Event Relations in Plan-Based Plot Composition

ANGELO E. M. CIARLINI
Universidade Federal do Estado do Rio de Janeiro
and
SIMONE D. J. BARBOSA, MARCO A. CASANOVA, and ANTONIO L. FURTADO
Pontifícia Universidade Católica do Rio de Janeiro

The process of plot composition in the context of interactive storytelling is considered from a fourfold perspective, that is, from the syntagmatic, paradigmatic, antithetic, and meronymic relations between the constituent events. These relations are shown to be associated with the four major tropes of semiotic research. A conceptual model and set of facilities for interactive plot composition and adaptation dealing with the four relations is described. To accommodate antithetic relations, corresponding to the irony trope, our plan-based approach leaves room for the unplanned. A simple storyboarding prototype tool has been implemented to conduct experiments.

Categories and Subject Descriptors: H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Interaction styles*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Plan execution, formation, and generation*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation languages; temporal logic*

General Terms: Design, Languages, Theory

Additional Key Words and Phrases: Interactive storytelling, plots, planning, narratology, tropes

ACM Reference Format:

Ciarlini, A. E. M., Barbosa, S. D. J., Casanova, M. A., and Furtado, A. L. 2009. Event relations in plan-based plot composition. ACM Comput. Entertain. 7, 4, Article 55 (December 2009), 37 pages. DOI = 10.1145/1658866.1658874 http://doi.acm.org/10.1145/1658866.1658874

1. INTRODUCTION

The role of storytelling in games has long been the subject of lively debates [Wardrip-Fruin and Harrigan 2004]. Although some authors believe that story and games are in direct opposition [Costikyan 2002], most agree that successful narrative in games is possible, and a few argue for the importance of story creation as part of gameplay [Wallis 2008]. However, a different sort of narrative is required: it must be nonlinear and play-centric, that is, it must revolve around

Authors' email addresses: angelo.ciarlini@uniriotec.br; {simone,casanova,furtado}@inf.puc-rio.br

DOI 10.1145/1658866.1658874 http://doi.acm.org/10.1145/1658866.1658874

the player's experience [Pearce 2002]. The player is no longer a mere consumer of the narrative, but both a consumer and a (co-) producer of the plot. The game designer typically selects a genre. In game-playing, interactive storytelling emerges, but care must be taken to ensure that the basic rules of the genre, as well as the corresponding tropes and narrative structures, are understood by the coauthors of the story [Wallis 2008].

A few computational systems and approaches have been proposed to support interactive storytelling. Some of them focus on the interaction among characters [Cavazza et al. 2002], whereas others focus on plot structure and coherence [Grasbon and Braun 2001], and a few others attempt to combine both [Mateas and Stern 2000]. What kind of system would be suitable for assisting users in creating stories within games or other interactive storytelling contexts? *Planning algorithms* have proven to be a useful alternative to help create narratives by exploring different chains of events to achieve the characters' or the storytellers' goals [Ciarlini et al. 2005; Riedl and Young 2004]. In game-playing, planning algorithms make it practical to create nonlinear narratives that are both coherent and diverse by allowing players to proceed in different courses of actions with varying results, and yet respecting the game structure, rules, and constraints.

To support the production of stories, we have drawn on what semiotic research has singled out as the *four major tropes* [Burke 1969], namely: metaphor, metonymy, synecdoche, and irony. By offering mechanisms derived from these tropes, we intend both to augment the expressiveness of narrative models and to provide better support to authors who are less familiar with or confident in creating and telling stories.

In this article, we associate those tropes with four types of relations between narrative events: *syntagmatic*, *paradigmatic*, *meronymic,* and *antithetic*. They play a basic role in an interactive plan generating a system that creates plots within a predefined genre.

Narratology studies distinguish three levels in literary composition: fabula, story and text [Bal 2002]. In the present work, we stay at the *fabula* level, where the characters acting in the narrative are introduced, as well as the narrative plot, consisting of a partially-ordered set of events. We focus on plots whose constituent events happen as a consequence of a predefined repertoire of actions, which we call *operations*, deliberately performed by the characters. Plot composition will be treated here as a plan generation process, and hence the terms plot and plan will be used interchangeably. But since narratives are often more attractive when shifts are allowed to occur, the user shall retain the power to issue certain *directives* when interventions are needed or desired.

Starting from such considerations, this article proposes a fourfold way to characterize plot composition at the fabula level. Section 2 describes the relations between events in correspondence with the four major tropes. Section 3 outlines how we model an intended genre, to whose conventions the plots must conform. Section 4 sketches a simple example, the main features of our plan-based prototype tool. Section 5 presents concluding remarks. Appendix A contains the full specification of our running example.

2. FROM TROPES TO EVENT RELATIONS

2.1 The Four Major Rhetorical Tropes

It has been suggested that the four major rhetorical tropes—metaphor, metonymy, synecdoche, and irony—provide models for remarkably comprehensive analyses in different areas [Burke 1969; Chandler 2007; White 1973]. They all involve *relations between pairs of words*, thanks to which, given two related words w1 and w2, a person can meaningfully use w1 to refer to w2.

They are not defined in a uniform way by linguists, there being much disagreement, especially on the distinction between metonym and synecdoche. A useful discussion is found in Chandler [2007], where many practical applications of Burke's four tropes theory are surveyed.

Metaphor [Lakoff and Johnson 1980; Ortony 1996] and synecdoche [Chandler 2007] have to do with hierarchical structures such as those represented in ontologies [Breitman et al. 2007]. If one concept C_1 can be metaphorically used to denote another concept C_2, the two concepts are said to be *similar* or *analogous*, and are placed under a more general concept $\hat{C}$ that subsumes both of them. C_1 and C_2 would be connected to $\hat{C}$ by **is-a** links. Also, C_1 would be connected to C_2 by an **is-like** link [Breitman et al. 2007]. Clearly, metaphor is a displacement along the verbal *paradigmatic axis* [Saussure 2006], from which we took the suggestion of a *paradigmatic relation* between events.

In synecdoche, concept C_1 is used to denote concept C_2 if C_1 is a part of C_2 (which calls for another link, C_1 **part-of** C_2); the converse substitution, from whole to part, is also usual in common parlance. The corresponding association between events is called a *meronymic relation* in the present article.

According to Chandler [2007], metonyms are based on various indexical relationships between concepts, notably the substitution of effect for cause, and to convey an idea of contiguity. Borrowing again from Saussure [2006], we require the presence of *syntagmatic relations* between events to justify their being meaningfully placed in sequence.

Irony is the most intriguing of the four tropes. In verbal communication, it reflects the opposite of the thoughts or feelings of the speaker or writer (as when you say "I love it" when you hate it) or the opposite of the truth about external reality (as in "there's a crowd here" when the area is empty). It also takes the form of substitution by *dissimilarity* or *disjunction*. Variations such as understatement and overstatement can also be regarded as ironic. At some point, exaggeration may slide into irony [Chandler 2007]. Disclosing paradoxes and hidden agendas in literary texts, in sharp contrast between the declared intentions and the real ones, is another source of irony, constituting a trend in critical studies known as *deconstruction* [Culler 1983].

Not only mental attitudes, feelings, and statements can be ironic—actions can also be ironic, but always in an unplanned, nondeliberate fashion. Irony is in fact a characteristic of certain situations of intrigue that are often referred to as *dramatic irony* [Booth 1974].

Consequently any kind of irony induces an *antithetic relation* between events that look, in principle, incompatible with each other, given their

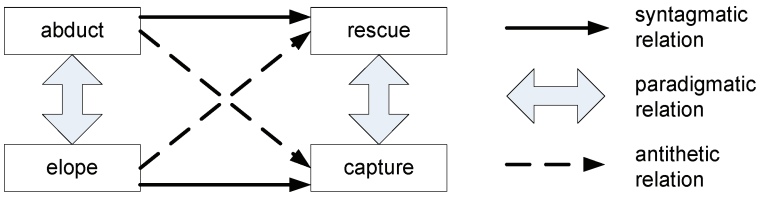

Fig. 1. Syntagmatic, paradigmatic, and antithetic relations.

dependence on contexts characterized by radically opposite properties. Mediating two such events, the until then well-behaved world must suffer a disruptive shift, whereby the truth value of certain facts or beliefs is inverted, or certain properties move from one extreme to the other within the ascribed value range (e.g., from helplessly weak to heroically strong).

To illustrate the event relations derived from the major tropes, we employ a simple example to be referenced in the article. Consider four types of events, all having one woman and two men as protagonists: *abduction*, *elopement*, *rescue*, and *capture*. As demonstrated in folktale studies [Propp 1968], many plots mainly consist of an act of villainy, that is, of a violent action that breaks the initially stable and peaceful state of affairs, followed ultimately by an action of retaliation, which may or may not lead to a happy outcome.

Propp distinguished seven character roles (*dramatis personae*) according to the events assigned to each one's initiative: hero, villain, victim, dispatcher, donor, helper, false hero. Curiously, in literary texts involving the four events above, this distribution is not unique: we called the violent initial act "villainy," but the perpetrator of an abduction, and more often of elopement, can be the hero of the narrative, and in such cases the woman's original guardian (husband, father) is regarded as the villain.

Figure 1 shows the relations thus far discussed.

2.2 Syntagmatic Relations

To declare that it is legitimate to continue a plot containing abduction by placing a rescue next to it, we say that these two events are connected by a *syntagmatic relation*. More precisely, we can define the semantics of the two events in a way that indicates that the occurrence of the first leaves the world in a state wherein the occurrence of the second is coherent. Similarly, a plot involving elopement followed by capture looks natural, and hence these two events are likewise related.

The syntagmatic relation between events induces a weak form of causality or enablement, which justifies their *sequential ordering* inside the plot.

2.3 Paradigmatic Relations

The events of abduction and elopement can be seen as *alternative* ways to accomplish a similar kind of villainy. Both achieve approximately—though not quite—the same effect: one man takes away a woman from where she is and starts to live in her company at some other place. There are differences, of

course, since the woman's behavior is usually said to be coerced in the case of abduction, but quite voluntary in the case of elopement. In fact, it is usual to assume that a sentence such as "Helen elopes with Paris" implies that Helen had fallen in love with Paris.

To express that abduction and elopement play a similar function, we say that there is a *paradigmatic relation* between the two events. Likewise, this type of relation is perceived to hold between the events of rescue and capture, which are alternative forms of retaliation. And, again, there is a difference between the woman's assumed attitude, associated as before with her feelings. An abducted woman expects to be rescued from the villain's captivity by the man she loves. On the contrary, she will only return through forceful capture if she freely eloped with the seducer.

As the present example suggests, the syntagmatic and the paradigmatic axes identified by Saussure are really *not* orthogonal, in that the two relations cannot be considered independently when composing a plot. Thus, in principle, the two pairs abduction-rescue and elopement-capture in Figure 1 are the only normal combinations (the former illustrated by the Sanskrit *Ramayana* [Valmiki 1999] and the similarly structured Arthurian romance of *Lancelot* [Chrétien 1983; Furtado and Veloso 1996] and the latter by the Irish *Story of Deirdre* [McGarry1979]). Yet the next section shows that such limitations can, and even should, be waived occasionally.

2.4 Antithetic Relations

While normal plots whose outcome is fully determined can be composed exclusively on the basis of the two preceding relations, the possibility of introducing unexpected turns is often desirable in order to make the plots more attractive—and this requires the construct that we chose to call *antithetic relation*. A context where a woman suffers abduction by a ravisher whom she does not love would seem incompatible with a capture event, since there should be no need to employ force to bring the victim back. So in this sense, abduction and capture are in an antithetic relation.

The mythical *Rape of the Sabines* shows what can happen as a consequence of a drastic reversal of the circumstances: King Romulus is facing a problem at the newly founded city of Rome: the population is entirely male. To remedy the lack of females, he leads his men to break into the dwellings of the Sabines and to abduct their women. Sometime afterwards the Sabine warriors march against the Romans, but the women have no wish to be taken back, leaving to their countrymen no option except their capture. King Romulus's men had lawfully married them and made them bear children. A Roman chronicler [Titus Livius] reports the radical change in the women's feelings, and tells how the seemingly inevitable confrontation ended with the reconciliation of the two parties.

In contrast, modern history provides some distinctly regrettable examples of abduction actually followed by capture, categorized by the psychiatrist Nils Bejerot as the *Stockholm syndrome*. One case in point is the abduction by a group of terrorists of the daughter of a millionaire, who ended up joining her

tormentors in crime, and was captured by the police in San Francisco [Hearst and Moscow 1988].

Elopement followed by rescue provides a much stronger case of an antithetic relation. Indeed, elopement only makes sense if the victim loves the seducer, whereas, for this very motive, she would resist any attempt to rescue her, leaving forceful capture as the only viable alternative. Even so, the legendary story of *Helen of Troy*, in spite of various discordant interpretations, seems to offer a counter-example. Married to king Menelaus of Sparta, Helen fled to Troy in the company of Paris, out of her free will according to a number of versions (e.g., the *Heroides* [Ovid]). But after their escape to Troy, where they married, her love started to wane while the Trojan War followed its bloody course and she continued to recall the far manlier Menelaus. The *Iliad* [Homer] repeatedly signals this critical change of sentiment. In the end, her recovery turned from capture into rescue, as noted in the *Aeneid* [Virgil]. Paris was dead, and she had been delivered to Paris's brother Deiphobus. When the Greeks left the wooden horse and stormed the Trojan palaces, Helen herself made sure that Menelaus would win, and that he would realize that she was helping him in atonement for her previous misconduct. The shadow of Deiphobus recounts this episode to Aeneas; and what better example of irony could we find than his calling Helen "this peerless wife"?

One more example appears in several versions of the story of *Tristan and Isolde* [Marchello-Nizia 1995]. The knight and the queen had eloped and lived in harsh conditions in a forest. The dramatic change in their feelings, which allowed Isolde's rescue by king Mark via a simple invitation, with no need to fight, had a very curious cause: the expiry date of the love potion they had drunk before, while sailing from Ireland to Cornwall [Béroul 1970].

Generally speaking, if some *binary opposition*—in the present case, the "to love or not to love" dilemma—is allowed to be manipulated via some agency external to the predefined events, then we can have plots that no longer look conventional. A discontinuity is produced by such radical shifts in the context. Intervening between abduction and capture, or between elopement and rescue, a sudden change of feelings can give rise to these surprising sequences. And both in fiction and in reality, things do not always proceed according to plan. Natural phenomena and disasters, the mere passage of time, the intervention of agents empowered to change the rules, supernatural or magic manifestations, and so on, cannot be discounted.

Specifically for the tragedy genre, the *Poetics* [Aristotle] distinguishes between simple and complex plots, characterizing the latter by the occurrence of *recognition* ($\alpha\nu\alpha\gamma\nu o\rho\iota\sigma\iota\sigma$) and *reversal* ($\pi\varepsilon\rho\iota\pi\varepsilon\tau\varepsilon\iota\alpha$). Unlike reversal, recognition does not imply that the world itself has changed, but rather that the *beliefs* of one or more characters about the actual facts have changed. Due to a change in belief (in addition to those enumerated in the previous paragraph), a reversal in the course of action can take place, usually in a direction totally opposite to what had gone on before. Yet another possible external cause of both recognition and reversal in the tragic scene is intervention by a god (who is lowered onto the stage using a crane, hence aptly termed *deus ex machina)*.

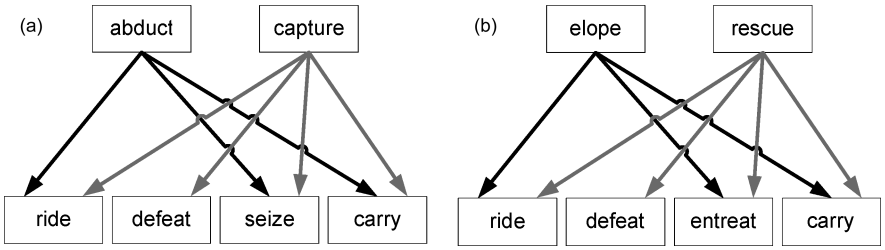

Fig. 2. Meronymic relations: (a) forceful actions and (b) gentle actions.

Aristotle's remarks are clearly relevant to the present discussion of plots in general. Following his lead, we admit state changes outside the regular regime of predefined events by allowing the user—literally acting as *ex machina* (via computer)—to impose variations to the context (both in terms of *facts* and *beliefs*), and thereby deviate the action from its predicted path.

This extreme device will be necessary to allow the elopement-rescue sequence. We decided, however, not to make it indispensable for abduction-capture in order to have a chance to present a good example of *erroneous* beliefs, contradicting the actual facts. Criminal records everywhere are full of simulated abduction pacts for drawing a ransom from a deluded family. Conversely, a man can unnecessarily decide that capture is the only way to bring back a woman, if he mistakenly believes her to love the ravisher.

2.5 Meronymic Relations

Meronymy is a word of Greek origin, used in linguistics to refer to the decomposition of a whole into its constituent parts. Forming an adjective from this noun, we call *meronymic relations* those that hold between an event and a lower-level set of events, with whose help it is possible to provide a more detailed account of the action on hand.

Thus, we could describe the abduction of a woman called Sita by a man called Ravana (characters from the *Ramayana* [Valmiki 1999]) as: "Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka". And her rescue by Rama could take the form: "Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace". But notice that such decompositions are not fixed, since the lower-level events are selected as required by the current state. For instance, with respect to the rescue event, the hero may already be present at the ravisher's dwelling, or perhaps the victim is not held in captivity, obviating the need for, respectively, the voyage or for fighting the enemy (Figure 2).

Details are most useful in passing from a somewhat abstract view of the plot to one at a more concrete physical level that is amenable (possibly after further decomposition) to the production of computer graphics animation [Ciarlini et al. 2005]. Mixed plots, combining events at different levels, also make sense, satisfying the option to represent some events more compactly while showing the others in detail.

3. A PLAN-BASED MODELING APPROACH

3.1 Modeling a Genre

To model a genre, to which the plots to be composed should belong, we must specify at least the following: (the subject of Section 3.1):

(a) what can exist at some state of the underlying mini-world;
(b) how states can be changed; and
(c) the factors that drive the characters to act.

In our model, we equate the notion of event with the state change resulting from the execution of a predefined operation. Being defined in terms of their preconditions and postconditions, operations can be readily chained together by a *plan-generating algorithm* [Ciarlini et al. 2005; Barros and Musse 2007] in order to achieve some character's goal. As a consequence, it becomes natural to equate plots (sequences of events) with plans (sequences of operations able to bring about the events). Also, to confer a degree of autonomy [Riedl and Young 2004] to the characters performing the operations, it is convenient to make their goals emerge from appropriately motivating situations.

Viewing plots as plans suggests an obvious plot composition strategy, that is, having a plan-generator as its main engine. This, and the fact that our conceptual model is expressed in Prolog, make the genre specification executable. In Sections 3.2, 3.3, and 3.5, we argue that, duly complemented by auxiliary routines, the planning strategy deals effectively with narrative plots in view of three out of the four event relations. To accommodate antithetic relations, however, it will be necessary to leave room to the *unplanned*, as proposed in Section 3.4, leading to plots that may to a limited extent break the conventions of the adopted genre.

3.2 Conceptual Design

We start with a conceptual design method involving three schemas: static, dynamic, and behavioral, which was developed for modeling literary genres encompassing narratives with a high degree of regularity such as fairy tales, and application domains for business information systems such as banking. Such applications are obviously constrained by providing a basically inflexible set of operations and, generally, by following strict and explicitly formulated rules [Furtado et al. 2008]. For brevity, we omit detailed logic programming notation; the full specification is shown in Appendix A.

The *static schema* specifies, in terms of the *entity-relationship* model [Batini et al. 1992], the entity and relationship classes and their attributes. In our simple example, `character` and `place` are entities. The attributes of characters are `name`, which serves as identifier, and `gender`. Places have only one identifying attribute, `pname`. Characters are pair-wise related by the binary relationships `loves`, `held_by`, and `consents_with`. The last two can only hold between a female and a male character. Two relationships, `home` and `current_place`, associate characters with places.

A *state* of the world consists of all facts about the existing entity instances and their properties at some instant. For example, `held_by(Sita,Ravana)` is a fact, meaning that Sita is forcefully constrained by Ravana, whereas `consents_with(Sita,Ravana)` is another fact that indicates that Sita has voluntarily accepted Ravana's proposals.

The *dynamic schema* defines a fixed repertoire of operations for consistently performing state changes. The *STRIPS* [Fikes and Nilsson 1971] model is used. Each operation is defined in terms of preconditions, which consist of conjunctions of positive or negative literals and any number of postconditions, consisting of facts to be asserted or retracted as the effect of executing the operation. Instances of facts such as `home` and `gender` are fixed, not being affected by any operation. Of special interest are the *user-controlled* facts which, although also immune to operations, can be manipulated through arbitrary *directives* (cf., Section 3.4). In our example, `loves` is user-controlled.

Again, for the present example, we have provided operations at two levels. The four main events are performed by level-1 operations: `abduct`, `elope`, `rescue`, and `capture`. Operations at level-2 are actions of smaller granularity, in terms of which the level-1 operations can be detailed: `ride`, `entreat`, `seize`, `defeat`, and `carry`.

The *behavioral schema* consists of *goal-inference* (a.k.a. situation-objective) *rules*, *belief rules*, and *emotional condition rules*, briefly explained by what follows.

In our running example, three goal-inference rules are supplied. The first one refers to the ravisher: if the victim, usually a princess, is not at her home and the hero is not in her company—a situation wherein she is clearly unprotected—the ravisher will want to do whatever is necessary to bring her to his home. The other goal-inference rules refer to the hero in two different situations, having in common the fact that the ravisher has the woman in his home. In one situation the hero believes that she does not love the other man; in the other situation he believes that she does. In both situations, he will want to bring her back, freely in the first case and by force in the second.

Informally speaking, beliefs correspond to the partial view, not necessarily correct, that a character currently forms about the factual context (for a formal characterization, cf., the BDI model [Cohen and Levesque 1990; Rao and Georgeff 1991]). The rules about belief that we formulated for our example look rational, but notice that they are treated as defaults, which can be overruled, as described in Section 3.4. A man (the hero or the ravisher) believes that the woman does *not* love his rival. If the rival has her confined, but if she has been observed in his company and on no occasion (state) was physically constrained, the conclusion will be that she is consenting (an attitude seemingly too subjective to be ascertained directly in a real context).

The emotional condition rules refer to the three characters. A man (or woman) is happy if currently in the company of his (or her) beloved, and bored otherwise. A special condition applies to the woman: she will be *absolutely* happy if, in addition to the first motive for contentment, she has never been constrained by either of the two adversaries.

3.3 Coherent Sequences

Moving along the *syntagmatic* axis is the primary task of the plan-generator, as it composes a coherent plot by aligning events in view of the pre- and post-conditions of the predefined operations.

For plot composition, it is convenient to proceed in a step-wise fashion, starting from a given initial state. At each state, the goal-inference rules are used to induce opportunistic short-term goals from which successive plot sequences will originate.

In an interactive environment, at any step, the user, henceforward called the *Author*, should be allowed to intervene, thereby reducing the characters' autonomy, but relying on the plan-generator to enforce consistency within the genre. To this purpose, the Author may indicate a goal to be tried by the plan-generator or even a specific operation, which the plan-generator may or may not find applicable.

A more complex request is to indicate a sparse list of operations, to be filled until a valid plot sequence containing all operations in the list, possibly interspersed with others, is formed. The Author may optionally also indicate the desired goal, which would otherwise be assumed to coincide with the effects of the last operation in the list.

After the stepwise process terminates, it should still be possible to perform various kinds of adaptations. Those that have to do with the syntagmatic relations include adding or deleting operations and changing the sequence if the partial order requirements imposed by the interplay of pre- and postconditions permit. For instance, consider plot P below:

```
P = start => ride(Ravana, Lanka, forest)
          => entreat(Ravana, Sita)
          => seize(Ravana, Sita)
          => carry(Ravana,Sita,Lanka)
```

which can be reordered to meet the author's preferences, to produce:

```
Ps = start => ride(Ravana, Lanka, forest)
           => entreat(Ravana, Sita)
           => carry(Ravana,Sita,Lanka)
           => seize(Ravana, Sita)
```

Curiously, both the original plan `P` and the reordered plan `Ps` suggest stories that may well happen in reality or fiction. In `P`, a voluntary elopement is disguised as an abduction, whereas in `Ps` elopement is followed by the woman's cruel confinement.

Also, a plot can be extended with more operations if the Author supplies an additional goal in an attempt to provide a continuation.

3.4 Alternative Choices

Moving along the *paradigmatic* axis provides an opportunity to obtain different plots by means other than simply changing the sequence of events within the partial order requirements.

First of all, alternatives may result by starting from a different initial state so that different goal-inference rules may be triggered. Notice also that more than one such rule may be ready for activation. In any case, the standard plan-generator's ability to *backtrack* is an expedient mechanism for engendering alternative plots.

To resort to violence as in an abduction or capture can certainly be regarded as excessive and unnecessary when the subject of the action loves the agent, even though our specification does not invalidate such an occurrence. Accordingly, if the goal-inference rules are in control and the context is not tampered with (but see Section 3.4), they will not figure in any generated plot. And yet the Author can have them as valid alternatives by simply using the option to directly indicate a goal to the plan-generator. Such a goal can be relatively non-specific, such as `current_place(Sita,palace)`, or else more restrictive, like `(current_place(Sita,palace), held_by(Sita,Rama))`, in which case only the forceful capture event will result.

At the adaptation phase, the ability to replace one or more operations is a way to produce alternatives. We must bear in mind that a replacement may require another if the Author is concerned with preserving consistency, so replacing `abduct` by `elope` normally implies the replacement of `rescue` by `capture`.

A particularly convenient way to deal with entire plots, rather than with individual operations, is to take advantage of the similarity or analogy among situations inherent in the notion of paradigms. Previously existing plots, no matter if composed manually or automatically, can be converted into *plot patterns* to be kept in a Library of Typical Plots [Furtado and Ciarlini 2001]. Plot patterns can then be *reused* to originate new plots, essentially by instantiating their variables in view of a new situation.

3.5 Shifts Along the Way

Until this point we restricted ourselves to planned and hence well-behaved plots. We now introduce a measure of transgression, disrupting the context in order to obtain plots with events in *antithetic* relation.

The Author, as *deus ex machina*, can interfere with the plan-generation discipline by issuing two kinds of *directives*, which can be applied both during composition and adaptation. One directive is `make_believe`, arbitrarily assigning a belief B to a character C, which overrules any previous belief on the same facts, either specified through the belief rules of the behavioral schema or stated by a previous application of the `make_believe` directive itself. If Sita was violently abducted by Ravana, Rama will believe (as a consequence of a belief rule) that she does not love the villain, and therefore that she will gladly consent to be rescued. However, the Author is allowed to induce Rama to falsely believe the contrary, which activates a goal-inference rule leading to a forceful `capture` event.

Another directive is `vary`, which manipulates user-controlled facts instead of mere beliefs. In our example, the only facts declared to be user-controlled are instances of the `loves` relationship, whose Boolean value will be inverted if the directive is applied. Sita can elope if she currently loves Ravana, and then be willingly rescued by Rama if the Author issues the directive to change her

feelings between these two events. But `vary` does not have to be explicitly called for. A helpful feature in the course of plan-generation can detect failures involving user-controlled facts, in which case the Author is asked whether or not the context should be tampered with accordingly.

In other example mini-worlds, we might have different kinds of user-controlled properties, for example, with numerical values within a range, such as degree of strength, which the `vary` directive could change in some radical proportion. Such representation is also appropriate for emotions in general, including love itself, enabling finely graded nuances of expression, obviously unattainable with simple two-valued Boolean alternatives.

In an attempt to offer clues to an Author intent on finding ways to, at a later stage, replace the external *deus ex machina* directives by some internal narrative device with a flavor of irony, almost crossing the borderline of plausibility, we began to investigate another line. Over time, rich repertoires of *motifs* have formed via folktales, myths, and popular culture [Aarne and Thompson 1987], often containing ingenious solutions to dilemmas arising from antithetic situations.

Authors have always felt free to borrow from all kinds of sources, and we can easily discover occurrences of certain motifs in the literature of different countries, modified as required by cultural differences. For our example, we found three convenient motifs:

(a) life token: an object whose aspect changes if the owner is in distress;

(b) love potion: stimulates romantic/erotic feelings; and

(c) ordeal: to vindicate a discredited or accused person.

where (a) (indexed as E761 in Aarne and Thompson [1987]) allows action without the unrealistic assumption that characters are omniscient (e.g., explaining how Rama learned that Sita suffered abduction in the forest); (b) provides an excuse for sudden variations in amorous attachments; and (c) serves to restore man's belief in his beloved's faithfulness. Curiously, both (b) and (c) occur in the *Tristan* romance, wherein the ordeal takes the especially ironic form of an ambiguous oath [Béroul 1970], while in the *Ramayana,* Sita has to walk through the fire [Valmiki 1999]. In our example, we treat these motifs as black boxes, merely associating their names to a <*situation*, *goal*> specification. Thus, if the Author wants to insert motifs (by simply mentioning their names) at the positions in a generated plot where the respective situation holds, this can be asked for at the adaptation phase.

Such insertions are therefore to be regarded as provisional annotations only, which the Author should later have to unravel by mapping the events in the motifs into analogous events congenial to the genre adopted in the plot. The mappings should preserve the <situation, goal> of the motif, and might require the definition of additional operations such as communicative acts. The persistence of motifs is a remarkable phenomenon with relatively modern versions: microchip implants for (a); aphrodisiac drugs like the LSD hallucinogen for (b); and lie detectors and truth serums for (c), all as dubious or controversial as their primitive counterparts, but equally acceptable to the general public.

3.6 Down to Details

As stated before, there may be *meronymic* relations between level-1 and level-2 operations. Creating plots in hierarchic fashion is a very common practice, starting with a broad view of the events, which in our example corresponds to the level-1 operators. At later stages, we would gradually decompose each event into finer-grain actions, possibly along more than just two levels, to the point of coordinated physical movements, as required for displaying animated scenes [Ciarlini et al. 2005].

When composing a plot, the plan-generator is free to mix the operations of the two levels, which is a reasonable default option considering that the Author may wish to treat some events more succinctly than others. But the Author may, on the contrary, settle for a uniform style by indicating that only one of the two levels will be used. This choice can be altered at any time, in composition or adaptation.

Once a plot is composed, it can be adapted either by detailing or summarizing its constituent operations. *Detailing* each level-1 operation `Op` in a plot into level-2 operations is treated as yet another plan-generation task, taking as *situation* the instantiated preconditions of `Op`, and as *goal* the effects of `Op`, and using the operations in the level-2 repertoire exclusively. More than one decomposition may be possible, depending on the initial state and on the changes effected by the preceding operations.

The inverse of detailing, *summarizing*, is also useful. We are currently restricted to a rather limited version, which only works if the detailed plan is divisible into subsequences that can be exactly subsumed by level-1 operations. This means that the process fails if other extraneous operations intervene. In other words, `summarize(P1,P2)` succeeds if and only if `detail(P2,P1)` also does.

Figures 2(a) and 2(b) are suggestive in that they illustrate a curious symmetry in how they map the example level-1 operations into level-2 operations. The decompositions in the two figures are the same, except for the substitution of `entreat` for `seize`. This is not surprising, since a similar decomposition comes as a consequence of the paradigmatic relation between the two villainy and the two retaliation events. Also notice that, in both figures, the event corresponding to villainy only differs from that of retaliation by the possible presence of `defeat`—reflecting our observation, after we surveyed a number of traditional narratives, that the villain almost always resorts to some trick, avoiding a confrontation that often (though not necessarily) occurs as part of retaliation.

The decompositions suggested by the two figures are typical but not unique, since the correspondence induced by the meronymic relations is not rigidly determined (i.e., it is context-sensitive, depending on the current state). For instance, `abduct` can be expressed by `seize` followed by `carry` if both the victim and the ravisher are currently at the same place, but they will need a preliminary `ride` if the former is in the forest and the latter still in his home.

All this suggests that it may be difficult to interpret what is happening by looking at a sequence of level-2 operations without examining the context.

In this regard, the ability to fill up (cf., Section 3.2) a sparse list of observed level-2 operations and then performing summarization, identifying what level-1 operation is taking place at some point, constitutes a nontrivial form of *plan-recognition* [Kautz 1991]. Plan generation is more directly relevant to the composition and adaptation of plots than the recognition of plans and objectives. But the latter task is an asset in interactive plan-supported game-playing environments, since each player might employ it as an aid to discover what the opponents are trying to do.

4. A PROTOTYPE IMPLEMENTATION

A very simple prototype, *PlotBoard*, was designed to experiment with the notions discussed here. Dealing with *storyboarding* [Truong et al. 2006], exclusively at the fabula level, serves to compose plots interactively with the help of an extended version of the early *Warplan* algorithm [Warren 1974]. Written in SWI-Prolog,[1] it interfaces with Java to show events in image format.

4.1 Some Features of the Plan Generator

The plan generator follows a backward chaining strategy. For a fact `F` (or `not F`) that is part of a given goal, it checks whether it is already true (or false) at the current state. If it is not, it looks for an operation `Op` declared to add (or delete) the fact as part of its effects. Having found such an operation, it then checks whether the precondition `Pr` of `Op` currently holds—if not, it tries recursively to satisfy `Pr`. Moreover, the plan generator must consider the so-called frame problem [Lloyd 1987], by establishing (in second-order logic notation) that the facts holding just before `Op` is executed stay valid unless explicitly declared to be altered as part of the effects of `Op`

In order to treat preconditions, we distinguish, and treat differently, three cases of facts:

(a) facts which, in case of failure, should be treated as goals to be tried recursively by the plan generator;
(b) facts to be tested immediately before the execution of the operation, but which will not be treated as goals in case of failure: if they fail, the operation simply cannot be applied; and
(c) facts that are not declared as added or deleted by any of the predefined operations.

Case (c) covers facts that are not changed by any operation and, therefore, are true iff they hold at the initial state. In this way, the planner may ignore operations containing preconditions that can never be achieved.

The preconditions of an operation are declared by a clause of the form `precond(Op, Pr):-B`, where `Op` is the signature of the operation (with its name and parameters), and `Pr` and `B` are conjunctions of literals. Each literal in `Pr` is a positive or negative fact, possibly preceded by a bar '/'. A fact preceded by

[1]http://www.swi-prolog.org/

'/' corresponds to case (b) above; otherwise, it corresponds to case (a). A fact included in B is simply tested against the current state, when the clause is selected and corresponds to case (c) above.

An example is the precondition clause of operation seize(M,W), where M is the agent and W the patient of the action. Clearly the two characters should be together at the same place, and, accordingly, the Pr argument shows two terms containing the same variable P to express this requirement, but the term for W is barred: /current_place(W, P), which does not happen in M's case. The difference has an intuitive justification: the prospective agent has to go to the place where the patient is, but the latter will just happen to be there for some other reason.

The proper treatment of (a) and (b) is somewhat tricky. Suppose the precondition Pr of operation Op is tested at a state S_1. If it fails, the terms belonging to case (a) will cause a recursive call whereby one or more additional operations will be inserted so as to move from S_1 to a state S_2 where Op itself can be included. It is only at S_2, not at S_1, that the barred terms in case (b) ought to be tested, and so the test must be *delayed* until the return from the recursive call, when the plan sequence reaching S_2 will be fully instantiated.

Operations can admit more than one precondition clause, so as to cope with different circumstances. This happens with the carry(M,W,P2) operation, whereby W will either freely consent to be transported to P2 by M, or will have to be forcefully held by him.

With respect to the added and deleted clauses declaring effects of operations, the plan generator also employs a barred notation to distinguish between two cases: (a) primary effects and (b) secondary unessential effects. In case (a), if a fact to be added by an operation Op already holds, or a fact to be deleted by Op does not hold, then Op is considered *nonproductive* and fails to be included in the plan. In contrast, in case (b), such lack of effect would be admitted and cause no failure.

As an example, consider the clause of operation capture(M1,W) that declares as deleted the fact held_by(W,M2), as a result of M1's action to take away W from M2. Notice that the fact may or may not hold prior to capture; it will hold if W was abducted by M2, but it will not hold if an elopement occurred instead—and that is why the barred notation is used for this particular deleted clause. On the contrary, the fact current_place(W,P2), where P2 is the home of M2, must necessarily be deleted by an effective execution of the operation, and so does not figure as barred.

During the generation of a plan for achieving a certain goal, pre- and postconditions are dynamically checked without modifying the initial state described in the Prolog database. When a generated plan is executed, however, the current state is updated accordingly by means of assert or retract Prolog commands on the facts to be added or deleted, respectively. A fact log(L)is used to keep track of the operations executed so far. Argument L contains the sequence of operations and is initiated with L=start. The sequence is extended with each successful plan execution and can be usefully retrieved for a variety of purposes. On the basis of the log and of the initial state, which is saved when a session begins, it is possible to query about facts at any intermediate state. It is

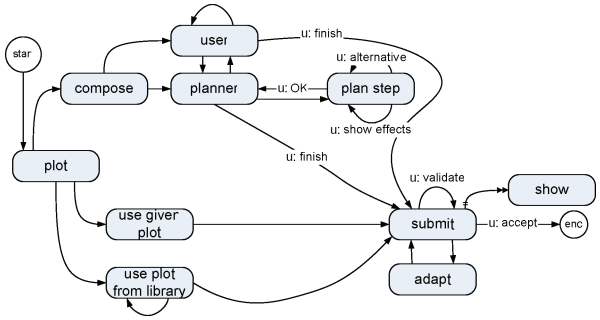

Fig. 3. Flow of control of the *PlotBoard* prototype.

also possible to save and restore any previous state S (initial or intermediate), which enables simulation runs.

User interventions, necessary to achieve unplanned situations, are permitted in a limited scale through *directives* that can be either intermixed with the operations in a plan or called separately. Two of these are used in our example, one for changing `loves` facts, immune to the predefined operations, and the other for changing characters' beliefs, which may not correspond to actual facts.

To finish this partial review of the plan features, we remark that the planning algorithm `plans(G,P)` is called in more than one way. More frequently, `G` is given as the goal, and `P` is a variable to which a generated plan will be assigned as output. However, an inverse usage has been provided wherein `P` is given and `G` is a variable; in this case, the algorithm will check whether `P` is valid and, if so, assign its net effects (a conjunction of `F` and `not F` terms) to `G`.

4.2 The PlotBoard Tool

We briefly describe how *PlotBoard* works after the Author enters the `plot` command. The diagram in Figure 3 will serve as a guide to the description.

The main option is to compose the plot from scratch in a step-wise fashion. Ideally, the Author should leave a measure of autonomy to the characters (branching into the `planner` node of the diagram). At each step, one subsequence of the plot will be generated. As if emerging from the mind of a character `C`, a short-term opportunistic goal `G` is instantiated by some goal-inference rule `(C,S,G)` if situation `S` of the rule currently holds. The planner is then used to generate a valid subsequence of events that reach the goal.

More than one rule may be simultaneously ready for activation, and the planning algorithm may find more than one alternative subsequence able to achieve the corresponding goals (whenever the planning algorithm backtracks). After the generation of the first alternative, the `plan step` node is reached, so that alternatives can be explored, as indicated by the self-loop around the node. While a subsequence is presented, the Author is prompted to either issue an `ok` reply or call for an alternative, possibly after inspecting what effects it would have. An `ok` reply is followed by a return to the `planner` node.

The subsequence thus selected is then executed in a simulated mode, and the Author is asked whether the `plan step` iterations should continue, producing further subsequences to be appended to the plot so far obtained, or whether the composition process is finished for the time being (passing to the `submit` node), though still subject to possible adjustments.

If the Author is more inclined towards a closer arbitrary control than to the character autonomy policy described above, several options are available to determine the goals that the planning algorithm should try to achieve. At the `user` node (which can be reached either from the `planner` node or directly from the `compose` node), the following menu is presented:

```
1: goal
2: operation
3: list of operations
4: query
5: operation level
6: directive
7: planner
8: finish
```

The first three options in this menu allow the user to try to explicitly insert goals and operations into the plot. Such insertions are automatically validated by the planning algorithm and can be rejected if they are not coherent. Again, the self-loops around the `user` node represent the possibility of alternative plot subsequences being offered to the Author.. These options permit step-wise composition, which can be entirely commanded from the `user` node, but they can also alternate with the activation of goal-inference rules by intercalating transfers of control to the `planner` node.

An additional purpose of the `user` node is to prepare and support the composition process, by allowing posing queries about the database state at each step, to change the operation level and to issue directives to alter the characters' beliefs and the value of user-controlled properties.

Whatever composition policy is preferred—autonomous, arbitrary, or mixed—the finished plot is passed to the `submit` node. At this point, the Author can either accept the plot, which terminates the process, or can go through one or more rounds of adaptation, using the options offered at the `adapt` node below.

```
1: detail
2: summarize
3: change sequence
4: add operation
5: delete operation
6: replace operation
7: extend
8: queries or directives
9: insert motif
10: back to the submit options
11: stop
```

To help decide whether to accept the current plot or perform other adaptations, the `submit` menu allows to validate the plot (again through the planning algorithm). This may be in order if the Author directly introduces specific changes (items 4-6 of the `adapt` menu), noting that in all other forms of adaptation the planning algorithm intervenes to prevent integrity violations.

Another feature available at the `submit` node deserves attention, since what it produces, together with the menu-based dialogues, constitutes the intended output of *PlotBoard*. If selected, via the `show` option, it provides a visual display that can be repeated for successive versions. For each operation in the current plot, the event it denotes appears as a rough drawing, side by side with a short template-driven natural language sentence.

We refer again to the diagram in Figure 3, to consider two ways to obtain a plot without requiring step-wise composition from scratch. In both cases, a full plot is used to start with, and in both cases the process converges afterwards to the `submit` node.

Branching into `use given plot`, the Author can either enter the intended plot or retrieve a previously composed one. The planning algorithm is automatically called to inspect the plot, operation by operation, to check whether each of them can be applied in view of the pre- and postconditions interplay. If an operation is found that can only be applied if a user-controlled property is tampered with, the possibility of changing the value of the property is indicated to the Author, who may or may not permit the execution of the necessary `vary` directive. If the Author denies permission, or if the offending property is not user-controlled, the plot is rejected.

The Library of Typical Plots (LTP) contains triples `(S,G,P)` specifying that, starting from a state where situation `S` holds, the execution o plot `P` leads to a state where goal `G` holds. When the node `use plot from library` is chosen, the system tries to find triples such that where situation `S` currently holds, thereby propagating the instantiation of the parameter variables figuring in `S` to goal `G` and plot `P`. If more than one such item is found, the Author will have once more an opportunity to select the preferred plot `P` among the alternatives presented.

4.3 An Example Run

At the initial state, both Rama and Ravana are in their homes, the royal palace and the city of Lanka, respectively, whereas Sita is alone in the forest. The two men love Sita, who only loves Rama. Starting to compose the plot, the Author invokes the planner in two stages, always selecting the detailed (level 2) alternatives. At this point, in natural language format, the plot would be:

> *Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka. Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace.*

Wishing to try different versions, the Author looks at the `adapt` menu, described in the previous section. The first change selected is the deletion of the two events that close the narrative. The next step is to issue directives to change

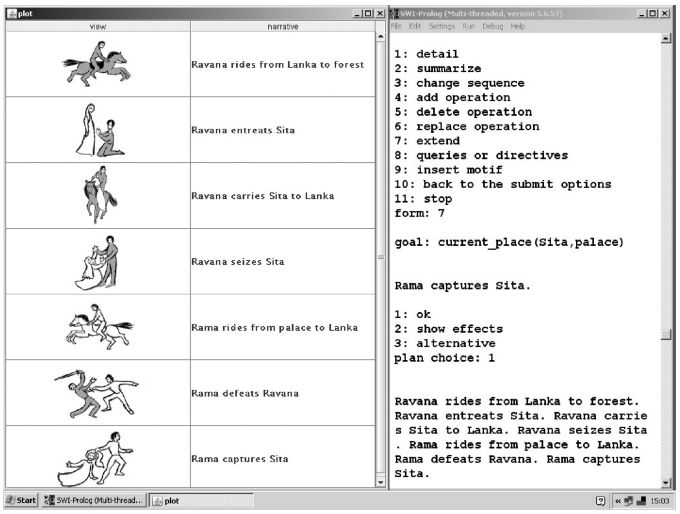

Fig. 4. A *PlotBoard* screen.

the emotional attachments and certain beliefs of the characters: now Sita loves Ravana and Rama believes it. This justifies adding `entreat(Ravana,Sita)` as the second event (after Ravana approaches the princess):

> *Ravana rides from Lanka to forest. Ravana entreats Sita. Ravana seizes Sita. Ravana carries Sita to Lanka. Rama rides from palace to Lanka. Rama defeats Ravana.*

The plot now suggests the fake abduction pattern, wherein the villain seizes his pretended victim only to simulate a violent action. The Author then wonders if the same events could be arranged in some different sequence, and a dialogue ensues when the option `change sequence` is chosen in the `adapt menu`:

```
[f1:entreat(Ravana, Sita), f2:seize(Ravana, Sita)]
choose one of the fi tags: f1
[f1:seize(Ravana, Sita), f2:carry(Ravana, Sita, Lanka)]
choose one of the fi tags: f2
[f1:seize(Ravana, Sita), f2:ride(Rama, palace, Lanka)]
choose one of the fi tags: f1
```

The following sequence is then obtained:

> *Ravana rides from Lanka to forest. Ravana entreats Sita. Ravana carries Sita to Lanka. Ravana seizes Sita. Rama rides from palace to Lanka. Rama defeats Ravana.*

This sounds as overt elopement, after which the seducer restricts the woman's freedom. What can happen next?

Selecting the `extend` option of the `adapt` menu, the Author proposes: `current_place(Sita,palace)` as a goal, and the planner responds Figure 4) with: `Rama captures Sita`. Is this a satisfactory way to end the narrative? The

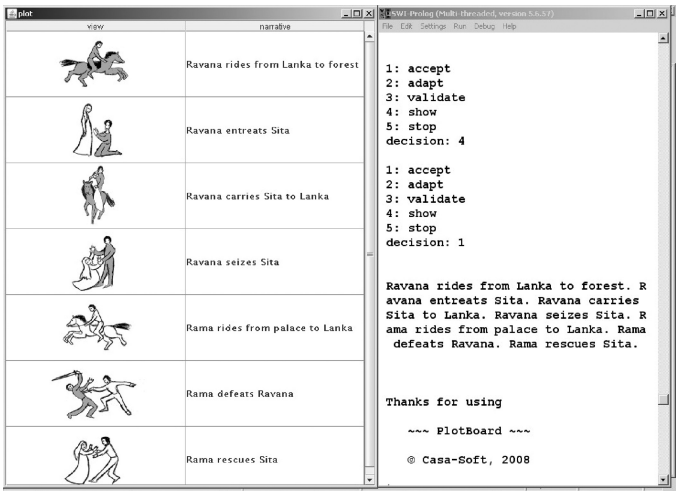

Fig. 5. The accepted plot.

Author selects option 8 and poses queries to learn what the characters think and how they feel:

```
query: beliefs
   Rama believes that Sita loves Ravana
   Ravana does not believe that Sita loves
   Rama
more queries?(yes/no): yes
query: emotional_condition
   Sita is bored. Rama is happy. Ravana is
   bored.
```

Sympathizing with the princess, the Author decides to reverse the situation. Perhaps her love for the hero could revive (as with Helen of Troy), and the last event is replaced according to this expectation: `capture(Rama,Sita)` turns into `rescue(Rama,Sita)`. How does it look now? Back at the `submit` menu, the Author asks to visualize the scenes and accepts this result, a happy ending for Sita as well as for the Author, who receives a grateful acknowledgment from the *PlotBoard* tool (Figure 5).

But much remains to be done. The *deus ex machina* directives should eventually be replaced by something internal to the narrative. Also, how to explain how Rama knew, without being told, that Sita had become Ravana's prisoner? To gather suggestions for possible reuse after modifications appropriate to the genre, the Author might have inspected (Figure 6) the applicable motifs, before issuing the final accept response. In which case the *life token*, the *love potion* (twice), and the *ordeal* motifs would be indicated at one or more positions in the plot wherein the respective motivating situation holds.

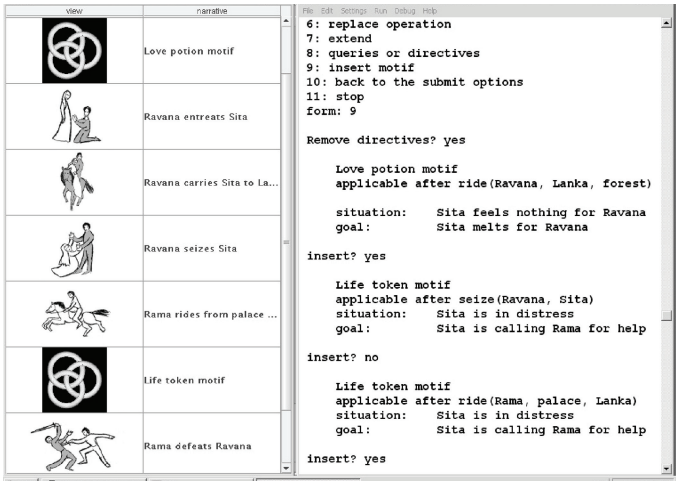

Fig. 6. Insertion of motifs (partial view).

5. CONCLUDING REMARKS

Although the process of plot composition could surely be enriched far beyond what is presented here, the suggested fourfold approach provides a sound initial basis. The conjecture that the interplay of the syntagmatic, paradigmatic, antithetic, and meronymic relations already permit ample coverage is reinforced by the connection between these relations and the four major tropes. Other concepts may be adduced to extend the model. If we see a disruption not as a discontinuity in one context but as an attempt to put together two originally incompatible contexts, the notion of *blending* [Fauconnier and Turner 2002; Casanova et al. 2008] comes to mind, as the technique or artisanship of conciliating the pending conflicts, which often requires a great deal of creativity.

The facilities associated with the four relations are adequate for other tasks besides storyboarding, under suitable user interfaces. In interactive storytelling systems designed for entertainment, as well as in games, they might prove instrumental to support the production of coherent stories that can cause surprise. Further research might investigate ways to adjust the generation of alternatives to user satisfaction models, so that there would no longer be a need to explicitly interfere to obtain varied and interesting outcomes.

Finally, let us recall that we have addressed the *fabula* level only, where one simply indicates *which* events should be included in the plots. A complex problem to be faced at the next level—the *story* level, where the concern is *how to tell* the events—is to find a plausible justification for the contextual disruptions introduced *ex machina* via user interaction. As mentioned, such elaborations may be suggested by some fanciful motif annotated in the plot. Moreover, a plurality of narrative objectives must be satisfied [Crawford 1984; Turner 1992; Montfort 2006].

At the third and last level—the *text* level—the narrative is *represented* in some medium, not necessarily the printed page. Natural language text-generation from plots of log-registered business transactions is covered in

Furtado and Ciarlini [2000]. In the realm of literary genres, an ongoing project applies computer graphic animation to display narrative plots [Ciarlini et al. 2005; Camanho et al. 2008].

APPENDIX A. CONCEPTUAL SCHEMAS FOR THE EXAMPLE

A.1 THE STATIC SCHEMA

```
/*
Syntax for specifying entities

entity(<E>,<ID>): entity <E> has identifier attribute <ID>
attribute(<E>,<ATTR>): entity <E> has attribute <ATTR>

Possible facts based on these specifications:
<E>(<ID_VALUE>) and <ATTR>(<E>,<ATTR_VALUE>)
In the example, characters and places are entities. Characters have attributes
name (identifier) and gender. Places are identified by attribute pname.
*/
entity(character,name).
attribute(character,gender).
entity(place,pname).

/*
Syntax for Specifying Relationships

relationship(<R>,<LPART>): relationship <R> has list of participants <LPART>

Possible facts based on these specifications have the form
<R>(<ID_PART_1>,<ID_PART_2>, ... )
Types of entities obey the order established in <LPART>

In the example, loves, held_by and consents_with are binary relationships
between characters (the first character is the subject); home and current_place
are binary relationships between characters and places.

Clause user_controlled(loves(_,_)) is used here to specify that the user can
change facts corresponding to the relationship loves
*/
relationship(loves,[character,character]).
relationship(home,[character,place]).
relationship(current_place,[character,place]).
relationship(held_by,[character,character]).
relationship(consents_with,[character,character]).
user_controlled(loves(_,_)).
```

A.2 THE DYNAMIC SCHEMA

```
/*
Syntax for Specifying Operations

operation(<OP_SIGNATURE>,<OP_LEVEL>): <OP_SIGNATURE> contains the name of
the operation and its arguments represented by variables. <OP_LEVEL> is a num-
ber corresponding to the operation's detail level.
```

precond(<OP_SIGNATURE>,<VARYING_LITERALS>): specifies preconditions of the operation. <VARYING_LITERALS> is a conjunction of literals that can be modified by operations. If there is a bar before a literal, it is not considered a subgoal to be fulfilled in the planning process. The body of the clause may contain a conjunction of nonvarying literals. If an operation has more than one clause of this kind, each clause corresponds to a different possibility for its execution.

deleted(<OP_SIGNATURE>,<DEL_FACT>): specifies, as a primary effect, that fact <DEL_FACT> has to be deleted by the operation. The body of the clause may contain a conjunction of nonvarying literals.

/deleted(<OP_SIGNATURE>,<DEL_FACT>): specifies that fact <DEL_FACT> will be deleted by the operation if it holds in the current state. The body of the clause may contain a conjunction of nonvarying literals.

added(<OP_SIGNATURE>,<ADD_FACT>): specifies, as a primary effect, that fact <ADD_FACT> has to be added by the operation. The body of the clause may contain a conjunction of nonvarying literals.

/added(<OP_SIGNATURE>,<ADD_FACT>): specifies that fact <ADD_FACT> will be added by the operation if it holds in the current state. The body of the clause may contain a conjunction of nonvarying literals.

*/

A.2.1 Level-1 Operators

/*

Operation abduct(M2,W): character M2 abducts character W.

- Preconditions: W is a woman that M2 loves; W and M2 have different home places; and W is currently at a place that is different from her home place and from M2's home place.
- Primary effects: W will no longer be at her current place (if it is different from M2's home place); W will be at M2's home place; and W will be held by M2.
- Other effects: M2 will no longer be at his current place if it is not his home place; and M2 will be at his home place.

*/

```
operation(abduct(M2,W),1).
deleted(current_place(W,Pc),abduct(M2,W)) :-
  home(M2,P2), not (Pc == P2).
/deleted(current_place(M2,Pc),abduct(M2,W)) :-
  home(M2,P2), not (Pc == P2).
added(current_place(W,P2),abduct(M2,W)) :- home(M2,P2).
/added(current_place(M2,P2),abduct(M2,W)) :- home(M2,P2).
added(held_by(W,M2),abduct(M2,W)).
precond(abduct(M2,W),current_place(W,Pc)) :-
   loves(M2,W),
   gender(W,female),
   home(M2,P2),
   home(W,P1),
```

```
   not (P1 == P2),
   place(Pc),
   not (Pc == P1),
   not (Pc == P2).
```

/*
Operation `elope(M2,W)`: character `M2` elopes with character `W`.

- Preconditions: `W` is a woman; `M2` and `W` love each other; `W` and `M2` have different home places; and `W` is currently at a place that is different from her home place and from `M2`'s home place.
- Primary effects: `W` will no longer be at her current place (if it is different from `M2`'s home place); `W` will be at `M2`'s home place; and `W` will consent with `M2`.
- Other effects: `M2` will no longer be at his current place if it is not his home place; and `M2` will be at his home place.

*/

```
operation(elope(M2,W),1).
deleted(current_place(W,Pc),elope(M2,W)) :-
  home(M2,P2), not (Pc == P2).
/deleted(current_place(M2,Pc),elope(M2,W)) :-
  home(M2,P2), not (Pc == P2).
added(current_place(W,P2),elope(M2,W)) :- home(M2,P2).
/added(current_place(M2,P2),elope(M2,W)) :- home(M2,P2).
added(consents_with(W,M2),elope(M2,W)).
precond(elope(M2,W),current_place(W,Pc)) :-
   loves(M2,W),
   loves(W,M2),
   gender(W,female),
   home(M2,P2),
   home(W,P1),
   not (P1 == P2),
   place(Pc),
   not (W == M2),
   not (Pc == P1),
   not (Pc == P2).
```

/*
Operation `rescue(M1,W)`: character `M1` rescues character `W`.

- Preconditions: `W` is a woman; `W` is currently at the home place of another character `M2`, who she does not love; `W` and `M2` have different home places; and `W` is not held by `M1`.
- Primary effects: `W` will no longer be at her current place (if it is different from `M1`'s home place); `W` will be at `M1`'s home place; and `W` will consent with `M1`.
- Other effects: `M1` will no longer be at his current place if it is not `W`'s home place; if `W` is held by `M2`, it will not be true any longer; and `M1` will be at `W`'s place, if it is not already the case.

```
*/
operation(rescue(M1,W),1).
deleted(current_place(W,P2),rescue(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(current_place(M1,P2),rescue(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(held_by(W,M2),rescue(M1,W)) :-
  home(W,P1), home(M2,P2), not (M2 == M1), not (P2 == P1).
added(current_place(W,P1),rescue(M1,W)) :- home(W,P1).
/added(current_place(M1,P1),rescue(M1,W)) :- home(W,P1).
added(consents_with(W,M1),rescue(M1,W)).
precond(rescue(M1,W),(current_place(W,P2), /(not held_by(W,M1)))) :-
   gender(W,female),
   home(W,P1),
   home(M2,P2),
   not loves(W,M2),
   character(M1),
   not (M1 == W),
   not (M1 == M2),
   not (P1 == P2).
```

```
/*
```

Operation `capture(M1,W)`: character `M1` captures character `W`.

- Preconditions: `W` is a woman; `W` is currently at the home place of another character `M2`; and `W` and `M2` have different home places.
- Primary effects: `W` will no longer be at her current place (if it is different from her home place); `W` will be at her home place; and `W` will be held by `M1`.
- Other effects: `M1` will no longer be at his current place if it is not `W`'s home place; if `W` is held by `M2`, it will not be true any longer; and `M1` will be at `W`'s home place, if it is not already the case.

```
*/
operation(capture(M1,W),1).
deleted(current_place(W,P2),capture(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(current_place(M1,P2),capture(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(held_by(W,M2),capture(M1,W)) :-
  home(W,P1), home(M2,P2), not (P2 == P1).
added(held_by(W,M1),capture(M1,W)).
added(current_place(W,P1),capture(M1,W)) :- home(W,P1).
/added(current_place(M1,P1),capture(M1,W)) :- home(W,P1).
precond(capture(M1,W),current_place(W,P2)) :-
   gender(W,female),
   home(W,P1),
   home(M2,P2),
   character(M1),
```

```
   not (M1 == W),
   not (M1 == M2),
   not (P1 == P2).
```

A.2.2 Level-2 Operators

/*

Operation `ride(C,P1,P2)`: character `C` rides from place `P1` to place `P2`.

- Preconditions: `C` is a man that loves a woman `W`, who is currently at place `P2`, different from place `P1` where `C` is.
- Primary effects: `C` will no longer be at `P1`; and `C` will be at `P2`.

*/

```
operation(ride(C,P1,P2),2).
deleted(current_place(C,P1),ride(C,P1,P2)) :- not (P1 == P2).
added(current_place(C,P2),ride(C,P1,P2)) :- not (P1 == P2).
precond(ride(C,P1,P2),current_place(W,P2)) :-
  loves(C,W),
  gender(C,male),
  gender(W,female),
  place(P1),
  place(P2),
  not (P1 == P2).
```

/*

Operation `seize(M,W)`: character `M` seizes character `W`

- Preconditions:
  - First possibility: `M` is a man; `W` is a woman; `W` and `M` are at the same place (which is different from their homes); and `W` is not held by the other man `M2`.
  - Second possibility (betrayal): `M` is a man; `W` is a woman; `W` and `M` are at `M`'s home place (which is different from `W`'s home place); `W` consents with `M` and is not held by the other man `M2`.
- Primary effects: `W` will be held by `M`.

*/

```
operation(seize(M,W),2).
added(held_by(W,M),seize(M,W)).
precond(seize(M,W),
  (/current_place(W,P),
   current_place(M,P),
  not held_by(W,M2))) :-
  gender(M,male),
  gender(W,female),
  gender(M2,male),
  not (M == M2),
  place(P),
```

```
  not home(W,P),
  not home(M,P).
precond(seize(M,W),
  (/current_place(W,P),
   /current_place(M,P),
   /consents_with(W,M),
   /(not held_by(W,M2)))) :-
  gender(M,male),
  gender(W,female),
  gender(M2,male),
  not (M == M2),
  place(P),
  not home(W,P),
  home(M,P).
```

/* Operation `entreat(M,W)`: character `M` entreats character `W`.

- Preconditions: `M` is a man; `W` is a woman; `M` and `W` love each other; `M` and `W` are at the same place (which is different from their home places); and `W` is not held by any man (neither `M` nor `M2`).
- Primary effects: `W` will consent with `M`.

*/

```
operation(entreat(M,W),2).
added(consents_with(W,M),entreat(M,W)).
precond(entreat(M,W),
  (/current_place(W,P),
   current_place(M,P),
   /(not held_by(W,M)),
   not held_by(W,M2))) :-
  loves(M,W),
  loves(W,M),
  gender(M,male),
  gender(W,female),
  gender(M2,male),
  not (M == M2),
  place(P),
  not home(M,P),
  not home(W,P).
```

/*

Operation `carry(M,W,P2)`: character `M` carries character `W` to place `P2`.

- Preconditions:
  - First possibility (gentle case): `M` is a man; `W` is a woman; `W` and `M` love each other; `W` consents with `M`; `W` and `M` are at the same place `P1`; and `P2` is `M`'s home place.
  - Second possibility (forceful case): `M` is a man; `W` is a woman; `M` loves `W`; `W` is held by `M`; `W` and `M` are at the same place `P1`; and `P2` is `M`'s home place.

- Primary effects: W and M will no longer be at the current place; and W and M will be at M's home place.

*/

```
operation(carry(M,W,P2),2).
deleted(current_place(M,P1),carry(M,W,P2)) :-
  home(M,P2), not (P1 == P2).
deleted(current_place(W,P1),carry(M,W,P2)) :-
  home(M,P2), not (P1 == P2).
added(current_place(M,P2),carry(M,W,P2)) :-
  home(M,P2).
added(current_place(W,P2),carry(M,W,P2)) :-
  home(M,P2).
precond(carry(M,W,P2),
  (consents_with(W,M),
  /current_place(W,P1),
  /current_place(M,P1))) :-
  gender(W,female),
  gender(M,male),
  loves(M,W),
  loves(W,M),
  home(M,P2).
precond(carry(M,W,P2),
  (held_by(W,M),
  /current_place(W,P1),
  /current_place(M,P1))) :-
  gender(W,female),
  gender(M,male),
  loves(M,W),
  home(M,P2).
```

/*

Operation defeat(M1,M2): character M1 defeats character M2.

- Preconditions: M1 and M2 are men; a character W is held by M2; M1 loves W; and M1, M2 and W are at the same place.
- Primary effects: W will no longer be held by M2.

*/

```
operation(defeat(M1,M2),2).
deleted(held_by(W,M2),defeat(M1,M2)).
precond(defeat(M1,M2),
  (/current_place(M2,P),
   /current_place(W,P),
   /held_by(W,M2),
   /current_place(M1,P))) :-
   gender(M1,male),
   gender(M2,male),
```

```
  not (M1 == M2),
  loves(M1,W),
  place(P).
```

A.3 THE BEHAVIORAL SCHEMA

A.3.1 Goal-Inference Rules

/* Syntax for specifying goal-inference rules

sit_obj(<E>,<SITUATION>,<GOAL>)
<SITUATION> is a conjunction of literals to be checked against the current state. If <SITUATION> holds, the conjunction of literals <GOAL> might be brought about as a goal for entity <E>. The body of the clause may contain a conjunction of nonvarying literals.
*/

/*
Rule for the ravisher: The home place of the ravisher M2 is P2. If a woman W is at place P3 (different from P2 and from her home place P1) and the other man M1 is not at P3, then M2 will want to change W's current place to P2.

*/

```
sit{\_}obj(M2,
  (current_place(W,P3),not current_place(M1,P3)),
  (current_place(W,P2))) :-
  gender(M1,male),
  gender(M2,male),
  gender(W,female),
  not (M1 == M2),
  place(P3),
  not home(P3,_),
  home(W,P1),
  home(M2,P2),
  not (P1 == P2).
```

/*
Rule 1 for the protector: Protector M1 has the same home place P1 of a woman W. If W is at the home place P2 of the other man M2, and M1 believes that W does not love M2, M1 will want to bring W back to P1 but without capturing her.
*/

```
sit_obj(M1,
  (current_place(W,P2), believes(M1,not loves(W,M2))),
  (current_place(W,P1), not held_by(W,M1))) :-
  gender(M1,male),
  gender(M2,male),
  gender(W,female),
  not (M1 == M2),
  home(M1,P1),
```

```
  home(W,P1),
  home(M2,P2),
  not (P1 == P2).
/*
```

Rule 2 for the protector: Protector M1 has the same home place P1 of a woman W. If W is at the home place P2 of the other man M2, and M1 believes that W loves M2, M1 will want to capture W and bring her back to P1.

```
*/
sit_obj(M1,
  (current_place(W,P2), believes(M1,loves(W,M2))),
  (current_place(W,P1), held_by(W,M1))) :-
  gender(M1,male),
  gender(M2,male),
  gender(W,female),
  not (M1 == M2),
  home(M1,P1),
  home(W,P1),
  home(M2,P2),
  not (P1 == P2).
```

A.3.2 Default Beliefs (Only for the Protector)

```
/* In the following predicates, a state is represented by the sequence operations
starting from the given initial state */
/* A character C believes at any state that literal F holds if it is explicitly declared
*/
believes(C,F,_) :-
  believes(C,F).
/*
```

A character C believes that literal F holds at state S if this literal can be inferred and there is no explicit declaration about F (or not F).

```
*/
believes(C,F,S) :-
  belief(C,F,S),
  not added_belief(C,F).

added_belief(C,F) :-
  (F = (not F1),Fe = F1,!;
   Fe = F),
  (believes(C,Fe);
   believes(C,not Fe)).
/*
```

Protector M1 believes at state S that woman W loves the other man M2, either if W and M2 were together at M2's home place and W was not held by M2 or if W was held by M1.

```
*/
```

```
belief(M1,loves(W,M2),S) :-
  gender(M1,male),
  gender(M2,male),
  not (M1 == M2),
  gender(W,female),
  home(M2,P2),
  (once((current_place(W,P2),
         current_place(M2,P2),
         not current_place(M1,P2),
         not held_by(W,M2)),S),!;
  holds(held_by(W,M1),S)).
```

/*
Protector M1 believes at state S that woman W does not love the other man M2 if W and M2 were together at M2's home place and W was held by M2.
*/

```
belief(M1,not loves(W,M2),S) :-
  gender(M1,male),
  gender(M2,male),
  not (M1 == M2),
  gender(W,female),
  home(M2,P2),
  once((current_place(W,P2),held_by(W,M2)),S),
  not holds(held_by(W,M1),S).
```

/* This predicate lists all current beliefs */

```
beliefs :-
  log(L),
  forall(believes(A,F,L),describe(believes(A,F,L))).
```

/* Predicate believes(_,_) is a user controlled predicate. In this way, characters' beliefs may change by means of directives */ `user_controlled(believes(_,_)).`

A.3.3 Feelings at an Indicated State

/* This predicate lists the current emotional condition of each character */

```
emotional_condition :-
log(L),
forall(character(C),
  (once(emotional_condition(C,S,L)),
  describe(emotional_condition(C,S,L)))).

emotional_condition(C,S) :-
  emotional_condition(C,S,start).
```

/* A woman is absolutely happy if she has never been held by anyone */

```
emotional_condition(C1,absolutely_happy,S) :-
  gender(C1,female),
```

```
  emotional_condition(C1,happy,S),
  not (state(Si,S), holds(held_by(C1,_),Si)).
/*
```

A character C1 is happy at state S if he or she loves a character C2, C1 is not held by another character C3, and C1, and C2 are together at the same place.

```
*/
emotional_condition(C1,happy,S) :-
  character(C1),
  loves(C1,C2),
  not (character(C3),
       not (C3==C2),
       holds(held_by(C1,C3),S)),
  once((holds(current_place(C1,P),S),
        holds(current_place(C2,P),S))).
```

/* A character C is bored at state S if he or she is not happy */

```
emotional_condition(C,bored,S) :-
  character(C),
  not emotional_condition(C,happy,S).
```

A.4 EXAMPLE INITIAL STATE

A.4.1 Fixed Properties

```
place(forest).
place(palace).
place('Lanka').
character('Sita').
gender('Sita',female).
home('Sita',palace).
character('Rama').
gender('Rama',male).
home('Rama',palace).
character('Ravana').
gender('Ravana',male).
home('Ravana','Lanka').
```

A.4.2 Varying Properties

```
current_place('Sita',forest).
current_place('Rama',palace).
current_place('Ravana','Lanka').
loves('Rama','Sita').
loves('Ravana','Sita').
loves('Sita','Rama').
```

A.5 LIBRARY OF TYPICAL PLOTS

```
/*
```
Syntax for Specifying Typical Plots

`ind(<E>,<SITUATION>,<GOAL>,<OPERATIONS>)`
A typical plot for entity `<E>` when situation `<SITUATION>` holds and `<E>` wants to achieve goal `<GOAL>` corresponds to the sequence of operations `<OPERATIONS>`.
```
*/

/*
```
When a man `M` loves a woman `W` whose home place `Pw` is different from his home place `Pm` and he wants to bring her to `Pm`, a typical plan could be the following: `M` rides from `Pm` to `Pw`, `M` entreats `W`, `M` seizes `W` and `M` carries `W` to `Pm`.
```
*/

```
ind(M,
     (gender(M,male),loves(M,W),loves(W,M),home(M,Pm),
      current_place(W,Pw), not (Pw==Pm)),
     (current_place(W,Pm)),
     start=>ride(M,Pm,Pw)=>entreat(M,W)=>seize(M,W)=>carry(M,W,Pm) ).
/*
When a man M loves a woman W whose home place Pm is the same as his, she is at another place Pw and he wants to bring her back to Pm, a typical plan could be the following: M rides from Pm to Pw, M seizes W and M carries W to Pm.
*/
ind(M,
    (gender(M,male),loves(M,W),loves(W,M),home(M,Pm),home(W,Pm),
     current_place(W,Pw), not (Pw==Pm)),
    (current_place(W,Pm)),
    start=>ride(M,Pm,Pw)=>seize(M,W)=>carry(M,W,Pm) ).
```

A.6 MOTIFS

```
/* Syntax for Specifying Motifs
```

`motif(<MOTIF_NAME>,<CHARACTERS>,<SITUATION>,<SITUATION_SPEC>,<GOAL>, <GOAL_SPEC>)`
Motif with name `<MOTIF_NAME>`, involving characters in `<CHARACTERS>`, can be applied when from situation `<SITUATION>`, specified by `<SITUATION_SPEC>`, it is necessary to achieve goal `<GOAL>` specified by `<GOAL_SPEC>`. The body of the clause may contain a conjunction of nonvarying literals.
```
*/

/*
```
Motif Life Token

- Situation: character `C1` is in distress because `C1` is held by character `C3` at some place `P3` different from `C1`'s home place `P1` and there is a character `C2` who loves `C1`.
- Goal: `C1` will need to find a way to call `C2` for help, so that `C2` comes to `P3`.
```

```
*/
motif(  life_token,
        (C1,C2,C3),
        in_distress(C1),
        (held_by(C1,C3), current_place(C1,P3),  current_place(C3,P3),
           loves(C2,C1)),
        calling_for_help(C1,C2),
        (current_place(C2,P3), not current_place(C2,P1))) :-
              character(C1), character(C2), character(C3),
              home(C1,P1),
              home(C2,P1),
              place(P3),
              not (P3 == P1).
```

/*
Motif Love Potion

- Situation: character C1 is indifferent to character C2 because they are both at the same place, but C1 does not love C2.
- Goal: C1 will melt for C2 (i.e., C1 will love only C2)

```
*/
motif(  love_potion,
        (C1,C2,C3),
        indifferent_to(C1,C2),
        (current_place(C1,P),current_place(C2,P),not loves(C1,C2)),
        melts_for(C1,C2),
        (loves(C1,C2), not loves(C1,C3))) :-
              character(C1),
              gender(C1,G1),
              character(C2),
              gender(C2,G2),
              gender(C3,G2),
              not (C3 == C2),
              not (G1 == G2),
              place(P),
              not home(C2,P).
```

/*
Motif Ordeal

- Situation: characters C1 and C2 have the same home place P1 and character C3 has a different home place P3. C1 is under suspicion because character C2 loves C1, C1 and C2 are at the same place and C2 does not believe that C1 loves C2.
- Goal: C1 will be vindicated (i.e., C2 will believe that C1 loves only C2)

```
*/
motif(  ordeal,
        (C1,C2,C3),
        under_suspicion(C1),
        (
           loves(C2,C1), current_place(C1,P1),
           current_place(C2,P1), not believes(C2,loves(C1,C2) ),
           not (C3 == C2)
        ),
        vindicated(C1),
        (
           believes(C2,not loves(C1,C3)),
           believes(C2,loves(C1,C2))
         ))  :-
                character(C1), character(C2), character(C3),
                home(C1,P1),
                home(C2,P1),
                home(C3,P3),
                not (P3 == P1).
```

REFERENCES

Aarne, A. and Thompson, S. 1987. *The Types of the Folktale*. Suomalainen Tiedeakatemia.

Aristotle. 2000. Poetics. In *Classical Literary Criticism*. P. Murray et al. (trans.). Penguin.

Bal, M. 2002. *Narratology*. University of Toronto Press.

Barros, L. and Musse, S. 2007. Planning algorithms for interactive storytelling. *ACM Computers in Entertainment 5*, 1.

Batini, C., Ceri, S., and Navathe, S. 1992. *Conceptual Design—An Entity-Relationship Approach*, Benjamin Cummings.

Béroul. 1970. *The Romance of Tristan*. A.S. Fedrick (trans.), Penguin.

Booth, W. 1974. *A Rhetoric of Tropes*. University of Chicago Press.

Breitman, K. K., Barbosa, S. D. J., Casanova, M. A., and Furtado, A. L. 2007. Using analogy to promote conceptual modeling reuse. In *Proceedings of the Workshop on Leveraging Applications of Formal Methods, Verification and Validation.*

Breitman, K. K., Casanova, M. A., and Truszkowski, W. 2007. *Semantic Web*. Springer.

Burke, K. 1969. *A Grammar of Motives*. University of California Press.

Camanho, M., Ciarlini, A. E. M., Furtado, A. L., Pozzer, C. T. and Feijó, B. 2008. Conciliating coherence and high responsiveness in interactive storytelling". In *Proceedings of the Third ACM International Conference on Digital Interactive Media in Entertainment and Arts* (DIMEA'08). ACM, New York.

Casanova, M. A., Barbosa, S. D. J., Breitman, K. K., and Furtado, A. L. 2008. Generalization and blending in the generation of entity-relationship schemas by analogy. In *Proceedings of the Tenth International Conference on Enterprise Information Systems* (ICEIS).

Cavazza, M., Charles, F., and Mead, S. 2002. Character-based interactive storytelling. *IEEE Intelligent Syst. 17*, 4.

Chandler, D. 2007. *Semiotics: The Basics*. Routledge.

Chrétien de Troyes. 1983. *Le Chevalier de la Charrete*. M. Rocques (ed.), Honoré Champion.

Ciarlini, A. E. M., Pozzer, C. T., Furtado, A. L., and Feijó, B. 2005. A logic-based tool for interactive generation and dramatization of stories. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*. ACM, New York.

COHEN, P. R. AND LEVESQUE, H. J. 1990. Intention is choice with commitment. *Artificial Intelligence 42*, 213–261.

COSTIKYAN, G. 2002. I have no words and I must design: Toward a critical vocabulary for games. In *Proceedings of the Computer Games and Digital Cultures Conference.*

CRAWFORD, C. 1984. *The Art of Computer Game Design*. Osborne-McGraw-Hill.

CULLER, J. 1983. *On Deconstruction: Theory and Criticism after Structuralism*. Cornell University Press.

FAUCONNIER, G. AND TURNER, M. 2002. *The Way We Think*. Basic Books.

FIKES, R. E. AND NILSSON, N. J. 1971. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence 2*.

FURTADO, A. L., CASANOVA, M. A., BARBOSA, S. D. J., AND BREITMAN, K. K. 2008. Analysis and reuse of plots using similarity and analogy. In *Proceedings of the 27th International Conference on Conceptual Modeling* (ER).

FURTADO, A. L. AND CIARLINI, A. E. M. 2000. Generating narratives from plots using schema information. In *Proceedings of the Fifth International Conference on Applications of Natural Language to Information Systems*.

FURTADO, A. L. AND CIARLINI, A. E. M. 2001. Constructing libraries of typical plans. In *Proceedings of the 13th International Conference on Advanced Information Systems Engineering*.

FURTADO, A. L. AND VELOSO, P. A. S. 1996. Folklore and myth in the knight of the cart. *Arthuriana 6*, 2.

GRASBON, D. AND BRAUN, N. 2001. A morphological approach to interactive storytelling. In *Proceedings of CAST01, Living in Mixed Realities*. Special issue of Netzspannung.org/ journal, the Magazine for Media Production and Intermedia Research.

HEARST, P. C. AND MOSCOW, A. 1988. *Patty Hearst: Her Own Story*. Avon.

HOMER. 1950. *The Iliad*. E. V. Rieu (trans.), Penguin.

KAUTZ, H. A. 1991. A formal theory of plan recognition and its implementation. In *Reasoning About Plans*, J. F. Allen et al. (eds.), Morgan-Kaufmann.

LAKOFF, G. AND JOHNSON, M. 1980. *Metaphors We Live By*. University of Chicago Press.

LLOYD, W. 1987. *Foundations of Logic Programming*. Springer.

MARCHELLO-NIZIA, C. 1995. *Tristan et Yseut*. Gallimard.

MATEAS, M. AND STERN, A. 2000. Towards integrating plot and character for interactive drama. In *Proceedings of the AAAI Fall Symposium on Socially Intelligent Agents: The Human in the Loop*.

MCGARRY, M. (ED.). 1979. The story of Deirdre. In *Great Folk Tales of Ireland*, Frederick Muller.

MONTFORT, N. 2006. Natural language generation and narrative variation in interactive fiction. In *Proceedings of the AAAI Workshop* on *Computational Aesthetics.*

ORTONY, A. (ED.). 1996. *Metaphor and Thought*. Cambridge University Press.

OVID. 1986. *Heroides and Amores*. G. Showerman (trans.), Harvard University Press.

PEARCE, C. 2002. Emergent authorship: The next interactive revolution. *Computers & Graphics 26*.

PROPP, V. 1968. *Morphology of the Folktale*. S. Laurence (trans.), University of Texas Press.

RAO, A. S. AND GEORGEFF, M. P. 1991. Modeling rational agents within a BDI-architecture. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*.

RIEDL, M. AND YOUNG, R. M. 2004. An intent-driven planner for multi-agent story generation. In *Proceedings of the International Conference on Autonomous Agents and Multi Agent Systems*.

SAUSSURE, F. 2006. *Cours de Linguistique Générale*. Payot.

TITUS LIVIUS. 1919. *History of Rome*, vol I. B. O. Fster (trans.), Harvard University Press.

TRUONG, K. N., HAYES, G. R. AND ABOWD, G. D. 2006. Storyboarding: An empirical determination of best practices and effective guidelines. In *Proceedings of the Sixth Conference on Designing Interactive Systems*.

TURNER, S. R. 1992. Minstrel: A computer model of creativity and storytelling. Tech. Rep. UCLA-AI-92-04.

VALMIKI. 1999. *Le Ramayana*. Philippe Benoît et al. (trans.), Gallimard.

VIRGIL. 1994. *Eclogues, Georgics, Aeneid*. H.R. Fairclough (trans.), Harvard University Press.

WALLIS, J. 2008. *Making Games that Make Stories*. www.electronicbookreview.com. Electronic book review.

WARDRIP-FRUIN, N. AND HARRIGAN, P. (EDS.). 2004. *First Person: New Media as Story, Performance, and Game*. The MIT Press.

WARREN, D. H. D. 1974. *WARPLAN: A System for Generating Plans*. Dept. of Computational Logic, memo 76. University of Edinburgh.

WHITE, H. 1973. *Metahistory: The Historical Imagination in Nineteenth-Century Europe*. The Johns Hopkins University Press.

Received January 2009; accepted June 2009