A COMPUTATIONAL APPROACH TO ELICITING AND MODELING STORIES WITH SOCIAL INTERACTIONS

by

Sergio Antonio Pino Gallardo

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Master of Science in Electrical and Computer Engineering

Fall 2013

A COMPUTATIONAL APPROACH TO ELICITING AND MODELING STORIES WITH SOCIAL INTERACTIONS

by

Sergio Antonio Pino Gallardo

Approved: _____

Kristina Winbladh, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: _____

Kenneth E. Barner, Ph.D.
Chair of the Department of Electrical and Computer Engineering

Approved: _____

Babatunde A. Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____

James G. Richards, Ph.D.
Vice Provost for Graduate and Professional Education

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Prof. Kristina Winbladh, who guided, encouraged, and supported me in my research and personal grow. Under her guidance, I acquired a vast set of skills and knowledge. Also, I would like to thank my coworkers for their support during the development of this project.

Special thanks to my beloved wife Irene for her support and understanding in this journey, to my mom for helping me to become the person whom I am, to my dad for his support and love, and to my grandparents for their love.

Finally, I am thankful for the help and support that I have received from the staff and faculty of the Electrical & Computer Engineering Department of the University of Delaware.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

In this work, we explored a middle point between the common approach of sharing knowledge using oral narratives and the automatic approach of generation of narratives by computational tools. We describe a new model to capture the flow of events of a story and the changes in the social interactions between the characters as the story progresses. We present a description of the software implementation of the model. Also, we provide a discussion about the design and implementation of the computational tool used to capture stories with the proposed model. The main motivation of this work lies on the fact that each domain of knowledge has people with a wide range of experience and expertise in that domain. Also, within each domain people use narratives as an effective medium of transferring their knowledge. A crucial point is that individuals of a domain have the potential of making valuable contributions to the body of knowledge and that those contributions are often driven in a narrative fashion. In addition, as interactive entertainment continues its role as a pervasive element of today's culture, the potential of a meaningful experience through the use of narratives can only be achieved if there are tools that aid a wider audience in the creation of those narratives. The AI community has recognized the overwhelming task of authoring stories for interactive entertainment, which demands expertise in computational models for the structure of the story and its execution, as well as expertise in creating the content of the story. Thus, establishing a computational representation of narratives that aid individuals to share their experience and expertise could open the door for capturing the underlying knowledge of a domain.

Chapter 1

INTRODUCTION

Each domain has people with a wide range of experience and expertise in that domain. Within a domain, it is a common practice for people to rely on narratives as a way to communicate their knowledge. As a result, narratives have been established an effective medium of transferring knowledge. For example, a Xerox PARC study of technicians [1] shows that the diagnosis process and the distribution of technical knowledge are accomplished within the technicians by means of oral narratives. This example reveals that narratives through storytelling are one of the ways individuals make contributions to the body of knowledge of a domain. Turner describes the problem with the storytelling process as an apparent simple task. On its surface, this task is composed of a simple form that can be captured by a formalism. However, it has an underlying content that is a complex web of author goals, reader expectations, and cultural knowledge [2]. A crucial point is that individuals of a domain have the potential of making valuable contributions to the body of knowledge and that those contributions are often driven in a narrative fashion. Thus, establishing a computational representation of narratives that aids individuals to share their experience and expertise could open the door for capturing the underlying knowledge of a domain.

Next, as interactive entertainment continues its role as a pervasive element of today's culture, the potential of a meaningful experience through the use of narratives can be only be achieved if there are tools that aid a wider audience in the creation of those narratives [3]. There are a number of recent contributions from the AI community that explores the creation of storytelling systems for automatic story generation [4, 5, 6, 7, 8]. These works recognize the overwhelming task of authoring stories for interactive entertainment, which demands expertise in computational models for the structure of

the story and its execution, as well as expertise in creating the content of the story. In general, they try to model the knowledge and processes used to tell a story [9], and by creating a computational tool they use the modeling in order to generate the narratives.

The main motivation of this work is based on the fact that each domain of knowledge has people with a wide range of experience and expertise in that domain. In fact, a crucial point is that individuals of a domain have the potential of making valuable contributions to the body of knowledge and that those contributions are often driven in a narrative fashion. Although many people find it easy to relate to narratives, they contain ambiguities and unstated assumptions. It is therefore important to establish a computational representation of narratives and provide automated help to capture those narratives in a formal representation. Figure 1.1 shows the concept of a computational tool to capture stories from people in a domain into a formal representation. First, a person introduces their story by interacting with the computational tool. Next, this process results in a formal representation of the story that can be stored for later analysis. Finally, since the stories are captured and then stored, this could open the possibility that other persons in that domain could reuse those stories.

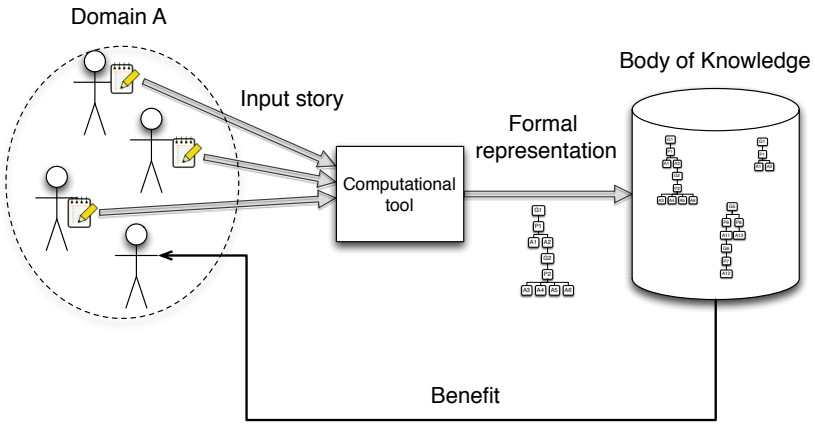

Figure 1.1: An overview of the computer-based tool that helps people of a domain tell their stories and make their contribution directly into a body of knowledge.

In this work, we explore a middle point between the common approach of sharing knowledge using oral narratives and the automatic approach of generation of narratives by computational tools. We describe a new model to capture the flow of events of a story. In addition, the model captures changes in the social relationships between the characters of a story that result from their actions. We describe the rationale for such a model and relate it to previous developments in storytelling systems presented in the literature. We present a description of the software implementation of the model. Also, we provide a discussion about the design and implementation of the computational tool used to capture the user's stories with the proposed model.

The main contributions of this work are: (1) a model to represent the flow of a story and the changes of the social interactions (social relationships) introduced by the actions of story characters; (2) the software design of the model and the details of its implementation in an object-oriented programming language; (3) a computational tool that aids users to create stories in the underlying model.

The remainder of this thesis is organized as follows. In the next chapter, we present a review of the storytelling systems that have been proposed in the literature. Chapters 3 and 4 describe the proposed system for eliciting and modeling stories as event flows with social interactions. Finally, conclusions are drawn from the experience on modeling and implementing the system.

Chapter 2

BACKGROUND

Storytelling appears to be a simple task because, because at a first glance, telling a story seems simple. However, telling a story is a difficult task even for adults, since in addition to the form, there is an underlying story content which contains the meaning of the story. Indeed, it is in the content where the difficulties originate due to the existence of a complex web of author goals, reader expectations, and cultural knowledge that are part of either its creation process or its assumptions. Thus, to understand the complexities of storytelling, it is necessary to understand what an author is trying to achieve. As a result, research with the intention to build a computer program with the capacity of telling stories have determined that it is key to figure out and model the process an author uses to achieve his goals [2].

The effort in storytelling systems (SS) can be decomposed into three main groups: author-centric, story-centric, and character-centric systems. First, author-centric systems try to model the thinking processes of an author. Character-centric systems focus on the modeling of the goals and plans of the characters, so that the stories result from characters pursuing their autonomous goals and plans. Finally, story-centric systems focus on the modeling of the structural properties of the story, as a consequence the system tells stories by manipulating these structural properties [9]. A SS is thus a system that seeks to model the necessary knowledge and process to tell a story, and it is based on a clear-cut definition of the concept of story. To enable further discussion, although within the storytelling systems there is not a standard definition of what a story is, this work uses the definition provided by Gervás [10]: *A story is a highly complex intellectual product that exercises a wide range of the cognitive abilities of humans, involving as it usually does perceptions of time and space,*

attribution of knowledge to particular characters, identifying character goals, validating character plans to achieve the goals, accepting plan failure in the face of obstacles, attributing feelings to characters, associating character intentions with feelings, developing empathy with characters, and including the underlying skill of natural language understanding.

On the other hand, for a story to be satisfactory its elements have to be merged in a complicated manner that is difficult to define. Those elements have to be manipulated and include characters, personality, knowledge that each character possesses, character's goals, character's feelings or emotions, dialogues between the characters, and so on [10].

2.1 Storytelling Systems

The creation of a successful story-generation system requires the consideration of both the intended audience experience and the structures presented for authoring the elements of that experience. Next, the system has to be expressive and controllable so that the underlying model effectively form the audience experience. Also, the system should be authorable by trying to mimic or fake in some feasible way the skills and expectations of traditional authors[11].

2.1.1 Author-centric Systems

In author-centric systems, the task of the storytelling system is approached from the perspective of a human author. Thus, these systems attempt to model the processes undergone by a human author during the creation of a story, whether consciously or otherwise [12].

Ani is an early storytelling system that when presented with a description of a film, attempts to create an animated film based upon that description. In this system, an author presents a partial description of the personality and appearance of the characters involved, of the relationships and interactions among the characters,

and of the type of film desired. Then, the system produces a detailed film description using the previous information and a set of more general knowledge [9, 13].

Lebowitz introduces the Universe author-goal based story generation model, which is based on a Hierarchical Task Network (HTN) style model. The former models the structure of a story as a set of hierarchical plans that describe one or more ways to achieve an author goal in the story [3, 14, 15, 16]. Based on this model, several authoring tools have been proposed, such as Wide Ruled a text-based story planner that aims to provide a friendly authoring tool for non-technical users [3, 5]. Thus, it extends the Universe model by providing an easy to use graphical user interface, support for the author/reader interactivity, and the addition of episodic memory elements to the model. Similarly, Story Canvas is a visual authoring tool for the creation of interactive and generative stories. Story Canvas uses as a framework that integrates, for authoring and interaction, the language of storyboards and comics [17, 18]. This integration provides the user with visual authoring metaphors for all the components of the story generator, since as in the films and computer animation tasks, these techniques provides to the user with a spatio-temporal visualization of the story. As a way to generate the stories, Story Canvas uses the reactive planning language, called A behavior language (ABL), designed specifically for authoring believable agents-characters that express rich personality [19].

Similarly, the system called Minstrel is capable of generating stories that make a point as well as being believable and logically consistent [20]. This system has received attention recently with the introduction of a rational reconstruction of the missing project [6, 21, 22, 23].

Other systems such as Scenejo, which makes special emphasis on the creation of conversational threads for virtual actors using pattern matching, employing transition graph representations as the main interface for authoring. This system prescribes that authors should be able to define the way a conversation develops at any given time [8, 24].

2.1.2 Story-centric systems

In these systems the story-generation proceeds from an abstract representation of the story by modeling the its structural properties. Subsequently, a story-centric system has an approach similar to a context-free grammar, and the discussion centers on an exhaustive analysis of the linguistic attributes of story grammars [12]. Capturing the idea of a story using a grammar-based representation, has the main objective of describing the inter-sentence bindings that arise in simple stories [20, 25].

As an example, the system called Brutus tells stories about a betrayed and a betrayer. In this system, the generation process is incremental since it needs to fill the roles of the characters in a constrained way that ensures the role of Betrayer or Betrayed. Then, the production of a plot requires a hand-creating process for the characters, plans, and events. Finally, the events must be structured into a story grammar for their presentation to the audience [26, 27].

2.1.3 Character-centric Systems

Under the category of character-centric systems, the task of story generation is addressed by the creation of a world, and the characters within it. Then, for each character there are a set of goals and plans so that the stories result from characters pursuing their autonomous goals and plans [12].

Tale-spin, one of the earliest approaches to storytelling systems, simulates a small world of characters and their motivation to act in that world [28]. Tale-spin pursues the previous process by modeling the goals and plans of animal characters [9, 12] and the resulting stories are similar to the fables of Aesop [29].

The Virtual Story Teller has a multi-agent approach in which there is a director agent that looks after the plot. Similarly to Tale-spin, each agent has its own knowledge base (representing what it knows about the world) and rules to govern its behavior. In addition, the director agent has basic knowledge about the plot structure and exercises control over actions performed by other agents. The director exercises control in terms of either environmental actions by introducing new characters and objects, motivational

actions by giving characters specific goals, or prescriptive actions by disallowing a character's intended action. However, the director agent cannot force other characters to perform specific actions [7, 10, 30].

Another system is Fabulist, which splits the narrative generation process into three tiers: fable generation, discourse generation, and media representation. Interestingly, the generation process of the fable uses a planning approach that simultaneously reasons about causality, character's intention, and character's motivation. As a result, the system produces narrative sequences that are causally coherent and have elements of character believability. Moreover, the system introduces specific features that allow the planner to apply modifications to the given input world in order to meet the required goal [10].

Comme il Faut (CiF) is a model that focuses on social interactions and is designed to allow autonomous characters to play social games. This system seeks to represent and reason over social situations and the different behaviors that may expose when different character personalities perform similar roles. In addition to the use of software engineering and computer science techniques, the model incorporates concepts from the areas of sociology, psychology, and knowledge of authoring and drama with the objective of supporting the social artificial intelligence system [4, 31]. Because this novelty, CiF has been used as the basis for the creation of several games, such as the role-playing games RPG [32, 33, 34], and social simulation games [35].

2.2 Support for Human Storytelling

Stories are a key part in the human daily life, for example, Storytelling is a cornerstone in teaching from kindergarten to higher education [36, 37]. Thus, the creation of systems that support people in telling stories to another have been proposed.

Systems like ReQuest, an intelligent story authoring support system that assists a non-expert author in creating meaningful narrative content, allow non-technical users for the authoring of plots. This system uses a natural language-based authoring paradigm in which the story events, states, and character goals are individually written

by the users. In addition, the system exploits this information and generates questions, using the Quest psychological model of question answering [38]. This model simulates the question-answering performance of humans when responding to open-class questions about the narrative content that a hypothetical audience might have about the story in order to help the author to further elaborate on the narrative [39, 40, 41]. In a similar fashion, the Asking Questions and Understanding Answers system (Aqua), uses a question-based theory of explanation, story understanding, and learning. So that it is possible to create a model of a dynamic understander that is driven by its questions or goals to acquire knowledge [42].

2.3 Discussion

As evidence from the previous section, the field of computational storytelling is rich, and the scope of possible research directions is substantial. Thus, having identified these trends from the previous works in the field, we now turn our attention to what is missing. Current trends and expressed needs include a midpoint research between Support for Human Storytelling and Authoring tools for Storytelling Systems; tighter integration between Story Database Systems and support for authoring a story; and the consideration of using stories as a representative of a body of knowledge.

Chapter 3

STORY REPRESENTATION MODELING

In this section, we describe the story representation that was proposed in this work. Our model of story representation is based in a great part on the analysis of the author-centric systems approach presented in the previous chapter. Within the models that were surveyed we found that the Universe author-goal based model created by Lebowitz offered a good starting point from which we can create our model. The key point of the analysis of those models is that by definition their intent is to capture an abstraction of the process that has to be followed to create a story in an automatic fashion. On the other hand, this work focuses on the elicitation of stories from a user rather than on the generation of stories with an automatic computational tool. Although the final application of the surveyed models is different, these models offer from a modeling and computational perspective good insights on how to create a solution that elicit stories from users. Indeed, we wanted to create a solution for the inverse problem of previous approaches, then in a concise form our problem is the capturing of a story from a narrative source (user) into a model that keeps the structure and complexity of the narrative, rather than generating automatically a narrative from a model of a specific type of story. In the same fashion, we extracted ideas for the modeling and capturing of the changes of the social interactions from currently proposed models, to create playable games, that take into account the social complexities among players.

3.1 Model representation for capturing the steps taken in a story

Lebowitz introduces the Universe author-goal based story generation model, which is based on a Hierarchical Task Network (HTN) style model. The HTN models

the structure of a story as a set of hierarchical plans that describe one or more ways to achieve an author-goal in the story [3, 14, 15, 16].

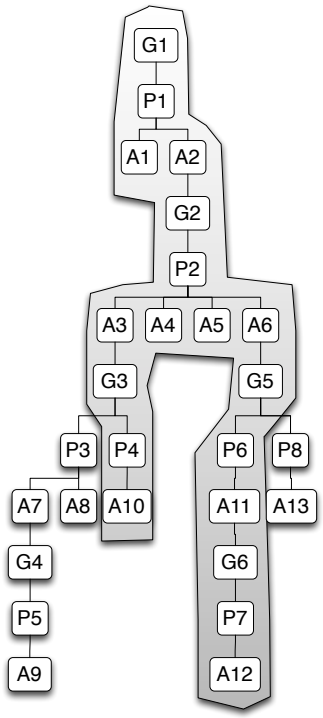

Figure 3.1: A story structure example of a Universe model hierarchy. Where G stands for author-goal, P for plot fragment, and A for the sequential step or action. Also, the highlighted path in the structure represents a single instance of a story which is executed in a top-down fashion.

The Universe author-goal story generation model approach is to model the story structure as a set of hierarchical paths that enclose one or more ways to accomplish a story goal assigned by the author. This model describes a Hierarchical task network (HTN-style) of the structure of the story. As a result, the generation of the story uses a hierarchical arrangement of author-goals, and tasks to accomplish each goal. Where an author-goal represents the story intentions of the author, and the tasks (denoted as plot fragments) consist of ordered steps that can achieve a goal [3]. The hierarchical nature comes from the possibility of having subgoals nested into those steps. Figure 3.1 shows this concept with an example, the outer goal has one plot fragment with two

steps. However, one of the steps has a nested subgoal that has a plot fragment and four steps, and so on. This hierarchical structure makes it possible for the user to create stories with a considerable complexity by nesting the basic building block composed by an author-goal, a plot fragment, and some steps. In terms of the generation, this model uses a randomized approach in which at runtime the program decides from an author-goal what plot fragment (of the pool of plot fragments that meets the input parameters) is going to be generated.

In our example in Figure 3.1, an instance of the story that could be the result of the story generation is as follows. First, let's assume that G1 is "prepare a romantic dinner", P1 is "at home", A1 is "clean kitchen", and A2 is "prepare dinner". Thus, the first level generates G1, P1, A1, and A2. Likewise, A2 has a nested subgoal that constitutes the second level of the hierarchy where the generation is G2, P2, A3, A4, A5, A6. To follow the example, G2 could be "pursue the preparation of the dinner" which expands the step A2 by adding more details. Next, P2 could be "italian recipe" with the steps A3 as "prepare pasta", A4 as "prepare salsa", A5 as "select wine", and finally A6 as "serve the dinner". However, it is important to notice that A3 has a nested subgoal, this means that before executing A4 the system has to go down in the hierarchy of A3 until reaching a leaf step (in this case A10). In our example, G3 could be "pursue preparation of pasta", P3 could be "use home-made pasta", and P4 as "use factory-made pasta". Also, an interesting characteristic of the execution is that for G3 there are two possible plot fragments, P3 and P4, in which the system decides randomly which one to generate. In the example, it is clear that the steps (A7, A8) for P3 are more complex than the steps for P4, because the preparation of homemade pasta is far more complex than preparing a factory-made package of pasta. Similarly, when the generation reaches A6 the system goes down in the hierarchy of the nested goals until A12 is generated. Finally, the story finishes its generation in A6 since is the last step generated.

A formal description of the components of the Universe model could be represented as a class diagram as in Figure 3.2. In its simplest form, the model begins

with the user defined story objects. These objects can be either a character or an environment, and contain associate trait pairs and relationships. Then, we can identify the author-goal as the principal unit of story planning since each goal implies a set of one or more plot fragments that capture a set of steps or actions that fulfill its parent goal. Each plot fragment could have ordered preconditions that contain ordered constraints. These preconditions have to be true in order to consider the plot fragment as a candidate for the generation. Additionally, each plot fragment contains a list of sequential steps that can modify the story objects (characters and environments) or pursue subgoals.

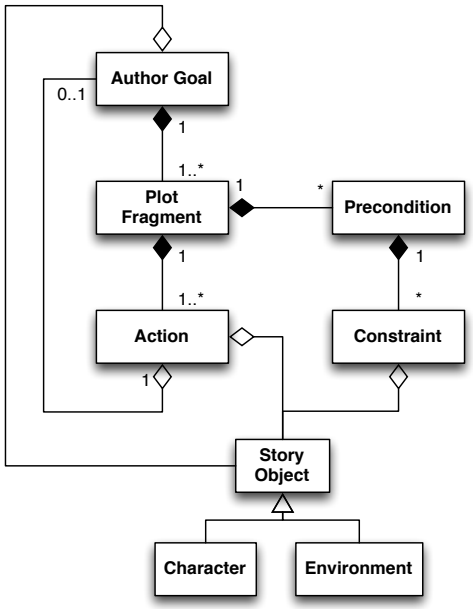

Figure 3.2: Class diagram of the concepts and the relationships between those concepts in the Universe author-goal based model. This diagram represents clearly the hierarchical structure of the model and the life cycle dependencies of the components on it.

3.1.1 Model for capturing the steps taken in a story

In this work, we focused on the creation of a model that could solve the inverse problem with respect to the automatic story generation. In this approach rather than going from the modeling of the story to a generated narrative that a user can read, we

focused on the backwards process. Thus, the high-level idea is to capture a narrative in terms of the components of the model.

For this purpose, we have been guided by assumptions that offer both challenges and opportunities. First, a user will input one story at a time, which is part of his first hand experience with a task, this means that we expect stories of something that did happen. Second, within a domain of knowledge the users will use a common set of concepts in order to describe a situation, their expertise, or their experience in that domain. Third, the stories that a user will input have a clear main goal and a set of actions that the user took to achieve the goal. Finally, as several users input their stories in the system, the accumulated body of knowledge represented by those stories grows and becomes richer.

3.1.1.1 Components of the Model

A story in our model consists of a hierarchical arrangement of basic building blocks called story units. A story unit is composed by one goal, a plot fragment, and at least one action on that plot fragment. As in the case of the Universe model, it is possible to have nested subgoals by creating an action that pursues a subgoal, this explains the hierarchical structure presented in our model. However, in our model each goal can have just one plot fragment, rather than multiple plot fragments as in the Universe model. This enforces the assumption that a story is a situation that did happen and, as a result, there are not possible ramifications to explain how the goal was achieved. Figure 3.3 shows a class diagram representation of the proposed model.

3.1.1.1.1 Goal

The goals in our model share some characteristics with their Universe's counterparts. In both cases, the goals constitute in the primary organizational components of the story. However, in our model just the first created goal can be the initial goal for a particular story, which is not necessary in Universe. Each goal has a name property that can have a weak or a strong semantic depending on the case where the goal is the

initial goal or not. First, in the case of the first goal, the name property has a weak semantic since its intent is to assign the global objective of the story entered in the system, for example, "Cooking a pie", or "Pulling over a car on the highway".

On the other hand, a subgoal has a name property with a semantic that captures specific information about the subgoal. In particular, it captures the data that answer the *"Who?", "Does what?", "to Whom?"*. First, the "Who?" information identifies in the story the character that performs the goal such as "Police officer", "Chef", or "Victor". Second, the "Does what?" is the piece of information that captures an action that the character "Who?" performed such as "prepare salsa". Finally, the "to Whom?" captures a second character who toward the action was performed. In fact, this piece of information could represent either an individual character such as "Driver", a group of characters "crowd", or could be empty.

3.1.1.1.2 Action

Actions are the core of the story elicitation process. They capture the data that provide information about the *"Who?", "Does what?", "to Whom?"*. Thus, the system captures a description of an event that took place in the story by eliciting the previous information. In terms of the semantics, an action has the same semantics as in the case of the subgoals 3.1.1.1.1.

3.1.1.1.3 Plot Fragment

Plot fragments organize, in a *first-input first-output (FIFO)* fashion, the specific actions (steps) that take place to fulfill its parent goal. The FIFO organization preserves the chronological order of the execution of those actions, and also enforces the non-overlapping of them. In addition, a plot fragment may have preconditions that contain constraints that can capture details about the environment, other characters, or other aspects. Taking the "prepare a romantic dinner" story presented in Figure 3.1, it is easy to see that the actions A3, A4, A5, and A6 are taken to fulfill the goal G2 when the plot fragment (P2) is selected. In fact, since P2 restricts the recipe to be an

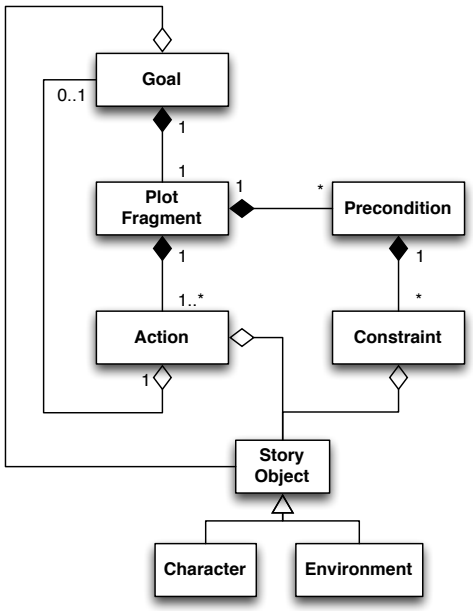

Figure 3.3: Class diagram of the concepts and the relationships between those concepts in Modified Universe Model which is proposed in this work. This diagram represents clearly the hierarchical structure of an actual story in which each author-goal has just one possible plot fragment. This is because each story is an instance of something that did really happen.

italian recipe, then the P2's actions to prepare the dinner are italian-based. Using this approach it is possible to record "why" the actions were taken in the specific way. The preconditions and their constraints have a natural language format that is up to the user of the model. The intuition here is that two different stories can have the same goal and even the same actions, but they could be taken in a different order. Thus, the preconditions can help to capture insights about why the actions were taken in some way and not in another.

3.1.1.1.4 Characters and Environments

Each character conceptually refers to an object that appear in the story and where each one of those objects can have a set of traits represented as attributes. For example, the character "Police officer" can have the traits: age, height, rank, department, and so on. On the other hand, each environment conceptually refers to

those locations where the story took place, and that as well as the characters they have a set of traits. Similarly, environments refer to an object that appear in the story and that have a set of traits. However, while a character takes an active role in the development of the story, an environment is a passive entity that is part of the story.

3.2 Expansion of the model representation to capture the social interaction changes

Recently, some approaches have proposed to construct interactive experiences in video games that enable the possibility of include social aspects in the game experience. A good example of an interactive narrative based game that offers a high level of social play is the experimental game called Façade [19]. Comme il Faut is a more recent work that explores the construction of a playable social model [4, 31], that in its first incarnation used the Goffman's dramaturgical analysis, to encode patterns of normal social behavior as the basis for the creation of an abstraction that enables a social play (social games). The definition of these games is a set of social interactions between characters whose function is to modify the social state existing within and across the characters. Also, this model tries to represent and reason over the social situations that appear as a consequence of the different personalities being placed in similar roles. Another important point is that the model focus on the logic of social statuses and relationships between characters. The social status is an important concept that represents either a boolean relationship between two characters, or a scalar relationship representing the degree of a single character's perception or feeling towards another character [31]. Finally, the system has the concept of "social network". This concept is a metric used to measure the relationship between any two characters, so every character in every net has a link to every other character.

3.2.1 Model support for a story with social interactions

This section presents an expansion of the model presented in section 3.1.1 so that it is possible for a user to introduce, along with the flow of steps in the story,

the information on how those steps could change the social relationships between the characters of the story. The design of the expansion of the model used the assumption that after a character performs one action over another character, there could be a possible change in the social state of the story. Thus, the social interactions, in the model, capture the potential changes produced by the execution of an action in the social interactions (relationships) of the characters with other characters.

3.2.1.1 Components of the model that support the social interaction

In order to incorporate the concept of social interactions into a story, the model has to be modified to include new components that make it possible to handle such stories. At a high level view, the model captures at particular times the social interactions between each character and its peers. In addition, after each action the model can capture the changes that the execution of that action generated in the interactions between characters.

3.2.1.1.1 Social Interaction

A social interaction is the basic concept that encodes the relationships or perceptions of a character with respect to another character. It is an unidirectional recording of the social interaction of character A with respect to character B. This unidirectional restriction is necessary since the two characters can potentially have different perspectives of each other. Also, a social interaction consists of both a temperature that summarizes the social interaction, and a list of social states.

A social state represents a single relationship or perception of character A with respect to character B, and it has a key-value pair representation. The value can be a boolean relationship true or false (e.g. Character A does not know Character B can be described as Character A knows=false Character B). Also, the value can be a numeric relationship that represents the interaction's power with a value between 0 to 5 (e.g. Character A has a big crush with Character B could be represented as Character A love=5.0 Character B). Figure 3.4 shows an example of how the social interactions

can represent an excerpt plot from the novel Les Misérables in [43]. The excerpt is as follows: "After he and Cosette leave, Marius asks Éponine to retrieve her address for him. Éponine, who is in love with Marius herself, reluctantly agrees to do so". This example shows the unidirectional social interaction from Éponine to Marius with two social states. First, the social state "love" captures the fact that she is in love with Marius and the value represents the strength of that perception, that in this case is the maximum. Next, the social state "unenthusiastic" captures the fact that Éponine reluctantly agreed to help Marius after his request.

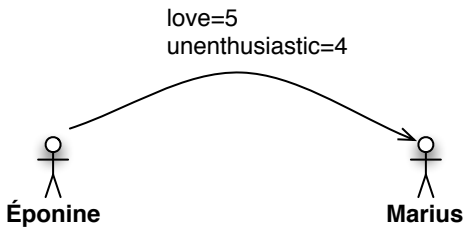

Figure 3.4: Social Interactions from Éponine with respect to Marius after him ask her to retrieve Cosette's address.

In the model, a social interaction is a weighted edge that links together Character A (source) with Character B (target) as in Figure 3.4. In addition, this weighted edge encapsulates the social interaction concept as two components, named a temperature and a list of social states. The weight on the edge represents the temperature of the social interaction with a number. This weight is an optional feature that the implementer of the model can decide on whether to support or not. In fact, if it is providing support for the weight, then it has a computational scheme based on a weighted average of the values of the social states, and can be defined as follows,

$$\bar{t} = \frac{\sum_{i=1}^{n} w_i p_i}{\sum_{i=1}^{n} |w_i|}, \tag{3.1}$$

where p_i is the power in the social states that have a numeric value (a value from 0 to 5). The w_i are the weights for each p_i and represents the importance of the social state in the overall temperature of the social interaction. In addition, the weight's

sign of w_i could be positive or negative which depends on whether or not the concept represented by the social state is a positive or negative concept. This computation can be accesses using the method call *getSocialMeter()* that returns a floating-point number. For example, in Figure 3.4 the social state "unenthusiastic" (state k) has power $p_k = 4$, but since it is a negative concept then its weight w_k should be a negative number that affects the temperature of the interaction. Hence, in this case the weight w_k is a negative value in order to reflect the impact of the concept. Finally, Figure 3.5 shows a class diagram representation of the social interaction concept.

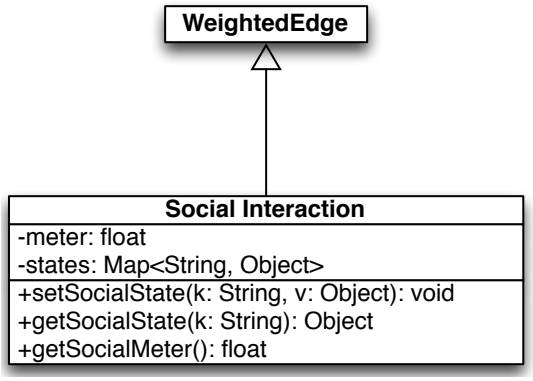

Figure 3.5: Class diagram of a social interaction.

3.2.1.1.2 Social Network

At a particular time, the social interactions between each character and its peers, are a description of the relationships or perceptions of the entire set of characters participating in the story. Since a social interaction encodes the relationships or perceptions between two characters, we can expect that there is a potential social interaction between each character and its peers. Thus, the model captures the different social interactions between the characters as a graph representation that we referred as the Social Network. This network records the different social interactions by linking two nodes (characters) with an edge (a social interaction) when such a link should be

established. This means that two characters have an edge just when the user declares that the story required it so.

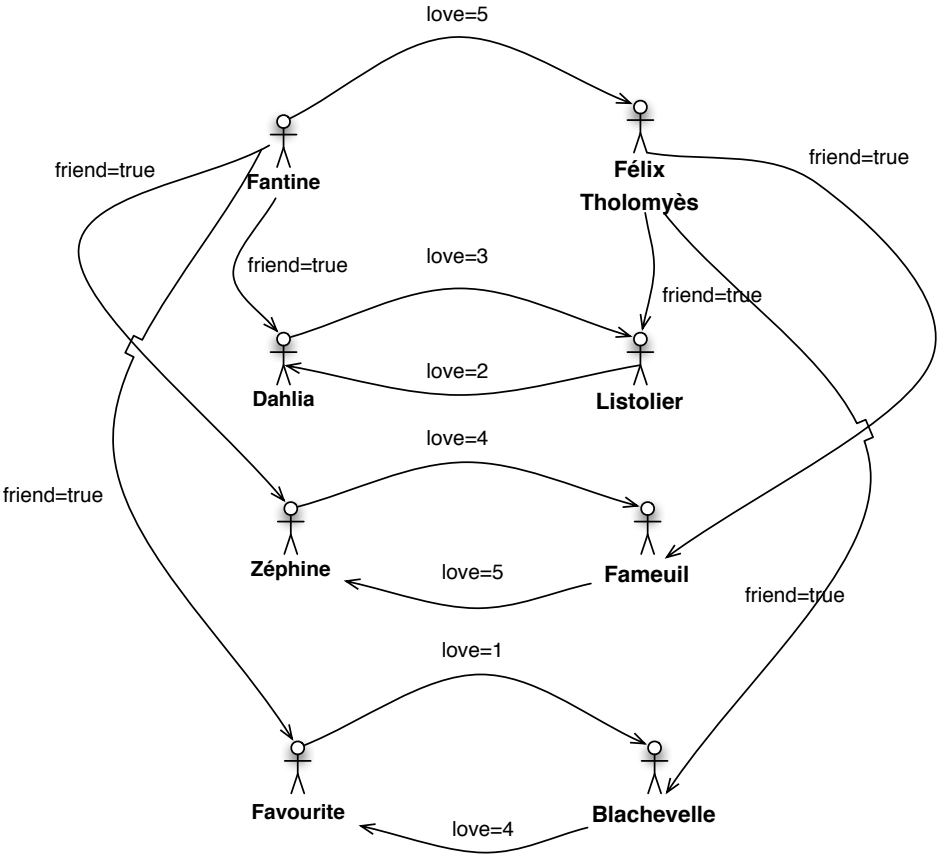

Figure 3.6: Social Network that represents the different social interactions in the excerpt plot. Since, in the excerpt, it is not explicit that Félix Tholomyès is in love with Fantine, then the social interaction between him and her could be left unstated. However, it is from the point of view of the user, which could be imply that he was also in love with her.

Figure 3.6 shows an example of how the social network represents an excerpt from a plot of the novel Les Misérables in [43]. In this plot, the story has eight characters named Fantine, Félix Tholomyès, Listolier, Fameuil, Blachevelle, Listolier, Fameuil, and Blachevelle; and the story dictates the interactions between them. In this case, the story is as follows: "Years earlier in Paris, a grisette named Fantine was very much in love with Félix Tholomyès. His friends, Listolier, Fameuil, and Blachevelle were also paired with Fantine's friends Dahlia, Zéphine, and Favourite". This example

shows the social interactions within the different characters in the story. From the Figure 3.6 it is possible to be aware of the different levels of power of each interaction. This could be helpful to understand how the actions taken before yielded into this social network or how the actions taken after produced certain results. Notice that, in the excerpt, it is not explicit that Félix Tholomyès is in love with Fantine, then the social interaction between him and her could be left unstated. However, is the user who has the discretion of adding or not a social interaction, which in this case could imply that Félix Tholomyès was also in love with Fantine, and as a result, add such social interaction in the social network.

The social network is modeled as a weighted graph, where the nodes are the characters participating in the story, and the edges are the social interactions connecting the two characters (Figure 3.6). The weight of the edges is the temperature of the situation as long as it is available as presented in section 3.2.1.1.1. Finally, Figure 3.7 shows a class diagram representation of the social network concept.

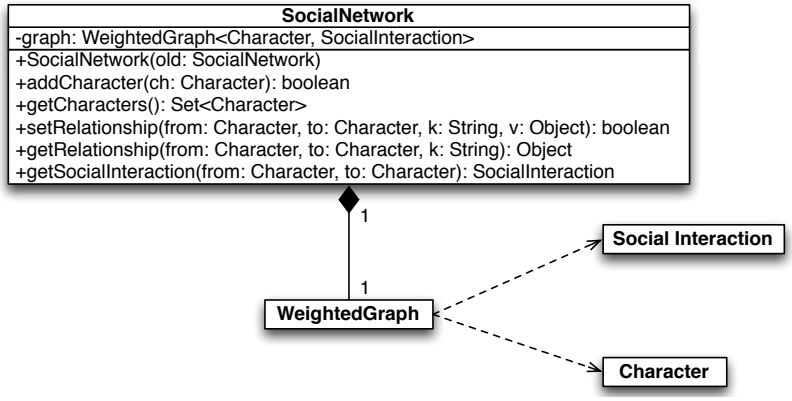

Figure 3.7: Class diagram of a social network.

3.2.1.1.3 Social Network Progress

One of the assumptions that guided the design of the model, states that after a character performed one action over another character there could be a potential change in the social state of the story.

Those actions could change the social interactions between one or more characters in a social network. Thus, if this action changed one or more social interactions between characters, then a new social network will represent the new social relationships. For example, an act of courage from character A to save the life of character B could trigger a change of perception in character B that add the social state "respect=5". Similarly, an act of betrayal from character B to character C can destroy the trust that character C had with respect of character B, by modifying the social state "trust" from value 5.0 to 0.

Thus, the changes in a social network at a given "time" to another social network could be captured using the concept of finite state machine. Since, each social network contains the social interactions between characters at a given "time", then a social network could be considered an analog concept as the state in a finite state machine. In addition, the assumption that an action introduces the changes in the social interactions can be considered analogous to the transition, caused by a trigger, from one state to another. Hence, while a state in the finite state machine represents a social network, a trigger that causes a transition between states represents an action. However, the creation of new social network (state) in this state machine happens if and only if the performed action introduces a modification (creation/update/deletion) in at least one of the social interactions.

Figure 3.8 shows an example of the concept of the social network progress by representing an excerpt of the novel Les Misérables in [43]. This excerpt of the story is a continuation of the one presented in section 3.2.1.1.2 and it is as follows: "The men (Félix Tholomyès) abandon the women (Fantine), treating their relationships as youthful amusements. Fantine must draw on her own resources to care for her and Tholomyès daughter, Cosette". In the example, we made a simplification by focusing on the social interaction between Fantine and Félix Tholomyès. As a consequence, the details of the characters and the social interactions that correspond to Fantine and Félix Tholomyès friend are hidden in the figure. Also, assuming a uniform distribution of the importance (weight w_i) for each social state, that in this example are "love"

and "sad", then (3.1) computes the temperature of interaction with a result of $\bar{t} = \frac{\sum_{i=1}^{n} w_i p_i}{\sum_{i=1}^{n} |w_i|} = \frac{5*0.5+5*-0.5}{|0.5|+|-0.5|} = 0$. In fact, this result of zero in the temperature of the interaction (perception of Fatine towards Tholomyès) can be seen as an overview of what Fantine was feeling after Tholomyès left her alone.

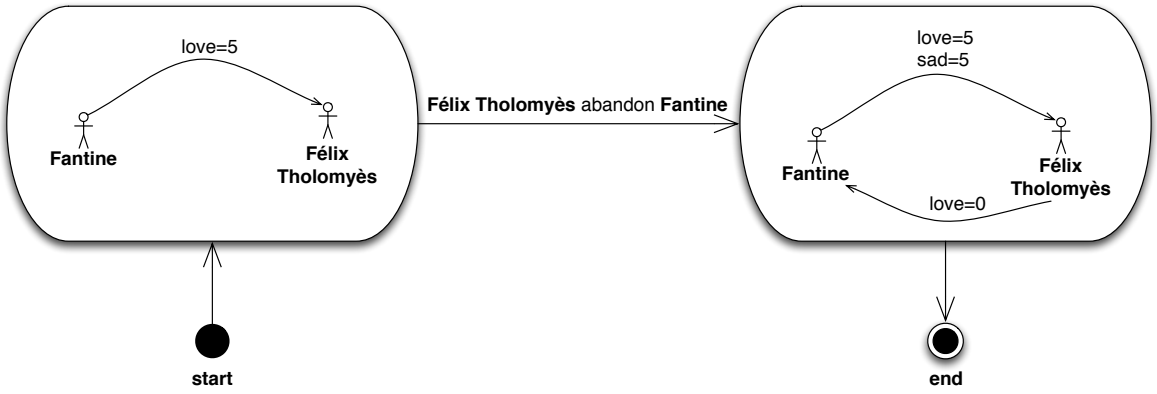

Figure 3.8: Change on the social interactions between Fantine and Félix Tholomyès. The social network after the action was performed captures the change on the social interactions after Félix abandon Fantine. The overall figure represents the social network progress.

Chapter 4

MEASURING TOOL: IMUSE

4.1 Software Architecture

In the design of the elicitation tool, we relied on concepts of system design to help us create a software solution by considering the internal structure of the system, its hardware configuration, and in general how the system can be realized [44]. In this section, we focus on the details of the software architecture of the system. Thus, this section provides a description of the subsystem decomposition in terms of subsystem responsibilities, the dependencies among subsystems, and the control flow of the system. The decisions were driven mainly by the following premises. First, the system decomposition has to be designed in a way where the complexity of the subsystems is such that could be assigned to an individual developer and realized independently. Second, in terms of coupling (number of dependencies between two subsystems) the design should enforce the loose coupling of the different subsystems so that the subsystems could be considered almost independent. Also, in terms of cohesion (number of dependencies within a subsystem) the design should clearly define the boundaries of each subsystem so that the dependencies within the subsystem yields in a high cohesion. Finally, since we anticipate a system with a growing complexity as more requirements are included, the design should use an architectural style that reduces the potential difficulty in modifying or correcting weakness in the system decomposition.

As stated before, this research sought for establishing a computational representation of narratives that aids individuals to share their experience and expertise on a domain. We anticipated that the component that would have more changes in the design was the graphical user interface (GUI), which is the component that supports

the elicitation of those narratives. Thus, the proposed software architecture has the goal of reducing the coupling between the GUI and the implementation of the story model. One approach uses a direct access to the story model subsystem as presented in Figure 4.1. In this approach a subsystem that needs to access the story model (create/read/update/delete) simply issue a method call through the objects from the story model subsystem. Unfortunately, this leads to high coupling between the GUI subsystem and the story model subsystem which is clearly not the goal of this project.

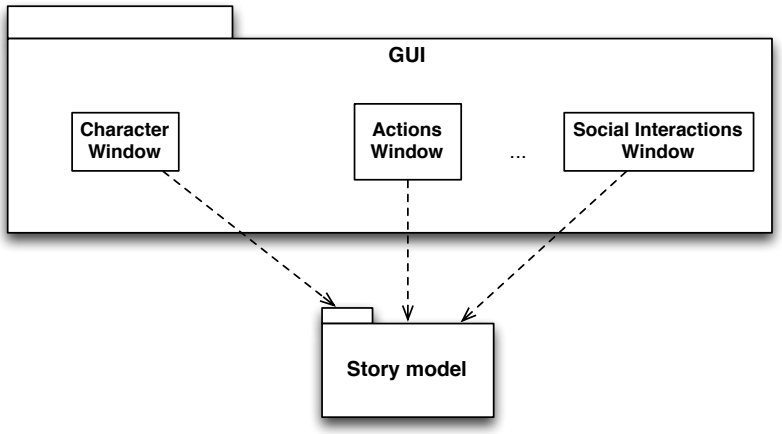

Figure 4.1: Software Architecture when a direct access to the story model is possible from other subsystems.

Hence, we introduce an intermediate subsystem, denoted as *backend*, to reduce the coupling between the *story model* and the *gui* subsystems. This subsystem has the goal of hide the story model subsystem from the other subsystems. In other words, the client subsystems will use the interfaces or services provided by the backend subsystem, which it the one responsible for issuing the method calls to the story model subsystem. For this reason, if a change in the story model occurs, then it is only needed to change the backend subsystem. Hence, this technique decreases the overall coupling of the subsystem decomposition. The proposed software architecture reuses the Model-View-Controller (MVC) architectural style to define how the Graphical user interface subsystem should be implemented. In addition, the architecture defines that the GUI subsystem can access the story model in the following way. First, the model

subsystem of the GUI consumes the services provided by the backend subsystem of the Story representation. Next, the backend deals with the complexities of using the Story model subsystem by consuming the services exposed by that subsystem.

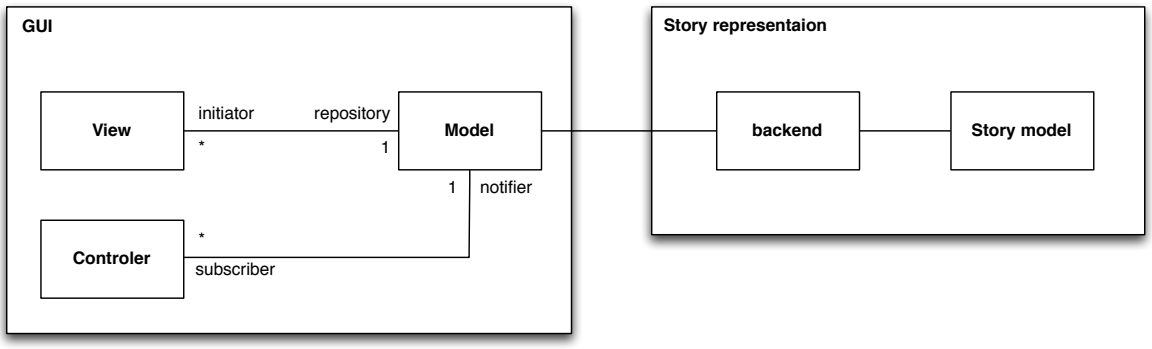

Figure 4.2: Proposed software architecture to reduce the coupling between the GUI and the story model. This architecture provides indirect access to the story model by using the backend subsystem.

Figure 4.2 shows the class diagram of the software architecture. This figure presents two main components named "GUI" and "Story representation". First, as described before the GUI use the MVC architectural style to separate the user interfaces (controller and view) from the domain knowledge (model), since these interfaces are more likely to change than the model. Second, the story representation component exposes the backend as the interface to consume services provided by the story model, and in that way it reduces the dependencies from the user interface and the story model.

Section 4.1.1 presents a discussion of the implementation of the different designs that realize the backend subsystem. Then, section 4.1.2 presents an overview of the development of the different prototypes of the front end graphical user interface. Finally, section 4.2 focus on the description of the final prototype of the user interface that includes the proposed techniques for facilitating the elicitation of the narratives from the users.

4.1.1 Backend

This section describes the different realizations of the backend subsystem and the exposed interfaces that allow client code to consumes the functionality of the subsystem. First, a context-free grammar was designed to represent the objects in the model so that it is possible to input stories by representing them as string generated by the grammar language. Second, a story engine was designed in order to access directly to the model objects without the need of the grammar format.

4.1.1.1 Grammar

As presented in section 3.1.1 the model for capturing the flow of steps in a story can be used for elicitation purposes. Thus, the elicitation process can be considered as a given set, S, of input stories expressed in natural language, and the task is to create an algorithm that maps each story into a hierarchical representation by extracting from the narrative: the Story World elements (Character and Environment), and Goals with their associated Plot Fragments and Actions.

$$G :=< V, \Sigma, P, S > \tag{4.1}$$

Since the model has a hierarchical representation of a story and a well defined set of components, we designed a grammar for expressing a story textually using the concepts presented in the model. We did this so that a story can be composed in the appropriate syntax and then automatically translated into the objects that represent the model. Thus, the definition of the grammar could be stated as: Let be G the context-free grammar designed to express the user's stories. The grammar G is composed of four elements V, Σ, P and S as in (4.1) and the set of rewriting rules P is presented in Grammar 4.1. The alphabet set V is defined based on a subset of the concepts proposed by Lebowitz in [14, 15, 16]. The definition of the set V is $V := \Big\{S, G, PF, PC, A, \lambda\Big\}$, where S denotes the initial axiom (the entering point of the story) used in the either sentence generation or parsing process. Next, G denotes the definition of the basic building block of a story which contains a Goal, a plot fragment, and actions. Finally, PF, PC,

and A are the definitions of a plot fragment, precondition, and actions, respectively. In addition, the alphabet set Σ defines the terminal elements that capture the structure of the story and the need to be easily identify the elements of the model. This set is defined as $\Sigma := \Big\{STORY, (,), GOAL, PLOTFRAG, PRECOND, ACTION\Big\} \bigcup W$, where W is the set of English words.

$\langle S \rangle$ ::= ‘‘`STORY`’’ $\langle W+ \rangle$ $\langle G \rangle$

$\langle G \rangle$ ::= ‘‘`( GOAL`’’ $\langle W+ \rangle$ $\langle PF \rangle$ ‘‘`)`’’

$\langle PF \rangle$ ::= ‘‘`PLOTFRAG`’’ $\langle W+ \rangle$ $\langle PC* \rangle$ $\langle A+ \rangle$ $\langle G+ \rangle$
| ‘‘`PLOTFRAG`’’ $\langle W+ \rangle$ $\langle PC* \rangle$ $\langle G+ \rangle$
| ‘‘`PLOTFRAG`’’ $\langle W+ \rangle$ $\langle PC* \rangle$ $\langle A+ \rangle$

$\langle PC \rangle$ ::= ‘‘`PRECOND`’’ $\langle W+ \rangle$

$\langle A \rangle$ ::= ‘‘`ACTION`’’ $\langle W+ \rangle$

Grammar 4.1: Context-free grammar definition for the elicitation of stories into the elements of the story model.

Using this interface a user that is able to translate stories into the grammar or a client code, can interact with the grammar and capture the story using the proposed model. In this work, there were two prototype implementations that exploit the former use of the grammar as the backend subsystem as presented in Figure 4.3.

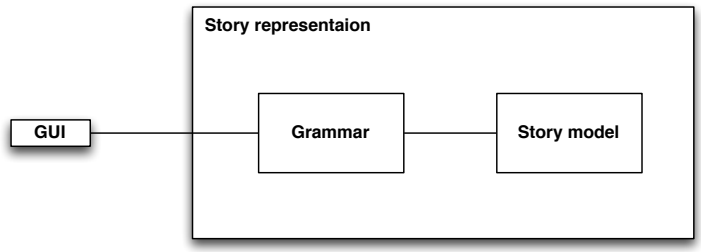

Figure 4.3: Software architecture to reduce the coupling between the GUI and the story model when the backend is implemented using the Grammar.

First, section 4.1.2.1 presents an eclipse plug-in that offers a text editor based solution to input stories directly using the grammar. Second, section 4.1.2.2 discuss the design of a plain writing English based user interface that hide the complexities of the grammar representation and in that way increase the range of users that can interact with the tool. Both implementations rely on the expose of a template method design pattern that identify the framework of the interaction of the grammar with the model as shown in Figure 4.4. This allows the implementation of different classes that define the required behavior of the processing of the story given the user interface selected. The common behavior that we wanted to encapsulate with the pattern was the generation of the model's objects from an input story in the grammar format passed as a raw String.

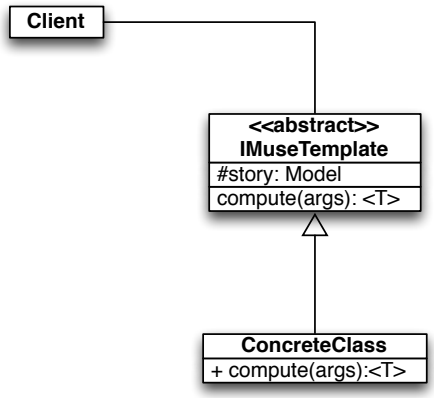

Figure 4.4: Single abstract way to access the model infrastructure. This pattern defines the required behavior for process the model objects and compute the desired representation for example text or a graphical representation.

4.1.1.2 Story Engine

One of the objectives of the backend is dealing with the complexities of using the story model subsystem. Especially, the story model has to be in synchronization with the backend subsystem, such that a change in the story model, reflects in other subsystems through the backend (e.g. the GUI) and vice versa. While the grammar representation relies on a compiler-like translation approach for the synchronization,

the story engine uses an ad hoc approach. The ad hoc approach is built up from the premise that it is possible to breakdown the story model into basic units. This is possible since the model has a hierarchical structure that allows a story to grow by adding a structural pattern composed by one goal, one plot fragment, and at least one action (Figure 4.5 left). In terms of the story engine, this basic unit is denoted as story unit, which conveys the idea that several story units compose a story in this system.

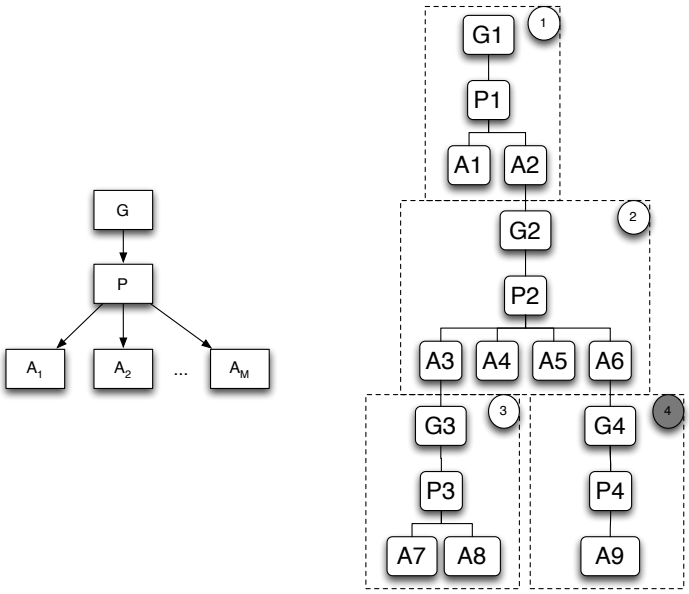

Figure 4.5: Left: shows a story unit structure that is composed by one goal, one plot fragment, and at least one action. Right: example of the breakdown of an instance of the story model into story units.

Figure 4.5 (right) shows an example of a breakdown of a story instance into story units. The story engine manages this set of story units by providing both a way to select a story unit and functionality to modify the content of the current story unit. In the example, the current story unit is the number 4, which means that the modifications (e.g. adding new actions, remove actions, modify the goal, etcetera) target the current story unit. Hence, if a client code of the backend invokes the functionality for adding a new action (A10) this action is added to story unit number 4. Also, a client code could tell the backend to choose a different story unit by invoking the corresponding

functionality. For example, if the story unit number 1 is selected, then adding a new action (A11) modifies story unit 1 rather than story unit 4.

The design of the story engine uses two software design patterns as shown in Figure 4.6. First, the adapter pattern exposes the high level functionality that abstracts thedetailsof the story model. This decoupling allows to introduce changes in the underlying story model without the need of modifying the behavior of the gui. Second, the observer pattern allows to synchronize the story model and the gui by notifying the changes of state of the model directly to the gui.

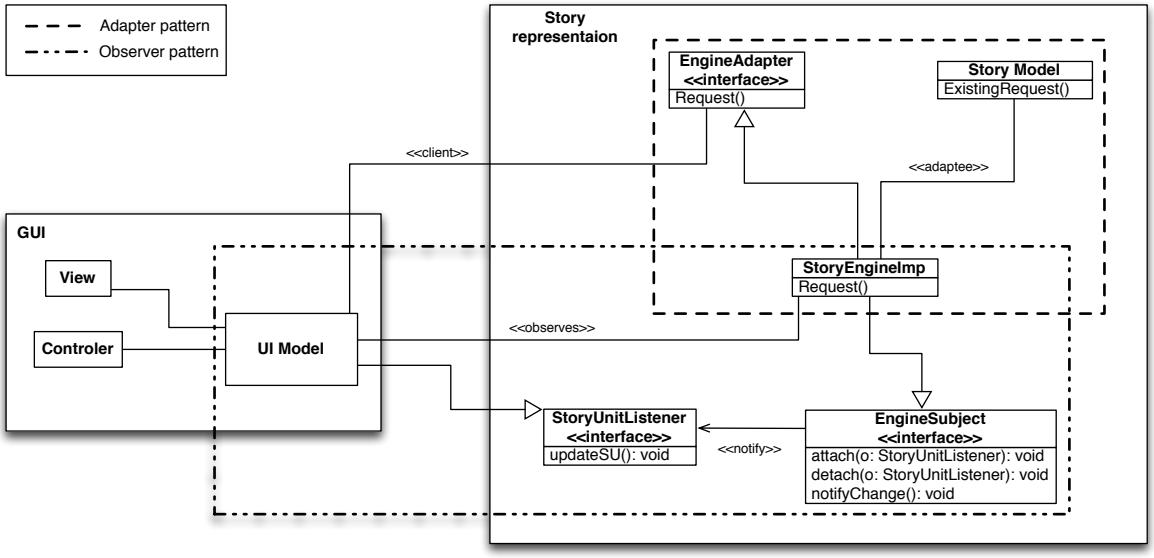

Figure 4.6: Class diagram of the story engine's design using the adapter design pattern as a delegation approach and the observer (listener) pattern as the synchronization mechanism.

4.1.2 Front end prototypes

The design of the front end was an interesting process since it was not clear what a good approach to elicit a story from its narrative form should provide. Thus, during the design process we used some ideas from the Agile development methodology for the design. Specifically, we relied in an iterative approach of development and the

creation of several prototypes to elicit a story. Several prototypes were sketched, and some of them were implemented for proof of concept purposes.

4.1.2.1 Eclipse plugin

The first explorative approach for a front end implementation of the project was an Eclipse-based application. The Eclipse platform provides good opportunities for creating new applications that can take advance of the underlying platform's infrastructure such as the update and help features, and the possibility of reuse other existent plug-ins. Based on the design of eclipse if a new feature needs to be included into the platform, it has to be designed and implemented using the Plug-In specification and the Plug-In Development Environment. For the reuse of the Backend we implemented a new plug-in that provided

1. A new wizard for creating an "iMuse File". Basically, this file will contain a String that represents a story following the iMuse grammar.
2. A multipage editor that will be used to input the story in the grammar format and two tabs, one for display the Universe Model objects and the other will display the Graph representation of the Story.

One of the first changes in order to create the plug-in is to extend one of the classes in the hierarchy of the abstract class org.eclipse.core.runtime.Plugin. Thus, we create the iMuse Activator that solved the requirement of creating the basis of an underlying Plug-in.

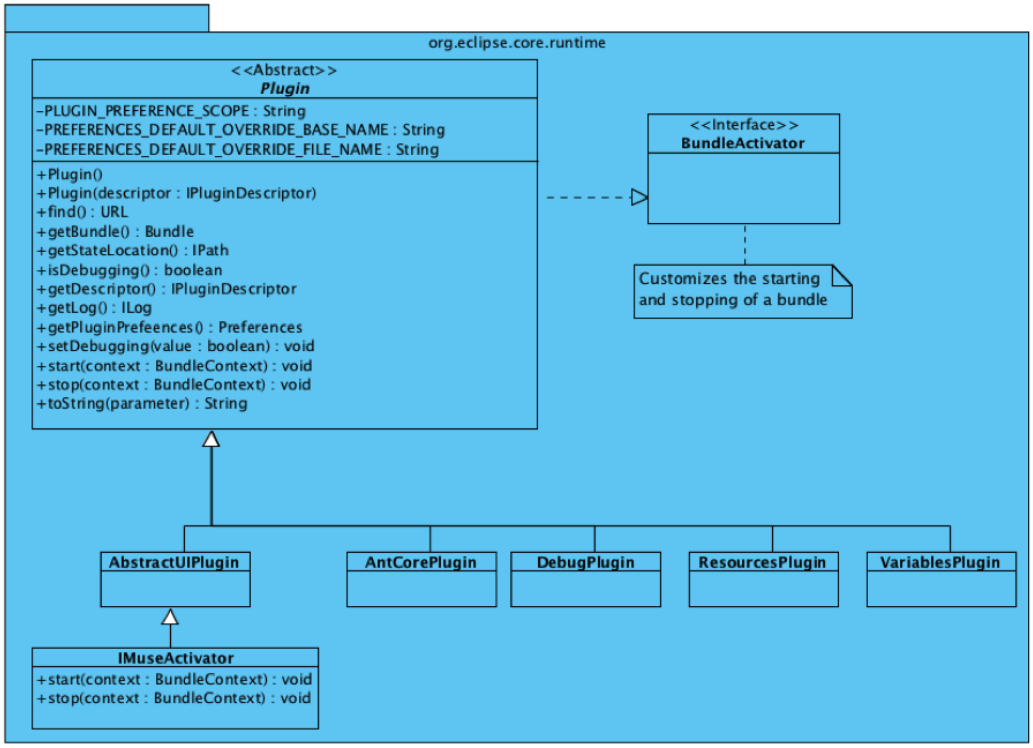

Figure 4.7: iMuse Activator solves the requirement of creating the basis of an underlying Plug-in for the project.

Next, given that we want to reuse an existing project, iMuse, in order to integrate the iMuse library with the plug-in Editor we figured out that the correct answer was to create a subclass of org.eclipse.ui.part.MultiPageEditorPart that has a dependent relationship with the iMuse Library. It means that the iMuseEditor will consume the interfaces provided by the reused library which solves the need of integration of the iMuse project with eclipse. In addition, given the requirement of a Wizard for creating files related to the plug-in, it was required to extend the class org.eclipse.jface.wizard.Wizard and create the iMuseWizard. These are known in eclipse as use of Extension Points because we depend on other plug-ins.

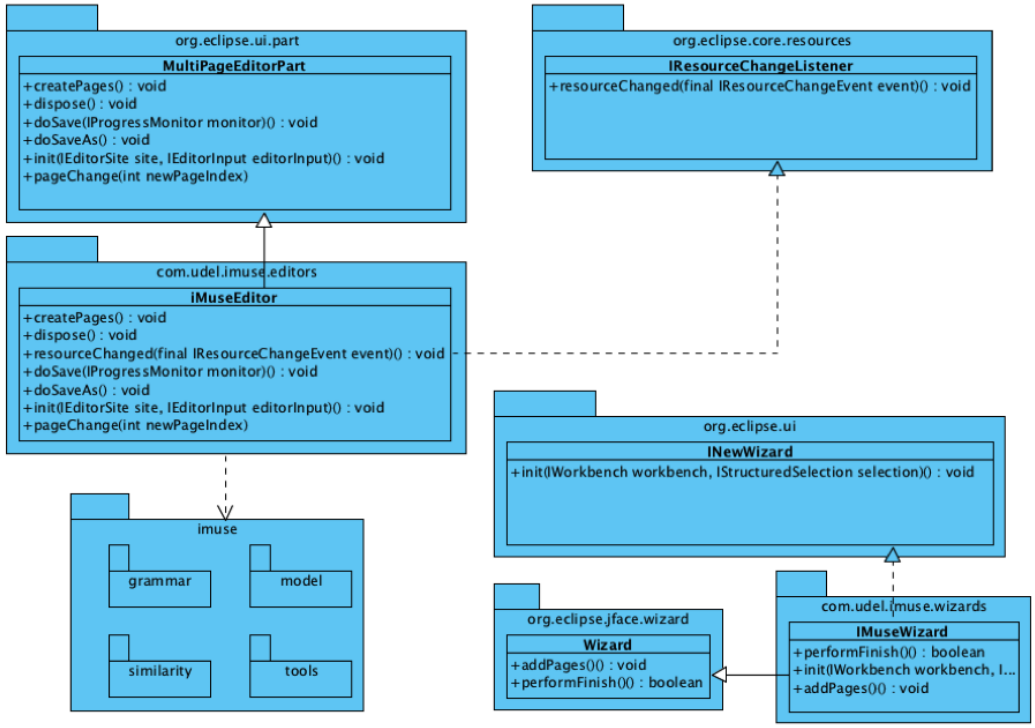

Figure 4.8: Extension Points that represent the dependencies on other plug-ins and resources.

In this implementation, the main editor has four tabs denoted as Text Editor, Universe Objects View, Graph Rep, and Graph View. Where the Text editor provides the input approach to the system through the grammar format presented in 4.1.1.1. The remained tabs provide a different representation of the story. The sample instance presented in Figure 3.1 can be expressed in the grammar format as in Figure 4.9. Also, the graphical representation can be compared with the structure of the one presented in the previous chapter.

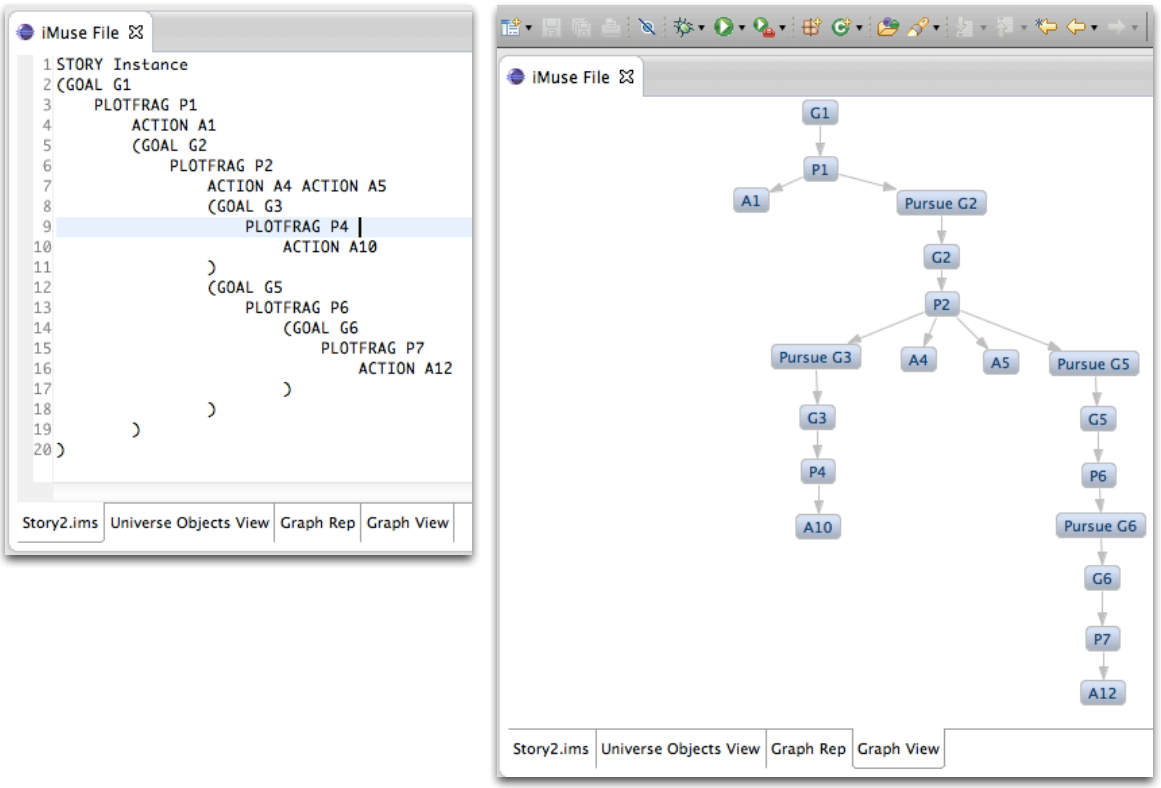

Figure 4.9: Screenshots of the plug-in implemented on eclipse. Left: grammar representation of the hypothetical execution of a Universe model presented in chapter 2. Right: Graphical representation of the story after capturing the model elements from the grammar-based input from the user.

However, this implementation shown to be complicated for a user without a computer science background, since the user has to type in the story directly using the grammar provided by the project.

4.1.2.2 Plain English writing style

The first version that considered the importance of the analyze of the system metaphor and its relationship with the purpose of the elicitation tool was a text editor-based input experience. It was the product of the realization that the system metaphor

should be related to the real world storytelling process. Since, in the real world, it is common to use a plain English writing style, then a good intuition was to design the input system to take advance of this metaphor. The objective of relying on a familiar metaphor for the graphical user interface is to speed up the user understanding on how the system works and how it supports the writing process of the user.

Then, in a concise form the target metaphor is based on ideas presented by Strunk in [45], where a subject of writing (story) requires subdivision into topics, each of which should be made the subject of a paragraph. Where, the beginning of each paragraph is a signal to the reader/writer that a new step in the development of the subject has been reached. In addition, a paragraph is composed of a group of sentences expressing one central idea and is complete itself. In fact, a paragraph has a general structure such as a topic sentence, development of the paragraph and concluding sentence. Finally, given the model and the target stories that we want to collect, the "chronological order" paragraph pattern adapts more to the model since it narrates a sequence of events from beginning to end.

Figure 4.10 presents the prototype implementation representing a partial story about a police routine traffic stop. The user entered the first three sentences of the story where he described the main goal of the story "routine traffic stop", some details about his location and his task at that time. Also, he described actions taken by the driver and himself. Then, he continues developing the story in the next sentences. Thus, as in this example the process of eliciting a story was related to the creation of sentences that describe the story. The system supports this creation by providing to the user with a dedicated panel that present some basic questions *"What?, When?, Where?, Why?, Who?, and How?"* that aid in the construction of the sentence. Then the user add some plain English details like transitions, conjunctions and so on, to increase the readability of the story.

Finally, the system lets the user preview the representation of the story in the underlying elements of the story model as presented in Figure 4.10. The interface worked as a wrapper of the grammar as the backend implementation presented in the

section 4.1.2.1 eliminating in this way the complexities of creating a story directly using the grammar interface. This implementation uses the Java programming language and the Swing framework as the Java GUI widget toolkit.

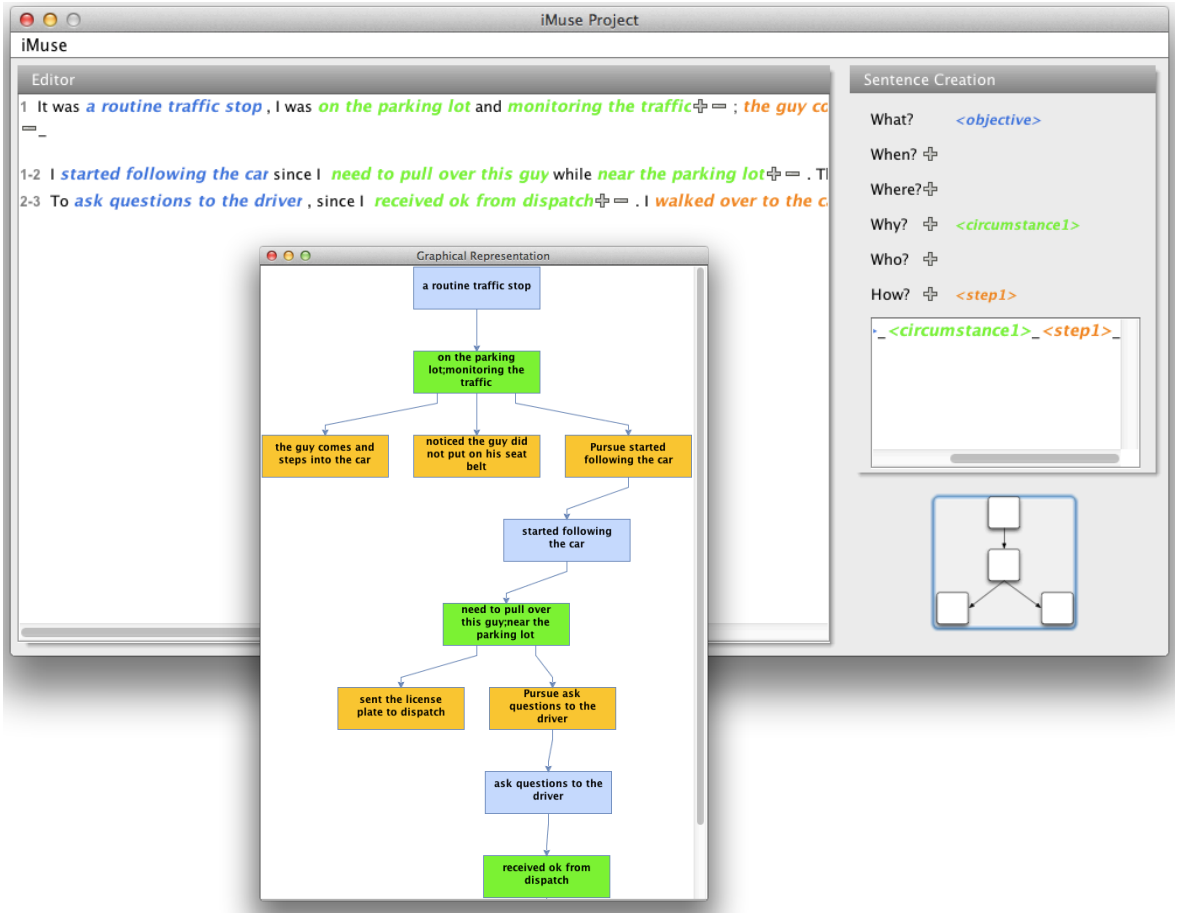

Figure 4.10: Screenshots of the plain English text metaphor used in the design of the text editor based prototype. While the sentence creation is on the right of the GUI, the actual story is on the left of the interface. Also, a graphical representation of the story is displayed as a directed graph that represents the current story.

However, this implementation shown to be impractical for the following reasons.

First, as a story gets more complex, the textual representation gets more complex. Since the model constraints, due to its hierarchical structure, the representation then a user does not have total freedom as he/she has when when using common writing. Thus, if a story has a lot of details about each action, then the resulting hierarchical structure can have several branches that translate into a difficult to read and complex textual representation. Second, although the user has to add plain English details (such as transitions and conjunctions) to increase the readability of the story, the system does not capture these details in the story model, and, as a result, it constitutes in a waste of time and effort from the user side.

4.2 Graphical user interface design

This section describes the latest iteration on the design of the Front end. This implementation addresses most of the issues related to the usability of the graphical user interface that were detected in the previous prototypes. Mainly, those prototypes suffer from the need of familiarity with the underlying model from the user side. As a result, the focus of this prototype was the creation of new metaphors for the graphical user interface to exploit concepts familiar for a wide range of users. First, this system uses a timeline concept (section 4.2.1) to exploit the common approach of representing the events that occur in the passage of the time as a line. Second, since the users are likely to be familiar with the concepts behind the social networks (e.g. Facebook, or Google+), the system uses a social network metaphor (section 4.2.2) for the elicitation of the social interactions between the characters of the story. This prototype uses the story engine 4.1.1.2 as its backend implementation, and it is developed in the JavaFX Rich Client Platform for the Java programming language.

4.2.1 The timeline approach for capturing a story

The elicitation tool relies on the model presented in section 3.1.1 for capturing the flow of actions taken by the characters in a story. By design, the model stores those actions in-order so that it is possible to capture the order in which they occurred in

the story. Hence, it is possible to restate the graphical representation of a hierarchical story presented in Figure 3.1 so that the hierarchy could be drawn as in Figure 4.11.

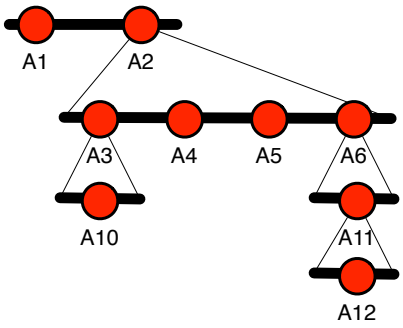

Figure 4.11: Presents an example of the story structure of the proposed model hierarchy using a set of nested timelines. Each A_i represents an action taken in the story and its location represent the chronological order in which was taken. The different levels represent the hierarchical nested structure, which means that a parent action is expanded to provide a detailed description of it. This structure represents a single instance of a story which is captured in a top-down fashion.

This representation of the hierarchical structure resembles a timeline with several nested timelines. Since timelines are particularly useful for studying history, as they convey a sense of change over time [46], this implementation exploits this concept as a metaphor for eliciting the flow of actions in the story. Thus, a user will rely on his familiarity with the timeline concept for inserting in a graphical fashion a story into the system. Figure 4.12 shows the graphical user interface that supports the elicitation of the flow of actions in a story.

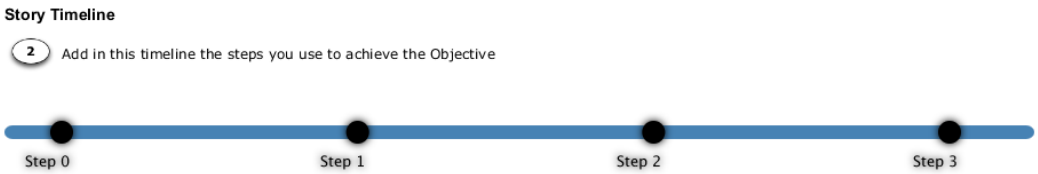

Figure 4.12: The graphical timeline available in the prototype for supporting the elicitation of the flow of actions in the story.

The user interacts with the graphical user interface which at the beginning displays a main window that has two sections named "story information" and "story timeline" as show in the Figure 4.13. The "story information" section is directly related with the information of the goal (referred as objective in the GUI) as presented in section 3.1.1. The left part captures the information about the goal either when it is the first goal (assign the global objective of the story), or when it is a subgoal (it captures the data that answer the *"Who?", "Does what?", "to Whom?"*). The right part is part of the implementation of the concepts behind the social interactions presented in section 3.2.1.1.1, and it displays the social network of the story in a graphical form.

Second, the "story timeline" allows the user to introduce in a graphical fashion the actions that were taken in the story. If the timeline is related to the first goal, a user normally will introduce the outline of the most important steps in the story as actions in the timeline. As an example, if a user introduces the plot of the movie "The terminator", then he/she can set the main goal as "A robotic assassin from a post-apocalyptic future travels back in time to eliminate Sarah Connor, whose son will grow up and lead humanity in a war against machines" [47]. Next, as an outline of the main actions taken in the story he/she can input the following actions,

1. The Terminator is sent from the future.
2. Kyle Reese is sent from the future.
3. The Terminator searchs Sarah.
4. Kyle Reese protects Sarah.
5. Kyle Reese kiss Sarah.
6. Sarah destroys The Terminator.

After the outline, the user can decide to add more details to each action in the outline, which results in the creation of a nested timeline in which the user can add more actions that describe the parent action.

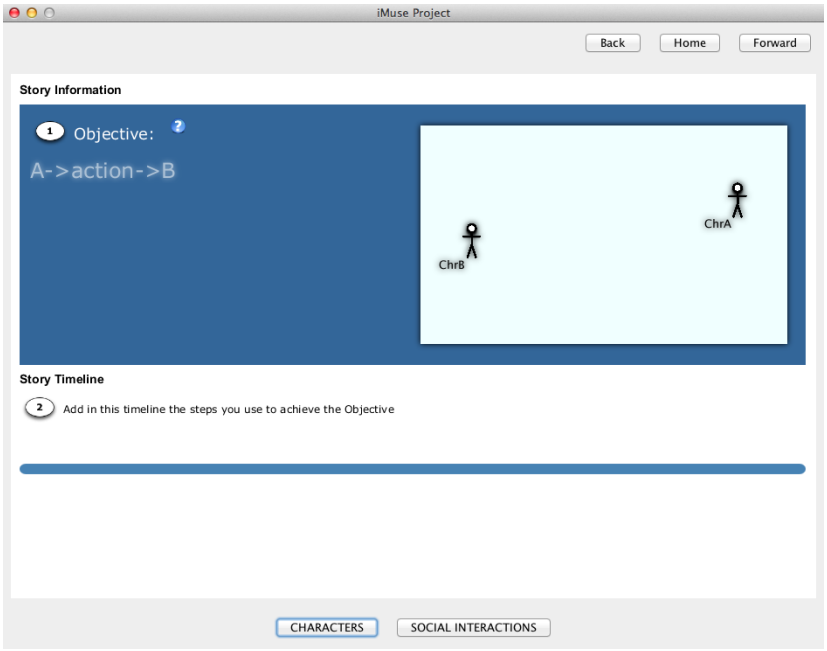

Figure 4.13: Main window of the graphical user interface of the prototype.

Figure 4.14 shows how a user will use the GUI to introduce the steps that represent a story. First, the user adds the outline of the story by interacting with the empty timeline. Second, the user clicks over the timeline to create two actions named "ChrA did this to ChrB" and "Step1". Third, the user decides to add another action named "Step2". Next, the user adds details about the action "ChrA did this to ChrB" by performing a right click onto that action, then selecting the option "Add details" and as a result, creates a nested timeline. Finally, the user adds three actions to the nested timeline, which are named as "Step3", "Step4", and "Step5".

4.2.2 Social Interactions as a Social Network.

As presented in section 3.2, the changes in a social network can be captured using a finite state machine model. However, in terms of the graphical user interface, this model does not make any assumptions on how to capture those changes or on how to capture what produced them. Since, it seems reasonable to expect that the social

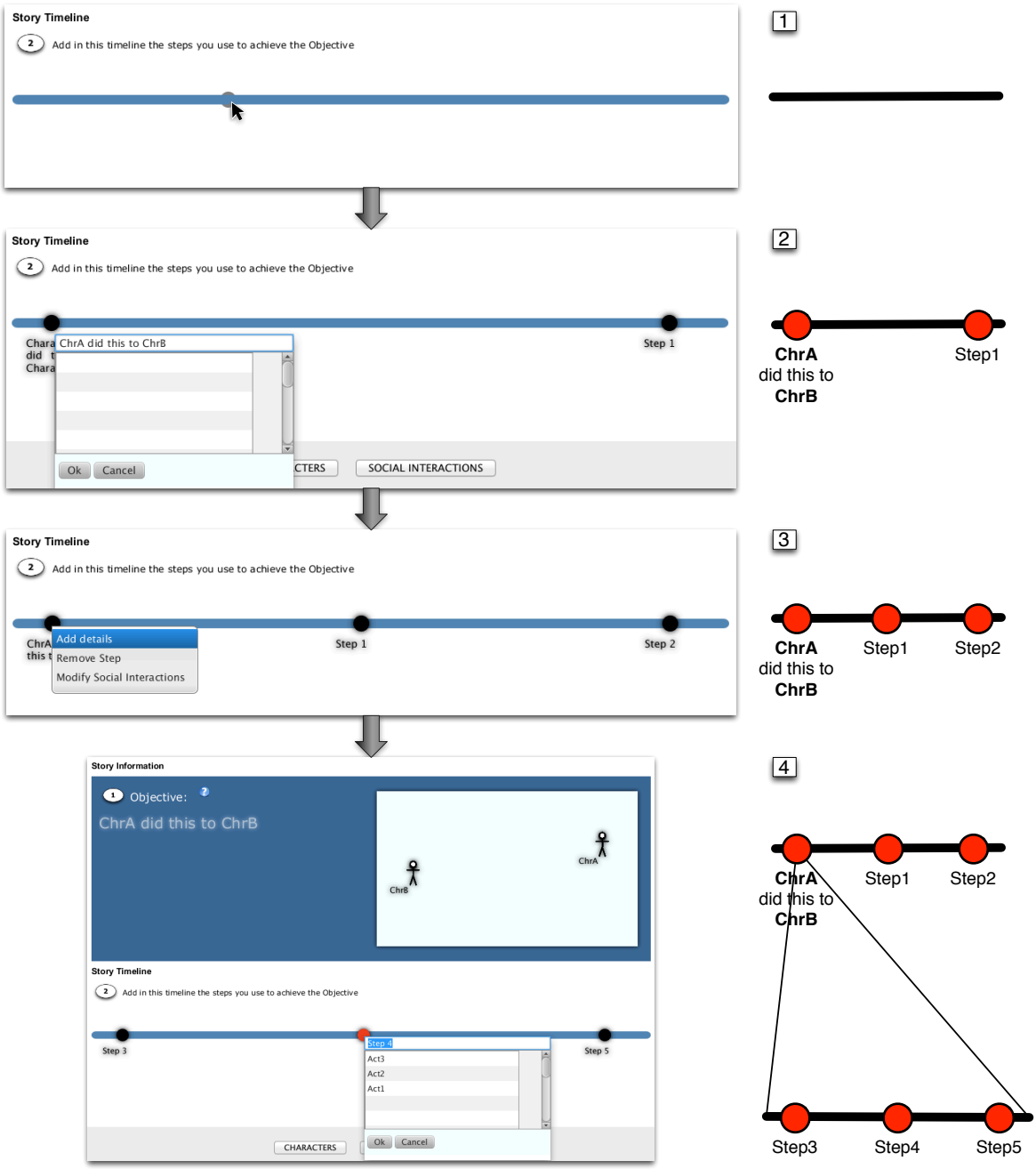

Figure 4.14: The steps follow by a user to input the actions of a story using the GUI. First, the user interacts with the empty time line. Second, the user clicks over the timeline to create two actions. Third, the user right click onto the first action to create a nested timeline using "Add details". Finally, the user adds three actions to the nested timeline.

network will potentially change by each action performed by one of the characters in the story. Hence, a natural way to define a change in the social network is through the explicit definition of whether or not an action introduces a change in the social network. Thus, in the process of telling the story the graphical user interface supports the direct specification that an action introduces a change in the social network.

Building upon an example from section 3.2.1.1.1, we use the excerpt from the novel Les Misérables to show how the system supports the elicitation of the social interactions. This excerpt is as follows "After he and Cosette leave, Marius asks Éponine to retrieve her address for him. Éponine, who is in love with Marius herself, reluctantly agrees to do so". As an illustration, Figure 4.15 shows just the state before the action and the new state after the user decides that the action introduces a social network change.

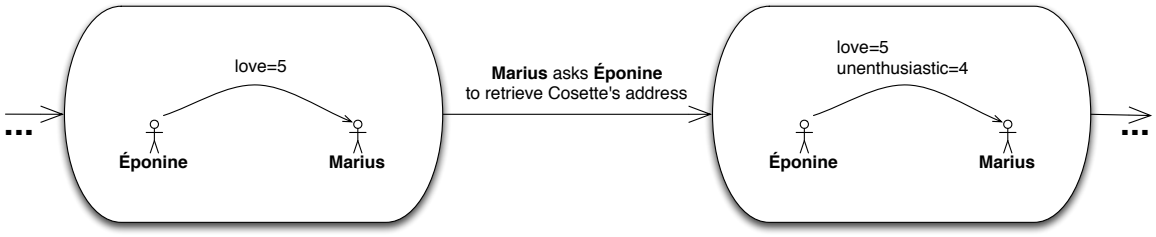

Figure 4.15: Change on the social interactions between Marius and Éponine. The social network after the action was performed captures the change on the social interactions after Marius asks Éponine to retrieve Cosette's address. This figure represents the social network progress.

Assuming that the characters (Marius and Éponine) in the excerpt the story already were added into the system. Then, a user will be use the following sequence of steps to introduce the previous story into the front end. First, the user will identify or chose the action that he/she thinks cause the change in a social interaction within the current social network. Then, the user presses the right click over the action "Marius asks Éponine to retrieve Cosette's address", which shows a pop-up menu with three

options. Since the objective is to modify the social interactions, the user presses the option called "Modify Social Interactions" as shown in Figure 4.16 (left).

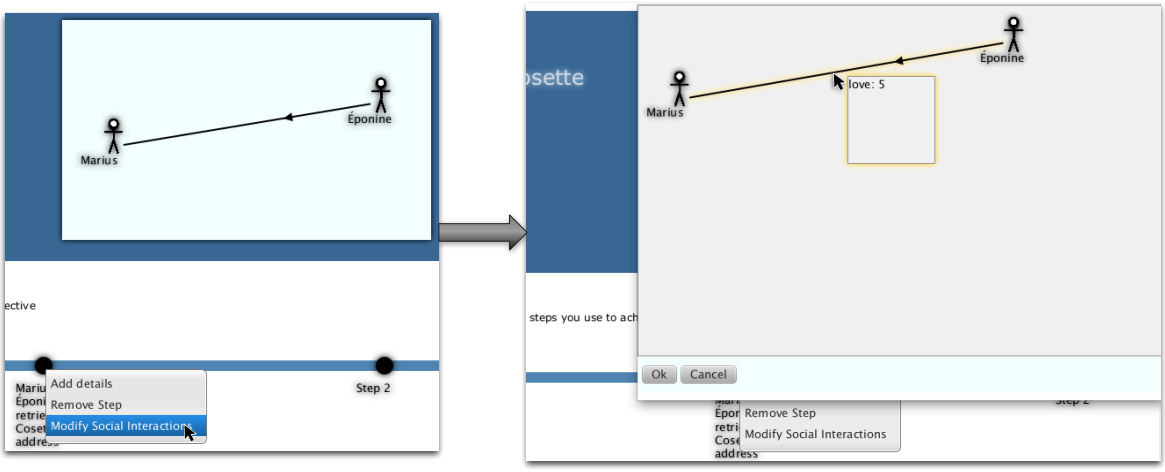

Figure 4.16: Left: it shows how a user can modify the social interactions. Right: it presents the current social network of the sample story.

Next, the GUI shows a dialog that displays the current social network at the "time" the action was performed. This dialog has a visual representation of the social interactions between the characters in the story as presented in Figure 4.16 (right). Next, the user presses the right-click over the social interaction that goes from Éponine to Marius, as a result he/she can add, modify, or eliminate (CRUD) a social state by selecting the option "Modify social states" in the pop-up as in Figure 4.17.

Then, in Figure 4.18 (left) the system displays a dialog that has the list of current social states and its values. Finally, the user can add or remove social states using the former dialog as in Figure 4.18 (right). Since in the plot Éponine reluctantly agrees to retrieve Cosette address, the user adds the social state "unenthusiastic" with value "4" to capture the change of perception of Éponine with respects to Marius due to the performed action.

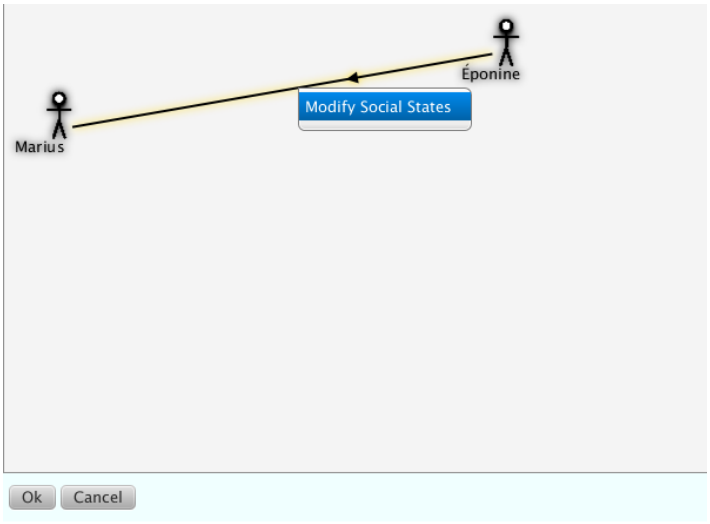

Figure 4.17: The user uses the pop-up option "Modify social states" to access the dialog that adds, modifies, or eliminates (CRUD) a social state.

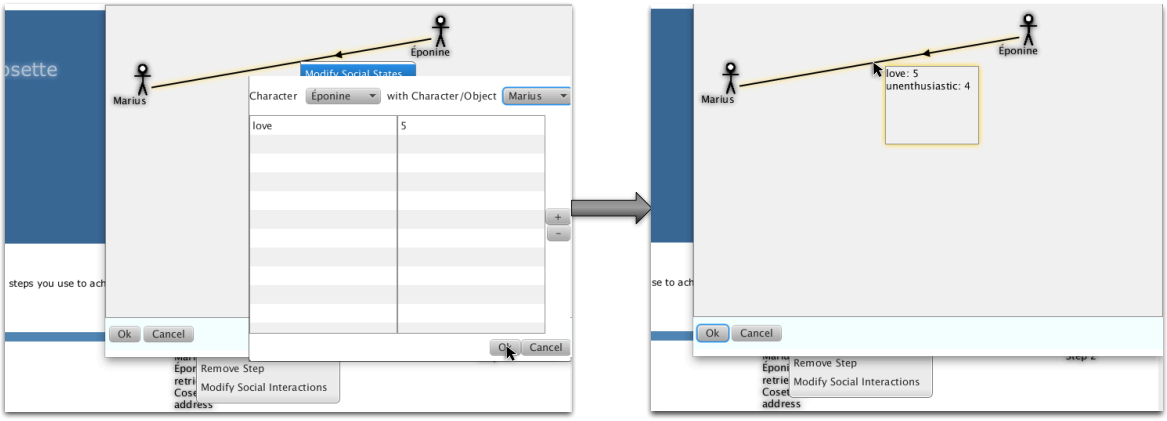

Figure 4.18: Left: it presents the dialog to add or remove social states into a social interaction. Right: it shows the list of social states of the social interaction between Éponine and Marius.

4.2.3 Sample Story

This section presents how a user can input a story with the flow of the actions and the corresponding social interactions. To this end, we use an excerpt of the novel Les Misérables which was discussed in a previous section. This story is as follows: "Years earlier in Paris, a grisette named Fantine was very much in love with Félix Tholomyès. His friends, Listolier, Fameuil, and Blachevelle were also paired with Fantine's friends Dahlia, Zéphine, and Favourite. The men (Félix Tholomyès) abandon the women (Fantine), treating their relationships as youthful amusements. Fantine must draw on her own resources to care for her and Tholomyès daughter, Cosette".

First, the user interacts with an empty story as presented in subfigure 1 in Figure 4.19. Then, the user fills up the first section "story information" by describing the overall goal of the story, which in this case can be stated as "Fantine is left by Felix Tholomyes". Hence, the user (or a reader) can have an overall view of the underlying reason for which the actions were taken. To fill up the goal, the user just clicks over the highlighted text "$A \rightarrow action \rightarrow B$" as in subfigure 2 in Figure 4.19.

Next, a user can add characters participating in the story by clicking on the "Characters" button at the bottom of the user interface. This action will open a pop-up dialog that let the user visualize the current set of character, add a new character, remove a character, or modify an existing character as in subfigure 3 in Figure 4.19. At this point, the user has introduced just the main characters named "Fantine" and "Felix Tholomyes" along with the overall goal. However, the process of adding or modifying the characters in a story is up to the user, and it is just required to add a character when it appears in either an action or a social interaction.

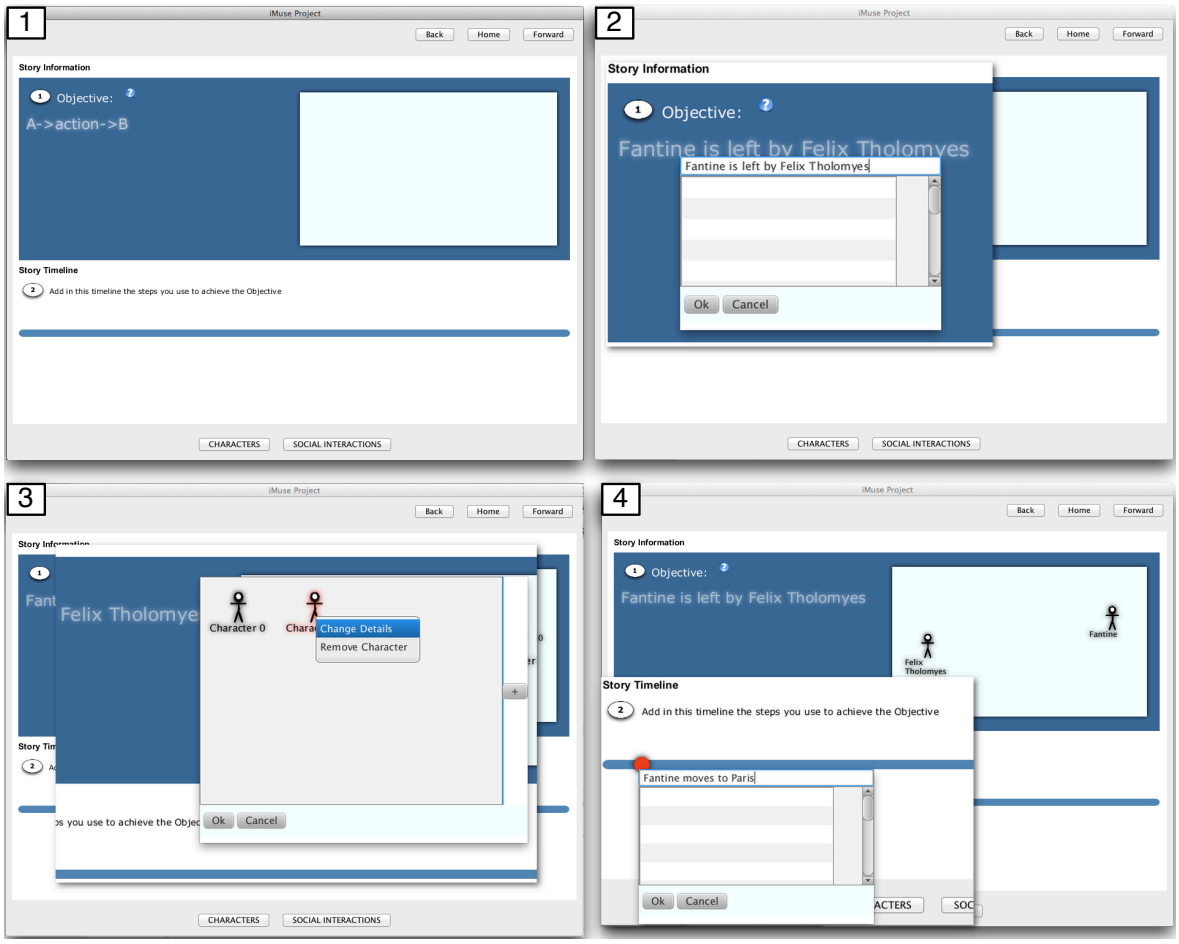

Figure 4.19: The steps that a user follow to input the main goal, the characters, and the first action for a story. Top-left: interaction with an empty story. Top-right: the user adds the main goal. Bottom-left: the user decides to add two characters to the story. Bottom-right: the user adds the first action to the timeline.

Then, the user introduces the outline of the main actions taken in the story as shown in subfigure 4 in Figure 4.19. In this example, the user introduces three main actions named "Fantine moves to Paris", "Fantine falls in love with Felix Tholomyes", and "Felix Tholomyes abandon Fantine". Later on, the user can add more details

about each of these actions. For example, he/she can add to the first action another subgoal that provide details about how Fantine move to Paris as well as insights of why she took that decision. Thus, if that were the case, the user could add (as in section 4.2.2) the following actions as part of the subgoal: "Fantine's parents die" and "Fantine move to an orphanage". Subfigure 5 in Figure 4.20 shows the three actions that were introduced by the user.

Next, the user decides that there was a potential change in the social interactions after the action in which "Fantine falls in love with Felix Tholomyes". Clearly, the perception of Fantine towards Félix Tholomyès changed since she fell in love with him. At this point, it is not clear why did she fall in love, but later on the user can decide to add details, by adding a subgoal and its actions, about the actions that yield in this situation. Continuing with the example, then the user clicks over the action "Fantine falls in love with Felix Tholomyes" and selects the option "Modify Social Interactions" as in subfigure 6 in Figure 4.20

In subfigure 7 in Figure 4.20, a pop-up dialog with the social interactions information appears, then the user can add social interactions between characters with a drag-and-drop gesture. This gesture begins with the selected source (in this case Félix Tholomyès) and ends with the target character (in this case Fantine). Then, the system displays a dialog similar to the one shown in subfigure 10 in Figure 4.21.

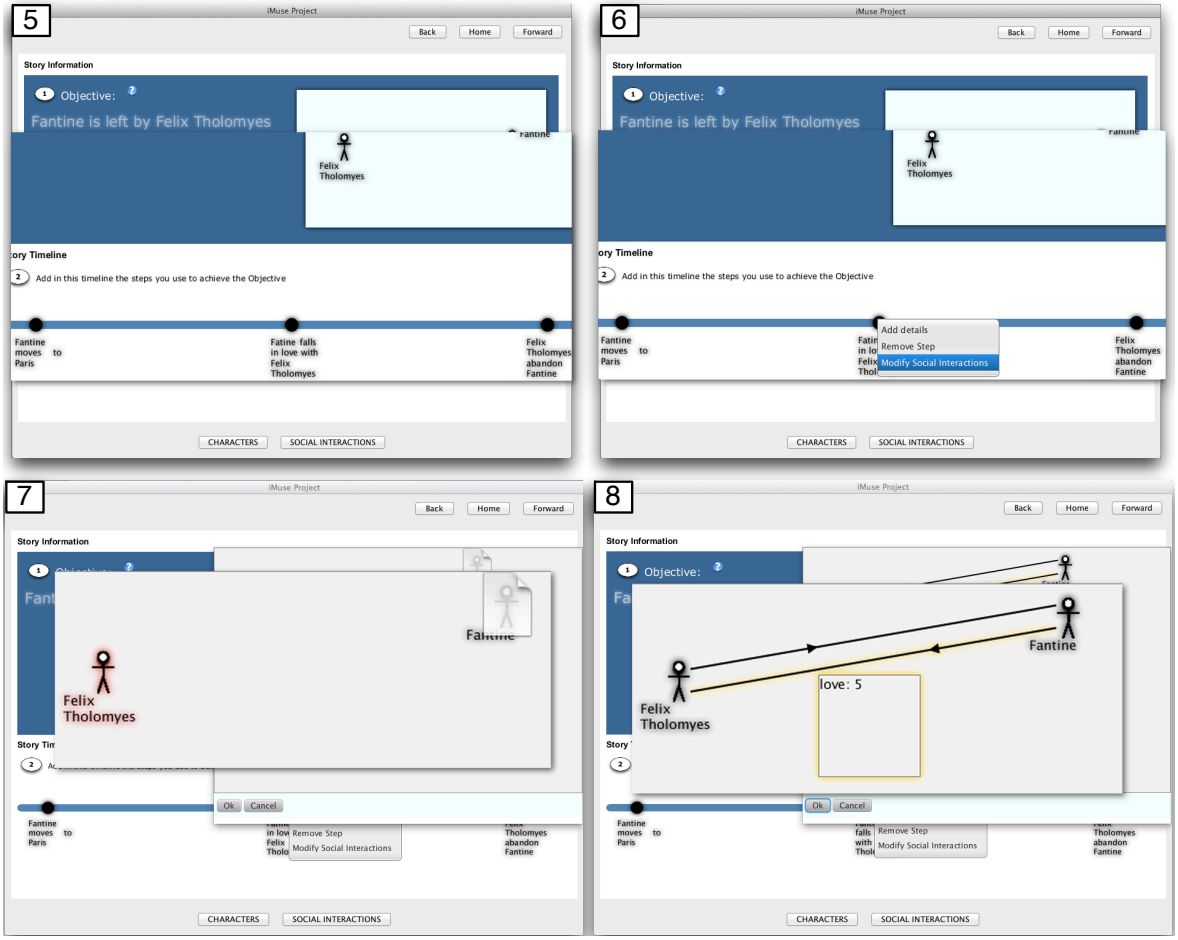

Figure 4.20: Top-left: three main actions inserted by the user. Top-right: the user decides to add details about the changes in the social interactions. Bottom-left: the user decides to add a social state from Félix Tholomyès to Fantine. Bottom-right: the user added social states from each character which results in a bidirectional social interaction.

After the user finishes his modifications of the social interaction, he/she can review the different social states in a specific social interaction between two characters in a graphical fashion as in subfigure 8 in Figure 4.20. In the example, the user added the social state "love=5" in the social interaction from Fantine to Félix, and assumed

the social state of "love=3" in the interaction from Félix to Fantine since it was not stated in the story. In this way, a user can assume details about the social interactions in order to explain the underlying reasons for the actions in the story.

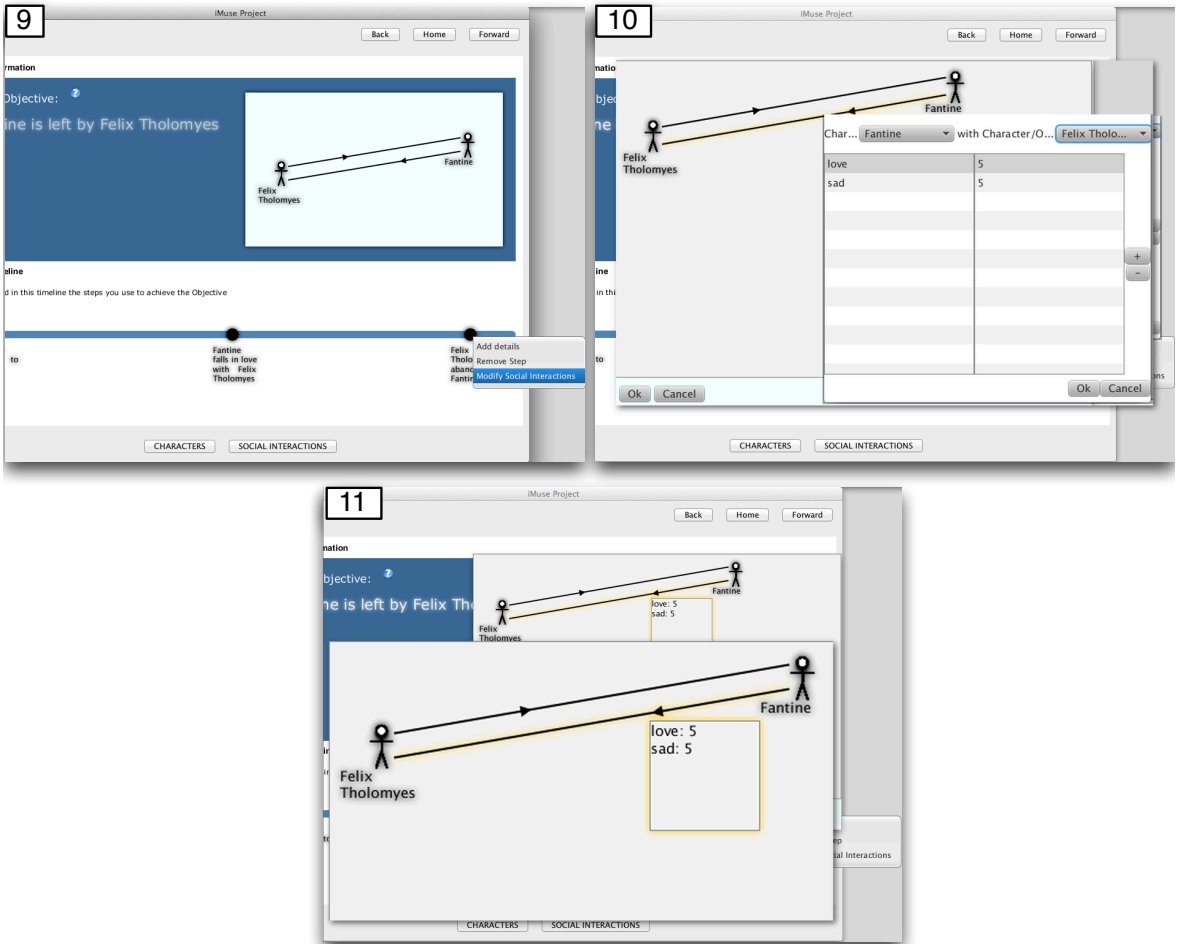

Figure 4.21: Top-left: the user decides that the social interactions changed after Félix Tholomyès abandon Fantine. Top-right: the user specifies how the values of the current social states changed and decides to add a new social state named "sad". Bottom: the user uses the interface to check the current values of the social interaction of Fantine with respects to Félix Tholomyès.

In subfigure 9 in Figure 4.21, the user decides that there is a potential change in the social interactions because Félix abandon Fantine. As a result, the system displays the social interaction dialog, and then the user decide to modify the social interaction from Fantine toward Félix which shows a dialog with the current social states as shown in subfigure 10 in Figure 4.21. Using this dialog, the user adds a new social state named "sad" with a value of 5 since Fantine is heartbroken. After the user finishes his modifications of the social interaction, he/she can review the different social states as in subfigure 11 in Figure 4.21.

Chapter 5

CONCLUSIONS

We have proposed a flexible story model that captures the flow of actions in a story as well as the social interactions between characters of a story. This is important since individuals of a domain have the potential of making valuable contributions to the body of knowledge, and those contributions are often driven in a narrative fashion. Thus, the use of the proposed story model can aid individuals to share their experience and expertise by means of stories that follow a specific syntax and semantics, thereby reducing the ambiguities and assumptions which are often present in natural language stories. The model is meant to open the door for capturing the underlying knowledge of a domain so that domain knowledge can be shared among individuals of the domain.

An important aspect of this work is that the model was designed to capture the changes in the social relationships between the characters that result from their actions. This is helpful in those situations when is necessary to reason about the consequences of the actions in terms of the social interactions of the characters.

As part of this work, we proposed several prototypes of a graphical user interface that provides users with automated help to capture their narratives into the formal representation. The goal of the GUI is to reduce the complexity for users to express formal stories and aid them to tell complete and precise stories of their experiences. This is achieved thanks to the explicit support in the tool for adding subgoals, new actions, social interactions, and characters.

A key aspect of the software design was to decouple modules to allow the tool to be extensible, so that we can continue to explore new ways to interact with the underlying model.

Chapter 6

FUTURE WORK

Further steps include the generation of representations of a story in other formats, which can be integrated with various simulation tools. This could open the door to simulate an existing story in a virtual world to make the knowledge store in each story more accessible and easy to consume.

Also, it is possible to reuse the existing stories in the system to generate automatically stories. In this way, it would be possible to combine several stories that have similarities and create new stories from that process.

It is also important to design a usability study, with several people in different domains, in order to further evaluate the system and use the feedback to improve the acquisition system.

Finally, it would be interesting to explore the applicability the proposed system in the domain of software engineering. In fact, the storytelling system can be used to identify different types of user workflow scenarios in a software development project.

REFERENCES

[1] Julian E. Orr. Narratives at Work: Story Telling As Cooperative Diagnostic Activity. In *Proceedings of the 1986 ACM Conference on Computer-supported Cooperative Work*, pages 62–72, New York, New York, USA, December 1986. ACM Press.

[2] Scott R. Turner. *The Creative Process: A Computer Model of Storytelling and Creativity*, volume 1994. Routledge, 1994.

[3] James Skorupski and Michael Mateas. Interactive Story Generation for Writers: Lessons Learned from the Wide Ruled Authoring Tool. *Cognition and Creativity, Digital Arts and Culture 2009, Arts Computation Engineering, UC Irvine*, 2009.

[4] Josh McCoy, Mike Treanor, Ben Samuel, Brandon Tearse, Michael Mateas, and Noah Wardrip-Fruin. Comme il Faut 2. In *Proceedings of the Intelligent Narrative Technologies III Workshop on - INT3 '10*, pages 1–8, New York, New York, USA, June 2010. ACM Press.

[5] James Skorupski, Lakshmi Jayapalan, Sheena Marquez, and Michael Mateas. Wide Ruled: A Friendly Interface to Author-Goal Based Story Generation. *VIRTUAL STORYTELLING. USING VIRTUAL REALITY TECHNOLOGIES FOR STORYTELLING - Lecture Notes in Computer Science*, 4871:26–37, 2007.

[6] Brandon Tearse, Michael Mateas, and Noah Wardrip-Fruin. MINSTREL Remixed: A Rational Reconstruction. In *Proceedings of the Intelligent Narrative Technologies III Workshop on - INT3 '10*, pages 1–7, New York, New York, USA, June 2010. ACM Press.

[7] I. M. T. Swartjes and M. Theune. The virtual storyteller: story generation by simulation. In *Proceedings of the Twentieth Belgian-Netherlands Conference on Artificial Intelligence, BNAIC 2008, Enschede*, pages 257–265, Enschede, October 2008. University of Twente.

[8] Sebastian Weiss, Wolfgang Müller, Ulrike Spierling, and Florian Steimle. Scenejo – An Interactive Storytelling Platform. In Gérard Subsol, editor, *Virtual Storytelling. Using Virtual Reality Technologies for Storytelling*, volume 3805 of *Lecture Notes in Computer Science*, pages 77–80. Springer Berlin / Heidelberg, 2005.

[9] Michael Mateas and Phoebe Sengers. Narrative intelligence. In *Proceedings AAAI Fall Symposium on Narrative Intelligence*, pages 1–10, 1999.

[10] P Gervas. Computational Approaches to Storytelling and Creativity. *AI Magazine*, 30(3):49, 2009.

[11] Noah Wardrip-Fruin. *Expressive Processing: Digital Fictions, Computer Games, and Software Studies*. MIT Press, 2009.

[12] Paul Bailey. Searching for storiness: Story-generation from a reader's perspective. In *Working notes of the Narrative Intelligence Symposium*, pages 157–164, 1999.

[13] Kenneth Michael Kahn. Creation of Computer Animation from Story Descriptions. August 1979.

[14] Michael Lebowitz. Story-telling as planning and learning. *Poetics*, 14(6):483 – 502, 1985.

[15] Michael Lebowitz. Creating characters in a story-telling universe. *Poetics*, 13(3):171 – 194, 1984.

[16] Michael Lebowitz. Creating a Story-Telling Universe. 1983.

[17] James Skorupski. Novice-Friendly Authoring of Plan-Based Interactive Storyboards. *Sixth Artificial Intelligence and Interactive Digital*, 2010.

[18] James Skorupski. Storyboard authoring of plan-based interactive dramas. In *Proceedings of the 4th International Conference on Foundations of Digital Games - FDG '09*, page 349, New York, New York, USA, 2009. ACM Press.

[19] M. Mateas and A. Stern. A behavior language for story-based believable agents. *IEEE Intelligent Systems*, 17(4):39–47, July 2002.

[20] R. Raymond Lang. A Declarative Model for Simple Narratives. In *Proceedings AAAI Fall Symposium on Narrative Intelligence*, 1999.

[21] Brandon Tearse, Peter Mawhorter, Michael Mateas, and Noah Wardrip-Fruin. Minstrel remixed: User interface and demonstration, 2011.

[22] Brandon Tearse, Noah Wardrip-Fruin, and Michael Mateas. Minstrel remixed: Procedurally generating stories, 2010.

[23] B Tearse, P Mawhorter, M Mateas, and N Wardrip-Fruin. Lessons Learned From a Rational Reconstruction of Minstrel. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[24] Ulrike Spierling, Sebastian Weiß, and Wolfgang Müller. Towards Accessible Authoring Tools for Interactive Storytelling. In Stefan Göbel, Rainer Malkewitz, and Ido Iurgel, editors, *Technologies for Interactive Digital Storytelling and Entertainment*, volume 4326 of *Lecture Notes in Computer Science*, pages 169–180. Springer Berlin / Heidelberg, 2006.

[25] David Rumelhart. Notes on a schema for stories. 1975.

[26] Selmer Bringsjord and David Ferrucci. *Artificial Intelligence and Literary Creativity: Inside the Mind of Brutus, A Storytelling Machine*. Taylor & Francis, 1999.

[27] Taisuke Akimoto and Takashi Ogata. A consideration of the elements for narrative generation and a trial of integrated narrative generation system. In *2011 7th International Conference on Natural Language Processing and Knowledge Engineering*, pages 369–377. IEEE, November 2011.

[28] James R. Meehan. TALE-SPIN, an interactive program that writes stories. pages 91–98, August 1977.

[29] Stephen Slade. The Yale artificial intelligence project. *AI Magazine*, 8(4):67–76, December 1987.

[30] Jasper Bragt. Towards believable characters in the virtual storyteller, July 2010.

[31] Joshua McCoy, Michael Mateas, and Noah Wardrip-Fruin. Comme il Faut: A System for Simulating Social Games Between Autonomous Characters. In *UC Irvine: Digital Arts and Culture*, 2009.

[32] Anne Sullivan, April Grow, Tabitha Chirrick, Max Stokols, Noah Wardrip-Fruin, and Michael Mateas. *Extending CRPGs as an interactive storytelling form*, volume 7069 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, November 2011.

[33] Aaron A. Reed, Noah Wardrip-Fruin, Ben Samuel, Anne Sullivan, Ricky Grant, April Grow, Justin Lazaro, Jennifer Mahal, Sri Kurniawan, and Marilyn Walker. SpyFeet. In *Proceedings of the 6th International Conference on Foundations of Digital Games - FDG '11*, pages 310–312, New York, New York, USA, June 2011. ACM Press.

[34] Anne Sullivan, April Grow, Michael Mateas, and Noah Wardrip-Fruin. The design of Mismanor. In *Proceedings of the International Conference on the Foundations of Digital Games - FDG '12*, page 180, New York, New York, USA, May 2012. ACM Press.

[35] Josh McCoy, Mike Treanor, Ben Samuel, Michael Mateas, and Noah Wardrip-Fruin. Prom Week. In *Proceedings of the 6th International Conference on Foundations of Digital Games - FDG '11*, pages 319–321, New York, New York, USA, June 2011. ACM Press.

[36] Meher Van Groenou. "Tell Me a Story": Using Children's Oral Culture in a Preschool Setting. *Montessori Life*, 7(3):19–21, 1995.

[37] W Swap, D Leonard, M Shields, and L Abrams. Using mentoring and storytelling to transfer knowledge in the workplace. *Journal of management information systems*, 18(1):95–114, 2001.

[38] Graesser Arthur C., Lang Kathy L., and Roberts Richard M. Question answering in the context of stories. *Journal of Experimental Psychology: General*, 120(3):254–277, 1991.

[39] Mark O. Riedl, Jonathan P. Rowe, and David K. Elson. Toward intelligent support of authoring machinima media content: story and visualization. page 4, January 2008.

[40] M.O. Riedl and B. O'Neill. Computer as Audience: A Strategy for Artificial Intelligence Support of Human Creativity. In *Computational Creativity Support Workshop at CHI'09*, 2009.

[41] Brian O'Neill and Mark Riedl. Supporting human creative story authoring with asynthetic audience. In *Proceeding of the seventh ACM conference on Creativity and cognition - C&C '09*, page 399, New York, New York, USA, October 2009. ACM Press.

[42] Ashwin Ram. AQUA: Questions that Drive the Explanation Process.

[43] Wikipedia. Les misérables — wikipedia, the free encyclopedia, 2013. [Online; accessed 14-March-2013].

[44] B. Bruegge and A.H. Dutoit. *Object-Oriented Software Engineering: Using Uml, Patterns, and Java*. Prentice Hall, second edition, 2004.

[45] W. Strunk, E.B. White, and M. Kalman. *The Elements of Style*. A Penguin book : Reference. Penguin Group, 2007.

[46] Wikipedia. Timeline — wikipedia, the free encyclopedia, 2013. [Online; accessed 22-March-2013].

[47] James Cameron. The terminator (movie), October 1984. [Online; accessed 22-July-2013].