pre-processing phase, where all possible narrative variants for the story domain are generated and stored as sequences of events (considering all possible combinations of user interactions and the possible orders in which the events of the narrative can occur). The set of all narrative variants is defined as Ω.

In the next step of the pre-processing phase, the author of the story domain defines a sequence of events that characterize the occurrence of each episode of the desired narrative structure in a plot, which is done by analyzing the story domain and some variants from Ω. For example, the "Failed Mediation" episode of the grail-based narrative structure (described in the previous section), can have its occurrence detected in a plot by the existence of a failure event involving the hero and the villain. Considering a story domain where the hero is called *luke*, the villain is called *crymson*, and knowing that the failure event can only be represented by the operator *fail-to-defeat*(*C1*, *C2*) (meaning that character *C1* failed to defeat character *C2*), the author of the story domain can use the event *fail-to-defeat*(*luke*, *crymson*) to characterize the occurrence of the "Failed Mediation" episode in a plot.,[3][4]The sequences of events identified in this phase are associated with their respective episode names to establish the *desired episodes* Ψ, which is a set of pairs $\Psi_i = (\alpha_i, \psi_i)$, where α_i represents the episode name and ψ_i is the sequences of events that characterize the occurrence of Ψ_i in a plot.

After generating Ω and establishing Ψ, an automated process is performed to identify the event patterns that can interfere in the occurrence of the desired episodes Ψ in a plot. For each desired episode Ψ_i, the following steps are executed:

1. Identify the narrative variants of Ω where the events $\psi_i \in \Psi_i$ occur and add them to a new set Λ;

[3] As in the definition of event patterns, the symbol "_" can be used to indicate terms that match any other terms when the event is compared to another event.

[4] Although this example uses a single event to characterize an episode, sequences of consecutive or non-consecutive events are also supported.

2. Identify the narrative variants of Ω where the events $\psi_i \in \Psi_i$ do not occur and add them to a new set Θ;
3. Search for single events that occur in all narrative variants of Θ and do not occur in Λ. If single events are found, return them as candidate event patterns to identify Ψ_i in an ongoing plot;
4. Search for the shortest common sequences of consecutive events that occur in all narrative variants of Θ and do not occur in Λ. If sequences of consecutive events are found, return them as candidate event patterns to identify Ψ_i in an ongoing plot;
5. Search for the shortest common sequences of non-consecutive events that occur in all narrative variants of Θ and do not occur in Λ. If sequences of non-consecutive events are found, return them as candidate event patterns to identify Ψ_i in an ongoing plot.

When searching for event patterns, the system automatically identifies simple conflicts within the events that can be solved by replacing constant terms by the "_" symbol (indicating a term that matches any other term when comparing events).

Considering again the "Failed Mediation" episode as an example, the procedure above will identify all possible event patterns that occur in narrative variants where the episode was not identified (i.e., stories where the event *fail-to-defeat*(*luke*, *crymson*) does not occur), ensuring that they do not occur in variants where the episode was identified (i.e., stories where the event *fail-to-defeat*(*luke*, *crymson*) does occur). Although multiple patterns might be identified, the simplest patterns are prioritized to simplify their identification in the plot of ongoing narratives. In the case of the "Failed Mediation" episode, the simplest identified pattern is a single event: *train*(*luke*,_), which indicates that the event of *luke* training in any location always compromises the occurrence of the "Failed Mediation" episode in an ongoing story.

4.3. Solving conflicts and maintaining the narrative structure

After identifying event patterns for the episodes of the narrative structure, the author of the story domain must define conflict resolution statements to recover the plot of narratives that lack the expected episodes. As described in Section 4.1, the conflict resolution statements can involve world state modifications and additional goals that are activated to guide the story towards the expected narrative structure.

Defining conflict resolution statements may involve modifications in the story domain to accommodate creative solutions to the situations that violate the expected narrative structure, such as the definition of new world state predicates and new operators to support new types of events. Such modifications require a general analysis of the events that characterize the episodes and the event patterns identified as the causes of the narrative structure violation. Our system helps the author in this analysis.

Considering the "Failed Mediation" episode as an example, which is characterized by the event *fail-to-defeat*(*luke*, *crymson*) and compromised by the occurrence of the event pattern *train*(*luke*,_), the author of the story can identify a causal link for the failure by analyzing the effects of operator *train* and the preconditions of the operator *fail-to-defeat*: the train operator increases the strength of *luke* and the *fail-to-defeat* operator requires the strength of *luke* to be less than the strength of *crymson*. Therefore, the author of the story can infer that a conflict resolution statement is required to either reduce the strength of *luke* or increase the strength of *crymson*. Assuming that the author decides to solve the situation by reducing the strength of *luke*, one can consider a solution that involves the occurrence of an event where *luke* gets hurt, causing his strength to be reduced. Considering that the story domain may not be ready to produce such type of event, the inclusion of a new operator in the story domain may be needed. Example:

O_2 = (
act_2 = *get-hurt* (*C*, *L*),
PRE_2 = {*protagonist*(*C*), *character*(*C*), *alive*(*C*), *location*(*L*), *at*(*C*, *L*),
has-trap(*L*), *reckless*(*C*), *can-get-hurt*(*C*)},
POS_2 = {¬*can-get-hurt*(*C*), ¬*has-trap*(*L*), *safe*(*L*), *strength*(*C*) − 12}
).

With the possibility of reducing the strength of a character through the operator *get-hurt*, the author of the story can create a conflict resolution statement to define new goals for *luke* (guiding *luke* to perform the actions that will lead him to get hurt) and establish the necessary world state modifications to provide the conditions for the occurrence of the *get-hurt* event. As anticipated in Section 4.1, the conflict resolution statement for this example comprises the definition of the goal *stronger-than*(*crymson*, *luke*) for *luke* and four world state modifications: *has-trap* (*dark-tower*), ¬*safe*(*dark-tower*), *reckless*(*luke*), *can-get-hurt*(*luke*). In this way, a trap that can hurt *luke* will be created in the *dark-tower* and the goal will make *luke* fall into the trap to ensure that he will be weaker than *crymson* before their first confront.

The process of identifying the occurrence of an event pattern in the plot of an ongoing story is performed when a character completes the execution of an action (i.e., when a new event is added to the ongoing plot). Consider for example the following ongoing plot fragment:

> *…, go(luke, grey-palace, mist-forest), go(luke, mist-forest, desert-palace), get-item(luke, pendant-of-power, desert-palace), go(luke, desert-palace, mist-forest), draw-sword(luke, master-sword, pendant-of-wisdom, pendant-of-power, pendant-of-courage, mist-forest).*

At this point of the story, the player that controls *luke* can decide *luke's* next move: confront the villain or train to become stronger. If the player decides to train, a goal to induce *luke* to train will be generated, which will result in the inclusion of the event *train*(*luke*, *mist-forest*) into the plot. The inclusion of the new event will trigger a process to identify the occurrence of event patterns that compromise the expected narrative structure. In this case, the event pattern of the "Failed Mediation" episode (single event *train*(*luke*,_)) will match the new event *train*(*luke*, *mist-forest*). Therefore, the conflict resolution statement of the "Failed Mediation" episode will be activated, which will result in a few world state modifications (creating a trap in the *dark-tower*) and the establishment of a new goal for *luke*: *stronger-than*(*crymson*, *luke*). The simulation will then continue and, in due time, the event that characterizes the "Failed Mediation" episode will occur (*fail-to-defeat* event):

> *…, go(luke, grey-palace, mist-forest), go(luke, mist-forest, desert-palace), get-item(luke, pendant-of-power, desert-palace), go(luke, desert-palace, mist-forest), draw-sword(luke, master-sword, pendant-of-wisdom, pendant-of-power, pendant-of-courage, mist-forest), train(luke, mist-forest), go(luke, mist-forest, dark-tower), get-hurt(luke, dark-tower), failed-defeat(luke, crymson), …*

5. Authoring tool

The process of creating a character-based interactive narrative requires the specification of the *Story Domain*, which encompasses four essential components: (1) the initial world state that defines all objects that exist in the game world (characters, locations, items, etc.) and their relations; (2) the operators that establish the range of feasible actions that can be performed by virtual characters; (3) the production and general rules that are used to define the virtual characters' behaviors in various situations that may arise in the game; and (4) the basic narrative structure that is anticipated to be present in the final plot. These elements are specified in an XML format,[5] which is interpreted by our system to extract all information needed to manage the narrative

[5] An example of story domain is available at: http://www.icad.puc-rio.br/~logtell/character-based/story-domain.xml.

generation process, including the creation of the planning domain for the POPF planner.

Although writing interactive narratives directly in the XML format is possible, it requires additional programming skills, which can limit the applicability of our model in real-world situations where story writing professionals are involved in the process of creating the narrative. In order to simplify the authoring process, we designed a tool that allows users to specify all elements of the story domain through a visual interface. As illustrated in Fig. 3, the authoring tool comprises five pages (tabs) where the author can design the various elements of the story domain using intuitive diagrams.

The diagrams used to represent the elements of the story domain were designed to simplify the creation and visualization of the various elements that constitute the world where the story takes place. The diagram of the initial world state is based on the well-known entity-relationship model commonly used in database design [40], which allows the creation of a visual representation for all entities of the world and their properties and relations (an example is presented in Fig. 3). Different colors are used to identify the various types of properties/relations: gray represents static properties/relations (i.e., properties/relations that never change), green represents dynamic properties/relations (i.e., properties/relations that can change during the planning process), and light red identifies numeric properties.

Operators are created through a diagram element that separates the three fundamental components of an operator: (1) *input*, which establishes the variables that are involved in the action (parameters); (2) *preconditions*, which defines the requirements for the occurrence of the action; and (3) *effects*, which defines how the execution of the action affects the world state. Fig. 4 shows the schematic representation of the kidnap operator in the proposed diagram format.

The schematic representation to define production and general rules comprises two elements: preconditions and goals. While the *preconditions* element defines constraints for the activation of the production/general rule, the *goals* element establishes new goals for the characters involved in the activated rule. Fig. 5 shows an example of the schematic representation of a production rule that leads a character (who does not like the victim) to inform the villain when the victim is unprotected.

The narrative structure can also be specified in a schematic visual format, where each episode of the narrative structure is represented by three elements: pattern, state changes, and new goals (Fig. 6). We used the Grail-based narrative structure in the examples, but the author can

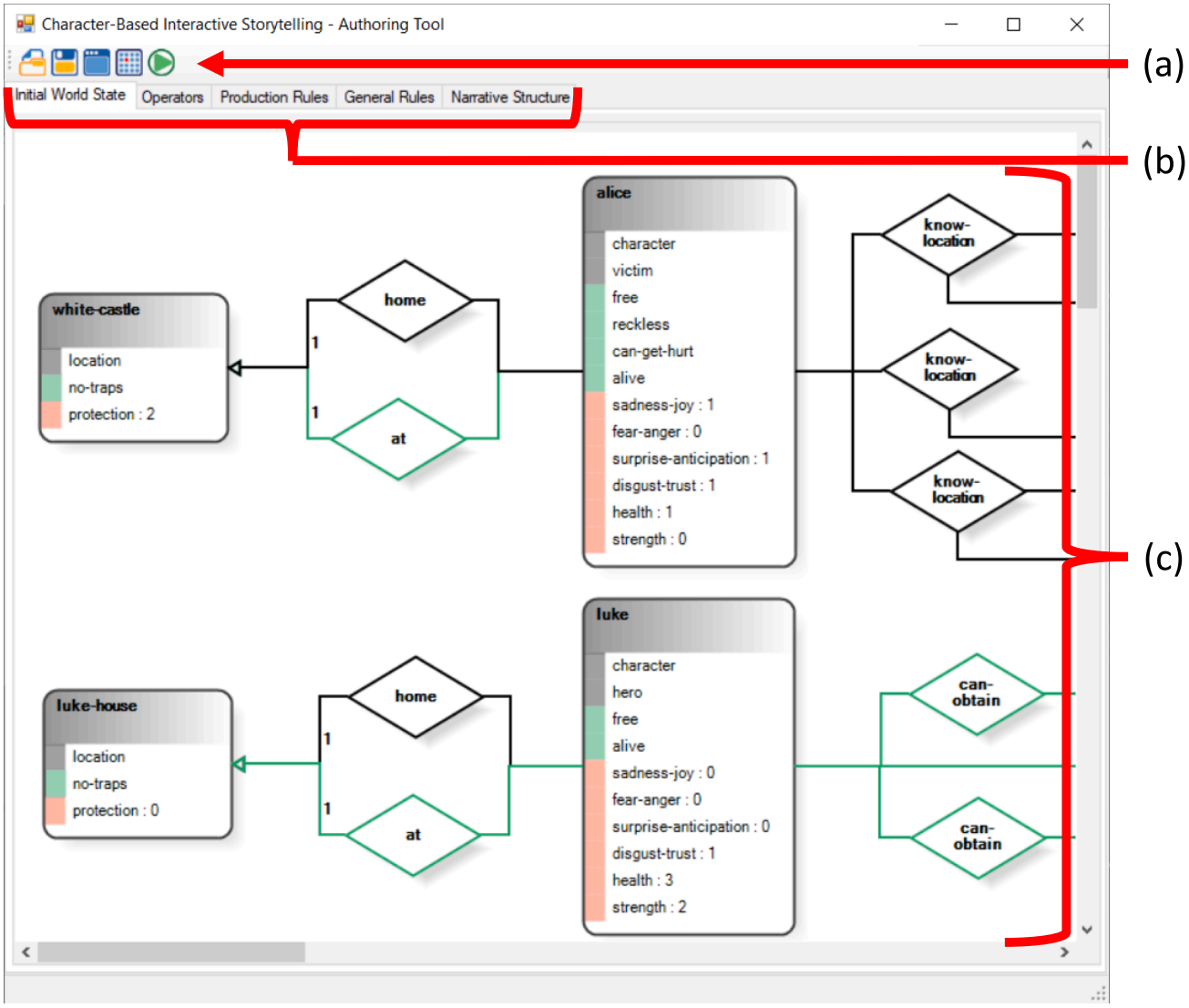


Fig. 3. Main user interface of the authoring tool, which comprises: (a) a tool bar that allows authors to open, save, change settings, identify event patterns for the narrative structure, and test the story domain; (b) tabs to access the elements of the story domain (initial world state, operators, productions rules, general rules, and narrative structure); and (c) a drawing area that allows the author to design the elements of the story domain.

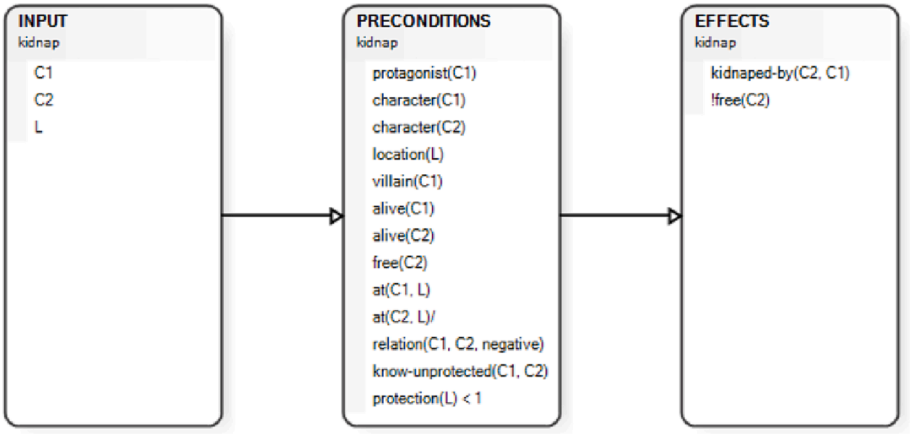


Fig. 4. Example of visual representation for the kidnap operator.

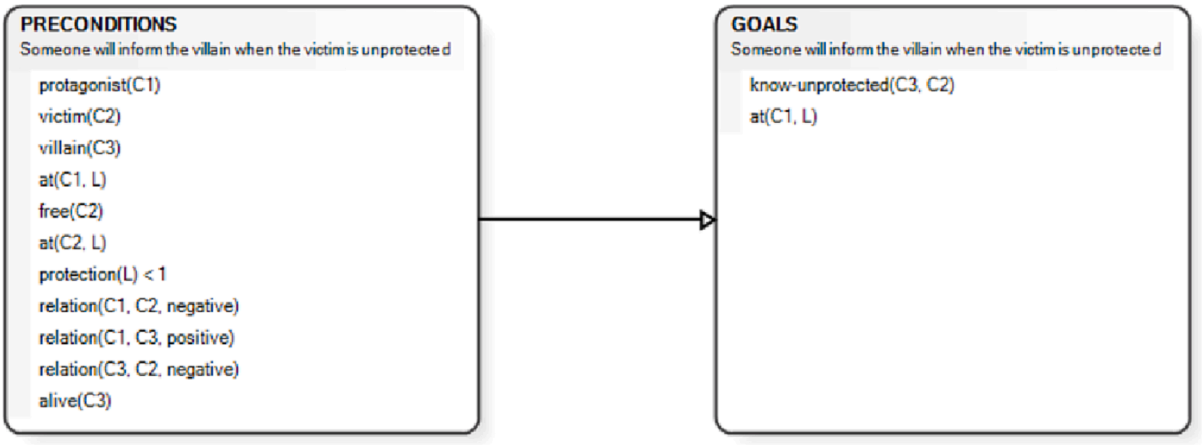


Fig. 5. Example of visual representation of a production that leads a character to inform the villain when the victim is unprotected.

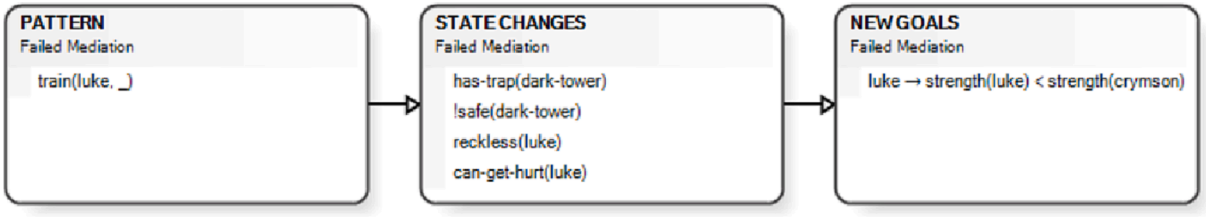


Fig. 6. Example of schematic representation for the "Failed Mediation" episode.

specify any other narrative structure. Similar to the definitions presented in Section 4.1, the *pattern* identifies the events that can compromise the expected narrative structure, the *state changes* establishes the world state modifications necessary to recover the plot structure, and the *new goals* establishes additional goals to guide characters towards the execution of the actions expected for the missing narrative episodes.

The diagrams used to represent operators, production rules, general rules and the narrative structure are created using the entities, properties and relations defined in the initial world state (i.e., the initial world state defines the vocabulary of symbols used to create the other elements of the story domain). The process of adding a new element to any diagram is performed through an intuitive interface sequence that allows users to select the options needed to create the elements. Fig. 7 illustrates the process of adding a precondition for the operator "go".

In order to assist users in the process of defining the narrative structure, the authoring tool provides access to a module that allows users to visualize all narrative variants generated for the current story domain. By analyzing the variants, users can specify the events that characterize the occurrence of the episodes of the desired narrative structure in a plot. Then, the automatic process described in Section 4.2 is performed for identifying the event patterns that can interfere in the structure of the narrative. Fig. 8 illustrates the process of using this module.

The general strategy to define the story domain for an interactive narrative requires a series of experiments to guarantee that production and general rules are sufficient to guide the autonomous characters towards interesting and meaningful situations. Therefore, to assist authors in this testing process, the authoring tool includes a plot visualization module that allows users to test the story domain by performing a step-by-step execution of the characters' actions in a timeline of events (Fig. 9).

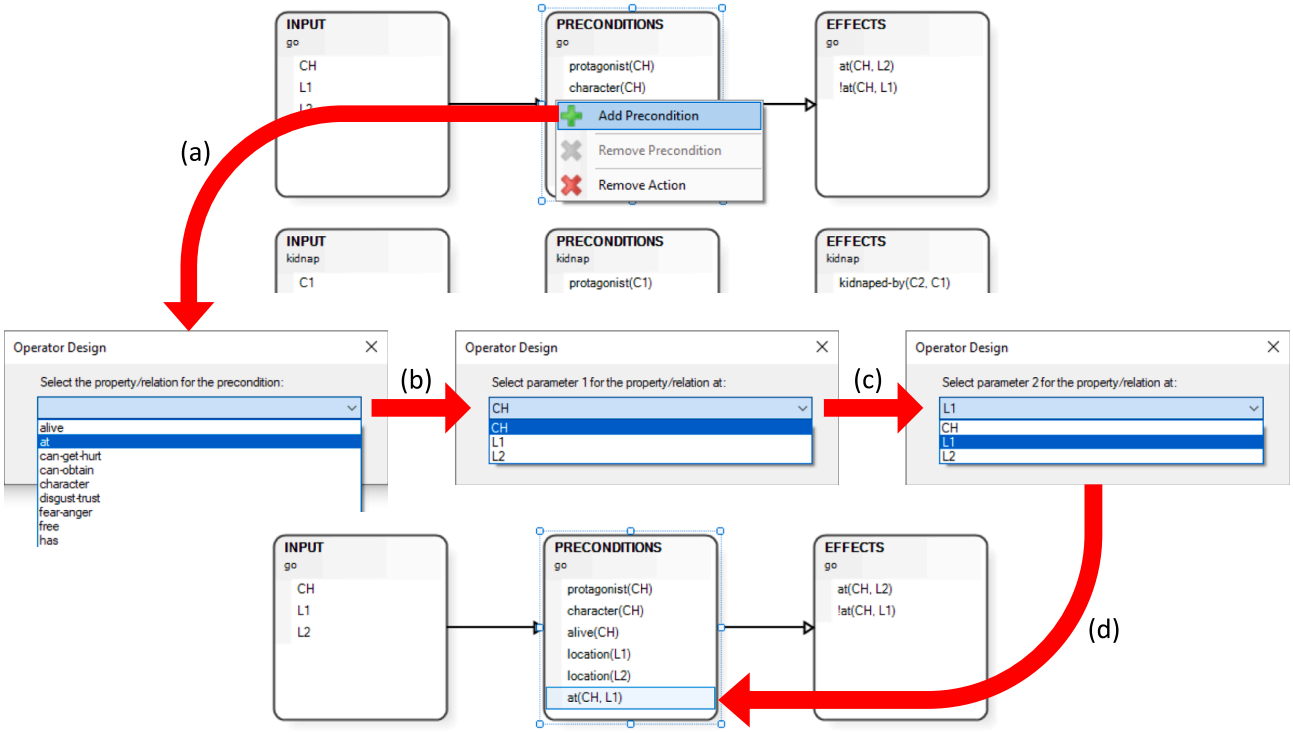


Fig. 7. Process of adding a precondition for the operator "go" in the authoring tool: (a) user selected the option to create a precondition for the operator "go"; (b) user selected the relation "at" to be used as precondition; (c) user selected the first parameter for the "at" relation; and (d) user selected the second parameter for the "at" relation, resulting in the precondition *at*(*CH*, *L1*).

The plot visualization module is designed to present the characters' actions in a table format, as illustrated in Fig. 9. The table is organized in columns that correspond to individual characters, and each column is labeled with the character's name screen (labels "Crymson", "Alice", "Luke", "Sara", and "Regan" in Fig. 9). At the bottom of each column, there is a button labeled "Act" that becomes enabled (blue) when the character has an action to perform. Above the button, there is a textual description of the character's intended action (light blue rectangle). Pressing the "Act" button causes the character to perform the action, and the textual description of the action is added to the character's column as a new row (light green rectangle). This organization of the table enables users to track the actions performed by each character throughout the simulation. In addition, by pressing the information icon (grey circle labeled "i" at the bottom left of the screen), users can visualize the current world state, which is useful for identifying conditions for new production and general rules.

An instance of the plot composition tool that allows for experimentation with the story domain used as example in this article can be accessed through the following URL: https://www.icad.puc-rio.br/~logtell/character-based/.

6. Evaluation and results

The evaluation of our method comprises three tests: (1) an applicability test to verify if our method can be effectively applied to generate and control the narrative of a game; (2) a technical test to assess the computational performance of the method; and (3) a user study to evaluate the process by creating a new interactive narrative using the proposed authoring tool.

6.1. Applicability evaluation

To assess the applicability of our character-based model, we utilized the Unity game engine to implement the prototype of a 3D RPG that uses the proposed character-based architecture for dynamically generating and controlling the narrative of the game (Fig. 10).

The game's story revolves around five primary characters: Alice, Luke, Crymson, Regan, and Sara. Alice is a charming princess that lives under strict protection and possesses knowledge about the location of powerful items that can be used for evil purposes. Luke is a brave young hero, who is in love with princess Alice. Crymson is an evil villain that seeks to gain strength through the use of the powerful items that only Alice knows where to find. Regan is an evil merchant, who is willing to provide Crymson with information for personal gain. Sara is a friendly villager that would alert Luke if anything adverse occurs to princess Alice. Furthermore, three support characters, Randall, Eric, and Abraham, are present in the game world, who provide items, challenges, and powers to the main characters. However, they are not modeled as agents in the simulation to avoid unnecessary plan calculations for characters with limited actions. The player takes on the role of the brave hero Luke and must overcome several challenges to save princess Alice.

In order to test the flexibility of our drama management method, we opted to use a specialized narrative structure in the game prototype, which is called "The Fall and Rise of the Grail Hero" [10]. This structure (here called grail-based narrative structure) is inspired on a 12th century romance of chivalry entitled *Le Conte du Graal* (*Perceval*), by the French poet Chrétien de Troyes [41], whereby the Grail literary tradition was inaugurated [10]. According to Lima et al. [44], a general grail-based narrative revolves around a protagonist that "begins as a naïve person, learns about himself along successive stages, falls down nevertheless in a crucial instant, but is then led to rise again and move towards a high position that nobody else could attain".

The grail-based narrative structure consists of nine distinct episodes: "Preparation 1", "Failed Mediation", "Apotheosis 1", "Mediation", "Errance", "Preparation 2", "Quest", "Apotheosis 2", and "Denouement" [10]. The first episode, "Preparation 1", involves the hero learning some essential skills for achieving his goals. The second episode, "Failed Mediation", occurs when the hero's current abilities prove to be

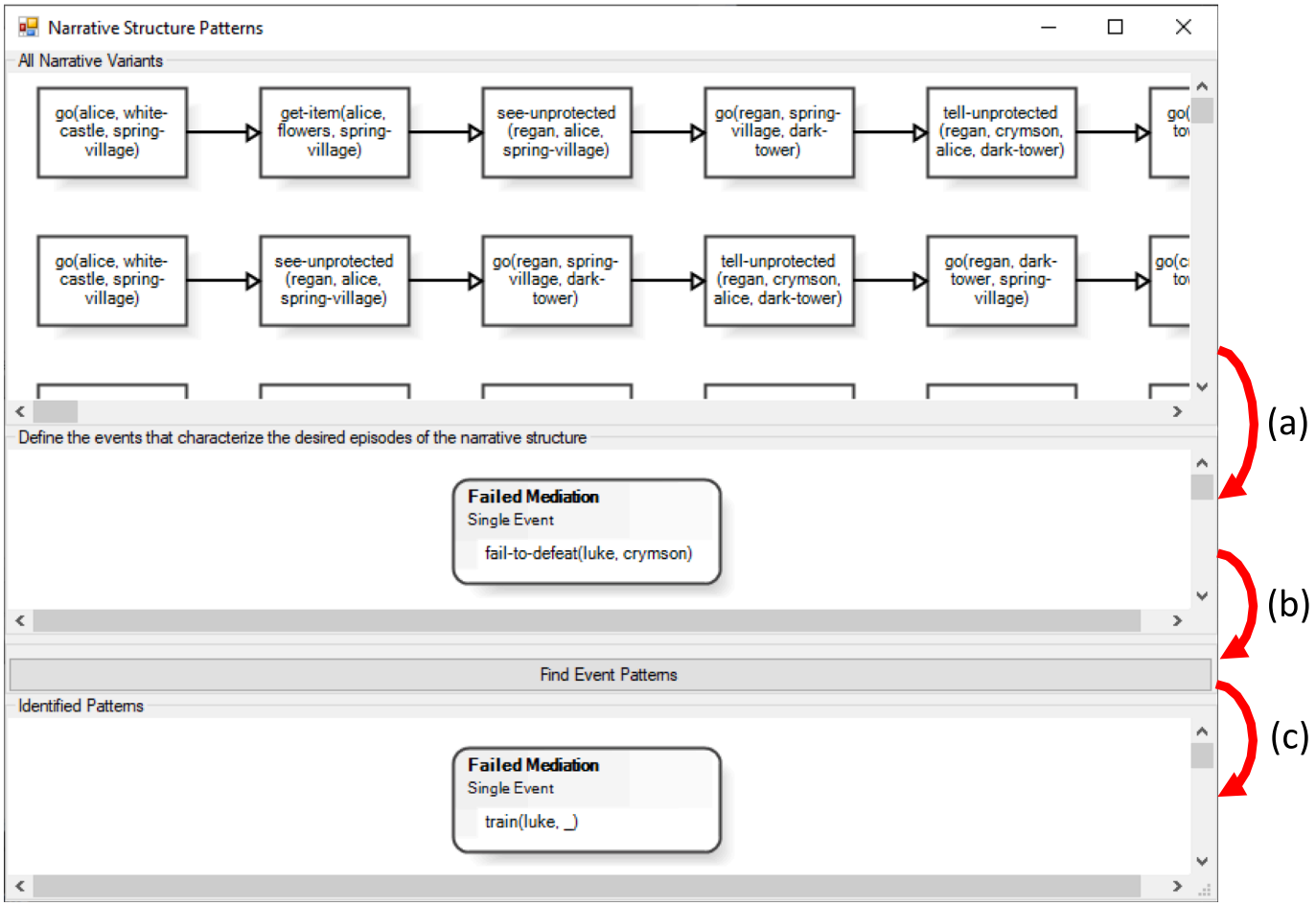


Fig. 8. Process of identifying the event patterns that can interfere in the structure of the narrative: (a) the user analyzes the narrative variants generated for the current story domain and identifies the event that characterizes the occurrence of the "Failed Mediation" episode; (b) the user presses the "Find Event Patterns" button to start the search for event patterns that can interfere in the occurrence of the "Failed Mediation" episode; and (c) the system shows the identified event patterns (the single event *train*(*luke*, _)).

inadequate to overcome the challenges that are presented. In "Apotheosis 1", the third episode, the hero joins a community and gains a highly respected status. "Mediation", the fourth episode, takes place when the hero is summoned to pursue a mission that was previously not understood. During the "Errance" episode, the fifth episode, the hero wanders aimlessly, still lacking a critical skill. In "Preparation 2", the sixth episode, the hero meets an old sage who imparts the missing knowledge. The seventh episode, "Quest", marks the effective beginning of the final mission, with ample opportunities for success. In "Apotheosis 2", the eighth episode, the hero achieves the quest and receives the reward. The final episode, "Denouement", follows the hero's new and transformed life.

The baseline story for the game was planned considering the main episodes of the grail-based narrative structure. The story involves the kidnapping and imprisonment of princess Alice by Crymson, who coerces her into revealing the locations of three elemental items. Crymson then embarks on a quest to locate these items in order to acquire unstoppable powers. Luke is informed about Alice's situation but acknowledges that he is not strong enough to defeat Crymson. Thus, he embarks on a journey to find a special sword that can enhance his strength (Preparation 1). Following the acquisition of the sword (Apotheosis 1), Luke confronts Crymson but is ultimately defeated by the villain's malevolent powers (Failed Mediation). However, after recovering from the battle, Luke sets off on a new quest to obtain a unique shield that can aid him in stopping Crymson's attacks (Quest). Luke successfully secures the shield by overcoming a knight's challenge (Preparation 2). Empowered by the shield, Luke confronts and vanquishes Crymson, ultimately leading to Alice's release (Apotheosis 2). The story culminates with the marriage of Alice and Luke (Denouement).

Various stories can emerge from the baseline story depending on player's decisions while performing Luke's actions and the order in which the other characters perform their actions. For example, Luke may still be unable to defeat Crymson after obtaining the special shield if the villain is able to perform a powerful transformation using the elemental power items before the final battle. In this case, Luke will have to pursue alternative ways to increase his strength, such as receiving dark powers from the evil mage Abraham. However, this alternative has unfavorable consequences, including a negative impact on Alice's feelings towards Luke and a potential prevention of their marriage at the end of the story, which will lead the drama manager to interfere in the plot to ensure the occurrence of the "Denouement" episode.

The gameplay of the prototype is driven by the tasks given to the player character (Luke), which involves completing various objectives such as traveling through different locations, collecting items, and interacting with other characters. The actions of non-player characters are presented through cutscenes. An example of a cutscene is presented in Fig. 11, which portrays the moment when Sara informs Luke about Alice's kidnapping.

A video demonstration exhibiting the impact of characters' actions on the world state and the utilization of the authoring tools for creating new production rules is accessible at the following URL: https://www.

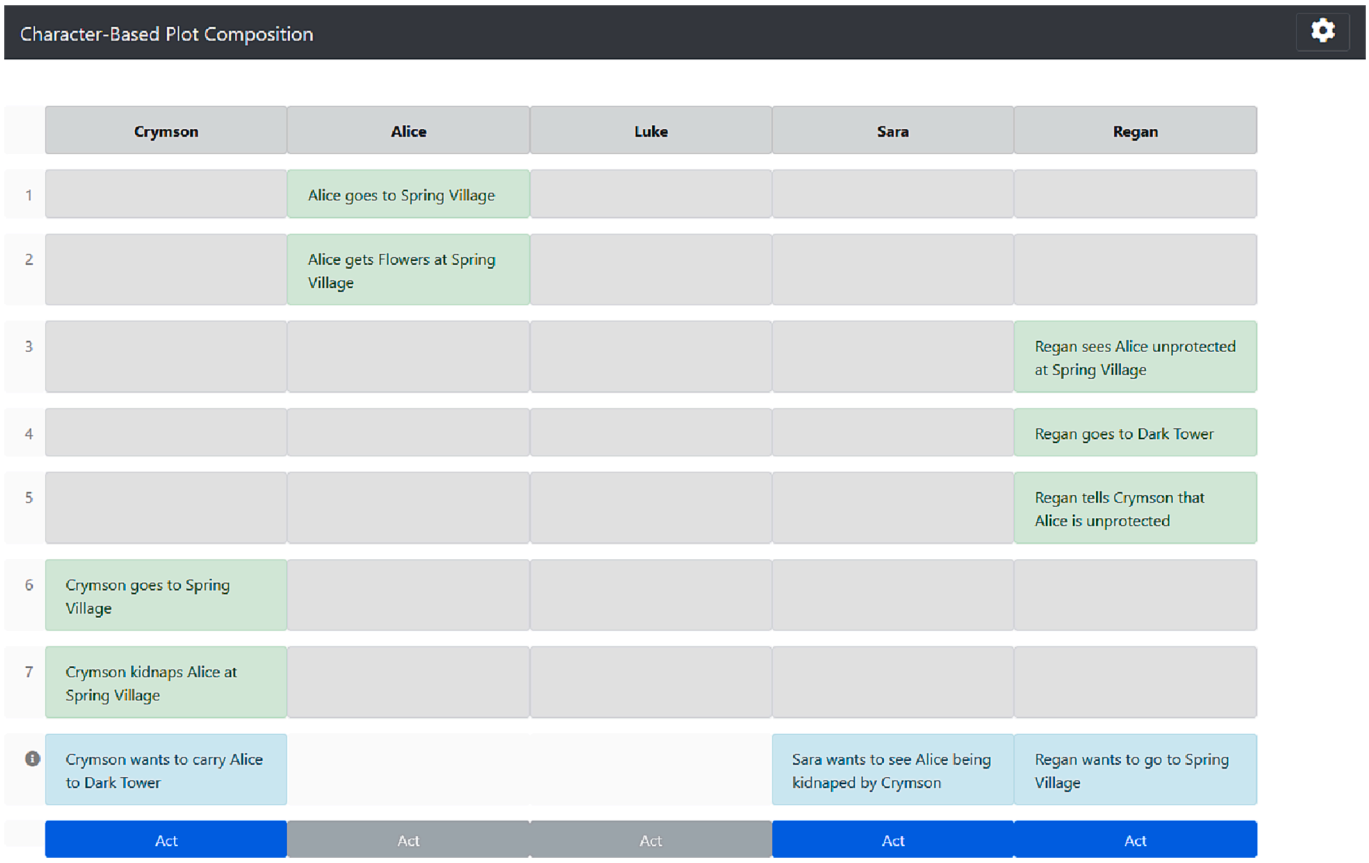


Fig. 9. Visualization module of the authoring tool.

Fig. 10. Scene from the prototype game: the player (controlling Luke) walks towards a place where he can collect an item called "Pendant of Wisdom".

youtube.com/watch?v=HGw9EUFeucA.

6.2. Technical evaluation

In light of the high computational complexity of automated planning algorithms, which increases as the number of objects and operators grows, we conducted a scalability evaluation of our proposed character-based model through a performance test across different story domains of increasing complexity. To conduct this experiment, we created five variants of the story domain used in our prototype game (presented in Section 6.1), each with varying numbers of characters, operators, objects, facts in the initial state, and episodes in the narrative structure. The main statistics of the story domains used in the experiment are presented in Table 1, with D3 representing the base story domain of our game prototype.

The system's performance was tested for each story domain by generating complete stories using the proposed system, with random choices made when player interactions were needed. Considering that the simulation of multiple characters in our model requires the planner to solve several planning problems with varying complexities during the

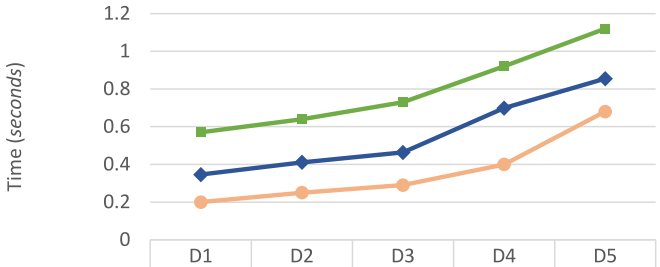
Fig. 11. Cutscene from the prototype game: Sara informs the player character (Luke) about Alice's kidnapping.

Table 1
Statistics of the story domains evaluated in the experiment.

	Story Domains				
	D1	***D2***	***D3***	***D4***	***D5***
Characters	3	4	5	6	7
Objects and Locations	20	25	30	35	40
Facts in Initial State	89	171	233	272	315
Operators	12	23	32	36	40
Episodes in the Narrative Structure	1	2	3	4	5

narrative generation process (a new problem is solved every time a new goal is generated), we calculated the minimum, maximum, and average time spent by the planning algorithm to find solutions for the planning problems that occurred during the simulation. This process was repeated 10 times for each story domain to calculate the average time. The experiment was conducted on an Intel Core i7 7820HK, 2.9 GHZ CPU, 16 GB of RAM, NVIDIA GTX 1070 8 GB GPU, using a single CPU core to process the planning algorithm. The results of this test are shown in Fig. 12.

The results of the experiment confirm that the performance of the narrative generation process is significantly impacted by the complexity of the story domain. Additional tests with story domain variations revealed that the number of objects, locations, and facts in the initial state do not have a significant effect on the planner's performance. However, the number of operators is the primary factor that affects the performance of the planner. Despite the scalability limitations, the system was able to achieve an encouraging level of variety and complexity with the story domain designed for the prototype game. The prototype generates plots with around 50 events and 7 different storylines with major variations, such as plots with different endings or significant differences in how the events unfold. Additionally, it produces numerous storylines with minor variations, such as plots with changes in the order of events or minor differences in content, like the location where certain events take place.



	D1	D2	D3	D4	D5
Maximum	0.57	0.64	0.73	0.92	1.12
Average	0.35	0.41	0.46	0.70	0.85
Minimum	0.2	0.25	0.29	0.4	0.68

Fig. 12. Minimum, average, and maximum time to solve planning problems in five different story domains (time is presented in seconds).

We also conducted an experiment to test the effects of the proposed narrative generation method on the real-time performance of our prototype game. During the experiment, we calculated the minimum, maximum, and average frame rates for three gameplay sessions of our game, each lasting approximately 15 min. The frame rate was measured with game running on the same computer used for the previous performance experiment, using a 1080p screen resolution (1920 × 1080). The results revealed an average frame rate of 132 FPS (frames per second), with a minimum of 84 FPS and a maximum of 152 FPS. We observed no significant frame rate drops when the planning algorithm generated characters' actions, as the planner instances were executed in a separate process from the rendering task. The variations in frame rate were related to the differences in the geometric complexity of various 3D locations in our game. These findings suggest that our method can be applied to games without affecting their real-time performance.

6.3. Authoring evaluation

The proposed method to manage the plot structure of character-based interactive narratives was developed to enable authors to define and control the narrative structure of emergent interactive stories. To support the use of our method by users with no programming skills, an authoring tool was developed, which allows authors to design and test interactive narratives, with no explicit requirement of programming knowledge. In order to evaluate whether our authoring tool achieved its goals, we conducted a usability test aiming at:

1. Evaluating the overall acceptance of the authoring tool for the specification of the story domain, including the definition of the narrative structure; and
2. Comparing the performance between users with programming knowledge and users without programming knowledge.

The comparison between programmers and non-programmers was devised considering that formal logic still is an important ingredient for the specification of a story domain. Although the authoring tool enables users to create all the elements for the story domain through a visual interface and intuitive diagrams, some formal logic concepts, such as operators with preconditions and effects, are still present and might affect the usability of our tool when users with no programming skills are considered. For our experiment, we assume that a successful authoring tool for interactive storytelling should enable users with no programming knowledge to create interactive narratives that are comparable to those created by experienced programmers.

6.3.1. Participants

Two groups of participants were recruited for the study: *Group 1* consisting of 16 Communication Sciences bachelor's students, which were enrolled in the Creative Writing module of the course (2nd year) and had no programming knowledge; and *Group 2* comprising 16 Computer Science bachelor's students that were enrolled in the Artificial Intelligence module of the 2nd year. The students were invited to participate in the study by their professors of Creative Writing and Artificial Intelligence at Universidade Europeia, IADE. The participation in the study was optional and was presented to students as an extra-curricular activity for them to learn about emergent interactive storytelling technologies.

An analysis of participants' profiles revealed that 6 subjects of Group 1 were male, 10 female, and their ages ranged from 19 to 24 years (mean of 20.3). For Group 2, 13 participants were male, 3 female, and their ages ranged from 19 to 26 years (mean of 20.8). All participants of Group 1 stated they had no programming knowledge. Participants of

Group 2 classified their programming skills as: 43.7% intermediate, 31.3% advanced, and 25% expert. Two participants of group 2 indicated that they had previous experience with automated planning. None of the participants had previous experience in creating interactive narratives.

6.3.2. Procedure

The study comprised a two-session workshop that was organized at Universidade Europeia, IADE. Both Group 1 and Group 2 attended the workshop at the same time. In the first session, participants filled out a consent form, answered a basic demographic questionnaire, and had a short lecture on interactive storytelling and narrative structures (Three-Act Structure, Hero's Journey, and the Grail-Based Narrative Structure). After the lecture, the instructor introduced our authoring tool and showed how to create a simple interactive version of the *Rapunzel* fairy tale as a step-by-step example (using the grail-based narrative structure). Participants were instructed to follow all the steps performed by the instructor and reproduce the interactive *Rapunzel* fairy tale on their computers. All participants' questions and difficulties were addressed by the instructor during the workshop. This session lasted 3 h.

The second session took place one day after the first session. In this session, participants were asked to create a new interactive narrative based on the *Little Red Riding Hood* fairy tale using our authoring tool. No restrictions related to the number of characters, events, or the general complexity of the story were imposed, but we asked participants to use an existing narrative structure to guide the narrative generation process. Participants did not receive any assistance from the instructor during this session. A time limit of 3 h was imposed for participants to complete their narratives.

After finishing the narrative, participants filled out a questionnaire with 10 questions derived from System Usability Scale (SUS) [42], which were slightly modified to better reflect the nature of our authoring system (replacing the terms "system" by "authoring tool" or "tool" and adding the usage goals in some items).[6] Table 2 show all statements of the modified SUS questionnaire used in this study. Each statement was given on a five-point Likert scale ranging from "strongly disagree" (0) to "strongly agree" (4). The final SUS score was calculated according to SUS scoring guidelines [42].

After completing the questionnaire, participants were interviewed about their experience with the authoring tool. The interview was caried out individually and comprised a short presentation of the narrative created by the participant, followed by general questions regarding the participant's expectations, the strategy used to design the narrative, the difficulties faced while using the authoring tool, and suggestions for improvements.

Table 2
Adapted SUS questionnaire used in our study.

ID	Statement
ST 1	I think I would like to use this authoring tool frequently to create interactive narratives.
ST 2	I found the authoring tool unnecessarily complex.
ST 3	I thought the authoring tool was easy to use.
ST 4	I think that I would need the support of a technical person to be able to use this tool to create interactive narratives
ST 5	I found the various functions in the authoring tool were well integrated.
ST 6	I thought there was too much inconsistency in the authoring tool.
ST 7	I would imagine that most people would learn to use this tool very quickly.
ST 8	I found the authoring tool very cumbersome to use.
ST 9	I felt very confident using the authoring tool to create interactive narratives.
ST 10	I needed to learn a lot of things before I could get going with the authoring tool.

[6] Previous studies have shown that replacing similar terms and adding additional details in a SUS questionnaire has no effects on its reliability and validity [45][46].

6.3.3. Results and discussion

We started the analysis of the study results by assessing the participants' success rate and the time they spent to finish the task. Twenty-eight participants (87.5%) delivered a complete narrative by the time limit. Out of the four participants that were not able to finish the narrative: two revealed during the interview that they underestimated the complexity of the story domain they were designing and ended up running out of time (one participant was from Group 1 and one from Group 2); one participant reported that he was not able to fix a logical issue with an operator, which prevented his narrative from progressing as expected (this participant was from Group 2); and the last participant that was not able to deliver a complete narrative had a technical problem with the computer (an unexpected reboot), causing the loss of most of the story domain (this participant was from Group 1).

Although 12.5% of the participants were not able to create an interactive narrative during the experiment (2 participants of Group 1 and 2 participants of Group 2), most part of the participants successfully completed the task independently of their programming skills. In addition, all participants that failed in finishing the task during the experiment stated that they believed that they would be able to complete the narratives if they had more time.

In order to verify if participants' programming skills had any impact in the time needed to complete the specification of the narrative, we conducted a two-sample *t*-test comparing the time spent by participants of the two groups (considering only the participants that finished the narrative, N = 26). For the analysis of the results, we considered a significance level $\alpha = 0.050$. The test revealed a significant difference in the scores ($t(26) = 2.440$, $p = 0.022$), with the mean score for Group 1 ($M = 2.764$, $SD = 0.295$) being higher than the mean score for Group 2 ($M = 2.507$, $SD = 0.261$) (see Table 3). The magnitude of the differences in the means was significant (Mean Difference = 0.257, 95% *CI*: 0.041 to 0.474). These results suggest that users with programming knowledge are able to complete the specification of a story domain in our authoring tool faster than user with no programming skills.

In order to evaluate and compare the complexity of the stories created by participants, we analyzed seven characteristics of their story domains: (C1) number of characters; (C2) number of operators; (C3) number of production rules; (C4) number of general rules; (C5) average story length (i.e., average number of events); (C6) number of outcomes (i.e., different endings); and (C7) number of episodes of the narrative structure. To compare the two groups, we conducted a two-sample *t*-test for all the analyzed characteristics of the story domains. As shown in Table 4, the test revealed significant differences in the scores of the number of characters (C1) ($t(26) = 2.455$, $p = 0.021$), with the mean score for Group 1 ($M = 4.357$, $SD = 1.008$) being higher than the mean score for Group 2 ($M = 3.571$, $SD = 0.646$); in the scores of the average story length (C5) ($t(26) = 2.255$, $p = 0.033$), with the mean score for Group 1 ($M = 16.857$, $SD = 2.825$) being higher than the mean score for Group 2 ($M = 14.429$, $SD = 2.875$); and in the scores of the number of outcomes (C6) ($t(26) = 2.798$, $p = 0.010$), with the mean score for Group 1 ($M = 2.214$, $SD = 0.802$) being higher than the mean score for Group 2 ($M = 1.500$, $SD = 0.519$). These results suggest that participants of Group 1 (Communication Sciences bachelor's students) created more detailed narratives than participants of Group 2 (Computer Science bachelor's students).

In order to assess the overall usability of the authoring tool, we analyzed the results of the SUS questionnaire. The mean SUS score for Group 1 was 79.843 (SD = 10.625) and for Group 2 was 80.781 (SD = 9.430), which indicates that our authoring tool scored above average for both groups of participants (considering the recommend SUS average score of 68 [43]). We also conducted a two-sample *t*-test comparing the SUS scores of the two groups (considering all participants, N = 32). The test revealed no significant differences between Group 1 and Group 2 ($t(30) = -0.254$, $p = 0.397$).

Table 3
Two-sample *t*-test comparing the time spent by participants of Group 1 (no programming knowledge) and Group 2 (with programming skills).

				Levene's Test		*t*-test for Equality of Means						
		Mean	SD	F	*p*	*t*	*df*	Two-Sided *p*	Mean Diff.	Std. Error Diff.	95% CI	
											Lower	Upper
Time	Group 1	2.764	0.295	0.470	0.499	2.440	26	0.022	0.257	0.105	0.041	0.474
	Group 2	2.507	0.261									

Table 4
Two-sample *t*-test comparing seven characteristics of the story domains created by participants of Group 1 and Group 2: C1 - number of characters; C2 - number of operators; C3 - number of production rules; C4 - number of general rules; C5 - average story length (i.e., average number of events); C6 - number of outcomes (i.e., different endings); and C7 - number of episodes of the narrative structure.

				Levene's Test		*t*-test for Equality of Means						
		Mean	SD	F	*p*	*t*	*df*	Two-Sided *p*	Mean Diff.	Std. Error Diff.	95% CI	
											Lower	Upper
C1	Group 1	4.357	1.008	3.000	0.095	2.455	26	0.021	0.786	0.320	0.128	1.444
	Group 2	3.571	0.646									
C2	Group 1	9.214	1.578	0.075	0.786	0.327	26	0.747	0.214	0.656	−1.134	1.563
	Group 2	9.000	1.881									
C3	Group 1	6.571	1.016	0.112	0.740	−0.17	26	0.863	−0.071	0.410	−0.915	0.772
	Group 2	6.643	1.151									
C4	Group 1	0.214	0.426	0.934	0.343	0.478	26	0.637	0.071	0.150	−0.236	0.379
	Group 2	0.143	0.363									
C5	Group 1	16.85	2.825	0.020	0.890	2.255	26	0.033	2.429	1.077	0.215	4.643
	Group 2	14.42	2.875									
C6	Group 1	2.214	0.802	0.444	0.511	2.798	26	0.010	0.714	0.255	0.190	1.239
	Group 2	1.500	0.519									
C7	Group 1	1.929	0.616	0.000	1.000	−0.61	26	0.545	−0.143	0.233	−0.621	0.336
	Group 2	2.071	0.616									

For a more detailed analysis of the usability results, we also inspected the normalized mean scores of each statement of the SUS questionnaire (Fig. 13). A two-sample *t*-test comparing the normalized mean scores of each statement for both groups revealed only a significant difference in the scores of statement 7 ("I would imagine that most people would learn to use this tool very quickly") ($t(30) = -2.967$, $p = 0.003$), with the normalized mean score for Group 1 ($M = 0.562$, $SD = 0.232$) being lower than the normalized mean score for Group 2 ($M = 0.781$, $SD = 0.179$). These results suggest that there are no relevant differences in the usability of the proposed authoring tool when considering users with programming knowledge and no programming knowledge.

The interviews conducted after the experiment also provided some insightful information about our system and the authoring process in general. 68.7% of the participants stated that their expectations regarding the tool and the generated narratives were met. The other 31.3% argued that they were expecting the process to be "easier" and "more automated" (they were somehow disappointed by the amount of information that needs to be specified by the author of the story). 56.2% of the participants also mentioned ChatGPT as "a new technology that should be explored to create stories" or suggested the integration of ChatGPT into our system to "assist them in the process of creating the story".[7] Although we did not evaluate the quality of the generated narratives from **the** players' perspectives, the participants were generally satisfied with the stories they created, with 71.8% of them indicating that they were able to create the story they wanted.

When questioned about the importance of the narrative structure in the story, 31.2% of the participants indicated that it was "useful to add some additional constraints in the narrative", allowed them to "make sure some events always happen", and helped them "avoiding the story to end prematurely" (7 of these participants were from Group 1 and 3 from Group 2). 40.6% of the participants stated that the narrative they created was not complex enough for them to "see the effects of the narrative structure" or indicated that they were not able to "test if the rules created for the narrative structure were affecting the story" (5 of these participants were from Group 1 and 8 from Group 2). The remaining 28.2% of the participants stated that they "didn't pay much attention to the narrative structure" or indicated that they added one of the episodes that was shown in the workshop as example just because it was a "requirement of the task" (4 of these participants were from Group 1 and 5 from Group 2). By analyzing the narrative structures chosen by participants for their story domains, it was also possible to notice that most of them (85%) used the grail-based narrative structure, which was presented in more detail during the workshop. Only three participants applied other narrative structures: two participants of Group 1 and one participant of Group 2 used the Hero's Journey, and one participant of Group 1 used the Three-Act Structure.

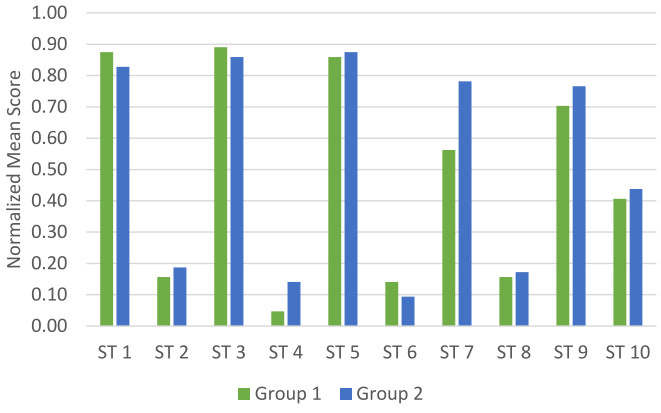


Fig. 13. Normalized mean scores for each statement of the SUS questionnaire. See Table 2 for a description of each statement.

[7] It is important to notice that this study was carried out in February 2023, which was when ChatGPT was a very popular topic among the general population.

The main difficulties pointed by participants during the interview include the complexity of creating a story by thinking about what exists in the world instead of what happens next (mentioned by 56.2% of the participants from Group 1 and by 31.2% of participants from Group 2); the difficulty of finding logical errors in the specification of the story domain (mentioned by 37.5% of the participants from Group 1 and by 43.7% of participants from Group 2); the overload of modeling concepts present in the different diagrams used to model the story domain (mentioned by 43.7% of the participants from Group 1 and by 18.7% of participants from Group 2); and the difficulty of visualizing the initial world state when the diagram has several objects and relations (mentioned by 18.7% of the participants from Group 1 and by 31.2% of participants from Group 2). Participants also provided several suggestions for improvements of our tool, including the addition of some shortcuts, the implementation of undo and redo features, the need for an option to automatically arrange the diagram of the initial world state, the inclusion of zoom-in and zoom-out features, and the need for debugging and error highlighting tools.

7. Concluding remarks

In this article we presented and evaluated a novel approach to manage the plot structure of character-based interactive narratives in games, which combines multi-agent planning with a drama management strategy based on narrative structures. The proposed model allows authors to establish an expected structure for emergent interactive narratives, which can avoid unexpected situations that may lead to stories that are not complex enough to create an interesting drama.

The main contributions of this work are: (1) a new method to manage the plot structure of character-based interactive stories in games (see Section 2 for a comparison with previous approaches); and (2) a new authoring tool that allows authors to create and test interactive narratives using graphical interfaces and intuitive diagrams. We believe that our method offers a viable alternative to the traditional branching structures commonly used in the game industry. Our method provides a practical approach to integrate emergent stories in games while ensuring that authors can control the basic structure of the narrative (avoiding the unpredictability problem constantly questioned by the game industry [47]).

In our experiments, the proposed method revealed encouraging results. Our model was successfully implemented in a 3D RPG, and the performance test results confirmed the effectiveness of our approach in highly interactive game environments. Despite the simplicity of the gameplay mechanics of the prototype RPG, the results demonstrate the capability of our method to generate and control character-based narratives in real-time.

Based on the evaluation of the authoring tool, we can conclude that our method can be effectively used by a wide range of users, regardless of their programming skills. Although users with programming knowledge (Computer Science students) were able to create interactive narratives faster than those without programming knowledge (Communication Sciences students), the Communication Sciences students were able to create more detailed and complex narratives. These findings suggest a potential impact of the user's background on the quality of the narratives created with our tool. The high SUS scores obtained in the evaluation also suggest that our authoring tool is well accepted, and there were no significant differences in the usability of the tool when considering users with programming knowledge and no programming knowledge. Overall, these results suggest that the proposed authoring tool can effectively support the development of character-based narratives without requiring programming knowledge.

While our experiments have shown promising results for the proposed method, it is necessary to address certain methodological considerations and limitations of this study. Broadly speaking, while claiming that the present results represent a relevant advancement, we recognize that future continuing work would be welcome, especially to reduce to an even larger extent the burden of authorial effort.

First, we must point out that the preliminary evaluation of our authoring tool presented here is still rather limited, having been applied to a small, under-representative, homogeneous sample (young undergraduate students with no story-writing background). Although the obtained results provide a general perspective on the usability of our authoring tool, further studies still are needed to validate our authoring support with more specialized users, such as industry professionals. It is especially important to comprehend how professional story writers would deal with the complexity of creating a story by thinking about what exists in the world instead of what happens next, which was pointed by almost half of the participants of our study as one of their main difficulties. Secondly, it is also important to notice that the use of a specific task and the imposition of a time limit may also have affected the normal workflow of the authoring process in our user study (considering as a normal scenario the case where the author has freedom and time to use his creativity to design any story). Therefore, future studies should consider more open scenarios to give more freedom to authors. We also plan to investigate users' satisfaction with the generated stories more deeply, considering both authors and prospective players.

Another important aspect to consider when evaluating the applicability of our model is that applying any schema-based approach, like ours, to game development involves certain difficulties. The specification of the story domain is a manual and time-consuming task, which requires the author to create a logical description of the game world, define planning operators, create production and general rules, and define the narrative structure to guide the plot. This limitation was even pointed out by almost a third of the participants of our study, which did not demand the specification of a very complex narrative. Further research on this subject may involve the development of methods to automatically extract part of the information used in the story domain from existing narratives or game structures, as done in [44,48,49]. The use of language generation models, such as ChatGPT and GPT-4, to generate or assist users in the process of specifying the story domain also represents a promising direction for future research.

The logic used in our automated planning system combined with the proposed plot management strategy guarantees the consistency of the generated stories. More precisely, we adopt a subset of the formalism proposed by Ciarlini et al. [50], where plots (event sequences) are generated by a planning algorithm, which lets each character of the story plan its own actions. Although the planning system ensures the logical consistency of the plans of each individual character, it does not take into account the actions of the other characters. Therefore, our system (the Drama Manager modules) needs to validate all characters' actions before executing them. When an inconsistency is found, a replanning procedure is performed to find an alternative – and logically consistent – plan of actions to achieve the character's goals. Although this approach was successfully applied in our system, there are alternatives extensively explored in the area of multi-agent planning [51,52], which might simplify this process, such as temporal logic [53]. Another essential open issue that demands in-depth future work is related with the need of a formalism to verify the consistency of the rendered story with respect to more abstract and higher-level goals, such as fun, flow, and engagement.

Lastly, it is important to notice that this article focused only on the author side of experience (to produce content), but the final consumer of the content (the players) cannot be ignored. Therefore, as further research, we intend to conduct user studies to evaluate the player experience in games that use our model to generate and control the narrative. We firmly believe that interactive narratives generated by artificial intelligence with guidance of a human author are one of the key ingredients needed for the emergence of a new and more immersive generation of digital games.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

We want to thank CNPq (National Council for Scientific and Technological Development) and FINEP (Funding Agency for Studies and Projects), which belong to the Ministry of Science, Technology, and Innovation of Brazil, for the financial support.

References

[1] M.O. Riedl, V. Bulitko, Interactive Narrative: An Intelligent Systems Approach, AIMag 34 (1) (2012) 67, https://doi.org/10.1609/aimag.v34i1.2449.
[2] C. Moser, X. Fang, Narrative Structure and Player Experience in Role-Playing Games, Int. J. Human-Comput. Interaction 31 (2) (2015) 146–156, https://doi.org/10.1080/10447318.2014.986639.
[3] R. Cardona-Rivera, J. Robertson, S. Ware, B. Harrison, D. Roberts, R. Young, Foreseeing meaningful choices, in: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 10 (1), AAAI Press, 2021, pp. 9-15. https://doi.org/10.1609/aiide.v10i1.12716.
[4] S. Stang, "This action will have consequences": Interactivity and player agency, Game Stud. 19 (1) (2019).
[5] V. Breault, S. Ouellet, J. Davies, Let CONAN tell you a story: Procedural quest generation, Entertainment Comput. 38 (2021) 100422, https://doi.org/10.1016/j.entcom.2021.100422.
[6] E.S. Lima, B. Feijó, A.L. Furtado, Hierarchical Generation of Dynamic and Nondeterministic Quests in Games, in: Proceedings of the International Conference on Advances in Computer Entertainment Technology, 2014, Article No. 24. https://doi.org/10.1145/2663806.2663833.
[7] E.S. de Lima, B. Feijó, A.L. Furtado, Procedural generation of branching quests for games, Entertainment Comput. 43 (2022) 100491, https://doi.org/10.1016/j.entcom.2022.100491.
[8] M. Cavazza, F. Charles, S. J. Mead, Interacting with virtual characters in interactive storytelling, in: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, 2002, pp. 318-325. https://doi.org/10.1145/544741.544819.
[9] J. Campbell, The Hero with a Thousand Faces, Princeton University Press, Princeton, United States, 1973.
[10] E.S. Lima, A.L. Furtado, B. Feijó, M.A. Casanova, Towards Reactive Failure-Recovery Gameplaying: The Fall and Rise of the Grail Hero, in: Proceedings of the XV Brazilian Symposium on Computer Games and Digital Entertainment, 2016, pp. 262-271.
[11] B. Ip, Narrative Structures in Computer and Video Games: Part 1: Context, Definitions, and Initial Findings, Games Culture 6 (2) (2011) 103–134, https://doi.org/10.1177/1555412010364982.
[12] B. Kybartas, R. Bidarra, A Survey on Story Generation Techniques for Authoring Computational Narratives, IEEE Trans. Comput. Intell. AI Games 9 (3) (2017) 239–253, https://doi.org/10.1109/TCIAIG.2016.2546063.
[13] A.I. Alhussain, A.M. Azmi, Automatic Story Generation: A Survey of Approaches, ACM Comput. Surv. 54 (5) (2022) 1–38, https://doi.org/10.1145/3453156.
[14] E.S. Lima, B. Feijó, A.L. Furtado, A Character-based Model for Interactive Storytelling in Games, in: Proceedings of the XXI Brazilian Symposium on Computer Games and Digital Entertainment, 2022, pp. 1-6. https://doi.org/10.1109/SBGAMES56371.2022.9961071.
[15] M.O. Riedl, A. Stern, Believable Agents and Intelligent Story Adaptation for Interactive Storytelling, in: S. Göbel, R. Malkewitz, R., I. Iurgel (eds.) TIDSE 2006, Lecture Notes in Computer Science 4326 (2006) pp. 1-12, 2006. https://doi.org/10.1007/11944577_1.
[16] R. Aylett, S. Louchart, A. Tychsen, M. Hitchens, R. Figueiredo, C.D. Mata, Managing emergent character-based narrative, in: Proceedings of the 2nd International Conference on Intelligent Technologies for Interactive Entertainment (INTETAIN '08), Article No. 5, 2008.
[17] Y. Cai, C. Miao, A. H. Tan, Z. Shen, A Hybrid of Plot-Based and Character-Based Interactive Storytelling, in: Technologies for E-Learning and Digital Entertainment. Lecture Notes Comput. Sci. 4469 (2007) pp. 260-273. https://doi.org/10.1007/978-3-540-73011-8_27.
[18] J. Porteous, A. Lindsay, Protagonist vs Antagonist PROVANT: Narrative Generation as Counter Planning, in: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '19), 2019, pp. 1069-1077.
[19] J. Porteous, M. Cavazza, F. Charles, Narrative generation through characters' point of view, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '10), 2010, pp. 1297–1304.
[20] J. Orkin, Symbolic Representation of Game World State: Toward Real-Time Planning in Games, in: Proceedings of the AAAI Workshop on Challenges in Game Artificial Intelligence, AAAI Press, 2004, pp. 26-30.
[21] R.M. Young, M. Riedl, M. Branly, A. Jhala, R.J. Martin, C.J. Saretto, An architecture for integrating plan-based behavior generation with interactive game environments, J. Game Develop. 1 (1) (2004) 51–70.
[22] J. Orkin, Goal-Oriented Action Planning (GOAP). https://alumni.media.mit.edu/~jorkin/goap.html (accessed 20 January 2023).
[23] G. Boeda, Multi-Agent Cooperation in Games with Goal Oriented Action Planner: Use Case in WONDER Prototype Project, AIIDE 17 (1) (2021) 204–207, https://doi.org/10.1609/aiide.v17i1.18909.
[24] V. Breault, S. Ouellet, J. Davies, Let CONAN tell you a story: Procedural quest generation, Entertainment Computing 38 (2021) Article No. 100422. https://doi.org/10.1016/j.entcom .2021.100422.
[25] E.S. Lima, B. Feijó, A.L. Furtado, Player Behavior and Personality Modeling for Interactive Storytelling in Games, Entertainment Comput. 28 (2018) 32–48, https://doi.org/10.1016/j.entcom.2018.08.003.
[26] A. Yacoub, G. Nicolescu, M. el-A. Hamri, C. Frydman, Towards using DEVS for modelling adaptive storytelling in virtual games, in: Proceedings of the 50th Computer Simulation Conference (SummerSim '18). Society for Computer Simulation International, Article 27, 2018, pp. 1-12. https://dl.acm.org/doi/10.5555/3275382.3275409.
[27] E. S. Lima, B. Feijó, A. L. Furtado, Procedural Generation of Quests for Games Using Genetic Algorithms and Automated Planning, in: Proceedings of the XVIII Brazilian Symposium on Computer Games and Digital Entertainment, 2019, pp. 495-504. https://doi.org/10.1109/SBGames.2019.00028.
[28] E. S. Lima, B. Feijó, A. L. Furtado, Adaptive Branching Quests Based on Automated Planning and Story Arcs, in: Proceedings of the XX Brazilian Symposium on Computer Games and Digital Entertainment, 2021, pp. 9-18. https://doi.org/10.1109/SBGames54170.2021.00012.
[29] M. Riedl, B. Li, H. Ai, A. Ram, Robust and authorable multiplayer storytelling experiences, in: Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'11), AAAI Press, 2011, pp. 189–194. https://doi.org/10.1609/aiide.v7i1.12450.
[30] D. Balas, C. Brom, A. Abonyi, J. Gemrot, Hierarchical Petri Nets for story plots featuring virtual humans, in: Proceedings of the Fourth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE'08), AAAI Press, 2008, pp. 2–9. https://doi.org/10.1609/aiide.v4i1.18664.
[31] H. K. H. Abd El-Sattar, A New Framework for Plot-Based Interactive Storytelling Generation, in: 2018 Fifth International Conference on Computer Graphics, Imaging and Visualisation, 2008, pp. 317-322. https://doi.org/10.1109/CGIV.2008.50.
[32] H.K.H. Abd el-sattar, A new plot/character-based interactive system for story-based virtual reality applications, Int. J. Image Grap. 10 (01) (2010) 113–133, https://doi.org/10.1142/S021946781000369X.
[33] E. S. Lima, A. L. Furtado, B. Feijó, M. A. Casanova, Plot Composition by Mapping Situation Calculus Schemas into Petri Net Representation, in: B. Göbl, E. van der Spek, J. Baalsrud Hauge, R. McCall (eds.) Entertainment Computing – ICEC 2022. ICEC 2022. Lecture Notes in Computer Science 13477 (2022) pp. 61-75. https://doi.org/10.1007/978-3-031-20212-4_5.
[34] T. Bylander, The computational complexity of propositional STRIPS planning, Artif. Intell. 69 (1–2) (1994) 165–204, https://doi.org/10.1016/0004-3702(94)90081-7.
[35] A.J. Coles, A.I. Coles, M. Fox, D. Long, Forward-Chaining Partial-Order Planning, in: Proceedings of the Twentieth International Conference on Automated Planning and Scheduling, 2010, pp. 42-49. https://doi.org/10.1609/icaps.v20i1.13403.
[36] R.E. Fikes, N.J. Nilsson, STRIPS: A new approach to the application of theorem proving to problem solving, Artif. Intell. 2 (3–4) (1971) 189–208, https://doi.org/10.1016/0004-3702(71)90010-5.
[37] G. Freytag, Technique of the Drama: An Exposition of Dramatic Composition and Art, University Press of the Pacific, United States, 2004.
[38] P. Parker, The art and science of screenwriting, Intellect Books, England, 1999.
[39] B. Feijó, E.S. Lima, A.L. Furtado, A Transdisciplinary Approach to Computational Narratology. Proceedings of the XX Brazilian Symposium on Computer Games and Digital Entertainment, 2021.
[40] P. Chen, The Entity-Relationship Model - Toward a Unified View of Data, ACM Trans. Database Syst. 1 (1) (1976) 9–36, https://doi.org/10.1145/320434.320440.
[41] W. W. Kibler (trans.), Chrétien de Troyes - Arthurian Romances. Penguin Books, England, 1991.
[42] J. Brooke, SUS: a "quick and dirty" usability scale, in: P. W. Jordan, B. Thomas, B. A. Weerdmeester, A. L. McClelland (eds.), Usability Evaluation in Industry, Taylor and Francis, 1996.
[43] J. Brooke, SUS: a retrospective, J. Usability Stud. 8 (2) (2013) 29–40.
[44] E.S. Lima, B. Feijó, A.L. Furtado, Computational Narrative Blending Based on Planning, in: J. Baalsrud Hauge, J. Cardoso, L. Roque, P.A. Gonzalez-Calero (eds), Entertainment Computing – ICEC 2021. Lecture Notes in Computer Science 13056 (2021) pp. 289-303. https://doi.org/10.1007/978-3-030-89394-1_22.
[45] A. Bangor, P.T. Kortum, J.T. Miller, An empirical evaluation of the system usability scale, Int. J. Human-Comput. Interaction 24 (6) (2008) 574–594, https://doi.org/10.1080/10447310802205776.
[46] J.R. Lewis, J. Sauro, The Factor Structure of the System Usability Scale, in: M. Kurosu (eds), Human Centered Design. HCD 2009. Lecture Notes in Computer Science 5619 (2009) pp. 94-103. https://doi.org/10.1007/978-3-642-02806-9_12.
[47] A. Koptelov, How Is AI Used in Video Game Development?, IT Chronicles, January 2022. https://itchronicles.com/artificial-intelligence/how-is-ai-used-in-video-game-development/ (accessed 22 January 2023).

[48] E.S. Lima, B. Feijó, M.A. Casanova, A.L. Furtado, Storytelling Variants Based on Semiotic Relations, Entertainment Comput. 17 (2016) 31–44, https://doi.org/10.1016/j.entcom.2016.08.003.
[49] E.S. Lima, A.L. Furtado, B. Feijó, Storytelling Variants: The Case of Little Red Riding Hood, in: K. Chorianopoulos, M. Divitini, J. Baalsrud Hauge, L. Jaccheri, R. Malaka (eds.) Entertainment Computing - ICEC 2015. ICEC 2015. Lecture Notes in Computer Science 9353 (2015). https://doi.org/10.1007/978-3-319-24589-8_22.
[50] A.E.M. Ciarlini, M.A. Casanova, A.L. Furtado, P.A.S. Veloso, Modeling interactive storytelling genres as application domains, J. Intell. Inf. Syst. 35 (2010) 347–381, https://doi.org/10.1007/s10844-009-0108-5.
[51] P.J. Hoen, K. Tuyls, L. Panait, S. Luke, J.A. La Poutré, An Overview of Cooperative and Competitive Multiagent Learning, in: K. Tuyls, P. J. Hoen, K. Verbeeck, S. Sen (eds.) Learning and Adaption in Multi-Agent Systems. LAMAS 2005. Lecture Notes in Computer Science 3898 (2005) pp. 1-46. https://doi.org/10.1007/11691839_1.
[52] A. Torreño, E. Onaindia, A. Komenda, M. Štolba, Cooperative Multi-Agent Planning: A Survey. ACM Computing Surveys 50 (6) (2018) Article 84. https://doi.org/10.1145/3128584.
[53] A. Nikou, J. Tumova, D. V. Dimarogonas, Cooperative task planning of multi-agent systems under timed temporal specifications, in: 2016 American Control Conference (ACC), 2016, pp. 7104-7109. https://doi.org/10.1109/ACC.2016.7526793.