D 3.2

EDUCATIONAL MATERIAL FOR CREATORS (NON-COMPUTER-SCIENTISTS) ON AI-BASED GENERATIVE NARRATIVE METHODS

Project Number	FP7-ICT-231824
Project Title	Integrating Research in Interactive Storytelling (NoE)
Deliverable Number	D3.2
Title of Deliverable	Educational Material for Creators (Non-Computer-Scientists) on AI-based Generative Narrative Methods
Workpackage No. and Title	WP3 - Authoring Tools and Creation Methods
Workpackage Leader	HSRM
Deliverable Nature	Report / Online Material
Dissemination Level	PU
Status	Final
Contractual Delivery Date	31th December 2010
Actual Delivery Date	10th Jan. 2011
Author(s) / Contributor(s)	Ulrike Spierling (FHE), Steve Hoffmann (FHE), Nicolas Szilas (UNIGE), Urs Richle (UNIGE)
Number of Pages	70

Table of Contents

Abstract

This report summarises and gives examples of educational material targeted at authors, which has been developed during the second year of the IRIS project in the work package of "Authoring Tools and Creation Methods". It sets out with a description of motivations, hypotheses and the state of the art concerning knowledge resources for story creators in AI-based Interactive Storytelling. The material is delivered in its first release with pointers to Online external resources (see List of Appendices) updated subsequently. It has been evaluated in parts at an international tutorial and workshop (ICIDS 2010) and practically in our Universities.

The material consists of theoretical introductions and practical exercises in offline conception and content implementation by use of authoring tools and script languages. It is organised into the following categories. Besides general introductions, each category introduces a new concept and contains exercises with one target IS system provided by IRIS partners.

- General introduction: Presentation and discussion of novel creative principles, such as abstraction of stories and thinking in conditional acting situations.
- Specific concepts:
    - Modelling conversation-based stories
    - Modelling stories with goals, tasks and obstacles
    - Modelling stories with AI-based planning
    - Modelling stories with state machines

The appendices include slides, software and tutorial files for download, and a printer-ready card game for the explanation of modelling planning operators. Further, as the material includes newly created interactive storyworlds, there are reports and draft materials available on the creation processes.

All updates are available at the central link http://iris.interactive-storytelling.de.

1. Introduction

1.1 Motivation and Assumptions

One of the IRIS project's main objectives is to *"make the next generation of Interactive Storytelling technologies more accessible to authors (scriptwriters, storyboarders, game designers), so as to make possible the transition to Interactive Narrative"*. The scientific progress to be made is the *"development of authoring methodologies independent from specific implementations"*. The goals of this work have been defined as, more specifically,

- to find or develop creation methods (general and/or specific to one approach),
- to conceptualise authoring tools dealing with the generative nature of IS, and
- to develop authoring and creation philosophies, connecting AI-based generation with traditional creation as best as possible.

In year one of the IRIS project, existing creative principles in storytelling domains and research in IS authoring approaches have been analysed and been compared. The result of this work has been described in the Deliverable 3.1 (IRIS-WP3, 2009), "Report on Pre-scriptive Narrative Formalisms and Creation Methods in Interactive Storytelling (Non-Digital and Digital)".[1] That report concluded with the detailed description of a gap between generative technical approaches for AI-based Interactive Storytelling and yet-to-be-developed creative knowledge to use these approaches in creative productions. Two directions of further work to overcome this barrier have been identified (see Fig. 1):

- Development of better accessible tools for authoring with generative engines, while lowering the threshold of technical knowledge requirements for authors, especially regarding the conception of interactive storyworlds.
- Education of potential / prospective authors in the technical principles lying behind these generative approaches, however only as far as this is necessary for creative conception – thereby focusing on the differences in story conception compared with their pre-knowledge of story creation using existing tools.

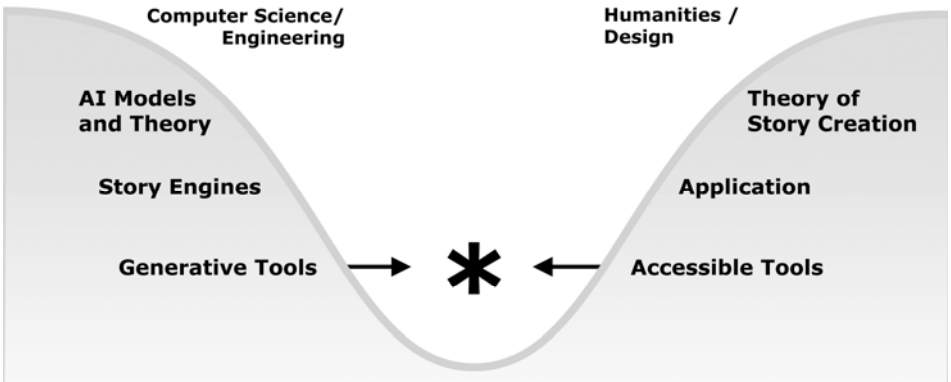

Fig. 1. Approaching the authoring problem from two directions

The tasks performed during the second IRIS year in the work package "Authoring Tools and Creation Methods" circled around both directions, however with an emphasis on the second

[1] IRIS Deliverable 3.1, publicly available at http://iris.scm.tees.ac.uk/publications#Deliverables

motivation of developing creative principles that can be used for the education of authors and story creators to access existing generative tools (and which, as a side effect, can also be directed at tool developers to support a common understanding of design processes). This Deliverable focuses on the description of this resulting "Educational Material". It consists of presentation material and practical exercises with existing tools. These used tools mostly stem from the IRIS project, and have partially been under construction during the exercises and further on during the project. The material has been developed by practical elaboration of assumed design principles in the IRIS team, at the same time leading to several example conceptions of interactive storyworlds. In these exercises, two professional story creators of the IRIS interest group in authoring have been directly involved (Urs Richle, Georg Struck). Parts of the material have then been evaluated during a pre-conference tutorial at ICIDS 2010 in Edinburgh (Spierling et al., 2010a).

1.1.1 Assumptions on a Collaborative Process

The scope delimiting the authoring process in our sense has been set out in the beginning of the project and has been described in Deliverable 3.1 (ICIDS-WP3, 2009). Namely, we assume that authoring and/or story creation in IRIS means the creation of a dynamic storyworld, which needs a runtime engine to be performed interactively by an end-user. This runtime engine is presumably developed in a complex software development process that either occurs before, alongside or interwoven with the development of a particular storyworld. The roles of authors therefore have to be located within a potentially interdisciplinary team, which also means that the task of programming storyworld rules and more complex dynamic models can be distributed.

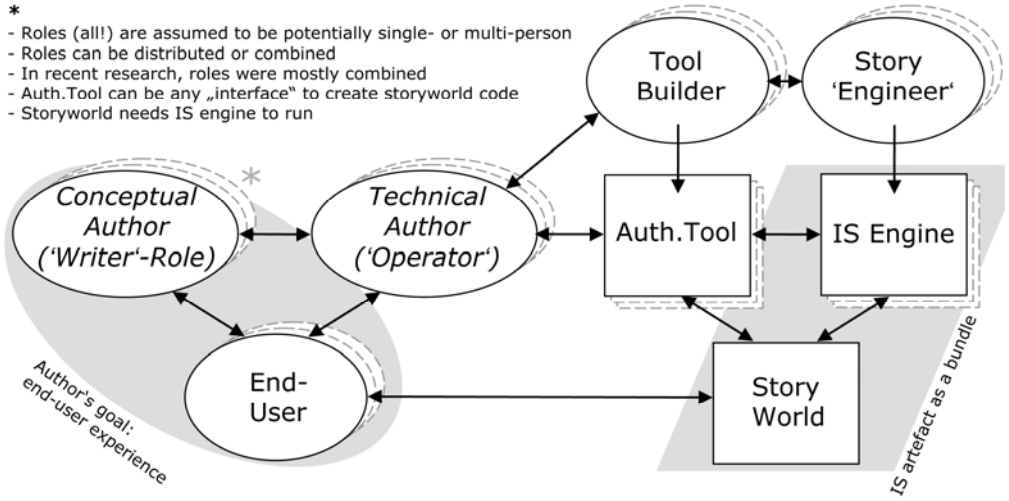

Fig. 2. Assumed development model of an interactive story artefact

Fig. 2 depicts this collaborative model. It is the result of interdisciplinary discussions about an ideal, future model of development of interactive story artefacts; it is not known to have been implemented within the IDS research community yet. Reportedly, this is due to resource issues because of the academic research character of most projects. In an ideal conception of the development process, the following roles can be assumed:

- Conceptual authors (CA) / writers come up with ideas of experiences they want to create for their audience, the end-users, and follow their idea through up to implementation and verification, comparable to directors in film or to lead game designers. They are advocates of end-users and envision the end-user experience.
- Technical authors (TA) are able to operate authoring tools to directly implement a storyworld. If they are not the same person as the CA, they work together with CAs in a tandem manner. It is important for CAs to get feedback of their conception and change

it accordingly and repeatedly, therefore it is necessary that storyworld versions can be built in an iterative process.

- Story engineers (SE) are responsible for the functionality of a story engine. Tool builders (TB) implement authoring tools, the gauges and handles of which are negotiated between TAs and the engine's functionality (the SE).

It can be assumed that individuals take more than one role at once, for example, as conceptual author and technical author in one person, or that the roles are distributed even wider (more than one conceptual author, etc.). The concurrence of roles has been the reality in recent research projects with generative engines. However, this situation has also been reported to be improvable, in order to allow specialists in storytelling and engineering to concentrate on their expert knowledge, as well as to scale up future projects.

1.1.2 Drafts of Content and Steps of Abstraction

From the engine's point of view, the 'content' – as the result of the authoring process – is storyworld code, saved as a data structure. Mostly, it contains not only declarations of events, entities and their parameters, but also rules as code that the engine can operate on. For many existing story engines, unique / proprietary XML dialects have been developed, which mostly structure the declarative parts of a storyworld in readable terms that lend themselves from narratological domains. Examples are IDtension (Szilas, 2007), Scenejo (Spierling et al., 2006), Storytron (Crawford, 2004), FearNot! (Aylett et al., 2005). Other examples exist where the code refers to technical domains, for example planning, by directly using PDDL code as technical content representation (Porteous and Cavazza, 2009) or simulation, by using Prolog (Swartjes and Theune, 2008). Authors need to be able to express causal relationships in a storyworld.

Crawford suggested: *"You need a language that allows you to express with clarity and precision the exact nature of each causal relationship […]. I have good news and bad news for you. The good news is that this language has already been developed; the bad news is that it's mathematics."* (Crawford, 2004, p. 101) Whether *"it's mathematics"* or Crawford just wanted to funnily tease authors who try to avoid abstract thinking, he made a point that we later want to take up with creative principles. Also Stern argued that *"authors must program"* – apparently only for the same reason (Stern, 2001). Actually, the idea of authoring tools is that they provide an easy way for non-programmers to create that code without typing it, by using a GUI. However, as illustrated in Fig. 3, the translation of a written script into code can not be assumed to be a straight-forward process, therefore programming is not the only issue.

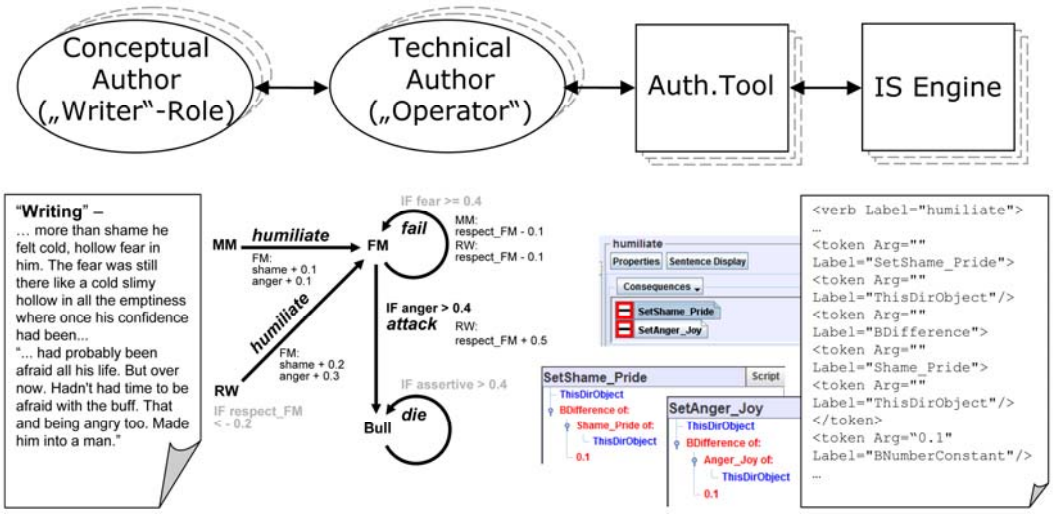

Fig. 3. Content abstraction example of an adaptation of a Hemingway short story to an interactive version with the Storytron engine (Spierling and Hoffmann, 2010)

The distribution of tasks in a team naturally implies that the communicative effort increases. This implies that even though conceptual authors might not need to programme themselves, they need a conceptual understanding of technical principles. In our view, writers cannot just hand a written outline to the technical implementers of the experience and let them implement it, because in the end they may want to co-design the interaction concepts with the generic features of the engine, in order to maintain shared control over the end-user experience. Adopting pair programming techniques from agile software development (Cockburn and Williams, 2001) and finding intermediate draft representation forms is envisioned as something that future work may include. The principles described herein build upon this assumption that in between the 'written story outline' and the 'code production' (with or without a GUI) lie important conceptual steps of building dynamic models, thereby preparing a storyworld for user interaction. It is this intermediate conceptual stage between a linear story outline and using the implementation tools where our educational material is meant to be applicable.

1.2 State of the Art

To our knowledge, there is hardly any educational material available that fulfils the goal of a general introduction to creative design principles for interactive storytelling with generative engines, except (Crawford, 2004) – see below. The state of the art based on conference publications in the domain of "authoring" has been revisited in Deliverable 3.1 (IRIS-WP3, 2009). Of these publications, most have been descriptions of concrete tools. A number of research papers in the area of planning refer to authoring. It has been noted, however, that this literature is generally hard (or even impossible) to grasp when taken as a primer for authors and creators who are not familiar with AI concepts. Certainly this is not at all the purpose of research papers targeted at a cutting-edge audience in their domain. However, it is – for example – the only literature available that makes the connection of AI planning and storytelling, and within the research community, the advice to read the papers has often been heard.

Authoring for planning can mean the construction of a hierarchical task network structure (Charles et al., 2003) or, most often, writing a domain based on STRIPS-like propositions and operators (Russel and Norvig, 2003, p.377 ff.). This is per se a straightforward task. However, from an author point of view, without practice, the consequences of own creations are hard to imagine. Pizzi and Cavazza (2008) have given an overview of the creation cycle involving several test runs and 'debugging' the planning domain. A planner alone can achieve automated plot, but is no guarantee for highly-interactive storytelling. However, planning can be a strategy for drama management and adaptation of plot to user actions. A simplest introduction to the employment of planning has been provided by Orkin (2006), the creator of the F.E.A.R. game.

The concept of intelligent agents for storytelling has been used in many approaches, usually covered in various domains outside the point of view of creative storytelling. The concept can also involve planning. Again, there is only little information about creation and storytelling from an author's point of view. Typically, agents are configured by their character traits and goals, which is a top-down approach. It is difficult to imagine concrete actions as an outcome of mere specification of high-level concepts. Current AI research investigates how goals can inversely be informed or 'fitted' by letting authors make local changes, as these are more easily to control (Si et al., 2007). A similar philosophy is that of a 'rehearsal' of the actions, letting the agent 'learn' a certain reaction in a certain context by specifying authorial context information (Kriegel et al., 2007).

Concerning creation principles, there is a body of publications stressing the topic of the 'narrative paradox' of emergent narrative, discussing scopes of authoring (Louchart and Aylett, 2005). Emergent narrative can either be seen as another means to achieve higher levels of interactivity, or as an end in itself, as in improvisational theatre. Radical claims of emergent narrative actually argue that there is no justified existence of an 'author' – as emergence and design are contradictions in terms. Accordingly, the author shall be replaced by purely generative mechanisms. There are also less radical points of view trying to integrate concepts of design and emergence, with concepts how authors should partially 'let go' of their will to

control the storyworld completely (Louchart et al., 2008). However, it is not easy to derive from this literature what tasks are left for authors to be performed during conception.

Similar, but without making extensive reference to 'emergence', are publications that report on possible general creation and authoring concepts that shall make IDS storyworlds more interactive. These are the many reports by Mateas and Stern (2005a) on the creation of the interactive drama Façade, with the claim for the concept of *"Procedural Authorship"*. This implies that authors have to think procedurally when creating an interactive storyworld, which they refer to the ability to programme. Although these papers have AI background, they do contain 'untechnical' / conceptual parts, informing authors about general design strategies of building interactive storytelling.

A book that is exclusively dedicated to typical creation issues in Interactive Storytelling is "Chris Crawford on Interactive Storytelling" (Crawford, 2004). It is unique in the sense that Crawford speaks to story creators instead of to engineers. He offers design strategies, mainly elaborating on the philosophy behind the Storytron system. The book contains many anecdotal references to creation in practice, some of which indeed relate to aspects that we have found and suggest to be general principles. For example, Crawford stresses the importance of 'abstraction' when modelling a storyworld, and the problems that occur when authors do not think about the player's possibilities from the beginning of conception. However, Crawford does not address basic conception phases of a storyworld, which at first may be independent of a system for implementation, as all the concepts are targeted at the Storytron system. In other words, when compared with our approach sketched in Fig. 3, Crawford leaves out the draft phases before (directly) using the authoring tool SWAT for implementation.

1.3 Approach and State of Results

The concepts presented in section 3 build the base for educational material that has been and will be used with different target groups of potential IS authors. As such and in its current state, it has to be employed within facilitated courses or tutorials and is not targeted at being used 'standalone' for self-learning. The material is the result of several exemplary content creation exercises performed during the second year of IRIS, a work which is still ongoing with the goal to create a finalised, more complex interactive piece during year 3. The philosophy follows a constructive design science principle of *"Build it to understand it"* (Mateas and Stern, 2005b). In the sense of Lévi-Strauss' concept of intellectual *"bricolage"* (Lévi-Strauss, 1962), we are aware that we use material means at our disposal as they *"come in handy"*, a *"catalogue of a previously determined set consisting of theoretical and practical knowledge, of technical means, which restrict the possible solutions"* (Lévi-Strauss, 1962). This means we deliberately take those tools, engines and practical concepts that are easily available within our project. In summary, these are

- the IDtension engine and authoring language (Szilas, 2007) (http://www.idtension.com/),
- the Scenejo conversational engine and authoring tools (Spierling et al., 2006) (http://www.scenejo.org),
- the planner used for the Madame Bovary Prototype (Pizzi and Cavazza, 2008) (http://redcap.interactive-storytelling.de/authoring-tools/emo-emma/),
- the Advanced Agent Animation system with SceneMaker (Damian et al., 2010) (http://mm-werkstatt.informatik.uni-augsburg.de/projects/aaa/).

As shown in overviews presented in the last deliverable (IRIS-WP3, 2009), the global situation of available tools is such that it is unlikely to do this work without facing natural restrictions in universality. Facing these conditions, our selection of systems for education forms a representative subset of concepts of the current state of the art in Interactive Storytelling. Further, these engines are of a convenient size to be tackled and used in small authoring exercises, rather meant as illustration to help conveying general principles and concepts than to

represent dedicated software instructions. IDtension illustrates concepts of abstraction in terms of instantiation and automatic generation of narrative actions. Scenejo stands for basic conversational principles and shows abstraction of dialogue acts. Utilising planning for automatic plot generation can be illustrated by exercises in the Madame Bovary authoring tool, while basic abstraction principles needed for planning are demonstrated by a designed card game to be used as educational material in a course. In that sense, also paper prototyping and conceptual modelling prior to using authoring tools are motivated. Finally, the Advanced Agent Animation system including the SceneMaker authoring tools are being used to illustrate the concept of state machines. Because of the existing connection to a representation in a 3D game engine (Horde 3D), it will be further used for authoring of a more complex storyworld demonstrator as a joint effort of partners – a task which will be continued through the last year of IRIS.

The first year of IRIS ended with the claim that creative principles have to be developed especially for Interactive Storytelling, which exceed or even partially replace existing storytelling principles from linear media domains. This points to a long-term endeavour, because the validation of these principles can only be done in conjunction with many productions of completed systems and IS experiences. Based on our previous experiences, in the second IRIS year initial general design principles have been enunciated as a suggestion. These principles are outlined in section 2 of this report. Then, brief story creation exercises to be used as educational examples of each principle have been newly conducted or revisited from existing realisations. Hence, at the same time producing this material has been creating more justified insights about general principles. However, almost naturally, as ‘educational material’ this is a snapshot of the current state of the art and will have to be developed further – on the one hand with more (and more complex) examples of one system, and on the other hand also with more diverse systems.

Concluding, we found it important to not only theoretically try to ‘educate’, but to include practical exercises. Especially the dynamic characteristics of certain principles have to be experienced in practice to fully embrace their nature. As an example, we found that static visualisations of graphs cannot fully explain the changing storyworld states when a planner is used, so we designed a card game to be played in order to better understand its procedural aspects. Also conversational principles and abstraction of actions can better be conceived after own creations have led to direct experiences of design decisions in a running example. In that sense, the provided software and tools – which are under construction – are also seen as material to educate potential authors. This report summarises principles in section 2 and presents the current state of the material in section 3, which connects readers to appendices and links to Online examples, which are meant to be updated subsequently.

2. Novel Creative Principles

2.1 Introduction and Motivation

In (IRIS-WP3, 2009), we have analysed and summarised creative principles that are at work in several linear and interactive storytelling domains. The conclusion was that for Interactive Storytelling, we would have to critically revisit these principles and potentially find novel ones, which address the particular production features of this new storytelling medium. The material presented herein targets at discussing these novelties that story creators have to face when designing storyworlds to be run by our engines. Especially, we do not want to go deep into basic storytelling knowledge that appears to be consistent over several media forms, such as how to create interesting or sympathetic characters, or how to catch the audience by interesting and conflicting goals. We assume that our target group has basic knowledge about how to tell a story. Instead, the focus of our presentation is on ways to increase the possibilities for interaction in storyworlds, and how to increase variability and flexibility if storyworlds are experienced repeatedly.

In particular, in line with the objectives of the IRIS Network of Excellence, we focus on principles of AI-based Interactive Storytelling. 'AI-based' here means that 'some' form of rule-based system or generative algorithm in the broadest sense is used to solve the typical problems of the 'combinatorial explosion', which is labour-intensive, if all possible variations in an interactive storyworld would have to be hand-crafted by authors.

2.1.1 Why are we exploring 'generative algorithms' in Interactive Storytelling?

We found (actually not very surprisingly) that although in the IS research community, there is a common understanding of the necessity to use AI-based generative storytelling algorithms for Interactive Storytelling, this is not that obvious to everybody in all practical interactive story authoring domains. The motivation for this research has to be explained first. In our courses, we provided the explanation that we want to achieve *"highly-interactive"* storytelling, a term that has once been coined by Kelso et al. (1993) in the *"Oz-Project",* an important predecessor of current Interactive Storytelling research. This can best be explained by examples of running interactive works, the most suitable today – and in the direct heritage of the Oz concepts – being *"Façade"* (Mateas and Stern, 2005a). We presented criteria and degrees of 'highly-interactive' works as

- degrees of bandwidth for communicative input (emotionality, multimodality – these are criteria not yet covered by our educational material, except marginally for language),
- degrees of frequency of interaction (possibilities to interact anytime / often), and
- degrees of significance of interaction (user choices that are meaningful for the story).

On the one hand, some researchers see generative algorithms as a paradigm, for example in the extreme case when talking about *"emergent narrative"* – implying that no explicit authoring of a concrete plot is at work. On the other hand, due to project realities and in most of current research, at most *partially* generative solutions are employed. Fig. 4 shows a gardening metaphor of *"implicit creation"* that has been used to illustrate the difference between implicit and explicit authoring (Spierling, 2007), i.e. the difference between explicitly defining a plot structure and only implying it by rules or by 'providing the seeds'. In completely explicit authoring, all possible paths are predefined and authors directly control all decision points. In completely implicit creation, authors only declare rules and abstract models as a base for a simulation. In that sense, we propose that we want to approach increased implicit creation by setting out from familiar concepts of explicit authoring, and acquire the novel concepts step by step. By doing this, we progressively enhance the possible frequency and significance of interaction, by increasing the variability and flexibility of events in our storyworlds. These

concepts require our two main basic principles of ‘abstraction’ and of ‘conditional events’ (see below), as well as an iterative development process similar to software production, including ‘debugging’ as a principle of approaching a dynamic simulation model of the storyworld.

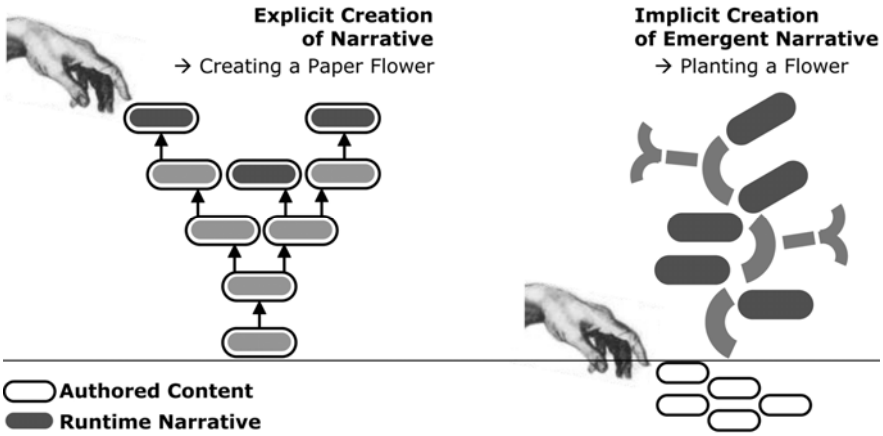

Fig. 4. The gardening metaphor explaining the difference between explicit and implicit creation for generating a narrative structure during runtime.

Writing conditions can typically mean ‘to programme’, however we focus on conceptual levels that occur before implementation issues. In order to be able to create events implicitly, it is necessary to anticipate to a certain extent what is going to happen under specified conditions. Working with a specific engine means that authors in the end need to have applied knowledge about the engine’s ‘mind’ or employed generative formalism and rules; however we believe that it is worthwhile to also consider an engine-independent conceptual phase. It has been found useful when authors are procedurally literate, which often has been compared with ‘being able to program’ (Mateas and Stern, 2005a). But it is more complicated. With implicit creation, there are several ‘unknowns’ at the time of creation. Authors have to conceive at a high abstraction level, and leave detail to an engine.

In short, design principles that have been analysed in several previous exercises and then enunciated in discussions are:

- To integrate **abstraction** as a creative premise for work with almost every potential story engine, and for implicit creation in the above sense,
- to think in **conditional actions and events**, making explicit the underlying conditions of acting situations (for end-users and story characters),
- to start any storyworld conception by **including the end-user**, seeing player actions as part of the abstract storyworld model,
- to practice **character-centred design**, which means decentralised design of actions,
- to **externalise the internal**, which refers to thinking about how to represent inner states and processes by the generation or choice of dedicated events,
- to see end-user **interaction design** as an important part of the authoring tasks, providing interesting acting situations, affordances and feedback, and
- to see **debugging as authoring**, not in the sense of fixing programming mistakes but of embracing iterative processes of model building and experiencing.

In the following, we elaborate on these creative principles. In the material presented in section 3, we show examples of how these principles are put into practice. The main emphasis is on the first two principles: 'Abstraction' and 'Conditional Events'.

2.2 Design Principle: Abstraction

As soon as we want to use generative algorithms, for example to automatically let regulate the flow of events and adapt it to end-user actions, we can only implicitly describe and thereby influence the concrete plot. This means that we write an abstract story world, for example consisting of rules. There are two dimensions of concretion following from implicit stipulations (see Fig. 5).

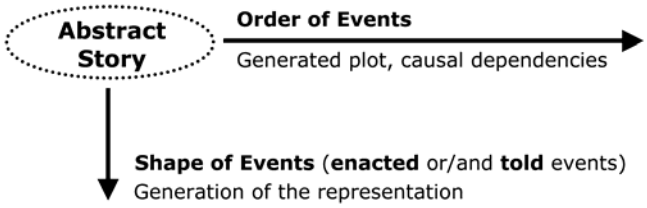

Fig. 5. Two dimensions of potential influence from abstract levels.

Both dimensions in Fig. 5 refer to automatic generation of events or actions. First, the arrow to the right is related to the ad-hoc selection of the best next actions and events suited to build the narrative flow, as a sequence of events. The generation mostly consists in searching a possibility space for suitable and fitting events. The arrow downwards concerns the situated shaping of the representational levels. This can e.g. stand for 3D rendering and behavioural animation, or for automated dialogue generation.

The concept of abstraction is at the heart of most approaches and can be seen as a general principle that authors have to adopt. However, not everything in a storyworld is generated. Therefore, the content to be written can be divided into two parts: a concrete part and an abstract part. Let us take an example in a precursor form of IS, the hypertext. In hypertext, the concrete part consists of all fragments of the piece, while the abstract part consists of the *hyperlinks* among fragments. A hyperlink represents the fact that if a reader comes to see a first fragment, by clicking on this part of the fragment (the anchor) he will then get to read a second fragment. Although we are all familiar with the notion of a hyperlink, it is an abstract concept. In hypertext, the major part of the storyworld creation, that is the writing of fragments, is concrete. In Interactive Storytelling, the balance between the abstract and the concrete can be varying, depending on the concept of the used system (see Fig. 6).

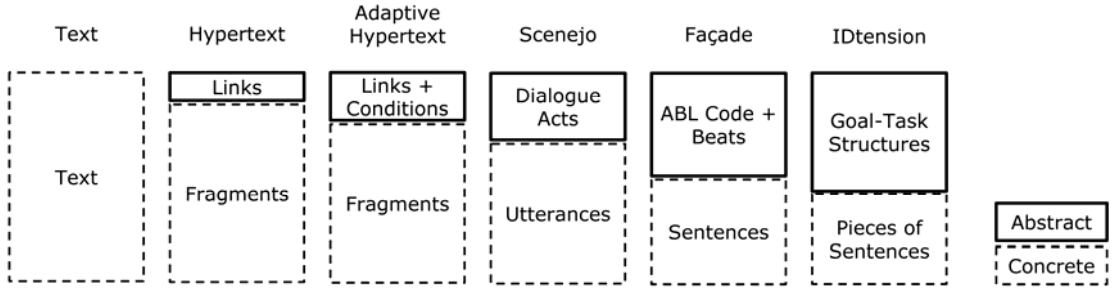

Fig. 6. Varying balance between abstract and concrete content to be written.

In *Scenejo* for example, while concrete writing consists in writing the utterances between characters in a chatbot-like system, a large part of authoring consists in structuring dialogue acts and their rules of chaining. In *Façade*, dialogue lines by themselves are concrete data written by the author, but there is a large abstract part in the authoring process that consists in

writing the mechanics of character behaviours in a specific computer language called *ABL*. In *IDtension*, only pieces of sentences are entered by the author as concrete data, the rest is more abstract data such as goals, tasks to reach the goals, obstacles, and their interrelations.

Traditional writers may not usually like writing at such abstract levels. However, these systems require abstraction in order to provide more variability, or generativity by possibly recombining content dynamically, thus leading to better interactivity. The more abstract a system is, the larger will possibly be the set of events generated during adaption to end-user interaction. For example, with a 'minor generative' system, in order to reach a given level of variability, we need to write a huge amount of alternative concrete data into the storyworld. If we instead adopt a 'highly generative' system, for the same variability we only need to write a reasonable amount of data, but with a larger effort of abstract writing. Abstraction is a tool to write manifold variations of experienced interactive stories within one single art work.

We need to judge how much abstraction we need for our goals. Abstraction is particularly worthwhile for cases in which we require a lot of variability in a big storyworld. If we only want a small degree of variation and generativity, abstract approaches do not make a big difference in workload compared to concrete writing of alternatives, and it is preferable not to use them, as they are more difficult to master. This observation is always an issue when starting from scratch with a new IS system. Either when just stepping into an iterative design approach, or especially as beginners, we usually start with a small storyworld. This small storyworld may not enable us to observe the added value of using a generative system. Only by further extending it, we will finally experience advantages of using AI-based IS technologies.

2.2.1 Different kinds of abstraction

The broad concept of abstraction can be realised in several ways. For example, different kinds of abstraction underlie various IS systems. Further, a system may not force creators to use abstraction, but by using abstract models in the conception stage, greater variability can be achieved.

- Abstraction as generalisation (e.g., for grouping)
- Abstract vs. concrete representation of events (perceptive abstraction)
- Temporal description (concrete) vs. atemporal model (abstract)
- Generic (abstract) vs. instantiated (concrete) events and actions (re-usable 'behaviour')
- Degree of technical formalism

Abstraction as generalisation: We can use abstraction to group elements into more general concepts, for example actions according to their similar pragmatic meanings. This is useful in the modelling of effects of actions or of answers to user input in the storyworld. For example, if we define an abstract dialogue act "greeting", on which we then map concrete utterances such as "Hi", "Hello", etc., we use abstraction to generalise many different ways to utter a semantic or pragmatic expression. It alleviates solving the conceptual issue of defining in every detail what many concrete and single actions can 'do to the world' (see also next principle on conditional events) – regardless if performed by a user or a story character.

Perceptive abstraction: In this case, the more abstract a data, the more distant it is from the end-user representation. Consider the five following ways of representing a given event:

- A fully rendered 3D animated sequence of a character serving a glass of whisky and drinking it (representation experienced by the end-user).
- The same animated sequence, but with only the skeleton of the character
- The script "Joe serves himself a glass of whisky and drinks it in one shot"
- The plotted events: First serve whisky, then drink whisky
- The goal: "Glass of whisky drunk"

While the first way is directly perceptible, the last one requires an effort to imagine what will be the corresponding final experience for the end-user. It is thus more abstract, in the sense that it is farther from the intended representation.

Temporal abstraction: When writing a linear story, we write a succession of events usually represented by a piece of text. In IS, we create models of the situation, which means we may write conditions that specify how narrative events are ordered. For example, partial order is used in IS to specify that one event must occur later than another one, but not necessarily just after it. It is more abstract to think in these terms, because we do not know precisely which event will happen directly after the other. More abstract temporal descriptions exist when the temporal order depends on a specification of a dramatic situation (see also next principle). In temporal abstraction, given descriptions are more abstract in the sense that the order of the resulting narrative events is harder to anticipate at the time of writing.

Abstraction and instantiation: In order to create more variability, IS systems may employ the use of *variables*, a typical computer and mathematical concept. A variable, as indicated by the term, represents the variability of a data, the concretion of which depends on the value this variable data takes during runtime of the story. Consider the two following statements:

- kiss(John,Mary)
- kiss(X,Y)

The second one uses variables, meaning that X and Y can take as values any possible character in the scenario. Statements with variables are often used within rules, for example:

- IF loves(X,Y) AND gender(X) ≠ gender(Y) THEN kiss(X,Y)
  (describing a condition in a conventional heterosexual love story)

In this case of abstraction, we write at the level of *generic* events (events with variables) so that the engine can produce specific events (variables *instantiated* with some specific values).

Formal Abstraction: Finally, as everything that we write as authors needs to be processed by the computer, this content includes formalisms that we might not be accustomed to handle. Consider the three following statements of the same rule:

- If a character loves someone of a different gender, he kisses him or her.
- IF loves(X,Y) AND gender(X) ≠ gender(Y) THEN kiss(X,Y)
- if ((x.relation(Relation.love,y)>0.5 and (x.gender().equals(y.gender))) {
  sendInterCharacterEvent(x,y,kiss) }

The last statement that corresponds to a piece of Java code is obviously more abstract and difficult to handle for an author. Technical authors implementing content in the final form of storyworld code need to master the degree of formalism defined by available authoring tools of the used system, whether it is based on a GUI or a programming language.

Conceptual authors may still want to express their wishes of conditions in less formal ways, but as precisely as possible, to minimise misunderstandings at this stage. The first three kinds of abstraction are most relevant to the conceptual stages of creation (compare also Fig. 3). However, we consider it as not sufficient to leave out the code generation completely in the beginning of a project. Typically, some way of representation of the end-user experience is needed in order to get a grip on the results of the 'generated' story (see next section).

2.2.2 Handling of abstraction during design

While writing at an adequate degree of abstraction is a skill that can be acquired to a certain extent, we found that writing only at abstract levels is not an option. Quite the contrary, in the beginning story creators mostly start out with a concrete imagination of how a storyworld can feel like, presented as a linear outline that later can be turned into abstract models. On the other hand, we also need to be able to quickly get an idea of what are the concrete events and interactive situations that are generated from the abstract data we entered into an IS system.

Therefore, a critical point in IS authoring is the ability to quickly navigate from the abstract to the concrete, and vice-versa (see Fig. 7).

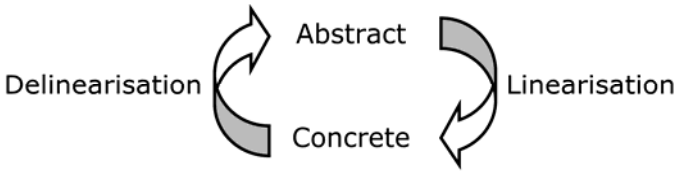

Fig. 7. Iteration through stages between abstract and concrete representations

A key point in IS authoring is to shorten this design cycle between abstract and concrete representations. This means:

- **Concrete to abstract:** As authors, we naturally think in terms of concrete events and their succession. This fact cannot be ignored. Mainly in the beginning of projects, authoring may involve writing concrete and linear parts as a draft form, and then think about how these concrete data can be transformed into more abstract models and abstract representations needed for the used IS system.
- **Abstract to concrete:** As soon as possible after conceiving and coding abstract data, we need to be able to get a corresponding concrete representation. Ideally this is the representation that end-users will get, but there are cases in which we want a reduced form if that is more quickly to explore. This depends on systems as well as on the authoring method we adopt. In IS, a progressive approach is preferred, in which a partial piece of data is quickly tested rather than a more elaborate first story.

The ideal authoring method for shortening the abstract-concrete cycle depends on the used systems, peripheral tools as well as on the composition and skills of the development team. Here are some recommendations that we found useful in an interdisciplinary team of writers and engineers:

- **To start with an outline:** A simple word processor and other traditional tools can be used to start out with an idea.
- **To use pen and paper:** Pen and paper remain appropriate tools for design, as this allows to quickly sketch and write down (or imagine) some possible models and abstractions of concrete events. Therefore, it is useful to define a way to 'draw' the abstract parts, creating *our own* intermediate language between plain ideas put on paper and formal abstract structures for the IS system.
- **Not to over-emphasise the role of authoring tools:** Authoring tools are central to authoring, but they are not solving the whole authoring issue with abstraction. They accelerate the process of entering the storyworld code, including abstract data, but there exist other and possibly better ways to facilitate the creation of abstract models.
- **To see authoring as debugging:** There are a lot of similarities between authoring a story in IS and debugging a software. It is useful to use similar tools and methods as debugging for writing a story in IS.
- **To invent sketchy representations:** As video games, IS technologies often require a big effort to display only a few seconds of interaction, especially in animated 3D worlds. This is not compatible with the idea of shortening the design cycle. Some forms of 'short cut' usually enable us to have an idea of the final result, even if we have not spent time specifying the final wording, visualisation and user interface.

We consider abstraction a design technique that – as every design technique – needs training and practice with real projects. Finding the right level of abstraction needs the ability to judge what is a 'right level' in each case, which typically needs skills based on experience. Also the

way how to move between abstract and concrete representation is a matter of experience, and it will depend on the quality and features of authoring tools that are then tailored to the creative process.

2.3 Design Principle: Conditional Events

During conception of any story, such as in screenwriting, authors are aware that concretely and orderly described actions of a character are situated in a range of possibilities. Often, writers put themselves inside the minds of their characters and deliberate on their possible and plausible actions. McKee (1997) has called this writing method *"writing from the inside out"*. If done successfully, these conditions of the characters' acting situations can finally be inferred by the audience solely by experiencing the expression of actions and events, directly at the representational level. Without a narrator precisely telling us about a character's goal, we usually get an impression of it. Further, we probably know about the state of the situation including the inner states of characters, which change over time depending on events. Also of these inner states or world states, usually there is nothing in the script delivered by the screenwriter. However, *"writing from the inside out"* means that there has been at least an implicit awareness of these states, if not drafts and sketches used prior to writing.

The difference of this method to using a generative system for IS is that with generative systems, these inherent conditions for actions have to made explicit and have to be described by authors at an abstract level. Lots of discussions about authoring in IS circled around the point that authors need to *"let go"* of their will to control the plot. We perceive as a problem if authors are presented a negative goal of 'what not to build', namely a 'sequence of events', following this *"let go"* maxim. Instead, here is a positive concept of 'what to build', which is 'conditional events', also acting situations or conditions for action. Further, it is necessary to define what each action 'does (or can do) to the world'. In short, this refers to building a sort of simulation model, the kind of which depends on the IS system used. This model can consist of information about possible actions and events, and of descriptions how the world state can be described (attributes) and how it can change (rules). While the concrete actions are then automatically chosen or generated at the runtime of the interactive experience, it is a creative decision of authors to select which predicates, attributes or facts are part of the storyworld, and by which rules or plan the actions are determined. The right side in Fig. 8 illustrates the 'generative' IS design principle of writing an abstract storyworld by building a model and by rules. The implied hand of the creator illustrates at which point creation can happen in traditional storytelling (left) and IS (right). It can also be applied to the concept of the design cycle of Fig. 7.

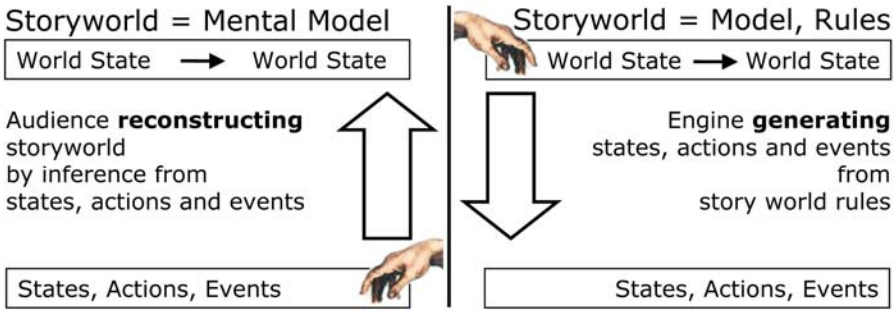

Fig. 8. Left: Creation as in traditional scriptwriting. Right: Creation with generative algorithms; the point of creation moves to abstract levels.

Several theories exist that represent actions including their conditions by a triadic structure, meaning that we have to describe three elements of an action. For example, the narratologist

Bremond (1980) described an elementary sequence of an action by 1) the possibility for action, 2) the actualisation of the action and 3) the result of the action (see Fig. 9). In Bremond's model, the agent has options to refrain from the possible action and to either succeed or fail when choosing to act.

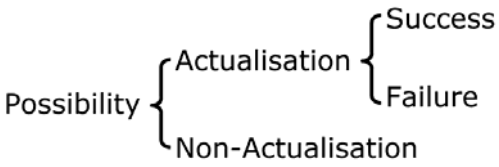

Fig. 9. Triadic form of an acting situation: Bremond's model.

Another theory of v. Wright (Herman, 2002) about the *"logic of action"* puts the concept of 'state' at the centre of an acting situation. According to this point of view, the most important aspect of action is a change in the world, respectively, the change of some *"state of affairs"*. This structure is similar to a creative principle proposed by McKee (1997) to writers, to judge every event according to what it *"does to the world"*. The three parts of an acting situation are then 1) the initial state of the world, 2) the end state after completion of the action, and 3) the state in which the world would be without the action. This model of a 'state change' as the reason for action brings it closer to computational models of state machines.

Finally, a similar triadic form exists in concepts of describing planning operators used in AI, also stemming from logical formulas for actions and states. The use of planning techniques is one prevalent method for automatic plot generation in current IS systems. When authoring a storyworld in form of a so-called planning domain, which is also an abstract description of possible events, the definition of a triad of *"pre-condition – action – post-condition"* as description of possible events *("operators")* is very common. With its typical technical formalism and terminology, it can be understood as an 'alien' concept of programming by traditional authors. However, we propose that the general concept including *pre-conditions* and *effects* of actions can be learned and adopted without programming skills to conceive the situational descriptions needed for interaction. A gap between design 'cultures' may frequently occur when having to specify more complex pre-conditions in a formalised 'math-like' language. According to Crawford (2002), who elaborated on the problem, this can only be overcome by creative partnership between engineers and artists. The artists' share is to *"endeavour to express their needs to the engineers as clearly and precisely as possible"* (Crawford, 2002).

2.3.1 Defining conditions: action-centred design vs. reaction-centred design

In different IDS systems, there are two opposed conceptual strategies how conditional events and acting situations can be defined. One is reaction-centred, the other one is action-centred.

- **Reaction-centred acting situations:** This refers to a 'stimulus-response' principle. Starting from a stimulus coming from an external source, a character reasons about the possible reactions to the stimulus. This concept is for example used in the chatbot principle that underlies the platform Scenejo (see section 3), as well as in Storytron. In the latter, the conception starts by looking at each verb as an event that can happen, then thinking about the conditions and options to react upon it. With this concept, it is harder to think of pro-active actions as part of a plan, and easier to define concrete reaction rules to events.
- **Action-centred acting situations:** This is the typical approach used in planning software, such as the Madame Bovary planner (or Emo-Emma Authoring Tool, see section 3.3). The conception starts by thinking about all actions that an agent may want to do and what their effects are on the world state, which also needs a description of facts in this world (for example, propositions, predicates, attributes etc.). Then, pre-conditions are defined that constrain the situations in which they may occur. With this

concept, the conception of interaction with a user can be left out at the beginning, but on the other hand, it is easier to think about the effect of an action for the story.

Both ways of defining acting situations are related to each other, but they afford different ways of conceptualising the interaction structure. We found that depending on the kinds of situation (managing the global flow of pro-active actions or needing to instantly react, for example when being within an argument or fight), one or the other approach is more useful.

Goals and Plans

When using a planner to accomplish automatic selection of next actions, it is necessary to define an initial state and a goal state. Goals and plans are concepts in line with traditional storytelling, and necessary even if no planning software is used, but manual scripting is accomplished. Following these premises, the characters' action selection strategies (or reaction strategies) should be defined so that there are no conflicts with goals. Whether or not automatic planning in a generative IS system is employed, these aspects require skills that define the 'art form' of storytelling. For example, the definition of simple goals that are never conflicting leads to quickly ending and boring plans with generative engines. The conditions have to be designed in a way that they contain real alternatives for action. It is a design challenge for authors to design meaningful alternatives for acting situations, which must be well-grounded in their interrelation with represented character attributes, goals and plans. Again, these have to be iteratively play-tested in a design cycle between abstract and concrete levels.

Issues of Visualisation

The visualisation of emergent and multiply variable processes is difficult. Typical graph and plan structure visualisations offer affordances for linear or hierarchical branching thinking; therefore, strictly visual thinking can hinder highly-interactive storytelling. Authors have to find ways of 'dynamic visualisations', for example by inventing sorts of paper-prototyping, such as with a card game (see section 3). It is recommendable to use several forms of visualisations to illustrate different points of view to the same elements in different contexts. In this context, state charts are typical visualisations of *reactive* systems. If used as a strict and only formalism, they can lead to linear and branching thinking – in other words, support creating a "paper flower" model as in Fig. 4. Pizzi and Cavazza (2008) have demonstrated a progressive plan visualisation within their authoring tool for the Madame Bovary system. It showed that at each step in the ongoing story, the planning of a possibility space leads to a different, updated state chart, taking the actual position as the starting position. Without such a tool, it is hard to imagine how planning and re-planning based on pre- and post-conditions of operators can open up new possibilities of further connections. The graph cannot be drawn manually at the authoring stage. In our card game presented in section 3, the world state changes dynamically and almost emergently, therefore visual representations of it can only appear as volatile snapshots between the operations. Such a card game paper prototype therefore can support conceptual understanding.

2.4 Further Design Principles

2.4.1 Including the user from the start of modelling

Storyworld modelling has to include a role (or possible roles) of the 'user' ('player', 'performer', 'participant' etc.) from the start. Creating a model starting from a given story without a user role and objective makes it difficult to insert the user afterwards. This is an experience made in all authoring examples mentioned in this report, and in the exercise of literature adaptation of a Hemingway short story to an interactive version (using the Storytron philosophy as a modelling approach), reported in (Spierling and Hoffmann, 2010). The interactive model needs to be balanced during iterations of redesign, taking into account all possible actions, and all possible states of characters, therefore including the user. The simulation/storyworld model, the implied plot (as a high-handed goal of the simulation) and user/player objectives have to be reconciled.

2.4.2 Externalising the internal

This creative principle from screenwriting enunciated by Howard and Mabley (1993) and reported in (IRIS-WP3, 2009) remains valid for Interactive Storytelling. It has been directly experienced more when it was missing than when it has been solved well. If inner thoughts of characters are important for the 'gist' of the story (for example a plan calculated by a generative engine), then designers have to find ways to reveal details of the plan or inner states and processes to engage the audience (or the participating user). This affords complex representation, one of the major research challenges if done automatically. Beyond representation as the 'shape' and media style of given events, it also has to do with creating extra events that express these 'internal' processes and states.

2.4.3 Interaction Design

Interaction design, which is a complementary principle to 'Externalising the Internal', is important for the user experience of agency. It is essential to design action conditions for users (affordances) in a way that they are easily grasped, and give users immediate feedback on their actions. It is important to consider that in agent-based and story-based interactions, the results of one's action are potentially delayed. This does not mean that immediate feedback is obsolete; it rather means that it may include techniques of foreshadowing or flashback. Therefore, the conception of a complete interactive storyworld includes the interaction design, which is also a part of authoring.

2.4.4 Debugging as authoring (iterating)

Proposed by several scholars in computer science (Pizzi and Cavazza, 2008) (Swartjes and Theune, 2009) and confirmed by the experiences of our research, 'debugging' in the context of IS creation is more than repairing software. It refers to the constant necessity to iterate changes in the model and test the result, due to the unlikeliness of being able to guess optimised rules in a simulation model before running a test. Swartjes referred to it as co-creation, in the case that a 'bug' turns out as a 'feature', contributing an unexpected suggestion to the storyworld. We have circumscribed it above with the necessary processes of linearisation and delinearisation (see Fig. 7). This principle also points to necessary tools to inspect the outcome without having to experience it over and over, which is a production issue.

3. Educational Material

This chapter – together with the documents and software links in the appendix – contains explanations and presentations to be used as educational material for authoring and creation in the field of Interactive Digital Storytelling. The following general topics are covered:

- Modelling content in conversational storytelling (3.1)
- Abstract story modelling with tasks, goals and obstacles (3.2)
- Modelling storyworlds making use of planning (3.3)
- Modelling storyworlds with state machines (3.4)

An excerpt of the offered material can be found in the appendix, together with links to full versions that are subsequently updated.

The rationale for most of the material is to first provide a theoretical basis that can be generalised in a way to several different systems approaches, and then to give instructions for interactive practical conception examples. As far as the use of concrete authoring tools is involved, the material covers the creation of "hello world"-like minimal storyworlds specific to the used systems as tutorials for the first steps, as well as the tools and tutorial example files. Further, examples of more complex storyworlds are available to be experienced.

This material has been tested and discussed in several IRIS-internal workshops, and parts of it have been publicly presented and evaluated at the ICIDS 2010 conference[2]. It is further used and upgraded during international public workshops in year 2011.

3.1 Modelling Content in Conversational Storytelling

This section covers general design steps and a concrete example to model interactive storyworlds based on verbal conversations. For a general theoretical introduction, it is intended to impart basic comprehension of the issues in modelling natural language based conversations in Interactive Storytelling. The practical exercises can also serve to deepen the understanding of conditional thinking, which is fundamental to IS, and abstraction in the sense of abstraction as generalisation (see section 2.2.1). Exercises are provided using the Scenejo system.

The following external material is also available:

- Presentation slides for theoretical introductions (appendix A.5 and online[3])
- Software *Scenejo*[4] including related literature and the *Scenejo Authoring Tool*[5]
- Tutorials for *Scenejo* and the *Scenejo Authoring Tool* (appendix B.1 & B.2, full versions online[6])

3.1.1 Abstraction of conversations

In Interactive Storytelling, there is a great motivation to use verbal dialogues (audible speech or typed text) for directly interacting with the storyworld characters. For example, Façade (Mateas and Stern, 2005a) as the first working prototype of Interactive Digital Storytelling uses conversation as the central means of interaction and of presenting content, in a highly-

2 Tutorial and Workshop at ICIDS 2010: http://icids2010.interactive-storytelling.de/

3 http://iris.interactive-storytelling.de/

4 Scenejo: http://scenejo.org/

5 Scenejo Authoring Tool: http://scenejoauthoringtool.origo.ethz.ch/

6 Complete link lists available at http://iris.interactive-storytelling.de/

interactive experience. However, with regard to generative conversational content and appropriate adaptability to user utterances, there are many difficulties involved, especially in terms of coherence of continuing conversations with frequent turn-taking. For example, although Façade is a well-designed piece, the interaction includes situations in which the characters are unable to react appropriately to user input. Especially the understanding of user utterances is a technical challenge. Story creators who want to include this kind of interaction have to be aware of limitations and necessary efforts in current state of the art dialogue systems, in order to work with appropriate (meaning, lowered) expectations and to find pragmatic solutions.

Scenejo is a simple platform that enables writers in an easy way to experience the creation of interactive dialogues, following a similar interaction paradigm as in Façade (user interaction with more than one actor). It is based on chatbots controlled by a simple dialogue manager. In contrast to Façade, it is still a bit less sophisticated in the analysis of the meaning of user utterances, because the system is limited to comparisons of text input with a text pattern database prepared by authors. Just like Façade, it cannot generate sentences of the characters; they have to be hand-written. On the other hand, we argue that it is well suited for introductory exercises into the topic, because of its accessibility. We can start writing dialogues in a linear way and introduce more abstract and interactive structures step-by-step.

For a general impression, Fig. 10 presents only a minimal overview of the specialisations involved in building a fully-automated speech-interaction system, to then focus on author-relevant aspects of the whole task. Currently available automatic systems work well in constrained contexts of solving well-structured tasks, such as in a booking system. Beyond that, they require a huge development effort of several sub-disciplines based in AI and Linguistics. We should also note that beyond verbal language, for a convincing result in storytelling there are representational non-verbal aspects to be solved, which are omitted here.

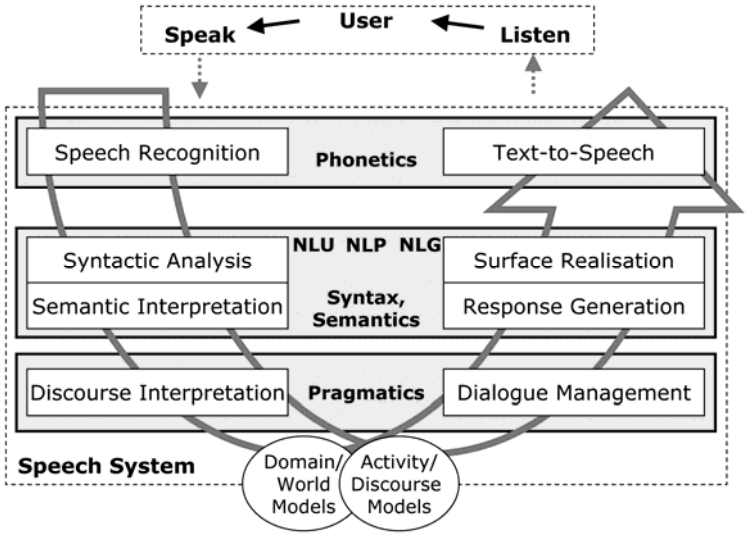

Fig. 10. General model of a spoken dialogue system

The grey blocks in Fig. 10 summarise complex steps of automation, with input processing on the left side and output generation on the right side. However, in systems for interactive storytelling, it is imaginable that only partial automation is implemented, and manual solutions are chosen to work around technical difficulties or unpleasant artefacts. E.g. the 'phonetics' layer is overcome in Façade by offering only typed input and recorded human sentences as output. Scenejo also uses typed text as input and TTS as output. However, synthetic voices often do not convince the audience – they are nevertheless worthwhile during prototyping.

Syntactic analysis determines the utterance structure, and semantic interpretation leads to the understanding of a word's or a sentence's meaning. Response generation composes

meaningful dialogue acts ('what to say') that are then turned into direct wording ('how to say it') by a surface text realisation based on correct grammar, taking into account styles and diction. This level contains technical challenges that are currently tackled in several sub-fields of natural language processing (NLP) with the two aspects of understanding (NLU) and generation (NLG). Pragmatics is the study of language use in practice, its goal-directedness and its implied actions. Discourse analysis is the study of conversational structure. Discourse Interpretation analyses what the user intends by taking into account the discourse context. In Dialogue Management, goals and plans are considered to either respond adequately or to obtain the initiative in the dialogue.

Abstraction of dialogues in general is therefore a complex topic and depends on the level at which we introduce automatic generation. For example, at the syntactic level, we would need to write grammar rules and abandon the written style of a storyteller. This is not exactly what we do in Scenejo. We are mainly concerned about the pragmatic layer, a similar approach like in Façade. While most of our utterances are hand-written (full or sometimes partial) sentences, we see dialogue "as actions" – a crucial pragmatic concept described by Speech Act Theory. We add the abstract description of a dialogue act to each utterance. Further, we raise awareness of discourse structures, such as so-called adjacency pairs (e.g., question/answer pairs) and other models of basic dialogue chunks. This enables us to break down a long linear thread of conversation to pieces of smaller size, which we then can manage to rearrange depending on rules. Thus, we increase possible variability in response to interaction, therefore interactivity.

Dialogue as Action: Speech Acts

For conceptual models of storytelling, seeing verbal utterances as actions is a crucial concept. The concept of Speech Acts has first been introduced by Austin (1962) and has been further developed and systematised by Searle (1969). Austin identified a variety of different "acts" one can "perform" during speaking, categorised into three aspects (all can be present within one utterance).

- **Locutionary act:** The simple act of producing an utterance, either by saying something, or even by a "phatic" act (that has no further 'illocutionary force').
- **Illocutionary act:** An intented "performative". This can be a statement or a change of dialogue context, for example, a question, a promise etc. It refers to the intended act of the speaker.
- **Perlocutionary act:** The actual act of "doing things with words", the effect on the relationship to or the actual effect on the hearer. This act not only depends on the intention of the speaker, but also on the context and the hearer.

For example, as shown in Fig. 11, an intended illocutionary command can be expressed by different locutionary acts, and can result in different perlocutionary acts. The expressed command can have different effects on the hearer, depending on whether the hearer follows the command or how he/she takes it. The success in performing a perlocutionary act depends on a complex state of the situation, including the roles and presuppositions of all involved.

Locutionary Act (possibilities)	**Illocutionary Act**	**Perlocutionary Act (possibilities)**
• "Jump into the water!" • "Come on coward, jump!!!" • No locutionary act (using whistle and hand gestures)	Command to jump into the water.	• Persuade • Irritate • Amuse • Annoy

Fig. 11. Correlation of Locutionary, Illocutionary, and Perlocutionary Act.

As a general rationale, we find that it should not be necessary to have a strong background in theory first, before authoring of dialogues can be commenced. However, we propose that with increasing complexity and interactivity of an intended conversation to be authored, some background in discourse theory can help to find models that are more flexible. It is natural to begin with a linear impression of a whole argument, then trying to informally abstract it. An example is given in Fig. 12., taken from draft material of the "Killer Phrase Game" running with Scenejo, described in (Spierling, 2008).

	Dialogue from preliminary script	**Draft dialogue acts**
Mrs PRO	I want to clarify something first: the fact that cars don't travel at walking speed on your street has nothing to do with the airport expansion. That is a phenomenon that you can find anywhere, airport or no airport. But think of the effect on employment in our area! New jobs will also be created!	Argument Objection, Argument: 'New Jobs'
Mister CON	Now, don't give me that! The cars come, because of the airport! But that's no skin off your noses, because you live out in the countryside, where you don't have to see or hear airplanes.	'Interjection', Killer Phrase: 'Reproach Careless'
USER	Mister CON, would you please stay to the facts.	Moderation
Mrs PRO	But, we're not going to arrive at any solution this way. Let's work on a concept together. We don't mean you any harm.	Moderation, 'Promise'
Mister CON	Yes, but you have to say that. You only care about money! You make promises you have no intention of keeping … we've seen that often enough.	Killer Phrases: 'Reproach Unfair', 'Doomsaying'

Fig. 12. Excerpt from a preliminary script of the Killer Phrase Game. As a first step of abstraction, dialogue act 'labels' were added.

This step of labelling utterances according to actions performed is a first step of abstraction, in the sense of a generalisation. As a result, we find that the performed actions can have several ways to be expressed (locutionary acts), which later can be chosen alternatively (see Fig. 13). It is a way to increase the potential variability of the conversation.

ARGUMENT PRO 'NEW JOBS'	**KILLER PHRASE 'REPROACH IGNORANCE'**	**KILLER PHRASE 'REPROACH NO CLUE'**
New jobs will also be created!	No one has ever cared about us.	Just come to my house on any given afternoon!
The economy will improve, which can also mean a job for you!	We residents don't figure into your plans.	You're not even from here … who are you to judge?
Think of the effect on employment in our area!	That's no skin off your noses, because you live out in the countryside, where you don't have to see or hear airplanes.	You have no idea what daily life is like for us!

Fig. 13. Alternative concrete utterances to express three different abstract dialogue acts

Using this abstraction level for the conceptualisation of dialogue acts has several advantages:

- For **'bot-to-bot' statements** or general conversations, a design level is introduced that facilitates
    - better overview in the conception phase,
    - conception of the effects of dialogic actions (speech as action),

- increasing variability by alternative utterances (to be chosen randomly or on conditions by the engine), and
- the possibility of re-editing final bot utterances to adapt to certain character style or language.

- For **'user-to-bot' statements**, an abstraction level is introduced that allows aggregation of many possible concrete utterances to one 'abstract input', representing a user dialogue act. This is a strategy to reduce the number of possible user dialogue acts, thereby alleviating the burden of having to react to many different possibilities. Similar input aggregation strategies have also been employed in Façade (called "many-to-few mapping") with the utilisation of 'discourse acts' (Mateas and Stern, 2004) and in FearNot! with 'speech acts' (Louchart et al., 2004).

3.1.2 Basic Scenejo concepts

The range of meaningful interactive stories that can be created with Scenejo is constrained – by the interaction principle similar to Façade – to situations, in which the participation in or the interruption of ongoing conversations between characters is part of the story's goal. This has to be considered when creating concepts for this system. The available system (see appendices) connects an animated 3D talking head to a chatbot, audibly speaking via a text-to-speech (TTS) system (Fig. 14, left), while the user types text as input. Fig. 14 (right) sketches the central communication principle underlying the Scenejo conversations. A dialogue manager, called the 'Dramatic Advisor' (DA), acts as a broker of turn-based textual exchanges at a central 'Meeting Point'. Utterances of bots and users are treated in the same way.

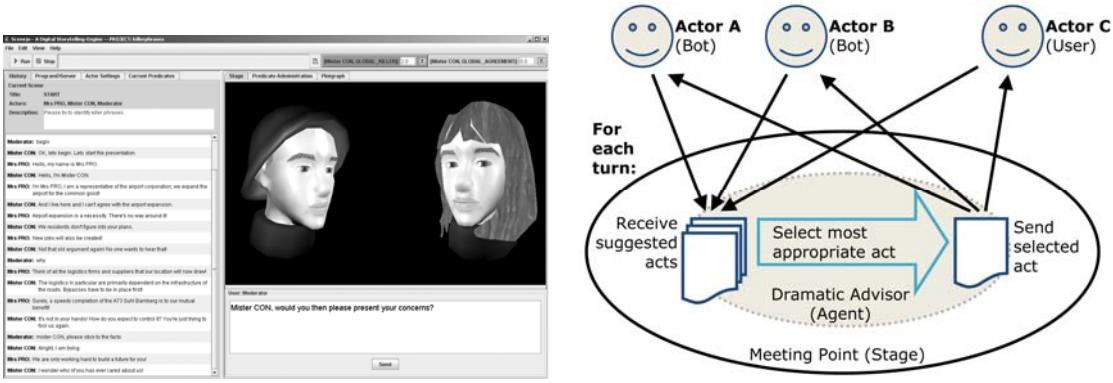

Fig. 14. Basic communication principle of Scenejo. The 'Dramatic Adviser' selects actions from incoming verbal utterances, which are provided by chatbot responses and/or user input.

Each chatbot works according to a stimulus-response principle. Assuming a dialogue made up of question/answer pairs, it generates one answer to a question that is required as an input. The finding of a single response, performed by each bot program at the periphery of the conversation at each turn, is a search process within a database of possible matches of word input patterns and associated templates defining the response. This co-called knowledge base has to be authored (or acquired otherwise) before. It is written in an XML dialect (SCAIML) based upon AIML. It is not absolutely necessary that authors are familiar with AIML, but an advantage in the debugging of complex storyworlds.

A special Scenejo construct is the "stimulus-response element" SRE. It embraces a bot utterance as a response, together with a stimulus, abstract dialogue act and potential conditions defining possible branching connections. Thus, one bot utterance is defined by creating an SRE. It has to be noted that all utterances of bots have to be conceptually defined as responses to a stimulus, following the "reaction-centred design" described in section 2.3.1. There is no proactive or plan-based behaviour. However, the Dramatic Adviser (DA) can deliver a 'prompt' to trigger a start of a certain dialogue, for example in a new scene (see Fig. 15). A linear

dialogue between two bots has to be arranged like in Fig. 15, by intertwining inputs and outputs providing prompts to react upon.

Fig. 15. Left: Schematic rendering of a bot utterance, represented by an SRE. Right: Dialogue between two bots, intertwining input and output.

Users can interrupt between every single turn, in which case a user utterance is treated the same way as an input from another bot. In the case of the long script of the Killer Phrase conversation in Fig. 12, the dialogue of course can be structured like the one above in Fig. 15 (right). However, that would mean that there are only few reaction possibilities to users. When designing for interactive storytelling, we want to react to users and then go on with something suitable after this reaction. Therefore, we need some dependencies on conditions. There are two types of explicit branching structures to be used in the stimulus-response-based dialogue:

- Authors can provide several possible cases of responses to one stimulus, based on evaluating current states of variable predicates of the bot (Fig. 16, left). This is useful in cases of recurring stimuli in different conversational contexts, either to avoid repetitions or to advance dialogue states.
- Another possibility is to anticipate several optional input patterns (Fig 16, right). This method is important in order to be able to react differently to diverse inputs from varying sources (user or bot), such as in a typical 'quiz' situation.

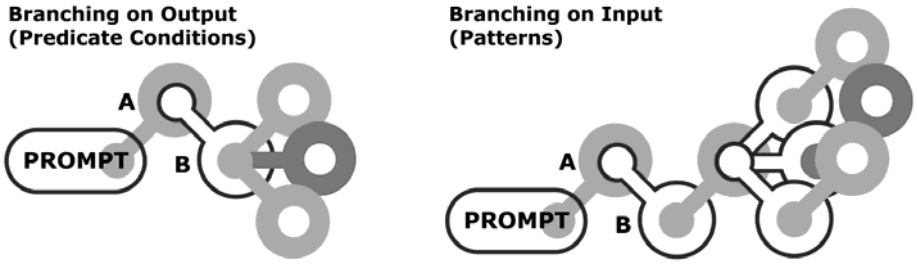

Fig. 16 Branching possibilities. Left: 'Bot A' has 3 different output options at one input pattern, based on checking further pre-conditions. Right: 'Bot A' expects 3 different possible inputs to continue differently from there.

Further dialogue abstraction

It is natural that a question leads to an answer in some way, before the addressee goes on with another initiative utterance. This can lead to chunks of dialogue that – between humans in a normal conversation – are naturally hard to be interrupted in the sense that someone begins a completely new topic, at least without apologising. Theories on dialogue chunks include concepts of dialogue moves and dialogue games, which can be helpful to understand the philosophy behind complex human turn-taking. For an overview, see (Dooley and Levinsohn, 2001). Further, we want to let the bots possibly vary their contributions according to emotional states that possibly can be influenced by the user. It is important that not a fixed branched conversation is written, but a variable model that can adapt the flow of concrete utterances to the change of states.

In the example of the Killer Phrase Game, such a complex structure that follows conversational rules can look like Fig. 17. This dialogue thread contains several options of variations

depending on the state of the argument, which is here the stress level in the debate. The argument is grouped into sub-chunks as pair units of “initiating” and “resolving moves”. Beyond questions and answers, such a pair can also consist of proposal/response or remark/evaluation (Longacre, 1996). In Fig. 17, the debate can be constructive and lead to a quick clarification, but it can also lead to an escalation if all the “countering moves” and the killer phrases are included. Whether or not this happens depends on conditions to be checked during runtime.

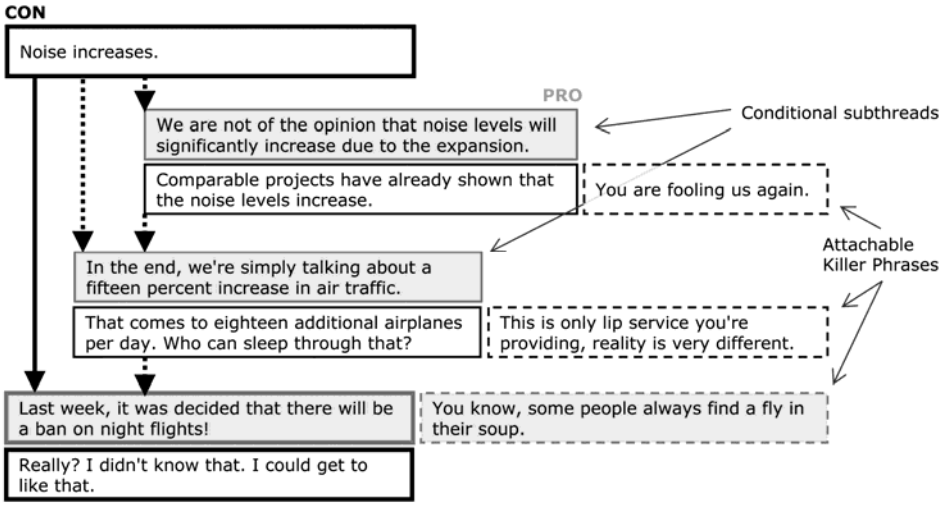

Fig. 17. Complex dialogue thread with options of sub-chunks and attachable elements. Only the three acts with bold outlines are mandatory.

In order to preserve the coherence in the dialogue, it is essential that no elements are taken out that would lead to an isolated remark or an evaluation that does not fit another remark. An even higher abstraction of the dialogue thread can be read like the following (where IM = initiating move, RM = resolving move, CM = countering move, KP = killer phrase, black and grey colour denote different actors):

1. **(IM, Remark)**
2. **(RM, Evaluation + CounterRemark)**
3. **(CM/IM, Evaluation + CounterRemark) (**optional: **KP)**
4. **(RM, Evaluation + CounterRemark)**
5. **(RM, IM, Evaluation + CounterRemark) (**optional: **KP)**
6. **(CM/IM, Proposal)** (optional: KP)
7. **(RM, Response)**

Concluding, structures like this have the potential to be reused with variable utterance wordings and some conditional structural variance, leading to less repetitive content, although the final utterances are hand-crafted. In conversations, the conditions for acting are often based on reaction rules rather than a plan, which only occurs when having the initiative in-between resolved dialogue chunks. This conceptual philosophy can be experienced and trained only in longer-lasting practical conception exercises, such as in a seminar exceeding a one-day tutorial, for example a summer school. It is possibly hard to do this modelling exercise directly in the authoring tools of Scenejo. Therefore, it is recommended to use paper prototypes, such as a stack of cards, drawings and fill-in tables, to test the variable flow of potential conversations before implementation. It combines the creative principles of abstraction and of thinking in conditional acting situations, combined with some theoretical concepts of discourse grammar. It is here meant to give a perspective on elaborate conversational modelling. In the following, less complex examples are used to explain the first steps.

3.1.3 Practical modelling exercise with Scenejo

Outline of a minimal storyworld example

The example story used in the step-by-step Scenejo tutorial is a little conversation between Adam and Eve who surprisingly find themselves in paradise at the beginning. They start with greeting each other and introducing themselves. Then Eve asks Adam how he feels and he feels OK. But Eve does not stop asking again and again, making Adam feel worse every turn, until his mood drops and he wishes Eve to hell. In hell both blame each other for being there. However, if the user interacts and offers Adam an apple before he feels too bad, this makes him feel much better, until he starts to feel fantastic. That helps neither, because too many apples let our two actors be kicked out off paradise into the cruel world where they again start to blame each other. The intended end user experience will be a short funny conversation, which can be influenced (by offering apples to Adam), regarding the scene in which the story ends.

Scenejo modelling of a basic conversation

For implementation, this conversation has to be turned into a formal description of 'SRE's, which are then coded in SCAIML, the Scenejo AIML dialect – in order to be finally used with Scenejo. A first step to this approach is creating a spreadsheet for single SREs ("stimulus-response elements") and to gradually expand it until all needed data is present and it can be entered into SCAIML directly or - more easily - with an authoring tool like the "Scenejo Authoring Tool".

The following example demonstrates the creation of Adam's reaction to Eve's question regarding his current mood. The created SRE will have the name "Adam Mood Status". The tutorial in the next section will expand it further, adding more conditions to it.

There are two basic ways to start; one is from the abstract side, the other from the concrete side, here named "final", meaning the final wording level. Normally it is easier for an author to think at a concrete level in the beginning and later transpose it to the abstract one, but both is possible and depends on preferences of the creators. Fig. 18 shows both possibilities.

	Abstract
Input	Mood Question
Output	Mood Status

	Final
Input	HOW DO YOU FEEL
Output	*I'm OK.*

Fig. 18. Formal description of an SRE - step 1.

By abstracting the question "HOW DO YOU FEEL"[7] to the abstract input "Mood Question", we can now think about other utterances expressing a mood question, like shown in Fig. 19.

	Abstract	Final
Input	Mood Question	HOW DO YOU FEEL
		HOW ARE YOU
		* FEEL
Output	Mood Status	*I'm OK.*

Fig. 19. Formal description of an SRE - step 2.

[7] Following the AIML specification, an input pattern has to be written in capital letters and without special characters beside the wildcards "*" and "_".

In the next step we want to explore giving different responses to a given stimulus. Therefore we divide our answer in a good and a bad one. Fig. 20 shows the two abstract outputs and their final wording. “Output 2” also shows how to add more variability to the wording levels by allowing two different utterances that both express a bad mood. They will be given randomly.

	Abstract	Final
Input	Mood Question	HOW DO YOU FEEL
		HOW ARE YOU
		* FEEL
Output 1	Mood Status Good	*I’m OK.*
Output 2	Mood Status Bad	*I feel bad.*
		I don’t feel good.

Fig. 20. Formal description of an SRE - step 3.

At this point it is not yet clear when these two answers will be given, so they have to be linked to Adams mood level by conditions. Fig. 21 shows the conditions for our two abstract outputs. Now the “Output 1” will only be given if the MOOD value is bigger or equals 2 and “Output 2” only if the MOOD value is smaller or equals 1.

	Condition	Abstract	Final
Input		Mood Question	HOW DO YOU FEEL
			HOW ARE YOU
			* FEEL
Output 1	MOOD >= 2	Mood Status Good	*I’m OK.*
Output 2	MOOD <= 1	Mood Status Bad	*I feel bad.*
			I don’t feel good.

Fig. 21. Formal description of an SRE - step 4.

The next step adds operations that change the mood values caused by the acts. Because we want Adam to be annoyed by being asked about his mood, we lower his MOOD value by 1, every time he has to answer. This is shown in Fig. 22.

	Condition	Abstract	Final	Operation
Input		Mood Question	HOW DO YOU FEEL	
			HOW ARE YOU	
			* FEEL	
Output 1	MOOD >= 2	Mood Status Good	*I’m OK.*	MOOD - 1
Output 2	MOOD <= 1	Mood Status Bad	*I feel bad.*	MOOD - 1
			I don’t feel good.	

Fig. 22. Formal description of an SRE - step 5.

The spreadsheet in Fig. 22 contains all necessary data to create the final SCAIML code to be interpreted by Scenejo. A simplified example of that code can be seen in Fig. 23.

```
…
<sre name="Adam Mood Status" …>
…
   <abstractAct name="Mood Question" …>
   <patterns>
      <pattern>HOW DO YOU FEEL</pattern>
      <pattern>HOW ARE YOU</pattern>
      <pattern>* FEEL</pattern>
   </patterns>
   <action>
      <if>
         <condition>
            <predicate-condition expression=">=2"
                  predicateName="MOOD"/>
         </condition>
         <action>
            <abstract-output name="Mood Status Good" …>
               <xaiml:random>
                  <xaiml:li>I'm OK</xaiml:li>
               </xaiml:random>
               <operation predicateName="MOOD" operator="-"
                     value="1"/>
            </abstract-output>
         </action>
      </if>
      <elseif>
         <condition>
            <predicate-condition expression="<=1"
                  predicateName="MOOD"/>
         </condition>
         <action>
            <abstract-output name="Mood Status Bad" …>
               <xaiml:random>
                  <xaiml:li>I feel bad.</xaiml:li>
                  <xaiml:li>I don't feel good.</xaiml:li>
               </xaiml:random>
               <operation predicateName="MOOD" operator="-"
                     value="1"/>
            </abstract-output>
         </action>
      </elseif>
   </action>
</sre>
…
```

Fig. 23. Formal description of an SRE in SCAIML.

This example demonstrated the creation of a single SRE. For a complete conversation, a lot of them are needed. Due to the principle of intertwining stimulus and response of different bots with each other, the corresponding side of Eve has to be created in the same way, but now the former final outputs "I'm OK.", "I feel bad." and "I don't feel good." should be her final inputs. A difficult task is the conception of appropriate responses to user input patterns. However, technically, this is written in the same way as an input from a bot.

It is recommended to first use tools outside the Scenejo authoring tools for designing the structure, for example a spreadsheet. Single SREs can be arranged in columns corresponding to the bots and the user, linking the outputs of one bot with the fitting inputs of the partner bot.

3.1.4 Step by step tutorial using the Scenejo Authoring Tool

The content code for the runtime engine Scenejo is created with the "Scenejo Authoring Tool". A detailed introduction to its basic use can be found in appendix B.1. We suggest beginners to first consult it before going on with the following scenario. In appendix B.2, a version of this tutorial can be found that has more illustrations and is intended to be used in courses. The full versions of the tutorials and the links to the project files of the single steps of this tutorial are available in our link collection[8].

Step 1: "Hello World" – the minimal story

The basics for a minimal scenario with Scenejo:

- 2 actors: Adam, Eve
- 1 scene: "Paradise"
- 1 SRE for each actor with abstract input (AI), final input (FI), abstract output (AO) and final output (FO), in which both greet and introduce themselves

At first the two actors Adam and Eve have to be created. Adam can keep the default actor settings (gender: male, avatar: "Alex"), but Eve will get the gender female and the avatar "Alexa". Such an avatar is a 3d representation of the actor as a talking head. Default heads are part of the Scenejo installation; these are used within the tutorials.

Then the scene "Paradise" is created and we add the two actors to this scene.

After that we create the first SRE for Adam, named "Adam Greeting", with its inputs: the abstract input "Adam Greeting AI" and the final input "*". Then the outputs are needed: the abstract output "Adam Greeting AO" and the final output "Hello there! I'm Adam." This means, whatever Adam is hearing (because of the wildcard "*"), he will say "Hello there! I'm Adam."

Note that the naming of abstract inputs or outputs (as well as SREs) is up to the authors of the code. Depending on how authors prefer to structure their work, they can either use abstract names with their own coding, or illustrative names, which can be used multiple times. Finally, when the implemented storyworld gains in size, it will be crucial to find dialogue pieces in long lists by their names. The final inputs and outputs can be redesigned later without changing the abstract structure, for example when translating the conversation to a different language.

Next, Eve will receive her SRE "Eve Greeting" and the corresponding inputs: the abstract input "Eve Greeting AI" and the final input "HELLO THERE *". Then she also needs outputs: the abstract Output "Eve Greeting AO" and the final outputs "Hi! My name is Eve.", "Hello! I'm Eve." and "Hi there, I'm Eve!" This means that each utterance beginning with "HELLO THERE" will be answered by "Hi! My name is Eve." or "Hello! I'm Eve." or "Hi there, I'm Eve!" (Fig. 24).

When we now run Scenejo with the created story (file: Tutorial_Adam_&_Eve_01.sce.v06) we experience an endless loop, in which both actors greet and introduce themselves (Fig. 25).

[8] http://scenejoauthoringtool.origo.ethz.ch/wiki/link_collection

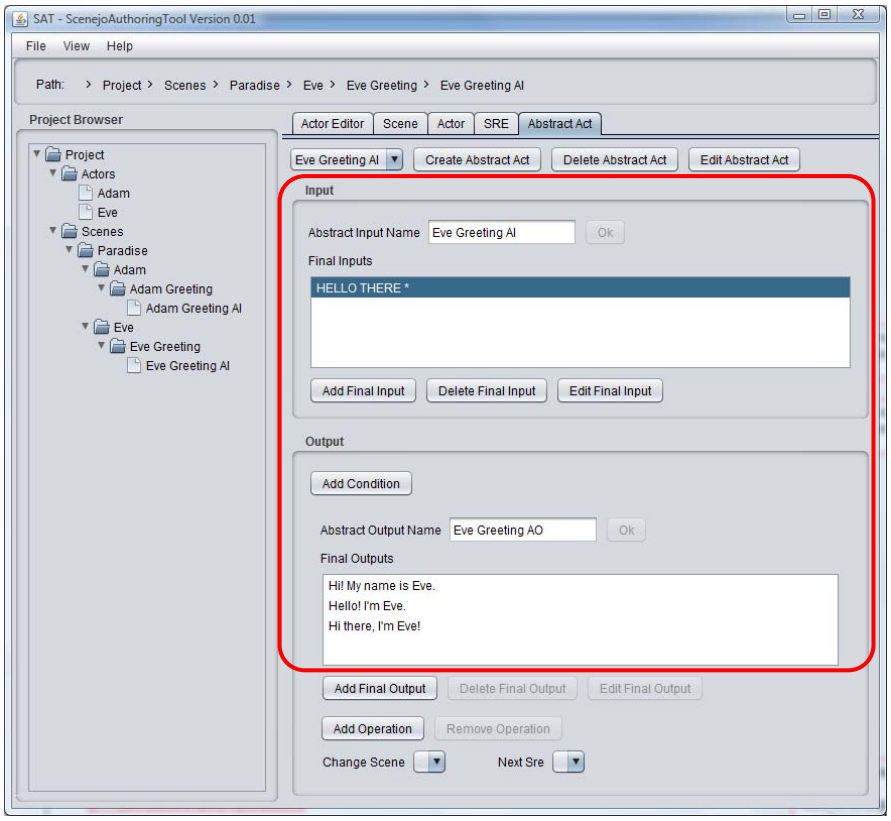

Fig. 24. Scenejo Authoring Tool - speech act creation

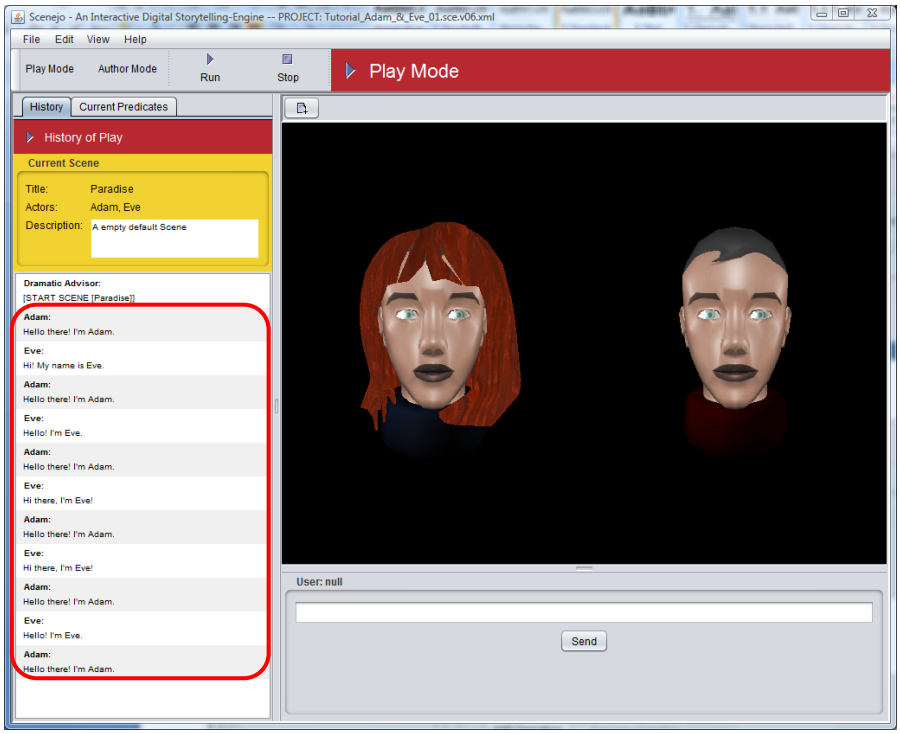

Fig. 25. Scenejo - "Hello World" example

Step 2: “More Dialogue” – adding more SREs

Three more SREs are added:

- 1 SRE for Adam in which he reacts to Eve’s greeting
- 1 SRE for Eve in which she asks how Adam feels
- 1 SRE for Adam in which he answers he feels OK

At first, we want to end the loop. So we replace Adam’s final input “*” with “START SCENE PARADISE”. “START SCENE” is a system word (command) that gets triggered by the start of a scene.

That way we ensure that Adam starts talking (file: Tutorial_Adam_&_Eve_02.sce.v06.xml).

In the next step, we want Adam react to Eve’s greeting by adding a second SRE for him, called “Adam Greeting Reaction”. We create the AI “Adam Greeting Reaction AI”, the FI “* EVE”, the AO “Adam Greeting Reaction AO” and the FO “Eve is a nice name and this place is nice, too. I think I will call it paradise.” (file: Tutorial_Adam_&_Eve_03.sce.v06.xml).

Now Eve shall ask him how he feels and Adam shall answer that he is OK. Therefore we add the SRE “Eve Mood Question” to Eve and create the AI “Eve Mood Question AI”, the FI “EVE IS A NICE NAME *”, the AO “Eve Mood Question AO” and the FO “I like paradise and I like you. How do you feel now, Adam?” (file: Tutorial_Adam_&_Eve_04.sce.v06.xml).

Finally Adam shall answer and we have to add the SRE “Adam Mood Status” for him with the AI “Adam Mood Status AI”, the FI “* FEEL”, the AO “Adam Mood Status AO” and the FO “I’m OK.” (file: Tutorial_Adam_&_Eve_05.sce.v06.xml).

The result in Scenejo looks like this (Fig. 26):

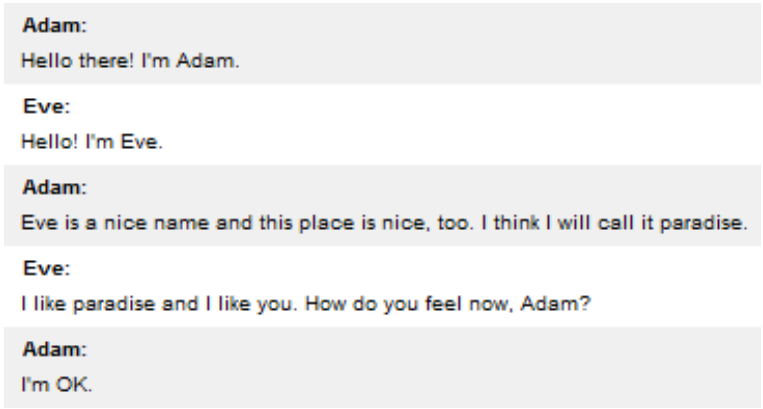

Fig. 26. Scenejo - “More Dialogue” example

Step 3: “Conditional Thinking” – adding conditions and operations

Some more things have to be added:

- 1 predicate “MOOD”
- 1 condition for Adam to change his output according to his “MOOD”
- 6 more outputs for the “Adam Mood Status” SRE
- 7 operations to lower Adam’s mood with every answer
- 1 SRE for Eve to keep asking him how he feels

At first we add the predicate “MOOD” for Adam, which is of the type “Integer” and has the initial value 2.

Then we create a condition in the SRE “Adam Mood Status” in the output area for the AI “Adam Mood Status AI”, so that “I’m OK.” can only occur if the value for “MOOD” is exactly 2 (Fig. 27, file: Tutorial_Adam_&_Eve_06.sce.v06.xml).

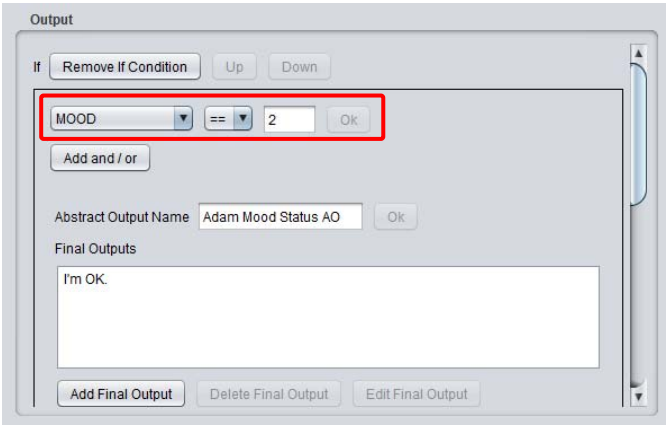

Fig. 27. Scenejo Authoring Tool - condition

A short test with Scenejo shows that nothing has changed in comparison to the previous example, because the “MOOD” is always 2, leading to the same utterance as in the last step. We need alternatives, so we add six more conditions to the SRE “Adam Mood Status”, one for each intended mood level. For each condition, we add a different output with the following dependencies/values (Fig. 28).

“MOOD”	Abstract Output	Final Output
<=0	Adam Mood Status AO 1	I feel very bad.
==1	Adam Mood Status AO 2	I feel bad.
==2	Adam Mood Status AO 3	I’m OK.
==3	Adam Mood Status AO 4	I’m quite good.
==4	Adam Mood Status AO 5	I’m good.
==5	Adam Mood Status AO 6	I feel very good.
>=6	Adam Mood Status AO 7	I feel fantastic!

Fig. 28. Output depending on the “MOOD” value

Still the mood value is always the same. Therefore, we add an operation to each output. For each output we reduce the mood by 1, resulting in the operation “MOOD - 1” (Fig. 29).

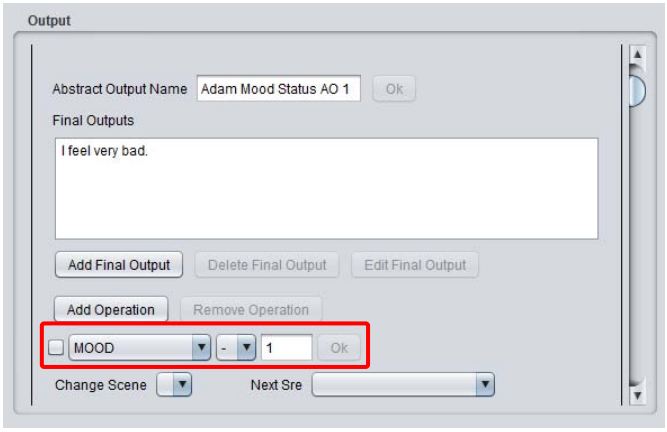

Fig. 29. Scenejo Authoring Tool - operation

We want to let Adam constantly talk about his changing mood, in order to enable the experience of the changing value for this tutorial test case. Therefore we create for Eve another SRE called “Eve Annoying Mood Question” with the AI “Eve Annoying Mood Question AI”, the FIs “I FEEL *” and “IM *”, the AO “Eve Annoying Mood Question AO” and the FO “I don’t believe you, how do you really feel?”

Playback of the result in Scenejo shows that Adam gets asked again and again and his mood drops each time until he feels very bad (Fig. 30, file: Tutorial_Adam_&_Eve_07.sce.v06.xml).

Adam:
Hello there! I'm Adam.

Eve:
Hi! My name is Eve.

Adam:
Eve is a nice name and this place is nice, too. I think I will call it paradise.

Eve:
I like paradise and I like you. How do you feel now, Adam?

Adam:
I'm OK.

Eve:
I don't believe you, how do you really feel?

Adam:
I feel bad.

Eve:
I don't believe you, how do you really feel?

Adam:
I feel very bad.

Fig. 30. Scenejo - “Conditional Thinking” example

Step 4: “Changing Scenes” – adding more scenes and switch between them

New scenes will be added now:

- 2 scenes “Hell”, “Cruel World” (the last one we need later)
- 1 command to change the scene
- 1 SRE for each actor in the scene “Hell” to blame each other

At first we create the two scenes “Hell” and “Cruel World” and add each actor to both of them. After that we connect both new scenes with the scene “Paradise”, indicating the possibility of switching from Paradise toward them. In Adam’s SRE “Adam Mood Status”, we can now use the “change Scene” command. We choose “Hell” as the chosen target scene to be jumped to in case the AO “Adam Mood Status AO 1” is played. To make it more interesting we also change the FO from “I feel very bad.” to “Stop asking me again and again! I wish you would go to hell! If I would know what the word hell means …” (Fig. 31).

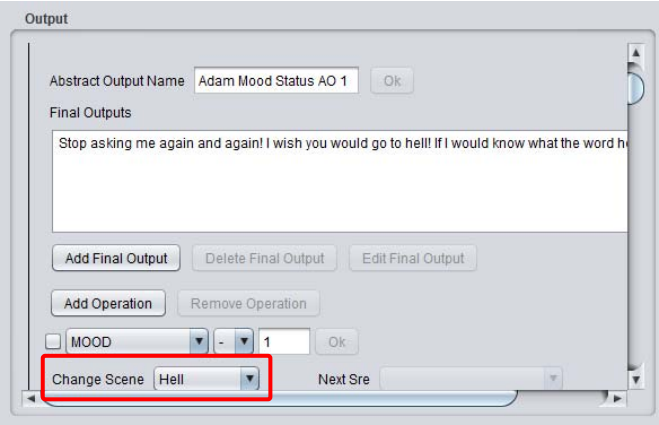

Fig. 31. Scenejo Authoring Tool - change scene

The scene “Hell” is empty yet, because all SREs can only belong to one scene and so far we only created a conversation in “Paradise”. Therefore we choose “Hell” in the Scene tab and create here the first SRE “Adam Blame Eve” for Adam with the AI “Adam Blame Eve AI”, the FIs “START SCENE HELL” and “* FAULT”, the AO “Adam Blame Eve AO” and the FO “Now we’re in hell. That’s your fault!”. Eve will receive the SRE “Eve Blame Adam” with the AI “Eve Blame Adam AI”, the FI “* FAULT”, the AO “Eve Blame Adam AO” and the FO “No! It’s your fault!”

If we now run the story in Scenejo, we observe that as soon as Adam’s mood drops to 0, he gets angry and wishes Eve to hell. Then, they find themselves in the scene “Hell” and blame each other in an endless loop (file: Tutorial_Adam_&_Eve_08.sce.v06.xml).

Step 5: “The User” – adding possibilities for user interaction

To enable user interaction we have to add the following:

- 1 SRE for Adam in the scene “Paradise” with many final inputs that are likely to be entered by the user, in our case containing the word “apple”
- 1 operation for that SRE that raises Adam’s mood

We plan to hint the user to raise Adam’s mood by giving him an apple, therefore we want to enable Adam to react to possible inputs of the user that can be interpreted as such an attempt. At first we add a new SRE named “User Interaction” for Adam in the scene “Paradise” with the AI “User Interaction AI”, the FI “* APPLE”, the AO “User Interaction AO” and the FO “They are delicious! I would like more of these … how are they called, apples?”

The above final input covers all user sentences ending with “apple”. In practice, it is crucial to provide many more alternatives as final inputs that are likely to match the abstract input purpose. Here, finally, the main difficulty lies in providing appropriate interactivity through

design. However, in later more complex tutorial steps, we cover exercises how to get a grip on that issue.

Because Adam has received a delicious apple now, his mood gets better. We do this by adding an operation to the output. His mood increases by 3, expressed by the operation “MOOD + 3” (Fig. 32, file: Tutorial_Adam_&_Eve_09.sce.v06.xml).

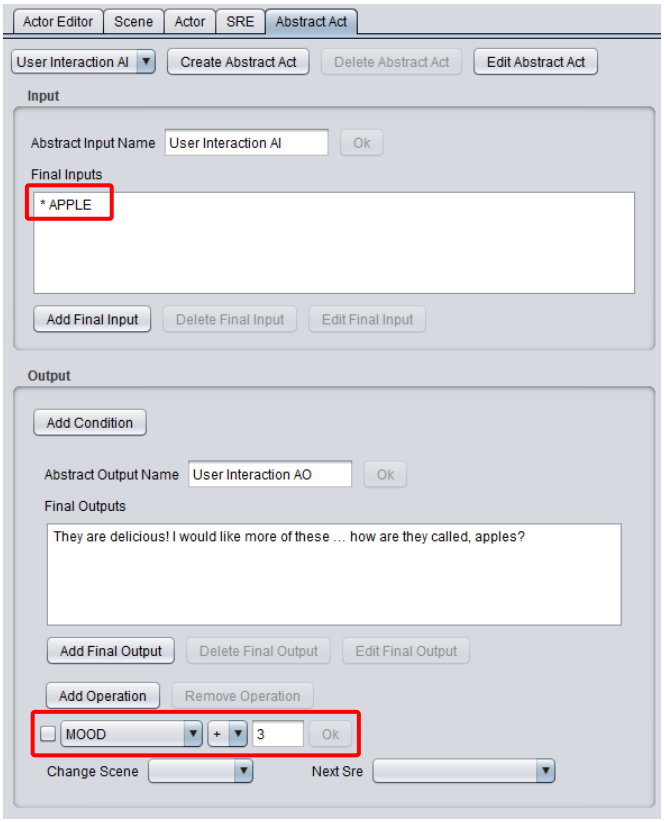

Fig. 32. Scenejo Authoring Tool - user input

In the next step, after too much consumption of apples – when Adam’s mood reaches 6 – we send Adam and Eve out of paradise to the “Cruel World”. We do this by inserting into the SRE “Adam Mood Status” a “Change Scene” to “Cruel World” attached to the AO “Adam Mood Status AO 7”, which takes effect when his mood reaches 6 and he starts to feel fantastic (file: Tutorial_Adam_&_Eve_10.sce.v06.xml).

Finally we create similar SREs like in “Hell” for the “Cruel World” (see end of Step 5), but change “Now we’re in hell. That’s your fault!” to “Now we had to leave paradise, what a cruel, cruel world. That’s only your fault!”

Now we play the final story (file: Tutorial_Adam_&_Eve_11.sce.v06.xml) in Scenejo. If we as end-users do not interact with Adam and Eve, we experience the same story as in Step 5, ending in “Hell”. However, if we offer Adam a few apples, for example by saying “Come on Adam, cheer up, here’s an apple for you” or “Eat an apple, that will make you feel better”, we can change the story and lead it to the “Cruel World” ending.

Concluding, in the last two sections a very simple conversation has been built to get acquainted with the tools. For more complex conversations, it is essential that the conception starts with an outline and more complex abstract modelling as described before.

3.1.5 Conclusion

This section presented a first version of educational material that can be used as introduction to the modelling of interactive verbal conversations, such as with the system Scenejo. It starts with presentations of basic general knowledge in the problems of speech interaction, as far as this is necessary to make authors understand limitations of natural language interaction, which are substantial in the current state of the art. Regarding this goal, it cannot be an introduction to computational linguistics, but gives an impression of some of the technical terms involved. Further, practical exercises with Scenejo have been conceived that shall allow insights into one possible way to transform authors' own ideas into abstract models that can be experienced directly, for example in a seminar. The software, the tutorial files and explanations are provided for download. In effect, the best use of the material is in a course or seminar, such as those recent and planned meetings with the IRIS interest group of authors. In that sense, the theoretical parts were presented to an interested crowd at the conference ICIDS 2010, with a positive feedback. (It has to be noted that the majority of participants already had a general understanding of interactive storytelling.) The practical tutorial with Scenejo needs more time than one day. It has been successfully tested with media and design students at HSRM[9].

The remaining difficulties of using these approaches lie in the transformation of own ideas into the necessary formal abstraction, and in learning how to take advantage of the specific technical boundary conditions of this platform. We assume that parts of the gained general knowledge can then be transferred also to other systems, although system-specific peculiarities of course can not. Scenejo's chosen approach to natural language interaction has limitations, which lie in the simple mechanisms of pattern matching as an approach to handling user input. However, the exact other side of this coin is the accessibility of the platform for beginners, making it convenient for getting started.

The material will be kept up to date during the ongoing IRIS developments. Therefore it is recommended to check out the Online versions and links.[10]

3.2 Modelling with Goals, Tasks and Obstacles

This section covers the process of writing a story with Goals, Tasks and Obstacles, as in a system such as IDtension (Szilas, 2007). This approach is exemplary for the high degree of abstract reasoning it requires from the author (Szilas et al., 2003). Therefore, a specific method needs to be developed to introduce authors with concepts usually dedicated to programmers and remote from the final interactive narrative experience. A new completed IS experience running in IDtension (written by Urs Richle) has been demonstrated at ICIDS 2010 (Szilas et al., 2010) and is evaluated with end-users. The core part of the tutorial is a series of increasingly complex examples, so that each concept is introduced gradually. For that purpose, we have developed:

- A series of variants of the same story, each of which contains its own set of authoring files, in particular for its "goal-task-obstacle" structure.
- Slides covering the key steps in this tutorial, designed to be projected to an audience (appendix A.4).
- A series of slides accompanying each step, designed to be used as a standalone support (ongoing IRIS development).

The story built for this tutorial will be run by IDtension and displayed in a specific interactive textual interface, in which the user is able to choose among several possibilities. The several

[9] Hochschule RheinMain, University of Applied Sciences, Wiesbaden, Germany

[10] Scenejo authoring link collection: http://scenejoauthoringtool.origo.ethz.ch/wiki/link_collection

variants for this story are available Online[11]. It consists of an installer and requires Java. Note that before playing the story, it is highly recommended to see the video included in the distribution. This installation does not include any authoring or development environment, as it is targeted at the end-user. All story code examples can be created with a text editor, using the step-by-step instructions below.

3.2.1 Example story outline

When designing a new scenario, the first step consists in writing the story in plain text. This description is by definition incomplete and inaccurate, but it is the first framing of the story and it is a necessary authoring step. As a working example for the goal/task/obstacle approach, we developed a set of tutorial-scenarios for the narrative engine IDtension with one simple story called “Shalima”:

Shalima is a princess living in her castle. Jack falls in love with her and wants to marry her. But there are Peter and Malcolm, Jack’s enemies, who want to marry her too. The user is playing the role of Jack. Since Shalima is not able to choose among Jack, Malcolm and Peter, the user needs to prove his love first. The best way to prove his love for Shalima is to fight against his enemies Peter and Malcolm.

3.2.2 "Pen & paper" design

Writing the Shalima story in terms of a goal, task, obstacle formalism requires cutting down a story-outline into a series of narrative elements. The first step to achieve this is “pen & paper” design. The advantages of “pen & paper” are the complete freedom in the form and the absence of technical constraints. The first step often lies in thinking about the story outside of the narrative engine to avoid a story-setting influenced too much by technical constraints. Then the author writes down goals, tasks and obstacles.

The very simple story described above can be cut down in the following two goals: “princess married” and “enemy defeated”. To achieve each goal, we can define one or more tasks. For the goal “princess married” we propose “get married” as a unique task. For the goal “enemy defeated” we propose “Fight” and “Fight with a knife” as choices of tasks. Once the user chooses “get married” to achieve the goal “princess married” he encounters a problem: he is “not attractive enough” for the princess Shalima. This obstacle leads him to the second goal “enemy defeated”, meaning that he first has to defeat his enemies in order to become more attractive for the princess. Then, when the user chooses to fight without a knife, he encounters the obstacle “enemy stronger”, which can possibly be overcome by fighting with a knife.

At this stage, this can be written as in Fig. 33.:

Goals	Tasks	Obstacles
Princess married		
	Get married	
		Not attractive
Enemy defeated		
	Fight	
		Enemy stronger
	Fight with a knife	

Fig. 33. Basic outline of two goals, three tasks, and two obstacles.

[11] IDtension example material: http://tecfa.unige.ch/~szilas/iris/IDtensionShalima_2010.jar

3.2.3 Goal, task, obstacle graphs

The next step consists in writing a more formal and complete description of the elements informally described above. This is performed with graphs, first on pen and paper then on Microsoft Powerpoint or equivalent. These graphs follow a certain structure and contain the majority of information necessary to program the scenario in XML.

For the first steps of the Shalima story described above, the graph is (Fig. 34):

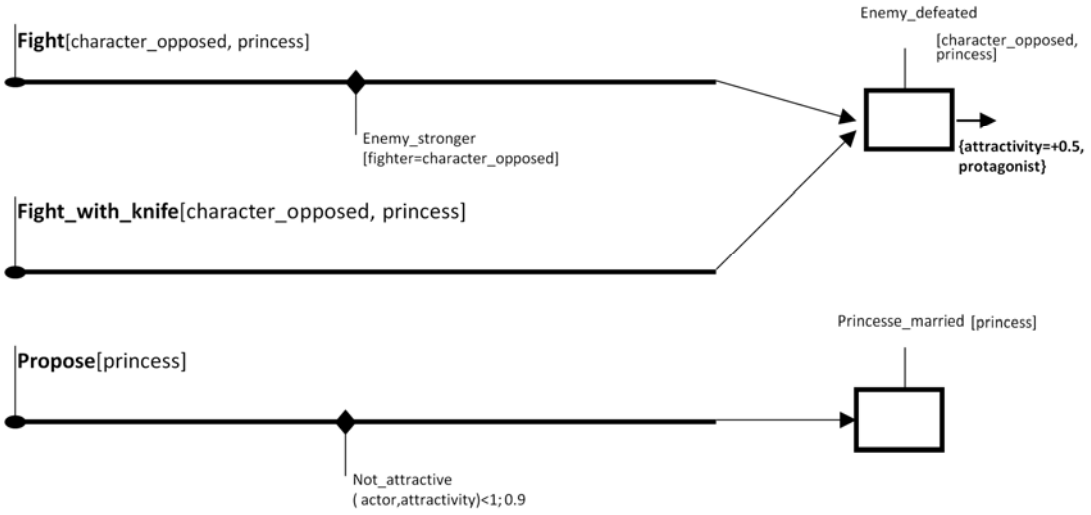

Fig. 34. Story graph with two goals, three tasks, and two obstacles.

Such a graph is rarely written directly in the XML file. Each element is entered piece after piece within the system and iteratively tested.

The tutorial material in the appendices (and available Online) leads step-by-step through the design of this representation with increasing complexity.

3.2.4 Step by step tutorial on writing content for IDtension

In order to create educational material for authors we created a series of mini scenarios based on the Shalima story. Each scenario shows one particular step in the scenario creation process for IDtension. For each scenario, the XML file for the structure and the corresponding CSV file for the texts to be used in the end-user interface are made available in the aterial. Each scenario can be played and tested by the learning author.

Step 1: “Hello World” – the minimal story

The basics for a minimal scenario with IDtension:

- 2 characters: Jack, Peter
- a location: the castle
- an initial goal: enemy_defeated
- an initial task: fight

This story is linear, since the player can only choose to fight and he will succeed in fighting.

The corresponding XML data looks like the following:

```
<decl_personnage>
      <protagoniste/>
      <nom>Jack</nom>
</decl_personnage>

<decl_personnage>
      <nom>Peter</nom>
</decl_personnage>
<decl_objectif>
      <nom>enemy_neutralized</nom>
</decl_objectif>
<decl_tache>
      <nom>fight</nom>
      <initiateur>acteur</initiateur>
      <objectif_vise>enemy_defeated</objectif_vise>
</decl_tache>
```

Step 2: "Several outcomes" – Introduction to obstacles

We add the obstacle:

- enemy stronger

When the player intends to fight his enemy, he might not succeed, because the enemy is stronger.

The following XML code is added:

```
<decl_obstacle>
      <nom>enemy_stronger</nom>
      <risque_pos>0.6</risque_pos>
</decl_obstacle>
```

Step 3: "Two choices" – Second tasks

We add a second task to the goal:

- fight with a knife

To reach his goal, the player has two possibilities.

The following XML code is added:

```
<decl_tache>
      <nom>fight_with_knife</nom>
      <initiateur>acteur</initiateur>
      <objectif_vise>enemy_defeated</objectif_vise>
</decl_tache>
```

Step 4: Variables for characters in goals, tasks and obstacles

We replace real names by variables.

For example, for the goal "enemy_defeated", the parameter "character_opposed" is added:

```
<decl_objectif>
      <nom>enemy_defeated</nom>
      <parametre_generique>
            <type>Personnage</type>
            <nom>character_opposed</nom>
      </parametre_generique>
</decl_objectif>
```

Step 5: "Dialogues" – More characters

We add two more characters:

- princess: Shalima
- second enemy: Malcolm

Due to the characteristics of IDtension, this simple modification enables dialogues among the characters. For example, the player can inform any other character about tasks he has succeeded or failed (obstacles).

Step 6: "Fight with (almost) everybody" – call the same goal for two characters

We call the goal "enemy defeated" for both enemies, Peter and Malcolm.

This simple modification opens several new options in terms of physical performances and dialogues.

Step 7: Attributes

We add the attribute "attractiveness" and assign it to the protagonist Jack

This feature will be necessary for the next step.

In the XML file:

```
<decl_proposition  invisible="false">attractiveness</decl_proposition>

                                    […]

      <decl_personnage>
            <protagoniste/>
            <nom>Jack</nom>
            <carac_num>
                  <nom>attractiveness</nom>
                  <val>0</val>
            </carac_num>
      </decl_personnage>
```

Step 8: Second goal & preconditions

We add the goal "princess married" and show how the obstacle "not attractive" is triggered as long as the player has not defeated an enemy. The goal "enemy defeated" has the consequence to make the player more attractive. This attractiveness level determines if the obstacle is triggered or not.

In the XML file, both the goal and the obstacle are modified. At the level of the goal, a consequence is added:

```
<csq_incrementation>
      <fait>
            <predicat>Relation_p</predicat>
            <param>acteur</param>
            <param>princess</param>
            <param>attractiveness</param>
      </fait>
      <increment>0.5</increment>
</csq_incrementation>
```

At the level of obstacle, a precondition is added:

```
<precondition>
      <contrainte>
            <type>est inférieur</type>
            <param><fait>
                  <predicat>Relation_p</predicat>
                  <param>acteur</param>
                  <param>o_princess</param>
                  <param>attractiveness</param>
            </fait></param>
            <param>0.6</param>
      </contrainte>
</precondition>
```

Step 9: "Propose first, fight after" – Subgoaling

In this case, the player does not at first have the goal to defeat an enemy. It is only when he meets with Shalima and proposes to her, that he fails and thus decides to defeat an enemy to become more attractive.

This is possible with the notion of *cause* which enables the automatic triggering of subgoals. In the XML, the *precondition* tag in the obstacle is simply replaced by the *cause* tag.

3.2.5 Conclusion

With the 9 steps described above, the author can learn progressively how to use abstract concepts involved in an IS approach based on Goals/Tasks/Obstacles. More steps can be added following the same pedagogical approach, to teach more advanced concepts.

This material has been tested during a one-to-many oral presentation. It was successful in giving a flavour of abstraction in a goal-tasks-obstacle approach, but it is not yet suitable for a practical learning of the approach. Thus, our next step is to use it as a more standalone material. Two pedagogical scenarios are envisioned:

- For technical authors, we provide a full development environment, with the task of programming the successive steps. They are assisted by an expert.
- For conceptual authors, we provide the playable stories, the related authoring files and graphical representations of the stories (graphs). Authors can walk through each step, while reading the accompanying material. Each step is punctuated by a series of questions that authors need to answer. The process is guided by an expert, preferably an author, to engage reflective discussion about the process.

3.3 Modelling with Planning

This section is concerned with a method to approach more generative ways to dynamically create the concrete plot of events, by making use of planning. Planning is used in AI-based Interactive Storytelling as one prevalent generative method to solve issues of automatic "drama management". It implies that the order of events is determined by a plan that is calculated by a "planner" – i.e., software able to create paths between given initial states and goal states, following rules. This can either happen during creation and validation of many paths prior of running the experience, or in real time during the interaction, in which case the remaining plot can be rearranged after actions of users. We found that for authors, there is not yet any introductory information available that explains how to integrate planning methods in creative conception, whereas planning seems to be widely used in AI-based IS, for example in (Charles et al., 2003), (Pizzi and Cavazza, 2008), (Porteous and Cavazza, 2009), (Riedl, 2009), (Roberts et al., 2009), (Thomas and Young, 2006), (Thomas, 2006) and (Skorupski, 2009). Therefore we provide educational material targeted at authors. The goals are to impart a basic understanding of the purpose, possibilities and limitations of planning, of basic planning functionality and of ways how to conceive storyworlds for being processed by a planner in general. As side effects, the provided conceptual exercises are also meant to advance conditional thinking and story abstraction in general, so they can be worthwhile for general conception in Interactive Storytelling.

The following material is available, beyond the descriptions herein:

- Presentation slides with a theoretical introduction to planning (appendix A.6)
- An example story conception ("Harold in Trouble") in form of a Card Game, which 1) allows experiencing a simulation of the planner's searching and rearranging functionality by playing cards, and 2) illustrates conception issues. This material consists of:
    - Printer ready version of the card game and use instructions (appendix C.1)
    - Spreadsheets of conception data and plot planning results for the storyworld used in the card game "Harold in Trouble" (appendix C.2 and C.3)
    - Original drafts and reports of our creation process of "Harold in Trouble" (appendices C.4, C.5, C.6)
- Software exercises allowing to reproduce the conception in the "Emo-Emma Authoring Tool", consisting of:
    - Descriptions of using the planning-based Emo-Emma Authoring Tool for the "Harold in Trouble" conception (appendix C.3)
    - Overview and literature links of the Emo-Emma Authoring Tool[12]
    - Links to software Emo-Emma Authoring Tool[13] and user documentation[14]

Playing the card game was tested with participants during the tutorial "Introduction to Interactive Story Creation" at ICIDS 2010 (Spierling et al., 2010a). It was well received as an untechnical and playful basic introduction to plan creation, being suitable for introductions and as a help for creative conception. On the other hand, there are limitations of using paper prototypes concerning the scale of possible storyworlds, therefore using a planning software is inevitable for more complex and continuing exercises.

---

[12] Overview and literature: http://redcap.interactive-storytelling.de/authoring-tools/emo-emma/

[13] Emo-Emma Authoring Tool: http://www-scm.tees.ac.uk/d.pizzi/LRRH09/EmoEmma-AuthoringTool.zip

[14] Emo-Emma Authoring Tool user documenation: http://www-scm.tees.ac.uk/d.pizzi/LRRH09/EmoEmma-AuthoringTool-Manual.pdf

3.3.1 Planning in general

In order to understand what planning is about and how it works, some fundamentals are necessary. Therefore we provide an introduction to planning in the presentation slides (appendix A.6) that covers the most important and crucial concepts:

- Definition and purpose of planning: Generating new paths through designed story events taking into account user interaction
- Definition of propositions, states, actions (operators), preconditions and effects
- Definition of initial and goal state(s)
- What parts are created by authors, and what is left to be solved by the planning engine?
- Two main steps of planning:
    1. Finding allowed actions
    2. Choosing preferred actions (need of a “quality function”)

The first question authors normally would ask when being confronted with the technical concepts of planning could be close to this one: “Why would I need to know something about planning and what are its benefits?”

As pointed out by (Barros and Musse, 2007), there is a clear correspondence between planning algorithms operating on the basis of causal relationships between actions, and stories, which are sequences of actions / events related through some form of causality. Because plans are composed of discrete operations, stories seen as sequences of events can easily be converted to computer-based representations. The role of planning in Interactive Storytelling applications *“is to define the actions or events that must occur during the story so that the world changes from its initial state to some goal state”.* Two main advantages of planning become obvious, the first one is the similarity of plans and stories and the second one is the simple transformation of plans into a representational level.

Planners create an order of actions (or events). Thus, they do not directly reduce the amount of content that has to be authored, but offer a high degree of non-linearity and variation within a given set of events. Once a story is decomposed into single actions (operators) of partial order, a high number of paths (stories) become possible. Another advantage of some planning algorithms is the ability to adapt to changes of the world state during runtime and to perform replanning if the user or other agents interfere with the current plan, and the firstly authored course of actions is possibly changed.

A main challenge in that process is to measure the quality of a plan. In traditional path planning for logistics, this quality may be determined by the length of a path. In storytelling, a short efficient path is most likely not the goal of creation; instead we deal with tension arcs and a range of desired events, even if contradicting a first-order goal. Our provided material is not only intended to teach the basics of planning, but also allows experimenting with and discussing different possible solutions for this challenge.

With the exercise of the designed card game, we make course attendees understand basic principles of a planning algorithm by putting them into the role of that (simple) algorithm. Every possible story event is represented by a card in the game. This whole set of actions / events (as cards) is handed out to the players who now collectively act as a representation of the planner, searching for possible next events. The next sections explain some details of it. Concerning the card game, if we talk about “players”, we mean the algorithm (engine side) of the planner, and “user” denotes the one who is virtually intended to interact later with the designed story.

3.3.2 Abstraction of a story to be used with planning

In the following, a creation and abstraction process that is typical for use with a planner is described by the example of the card game story “Harold in trouble”. More detailed draft documents are found in the appendices.

Short story outline “Harold in Trouble”

The created example story is a “James Bond” parody with some humorous situations, circling around the villain “Silvertoe”, the criminal super brain. After analysing several possible scenes and sub-stories in the thematic framework of “Silvertoe”, we chose one situation that we found suitable for a planning exercise. It is a scene in which Harold, one main character of the story, creates havoc at Silvertoe’s party. The goal is to make Silvertoe so angry that he leaves his party. Harold himself is the clumsy assistant of a top agent (the potential user) - who wants to stop Silvertoe - and a wannabe womaniser. His main personal goal is to seduce women at that party. Trying this by impressing a lady, Harold creates some chain reactions that let the event get out of control, like inflaming a poodle, poison the punch, drenching other guests and damaging the music equipment. The final story called “Harold in Trouble” (formerly “Silvertoe”) consists of one scene: “The garden party”.

Engine-independent design

The basic conception steps of “Harold in Trouble” illustrate several creative principles explored in an interdisciplinary process. The goal in the technical and formal abstraction process is a coded plan representation, consisting of a collection of plan operators (the actions in the story), together with descriptions of their pre-conditions and post-conditions. Less technically, the design approach can be seen as thinking about action-centred acting situations (compare section 2.3.1). The design goal is to define possible actions and events in the storyworld, and to describe the conditions in which they can happen, as well as what they “do to the world” in terms of their consequences.

The first outline provided by Georg Struck[15] has been based on creative principles of situation comedy, as those presented by (Vorhaus, 1994). In particular, Harold is conceived as a “comic character”, someone with a “bumbler’s comic perspective” of acting clumsily, while there is a mismatch of high self-concept (fantasy) and his performed skills (reality). As a consequence, all actions performed by Harold shall turn out with the opposite of what he obviously intended to achieve, which results in chaos. This is used as a guideline to create the effects (post-conditions) of actions. Following the abstraction principle of iterative delinearisation and linearisation (see section 2.2.2), then the story has to be described in terms of “propositions”, which describe facts and knowledge about possible states in which the world can be. This creation step could be based on several abstraction techniques already explored and tested in IRIS with other examples, such as described in (Spierling and Hoffmann, 2010), in which main character attributes and events are extracted from a given story, then being categorised and grouped concerning their similar effects on the main world states. Concluding, we see the design process of this story example as exemplary for a successful approach, by combining novel creative principles (described in chapter 2) with traditional principles of storytelling. Intermediate drafts created by Georg Struck and Steve Hoffmann can be found in the appendix C, following the here proposed approach of steps:

1. Creating a draft linear story
2. Extracting / defining possible events and actions
3. Extracting / defining story relevant variable states and attributes
4. Abstracting the above by building groups of events with similar effects
5. Adding interactivity

[15] Georg Struck is a member of the IRIS interest group of authors. He has been consulted in IRIS as an independent story author.

The result of this first iteration is an overview of the intended possible actions and partially ordered chains that now can be expanded to meet the requirements of a planning domain. One example of such a chain can be seen in Fig. 35. It also illustrates how events are grouped to more general building blocks, allowing for variations later.

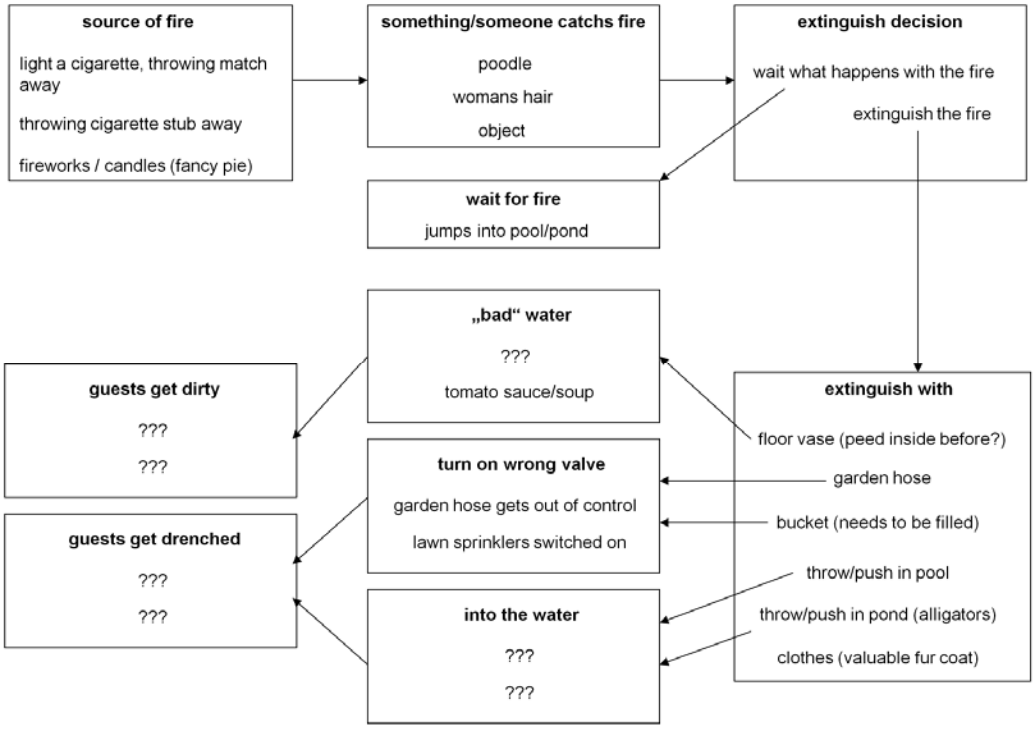

Fig. 35: Example draft of a creation step - one chain in detail

3.3.3 Planning-based modelling

The following examples demonstrate the advancement of the concept towards using a planner. For example, one chain begins with Harold who wants to light the cigarette of the lady he tries to seduce. But after carelessly throwing away his match, the Lady's poodle catches fire. This short list of actions (called "operators" in planning) can be seen in Fig. 36.

Actions
Harold lights cigarette with a match
Harold carelessly throws match away
Poodle burns

Fig. 36. Description of planning operators - step 1.

Further, we need to define the conditions under which a certain action can be performed, depending on the current world state. This is done by listing "propositions" that have to be first checked as true by the planner in order to enable this action – as so-called "pre-condition". In order to change the world state as a consequence of an action, effects need to be described as well. In the kind of planner we use in our example, as effects we simply add and/or delete "propositions" from the world state, which means, added propositions become true after the action, and deleted propositions are then not true anymore.

For a simple start, we suggest to begin with describing propositions as effects, following up on the list in Fig. 36. We think about what can be the result of the first action and - going one step further - how this effect could be connected to any next action. One solution can be that the “Cigarette is lit”. This is an obvious result of the first action and can be a reason for the second one, because Harold does not need the match any more if the cigarette is lit.

An effect of the second action could be a new proposition stating that “Fire is spread”, which will not only allow our poodle to burn but also to leave an opportunity to let other things catch fire. Finally our last actions lead to the effect that the “Poodle is burning” and the raise of Silvertoe’s anger. “Something burns” is an example of an abstract proposition, because it is general enough to cover multiple other propositions. Adding it to the world state allows us to use it as a more general precondition for actions, for example if it is not important what exactly is on fire. It could then be used as a condition to let a guest scream “Fire! Fire!”. Fig. 37 shows the resulting spreadsheet with the added propositions after each operator (action).

Actions	Effects
	Add
Harold lights cigarette with a match	Cigarette is lit
Harold carelessly throws match away	Fire is spread
Poodle burns	Someone burns
	Raise Silvertoes anger by 1
	Poodle is burning

Fig. 37. Formal description of planning operators - step 2.

Now we add preconditions to the actions. Because we want to chain them, we use the added effect of one action as a precondition for the following one. Fig. 38 shows the results.

Preconditions	Actions	Effects
		Add
	Harold lights cigarette with a match	Cigarette is lit
Cigarette is lit	**Harold carelessly throws match away**	Fire is spread
Fire is spread	**Poodle burns**	Someone burns
		Raise Silvertoes anger by 1
		Poodle is burning

Fig. 38. Formal description of planning operators - step 3.

The described preconditions of an action are not “sufficient” to make actions happen, but they are “necessary” before the action can be chosen. For any action, there can be a number of propositions defined as preconditions. In other words, an action can only be performed (which means, an operator can be selected by the planner) in situations, in which all preconditions are true. We can also say: All propositions defined as a precondition for an action have to be “in the world state” in order to allow this action to happen.

With each added proposition as an effect, the selection of other actions in need of this proposition may become possible (depending also on their other preconditions). This can also lead to unwanted side effects, such as loops. For example, Harold can now throw away the match again and again, because “Cigarette is lit” will always be active in the current and the following world states. The solution to prevent such situations is to remove unneeded propositions from the world state. We do so by deleting “Cigarette is lit” and “Fire is spread” after they have been “used” by the corresponding action, leading to the spreadsheet in Fig. 39.

Preconditions	Actions	Effects	
		Add	Delete
	Harold lights cigarette with a match	Cigarette is lit	
Cigarette is lit	**Harold carelessly throws match away**	Fire is spread	Cigarette is lit
Fire is spread	**Poodle burns**	Someone burns	Fire is spread
		Raise Silvertoes anger by 1	
		Poodle is burning	

Fig. 39. Formal description of planning operators - step 4.

Finally, we add a few more preconditions to the first action, the presence of a “Bored woman” and the fact that the woman has a cigarette in her hand. The third action requires that the “Poodle is present”. Removing “Cigarette is not lit” from the first action prevents an unwanted loop. Fig. 40 shows the final description of our three example actions, which now can be directly transferred into an action card for the card game or being entered into a planning tool like the Emo-Emma Authoring Tool.

Preconditions	Actions	Effects	
		Add	Delete
Bored woman	**Harold lights cigarette with a match**	Cigarette is lit	Cigarette is not lit
Woman has cigarette in hand			
Cigarette is not lit			
Cigarette is lit	**Harold carelessly throws match away**	Fire is spread	Cigarette is lit
Fire is spread	**Poodle burns**	Someone burns	Fire is spread
Poodle is present		Raise Silvertoes anger by 1	
		Poodle is burning	

Fig. 40. Formal description of planning operators - step 5.

3.3.4 Using the Card Game

Accompanying material can be found in the appendices C (C.1 is the printer ready card game, to be printed on 10x15 cm photo cards).

Description of the cards

The game consists of two general types of cards: 1. action cards and 2. proposition cards. The action cards are shown in Fig. 41 and Fig. 42, the proposition cards in Fig. 43. They can be divided into three subtypes: 1. general propositions, 2. initial propositions and 3. goal propositions.

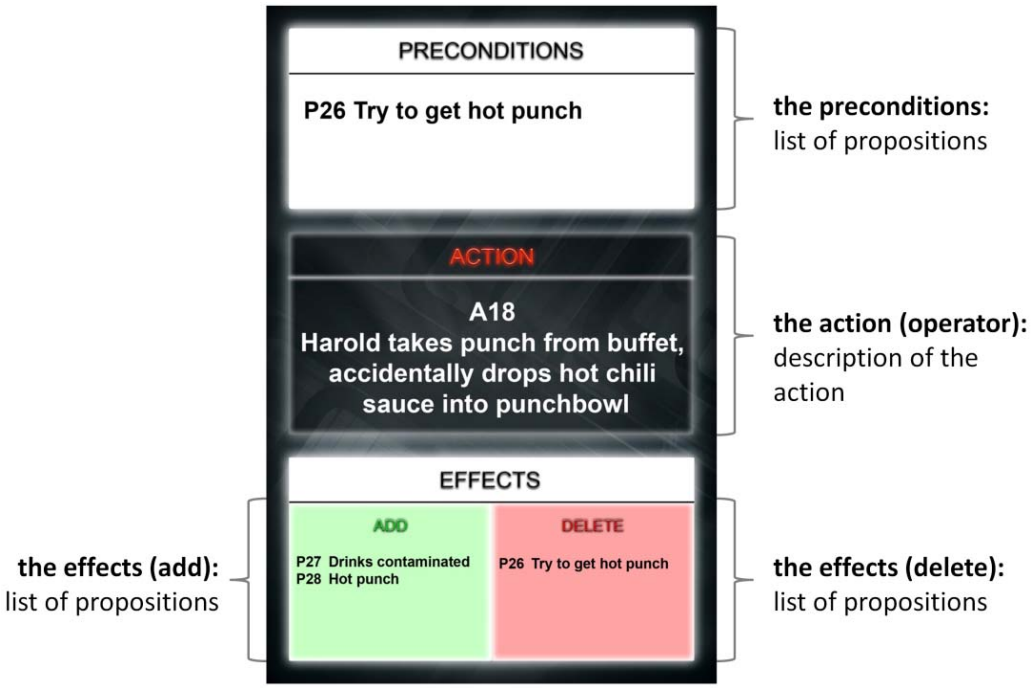

Fig. 41. Description of action cards.

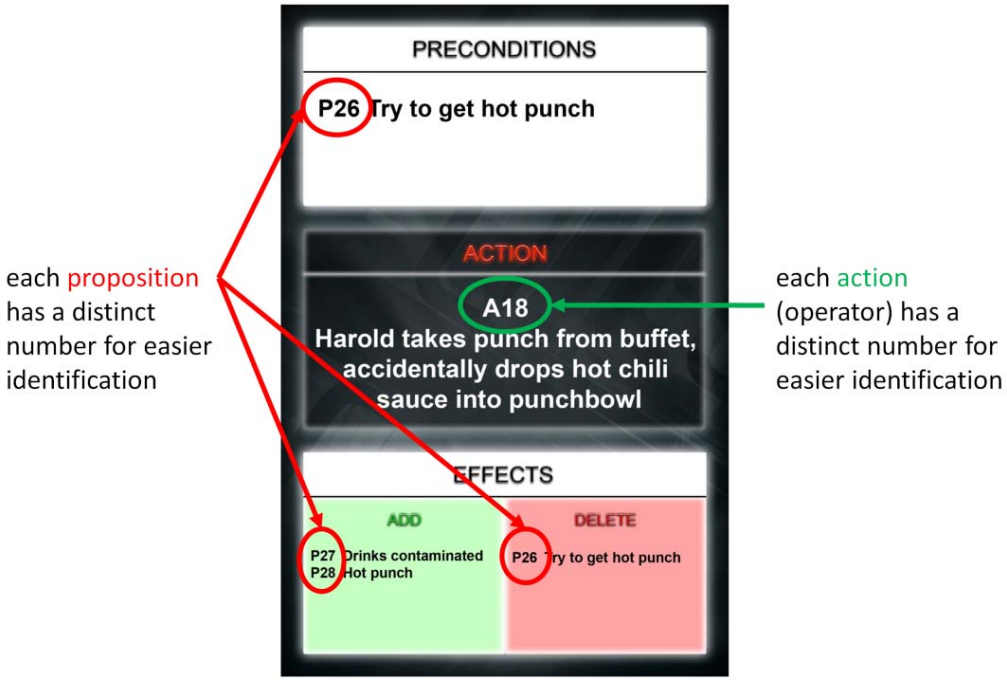

Fig. 42. Description of action cards.

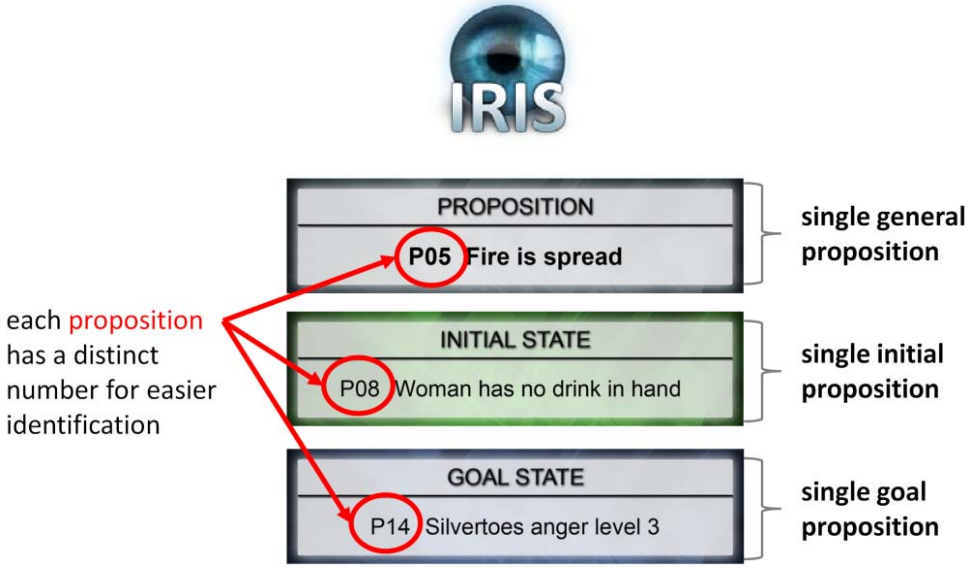

Fig. 43. Description of proposition cards.

The actions cards are intended to be handed out to the players, the proposition cards are needed to describe the world state and can be handled by the game facilitator in the first steps. They can also be used by the players after they understood the principles to get some practice and deeper understanding.

Preparations

We suggest using one big or two smaller tables with a size of approximately 1.5 x 1.5 meters. One half of the area should be used for the representation of the storyworld state, the other half is used to keep track of the evolving story by connecting action cards. The largest part of the storyworld side is marked as “world state” (we have drawn a cloud), similar to the images in the educational slides. The cloud shall imply that here, only temporal snapshots of an ever-changing current state can be read. The green proposition cards are assigned to a field called “Initial state”, the only blue proposition card to a field called “Goal state”. All other (grey) proposition cards are placed on the table in their described order (P01 to P46).

Before the game can start, the initial state has to be arranged in the “world state” cloud by including those grey propositions that match the green ones of the “Initial state”. The setup can be seen in Fig. 44 and a practical example from the tutorial at ICIDS 2010 in Fig. 45.

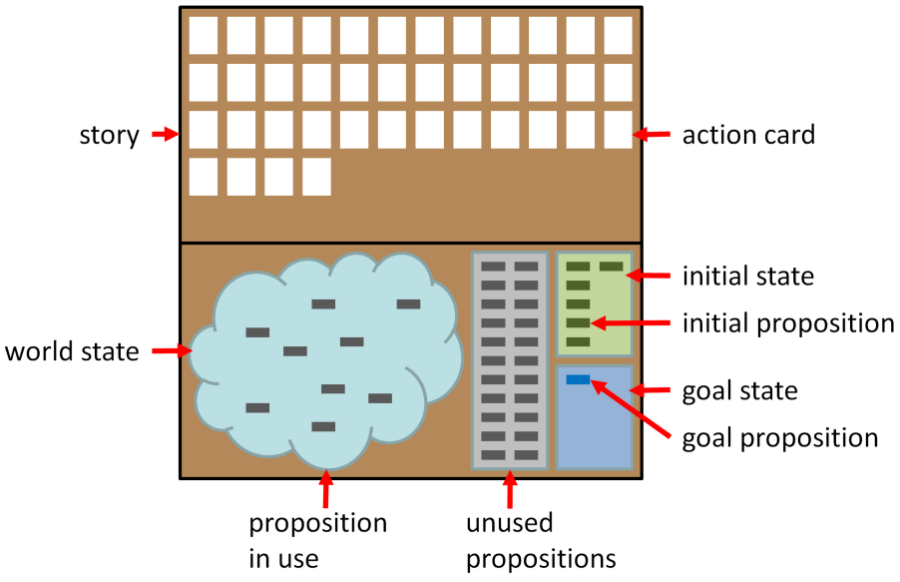

Fig. 44. Possible setup of the card game on two tables.

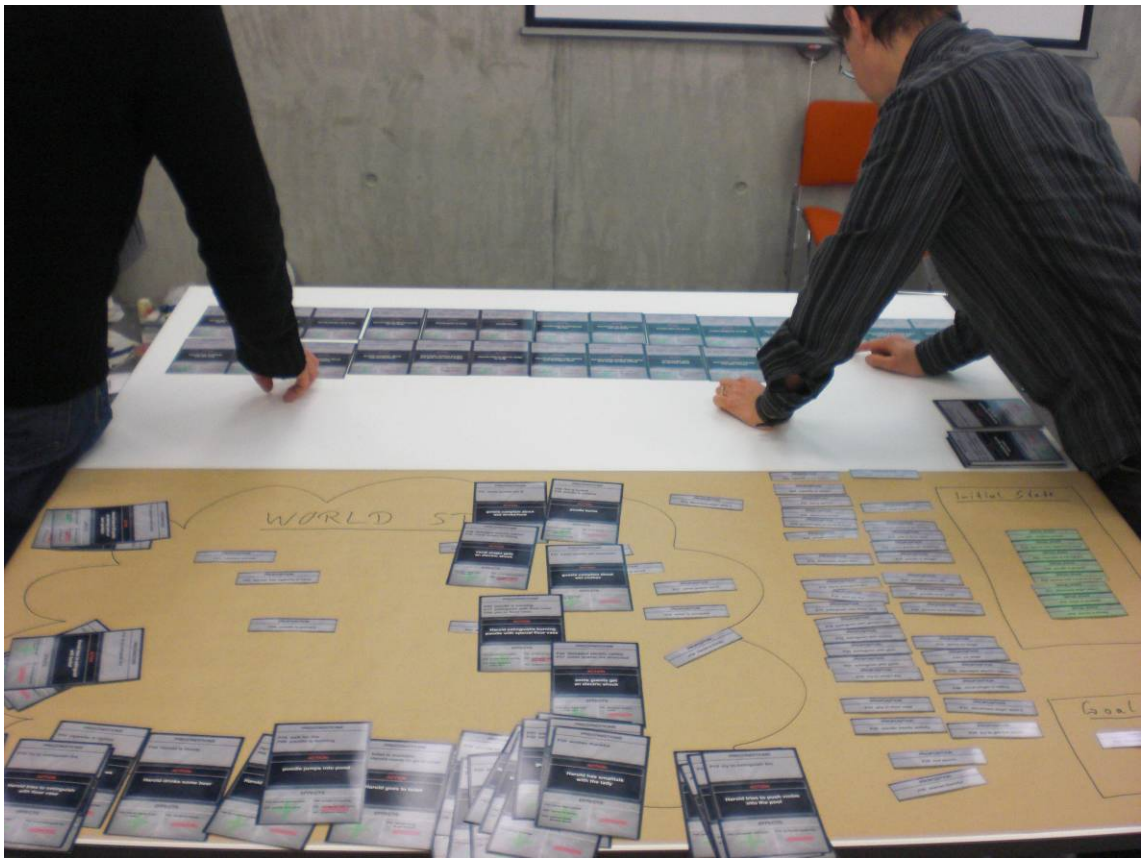

Fig. 45. Example of the setup at ICIDS 2010 tutorial.

Finally the action cards will be handed out to the players. The number of cards for each player depends on the number of possible players and the amount of available time for the exercise. We suggest handing out a minimum of three action cards to each player and using all 40 of them.

Game modes

The task of the players is to think like a collective planner in the first place, which means, to search their cards for possible actions that can be played in a given situation. Constantly and without taking turns, players compare the propositions listed as preconditions on their hand of cards with the current world state. If there is a matching card, they can place it like in a Domino game at the end of the evolving sequence of cards.

It is possible to play the game in different modes and with different rule sets, depending on the intended complexity of the conveyed knowledge. The first game mode describes the general handling of the action cards, the second one the procedure of choosing an action card. The alternative modes of the two categories can also be used in different combinations.

1. “Remove” and “Put Back”

Two types are possible: “Remove” and “Put Back”. In the first one, each action card can only be used once and will be placed on the story side of the table after it was played.

The second mode allows keeping the action card that was played. Therefore more sets of action cards are needed to keep track of the story. Alternatively it is possible to write down the actions that make up the story instead of laying them out with action cards.

In the “Put Back” game type, loops in the resulting plot are possible. For example, Harold can drink some beer and, as an effect, wants to go to the toilet after that. The actions “A22 Harold drinks some beer” and “A24 Harold goes to toilet” can be played in an alternating order, which is coherent, but not very interesting for the progress of our story. However, if sometime during the story the toilet is blocked and he can not use it, another story chain becomes possible, leading to more interesting actions. This example shows the possible occurrences of either wanted or unwanted loops. Authors here experience the limitations of doing this design on paper, because with increasing complexity, the results of multiple interfering loops in a planning domain are not easy to predict. Computation can make their life easier, but still these are design decisions to be made based on the desired result.

2. “First come, first served” and “Picking the best”

The second game mode covers the strategy of deciding which action card has to be played if more then one is possible. We suggest to start with the simplest strategy “First come, first served”, which means that the first found “valid” card is to be played. Practically this can be a kind of competition amongst the players to be the first one who finds a suitable action card. Whoever finds the most during the game can be declared as the “winner”. After the game, this strategy has to be revisited. The result can illustrate a planner optimised for quick search of suitable actions, resulting in one of many possible trajectories. It can then be discussed how to proceed to find the best path.

The second one results in a discussion of several planning strategies, because here we want to discuss how to “pick the best” action cards for our story. In planning, “quality functions” need to be described, on which evaluations of a “best action” can be based. In traditional planning for logistics, these quality functions are mostly concerned with efficiency issues, for example, reaching a goal with the lowest cost. In Interactive Storytelling, there is not yet enough authoritative knowledge how to define this quality, as this is still state-of-the-art research. It also depends on the intentions of authors of a storyworld concerning the end-user experience, in terms of maximising fun, tension, challenge, immersion, emotion, learning or whatever the goals are. For the time being, a proposed approach is that in an interdisciplinary team, authors and engineers collaborate in finding these strategies, as the community has just begun to find such quality functions. Here we offer a field of discussion and experiments to participants, who can try to find the “best actions” and discuss how their strategy could be formalised into general rules.

User interaction and prototyping

Note that in our exercise and card game, user interaction is not (yet) in the focus and therefore not represented in the game elements. However, through extending the game, users can also participate and can therefore be simulated. This can be simply done by defining one (or more) participants in the game as “player”, who is allowed to add or delete propositions to/from the world state by performing self chosen actions. The game facilitator can decide in which scope these actions can take place.

As a consequence, we can also think of this as a general prototyping method to be performed by future authors within the design iterations of storyworld ideas.

Possible stories

It is not straight-forward to get an overview of all possible stories that can be created with our card game. Therefore one possible general structure of the game is provided in Fig. 46. The arrows are showing a partial order, so will action card A03 normally be played after A02. If arrows branch out, alternative chains become possible.

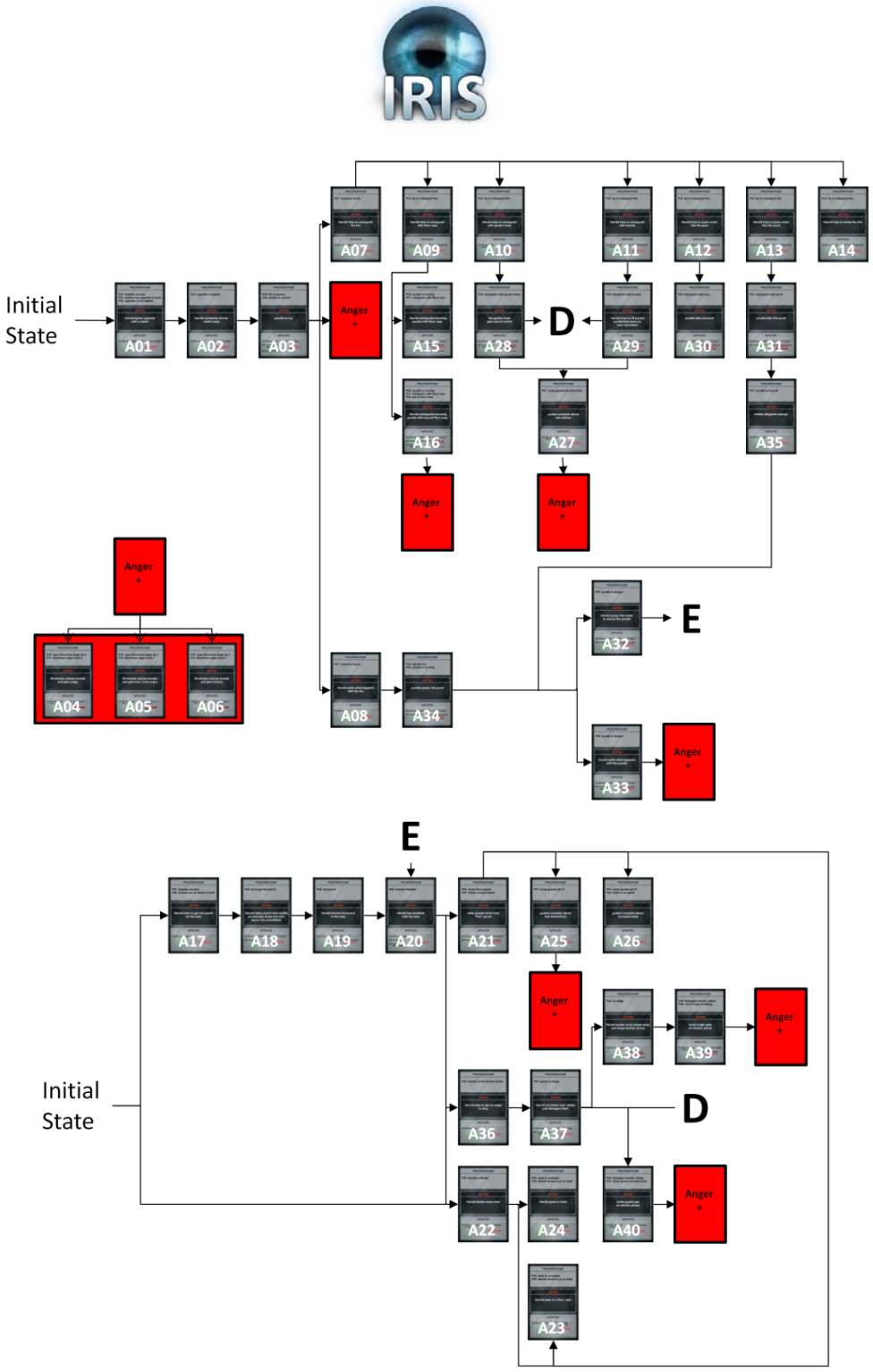

Fig. 46. Possible structure of the card game story.

Fig. 47 shows the example plot that has been achieved in the card game session during the tutorial “Introduction to Interactive Story Creation” at ICIDS 2010. It depicts the created order of actions and at each step, the resulting impacts on the world state. A more detailed version also including preconditions can be found in appendices C (Story 1).

Initial World State		
Bored woman Woman has cigarette in hand Cigarette is not lit	Poodle is present Woman has no drink in hand Silvertoes anger level 0	Toilet is available Harold is thirsty

Action	ADD Effect	DELETE Effect
Harold lights cigarette with a match	Cigarette is lit	Cigarette is not lit
Harold carelessly throws match away	Fire is spread	Cigarette is lit
Harold drinks some beer	Harold needs to go to toilet	Harold is thirsty
Harold tries to get hot punch for the lady	Try to get hot punch	
Harold goes to toilet		Harold needs to go to toilet
Poodle burns	Someone burns Raise Silvertoes anger by 1 Poodle is burning	Fire is spread
Harold tries to extinguishs the fire	Try to extinguish fire	
Harold tries to push victim into the pond	Extinguish with pond	Try to extinguish fire
Poodle falls into pond	Poodle is drenched Poodle is in pond	Poodle is burning Extinguish with pond Someone burns
Hidden alligators emerge	Poodle in danger	
Harold jumps into water to rescue the poodle	Poodle is saved Woman thankful	Poodle in danger Poodle is in pond
Silvertoe notices trouble and gets angry	Silvertoes anger level 1	Raise Silvertoes anger by 1 Silvertoes anger level 0
Harold takes punch from buffet, accidentally drops hot chili sauce into punchbowl	Drinks contaminated Hot punch	Try to get hot punch
Harold serves hot punch to the lady	Woman thankful	Hot punch Woman has no drink in hand
Harold has smalltalk with the lady	Some time passes Harold is thirsty Woman loves Sinatra music	
Other guests drink from "hot" punch	Some guests are ill Toilet is occupied	Some time passes Toilet is available
Guests complain about bad drinks/food	Raise Silvertoes anger by 1	
Silvertoe notices trouble and gets even more angry	Silvertoes anger level 2	Raise Silvertoes anger by 1 Silvertoes anger level 1
Harold tries to get on stage to sing	Going on stage	
Harold stumbles over cables and damages them	On stage Damaged electric cables	Going on stage
Harold pushs vocal singer away and sings terribly wrong	Vocal singer is falling	On stage Woman loves Sinatra music Woman thankful
Vocal singer gets an electric shock	Raise Silvertoes anger by 1	Vocal singer is falling Damaged electric cables
Silvertoe notices trouble and gets furious	Silvertoes anger level 3	Raise Silvertoes anger by 1 Silvertoes anger level 2

Fig. 47. Final story 1 created during tutorial at ICIDS 2010.

Fig. 48 shows an idealised alternative plot with the same layout. It was created by using the Emo-Emma Authoring Tool, the planning tool presented in (Pizzi and Cavazza, 2008). Details and two other example stories can be found in the appendices C (Story 2, Story 3&4).

Initial World State		
Bored woman Woman has cigarette in hand Cigarette is not lit	Poodle is present Woman has no drink in hand Silvertoes anger level 0	Toilet is available Harold is thirsty

Action	ADD Effect	DELETE Effect
Harold lights cigarette with a match	Cigarette is lit	Cigarette is not lit
Harold carelessly throws match away	Fire is spread	Cigarette is lit
Poodle burns	Someone burns Raise Silvertoes anger by 1 Poodle is burning	Fire is spread
Silvertoe notices trouble and gets angry	Silvertoes anger level 1	Raise Silvertoes anger by 1 Silvertoes anger level 0
Harold tries to extinguishs the fire	Try to extinguish fire	
Harold tries to extinguish with bucket	Extinguish with bucket	Try to extinguish fire
Harold tries to fill bucket, accidentally turns on lawn sprinklers	Poodle is drenched Some guests are drenched	Poodle is burning Extinguish with bucket Someone burns
Guests complain about wet clothes	Raise Silvertoes anger by 1	
Silvertoe notices trouble and gets even more angry	Silvertoes anger level 2	Raise Silvertoes anger by 1 Silvertoes anger level 1
Harold tries to get hot punch for the lady	Try to get hot punch	
Harold takes punch from buffet, accidentally drops hot chili sauce into punchbowl	Drinks contaminated Hot punch	Try to get hot punch
Harold serves hot punch to the lady	Woman thankful	Hot punch Woman has no drink in hand
Harold has smalltalk with the lady	Some time passes Harold is thirsty Woman loves Sinatra music	
Harold tries to get on stage to sing	Going on stage	
Harold stumbles over cables and damages them	On stage Damaged electric cables	Going on stage
Some guests get an electric shock	Raise Silvertoes anger by 1	Damaged electric cables
Silvertoe notices trouble and gets furious	Silvertoes anger level 3	Raise Silvertoes anger by 1 Silvertoes anger level 2

Fig. 48. Final (idealised) story 2.

3.3.5 Conclusion

With the created material and the story examples, we provide prospective authors and story creators with a first insight into basic principles of planning-based Interactive Digital Storytelling. We show advantages of such a generative approach, for example the automatic search for an order of actions that has been only implied by an author, to be able to achieve a great variability of plot alternatives during runtime. We also show limitations, which mainly lie in the difficulty of judging the best order of actions in the context of an interesting story experience. The material enables interdisciplinary discussions, through which authors can get involved to further develop the research field, crossing accessibility barriers due to the technical concepts.

As side effects, the materials and courses train the ability of thinking in conditions and of structuring a storyworld domain with regard to possible world states, which is useful whether or not planning software is going to be employed. The importance of the principle of abstraction as a helpful creation technique is another topic that is conveyed.

It is important to raise realistic expectations of how generative methods can support the creation. In our experience, authors attending meetings and workshops sometimes had the a-priori optimistic assumption that generative AI technologies can relieve them from a lot of work, which is supported by claims made in research papers. However, our material can be used to illustrate and further explore the boundaries of using planning for “solving the authoring bottleneck” in Interactive Storytelling. Planning cannot be seen as a universal remedy, as there is still a huge amount of content that has to be authored, combined with more technical barriers. As mentioned before, in order to appreciate generative methods, a certain threshold of complexity in a storywold has to be crossed. It is part of future work in the project to design follow-up versions based on the experiences with the first set of material.

We had a lot of positive feedback for the card game in the evaluation questionnaires at the ICIDS 2010 tutorial (see appendices). However, we are well aware that this prototype can not demonstrate all aspects of planning. It is suitable to depict the process of a single search step in a given world state and to raise awareness for the time needed to find possible next actions, depending mainly on the size of the search space. It is further a base for discussion regarding the design of quality functions that determine the “best” possible actions. In this respect, it has limitations for understanding the full planning process. Therefore, after using it as introductory material, we recommend practical exercises with a planning tool. Ways to convey these aspects still have to be further explored and are future work.

3.4 Modelling with State Machines

This section reports on work in progress that leads to a more complex creation example of a storyworld, to be finalised in the third year of IRIS. Extending the educational material, it can then serve as a released and presentable example of Interactive Storytelling, accompanied by detailed reports on the creation methods. It is a joint effort of IRIS partners, involving Georg Struck as an author. The first version of material presented here complements the sections above by the concept of state machines.

State machine modelling is another common approach to Interactive Storytelling, used not only in computer science. The advantage is a relatively simple structure and ability to describe states and state changes, which relates to stories and also to planning (see above). It has also been employed in the Scenejo platform (see above) and many more IS systems.

The storyworld under construction is intended to have an approximated duration of 15 minutes in the interactive user experience. It mainly runs on the “Advanced Agent Animation” system (Fig. 49) described below (Damian et al., 2010), with excerpts that can be run in Scenejo (see section 3.1). Thereby it combines interaction in a 3D environment with conversational elements. For authoring based on state machine modelling, the software “SceneMaker” is used. Conceptually, an outline and a first abstraction of events and states exist. Based on being able to walk around in a scenario similar to the “beer garden scenario” depicted in Fig. 49, and being

able to have conversations with story characters, we model a class reunion with three interwoven plots of relationship issues, mystery disclosure, and funny aspects similar to the "Harold in Trouble" story (see section 3.3).

Fig. 49. First glimpse of the "beer garden" example scenario.

At this stage, this section provides a basic understanding of state machines and shows again the need of the ability of conditional and abstract thinking. The following material is provided:

- Software "Advanced Agent Animation"[16]
- Original draft of the linear story in progress (available as appendix on request)

3.4.1 State machines

State machines or finite-state machines (FSMs) are a commonly used behaviour model in computer science that consists of three basic elements:

- states
- transitions (that connect states to each other)
- actions (entry actions - entering a state, exit actions - exiting a state, input actions, transition actions)

State machines are represented with state transition tables or state transition diagrams. Fig. 50 and Fig. 51 show simple examples of a door that can only be opened or closed. In a state table the combination of the current state and an input shows the next possible state at its crossing point.

[16] Advanced Agent Animation: http://mm-werkstatt.informatik.uni-augsburg.de/projects/aaa/

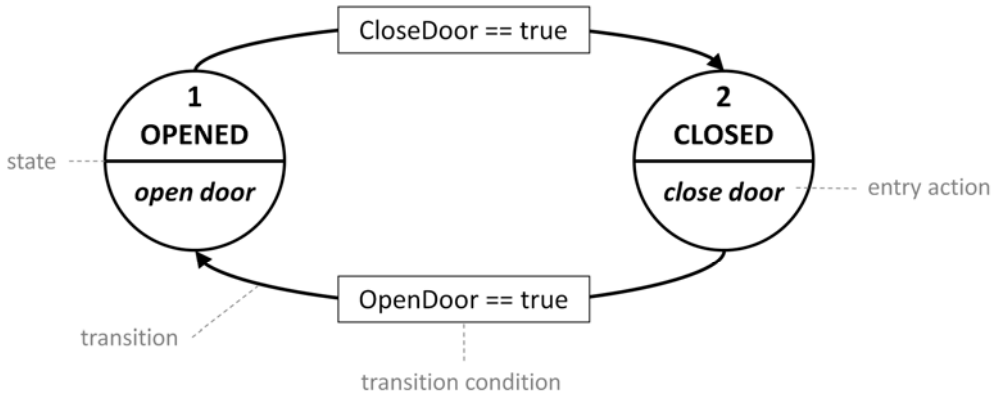

Fig. 50. State transition diagram - door example.

	State 1	State 2
OpenDoor == true	-	State 1
OpenDoor == false	-	-
CloseDoor == true	State 2	-
CloseDoor == false	-	-

Fig. 51. State transition table - door example.

A state machine always begins in a special state called “start state” and then goes through transitions to other states, ending (if successful) in another special state named “accept state” (or one state from a set of “accept states”). The transitions are responsible for the state changes and can have conditions that first have to be fulfilled to enable the transition and to realise the state change.

A more complex example can be seen in Fig. 52. This example was created with the authoring tool “SceneMaker” and demonstrates a broader range of abilities that state machines can have. In the figure circles are simple states (called nodes) and squares are so called super nodes, which means that they can contain another state machine. With super nodes, state machines can be hierarchically structured.

It also shows some of the different kinds of transitions that SceneMaker uses:

- Simple transitions (without conditions)
- Timeout transitions
- Conditional transitions
- Probabilistic transitions
- Interrupting transitions

The results created with the authoring tool SceneMaker can be directly connected to different representation platforms. One is the Advanced Agent Animation system that makes use of the Horde 3D engine. For example, certain scripts created by SceneMaker can control the avatars in the Advanced Agent Animation. These scripts can contain text to be spoken, as well as animation commands (see Fig. 53). Such little scripts can be connected to nodes in a state graph. They will be played if the running state machine reaches the specific node.

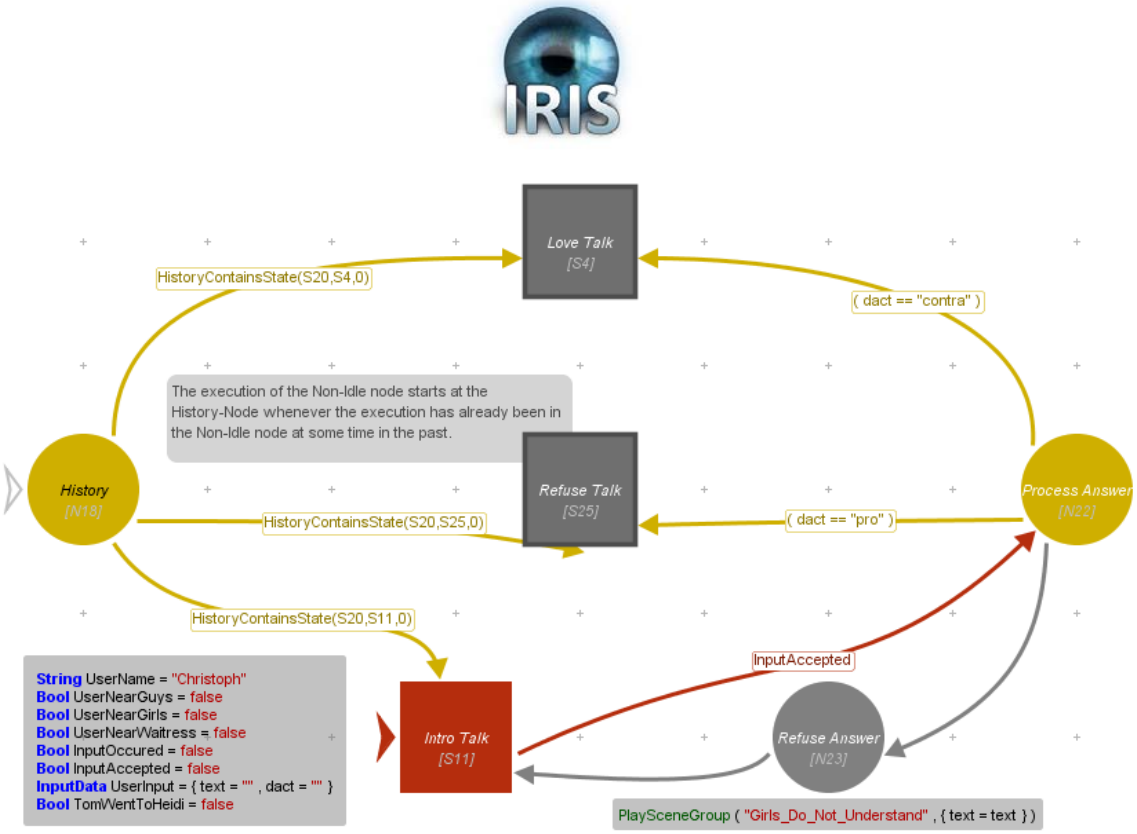

Fig. 52. Example of a state machine realised with SceneMaker.

Scene_en : Girls_Introducing_Talk_Part_1

G:[anim name=Becking_2] **Hi** `$UserName`**. Nice to see you again.**

S:[anim name=Becking_1] **Hey. How are you?**

Fig. 53. Example script from SceneMaker.

3.4.2 Example story outline

The intended example story under development is called “The Downfall of Frank Jefferson Junior” and circles around a reunion party in a beer garden. It has a soap opera like setup containing a love story and will make use of humoristic elements known from sitcom. The story is structured in three subplots that blend into each other during the unfolding of the actions.

The main character prospectively played by the user is Ben, whose goals are set out to try to win back his lost love Julia, who will be proposed by Frank Jefferson Junior at the end of the evening. This is the main plot of the story which serves as a spine for the other two. One is the subplot of Barry, Ben’s best friend, who tries to seduce Lydia, which leaves a lot of chaos on the way. The other one circles around Ben and Jill, Julia’s sister, who both try to have some fun at the party by teasing the other guests. The full story draft (26 pages in the current state of development) is available on request.

Engine-independent abstraction

We started again with a linear draft story outline and are right now at the point of identifying and developing new meaningful actions for the story which can be finally used as basis for a formal representation. Another parallel step is to extract emotional states that drive the story for each character. One example draft in this process can be seen in Fig. 54, where we try to explore how the action “confess” (performed by Lydia) depends upon different emotions of her own and of other characters.

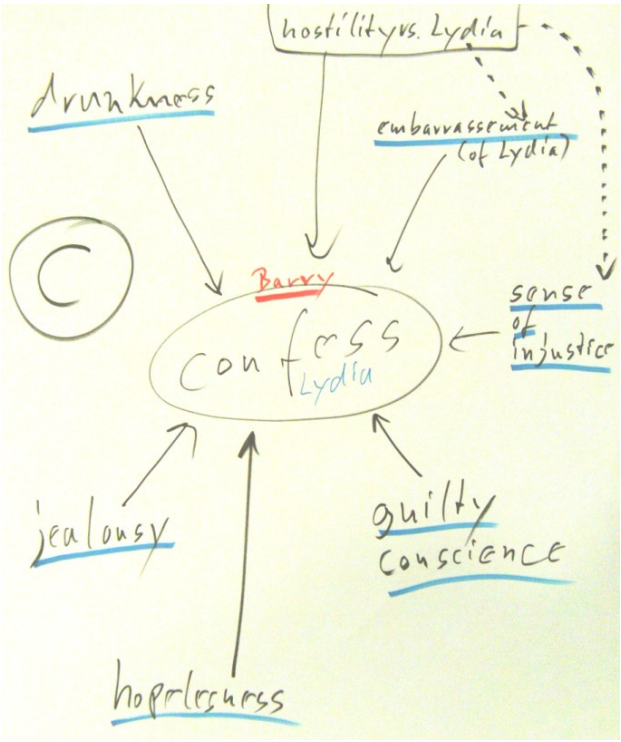

Fig. 54. The action "confess" (Lydia) and influencing emotions.

The found emotions were then combined into a single variable character attribute, the "chattiness" of Lydia, which can be influenced over the time of the experience by actions of other characters, by her consume of alcohol (which can also be influenced by other agents) and by user interaction. Fig. 55 shows a first draft of the impact of Lydias changing "chattiness" level. It will be used for certain preconditions and lead to different shocking and state-changing confessions and revelations, which will then influence other characters.

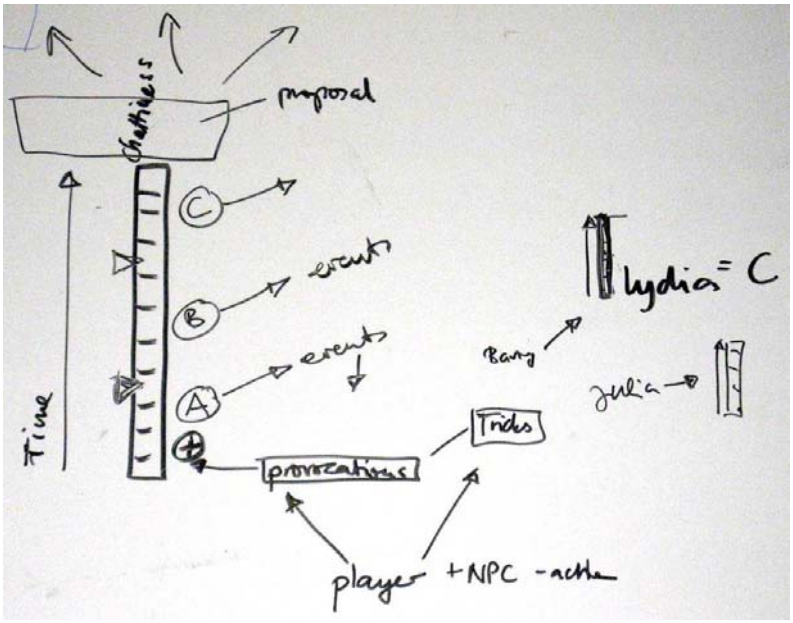

Fig. 55. Drafting the "chattiness" of Lydia.

3.4.3 Conclusion

The creation of this story and the case-based educational material is still in progress and will evolve throughout 2011 with the goal of a useful general example of Interactive Storytelling and as a test case for explored principles put in practice.

4. Conclusion and Outlook

4.1 Summary

This report describes educational material targeted at current and prospective authors and story creators in Interactive Storytelling, who want to explore increasingly generative AI-based methods for the conception of interactive story experiences. The material is available as appendices to this report, most of it accessible Online (and beyond that on request). After its initial release, it will be updated subsequently.

The material is the result of practical exercises in authoring Interactive Storytelling artefacts, performed in teams of computer scientists with non-computer scientists, partly members of the IRIS authors' interest group. These exercises have been accomplished and/or analysed during the second year of IRIS, involving the work of several partners. IRIS-based systems used as target platforms include the conversational platform Scenejo (Hochschule RheinMain), the goal/task/obstacle models of IDtension (University of Geneva), the planning software Emo-Emma Authoring Tool (University of Teesside) and the state-machine-based AAA System with SceneMaker (University of Augsburg).

We have demonstrated novel creative principles in practice that are complementary to traditional storytelling principles. They build a bridge between AI-based approaches and story concepts. The main researched principles concern "abstraction" at several levels and thinking in "conditions" of events. Further creative principles have been enunciated that are to be explored in more detail later.

4.2 Evaluation and Self-Assessment

Parts of the material have been publicly presented at a pre-conference half-day tutorial at the ICIDS 2010 conference in Edinburgh, on 31st October 2010 (Spierling et al., 2010a). After the session, we evaluated the tutorial by questionnaires and personal discussions. The questionnaires are available as appendices of this document. The questions asked had the goal to evaluate the general scope and form of presenting, without an attempt to ask detailed questions of what was learned. The latter is something that needs to be done at further events that are designed to go into more detail, in other words, that are not limited by their half-day duration. As a self-assessment remark, we found the time of a half day too short to convey the complex topics. However, many participants gave us positive feedback on this question concerning the duration. Practical exercises were not included, apart from playing the card game for the introduction to planning. We estimate that learning effects will be stronger if more practice is involved. We designed the practical exercises for future use in a longer-lasting summer school with interested authors. Practical tool exercises have so far been tested with students at our Universities.

Directly after the tutorial, we conducted another half-day workshop on education in Interactive Storytelling (Spierling et al., 2010b). This workshop pointed out several more existing issues in cross-disciplinary work. It was common sense at the workshop that education materials for AI-based Interactive Storytelling are missing, that technical introductions are often too general to be usable by people who only want to apply these techniques for storytelling (such as, pointers to general planning that do not consider storytelling concepts and examples). As a result, participants expressed their interest of mutually sharing educational material across disciplines. As a follow-up, Online repositories for educational material are organised by IRIS, with the material presented herein as a start.

The tutorial evaluation questions at ICIDS 2010 were grouped into assessing the previous experience of the participants, evaluating the reception of our presentation of abstract concepts,

and estimating the usefulness of some of the principles and the whole idea of educating authors. In general, the overall feedback was encouraging. The positive evaluation of the card game exercise was standing out in the feedback. A statistical analysis is inapplicable because of the participant number of 18. We further received useful hints for improvements of the material. It has to be noted that contrary to our preferences, only few (5) story creators without a computer science background attended the session. Looking for correlations in the questionnaires, it is remarkable that for example, people with prior knowledge of planning found that this concept is important for authors, whereas people with no prior knowledge gave the contrary feedback (however again, statistically this is a weak observation). More than through the questionnaires, personal discussions after the workshop day supported this impression that one possible effect of education may also be to appreciate the usefulness of a certain technique in the first place. Further, we again faced the need to have possibly more demonstration material available in general in this community.

This underlines the importance of more available completed interactive experiences in IS, especially for education and motivation. These need to be bigger as a software demonstrator, but do not need to be extensive. They are necessary to achieve more detailed feedback of creative principles by going through the whole cycle of creation up to the evaluation of user experience. Then, the representation level (such as visuals, animation and sound) is another difficulty to be faced, which has been left out by all our exercises that circled mainly around the issues in finding variable and flexible orders of events.

More or less “formative” self-assessment has been done constantly during our creation exercises. As such, we appraise and assess our own work as a case-based research approach, leading to new insights and awareness of those things that work well and other possible issues that need more attention in the future. Some personal reflections of involved team members on this creative work are available as appendix E.

4.3 Future Work

In the introduction, it was motivated to have two directions of possible developments to make AI-based systems more amenable to authors; first, the development of accessible and better authoring tools for generative engines, and second, the education of authors. In this report, the main emphasis was on the second motivation, by making an effort to bridge comprehension gaps created by the same technology that shall actually help creators in their processes. Nevertheless, we have further developed and enhanced little tools as part of the educational material. Future work – which has already begun – focuses also again on the first part, by analysing user requirements based on new insights gained through our content developments.

The ongoing storyworld development of the beer garden scenario with the current title “The Downfall of Frank Jefferson Junior” continues, with the goal to achieve a completed experience of 15 to 20 minutes duration based on personal conflicts, comedy and interaction through conversation. This will inform to a great extent the enunciation of further creative principles in combination with the possibility to evaluate the end-user perspective of what has been created. In general, the development and evaluation of creative principles put in practice in productions is a long-term endeavour reaching beyond the duration of the IRIS project. In the third year, concluding the experiments, the intended international outreach to creators and practitioners includes a summer school in Interactive Storytelling.

5. List of Appendices

This report has the following appendices, which consist mainly of educational material to be used in a course (slides and exercise material), background material such as drafts of our own story conception exercises, Online links to further sources such as software, documentation and papers, and evaluation documents.

All Online links are accessible via http://iris.interactive-storytelling.de. It is recommended to check Online for the newest versions of the material and software.

A ICIDS 2010 Tutorial – "Introduction to Interactive Story Creation"

The ICIDS 2010 tutorial material is available at http://icids2010.interactive-storytelling.de

A.1 Complete set of tutorial slides
presented at the ICIDS 2010 workshop in Edinburgh, 31st October 2010

A.2 Tutorial material excerpt: "Introduction"

A.3 Tutorial material excerpt: "Creative Principle – Abstraction of Stories"

A.4 Tutorial material excerpt: "IDtension Tutorial Models"

A.5 Tutorial material excerpt: "Creative Principle – Conditional Events"

A.6 Tutorial material excerpt: "Planning in Interactive Storytelling"

B Topic: Conversational Storytelling

All materials concerning Scenejo and the Scenejo Authoring Tool are available at http://scenejoauthoringtool.origo.ethz.ch/wiki/link_collection

B.1 Basic tutorial – Scenejo Authoring Tutorial 1

B.2 Step-by-step tutorial – Scenejo Authoring Tutorial 2

B.3 Tutorial files to be opened in the Scenejo Authoring Tool or played back with Scenejo

B.4 Software download of the Scenejo Authoring Tool

B.5 Software download of Scenejo

C Topic: Planning Introduction, Card Game Material

The material is available at http://icids2010.interactive-storytelling.de .

C.1 Complete set of cards, printer-ready (high resolution versions available on request)

C.2+3 Documentation of possible stories: spreadsheets and listings of the Emo-Emma Authoring Tool

C.4-6 Documentation of the creation process of "Harold in Trouble": Story outlines, abstraction drafts

C.7 Links to download the Emo-Emma Authoring Tool and accompanying information are available at http://redcap.interactive-storytelling.de/authoring-tools/emo-emma/

D Topic: Modelling with IDtension

Materials concerning IDtension are available at http://tecfa.unige.ch/~szilas/iris/

D.1 IDtension step-by-step tutorial, http://tecfa.unige.ch/~szilas/iris/IDtension_Tutorial.pdf

D.2 Software download of the IDtension Shalima examples to be run in IDtension http://tecfa.unige.ch/~szilas/iris/IDtensionShalima_2010.jar

D.3 Accompanying information is available at http://redcap.interactive-storytelling.de/authoring-tools/idtension/

E Feedback and Evaluation

E.1 Evaluation questionnaires of the ICIDS 2010 tutorial

E.2 Personal feedback of authors involved in the creation exercises of tutorial material

F Publication

Available on request: Description of a modelling process concerning a literature adaptation to Interactive Storytelling of a Hemingway short story.

F.1 ICIDS 2010 Springer publication – “Exploring Narrative Interpretation and Adaptation for Interactive Story Creation” (Spierling and Hoffmann, 2010), http://www.springerlink.com/content/427m35n3404u4l40/

Available on request: Demonstration description of a modelled completed experience with IDtension.

F.2 ICIDS 2010 Springer publication – “Using Highly Interactive Drama to Help Young People Cope with Traumatic Situations” (Szilas et al., 2010), http://www.springerlink.com/content/u4734498pk205368/

6. References

1) (Austin, 1962)
Austin, J.L. (1962). How to do things with Words. The William James Lectures delivered at Harvard University in 1955, Oxford University Press.

2) (Aylett et al., 2005)
Aylett, R.S., Louchart, S., Dias, J, Paiva, A., Vala, M. (2005). FearNot! - An Experiment in Emergent Narrative. In: Proceedings of IVA 2005, Intelligent Virtual Agents, LNCS, vol. 3661, Springer-Verlag Heidelberg, pp. 305-316.

3) (Barros and Musse, 2007)
Barros, L. M., Musse, S. R., 2007. Planning algorithms for interactive storytelling. In: Computers in Entertainment, Volume 5, Number 1, 2007.

4) (Bremond and Cancalon, 1980)
Bremond, C., Cancalon, E. (1980). The Logic of Narrative Possibilities. On Narrative and Narratives: II, New Literary History 11, 3 (Spring 1980), pp. 387-411.

5) (Charles et al., 2003)
Charles, F., Lozano, M., Mead, S. J., Bisquerra A. F., Cavazza M., 2003. Planning Formalisms and Authoring in Interactive Storytelling. In: Technologies for Interactive Digital Storytelling and Entertainment, Proceedings of TIDSE 2003, ZGDV Computer Graphik Edition Band 9, Fraunhofer IRB Verlag, Stuttgart, pp. 216-225.

6) (Cockburn and Williams, 2001)
Cockburn, A., Williams, L. (2001). The costs and benefits of pair programming. Book section: Extreme programming examined. Addison-Wesley Longman Publishing Co. Inc, pp. 223 – 243.

7) (Crawford, 2002)
Crawford, C. (2002). Artists and Engineers as Cats and Dogs: Implications for Interactive Storytelling. Computers & Graphics, Volume 26, Issue 1 (February 2002), Elsevier Science Ltd., pp. 13-20.

8) (Crawford, 2004)
Crawford, C. (2004). Chris Crawford on Interactive Storytelling. New Riders Publishing.

9) (Damian et al., 2010)
Damian, I., Huber, P., Endrass, B., Bee, N. (2010). Advanced Agent Animation. GALA 2010 (Gathering of Animated Lifelike Agents) of the IVA 2010 conference, Philadelphia, USA. Online: http://mm-werkstatt.informatik.uni-augsburg.de/projects/aaa/

10) (Dooley and Levinsohn, 2001)
Dooley, R.A., Levinsohn, S.H. (2001). Analyzing Discourse: A Manual of Basic Concepts. SIL International, Dallas, USA.

11) (Herman, 2002)
Herman, D. (2002). Story Logic: Problems and Possibilities of Narrative. University of Nebraska Press, Lincoln.

12) (Howard and Mabley, 1993)
Howard, D., and Mabley, E. (1993). The Tools of Screenwriting: A Writer's Guide to the Craft and Elements of a Screenplay. St. Martin's Press, New York, USA.

13) (IRIS-WP3, 2009)
Spierling, U., Hoffmann, S., Szilas, N. (2009). Report on Prescriptive Narrative Principles and Creation Methods in Interactive Storytelling (Non-Digital and Digital). Deliverable 3.1, IRIS NoE FP7-ICT-231824, http://iris.scm.tees.ac.uk/publications

14) (Kelso et al., 1993)
Kelso, M.T., Weyhrauch, P., Bates, J. (1993). Dramatic Presence. In Presence: Journal of Teleoperators and Virtual Environments, 2(1), 1–15.

15) (Kriegel et al., 2007)
Kriegel, M., Aylett, R., Dias, J., Paiva, A., (2007). An Authoring Tool for an Emergent Narrative Storytelling System. In: AAAI Fall Symposium On Intelligent Narrative Technologies, Technical Report FS-07-05, AAAI press, Arlington, USA, pp. 55-62.

16) (LaValle, 2006)
LaValle, Steven M., 2006. Planning Algorithms. Cambridge University Press, 2006.

17) (Lévi-Strauss, 1962)
Lévi-Strauss, C., 1962. The Science of the Concrete. From “The Savage Mind”, University of Chicago Press.

18) (Longacre, 1996)
Longacre, R. E. (1996). The Grammar of Discourse. 2nd Edition, Topics in Language and Linguistics, Plenum, New York.

19) (Louchart and Aylett, 2005)
Louchart, S., Aylett, R. (2005). Managing a Non-linear Scenario - A Narrative Evolution. In: Virtual Storytelling, Proceedings of ICVS 2005, LNCS, Vol. 3805, Springer-Verlag Heidelberg, pp. 148–157.

20) (Louchart et al., 2004)
Louchart, S., Romano, D.M., Aylett, R., Pickering, J. (2004). Speaking and Acting – Interacting Language and Action for an Expressive Character. In: Proceedings of the Symposium on Language, Speech and Gesture for Expressive Characters, part of the AISB 2004 Convention: Motion, Emotion and Cognition, University of Leeds, UK, 29 March - 1 April 2004.

21) (Louchart et al., 2008)
Louchart, S., Swartjes, I., Kriegel, M., Aylett, R., (2008). Purposeful Authoring for Emergent Narrative. In: First Joint International Conference On Interactive Digital Storytelling, Proceedings ICIDS 2008, LNCS, Vol. 5334, Springer-Verlag, Heidelberg, pp. 273-284.

22) (Mateas and Stern, 2004)
Mateas M., Stern A. (2004). Natural Language Understanding in Façade: Surface-Text Processing. In: Göbel et al. (eds.): TIDSE 2004, Proceedings, LNCS, vol. 3105, Springer-Verlag Heidelberg, pp. 3-13.

23) (Mateas and Stern, 2005a)
Mateas, M., Stern, A. (2005). Procedural Authorship: A Case-Study Of the Interactive Drama Façade. In: Proceedings of Digital Arts and Culture (DAC), Copenhagen

24) (Mateas and Stern, 2005b)
Mateas, M. and Stern, A. (2005). Build it to Understand It: Ludology Meets Narratoloy in Game Design Space. In: Selected papers from the DiGRA 2005 International Conference: Changing Views, Worlds in Play. Vancouver, Canada.

25) (McKee, 1997)
McKee, R. (1997). Story: Substance, Structure, Style, and the Principles of Screenwriting. Harper Collins Publishers, New York, 1997.

26) (Orkin, 2006)
Orkin, J. (2006). Three States and a Plan: The AI of F.E.A.R. Proceedings of the Game Developer's Conference 2006 (GDC).

27) (Pizzi and Cavazza, 2008)
Pizzi, D., Cavazza, M. (2008). From Debugging to Authoring: Adapting Productivity Tools to Narrative Content Description. In: Spierling, U., Szilas, N. (Eds.): Interactive Storytelling,

Proceedings of ICIDS 2008, LNCS, vol. 5334, Springer Verlag Berlin-Heidelberg, pp. 285-296.

28) (Porteous and Cavazza, 2009)
Porteous, J., Cavazza, M. (2009). Controlling Narrative Generation with Planning Trajectories: the Role of Constraints. In: Zagalo, N., Iurgel, I. Petta, P. (Eds.): Interactive Storytelling, Proceedings of ICIDS 2009, LNCS, vol. 5915, Springer Verlag Berlin-Heidelberg, pp. 234-245.

29) (Riedl, 2009)
Riedl, M. O., 2009. Incorporating Authorial Intent into Generative Narrative Systems. In: AAAI Spring Symposium (AAAI 2009), Stanford, California, USA, March 23–25, 2009.

30) (Roberts et al., 2009)
Roberts, D. L., Riedl, M. O., Isbell, C. L., 2009. Beyond Adversarial: The Case for Game AI as Storytelling. In: Proceedings for the Conference of the Digital Games Research Association (DiGRA 2009), London, UK, 2009.

31) (Russell and Norvig, 2003)
Russell, S. Norvig, P. (2003). Artificial Intelligence – A Modern Approach. 2nd Edition, Pearson Education International, Prentice Hall, Upper Saddle River, USA.

32) (Searle, 1969)
Searle, J. R. (1969). Speech Acts: An Essay in the Philosophy of Language. London: Cambridge University Press (1969). German Translation, Sprechakte: Ein sprachphilosophischer Essay, Suhrkamp, Frankfurt (1971).

33) (Si et al., 2007)
Si, M., Marsella, S. C., Pynadath, D. V., 2007. Proactive Authoring for Interactive Drama: An Author's Assistant, in: Proceedings of IVA 2007, 7th International Conference on Intelligent Virtual Agents (Paris, France).

34) (Skorupski, 2009)
Skorupski, J., 2009. Storyboard authoring of plan-based interactive dramas. In: Proceedings of the 4th International Conference on Foundations of Digital Games (FDG 2009), Orlando, USA, April 26-30, 2009.

35) (Spierling, 2007)
Spierling, U. (2007). Adding Aspects of “Implicit Creation” to the Authoring Process in Interactive Storytelling. In: M. Cavazza, S. Donikian (Eds.): Virtual Storytelling. Proceedings of ICVS 2007, LNCS, vol. 4871, Springer-Verlag Berlin-Heidelberg, pp. 13-25.

36) (Spierling, 2008)
Spierling, U. (2008). ‘Killer Phrases’: Design Steps for a Game with Digital Role-Playing Agents. In: Z. Pan et al. (Eds.): Transactions on Edutainment I, LNCS 5080, Springer-Verlag Berlin Heidelberg, pp. 150–161.

37) (Spierling and Hoffmann, 2010)
Spierling, U., Hoffmann, S. (2010). Exploring Narrative Interpretation and Adaptation for Interactive Story Creation. In: Interactive Storytelling, Proceedings of ICIDS 2010, Edinburgh, LNCS, vol. 6432, Springer Verlag Berlin-Heidelberg, pp. 50-61.

38) (Spierling and Szilas, 2009)
Spierling, U., Szilas, N. (2009). Authoring Issues Beyond Tools. In: International Conference on Interactive Digital Storytelling (ICIDS 2009), Guimarães, Portugal, pp. 50–61.

39) (Spierling et al., 2006)
Spierling, U., Weiß, S., Müller, W., (2006). Towards Accessible Authoring Tools for Interactive Storytelling. In: Technologies for Interactive Digital Storytelling and Entertainment, Proceedings of TIDSE 2006, LNCS, vol. 4326, Springer Verlag Berlin-Heidelberg, pp. 187-192.

40) (Spierling et al., 2010a)
Spierling, U., Szilas, N., Hoffmann, S., Richle, U., 2010. Tutorial: Introduction to Interactive Story Creation. In: Interactive Storytelling, Proceedings of ICIDS 2010, Edinburgh, LNCS, vol. 6432, Springer Verlag Berlin-Heidelberg, pp. 299-300.

41) (Spierling et al., 2010b)
Spierling, U., Szilas, N., Hoffmann, S., Richle, U., 2010. Workshop: Education in Interactive Digital Storytelling. In: Interactive Storytelling, Proceedings of ICIDS 2010, Edinburgh, LNCS, vol. 6432, Springer Verlag Berlin-Heidelberg, pp. 289-290.

42) (Stern, 2001)
Stern, A. (2001). Deeper conversations with interactive art, or why artists must program. Convergence: The Journal of Research into New Media Technologies, Vol. 7, No. 1. Available at http://interactivestory.net/papers/deeperconversations.html. (Last accessed: 28.12.2010)

43) (Swartjes and Theune, 2008)
Swartjes, I., Theune, M. (2008). The Virtual Storyteller: Story Generation by Simulation. In: Proceedings of the 20th Belgian-Netherlands Conference on Artificial Intelligence (BNAIC).

44) (Swartjes and Theune, 2009)
Swartjes, I., Theune, M. (2009). Iterative Authoring Using Story Generation Feedback: Debugging or Co-Creation? In: Zagalo, N., Iurgel, I. Petta, P. (Eds.): Interactive Storytelling, Proceedings of ICIDS 2009, LNCS, Volume 5915, Springer-Verlag Heidelberg, pp. 62-73.

45) (Szilas, 2007)
Szilas, N. (2007). A Computational Model of an Intelligent Narrator for Interactive Narratives. Applied Artificial Intelligence, 21(8), pp. 753-801.

46) (Szilas et al., 2003)
Szilas, N., Marty O., Réty, J.-H. (2003). Authoring Highly Generative Interactive Drama. In: Balet et al. (Eds.): Virtual Storytelling, Proceedings of ICVS 2003, LNCS, Vol. 2897, Springer-Verlag Heidelberg, pp. 37-46.

47) (Szilas et al., 2010)
Szilas, N., Richle, U., Boggini, T., Dumas, J. (2010). Using Highly Interactive Drama to Help Young People Cope with Traumatic Situations. In: Interactive Storytelling, Proceedings of ICIDS 2010, Edinburgh, LNCS, vol. 6432, Springer Verlag Berlin-Heidelberg, pp. 279-282.

48) (Thomas, 2006)
Thomas, J. M., 2006. Collaborative Authoring of Plan-Based Interactive Narrative. In: Workshop on AI Planning for Computer Games and Synthetic Characters (ICAPS 2006), Ambleside, The English Lake District, U.K., June 6-10, 2006.

49) (Thomas and Young, 2006)
Thomas, J. M., Young, R. M., 2006. Author in the Loop: Using Mixed-Initiative Planning to Improve Interactive Narrative. In: Workshop on AI Planning for Computer Games and Synthetic Characters (ICAPS 2006), Ambleside, The English Lake District, U.K., June 6-10, 2006.

50) (Vorhaus, 1994)
Vorhaus, J. (1994). The Comic Toolbox – How to be funny even if you're not. Silman James Press, Los Angeles.