An Authoring Tool for an Emergent Narrative Storytelling System

Michael Kriegel, Ruth Aylett
School Of Mathematics And Computer Science
Heriot-Watt University
Edinburgh E14 4AS UK
{michael,ruth}@macs.hw.ac.uk

João Dias, Ana Paiva
INESC-ID
IST, Taguspark
Porto Salvo, Portugal
joao.assis@tagis.utl.ist.pt, ana.paiva@inesc-id.pt

Abstract

In this paper we present the initial conceptual design of an authoring tool for the emergent narrative agent architecture *FAtiMA* that powers the virtual bullying drama *FearNot!*. We explain that the process of authoring emergent narrative to a large part consists of designing a planning domain for a virtual character planner and explain the difficulties this task poses to the non-technical author. After reviewing existing authoring tools and evaluating them in terms of their applicability to *FAtiMA*, we introduce a novel concept of approaching the authoring task, in which the author is playing through example story lines that are used to gradually increase the knowledge and intelligence of a virtual character. This concept is extended by a mixed initiative feature, which allows the author to cooperate with the character planners while providing the example stories. Finally we concretize our idea and explain our intended implementation of it within the *FearNot!* Framework. We believe that our design, although being specified with a particular architecture (*FAtiMA*) in mind, may provide some interesting ideas to others, who are trying to solve the authoring problem for interactive storytelling systems.

Introduction

The field of digital interactive narrative is becoming a recognized branch of AI research, with an increasing number of research groups trying to tackle this problem. The prospect of a story that fully adapts to the user's choices in real time is very tempting and such a technology could be of great use for entertainment and education purposes. After books and films, many regard the interactive story as the next revolutionary narrative medium of the future. However, currently interactive storytelling is in its infancy at best as there are many technological problems to solve. There are several approaches to interactive storytelling but despite some potential differences in their philosophy they all seem to struggle with the same problem: In order to let the computer tell and adapt a story it needs some level of understanding of the story world and that can only be achieved by a storytelling system that is backed up with a huge amount of content, describing the system's knowledge about the story world. This content can take different forms (e.g. character scripts, planning actions, logical sentences or story segments), but it will exist in some form in every interactive storytelling system and either implicitly or explicitly encode the 'meat' of the story. The process of creating this content, which we will henceforth refer to as authoring, seems to be the main bottleneck in the creation of interactive stories. As a result, most implementations of storytelling systems have only been tested with small example stories that are usually too small to convey the feeling of agency and free choice that interactive stories should give a user. Façade (Mateas & Stern 2005) is a notable exception here.

This situation is not very surprising given the fact that most of these systems originate from a computer science academic background and their creators simply lack the spare resources to allocate to the authoring task. So it seems to be a logical and ultimately unavoidable step to involve external authors in order to let them create the stories. However, currently the authoring process in interactive storytelling systems is a technically complex task, which prevents traditional story authors with a non-technical background from creating interactive story content. Furthermore, the amount of content required to experience real interactivity is enormous. Those problems might both be tackled with authoring software to support the story content creation process. Ideally this authoring software is both accessible (i.e. easy to use) and productive (i.e. even speeds up the authoring process for expert users). In this paper, which is a continuation of work initially described in (Kriegel & Aylett 2007) we will introduce the emergent narrative approach to interactive storytelling through the virtual drama *FearNot!* and describe the tasks of an author in the emergent narrative framework in general and in *FearNot!* in particular. We then review existing authoring software for interactive storytelling and introduce our own conceptual approach to authoring software. Finally we will relate this approach to *FearNot!* and come up with a more detailed design concept for *FearNot!* authoring software that we intend to implement in the near future.

Emergent narrative and *FearNot!*

Emergent Narrative (Louchart & Aylett 2004) is a theory of how to address the interactive storytelling problem. According to this theory, a narrative or storyline can emerge from the interactions between intelligent virtual agents.

Those agents are autonomous, intelligent and emotional. Implementation-wise that means that each agent is an individually running program that includes a continuous planner. Unlike other storytelling approaches where a story is planned, in emergent narrative only agent behavior is planned. That means characters do not take commands from a higher level entity and do not directly follow a plot. This way character believability is maximized since they will not carry out actions that help advance the plot but are not in character. Instead an author has to ensure that the characters are configured in such a way that they exhibit the intended behavior for the anticipated story. It is thus the author's main task to configure the planners that drive the agent's behavior. A good way to understand the main philosophy behind emergent narrative is to look at real life role-playing games (RPGs, live action or pen & paper). Players are briefed in advance of the game session in order to make sure they understand their character's role in the world. Once the game starts no one restricts their freedom of decision making. The game master can influence the course of the story by influencing the environment and action outcomes, but he has no direct control over the decisions of the player. In emergent narrative the agents are let loose in a similar way as the players in an RPG. The author's task of configuring the agents is then analogous to briefing the players of an RPG.

The emergent narrative concept described above could be implemented in many different ways. We will now quickly describe its implementation within the educational anti-bullying application *FearNot!* (Aylett, Dias, & Paiva 2006). In *FearNot!* small groups of agents representing school children interact within a virtual school environment. Each of those agents runs a separate instance of the agent mind software *FAtiMA*. *FAtiMA* incorporates two levels, a deliberative level that controls the agent's goal-oriented behavior and is built upon an emotional continuous planner, and the reactive level which defines a set of fast-activation rules that allow the agent to experience an emotion and react very quickly. FAtiMA's emotion system is based on the OCC emotion theory (Ortony, Clore, & Collins 1988).

Deliberative level

The deliberative level represents the agent's reasoning process and its intelligence along with its goal directed behavior. On this level the author specifies goals and actions in a STRIPS like planning language. A goal is represented by a desired world state (success condition), a forbidden world state (failure condition) and an activation world state (activation condition) that makes the goal become active, i.e. when that condition is reached the agent creates an explicit intention to achieve the goal and will start planning for it. An action on the other hand is represented by its pre conditions (necessary world state in order to carry out that action) and effects (changes that this action brings to the world). World states are annotated using logic formulations. It is possible to equip an action with variable parameter slots and to access those variables. In Figure 1 the concept is demonstrated with an exemplary action and goal. The goal describes the generic goal of inviting a friend to a party. It is activated if the agent is hosting a party and sees someone he likes, who is not invited yet. In this situation it will create an instantiated intention to invite that agent by replacing the unbound variable *[friend]* with the agent's name. The goal will succeed if the invited agent replies positively and will fail if the agent replies negatively. The action represents the generic *Question* action, which has the precondition of not asking a question to yourself and has the effects of getting a negative or positive answer. This action can be used by the planner to achieve the invite goal by replacing the unbound variables *[question]* and *[target]* by values that match the goal's success conditions.

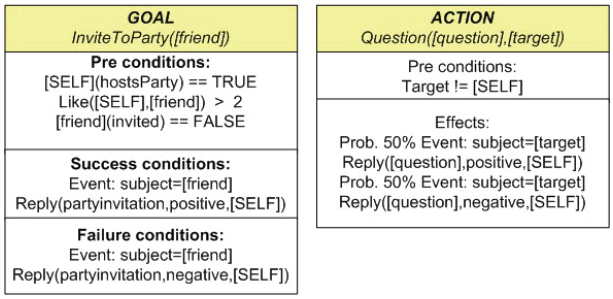

Figure 1: example for a goal and action

The deliberative layer is also responsible for the generation of OCC's prospect based emotions (e.g. Hope, Fear, Satisfaction). These emotions are triggered by the state of plans and goals in memory. For instance, if the plan that the agent is considering is likely to succeed the agent will feel hope. But if the plan then fails, it will feel disappointment. These emotions are then used to influence the coping strategies applied to the plans (e.g. if the agent is fearful it might give up its goal).

Reactive level

The reactive level handles both the reactive appraisal of external events and the agent's reactive behavior. Events are appraised by the agent using appraisal rules (emotional reactions) provided by the author. These rules define the desirability , the praiseworthiness and desirability for others of perceived events. These values are then used to create most of OCC's emotions. The agent's reactive behavior is handled by a set of action tendencies. An action tendency represents an unplanned instinctive behavior that is triggered by an emotion. For example, an agent that is very sad might start to cry involuntarily. Figure 2 shows an example emotional reaction and action tendency.

The emotional reaction is triggered when anyone refuses the agent's party invitation, which is considered undesirable and a bad action by the agent. The action tendency described is activated (triggers a crying action) whenever there is an emotion of type *distress* with an intensity of seven or more.

All together, actions, goals, action tendencies and emotional reactions encode almost the complete story content.

Figure 2: example for an emotional reaction and action tendency

Currently all of this needs to be hand coded by an author using *FAtiMA*'s XML syntax. It is not hard to see that creating and debugging this kind of data is a tedious task that could be greatly improved via the use of an authoring tool and without such could quickly become unmanageable, especially when considering the huge amount of content that is needed to tell a truly interactive dramatic story.

From our experience in authoring*FearNot!* the greatest problem for content authors has been to think more in terms of characters and less in terms of stories. It seems inevitable that authors first think about interactive stories in terms of a variety of possible linear stories, instead of concentrating only on the characters. Being able to let go of the control of the story as an author is one of the key concepts that emergent narrative authors need to learn. We will investigate how authoring software can facilitate this process.

Related work

In this section we review existing narrative authoring, planning and content generation tools in order to identify the state of the art and to find out whether any of them can be facilitated in the task of authoring characters, described in the previous section. It should be clear that there will be no tool that is directly applicable to our *FearNot!* domain, because the *FearNot!* architecture requires its narrative content to be described in a quite specific way. However, certain ideas and concepts might be helpful in our task of designing a *FearNot!* authoring environment.

Interactive storytelling authoring tools

Other research groups that work on interactive storytelling have also identified the creation of story content as a serious bottleneck for the evolution of serious interactive storytelling and have thus worked on authoring tools as well as storytelling engines. A good overview of the tools available can be found in (Medler & Magerko 2006). Unfortunately many tools in this category as for example (Zagalo *et al.* 2006) or (Sauer *et al.* 2006) focus on the configuration of a storytelling environment but not on the creation of narrative content per se. Those tools are very similar to the editing software for video games such as Neverwinter Nights or Warcraft 3. They allow the user to choose and modify environments for the story, place characters and objects within the environment and set up basic properties of those entities. We created a similar tool for *FearNot!* and can use it to quickly try different character constellations in different situations. However, we cannot use it to create or modify the content of a character's mind (actions, goals, emotional reactions and action tendencies). There are also a few tools designed to allow the creation of lower level narrative content, for example SWAT, the authoring tool for Chris Crawford's storytelling engine Storytron, the design of which is described in (Crawford 2005). However in SWAT this more low level control over the involved narrative elements introduces a complexity which makes the tools less user friendly for non technical authors. In fact the authoring approach using for example SWAT is very similar to programming using a visual programming environment.

Plan authoring and knowledge engineering tools

Since we are talking about authoring characters that are driven by planners it is worth having a look at the research field of AI planning and the tools used to specify planning domains. Knowledge engineering tools like GIPO (Simpson *et al.* 2001) or KANAL (Kim & Blythe 2003) usually offer some graphical editing of planning domains. They also include other very useful features such as debugging and error checking. The generated plans can be exported to a planning language like STRIPS and thus be reused in very different planning systems. An application to emergent narrative authoring (*FearNot!* authoring in particular) would be possible. One would only have to provide a piece of software to translate from the syntax generated by the plan authoring tool to the syntax used by *FearNot!* and vice versa. Compared to hand coding this is definitely an improvement, but since those plan authoring tools are not domain specific, they will not cover all that is required by the specific domain of narrative, in case of *FearNot!* for example the reactive level of an agent's mind.

Storyboarding tools

Another quite different sort of authoring tools is storyboarding tools. This type of software is very user friendly and easy to use for non technical authors. The programs Kar2ouche and Mediastage by the company Immersive Education are examples of story boarding tools. To varying degrees tools like this allow an author to easily create or even perform a story and capture that performance, which can then be used to demonstrate ideas to other people, just like story boards are used in the film production process. Storyboarding tools of course cannot be used to create intelligent characters, but the user interfaces of those tools are worth paying attention to, as they are very accessible - and as we argued earlier, that is one of the key factors, if we want creative authors to start building interactive story worlds.

Authoring by learning

The kind of software that has been reviewed so far is either too simple or too complex for our intended task. By that, we mean that the tools that allow to create real narrative content are very complex and too much like real programming environments for our intended user group whereas the tools with an easy user interface only allow superficial content creation like placing characters and objects in a 3D environment. This might in fact be an inevitable conflict, when we look at the complexity of the narrative content that we want

to create with the help of those tools. A solution however, might be an authoring tool with some intelligence that gives the user an easy interface and adds the required complexity by employing AI techniques like machine learning. In a recent interview with the Games For Windows Magazine Façade creator Michael Mateas said: "In order to move toward more radical end-user authoring, the authoring tools themselves have to start knowing something about characters and stories, and offer A.I. assistance to the author to take the bit of authoring they do and multiply the effort."[1]
The idea of AI assisted content creation is reflected in the field of 'Programming by Example', which is described in (Cypher 1993). The idea here is that if a user knows how to perform a certain task at the computer, instead of writing a program to solve it, the user can instruct a software to 'watch what I do'. By observing the user's actions the software can then learn to perform those actions itself.
The aforementioned plan authoring tool GIPO also contains a module that allows the induction of operator descriptions from examples (McCluskey, Richardson, & Simpson 2002). By providing examples for solutions to a certain problem the program can infer the operator (action) descriptions. Applied to our narrative planning domain this would be very useful and ease the task of the author. But also in the field of interactive narrative itself, examples for intelligent authoring can be found. In Thespian (Si, Marsella, & Pynadath 2005), a character based storytelling system, the author can adjust the personality of a character by providing a set of example stories/scripts that demonstrate how this character behaves. The software will then automatically modify certain parameters related to the characters goals, in order to best match the demonstrated examples.
At the MIT Media Lab, researcher Jeff Orkin has recently launched a project, entitled *The Restaurant*[2]. In the first stage of this project a computer game for 2 players was released, in which one takes the role of the waiter and the other one that of a guest in a restaurant. Members of the public were encouraged to play the game online on the MIT's servers, with all game sessions being logged. The project website states that the long term goal of *The Restaurant* is to process the logged game sessions with machine learning algorithms and to use the collected data to create a single player game, in which the role of waiter or guest would be taken by a synthetic character.

Summarizing, this review showed that there are many different approaches to authoring, which all are useful for certain purposes. There is no existing solution that solves our problem of user friendly authoring of an emergent narrative planing domain, but in our task of creating that solution, we can draw from all the mentioned work, as we will show in the next section.

[1]Games For Windows Magazine, Issue 6, May 2007, page 34

[2]http://web.media.mit.edu/~jorkin/restaurant/

A suggested authoring environment

In our emergent narrative framework the major authoring task is to create the minds of the intelligent virtual agents. A substantial part of this task involves configuring the planners that drive the characters, i.e. by specifying the actions they can perform, the goals they have and the way they react to events. We are now going to introduce the design of an application that allows the author to do this as intuitively as possible.
In a nutshell our proposed authoring environment can be best described as a rehearsal space for intelligent virtual actors, where the author would take the role of the director. The author would first set the scene for a rehearsal, by deciding on a setting and placing characters and objects in this setting. This stage of the authoring process is similar to the traditional authoring tools and game editors mentioned earlier. Once the scene is set, the author would start a new rehearsal. During this rehearsal the author controls all participating characters and selects their actions (or inactivity, if that is desired). We will also adopt the authoring by learning strategy, so for the software itself the rehearsal serves as learning material to extrapolate data that represents character behavior like new actions or goals. By running through a number of rehearsals the character's planning domain would gradually grow until it is eventually detailed enough to use the character as an autonomous actor in a virtual drama that exhibits the desired behavior. Regarding the look and feel of this authoring environment, games like *The Sims* but also the storyboarding tools mentioned earlier provide good examples of effective user interfaces for an application with our intended purpose.

Mixed initiative

Taking the theater metaphor a bit further, a real theater director has to engage in discussions with his actors during rehearsals. In order for the actors to really delve into their role the director has to communicate his vision of the character to his actors. Similarly, our system will be able to build characters much quicker and better if the author communicates with it. For that purpose we intend to run the rehearsals in a mixed initiative planning mode. Instead of merely working with lifeless puppets, the character's minds will actually be activated during the rehearsals. Characters will be able to make decisions and act autonomously in the rehearsals. This is however, merely a way to assist the author, who ultimately has the last word on the character's actions. At any time the author can override the character's planning process and order the character to do something. This interaction between a human user and an AI planner is called mixed initiative planning. The idea of applying this to our field is not entirely new: Although, targeted at a different system architecture (plot rather than character based), the work described in (Thomas & Young 2006), suggests that interactive narrative authoring could benefit from mixed initiative planning.

Communicating the author's intent

In our system, each time the author interferes with the planner for a character, they will be asked to justify their actions. That might either happen, when the author orders the character to carry out an action that the character would not have come up with itself or if the author cancels an action that the character would have carried out by itself. This justification of the author's actions serves two purposes: Primarily it is a way for the software to collect data. Although it is possible for the software to extract some information from merely observing action sequences, it will be much more efficient if the actions are backed up with some background information. Imagine for example a barkeeper agent. That agent would appear in a number of stories and it would quickly learn the actions associated with his role of selling drinks. However from time to time in a rehearsal the author would cancel the agent's learned action of selling a drink to someone who requests it. If the author has the chance to justify their decision, they can very quickly teach the barkeeper agent that there are exceptions to the 'sell drink' rule, for example there is no selling of alcohol to customers that are underage or too intoxicated. Technically the software could encode those exceptions for example as pre-conditions of the sell drink goal. If, however, the author had no way of communicating their motivation, it would take many rehearsals, before the software would have collected enough examples of barkeeper related behavior, to learn those exceptions by itself. Eventually, with enough examples to observe, it would still learn this though, so we will not force the author to justify their actions. Otherwise one could even argue, that this action justification process is a way to prevent the software from using machine learning.

The second reason to ask the author for action justification is that by doing this we force them to think more in terms of character and less in terms of plot. If an author would use the same character and let it exhibit radically different behavior during several rehearsals, the software would be confused and would confront the author with a lot of questions about the motivation behind their directing orders. Hopefully this would help in forming an awareness in the author about the technical capabilities of the kind of system they are dealing with. For example it might be a bit overambitious just yet to model the mind of a schizophrenic character with the current technologies. The idea that being forced to justify an actor's action is facilitating the author's reflection process, is backed up by findings of the virtual Theater project Teatrix (Paiva, Machado, & Prada 2001). Teatrix is a system that has in fact a lot in common with our authoring environment. Children take the role of a virtual character in a collaborative environment. Other children take the role of other characters while unallocated roles are filled out by intelligent virtual agents. While a child acts, the character that is played by the child is also simulated in the background using an agent model based on Vladimir Propp's analysis of the basic plot components of Russian folk tales. If the actions the child takes diverge too far from the actions that are suggested by the role, the hot-seating tool is activated. Here the child has to justify their decisions and assess the character's feelings. Teatrix evaluation results suggest that the hot seating tool helped the children in reflecting about the role of the character they play.

Figure 3 summarizes the main authoring method with included mixed-initiative and action justification features.

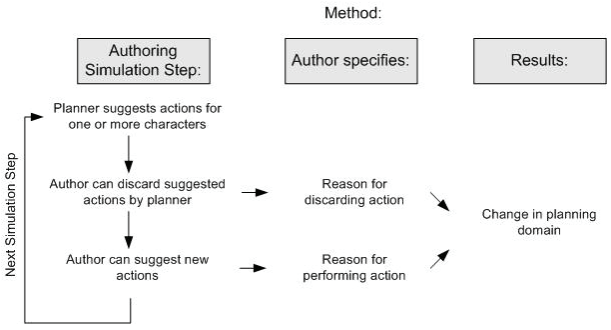

Figure 3: main authoring method

As the figure suggests we are not envisioning a real time application but rather one where the story is divided into concrete time steps. A new time step would occur whenever an agent has finished carrying out an action. The author is in complete control of the time-flow during the rehearsal. The author decides when to step forward in time and furthermore will also be able to rewind time, just like a theater director during a rehearsal could also decide to restart again, beginning from a certain scene.

Since we are using the theater metaphor quite extensively throughout this paper, it should be made clear at this point that it only works to a certain degree and should not be taken too literally. In a real theater environment the same story is played during each rehearsal. In our case, the rehearsals need to be different stories or at least different variants of the same story (although using the same characters), otherwise there would be no new knowledge that could be added to the agent minds.

Integrating the authoring method in *FearNot!*

After we introduced the generic authoring method in the last section we are now going to explain its application for authoring *FearNot!* characters, driven by the *FAtiMA* architecture. Especially we are going to focus on how the different components of the *FAtiMA* agent specification as they have been described earlier (goals, actions, emotional reactions, action tendencies) can be created.

Graphical presentation

The authoring method we suggested earlier does not necessarily require a 3D authoring environment, but the *FearNot!* authoring tool, we are going to create will be using 3D characters in 3D environments for several reasons, the main one being reusability. The environment in which the rehearsals take place needs to match the environment in which the agents will finally perform. Since we already have an integrated 3D graphics engine, a collection of 3D characters

and settings and a behavior planner for performing physical actions in a 3D environment for *FearNot!*, we should also use those components during the authoring process. If we want our characters to perform in a 3D environment later, we should also train them in a 3D environment. That means that visually the authoring environment will look similar to *FearNot!* (see Figure 4).

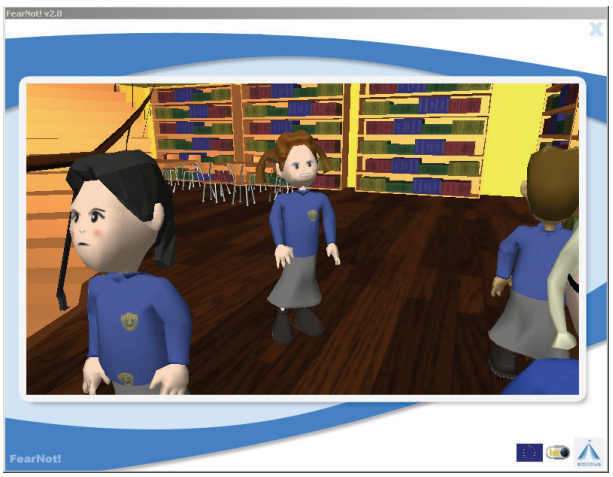

Figure 4: screen shot of *FearNot!*

Initial state

We will start our discussion by defining the state of the software, whenever the author starts working on a new set of characters, i.e. before they start a series of rehearsals. Since the software cannot know in which direction the characters are going to develop it is desirable that there are no predefined goals, action tendencies or emotional reactions. Of course one could also imagine the author choosing one of several character templates that already contain some very common behaviors for a certain personality or role, but for our first design we will not consider that. The more interesting question is how we treat actions. Since it is part of our design that the author acts for a character, they need to be able to select an action from a list of available actions. Since every action at the mind level needs to have a correspondent at the physical (i.e. graphical animation) level, we need to provide a list of available actions, which is defined by the available animations and other physical actions in the system. The alternative would be to let the author create new actions, but in order to do that, they would also have to define the animation or other physical effect of this action. It seems more appropriate to use existing 3D graphic design tools to create graphical content such as animations. The integration of such a component would be too big a diversion from our goal.

That means the basic actions themselves are already part of the initial state and they are knowledge that every 'newborn' agent shares. However, that does not mean that the action library will not grow and be refined during the authoring process. The initial state contains only basic actions, but during the authoring process specializations of those actions might be added. Consider for example the basic action Say([speechact],[target]). This action can be used by the character to say anything to anyone, as [speechact] is a variable that can be replaced by any speech act and [target] can be replaced by any character. However, there might be certain assignments for those variables, when the say action has special pre conditions and/or effects and in this case a new specialized action could be created. For example, the specialized action Say(partyinvitation,[target]) might be a specific entry in the action library that the agent adds during the authoring process because it learned that this action has additional specific effects and should only be performed, when there is a party to invite someone to.

Setting up a new rehearsal

Before a rehearsal can be started, it needs to be set up. In order to set up a new rehearsal, we need to define the setting, the characters that are participating and the props involved. We might also want to predefine certain properties of items and characters, for example that character A is hungry or character B is injured. This task very much like defining one episode for *FearNot!*. We mentioned earlier that a scenario authoring tool for *FearNot!* had already been developed (see Figure 5) , the main purpose of which is in fact setting up episodes for *FearNot!*. With a few modifications we might be able to adapt this tool to set up a rehearsal instead.

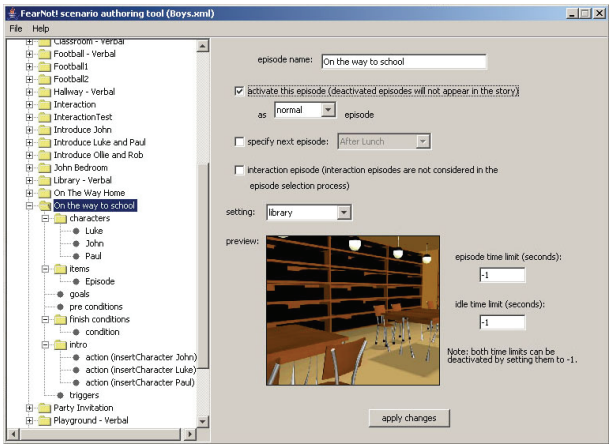

Figure 5: screen shot of the existing scenario authoring tool for *FearNot!*

Ordering an action

When the author issues the order to carry out an action to a character, they will have to specify the reason for that order. According to the *FAtiMA* architecture there are really only two reasons, why an agent performs an action: It is either an instinctive reaction to an event and/or emotional state (action tendency) or part of a plan to achieve a certain goal. In the former case, the author will have to define which emotion triggered the action and whether it was just the emotion itself or the emotion coupled with a recent event. This is enough information for the system to build a new action

tendency. In case of an action that is part of a plan, it is slightly more complicated. Since we start with an empty goal library, the author will have to name goals during the authoring process. The author has to specify which goal an action contributes to. This could be either a new goal (action is the first step in the plan) or an already active goal. It might also be possible that by carrying out the action a goal will be achieved. If this is the case the author also needs to indicate this. The author should also indicate whenever an action leads to the failure of any active goal of any character. By comparing several plans that lead to the same goal's success or failure the system can generalize a goal's pre, success and failure conditions after a few rehearsals. Additionally to this required information expert authors can also decide to specify their intent in a more detailed way. For example when starting a new goal, they could indicate which part of the current world state was responsible for activating this goal. The same applies to goal failure and success.

Canceling an action

Canceling an action that a character was planning to carry out is the inverse process of ordering an action. Now the character specifies why it wants to carry out that action, just as the author justifies their motivation when ordering an action. The character could either tell the author that the action is an action tendency or part of a plan. It is now the authors task, should the character's justification not please them, to correct the character. At least the author has to tell the character that the action does not work in a plan toward a certain goal, that it is not triggered by a certain action tendency or that it endangers another goal. Again, the author can decide to help the learning process by being more specific: If the action was an action tendency the author could specify an exception, that distinguishes the triggering event from other events where the action tendency would apply. For actions that are part of a plan they could specify why a goal cannot be activated in a certain situation or why this particular action does not lead to the success of a goal as expected by the character.

Referring to world states

In the previous sections, we have spoken about world states that the author is referring to, for example to indicate an exception. We earlier mentioned the example of the barkeeper who should learn not to sell drinks to underage customers. If the author wanted our agent to learn that exception to the rule, they would need a way to refer to that particular property (the age of the current customer) as the discriminating factor. This will probably happen by selecting an entity in the world (a character or an item) and then selecting the property from the list of properties this entity possesses. Additionally we must however also consider the possibility that the discriminating property we would like to refer to is not part of the world model yet. In our example this would mean the world model does not include the age of agents as a property. In this case we must allow the author to create the new property within the authoring tool. Apart from properties, those distinct world states could also refer to past events, a character's emotional state or a conjunction and/or disjunction involving several of these. We hope that by using a user friendly interface, which is inspired by the metaphor of a dialog between the agent and the author, this will not become too complex.

Emotional Reactions

We already have explained how the software can slowly create a library of actions, goals and action tendencies through several rehearsals. We have not defined yet however how the emotional reactions of a character are defined. Emotional reactions are very important in the *FAtiMA* emergent narrative architecture and they are required to make the story dynamic but as they reflect a very subtle part of the human mind, they can hardly be learned by the authoring software through story observation. That means the author will have to specify them manually. We should however still think about ways of how this manual specification can be best integrated within our authoring framework. Emotional reactions are the appraisal rules that define how a character perceives an event and which emotions that triggers in him. *FAtiMA* expects an emotional reaction to be specified through 3 values: the praiseworthiness (subjective moral judgment of an action, i.e. was it right or wrong), the desirability (does the agent like it or not) and the desirability for other agents involved. Out of these 3 values, several emotions can be generated according to the OCC (Ortony, Clore, & Collins 1988) emotional model. The best way we found of integrating that into our authoring environment would look something like Figure 6.

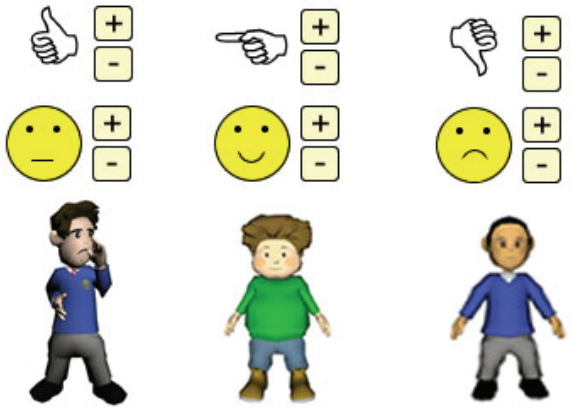

Figure 6: a possible way to define emotional reactions

Whenever an event occurs the author could switch to a mode where symbols would appear above the characters heads. Those symbols would indicate every character's subjective feeling of praiseworthiness(hand)and desirability (smilies) of the event. The values could be adjusted by the author. When a character already has an emotional reaction for a certain event that would be automatically displayed, so those symbols would also be a good way of displaying the character's emotional state during the authoring process.

The 'desirability for other' value would not have to be additionally specified, since it should be equal to the other agent's desirability value.

Conclusion

We think that an authoring tool as described in this paper could possibly solve the problem of authoring emergent narratives. Allowing an author to specify content in the form of linear stories should ease the transition to the different mindset that is required by emergent narratives. At the same time by being asked to justify the decisions that a character makes, the author will be assisted in thinking in terms of character. Working in a mixed initiative mode, i.e. not 'switching off' the characters minds during the rehearsals will not only help the author in getting a better idea of what a character has learned during previous rehearsals but also speed up the authoring process, since obvious and repetitive actions are taken by the character autonomously, so that the author can focus on the actions that are new for the character.

Our future work will now focus on the actual implementation of the work described here. We try to approach the software development using the spiral methodology, which suggests several iterations of the stages 'understanding the requirements', 'design', 'implementation' and 'test'. While we will soon enter the first 'implementation' stage, tests of the initial prototype system will probably reveal new requirements, which will then have to implemented again. We will also use tests with the first prototype to decide how we deal with some more complex questions, for example how to prevent the software making wrong conclusions or how to determine how many rehearsals are necessary.

Acknowledgment

This paper is supported by the eCIRCUS (Contract no. IST-4-027656-STP) project carried out with the provision of the European Community in the Framework VI Programme, and by a scholarship (SFRH BD/19481/2004) granted by the Fundação para a Ciência e a Tecnologia. The authors are solely responsible for the content of this publication. It does not represent the opinion of the European Community or the Fundação para a Ciência e Tecnologia, which are not responsible for any use that might be made of data appearing therein.

References

Aylett, R.; Dias, J.; and Paiva, A. 2006. An affectively driven planner for synthetic characters. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 2–10. AAAI press.

Crawford, C. 2005. *Chris Crawford on interactive storytelling*. New Riders.

Cypher, A., ed. 1993. *Watch What I Do: Programming by Demonstration*. MIT Press.

Kim, J., and Blythe, J. 2003. Supporting plan authoring and analysis. In *proceedings of the 8th international conference on Intelligent user interfaces*, 109–116.

Kriegel, M., and Aylett, R. 2007. A mixed initiative authoring environment for emergent narrative planning domains. In *Proceedings of the AISB Annual Convention*, 453–456.

Louchart, S., and Aylett, R. 2004. Narrative theory and emergent interactive narrative. *Int. J. of Continuing Engineering Education and Life-long Learning* 14(6):506–518.

Mateas, M., and Stern, A. 2005. Structuring content in the facade interactive drama architecture. In *Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*. AAAI press.

McCluskey, T. L.; Richardson, N. E.; and Simpson, R. M. 2002. An interactive method for inducing operator descriptions. In *Proceedings of the 6th International Conference on AI Planning and Scheduling (AIPS-2002), Toulouse, France*.

Medler, B., and Magerko, B. 2006. Scribe: A tool for authoring event driven interactive drama. In *Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*, 139–150. Springer.

Ortony, A.; Clore, G.; and Collins, A. 1988. *The cognitive structure of emotions*. Cambridge University Press.

Paiva, A.; Machado, I.; and Prada, R. 2001. The child behind the character. *special issue on Socially Intelligent Agents. Editor Dautenhahn, K. - IEEE Systems, Man and Cybernetics Society* 31.

Sauer, S.; Osswald, K.; Wielemans, X.; and Stifter, M. 2006. U-create: Creative authoring tools for edutainment applications. In *Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*, 163–168. Springer.

Si, M.; Marsella, S. C.; and Pynadath, D. V. 2005. Thespian: Using multiagent fitting to craft interactive drama. In *Autonomous Agents and Multi Agent Systems (AAMAS)*, 21–28. IEEE Computer Society.

Simpson, R. M.; McCluskey, T. L.; Zhao, W.; Aylett, R.; and Doniat, C. 2001. An integrated graphical tool to support knowledge engineering in ai planning. In *European Conference on Planning, Toledo, Spain*.

Thomas, J., and Young, R. M. 2006. Author in the loop: Using mixed-initiative planning to improve interactive narrative. *ICAPS 2006 Workshop on AI Planning for Computer Games and Synthetic Characters*.

Zagalo, N.; Goebel, S.; Torres, A.; Malkewitz, R.; and Branco, V. 2006. Inscape: Emotion expression and experience in an authoring environment. In *Technologies for Interactive Digital Storytelling and Entertainment (TIDSE)*, 219–230. Springer.