An Authoring Framework for Interactive Narrative with Virtual Characters

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

DISSERTATION

zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)

von

Dipl.-Inform. Ido A. Iurgel, M.A.
geboren in São Paulo, Brasilien

Referenten der Arbeit:
Prof. Dr. José L. Encarnação, Technische Universität Darmstadt
Prof. Dr. Marc Alexa, Technische Universität Berlin
Prof. Dr. Adérito F. Marcos, Universidade do Minho, Portugal

Tag der Einreichung: 02.08.2007
Tag der mündlichen Prüfung: 21.09.2007

Hochschulkennziffer D17

Darmstadt 2008

Table of Contents

Kurzfassung

Im Forschungsgebiet „interaktive Geschichten“ stellt der Erstellungsprozess einen entscheidenden Engpass dar, der die Entwicklung dieses Gebiets derzeit behindert. Ein Grund hierfür ist eine Besonderheit des Gebiets der „Interaktiven Geschichten“: Hier ist es notwendig, die Anwendungen so zügig wie möglich zu realisieren, um Ideen konkretisieren und Technologien erproben zu können. Theoretische Überlegungen allein sind in diesem Gebiet nicht ausreichend, weil die Anforderungen erst im Laufe des Entwicklungsprozesses deutlich werden können.

Bisher existierten keine Verfahren und keine Softwareumgebungen, um unterschiedliche Ideen für interaktive, narrative Anwendungen schnell zu realisieren und iterativ weiterzuentwickeln, bei denen zugleich sowohl das Anwendungsdesign, die Autorenwerkzeuge als auch die Technologien nach und nach angepasst werden können.

Diese Dissertation stellt den ersten generischen Softwarerahmen vor, mit dessen Hilfe ein solcher agiler Entwicklungsprozess für interaktive Geschichten mit virtuellen Charakteren möglich wird. Der Fokus liegt dabei auf visuellen Erstellungsmethoden. Der Softwarerahmen besteht aus einer vollständigen, funktionsfähigen Softwareumgebung mit Autoren-werkzeugen, die ihren praktischen Nutzen und ihre Allgemeinheit im Rahmen verschiedener Projekte bereits bewiesen hat.

Als Basis des Softwarerahmens wurde ein neuartiger, hybrider visueller Formalismus entwickelt, der verschiedene Arten der Kombination von gerichteten Graphen mit anderen Kontrollmechanismen erlaubt. Durch diese Kombination kann die Entwicklungsgruppe den gerichteten Graphen als sofort einsetzbaren Hauptformalismus verwenden, und andere Kontrollverfahren partiell und inkrementell bei Bedarf im Rahmen des fortschreitenden Entwicklungsprozesses hinzufügen. Durch den Einsatz des relativ einfach handhabbaren gerichteten Graphen als Basis wird garantiert, dass die zusätzlich eingesetzten Technologien tatsächlich den konkreten Erfordernissen der Anwendung und des Inhalts entsprechen.

Dieser Softwarerahmen erleichtert die Zusammenarbeit von Informatikern, Fachleuten und Inhaltsspezialisten bei der Entwicklung innovativer und in der Regel experimenteller Anwendungen. Er ist besonders geeignet für Systeme, die auf einem Satz festgelegter Basiselemente oder festgelegter Verhaltensschablonen beruhen, und bei denen sich das Problem der Erstellung auf die Frage reduzieren lässt, wie die Basiselemente in eine korrekte Reihenfolge gebracht werden können.

Eine Reihe neuartiger Konzepte für narrative Anwendungen wird ebenfalls vorgestellt, zusammen mit den grundsätzlichen Schemata, die erforderlich sind, um den Softwarerahmen für die Erstellung dieser Anwendungen einzusetzen. Hierbei zeigt sich, dass der Rahmen die Anforderungen der Erstellung erfüllt, und konkretisiert auch bestimmte Hypothesen des Verfassers darüber, warum Interaktivität und Erzählung kombinierbar sind.

Schließlich stellt diese Dissertation auch das Modell eines Autorensystems vor, welches das Erstellen von Anwendungen ermöglichen soll, die autonome virtuelle Charaktere verwenden. Bisher existierten keine Modelle davon, wie Autoren auch Details des Verhaltens von autonomen virtuellen Charakteren bestimmen können. Es wird allgemein angenommen, dass autonome virtuelle Charaktere bestenfalls abstrakt parametrisierbar sind. Das hier vorgestellte Modell zeigt demgegenüber, wie das Verhalten von autonomen virtuellen Charakteren, trotz ihrer Autonomie, präzise durch Autoren kontrolliert werden kann. Das Modell basiert auf dem Konzept einer Menge von „gebilligten Geschichten“, und beinhaltet zusätzlich Komponenten für maschinelles Lernen.

Abstract

Within the research area of “interactive narration”, the production process constitutes a decisive bottleneck. Currently this bottleneck constricts the progress of the field, since the peculiarities of the research area demand

that applications must be actually built in order to concretize ideas and to test the technologies. Purely theoretical considerations are not sufficient, given that the requirements can only be clarified during the production process.

Methods and software environments have been lacking that could be employed to rapidly implement and to iteratively enhance applications related to interactive narration, so that this iterative process would encompass the authoring tools, the application design ideas, and the requisite technologies.

Taking this need as a point of departure, the work presented here demonstrates the first generic software framework for the creation of interactive narratives with virtual characters, with a strong focus on visual authoring methods. The framework consists of a complete functional software environment that has already proven its efficacy and generality within a variety of projects.

As basis of this framework, a novel hybrid visual formalism was developed. It enables various ways of combining directed graphs and of algorithmic control methods. With this combination, the development team can employ the directed graph as a main formalism that can be put to use immediately, and add other control mechanisms partially and incrementally as required during the progressing implementation of the application ideas. Using the directed graph, which is fairly easy to manage, as the basic structure, it is possible to guarantee that the additional technologies employed respond precisely to the requirements of the application and of the content.

In this way, the framework described here enables computer scientists, domain experts and content creators to work together to create novel, often experimental applications. It is particularly suited to systems which rely on a set of fixed elements or of fixed templates, and where the creation effort is centered on the question of how to determine the sequence of basic elements.

A series of novel design ideas for narrative applications is also presented in this dissertation, together with the basic patterns for employing the authoring framework to implement them; this demonstrates its aptness for the creation task, and also concretizes certain hypotheses that I have developed on why and how interactivity can be combined with narration.

Further on, this dissertation presents a model of an authoring framework that shall allow for concrete authoring of applications that rely on autonomous virtual characters. No alternative models currently exist on how authors could control even the details of the behaviors of autonomous virtual characters. It is assumed that an autonomous virtual character can be at most parameterized abstractly. The model presented here shows how autonomous virtual characters can be precisely authored, in spite of their autonomy. It is based on the concept of a set of “endorsed stories”, and also contains components for learning algorithms.

1 Introduction and Overview

Stories pertain to those elements of life that can go unnoticed exactly because they are so utterly elemental. Stories as common history build up cultures and nations (e.g. through religious or historical stories), provide reasons for wars and coalitions (stories of betrayal and solidarity); stories as told biographies and self-presentations are an essential part of our social and psychological identities; stories as examples are a major form of teaching, learning, and transmitting knowledge; stories in movies, television, and computer games are one of the major industries and leisure activities of our time. Stories' importance in the constitution of our existence lays not much behind food, sleeping, or physical health.

Interactive stories could, in a near future, become as important as linear stories, paralleling the enormous growth of the divulgation and weight of computer games in recent years. The author of this dissertation argues that a major branch of forthcoming interactive stories will deal with psychological facts of our life ([Iurgel03a], Section 3). They will intensify our psychological and social understanding, both of others, and of ourselves. As a specific kind of simulations of the fabrics of groups and of emotionally touching situations, they will provide the emotional, artful intensity of traditional stories, combined with the possibility to interactively examine various aspects of the problems and constellations. The author of the present work has elaborated a hypothesis on how psychology is a decisive link between interactivity and narration, and on why the artfulness and typical validity of the presentation form will count for the difference between a simulation and an interactive story (cf. Section 2 and 3, and [Iurgel03b]). Interactive narration would thus emerge as a compelling, immersive, artful simulation. But there is still some way to go, and more research is needed, until a fully satisfactory understanding of the relationships between simulation, art, psychology, and narrative presentation is reached.

Psychologically intense interactive narrations can also be employed within applications with different specific goals, ranging from presentation, education, to shopping (cf. Sections 6.1, 6.2, 6.3 respectively; [Iurgel04b], [IurgelMarcos06], [Iurgel05a], [Iurgel05b]). Utterly new experiences, much more motivating, convincing, or entertaining will shortly become possible, integrated within such different application domains.

Why are such thrilling applications not yet there on the market, and even barely visible within the research community? In fact, the precise design of interactive narration applications, i.e. the “rules of the game”, the interaction modes, the required actions and reactions of the virtual characters and of the environment, the expressive requirements, the required properties of the content – are not yet sufficiently clear. Much more clarity on the properties of the interactive narration application must be acquired (cf. Section 2 and 3) for the emergence of successful interactive narration applications.

The only way of attaining this missing clarity is by creating exemplary applications, until initial vague experience design ideas become clear and accepted, or refuted and replaced by better approaches. But here, the research area of interactive storytelling, which is essentially interdisciplinary and involves not only technology and sciences, but also creativity and even art, is faced with a decisive bottleneck: There is no tool, software environment, and methodology available for rapidly creating novel interactive narration applications, and for experimenting with content, experience design, and technologies.

The current, actual production process for experimental interactive narrative applications requires enormous investments on time and efforts. For example, the seminal work of Mateas and Stern, “Façade”, has taken seven years from the first publication [MateasStern00] to a release of a publicly available functional application – arguably the very single interactive narrative that was ever made available to the general public. After several years of research and development, Szilas is still working towards a first functional prototype of his very promising approach[1]; the famous and successful game developer and theorist Chris Crawford has been working at least for ten years[2] towards an authoring framework that shall later en-

[1] According to personal communication during the “Red Cap” workshop on authoring for interactive drama, cf. [SpierlingIurgel06]. Cf. also [Szilas99].

[2] Cf. www.erasmatazz.com/library/JCGD_Volume_6/Dragon_Speech.html, and personal communication.

able the creation of first interactive narratives, without publicly known functional productions.

These examples demonstrate that the production process of interactive narrative is a major problem that is impeding progress. In this dissertation, I will present the first generic software framework with focus on authoring methods that fosters a rapid and agile development of interactive stories. This framework greatly facilitates the development of novel applications that test design ideas related to interactive drama, and the corresponding development of technologies.

The authoring framework developed in this dissertation is called “Cyranus”. Its main theoretical innovation and basis consists of a new visual formalism. This formalism, the “Hybrid Control Formalism”, responds to problems of determining sequences, and of the integration of generative modules with approaches that rely on action selection of basic units. Determining sequences is a major quest of interactive narration research (cf. [CavazzaPizzi]). The question of determining sequences is the task of organizing scenes, acts, beats, utterances, and other elements of a story in a way that, in spite of interaction, the combination of these elements results in a compelling story.

The usual way of approaching a novel design idea on interactive stories, as far as the question of determining an appropriate sequence is concerned, consists in the creation of a complex architecture with complex algorithms that are both difficult to manage (cf. for example [MateasStern00], [Szilas03], [Iurgel02b]). Thus, from scratch on, the software design is complex and very difficult to author, whereas at the same time the application design is not clarified in the details. The result of this all-too ambitious start is the forbiddingly time consuming, complex, and uncertain development process mentioned above.

This dissertation proposes and enables a different approach, which focuses on an agile, iterative production process with short iteration cycles, authoring methodologies, early testing of design ideas, and team work.

A standard way of visually authoring reactive systems other than interactive stories consists in employing a directed graph that links a finite set of states. As many researchers in the area of interactive stories correctly maintain, authoring with directed graphs is a restricted methodology ([Crawford], [RiedlYoung]). Explosion of complexity can result in directed graph control structures becoming unmanageable, and then other algorithms for determining sequences are needed, e.g. planning, generative grammars, or tailored algorithms.

But since at the beginning of a production process of interactive stories the application design is usually vague and tentative, and hence the technological and authoring requirements are not clear at the beginning, starting the development process with uncertain technologies that are difficult to adapt and to author leads exactly to the development problems typical of this research area.

The solution to this production dilemma presented in this dissertation lies in enabling an immediate start of the production process employing directed graphs, which are usually easy to use and to adapt. The migration to other control algorithms can occur lazily, not before the requirements for the algorithms and for the application design become apparent and clear. The hybrid visual formalism developed in Section 4 (cf. also [Iurgel03d], [Iurgel04a], [Iurgel05b], [Iurgel06a]) enables this migration to different algorithms whenever the need arises and to the extent required by the actual application development.

A core property of the hybrid visual formalism is that it offers all of the advantages of Harel's Statecharts (cf. Section 4), but employs the composite states of Statecharts to define ways of integrating other control strategies. For example, a distinct control strategy *C* can be attached to a composite state of the Statechart, and it can be employed to activate inner states of the composite state, without explicit transitions. This "transitionless activation[3]" can be combined with the control of the Statechart, enabling the local choice of the most appropriate control strategy, and also enabling migration from an initial Statechart-based design to a different, more powerful control strategy, when necessary. In a similar manner, not only "transi-

[3] In the sense of an activation that does not presuppose the traversing of the edge of a predefined graph of a transition network representation

tionless activations" of visually represented states can be employed, but also calls to modules that output actions without first activating a state is possible. This extends the capabilities of the hybrid formalism, and also allows for the creation of consecutive control layers.

Thus, the formalism enables control algorithms to be employed partially, i.e. the algorithms are used to control only an exactly specified, limited subset of the elements that shall be selected; it enables different algorithms to be modularly combined in different ways with each other and with directed graphs, within well defined processes; it allows the authors and developers to employ layers of reactions, such that algorithmic control strategies can assume the control in case that the directed graph fails, thereby enabling the development team to maintain a balance between generic responses and precise reactions; finally, it also allows integration of arbitrary generative modules that do not make use of the sequence choice formalism, by defining the processes and occasions that will call these external modules. With these measures, the Hybrid Control Formalism fosters rapid, visual, gradual, iterative, controlled, team oriented development. With the hybrid visual formalism presented here the content creation team can, for the first time, start right away, without final commitment to any sequencing algorithm or vague interaction design idea, coping with problems only when they actually emerge.

Thus, the Hybrid Control Formalism allows to break the deadlock found in the research area of interactive narrations: The deadlock stems from the fact that it is extremely difficult to experiment with novel ideas, because no appropriate authoring methods and technologies exist; but the authoring methods and technologies are not available due to the fact that the ideas are too innovative and experimental. With the novel formalism presented in this dissertation, the authoring and creation process can start right away, employing acknowledged technologies and cautiously testing and lazily introducing new concepts and methods.

The Hybrid Control Formalism forms the theoretical basis of the formalism. The software engineering framework, i.e. the authoring framework per se, "Cyranus", implements the hybrid formalism. Besides containing this implementation, Cyranus is a complete software framework for creating interactive stories developed by the author of this dissertation. Various storage and communication formats were devised; Cyranus contains as main parts a complex authoring tool; a modular runtime control environment; viewers for 3D and 2D; handlers for various input devices (gesture recognition, text input, etc.); a client-server system for online, multiuser applications; emotionally expressive virtual character animation; virtual environment control modules; integration of TTS for lip-synchronized, speech-enabled, virtual characters; ontology integration; implementation of a functional XML-programming language as part of the control facilities; and much more; cf. Section 5, [Iurgel06a], [Iurgel05c], [Iurgel04b], [Iurgel03d], [IurgelHoffmann06] for more details.

Both Cyranus and the underlying Hybrid Control Formalism are construed to be as general as possible. The only a priori assumptions about the applications to be created with this framework are (i) that action selection of basic units is a central method of handling interaction, i.e. that there is a possibly very large, but finite pool of allowed actions and reactions of the system, and that (ii) virtual characters are employed.

Cyranus has already proven its appropriateness for the task in the context of several interactive narration projects. It has been successfully employed by the author of this dissertation to create the applications of the following projects:

- **Art-e-Fact:** This EU-project has exemplified interaction design ideas of the author of this dissertation concerning presentation of themes form the humanities. The author contends that a discussion group of virtual characters is particularly appropriate for the task, and that stories are required to reveal "deep" personality layers of the virtual characters. This is important since knowledge concerning arts and humanities does not develop independently of the personality and life of the persons involved. (Cf. Section 6.1, [Iurgel02a], [Iurgel02b], [Iurgel03b], [Iurgel04b], [IurgelHoffmann04], [Iurgel05c], [SpierlingIurgel03]). The author has created the first version of Cyranus in the context of this project.

- **Ask & Answer:** This application implements some ideas on how to create edutainment applications with chatting virtual characters. The author contends that interactive narration as psychological revelations can be introduced into educational games by urging the students to support a virtual character in coping with personal problems, thus enhancing motivation and entertainment. He has created Ask & Answer in order to exemplify and study this and other hypotheses. Cf. Section 6.2, [IurgelZiegler], [Iurgel05a], [Iurgel05b], [HoffmannIurgel].

- **Virtual Human with Social Intelligence:** Within this project, commissioned by SAP AG, the author has created a virtual shopping assistant. As part of the project, he has devised his application design concepts on how to introduce entertaining narrative elements into e-commerce: By creating programmed entertaining conflicts between shopping assistant and user that reveal "deep feelings" of the virtual character. This is amusing and motivating, and shall foster identification with the shopping site. (Cf. [IurgelMarcos06], [Iurgel06b], [Iurgel06a], and Section 6.3)

- **Virtual Human.** This German funded project employs autonomous virtual characters to create simulations of a learning situation, and of a television show. The author has used his Cyranus technology within this project to create and control the sequence of scenes, and of other parts of the simulation. ([Iurgel06a], [Iurgel06c], [GöbelIurgel07], and Section 6.4). Departing from the experiences gained within this project, the author has also developed further models for authoring of interactive narratives (Cf. Section 7).

Other groups also have already borrowed the Cyranus authoring framework to implement novel, research oriented applications, e.g. [SpierlingMüller04], and the Cyranus framework already served as a basis for the INSCAPE and the U-CREATE R & D projects, which have both a slightly different focus[4].

The interaction and application design ideas of the first three projects listed above

[4] Cf. www.inscapers.com and www.zgdv.de/zgdv/zgdv/departments/z5/Z5Projects/u_create

(Art-e-Fact, Ask & Answer, Virtual Human with Social Intelligence) rely on the aforementioned hypothesis of the author that interactive narrative applications should incorporate psychology, and “deep” personality layers that are uncovered in the face of conflicts. These applications and their design provide examples of experimental applications that the research community will necessarily have to build (cf. above, cf. Section 2), in order to gain a more thorough understanding of the properties and requirements of interactive narration applications, and in order to clarify the interaction design ideas.

This dissertation also develops a model of an authoring framework for autonomous virtual characters. This model shows that some of the principles of Cyranus, i.e., focusing on rapid and agile authoring, and employing a generic framework that enables gradual growth of ideas and technologies, can also be used for the authoring of partially autonomous virtual characters within interactive narrative. Currently, various researchers claim (e.g. [Aylett05], [SobralMachado]) that autonomous virtual characters cannot be authored but very abstractly, e.g. by setting personality parameters. This would not be a satisfactory authoring method, because it generates exactly the production problems and deadlocks that have already proven to be a serious bottleneck to the progress of the field: It is a very unintuitive and complex task to program and model an autonomous virtual character, a task which only very specialized computer scientists can accomplish, and as long as the application design is not acknowledged and understood, the programming efforts may turn out to be futile.

The model developed in Section 7 on authoring autonomous virtual characters is called “Creactor”. Its core notions are also described in [IurgelMarcos07]. Creactor shows that, in principle, the following scenario is perfectly feasible: To employ autonomous actors while focusing on concrete interactions of the application, working agilely and iteratively within a team, and defining precisely, facing a concrete interaction example, what the autonomous agent shall do. Creactor is a top level view on future work, and the details of it and the theory required for the integration of Creactor and Cyranus, for applications that employ both decision on sequences and autonomy, is left for upcoming research.

This dissertation is organized as follows: Section 1 provides a general introduction and overview. Section 2 deals with the different aspects of current interactive narration research. Section 2.1 presents and discusses general concepts and intuitions related to interactive narration. Section 2.2 describes the main different approaches, both from the point of view of application design and technology; Sections 2.3, 2.4 and 2.5 continue the discussion of Section 2.1, on a more concrete level, in an attempt to understand the requirements for an authoring environment for interactive narrative. Section 2.6 is a general discussion of authoring methods and processes for interactive narration. Section 3 deals with the two fundamental theoretical areas from the humanities for interactive narrative research: drama theory (corresponding to the structural aspects of a story), and psychology (corresponding to the point of view of a story person). Thus, Section 3.1 deals with drama theory and with the question of how interaction can be reconciled with narration. Section 3.2 presents and discusses theories of emotion and personality, with focus on the question on whether standard theories are, in principle, sufficient for interactive narrative research. In Section 4, the Hybrid Visual Formalism will be developed, the theory to the novel approach to authoring interactive narration. Section 5 explains and presents the complete software environment Cyranus, which implements this theoretical basis. In Section 6, a series of innovative applications are presented, both from the point of view of application design and with respect to the question about how to employ the framework developed in Section 4 and 5 for developing these applications. Section 7 presents a model of an authoring framework for partially autonomous virtual characters ("Creactor"), which are not covered by the authoring methods addressed until then. The short final section contains the final conclusions.

Some notes on terminology: "Interactive narrative" will often be abbreviated with "IN", from now on. There is sometimes controversy within the IN community on whether the field should be called "interactive narrative", "interactive drama", "interactive storytelling", or something else. *Virtual characters* sometimes are denominated by "intelligent (embodied/virtual) agents", "virtual humans", and others. There is also no convention on whether the person employing an interactive narration application is a "user", "player" or "participant", and accordingly whether an IN application that aims at entertainment is also "game", or whether it belongs to a different category. The terms “interactive narration” or “interactive narrative” will be preferred, both employed as synonyms, “virtual character”, and “user” through this text, but other terms will also appear, for purely stylistic reasons, without any implication of different meaning.

A virtual character will be referred to by the pronoun “it”, even when sometimes a certain virtual character with a given name is being discussed, e. g. the anaphoric reference to “Fritz” will be “it”. All references to an “author”, “content creator”, “computer scientist” etc. are certainly meant to be gender neutral, but, for simplicity and readability, the pronoun will always be a simple grammatical “he”.

Sometimes, a first person personal pronoun is employed, when it may not be fully clear from the context that something is indeed the author’s achievement or opinion.

2 Interactive Narrative: Ideas, Concepts, Applications, and Experiences

2.1 What is "Interactive Narrative"? An Introduction to the Fundamental Intuitions

- In this section some basic intuitions underlying the appeal of the concept of interactive narrative are presented.

2.1.1 The Intuition behind the Fascination of Combining Interaction with Narration

"Interactive storytelling", "interactive drama", or "interactive narration" are concepts that are of crucial importance to the life of everyone. It is clear that important novel applications could be created if some of the fascination that stories evoke could be transferred and reused for interactive applications. Thus, the quest for "interactive narration" implies rather a question about what could be learnt from traditional and received ways of telling stories, in order to devise novel, enjoyable, enriching, and meaningful interactive applications. However, this is not a question that could have one definite answer, because there are certainly many things to learn and many ways of transposing the lessons into the interactive realm. As Mateas and Sengers [MateasSengers] have put it:

"NI is radically interdisciplinary, drawing on narrative concepts from many humanistic fields of study. Narrative is not a single entity or a single, tightly related set of concepts. As the term is used in humanistic discourse, narrative can mean many things."

Szilas [Szilas03], for example, has tried the following definition of "interactive drama":

"Interactive Drama is a drama on computer where the user is a character. Being a character means being able to perform any action on the fiction world that the other characters can perform."

Another definition stems from Laurel ([Laurel], cited after [Magerko]:

"An 'interactive drama', then, is a first-person experience within a fantasy world, in which the User may create, enact, and observe a character whose choices and actions affect the course of events just as they might in a play. The structure of the system proposed in the study utilizes a playwriting expert system that enables first person participation of the User in the development of the story or plot, and orchestrates system- controlled events and characters so as to move the action forward in a dramatically interesting way."

Those approximating definitions help understand the aims behind the attempt to create interactive drama: As Szilas describes it, being able to act in the same way as the virtual characters of an IN application could be an important feature; this expresses the wish to be part of a (social) virtual world, to be in that world, in an emphatic sense of being which includes a feeling of immersion and of sense. In Laurel's view, the experience of a fantasy word in the first-person, with an emphatic *I* that takes part in a play, is the essence of an IN application.

2.1.2 Discussion and Elaboration of the Intuitions

As tentative definitions are being developed in emerging fields while researchers are creating new conceptions of innovative applications, they cannot be very accurate. For example, the FearNot! system described below is generally regarded as an example of IN, even though it does not offer the same freedom of action to the user that the virtual characters possess, thus violating Szilas definition. And Laurel's definition leaves open what is exactly a "dramatic way" in which the course of actions would go on. In fact, the most mature attempts at IN do not offer a first-person perspective, in the sense that the player would be the protagonist of a story (cf. below).

The Star Trek idea of a Holodeck was since long recognized as an appropriate vision showing into which direction the quest for interactive stories should go [MurrayT]: In fact, a full-fledged, utopian IN system would provide us with the experience of being in a world, as responsible and free agents that act, react, influence and are being influenced. I believe that the major difference of this narrative Holodeck, compared to the real world, is that it would be, in a certain sense, better than life. In the narrative Holodeck, there is no boredom, senselessness, irrevocable decisions, time consuming routines, or pointless chores. Everything is interesting, rewarding, playful, enriching, meaningful, and full of suspense, as in a fairy tale (Figure 1).

Figure 1 – The narrative Holodeck would enable full emotional immersion into a simulated world, but, better-than-life, every event in this world would be interesting and touching[5].

For example, an educational narrative Holodeck about history could teleport us to the time of the French revolution, so that we could experience the events, understand the cultural background, the mentality of the people, their daily lives, the conflicts they had to live with, their religion, as well as ideological and political institutions and ideas. This Holodeck must necessarily be guided, since it would be pointless to land in France of the revolutionary time, after teleportation, and be forced to live an isolated life of a rejected foreigner, without involvement into the historical events and circumstances of the people. In the Holodeck, all virtual persons around us are always trying to provide us with the relevant experiences; they are actors rather than simulations of human beings, and their task is to involve us into stories, creating a meaningful experience. A hidden instance, the "narrator", "story engine", or "facilitator", as it is sometimes called, coordinates all the actions of the virtual actors and decides on the occurrence of contingent events, like a story person unexpectedly entering into our room, just in time to prevent some story dead end, or triggering a flood that washes away some bandits that would otherwise kill our story self.

[5] Source of image: http://images.amazon.com/images/G/01/dvd/cinderella-pumpkin-large.jpg

The narrative Holodeck is not only a technological quest. In fact, very few real-world starting points had been devised. A theatrical Holodeck, with real actors in a real setting, does not exist either. Some (real, not virtual) theatrical applications already exists that come close to this vision, such as improvisational theater, participatory theater, theater therapy, and detective story and role playing games (e.g. [Johnstone], [Louchart], [Moreno]). But in general, we are currently not able to instruct and train the real actors for the narrative Holodeck, and to instruct the "player", so as to ensure the desired narrative effects, at least not without giving those actors the time to develop their own specific methodology and to gain practical experiences in a concrete situation.

Arguably, there is no fully executed theatrical Holodeck yet because it is too expensive; it inverts the relationship of the amount of actors to spectators/players. But in an electronic setting, specific maintenance costs do not exist; what counts are only developing costs. Maybe even that exactly the technological particularities of a virtual narrative Holodeck would facilitate developing the missing concepts, because it is natural and necessary to limit the action range of the player, in the virtual world. Thus, it is easier to predict and control his actions. For example, it is possible not to enable the player to take any object from the table, which could be used to kill the king, when killing the king would change the narrative and historical preconditions.

2.1.3 Preliminary Summary

In the previous two sections, the starting point and fundamental intuitions behind the concept of "interactive narrative" were presented. The following sections aim at further clarification of the concepts and technologies. In particular, a definition of "interactive narrative", and later a hypothesis on how to combine interactivity with narration will be developed (Section 3).

2.2 Concrete Interactive Narrative Concepts and Applications

- In this section, actual IN applications are examined, and a definition of IN is presented.
- Departing from problems related to application design, some preliminary conclusions as to the required properties of an authoring framework for IN are presented.

2.2.1 Core Concepts of Interactive Narrative

Early in IN research, two main types of IN were distinguished, the plot centered and the emergent narratives [Louchart]. The plot centered approach attempts to maintain a story line, in spite of interactivity. This attempt is one of the major dilemmas of interactive narrative. Figure 2 shows this and other sources of dilemmas: For example, authoring is a dilemma when partially autonomous virtual characters are employed, because "autonomy" is difficult to reconcile with "authoring"; the believability of an interactive, personality-rich virtual character is likely to diminish if it is forced to follow a story line

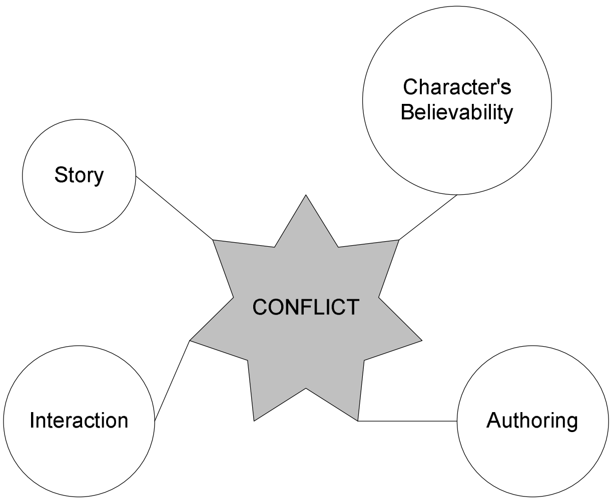


Figure 2 – The dilemma of interactive narration stems from different conflicts between its main components interaction, story, virtual character's believability, and authoring.

The plot, within the plot centered approach, is usually controlled by a module that is referred to by several names, e.g. "Story Engine" or "Narrator". Whatever the action of the user, the resulting experience sums up to a plot with a dramatic arc. The problem of this approach lies in maintaining a coherent and interesting story line in spite of the interventions [Clarke], cf. Figure 3. Some authors deny that this is at all possible, given that the freedom of action of the player would require heavy computation and variability of the possible story lines, e.g. [Glassner]. In fact, this is a crucial problem for IN: How variable should the plot be, how much of the plot can be computed – or should there be a plot at all? Façade [MateasStern06] and Geist [GrasbonBraun] are two prominent representatives of approaches with strong focus on plot coherence.

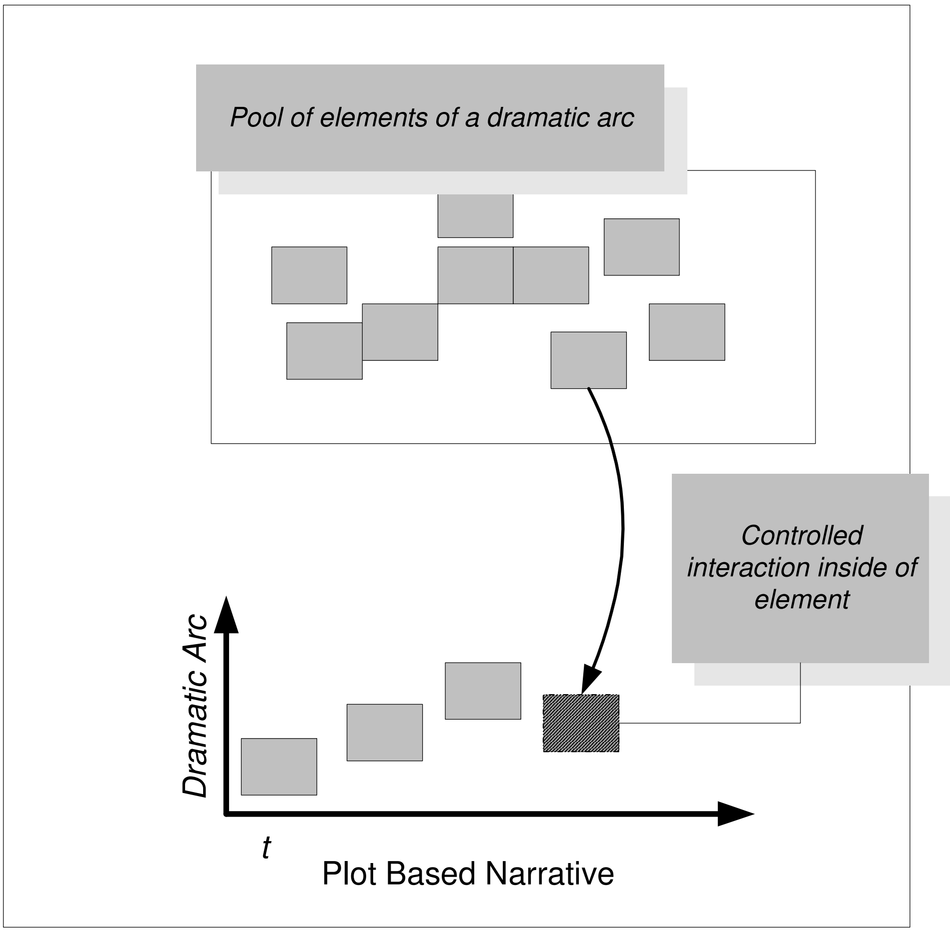


Figure 3 – Plot Based Narrative assumes that a dramatic arc can be constructed, in spite of interactivity. The interaction itself must be guided and controlled, in order to maintain the main arc.

Every major IN architecture contains a component responsible for the sequence of basic units (Cf. [CavazzaPizzi]). The principles of determining the sequences can follow rules of drama theory, training goals, or other, and the largest units are scenes, beats, utterances, or other basic elements. Thus, a "sequence selection approach" is a representative of those approaches that focus technologically and rely conceptually on determining the sequence of a set of more or less fixed basic elements. This emphasis on sequence selection is usually linked to a design concept that attempts at maintaining a plot structure.

The other main type of IN, at the opposite side of the scale, is the emergent approach. In this approach, there is no plot computation. Instead of the plot, the dramatic situation is on focus. The idea is that it is possible to construct such situations with virtual characters involved so that almost inevitably something dramaturgically interesting will occur, e.g. a situation where goals, roles and personalities of the dramatis personae, together with the background story and the settings, will almost certainly lead to a conflict and coping attempts (Figure 4). The emergent approach, in its pure form, is much closer to a simulation of reality than the plot based approach. It is much less "storytelling", in the sense that no elaborate dramatic arc is involved, which would only emerge as a matter of chance or with a certain probability. The major challenge of the emergent approach is the uncertainty about what will happen, and the difficulties of imposing some interesting course of action by an author. It is possible that a certain situation was intended to generate interesting interactions, but in fact the actual outcome turn outs to be tedious. The FeatNot! [Aylett05], [Aylett06] application is an example for the emergent approach.

The dichotomy between plot centered and emergent INs is still useful, as a global view on the field, in spite of certain recent attempts not being clearly on either side, e.g. [Machado], [Riedl03]. In front of the dichotomy, the problem of authorship becomes evident, and it is present in every known attempt: How possibly can a content creator invent an interesting IN if everything were delegated to algorithms and models? The eminent threat is that, even if the principles of IN soon should attain more technical maturity, the experienced results will be exactly "algorithmic", that is unexciting, predictable, and shallow. The place for the artistic genius, taste and intuition might get lost, on which linear storytelling certainly depends, and which is equally indispensable for the creation of INs.

The next sections will present two important current research oriented application concepts and their technologies in more detail, and relate other approaches to them. An assessment will follow. The design of these applications will be presented and discussed with some accuracy in order to examine the problems that are still associated to the application concepts of INs.

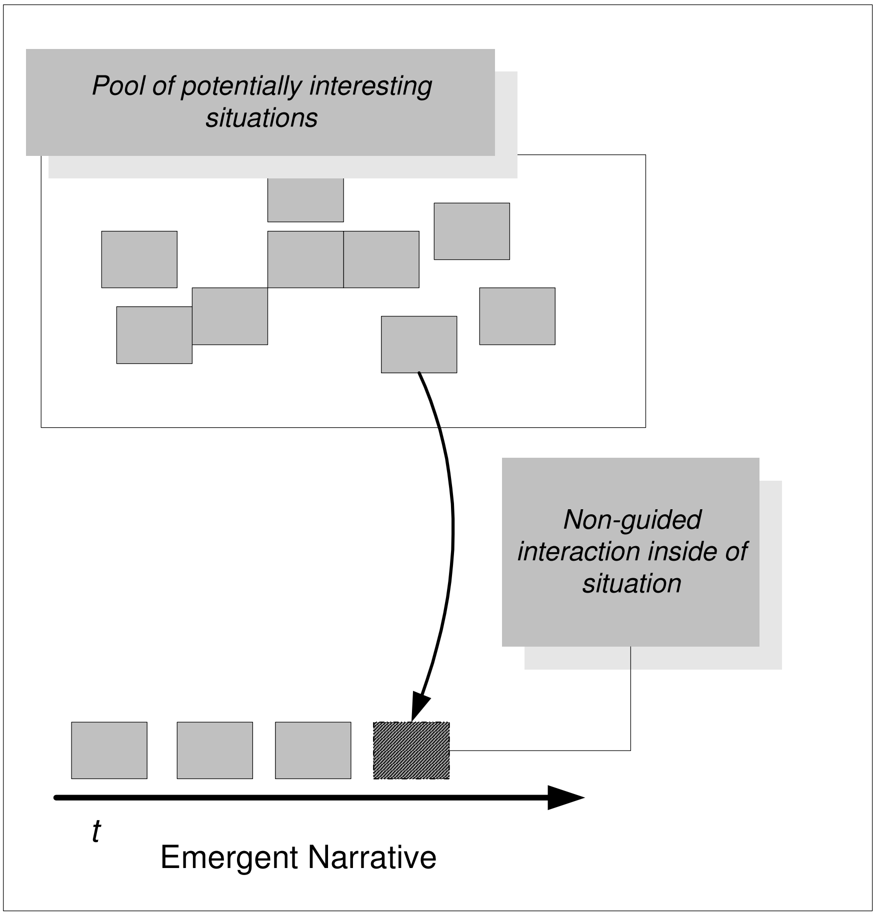


Figure 4 – Emergent Narrative assumes that suitable initial situations will combine into an interesting interaction, but without elaborate dramatic arc.

The discussion of technological issues of the approaches is deliberately kept separated. The choice criteria for these both approaches as representatives are theoretical and scientific relevance and innovativeness, but also a certain mature implementation level of the prototypes, i.e., they are more than general ideas and have undergone an authoring process on which the authors have already published.

2.2.2 Façade – Plot Centered Interactive Narrative

2.2.2.1 Application Design

The most influential work with strong focus on plot based interactive narration is Mateas' and Stern's Façade [MateasStern06]. Their system is not a pure plot centered approach, as their autonomous actors posses technologically a certain degree of autonomy. Façade is presented here as a representative of a system with focus on plot not only because of the technological solutions, but because the application design concepts strongly rely on a traditional narrative arc. The inspiration of the creators on linear storytelling is clearly visible, since this application is an adaptation of E. Albee's theatre play "Who is Afraid of Virginia Woolf?" [Albee]. The following is an accurate synopsis:

"In the play, Martha and George, a bitter erudite couple, invite a new professor and his wife to their house after a party. There they continue drinking and engage in relentless, scathing verbal and sometimes physical abuse in front of them. Martha is the daughter of the president of the university where George works as a history professor. Nick is a biology professor (who Martha insists teaches math) and Honey is his mousy, brandy-abusing wife. (…) Nick and Honey are simultaneously fascinated and embarrassed, and stay even though the abuse turns periodically towards them as well.[6]"

This play is particular apt for a procedure of "interactivating a linear story" [Iurgel03a]: Transform a linear story into an interactive version by assigning supportive roles to the user. The play does not possess a tight plot, but is rather a loose sequence of beats that increasingly involve the visiting couple, and by consequence the spectator, into the drama of a failing marriage. There is not much action besides dialogues.

The dramatic role of the player of Façade is not easily described by the common narration terminology. The user exchanges Nick and Honey of the original play; he is certainly not the main person, as the drama is only influenced by his actions, and would proceed without any action at all. He is a secondary person who happens to witness a marriage quarrel, and who is able to influence and is urged to take sides.

[6] From Wikipedia entry on "Who is Afraid of Virginia Woolf, at 5/31/07

Figure 5 – Screenshot of Façade[7]

According to Mateas and Stern, there is no explicit goal for the user [MateasStern03]. He is expected to investigate his possible influence, and the different courses the story takes, depending on his actions. His main influence mode is through typing in short English sentences, i.e., entering into dialogues with the couple, though he can walk within an apartment, and take and use a few objects (Figure 5).

Whatever the actions of the user, the resulting story will follow an Aristotelian [Aristotle] dramatic arc, arriving at a turning point of conflict, and will then come to a definite end: One of the three participants (consisting of the couple and the user) will leave the apartment, and the marriage will either be saved or lost.

Façade is a seminal work since it is the very first fully playable application of its kind. Assessment is still nevertheless quite difficult, because no systematic evaluation is available, and no impact on the market was intended.

The relation of dialogue and plot is problematic, which is an eminent problem of plot focused approaches: The user is required to enter into personal, intimate di-

[7] Source: http://blog.game-play.org.uk/files/Façade.jpg

alogues with the virtual characters, but these dialogues are interrupted before gaining any depth, in order to ensure the pacing and coherence of the ongoing plot. This violates the coherence of the dialogues, which is sacrificed in favor of the plot.
It is remarkable that Façade, with its strong emphasis on IN, so much resembles a crisis intervention simulation, and even a marriage therapy session. Mateas and Stern did not borrow from the psychological research of this kind, which could have been elucidating, though they occasionally describe a sequence of the play as "therapy game":

"The second part of the story is organized around the therapy game, where the player is (purposefully or not) potentially increasing each character's degree of self-realization about their own problems, represented internally as a series of counters."[MateasStern05b]

These therapeutic elements necessarily have to remain superficial, due to the necessity of maintaining the plot, following the plot centered approach. In fact, these therapeutic elements are almost totally absent in Albee's play, where the visiting couple is rather forced to take part in sometimes cruel games between the two main persons. In Section 3.1, I will argue that this correlation of IN and therapy is quite natural, and that is would be beneficial to pay more attention to the application concepts from this point of view.

Causality concepts also deserve some attention. Façade is a one room, single scene IN, were narrated time and narrative time are identical, i.e., there are no time gaps. This is natural, given the experimental character of the application and the enormous efforts required in order to create it. However, this forces demonstration of agency, i.e. showing the consequences of the player's actions, into a narrow time frame, where normally psychologically slowly grown and complex problems such as a decaying marriage aren't normally solved or much influenced in short time frames. This probably deteriorates the notion of agency felt by the player. In Albee's original play, the marriage conflicts are certainly not solved in story time. However, Mateas and Stern compel an arrangement where there are clear dramatic endings, with a sort of solution to the conflict ((i) one of the participants leaving the scene; (ii) the marriage being broken, in case the leaving person is one of the virtual characters, and (iii) the marriage being saved). This responds for the necessity of providing a sense of responsibility for the outcome, but also fosters “illogical” behavior. Possibly, employing different scenes, where some time has passed in between and important non-interactive events are narrated, could be an appropriate measure to increase the experience of believable agency.

Another open conceptual question concerns the use of natural language as main interaction mode. The natural language processing of Façade is very imperfect and full of flaws, but nothing else is possible with currently available technologies. The question is still open whether end users, on the long run, would accept this fallibility.

2.2.2.2 Technology

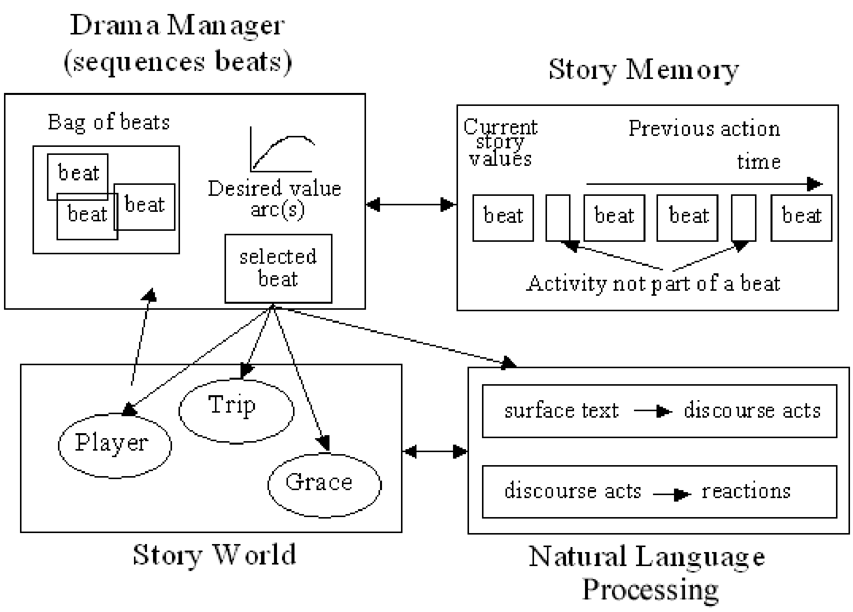


Figure 6 – The architecture of Façade (Source: [MateasStern06]).

Figure 6 shows the software architecture of the Façade system [MateasStern06], which is technically a sequence selection approach, in its main parts. Façade possess two very distinguishing technological features: The extensive use of shallow natural language processing (NLP), and of structured beats. Beats are chosen such as to build a dramatic arc. A beat is a set of procedures to handle user input in the context of a dramatic situation, and to drive and create that situation. In general, each beat requires case-by-case complex programming. A beat possess a rich internal structure. It includes the commands for the virtual actors on how to act to perform a particular dramatic task, e.g. serving a drink and quarreling on this occasion, and on how to involve the user and to react to his input. Depending on the outcome of a beat, e.g. on which drink the user chooses, the story arc will be build up out of different beats. The virtual characters possess only limited autonomy related to the coordination of expressive goals, but they do not possess emotional or personality models that would allow them to decide on which action course to follow. Higher levels goals of the virtual characters are scripted within the beat. The creation of a beat is thus an intertwining of procedural programming and artistic expression.

Natural language processing within Façade is "shallow" [MateasStern04], which means that it does not employ any "deep" semantic or syntactic transformations of the surface texts. The handling of natural language is reduced to a sequence selection problem, because no text is generated; every possible utterance of the virtual characters is pre-recorded by human speakers. The input text is mapped onto a set of speech acts, and the beats react to the speech acts. NLP inherits, in this case, the "wickedness" of the IN field (cf. below): it is not really clear at the beginning of the design process what the exact requirement for the dialogue management is going to be. But since none of currently available dialogue management and natural language processing technology is able to model "psychologically deep" conversations and linguistically expressive utterances, cf. [Bernsen], this emphasis on shallowness, i.e. on ad hoc solutions tailored for the particular application design, is inevitable.

2.2.3 FearNot! – Emergent Interactive Narrative

2.2.3.1 Application Design

FeatNot! is an application developed within the recently concluded European VICTEC project, cf. [Aylett06]. It is an example for the emergent approach to IN. It does not offer a story experience in the very prototypical sense of the word, according to which a story should contain some dramatic arc with a beginning, a climax and an end. This absence is typical for the emergent approach. It is, more than a story that contains interaction, a specific kind of simulation.

In FearNot!, school children can playfully examine different aspects of bullying. A certain virtual protagonist is the victim of bullying, and it asks the user for advice. Through simple typed sentences, the student can tell the victim what to do in a bullying situation, e.g. hitting back or telling the teacher. This counseling takes place during special short scenes, where the virtual victim approaches the user directly, asking what to do [PaivaDias]. The virtual characters are autonomous, in the sense that they decide on whether to accept the advice and on the choice of action, in concrete situations. After a counseling session, a non-interactive scene is showed to the students that shall demonstrate the consequences of the decision of the virtual victim (Figure 7). This non-interactive scene is chosen as to probably exemplify interesting and relevant consequences. It is possible, though, that nothing remarkable will occur during a particular scene that is presented, because of the autonomy of the virtual agents, which makes it impossible to the system to foresee exactly what will happen [SobralMachado].

Figure 7 – Screenshot of FearNot![8]

FearNot! is a training system for learning about the possible choices in such situa-

[8] Source: http://www.macs.hw.ac.uk/EcircusWeb/images/Picture-resource/FearNot/Figure2-2.gif

tions of conflict, and about the possible consequences of particular choices, in dependence of the social constellations, and the personality traits of the social actors involved.

The principles of FearNot! are easier to assess than those of Façade, since FearNot! is a learning application. The principle of having a playful demonstration of causes, consequences and dependences is widely acknowledged as useful [Prensky], and the target group and intended use is clearly discernable. The natural language interaction has a different function from Façade; it is not used to enable complex dialogues, but only to choose from a limited set of options. Some multiple choice style user interfaces would also have been possible.

The possibly most questionable feature of FearNot!'s application concept is exactly the uncertainty stemming from the use of emergent technologies. From the point of view of the learning student, a scene where nothing happens is pointless. This also defies some essential intuitions on how to combine narrative and interaction – to offer a series of interrelated events that makes sense and is elucidating and possibly entertaining. It is an open question whether a concept where such scenes are not displayed at all wouldn't be more advantageous. In fact, the concept of emergent IN, implies that it may happen that no-occurrence scenes are be presented, whereas a potentially relevant scene is skipped by the system, because the system does not know what the virtual characters will do. In short, FearNot! also exemplifies the known problem of emergent narrative concepts: It is possible that nothing interesting will happen, which defies the very idea of "narration".

2.2.3.2 Technology

The FearNot! architecture is partially depicted in Figure 8, where the components necessary for the functioning of a virtual agent within a scene are shown. The Story Facilitator is not depicted. The Story Facilitator, equivalent to the Narration Engine of other approaches, chooses the scene, after the virtual victim has been given advice, trying to demonstrate the consequences of the advice. The virtual agent is fully autonomous within a scene. Its emotions are computed employing rules according to a version of the OCC model (cf. Section 3.2), and they determine reactive (immediate emotional) and deliberative behaviors. A BDI architecture (cf.

[RaoGeorge]] is employed for the deliberation and actions.

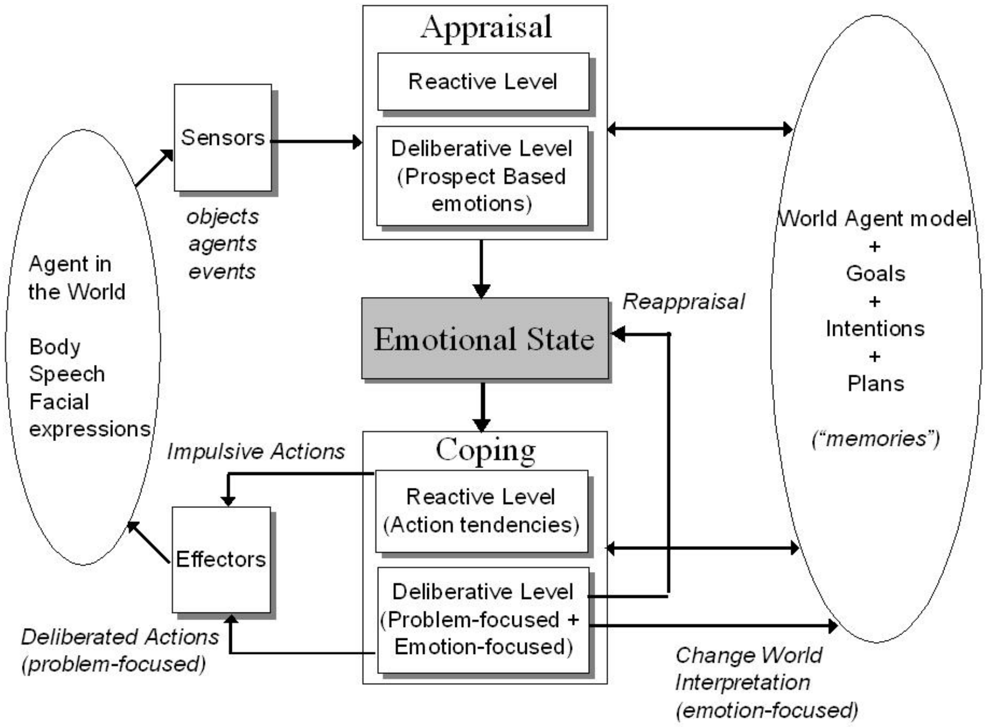


Figure 8 – The agent's architecture of FearNot! (Source: [Aylett05])

2.2.4 Other Applications in the Research Area of Interactive Narrative

Façade and FearNot! were presented in some detail because they are important examples for the two divergent approaches to IN, plot centered and emergent. Several other applications concepts exist, though, reflecting the divergent nature of the field of IN.

An example for an application design approach that can neither be clearly classified as emergent nor as plot centered is the work by Cavazza et al. [Cavazza-Charles]; their concept is character based, but not emergent in the sense that the virtual actors possess narrative goals, and thus the story line is, in a certain sense, part of their configuration. However, there is no drama manager ("story engine" etc.) to create a dramatic arc. The prototype is inspired by the sitcom "Friends". Four virtual characters act as protagonists of the sitcom, and the user takes an intermediary position between spectator and participant. He is allowed to interact at any time by employing or hiding certain story objects that possess a narrative meaning, and by means of spoken natural language. Employing natural language, the player gives advices, information, or tries to influence their moods. Other than with Façade, conversation is not on focus. Rather, the user changes the course of the narrative by means of his interaction with the virtual characters. This approach is oriented towards technological research, and the question about application design issues is thus difficult to address.

Another application that conforms to the plot centered paradigm was developed within the project "Geist" [GrasbonBraun]. In this prototype, visitors of the Heidelberg castle, wearing augmented reality equipment, were presented with a coherent story that conforms to Propp's [Propp] fairy tale structural analysis, no matter which walking path and which pace they took. This is a very specific setting, where the influence of the user on the story adaptation should be hid from him, and where thus the notion of agency, of the user influencing the narration, did not play any important role. On the contrary, his influence on the plot should go unnoticed. This approach therefore does not provide concepts for dealing with more intense user interaction.

Szilas [Szilas04] presented another approach with a strong bias towards plot. Here, the user chooses between varieties of possible actions, employing a specific user interface that offers more than a multiple-choice GUI, but less than natural language, and the story develops coherently, presented as text, depending on the user choices. This is still a formal approach on many levels, e.g. on narrative theory, formal control methods, and software architecture, and thus there is no primary concept for a certain concrete application that could be assessed.

Riedl et al. have presented a hybrid control that combines elements of emergent narrative and of centralized plot control [RiedlStern], an approach that the authors call "mixed simulation control". The authors describe a military training scenario, "IN-TALE", where the user plays the role of a Captain in the U.S. Army, in a foreign country, responsible for the security of a marketplace. The interaction of the user with the virtual persons of the marketplace is basically a simulation without narrative intentions. Narration enters the scene through challenges posed to the user. The system takes care that the challenges will plausibly be posed, in spite of the user, through his interactions, possibly endangering their preconditions. For example, an attempt at a terrorist attack will be carried through even when the user puts the supposed bomber into prison. In this case, the system could name another story person to continue the insurgence; here, too, the research is, to a large extent, technology oriented.

However, the vast majority of existing approaches with some interactive storytelling elements have been developed not in academia, but in the gaming industry. In the gaming industry, pure simulations without narrative elements employing specific algorithms and rules, close to the emergent approach of IN, are very successful, e.g. the Sims. Games with strong story elements, in contrast, usually posses just a few – or only a single – story lines, and the player experiences the story by solving quests, and by advancing from level to level. In fact, the concept of level can be regarded as a specific kind of scene full of interactivity, but the user has only very little influence on the scene that follows.

2.3 Discussion – The Problems of Application Design of Interactive Narrative

The previous presentation of IN application concepts and technologies has shown that current research in IN sees itself in a very specific situation that distinguishes it from other areas of computer science: the research on new technologies and the development of novel and compelling application ideas go hand in hand. There is no clear requirement for a certain specific technology that would emerge from practical creation of interactive stories. No application exists yet that can be unanimously assessed for its strengths and weaknesses, or for which the benefits and the demand of a target audience were already known.

On the one hand, this is in fact a most challenging and fruitful situation, because of the potential for the development of really meaningful novel virtual world experiences that, can have a major impact on future entertainment, education, training, and even on our self-understanding and on our identities.

On the other hand, there are considerable pitfalls, because the novel application idea might turn out to be unsatisfactory, or the technology inadequate for the application's actual requirements, only after we have implemented it.

For example, it is not clear whether the principle of Façade is indeed promising, on the long run. Possibly, the shortages of agency, which are tightly coupled with the very idea of imposing a plot on the interaction, won't allow for the dissemination of Façade-like applications. Equally, it is not clear whether the no-occurrence scenes, which are implied by the emergence approach of FearNot!, will not pose a serious problem that requires further fundamental conceptual and technological measures.

2.4 The "Wickedness" of Interactive Narrative Research

The existence of problems of the application design of actual interactive narrative research does not say that the very ideas or endeavors associated with IN are wrong. Mateas and Stern have called the challenges associated with IN, employing a term introduced by [RittelWeber], "wicked problems", a view and term that will be adopted throughout this dissertation. A wicked problem is a problem with a specific feedback loop, such that the definition and understanding of the problem is continuously being changed by any attempt to find a solution to the original problem. A wicked problem is thus a problem where posing the question is part of the problem. According to Mateas, for

> "(…) a wicked problem such as game design, exploring design space consists of navigating the complex relationships and constraints among individual design features, while at the same discovering or inventing new features and approaches that expand the design space. (…) Theoretical and empirical analyses certainly provide the designer with useful approaches, techniques and vocabulary for thinking about the design problem. But such analyses can never be strongly normative. The only way to explore new regions of design space is to make things." [MateasStern05a]

The wickedness of IN implies an experimental, creative, open process of growth, in the course of which technologies, usage of technology, technological demands, application design ideas and understanding of the creation process depend on each other. Thus, even if some novel IN application design ideas seems clear to the inventors and feasible at the first conceptual phase, its conception is doomed to change during the production process, at least as long as no prior functional examples are available. In this sense, in which the design and the technologies "grow" during development, IN possesses aspects of art creation.

Thus, the hypothesis in this dissertation is that the wickedness of IN implies that technology, application concepts, and authoring methodologies must be developed hand-in-hand (Figure 9 and Figure 10), and that the fact that this interdependence has been neglected so far has been is a major obstacle for the development of interactive narratives. Thus, it is difficult to assess whether a certain technology will "solve the (authoring) problem", since the "problem" (the application concept) has not been well established in the first place, and its concretization depends on the technologies available for its realization. This is a vicious cycle, since without the authoring methodologies, it is not possible to try out and experiment – and without technologies, no authoring methodology can be developed.

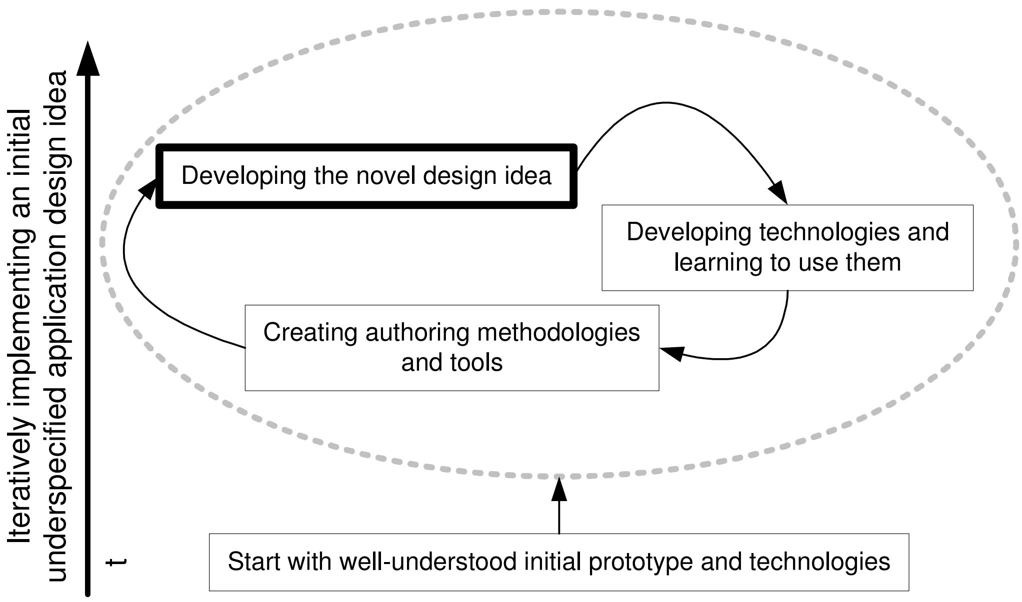


Figure 9 – Partially, the "wickedness" of the interactive narrative research stems from the interdependence of technologies, application design, and authoring issues, which grow during the creation process. The process needs to start with well-understood technologies, in order to be able to tackle with novel quests as they appear.

The other source of wickedness of IN, in my view, is the imminent attempt at transferring. Essentially, IN is the attempt to learn from non-digital narration and drama, and to transfer the lessons into novel interactive applications. This is a metaphorical process, where structures of something that is valid and understood are transferred into a realm that is promising but still problematic (Figure 11). As such, this process has already proven to be at the core of scientific inventiveness and cognitive progress (cf. [Fauconnier]). But, because it requires inventiveness and creativity, and multiple solutions may be valid, it also fosters the "wickedness" of the research.

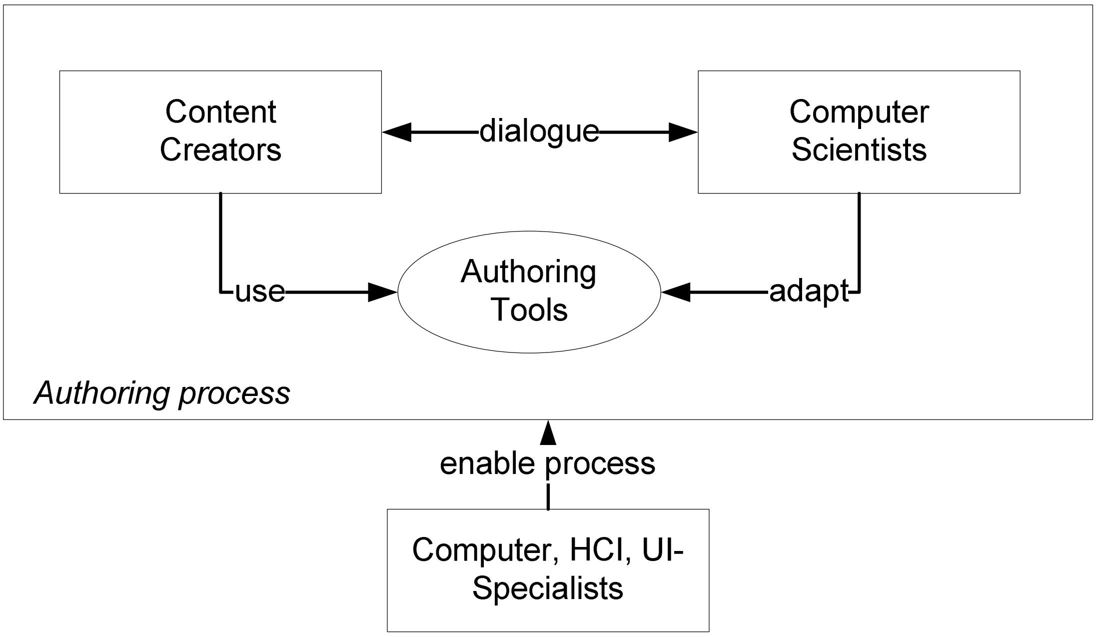


Figure 10 – The creative process of innovative, "wicked" applications with interactive narrative components should be centered on the authoring tools, and involves a team with team members fulfilling different roles.

Thus, a large variety of approaches are covered by the umbrella concept of "interactive narrative", and no unified theory of "interactive narrative" is available. The common umbrella also covers many essential differences. For example, the maintenance of a dramatic arc is essential for Façade and Geist, but irrelevant for FearNot! (cf. above). In this dissertation, "interactive narration" will be not defined in terms of features of a final application. The focus of the following definition lies more on the process behind the intuitions that belong to the research area. Thus, the following will be the definition employed within this dissertation:

'Interactive narrative' denominates a metaphorical process in which structures and experiences of traditional storytelling are transferred to devise novel, useful, entertaining, expressive, or meaningful interactive computer applications. As such, "in-

teractive narrative" is an open-ended process that can encompass divergent approaches and attempts.

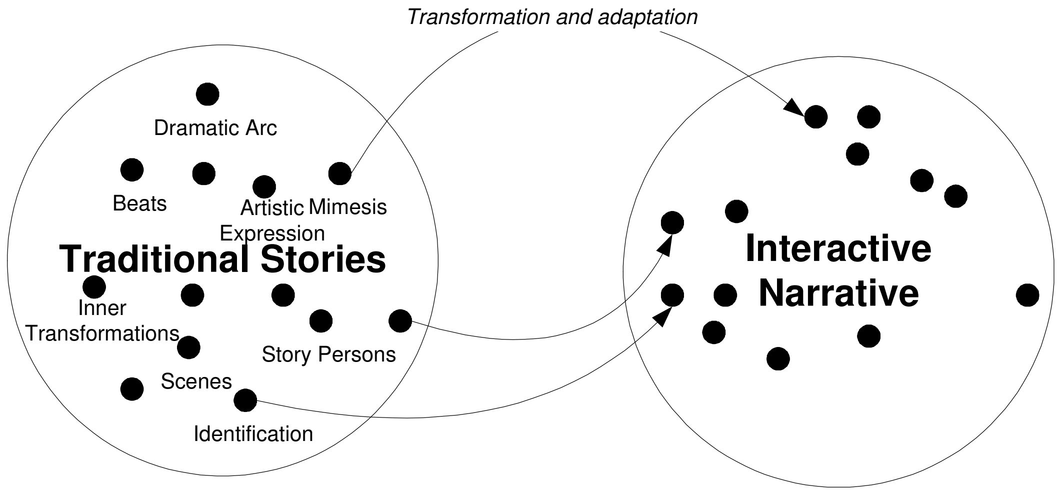


Figure 11 – The term "interactive narration" denominates an open-ended metaphorical transfer from the realm of traditional narration into the area of interactive applications. This is an additional source of "wickedness" of interactive narration research.

Despite all conceptual and technological difficulties, IN has reached certain maturity, in the past few years of research, as compared to initial rather visionary and tentative developments, little more than a decade ago (cf. [MurrayJ]) – in the sense that in fact the first fully functional applications are emerging, albeit on an experimental level. It is not yet clear enough which kind of application concepts are desirable and worthwhile pursuing, which technologies are appropriate to the demand, and which will reveal as dead ends. A main reason for this still missing clarity, for this "wickedness", is that we do not have functional examples because the implementation of any novel idea on IN, even if only as a partially functional prototype, is forbiddingly complex and time consuming. We cannot simply try out and iteratively let novel, tentative design ideas mature. Cyranus, the authoring framework that will be presented in Section 4 and 5, is a (partial) solution to this problem. It is a novel generic framework for experimenting with a large class of possible IN design concepts that responds to the "wicked" situation of IN research.

Consolidation implies that we need a clearer understanding of what IN can mean. Methodologically, this is not a "semantic" question, but a quest for an explicit and plausible exposition of application concepts, and for conceptual and prototypical examples of those applications. Thus, the application (or "experience") design has to gain more weight. The research area is still in need of a better understanding of what are a “good” application idea and a “good” experience design, and what is mistaken. A rapid, agile, iterative creation of application prototypes is required.

2.5 Conclusion – Many Aspects of Interactive Narrative are Still Questionable

The presentation and discussion of major concrete interactive narrative applications and technologies has lead to the following insights:

- **Application Design Problem.** Interactivity is not only, and possibly not even primarily, a technological problem. The concrete design of the applications, i.e. how to devise them to make them compelling and convincing, is an open question, as well.
- **Interdependence of Authoring, Technology and Application Design.** Interactive narrative is a “wicked” research area, in which authoring methodologies and tools, technologies, and application design, are highly interrelated and grow together during the experimental development of exemplary applications.
- **Importance of Sequence Selection.** Every major approach to interactive narrative relies on and requires sequence selection methods, i.e. employs a set of (more or less) fixed elements that are combined at runtime to form the desired user experience.
- **Interactivity Dilemma.** The dilemma of combining interactivity and coherent and dense narrative structures has is not yet been solved.

This dissertation adheres to the hypothesis that a proper response to the current “wicked” research situation consists in devising generic tools that facilitate the combined development of authoring, application design, and technologies. This dissertation will present the first generic tool that aims at facilitating exactly this joint development.

2.6 Authoring for IN

- In this section, previous research on authoring for IN is presented and discussed.
- A main conclusion is that more attention can and should be paid to acknowl-

edged non-programming authoring methods.

2.6.1 The Experiences of Other Researchers

This section presents a general discussion of the authoring process for IN. Note that no dedicated authoring tool for IN in general exists yet, and that thus the discussion must remain, to a certain degree, tentative. The discussion will focus on reports and considerations by the creators of Façade and FearNot!, cf. Section 2.2.

FearNot! represents an approach to authoring that has been designed to meet the needs of emergent narratives. Where Façade (cf. Section 2.2) relies much on the procedural intuition of the creators, and on their ability to solve problems as they appear and to implement creative ideas by programming, FearNot! employs general purpose technologies and established models. The author of FearNot! is on the one hand expected to parameterize and fine-tune these established models and, on the other hand, to devise an appropriate set of exemplary scene configurations (cf. [SobralMachado]). Hence, this approach follows the paradigm of separating strictly the processes of the system from its configuration through declaration and parameterization by employing more definite modules and models. Thus, authoring of FearNot! is normally a process of declaring and tuning, and of examining the consequences of this process. This approach to authoring is called by [Aylett05] "abstract authoring". This is also true of the approaches that lay their focus on planning technologies, e.g. [CavazzaCharles], [RiedelStern], [Mott], [Theune]. Figure 12 shows the process of abstract authoring.

Ruth Aylett [Aylett05] writes on the production process of FearNot!:

> "The author must think in terms of interactions between characters and the likely occurrence of actions, interactions and goal conflicts in the episodes being created. (…) This bottom-up approach can be a relatively complex exercise in finding the right balance between delimiting the boundaries of the episodes with their associated character definitions, and allowing the characters to take charge within episodes."

This approach, typical of emergent narrative, puts the emphasis on the declarative definition of the virtual character traits. The virtual characters are defined such as to act, in the relevant episodes, in the way that the author has in mind. This can be very difficult:

“The author is required to give up low-level control of the story and instead to develop much more detailed character specifications: the outcome of this process cannot be wholly assessed by inspection but requires simulation runs in order to develop adequate actions and goals or respond to specific needs for a scenario.”

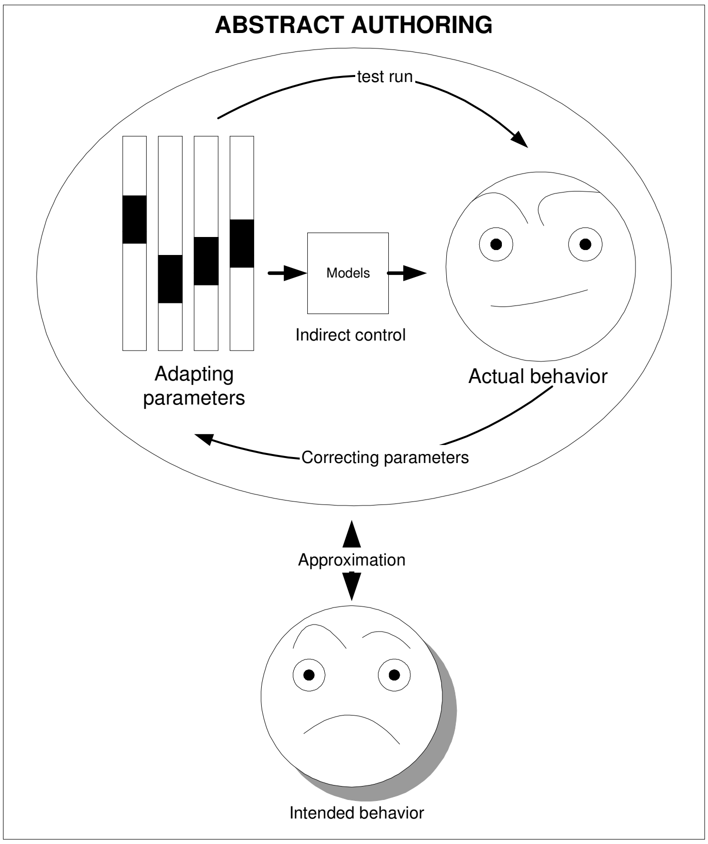


Figure 12 – Abstract authoring is a process of approximation of a certain intended behavior. It does not allow for procedural creativity, and requires a deep understanding of the system’s models.

Thus, the author is required to build a hypothesis about how a particular virtual character declaration or parameterization will lead to the desired behavior of a specific virtual personality within a specific set of scenes he has in mind.

As an authoring principle, this is problematic, since it requires a deep understanding of the system by the author in order to correctly define the virtual personality, and to assess whether the system’s models support the desired coverage. The author has some specific behavior in mind, and then he needs to find out which “screws” could possibly provoke the desired behavior. When one has to assume a large amount of parameters and of possible initial constellations, this can be very difficult, and eventually the desired result might turn out not to be feasible within the limits of the system.

Another exemplary attempt that promises simplification of the authoring process by reducing the abstraction level is provided by the system of [FaircloughCunningham], which uses the fairy tale structures of Propp [Propp] and adapts them to a MMORPG, employing CBR (Case-Based Reasoning, cf. [AamodtPlaza]). There, the particular exemplary stories are derived from Propp's analysis, i.e. they are not generated. The current story situation on the MMORPG is analyzed at runtime, and a similarity to each exemplary story is computed. Then, if possible, the story engine will employ the most similar story to the current state, assigning for example a certain task to a NPC (non-player character). Adaptation is made based on the story functions and similarity of roles, e.g. a "murder" can be assigned a task that originally the "villain" would assume, because a murder is similar in narrative function to a villain.

In Façade, the whole process of handling user actions and input, of selection of beats and of other elements is based to a large extent on intuition and resourcefulness of the creators [MateasStern05b]. The programming of Façade is, to a large extent, a difficult quasi-artistic endeavor. This lays much control of the behavior of the system into the hands of the creators, at the price of an exceedingly complex and time-consuming creation process. What is more, no visual authoring tools support this process.

The authors report on three man-years of content authoring efforts to create the prototype, apart from technological development [MateasStern05b]. Compared to a commercial game production, this is not much, as the authors rightly claim, but for a system that aims at the examination of novel ideas and of the appropriateness of technologies for these ideas, as targeted by this dissertation, this would not be appropriate.

2.6.2 The Increased Requisitions Posed to the Author

The hardships of authoring Façade stem largely from the fact that much programming is required. The system design imposes programming tasks onto any content creator that would employ it. But a technology that in principle could be employed to generate exciting, novel applications, but where creating content requires seldom genius, is not likely to succeed.

Mateas and Stern justify their peculiar approach with reference to a novel type of "procedural authorship" that emerges with the rise of highly interactive computer games, and especially with IN. Such as a novel writer requires language literacy, the new IN author would require "procedural literacy".

"By procedural literacy, we mean the ability to read and write processes, to engage procedural representation and aesthetics, to understand the interplay between the culturally-embedded practices of human meaning-making and technically-mediated processes. (…) In the extreme case of developing new modes of computational expression, authors must be highly proficient in the use of general purpose programming languages, used to construct new languages and tools specialized for the new representational mode." [MateasStern05c]

This is certainly true as far as the focus is laid on specific requirements for persons producing interactive works of art. In this line of thought, however, a consideration of the aspect of teamwork is missing. It seems that the isolated traditional writer was taken as a prototype for the author of IN. It is, however, not conceivable that, in IN, apart from experimental and academic contexts, the same person will have "high proficiency in using general purpose programming languages" and will "construct new languages" as well as new "tools specialized for the new mode", in addition to inventing and creating the content. As the example of successful game productions shows, large teams are required, with clear separation of roles, and a dedicated process to coordinate the work of these persons.

2.6.3 Discussion – Well Founded Formalisms and Methods versus Procedural Genius

Mateas' and Stern's allegations on the procedural nature of authoring for IN are too general; several methods for visually handling procedures with visual languages are known, and other non-visualizing processes are available that do not require programming, e.g. machine learning, optimization (cf. [Nelson]), and others. "Programming" is a very generic term. Possibly, some kind of programming is an appropriate authoring approach for certain problems. For example, standard authoring environments for interactive applications – e.g. Flash, Director – offer both visual support and an adapted programming interface.

But the problem of focusing on authoring begins when technological, "procedural", algorithmic brilliancy is required from the content creator, in order to invent solutions that will permit the author to put into practice his narrative ideas. These extreme demands are a reflection of the currently still very experimental, "wicked", and not fully mature stage of the field. The reference to programming as an artistic means is not precise enough, since it does not answer the question on how to perform certain tasks with a given system, and what the system can effectively achieve.

Thus, when the problem of sequence selection is concerned, the main challenge is to simplify and specify the creation process. Simplification shall enable trained and skilled content creators to take full profit of the system, without demanding full training as computer scientists, especially without requiring advanced programming skills. Sections 4 to 7 will show that this is possible without reducing the generality and expressive power of a sequence selection system.

A different problematic aspect of the authoring process becomes evident in light of the configuration process of the emergent approach. The author has to configure the system until it behaves as he desires, and covers the intended cases. For this, he requires a thorough understanding of the system, a clear vision of the cases he wants to achieve a certain behavior for. This will usually be a most difficult approximating task. Given the knowledge acquisition problem mentioned in Section 3.2, it might also easily turn out that no solution is possible, because the employed models are not adequate for producing the intuited behaviors. Section 7 will dwell on possible solutions and alleviations of this problem.

2.6.4 Definitions of Content Creation and Concrete versus Abstract Authoring

"Content creation" shall be defined as the process where a skilled and trained content creator only works on adding content, and does not enter into any scientifically or conceptually open technological quests. Thus, the content creator is indeed the person responsible for content, employing technology and possibly even sometimes programming, but not entering into technological innovations, and with clear focus on domain knowledge or artistic intuition, and not on programming. Trivial as this role may appear at first glance, it is questionable whether Façade and FearNot!, for example, allow for a content author to work with the system, on the behavior control parts. In Façade, procedural innovations are demanded from the author, and in the authoring-by-configuration case exemplified by FearNot!, the author might not be able to create the desired behavior, and he might not be able to decide on whether a certain behavior is feasible before extensively trying.

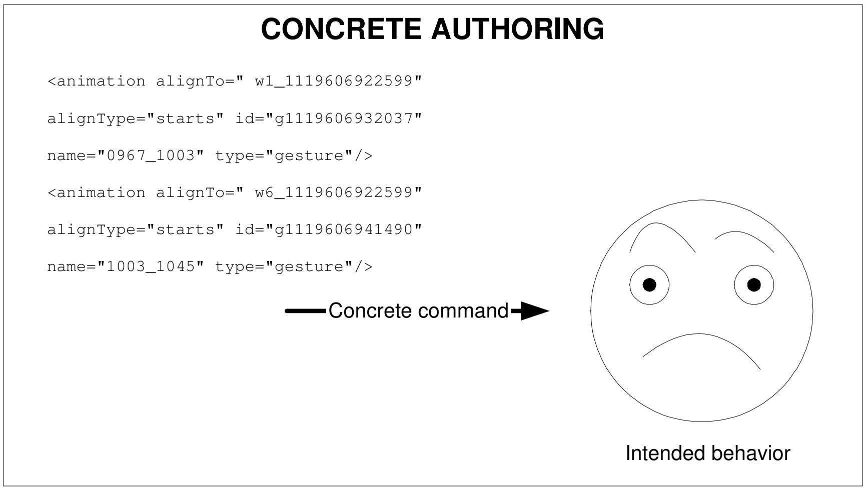


Figure 13 – Concrete authoring means to direct the virtual character directly, e.g. employing a script. It is much more intuitive and allows for more control by the content creator. Concrete authoring for interactive behavior is still a challenging research quest, because of the questions related to generalization.

“Abstract” authoring shall be defined as the authoring process where the author works on an abstract level, e.g. setting parameters, in order to achieve a desired result, and the abstract work applies to a set of cases. Configuring, but also developing a model is “abstract authoring”, in this sense. For example, “concrete” authoring means that the author is able to define or employ a concrete case at will, without having to approximate the necessary parameters.

Thus, a framework is required that facilitates working on the concrete cases, i.e. that also, though not necessarily only, allows for concrete authoring (Figure 13). This is derived from the postulated primordiality of the application (experience) design (cf. Section 2.3 to 2.5). The system shall empower the author to define exactly (within the expressive range of the system) what will happen, at a certain situation, to demonstrate, test and visualize his intentions. The author shall, in other words, always be able to express what he wants the system to do, at a certain story situation. He shall be able to postpone the required generalization, which is likely to be teamwork.

2.6.5 Conclusion and Summary – The Properties of a Suitable Authoring Framework

The preliminary decisions of this dissertation – primacy of a sequence selection framework over questions of modeling, dedicated role of content creator, and focus on the concrete authoring – imply tradeoffs. Setting fine state approaches over modeling makes reusability of content difficult, and the focus on the concrete authoring implies possibly that, in certain cases, the generalization task is only delayed, and may even not be possible to solve. Given the priority assigned to the application concept, these tradeoffs are acceptable: In the present context, reusability is not as important as understanding the possible applications the content shall be reused for, and delaying the generalization task gives priority to the clarification of examples from which to generalize.

The discussion of the general thoughts on authoring of IN is described by the following items:

- **Procedural Creativity in a Team.** Procedural creativity is indeed currently a requirement for creating novel IN applications. But the authoring process and tools must foster teamwork and different roles. It should not be required that the same person must be both creative on the content level, and able to invent and program new algorithms.
- **Developing Non-Programming Authoring Methods.** Several methods exist that allow for the creation of interactive content without sophisticated programming skills, e.g. finite state machines, learning algorithms, linear optimization. An authoring environment for IN must offer such methods to the content creator, who should not be expected to program, at least not extensively and inventively.

- **Concrete Authoring is better.** Abstract authoring faces the knowledge acquisition bottleneck (cf. Section 3.2), i.e., content creators may not be able to decompose the desired behavior into the required parameters, and the system might turn out not to be suited for producing the intended behavior. Authoring should therefore be as concrete as possible, in order to allow for creative expression of content creators.

Section 3 will discuss whether an authoring methodology for IN can find firm ground in the Humanities, and Section 4 will lay the theoretical basis for the solution of the authoring problems of IN.

3 Story Structures and Story Persons

3.1 Drama Theory

- In this section, the question on why interactivity and narration can be combined is examined.
- A conclusion of the examination is the hypothesis that an important way of combining interactivity and narration recurs to a specific form of presenting and simulating psychologically relevant problem solving strategies (cf. also [Iurgel03c], [Iurgel03a], [Iurgel05a]).

3.1.1 Searching for a Place for Interactivity within Drama Theories

This section will deal with some basic elements of drama theory. The goal is not a complete coverage of this vast field, but rather a focused examination of the literature with the aim of elucidating how interaction can sensibly be integrated into narrative structures. Since, as exposed in Section 2.4 and 2.5, IN denominates a metaphoric process that can lead to several divergent, but equally valid innovations, the answers that will be given in this section cannot be but partial. The exposition and analysis of the theory on screenwriting will show the interdependencies of character development and story line, and will allow for an answer to the question of how to integrate interactivity into narration. The two theories that will be discussed below both focus on character driven drama.

3.1.1.1 Categories of Story Structures

Robert McKee is one of the most renowned theorists of screenwriting. He has proposed the following categories for understanding successful screen narration

Structure: The sequence of scenes and events that combine into a story in order to create suspense, emotion, and to transmit a message.

Story Values: Universal qualities of human experience. They are either positive or negative, and they change during a story event, moving from positive to negative, or vice-versa.

Story Events: The events that change and express the true self of a story person. Normally story events are changes in his life situations that change story values.

Beat: The minimal elements of the story concept ontology. Beats are the concrete actions of the protagonist and of its environment, and composed around the changing story values.

Scene: An element determined by continuation of space and time, where the "fabula" reaches is maximum closeness to the "sujet" (there are no time gaps and no space jumps). Within a scene, the story values changes, and this change is the meaning of the scene.

Sequence: A sequence of scenes with its own local apex (that may be identical to the global apex of the story).

Act: A series of sequences with an own local apex (that may be identical to the global apex of the story).

Story: A series of acts the apex of which is also the final climax.

3.1.1.2 Story as Revelation of Deep Personality Layers

McKee's analysis is centered on the theme of personal adaptation and growth. A story is a series of acts that are composed of sequences of scenes, which employ beats to expose how the story persons react and adapt to the changing story values. The change of story values shakes the inner self of the main story person, in an increasing manner, until the hero becomes another person.

"True character is revealed in the choices a human being makes under pressure — the greater the pressure, the deeper the revelation, the truer the choice to the character's essential nature." ([McKee])

Though it might appear difficult to see how this analysis carries over to action centered Hollywood movies, e.g. to movies with a character like James Bond as a protagonist, it is sufficient for the argumentation developed in this dissertation that a large number of movies and also novels conform very well to this scheme, e.g. Casablanca, Sleepy Hollow, Oliver Twist, Effi Briest, just to name some random examples.

According to [McKee], conflict is the key motivator of a story. And conflict is so all-important because it reveals the real and true self of the main character. According to his analysis typical stories reveal deeper personality layers of the story persons, especially of the protagonist. In the course of the story, the protagonist reveals surprising inner traits, as a result of inner transformations induced by the challenges of the story. Thus, the function of a story is to expose the inner transformation of a person, which is provoked by extraordinary challenges. A story thus exemplifies forms of human adaptation and growth.

Robert McKee:

"In story, we concentrate on that moment, and only that moment in which a character takes an action expecting a useful reaction from his world, but instead the effect of his action is to provoke forces of antagonism. The world of the character reacts differently than expected, more powerfully than expected, or both."

This analysis is supported by Frank Daniel's models. According to Daniel, the protagonist of a story has a set of wants; the story revolves around the appearance of obstacles to these goals, the overcoming of the obstacles, and the stakes of this struggle, i.e. the risks that the main person take (Cf. [Howard]). The very core concept of Daniel's drama theory was expressed by Howard in a single sentence: "Somebody wants something badly and is having difficulty getting it. "

A story is a structured series of obstacles and stakes that create a dramatic arc by exposing increasingly difficult obstacles and dangerous stakes that induce a transformation onto the character, which causes it to drop or adapt the initial want and that throws him back to more fundamental needs. Struck has noted the imminent learning aspect of this theory:

"No other dramatic model offers this parallel to the educational paradigm of learning by doing and problem-based learning." [Struck]

3.1.2 Interactivity as a Specific Kind of Problem Solving

In a certain sense, according to McKee and Daniel, character driven story is a kind of experiment the character is exposed to, devised in order to reveal his true nature, together with a description of the relevant settings and circumstances of the experiment, and the actions and reactions as attempts at coping by the protagonist. Other theorists could have been named to support this view, e.g. Linda Seger [Seger].

For interactive storytelling, this approach implies, in my view, that either (i) the player is confronted with stakes, or (ii) the non-player characters are confronted with stakes, or (iii) both. The unexpected emergence of an imbalance of life's equilibrium by the structured presentation of obstacles, together with the attempts to restore balance, are then essential elements of the interactive narrative.

Now, based on this analysis, it is possible to extract some features of stories that make them apt for interactivity: The omnipresence of problem solving lends itself into the realm of simulation ("emergent stories"), where the "plot" is the presentation of consequences of decisions and strategies, and of typical aspects of the problem; the problem arises that some strategies may in fact prematurely solve, or maybe ignore the problem, so that no further "story" can be told. This is probably the core problem of "emergent narrative". For example, killing a spouse in a marriage simulation is a definite solution to the marriage problems; the remaining of the story (police investigation, prison, and so on) might be a very interesting story in itself, but is certainly a different problem.

3.1.2.1 The Importance of Typicality

It is most improbable that IN will be able to succeed without a concept of typicality, i.e. of typical strategies and of typical presentations and consequences of those strategies. The infinite range of non-typical events and actions – e.g. killing one's spouse – cannot be considered by the software based story system. In this sense, also emergent narrative is likely to require Aristotelian "mimesis" [Aristotle], i.e., an intention of the author to expose typical and exemplary courses of reality, beyond mere simulation. With appropriate restrictions, it might be possible to devise some IN with focus on "external", not psychological problem solving where the user is the protagonist, though the practical or conceptual "proof" (e.g. a functional example application) for this is still missing.

There are various ways of conceiving of IN as experiments in human, social and emotional problem solving. My hypothesis is that an emphasis on typicality will enable the transition from pure simulations without plot, or adventure games where the plot line is fixed, to compelling interactive narrations (Figure 14). An interactive story touches and compels because typical problems and typical consequences of certain coping strategies are presented in an elucidating, condensed and touching form, neglecting superfluous details and omitting irrelevant slides of time.

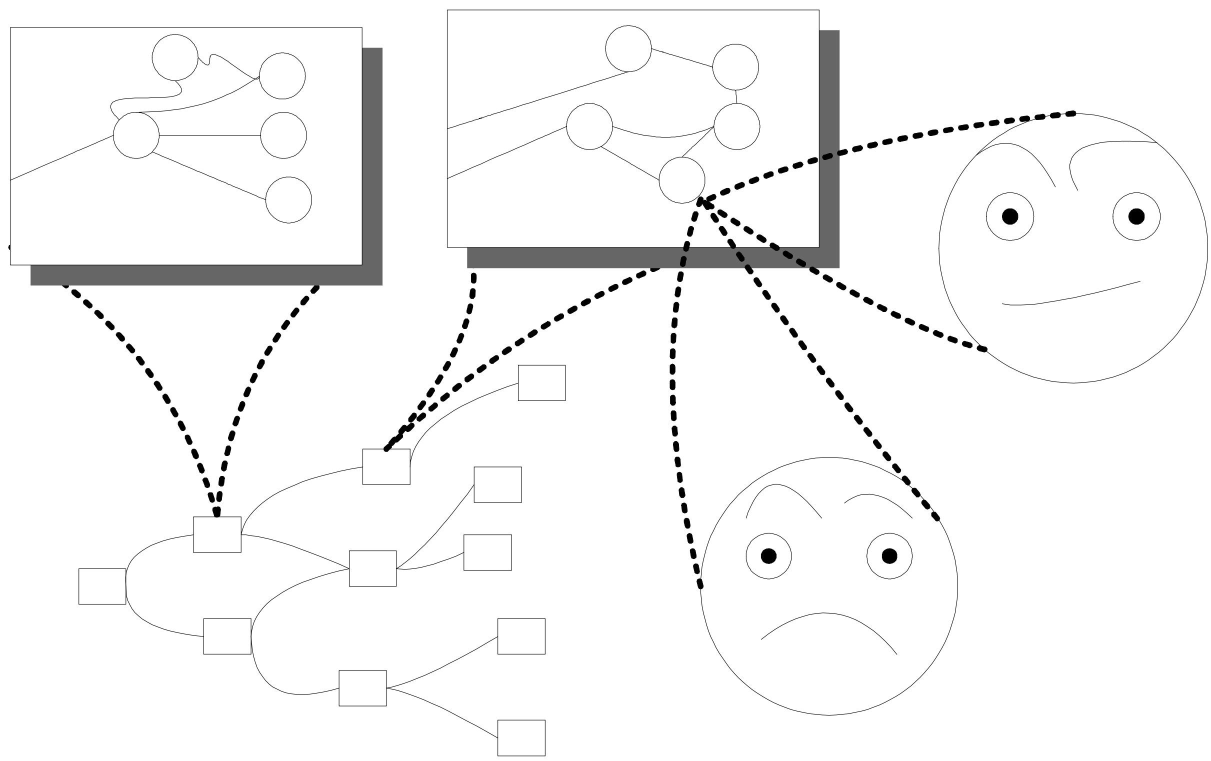


Figure 14 – Typicality is a conceptual solution to the problem of making simulations interesting and artistically appealing. It also promises a simplification to the problem of explosion of possible consequences of interactions. The *typical* consequences amount to some sort of branching structures. Some additional variation can be achieved by improvisational behavior of the virtual characters.

3.1.2.2 Interactivity and the Inner Self of the Protagonist

An essential aspect of character driven linear stories that predisposes them for a natural addition of interactivity is the psychology involved in the coping attempts of the protagonist. Here, assigning to the player the role of protagonist is impossible, because the "inner", psychological processes and emotions of others are in focus (cf. [Iurgel03a]. The influence of the user must be indirect, e.g. as sidekick of the protagonist. The most natural way of gaining insight into the inner life of another person's problems is by speaking to them, and possibly to their relatives and friends. If a story reveals the changing "inner self" of persons and the enjoyment derives from observation and identification, a deeper understanding of this inner self is likely to increase the story enjoyment. We can interrogate the virtual protagonist about his feelings, his motives, and we could try to gain its trust and give advice.

This brings IN close to psychotherapy, because speaking about someone's psychological problems, his versions of stories, and his social relationships is then part of the function of the sidekick. Thus, I believe that this explains why therapy elements are already present in the Façade-IN application.

3.1.2.3 Natural Langue Interaction as a Game Challenge

The role of natural language interaction in such psychological IN then becomes eminent. I suggest that speaking to the virtual protagonist can be made into part of the games quest, i.e. finding the right words and the right questions, and the right way of giving advice. Exactly this search for the right language is part of the novel psychological exploration that IN enables ([Iurgel05b]). This makes the use of NLP \(Natural Language Processing) necessary. In contrast, some uses of NLP in IN are not necessary (i.e. they are contingent), because the interaction could also be designed employing a GUI or a set of commands. Finding the right way to interact is not part of the game, at these applications.

3.1.3 Conclusion and Summary – How Interactivity can be Combined with Stories

IN remains a "wicked" problem field. There is more to IN than a simulation of problem solving, because imposing obstacles on the user and presenting consequences of actions and of coping strategies must follow a dramatic composition that has not yet been fully understood and transposed to interactive applications (Figure 15). Hence, the need for an experimentation platform with focus on authoring: It is necessary to try things out. At best, drama theory can provide some guidelines for novel IN applications, but cannot replay experimental development and innovative creativity.

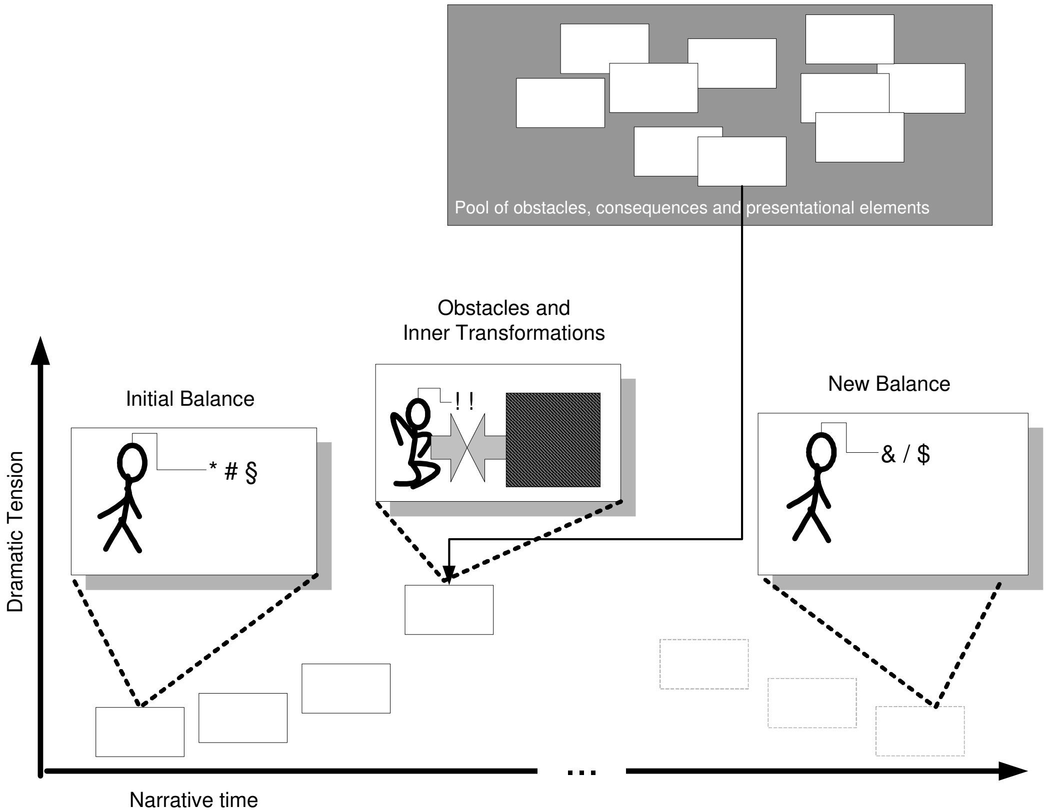


Figure 15 – According to McKee and Daniel, story consists of a presentation of an initial harmony that is disturbed, a series of obstacles that the protagonist has to overcome to restore the harmony, and the revelation of the inner processes and changes that he undergoes during the course of the story. An interactive system that adopts this theory has to deal with the choice of obstacles, consequences, and presentational elements. Many technological and design questions are still unanswered, concerning how to combine an interesting story arc and a believable simulation of problem exposition and problem solving.

Following character centered narrative theories, the assumptions were developed:

- **Interactivity as Support and Psychological Discovery.** Since standard stories deal with inner transformations of persons that are faced with difficult obstacles, the most natural way of transforming a story into an interactive application consists in aiding and assisting the protagonist in his coping attempts. Thus, IN is closely related to a psychotherapy simulation. A major appeal of future INs then could be derived from discovering “hidden”, “deep” layers of the personality of the protagonist.
- **Story as Problem Solving.** In general, stories are creative, artful presen-

tations of how people deal with exemplary problems. Every interactive narrative is likely to contain a simulation of solving emotionally dense and socially intricate problems.

- **Problem of Presentation of Obstacles and Consequences.** Since stories contain a structured presentation of obstacles that the protagonist has to master, a major "wicked" research challenge lies in understanding how to present the problem in focus, and the consequences of coping strategies, in a manner that is still "narrative", i.e. emotionally immersive and convincing. The notion of typical courses both allows for mastering technological problems and for a compelling experience.
- **Natural Language as Task.** Since psychological assistance and exploration can be made part of the application design, employing natural language can be conceived as part of the interaction task, i.e. "finding the right words" and the best dialogue strategies need not be trivial, for a class of psychologically inspired IN applications; rather, wording and dialogue strategy should remain a challenge, reflecting interpersonal reality.
- **Story is hierarchically structured.** McKee's analysis has shown that stories possess various levels of descriptions, e.g. scenes, acts and beats. An authoring framework for IN should enable authoring on these different layers.

The detailed understanding of these items remains a considerable research and interaction design challenge; the applications described in the present work are attempts at approaching solutions (cf. Section 6 and 7).

3.2 Emotion and Personality

- In this section the standard theories of emotion and personality are presented and discussed. A conclusion is that much caution must be exerted when employing them for IN.

3.2.1 Introduction – The Standard Theories

As has become apparent in the previous section, psychology can be crucial for IN. Virtual characters with some model of emotion and personality can be useful for IN in general, and are necessary for emergent approaches.

As with other disciplines related to IN, emotion psychology and personality theory are vast and complex research areas that mostly produce results that are not intended for computation or simulation, and certainly not for being used within IN. Thus, no overview of psychological theories in general is intended, and no decision is taken in favor of a certain theory; rather, the investigation will deal with the question of whether IN-research can safely build upon emotional and personality models in general.

Virtual characters, both within and without the context of IN, are almost invariably implemented with the help of only two theories: first, OCC (named after the authors Orthony, Clore, and Collins, [OrtonyClore] is the standard theory for emotional calculations; second, the so-called “big five” are the standard parameters for personality modeling. A few other emotion theories were also already used, but employing them does not solve the problems that will become apparent with OCC. Cf. [Gratch] for an overview of other implemented theories.

OCC was explicitly developed for being employed in a computational context. It consists of a set of rules, together with a few formulae. The rules describe which antecedents cause which emotion out of a set of 22 emotions. The antecedents include descriptions of the intrinsic value of an event, the time frame of it (future, present, and past), and the responsible agent (who is responsible and possibly blameworthy). Figure 16 shows the types of antecedents. Figure 17 depicts a tree representation of some rules. For example, the model dictates that “fear” emerges when a non-desirable event is likely to happen in the future, and the intensity of the emotion is dependent on the degree of undesirability, and the likelihood.

Focus of Attention	**Outcomes of Events**	**Actions of Agents**	**Attributes of Objects**
Source of Value	Goals	Standards	Tastes/Attitudes
Appraisal	Desirable or Undesirable	Praiseworthy or Blameworthy	Like or Dislike
Affective Reaction	Feel Please or Displeased	Feel Approval or Disapproval	Feel Liking or Disliking
Emotion	Joy, Sadness, Fear, etc.	Pride, Shame, etc.	Love, Hate, Disgust, etc.

Figure 16 – Emotions are, according to OCC, a result of the appraisal of events, actions or objects, in dependence of their intrinsic qualities. The table was reproduced according to [OrtonyClore].

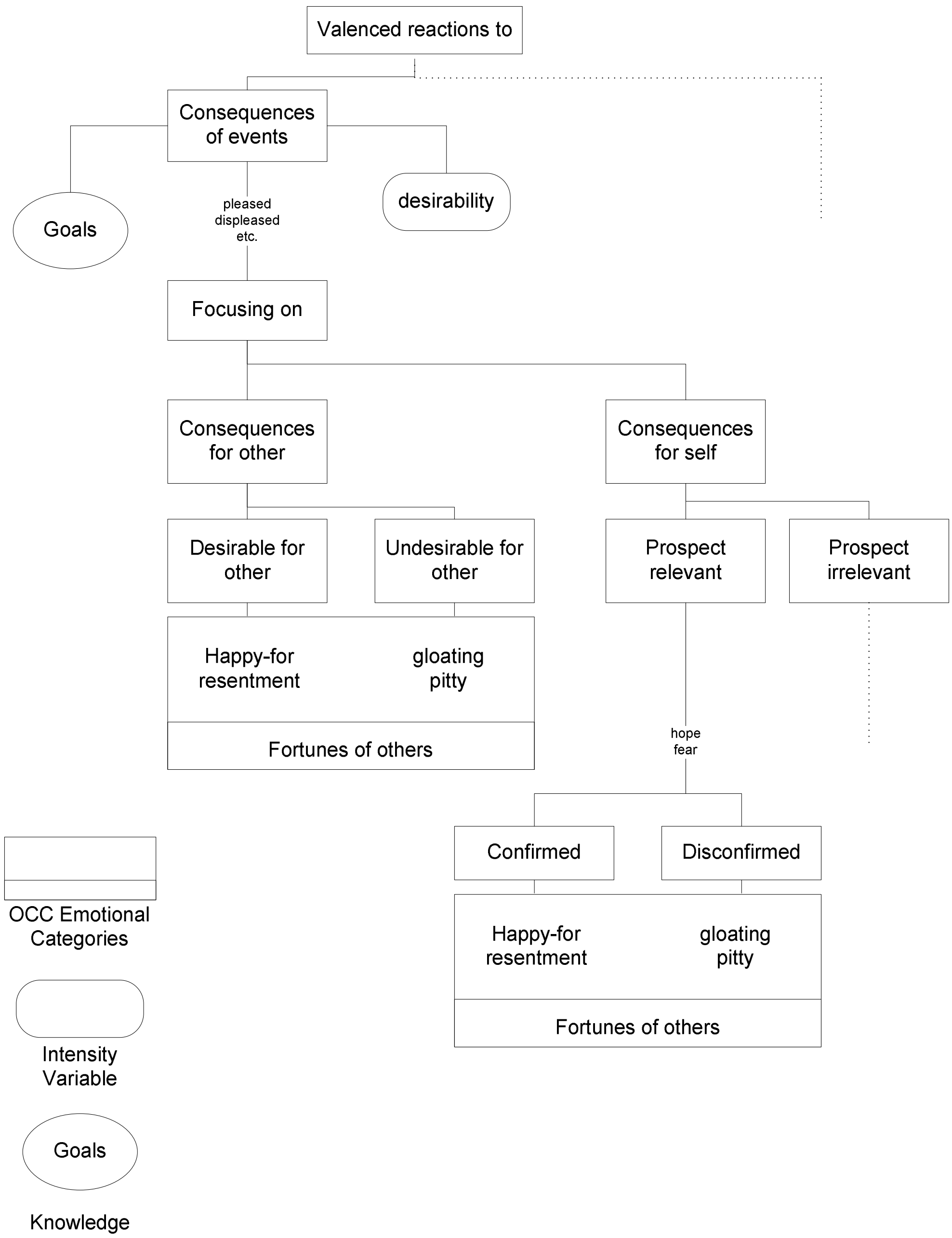


Figure 17 – A partial tree that shows antecedents of emotions. Reproduced after [OrtonyClore].

The standard personality theory is the "big five" or "OCEAN" parameterization of personality. OCEAN is the acronym for "openness", "conscientiousness", "extra-version", "agreeableness", and "neuroticism". The theory is linguistically based, and states that nearly all adjective can be grouped into clusters that are subsumed under these parameters, cf. [Saucier]. The parameterization of the big five provides exactly an abstraction over the vast range of daily personality adjectives, allowing to group them into clusters, but thus also ignoring the difference between the members of the cluster.

3.2.2 Discussion – Standard Theories are "Trivial"

OCC has already proven its usefulness with virtual teachers, salespersons, and assistants. However, when IN is concerned, OCC is not "deep" enough. In fact, no rule-based theory is likely to be "deep" enough to provide an emotional model for IN in general. This is because OCC deals with very standard emotions, e.g. with the fear induced by a thief that enters our house. But when we follow McKee, narratives become meaningful only at the point where "the inner self" of a person comes to the surface, driven by the stakes and the coping attempts. The narratologically interesting emotions are those that are related to inner transformations, complex problems, and psychologically intriguing coping attempts. Certainly, OCC does not target at "deep" layers of personality; it is a system for describing the most basic emotions. From a storytelling point of view, these emotions are trivial. An example of a storytelling relevant emotion could involve a story person that feels aversion and fear towards someone exactly because he is similar to that other person, but is not able to recognize and accept this similarity (cf. Section 7). The process that leads to an acceptance of the other person and, as a consequence, to accepting oneself, may amount to a fascinating narrative. But such important aspects are missing in OCC.

Usually, emotion theories like psychoanalysis are associated with "deep" emotions. Unfortunately, no known psychoanalytic-style theory can be readily implemented.

Concerning the OCEAN personality theory, a similar assessment is valid: Depending on the intention of the IN, OCEAN cannot be subtle enough for the intentions of the author, because it was created exactly in an attempt to abstract from the subtle differences expressed by common sense words for personality traits. An author way desire to create a certain virtual person with very specific personality traits, but these personality traits usually cannot be derived from a configuration of the OCEAN parameters.

In fact, part of the problem is due to the well known "knowledge acquisition bottleneck” (cf. [MurrayJ]). This problem has already proven to be a major obstacle for employing experts systems, within many domains. Domain experts do have the required knowledge, but it often turns out to be impossible to extract the knowledge from the experts and to express the knowledge in form of reliable rules. The knowledge is “intuitive” and “tacit”, it is often more a “know how” than a “know why”.

Thus, IN is susceptible to the knowledge acquisition bottleneck. The more “psychological” and “deep” the interactive story is intended to be, the less sufficient a verified rule based system will be for producing the required behaviors and emotions.

OCC also contains other pitfalls that make its use difficult for IN: The antecedents of the rules, the emotion eliciting contentions proper, are very abstract. Formal objects that elicit emotions – actions, events, and things – are intrinsically “good” or “bad”. Within IN, when social constellations and “deep” feelings are involved, and especially when interactive dialogues are on focus, it is very difficult to determine what is “desirable” and which rules determine that a certain utterance in a certain context shall cause anger in a virtual personality (cf. [Iurgel02b]). For example, an author might intend that giving premature advice to a virtual character, before having established mutual trust, shall cause it to become angry with the player – this is easy to “hard-code”, but we should not expect any generic available emotional model to be able to generate this behavior.

Equally, OCC provides little information about how and when emotions are expressed, e.g. little more than words like “anger” can be offered. A gap remains to be filled when it comes to the subtleties of emotional dynamics present in a meaningful and emotionally loaded conversation, cf. [Bartneck].

3.2.3 Conclusion – A Generic Authoring Framework Should Not Rely on Standard Theories

A commented overview of the standard theories of personality and emotion for virtual characters was presented. At least for IN that follows McKee’s lead and focuses on “deep personality layers”, existing theories are not subtle enough. The content creator or domain expert normally will intuitively know much more than a standard theory can express.

The implication is that a framework for the creation of IN applications must provide some measure to enable the implementation of psychological complexities that go beyond standard models. Cyranus (cf. Section 5) supports the usage of ad hoc models that content creators and computer scientists devise by employing their own procedural intuitions, and allows for using directed graphs for predictable interaction courses; thus, the impression of psychological depth can be created, while some standard model can be employed for non-critical interactions and for unpredicted input. The model of Creactor (cf. Section 7) dwells on how an iterative and intuitive authoring cycle devised specifically for virtual character behavior could integrate domain knowledge implemented declaratively, and exceptions that enhance the impression of depth.

Thus, the hypotheses that follow from the present section are described by the following items:

- **Primacy of the Surface.** Since standard models cannot provide subtle and “deep” emotional responses and personality features, an IN system that shall allow for a design based on “deep” character transformations should include means of creating the impression of psychological depth, even if this depth is not modeled, but only forged, i.e. superficial.
- **Ad Hoc Modeling.** The IN system must also provide a means for creating ad hoc models, i.e. models that are based on the intuitions of the content teams and that are only valid within a narrow context of the application, e.g. within a part of a scene. Thus, the impression of depth can be maintained, without requiring further open-ended scientific studies.
- **Knowledge Acquisition Bottleneck.** Interactive narration applications must be created facing concrete interactions of the application, because intuition and expertise of content creators on how a virtual character should behave need the concrete case to be applied. The content creators and experts will usually know at concrete story situations what should happen, but

they usually will not be able to express a valid rule system that generates these behaviors. This is due to the knowledge acquisition bottleneck, and is a central problem of IN.

At this point, the basis for understanding the requirements towards a generic authoring framework is laid: The framework must not implement or rely on any psychological or narrative theory, but must enable and facilitate experiments with different established theories and ad-hoc models, and must allow for a creation process that, at least at its initial phase, does not endorse any theory from the domains of arts and humanities. The next Section will elaborate the theory for such an authoring framework.

4 A Hybrid Visual Formalism for Authoring IN

- In this section, the formalism that responds to the requirements of interactive narration is developed. Cf. also [Iurgel03d], [Iurgel04a], [Iurgel04b], [Iurgel05c], [Iurgel06a]. It constitutes the theoretical basis of the authoring framework.

4.1 Motivation

The most established authoring formalisms for the control of the flow of interactive, multi-media presentations, interactive narration, educational applications and games are based on directed graphs, e.g. [GebhardtKipp], [BourgSeemann], [GrützmacherWages], or the commercial system for game creation Simbionic[9]. However, one drawback of directed graphs is that they are not able to handle all the problems of sequence selection in IN, because of possible explosion of complexity [Crawford]. For these cases, mechanisms for integrating alternative control algorithms such as planning will be devised, and mechanisms for migrating smoothly from the directed graph to another sequencing method whenever necessary (cf. below).

Despite the aforementioned drawback of directed graphs, there are good reasons to use a directed graph subsystem. They are as follows:

- **Well proven**. Many experimental or standard type IN applications can be fully built with directed graphs, without need for further enhancements.
- **Prototyping.** Many IN applications can be rapidly prototyped with directed graphs, even if the final system is known to require different technologies.
- **Agile development.** Directed graphs can be employed for fostering short production cycles, when other required algorithmic methods are not fully developed or not fully understood and must be co-developed with the application design, due to the “wickedness” of the area.
- **Ubiquitous applicability.** In praxis, IN applications usually possess plenty of parts, scenes, beats, dialogue games etc. that can be modeled with directed graphs – even if other parts require other technologies.
- **Predictable events.** Directed graphs can be employed for generating

[9] Cf. www.symbionic.com

"hard-coded", easy-to-create, excellent reactions to predictable input, e.g. for utterances of virtual characters as reactions to certain user input, while algorithmic (e.g. a generic and generative dialogue manager) modules can be used when the prediction fails.

- **Intuitive.** Directed graph formalisms are intuitive enough to be employed by non-computer scientists.
- **Concrete.** Finite states facilitate concrete authoring, because usually a state of the system can be interpreted as a particular situation in the story, and the author can then define exactly what should happen next.

In this section, it will be shown how a directed graph basis can serve as starting point for an authoring environment for IN, given the current situation of a "wicked" state of research, where technologies, application concepts and authoring methods have to be developed together, and are interdependent. The directed graph serves as a basis, since it is always possible, within the hybrid formalism, to handle system behavior with the directed graph, and insofar as a first intuitive, concrete approach to a certain problem can be initially and experimentally handled with directed graphs. Directed graphs are also basic in the sense that the introduction of a delegation chain (cf. below, Section 7.3) will allow to employ directed graphs as first choices of the system, in order to produce excellent reactions of the system when a predicted input occurs, and generic, less accurate reactions when the input differs from the expected flow. Thus, directed graphs serve as a visual formalism that can always be employed, in appropriate parts of the system, and that can also be used for prototyping.

The main strategy behind the decision to address complex sequence selection questions by starting with directed graphs is to allow for a cautious migration into the sphere of "wickedness" form a starting point that is not wicked at all: Directed graphs are well understood and extensively employed for varied interactive applications, including games and educational applications. An incremental and intuitive authoring of an interactive narrative must support a stepwise enlargement of the amount of possible states, of the level of complexity of permutations, and finding out of appropriate structures and algorithms during the creative process. This responds to the fact that appropriate precise story algorithms and declarative structures, e.g. in the Proppian vain [Propp], are normally, in IN, not known beforehand, at the beginning of the creation process.

A lesson learnt from the Façade system is that it is paradoxically necessary to handle sophisticated natural language interaction in IN with shallow methods. But Façade does not offer any visual support for authoring of shallow natural language processing. The hybrid formalism developed in the next sections is perfectly suited for this task, but at the same time it offers a visual concept, and allows for integrating external, more generic talking modules (cf. also Section 6, and [Iurgel05b]). The "shallow" method is built upon predictability, whereas a "deep" method of NLP relies on principles. Predictability means that, in a given dialogue situation, a certain set of reactions from the user can be expected, and these can be handled with different grades of precision, in order to obtain appropriate reactions of the virtual characters. Predictability is a fallible principle for NLP, as many sensible inputs are possible that will not be predicted, but it can often be employed rather efficiently. Predictability is the reason why a use of directed graphs in dialogue management for IN is possible.

The novel Hybrid Control Formalism will thus allow for experimenting with sequencing methods, for an incremental creation process, and for a concrete authoring. It also supports content creation in its proper sense, where the author is not required to program, and teamwork of content creators, computer scientists, and content specialists.

Some details of the formalism may require adaptations on a project basis. In the projects described in Section 6, the only project-specific addition to the formalism has been a function to exchange memory state data, cf. Section 6.2.2. Nevertheless, also modifications to the authoring formalism on the programming and definition level must be regarded as a normal part of the development of a "wicked" development process on innovative IN applications.

It is necessary to stress that a generic formalism is not a generic solution to all authoring problems of IN. When the authoring team decides to employ sequence selection methods different form directed graphs, the framework developed in the present section and in Section 5 offers appropriate interfaces, data slots, and processes for integrating these methods. It is generic in this sense of allowing for the use of any additional sequencing paradigm, with additional specific authoring support. No assumption is made with regard of these methods or their specific authoring process, besides the interface and data structure definitions.

The formalism can be employed for gradually testing and developing specific algorithms to IN sequence selection problems, departing from the proven concepts of visual formalisms. For simplicity, every control approach that is derived from finite-state machines, i.e. that employs a finite set of states and transitions between them that define the control flow, will be called here the “directed graph” approach. In the following, any method to determine the sequence of storytelling elements that is not based on directed graphs will be called, for simplicity, an “algorithmic method”, or “algorithmic approach”. For example, the choice of a subsequent scene according to a story engine that implements an Aristotelian model is an “algorithmic approach”.

Given the case of a limited set of possible states, current authoring tools based on directed graphs reach their limits when

- the complexity of the connections is too high
- the guard conditions become too complicated

The measure of what counts as “too complicated” depends much on the background and talent of the author. Employing the Hybrid Control Formalism based on directed graphs developed here, it is possible to approximate desired complex system behavior, and to migrate to an algorithmic, non-graph based sequencing method when necessary and when possible, i.e. when the stage of a graph that is “too complicated” is reached.

The authoring process that is assumed enabled by the Hybrid Control Formalism initiates at an almost linear story idea with few bifurcations, which is then gradually extended by further branches, until possibly the use of some algorithmic approach appears to be more appropriate. Thus, it is possible to start with the simple use case, and to increase complexity and involvement of computer scientists stepwise, as required (Figure 18). Examples will be given below. When things start to get wicked, i.e. when the application idea and the necessary technology are not sufficiently clear, it is then possible to experiment with different ideas and technologies that are added to the formalism only when and at the places where this is required, whereas other parts of the envisaged application can still employ the standard graph based approach.

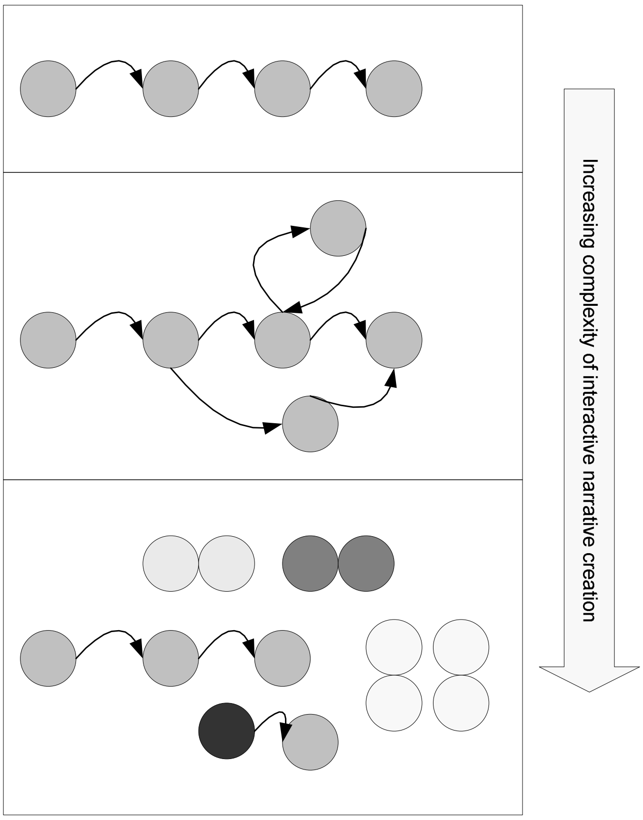


Figure 18 – The approximative and experimental authoring process that is enabled by the Hybrid Control Formalism. Starting with few or no interactions, specified with directed graphs, complexity can be added stepwise, until the directed graph must be partially or totally replaced by algorithmic sequence selection methods.

4.2 Directed Graph Basis – Harel's Statechart

The following section deals with an appropriate visual formalism based on directed graphs to form the basis for an authoring platform for experimenting with novel IN applications. The main feature required is the use of hierarchies, because the composite states (compounds) allow for an interpretation as story elements, such as beats, act, or scenes. Harel's Statecharts are the most common and well-understood representation that fulfils this requirement [Harel], cf. Figure 19. Many products are widely used that are based on Statecharts, e.g. Statemate, Stateflow, Statecharter[10]. Harel himself has expressed the short formula:

“Statecharts= state-diagrams + depth + orthogonality + broadcast-communication”.

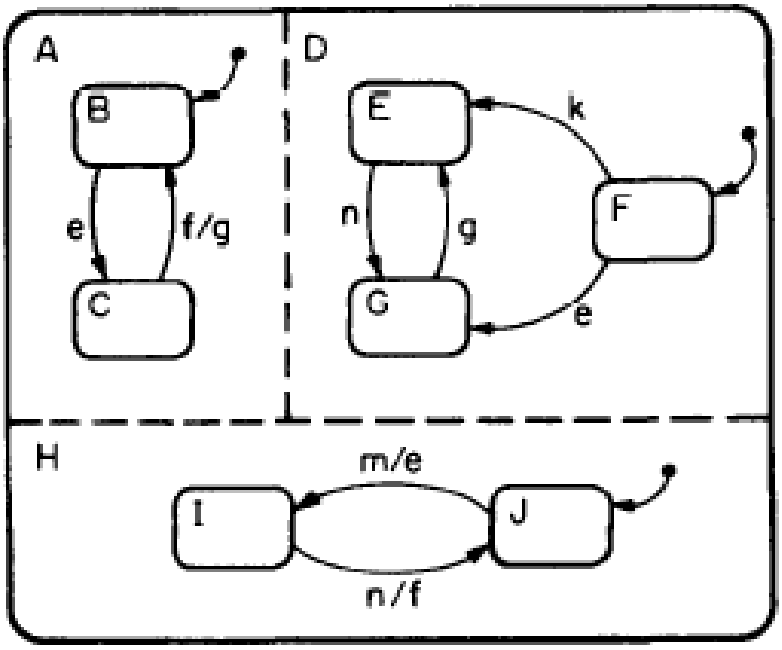


Figure 19 – A figure of taken from Harel’s original paper, depicting a Statechart. cf. [Harel]

Statecharts introduce hierarchies into the representation of finite-state machines (“depth” in Harel’s quote, above). Equally, the other remaining faculties of Statecharts are important for IN, i.e. the property of memory as part of the hierarchical structure (remembering which terminal state was active when the compound is interrupted, in order to be able to reactivate it when the compound is reactivated), orthogonality (parallelism), and broadcast-communication (i.e., activated states can generate events to orthogonal parts). Thus, the original Statechart definition by Harel can serve as the basis of the present formalism.

Parallelism is an important innovation introduced by Harel. Here, two important benefits for the present context shall be highlighted: coordinated virtual character

[10] Cf. www.ilogix.com, www.mathworks.com, www.nwoods.com/, respectively.

behavior, and implicit modularization. “Implicit modularization” means that parts of the transition network can be employed as otherwise modules of a software architecture would be, and broadcast can be employed as interface communication. This is advantageous for prototyping, since this feature enables the content creation process to start right away, without having to create new software modules. If necessary, the control structures can later be reused for driving the postponed modules. For example, every virtual character can obtain an own AND-coupled compound state, whereas the control of the choice of scenes could be assumed by another parallel compound. Figure 20 shows an example of orthogonality for IN.

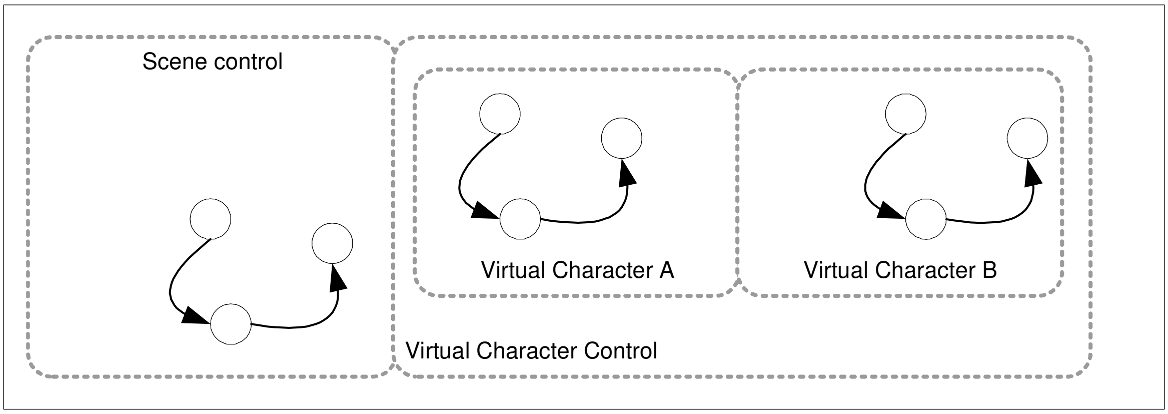


Figure 20 – Example of orthogonal (parallel) use of compounds. The dotted lines signalized orthogonality. In this example, the control of the scene objects, e.g. cameras, is done in parallel with the control of the virtual characters, which are “autonomous” in the sense of employing independent compounds.

Parallelism can also be employed for camera control and control of secondary scene objects in general.

Hierarchies of the Statechart representation present the following advantages over simple graphs:

- **Break conditions**
- **Memory**
- **Visual organization**

"Break conditions" is a term that refers to outgoing transitions attached to a compound state. They are basically equivalent to attaching copies of this transition to any transition inside the compound state. This is used when some event or complex condition shall stop the actions being carried on inside the compound, e.g. when the user changes theme during a conversation.

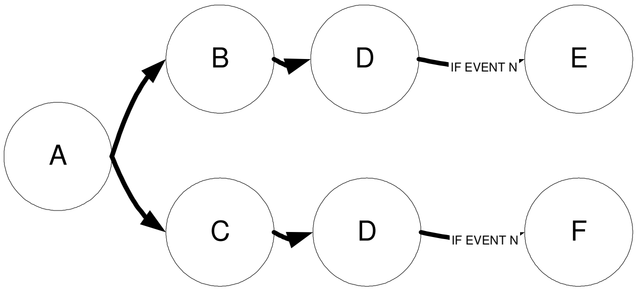
"Memory" allows going back to a compound, e.g. when the user goes back to a conversation topic.

"Visual organization" means that often, grouping action units on semantic considerations might be useful, even if no saving in transitions can be achieved, for example when a group of node form a "scene".

Note that multiplication of otherwise identical states to cover different trajectories would only be necessary if a strict Finite State Approach were adopted, where the conditions of transitions are only checked against identity with incoming events. But, because complex functions can be employed in the conditions of the present formalism (cf. below), and common data storage is employed, different trajectories can be defined without multiplication. This is mentioned here because the argument of multiplication is sometimes dogmatically employed against the directed graph approach [Crawford]. Of course, conditions can become unmanageably complicate – this can be handled either with GUIs that hide this complexity from the content creator, or by employing Sequencing Engines (cf. below). Figure 21 exemplifies how complex conditions allow avoiding state multiplication.

The simple directed graph patterns are the standard technique employed in games [BourgSeemann].

Several extensions to the Statecharts are required by storytelling considerations: (i) Reentering transitions, (ii) reference frames, (iii) attaching conditions and actions to compounds.



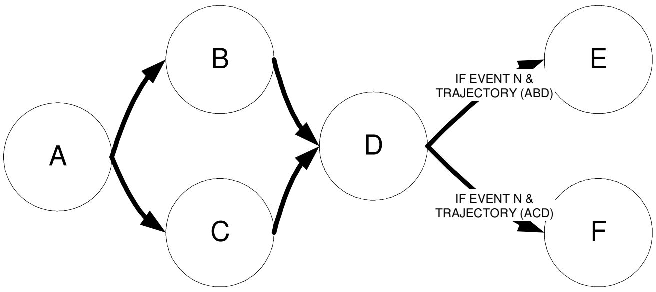


Figure 21 – The trajectory DE shall be followed if it is preceded by the trajectory ABD, and DF shall be followed if the previous trajectory was ACD. The graph above depicts the strict Finite State approach, where, due to the absence of memory, duplication of state representation of D is necessary. Below the transition network approach is shown, with data storage, where no duplication is necessary.

4.2.1 Extending the Statechart by "Reentering Transitions"

Harel's usage of the memory state is not sufficient for IN. For Harel, "memory" means that, when a composite state is reentered, the last active state will be activated [Harel]. A memory state of a Statechart is usually represented employing a circle with an "H" inside, cf. Figure 22. This is a kind of dummy-state; it causes the state that was active when the compound was left to be entered. When no such state exists, a special "start" state is activated (not shown in the figure). However, in storytelling, reentering a state may require some independent action. This is because the user might need some orientation about what is happening, e.g. the system might be required to express "then, what kind of instrument will you choose?", to remind the user that the state was left at a point where some decision on the interaction instrument (prop) was expected from him. Therefore, a special kind of transition, a "reentering transition", will be introduced. If a reentering transition departs from an inner state that was previously interrupted, the state to be activated is the state the reentering transition points to.

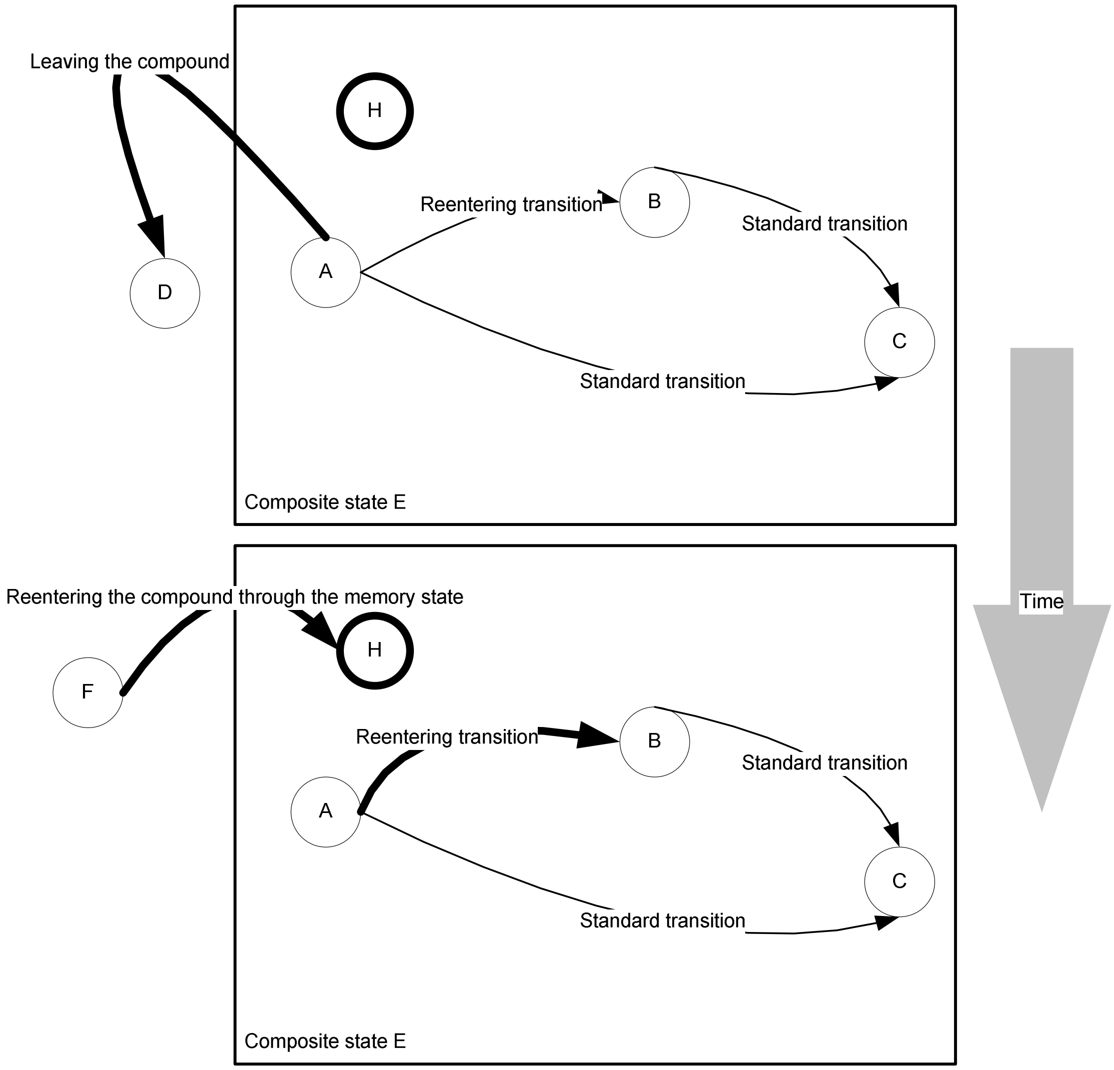


Figure 22 – Example of a reentering transition use. Bold transitions in the diagrams are being traversed. When the compound is interrupted when active at state A, and reentered through the memory state H, the state B will be activated, and any action attached to B will be carried on. This is because a reentering transition leads from A to B. A reentering transition thus indicates a state that shall be activated when the compound is reentered.

4.2.2 Extending the Statechart by "Reference Frames"

This extension of the state-chart formalism responds to the fact that, in IN, some composite states are reusable at several places, for example composite states that determine a waiting behavior. Virtual characters may wait for a user input at several places of an application, employing identical waiting behavior. In order to facilitate reusability, "reference states" are introduced. Reference states are placeholders for composite states. Employing references for composite states enables to simplify the graphical representation, because no borders must be crossed any more, and no difficult variable setting and conditions must be expressed that would allow to identify the state that was linked to the composite state (Figure 23). Reference frames (that are not orthogonally employed) share the same memory for states.

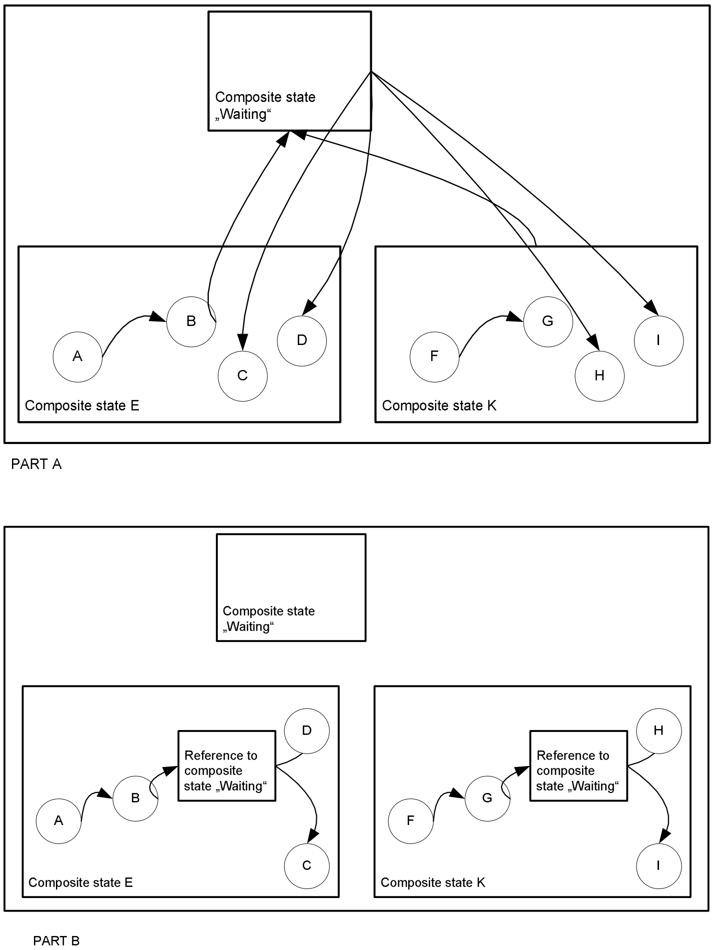


Figure 23 – Employing reference frames. In Part A of the figure, the standard usage of the "Waiting" compound is shown – note that complex guard formulae are required to bring the system back to the correct frame, after activation of the waiting behavior. In Part B of the figure, the same waiting behavior can be reused without a complex graph or complex conditions.

4.2.3 Extending the Statechart by Conditions and Actions Associated to Compounds

Other than with the original Statecharts, in the present system, compounds also can carry conditions and actions. A condition attached to a composite state is equivalent to adding the condition with AND to the condition of any transition pointing to or entering the composite state, cf. Figure 24. An action attached to a compound is carried out whenever the compound is activated. This addition stems from the possible interpretation of composite states as story structure elements, e.g. scenes or beats. A scene, for example, can be located at a certain virtual room, which will be loaded when entering the scene and unloaded when leaving it. And for this scene, it may be a prerequisite that a certain character was introduced. This can be expressed in the condition. This example also explains the motivation to employ also "postludes", i.e. actions that occur when a state (basic or compound) is left (deactivated). Equally, not only conditions (preconditions), but also post-conditions are employed, which controls whether a state may be left. A post-condition is equivalent to attaching the condition with AND to the conditions of the transitions that leave this state.

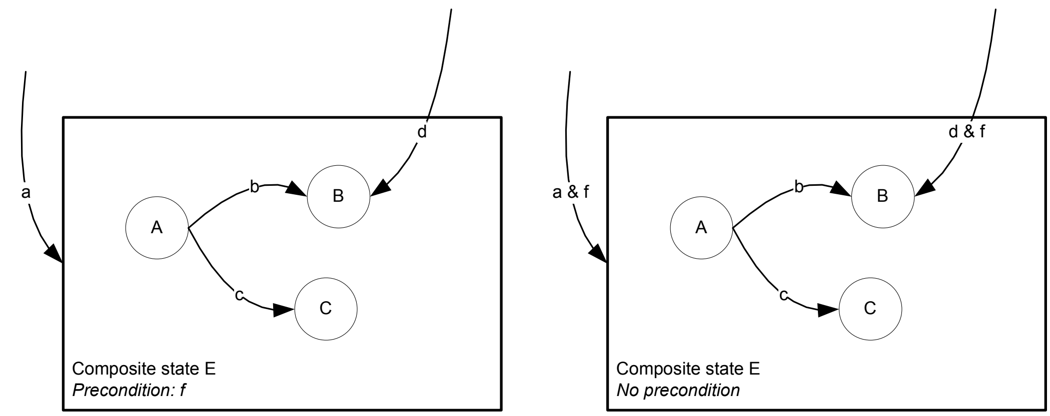


Figure 24 – Two equivalent compounds; on the left side, the compound carries a precondition f; on the right side, this precondition is distributed among the entering conditions.

4.2.4 Notation

The basic symbols of the formalism will be introduced. Formally, the elements of the present directed graph basis consists of a set Π of states,

$$\Pi = \{\sigma_1, ..., \sigma_m\}$$

and Ω represents the terminal (basic) states:

$$\{s_1, ..., s_n\} = \Omega \subseteq \Pi$$

The other states of $\prod$, i.e. $\Pi - \Omega$, are the compounds Λ. Compounds will be represented by an upper case S.

$$S_i \in \Lambda \Rightarrow \exists s_j \in \Pi : (s_j \subset S_i).$$

The elements of the set of transitions T are relations between sets of states[11] of $\prod$:
$t_i \in \mathrm{T}$,

with

$t_i =< p,q >$, and $p,q \subseteq \Pi$.

p,q are the source and sink of the transition, respectively.

A reference state $S_k^{R,j}$ is an identical copy of the state S_j it refers to. At runtime, all reference states share the same memory state. A reentering transition t^r is a transition of T that will be traversed when a state σ would be activated through the memory state H, and

$t^r =< p,q >$, with $p = \{\sigma\}$,

and the condition of t^r evaluates to true. When t^r is traversed, state σ is not activated by memory state H.

To each state σ and transition t, a set of conditions and actions can be associated. A condition c is a Boolean function of arbitrary complexity; no limitations to operations on the current event are made. An action is a procedure call of any

[11] Most transitions are relations between single states, but, due to orthogonality, some transitions can possess multiple sources or multiple sinks.

procedure available to the system. Outputting events is the elementary and normal action, but no restrictions are made on possible other actions. In particular, conditions and actions can access a common data base D.

4.3 A Hybrid Visual Formalism for Activation Engines

With the extended state-chart formalism presented at the previous section, already many kinds of interactive story applications can be created with a high degree of visual support. The next step requires that algorithmic approaches (Activation Engines) and their relationships to this visual formalism be addressed.

The reason to extend the directed graph approach with the features described in this section is that:

- No approach exists yet which responds to the fact that, in IN, migration to algorithmic, non-directed graph based strategies is likely to become necessary, but the exact algorithms and structures can only be discovered during the development process, when some result is already visible and the requirements and possibilities become more clear
- No approach exists yet which responds to the fact that, in IN, several sequence selection algorithms might be required at once, and that it is necessary to allow for experimenting with sequencing rules that are only valid within a very narrow range of a particular story
- No approach exists yet which responds to the fact that, in IN, reactions of different specification levels may be required, and that generic, possibly generative modules must be integrated to catch input and story events that cannot be catch by the main system

No generic visual representation form for any arbitrary kinds of sequence selection algorithm is possible. Algorithms, e.g. forward chaining rules, hierarchic planning, generative grammar approaches, require each their own approaches to visual editing tools.

However, some generic visual support can be provided by representing the basic element that shall be brought into a sequence. Let's call these elements "sequencing elements". The decisive move consists now in employing terminal or composite states of the Statechart as such sequencing elements. Because the same symbols are employed for the Statechart and for the sequence selection algorithms, a combination becomes possible.

4.3.1 Introducing Activation Engines

Let's call the module that chooses a sequencing element based on a sequence selection algorithm a "Sequencing Engine". Thus, the internal structure of each composite state can contain either a Sequencing Engine with associated internal sequencing elements, or a directed graph that controls the flow. For example, a "scene" might be a sequencing element of a generative grammar of story structures. I.e., each scene will be represented as a compound state of the Statechart. In this example, sequencing elements are composite states, because a scene itself contains internal structure, i.e., what happens in the scene. The scene might contain further beats, dialogue games, and other structured elements that require a distinct sequence selection algorithm.

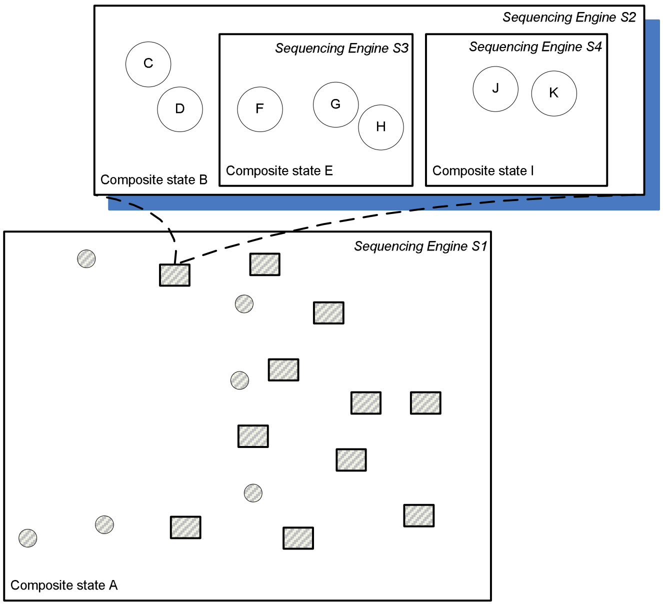


Figure 25 – A Sequencing Engine is attached to a compound; in this example, the engine S1 is attached to compound S1. The sequencing elements are located inside the compound. A sequencing element might be itself a compound, containing further Sequencing Engines.

This internal structure can then be represented in the same way as a set of sequencing elements of an inferior level with a corresponding sequence selection algorithm (or as a directed graph, cf. below). Thus, the basic visual representation for arbitrary sequence selection algorithms will consist of a hierarchy of sequenc-

ing elements, and to each sequencing element that contains an internal structure a Sequencing Engine can be attached. Figure 25 depicts this structure. Thus, arbitrary sequence selection strategies can be combined within the same formalism, including with a partial flow control by a directed graph (
Figure 26).

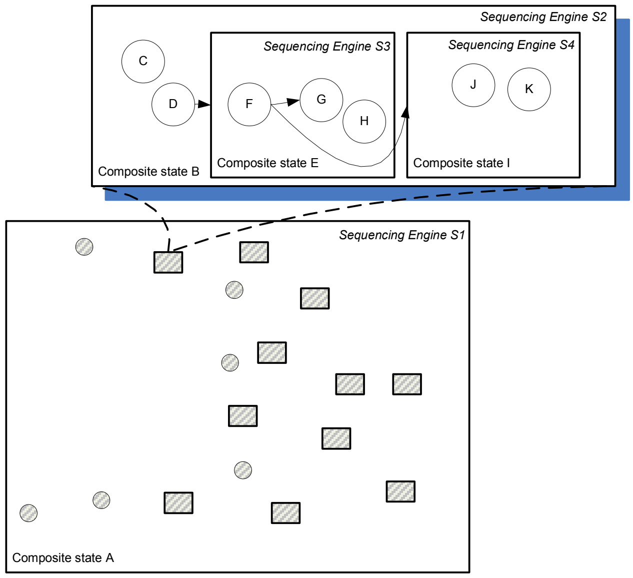


Figure 26 – A Sequencing Engine can also be combined with the directed graph of the Statechart. The semantics is defined by a delegation pattern.

Formally, the additional structural elements are a set of Sequencing Engines, θ, and a set of Fallback Engines, Ξ, with

$$\theta = \{m_i, \ldots, m_j\},$$

and $\Xi = \{f_i, \ldots, f_j\}$.

Each sequencing and Fallback Engine is associated to a unique compound S. Thus, these engines belong to a unique triple

$$< f_i, m_{j,} S_k >.$$

The association of compound and engines is assured by attaching meta-data to the compound.

A Sequencing Engine m_j can activate a state σ when $\sigma \subset S_k$, and when σ carries appropriate meta-data. Transitions may also carry meta-data.

Now, it will be shown how a combination of both directed graphs and Sequencing Engines within a compound can be employed. I.e., the next step consists in combining Sequencing Engines and the Statechart. A consistent semantics (or at least a semantic not less consistent than Statecharts[12]) can be achieved by introducing a delegation chain for the Sequencing Engine and the transition network (i.e., Statechart). The term "delegation" is employed from the point of view of the directed graph, which delegates the choice of the next scene to another module.

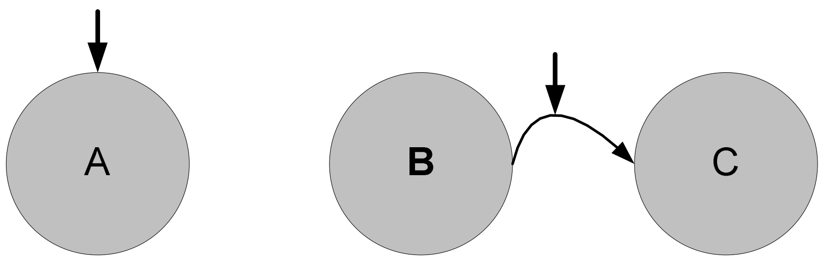


Figure 27 – The left states carries meta-data for a Sequencing Engine. This is indicated by the short arrow on top. This symbolizes that the state can be activated without traversal of a transition. When a transition carries meta-data (right side), the arrow is employed by analogy.

When representing a state that contains meta-data that can be read by an Activation Engine, a short arrow on top of the state will be employed from now on (cf. Figure 27). Transitions are also allowed to carry meta-data to be read by the Sequencing Engine. This allows providing the Sequencing Engines with additional structural information, e.g. meta-data attached to a transition can express that the source must have been activated before the sink can be activated. By analogy, the visual representation of such transitions also carries an additional short arrow on top (Figure 27).

[12] Some semantic problems of Statecharts are known, under distinct conditions; they are discussed in the literature, and are not subject of this work. Cf. [vonderBeek]

Figure 28 shows a simple example where the scenes are controlled by a dedicated Sequencing Engine for scenes (SceneEngine), and the beats by BeatEngines. The internal structures of the beats are directed graphs.

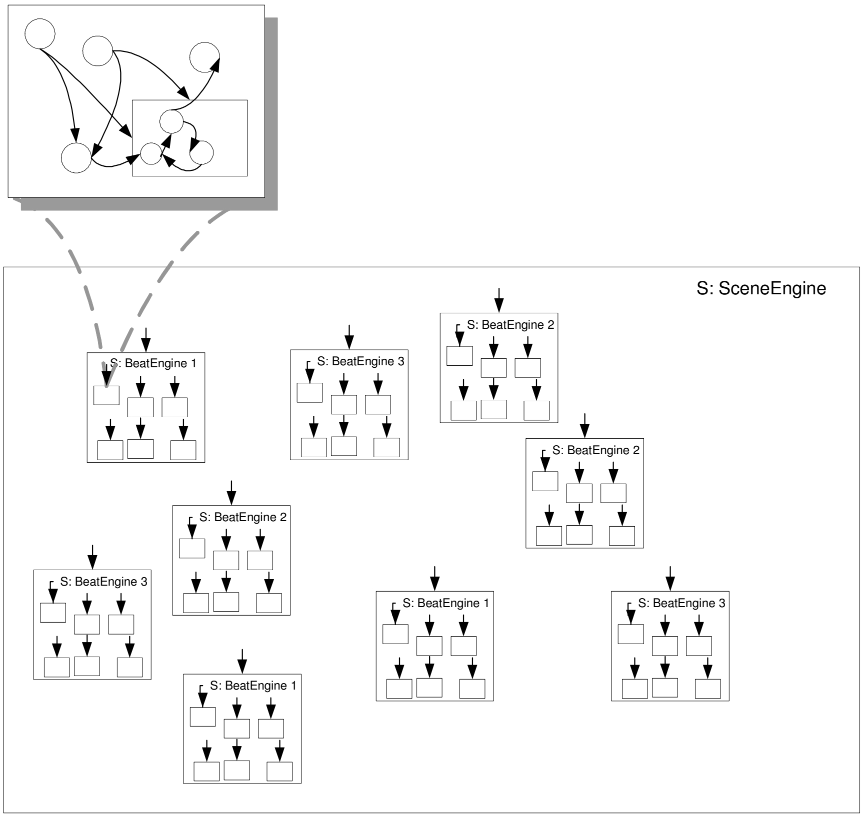


Figure 28 – The Hybrid Control Formalism allows for combinations of sequence selection algorithms with directed graph control. In this example, the scenes are the direct daughters of the top most compound, which selects the scene-compounds with the aid of the SceneEngine for sequence selection. The scenes contain a further hierarchy controlled by a Sequencing Engine, the BeatEngine. The daughters of this level are beats. In this example, the beats themselves contain a structure controlled solely by a directed graph.

Figure 29 illustrates the migration process from the directed graph to the partial control by Sequencing Engines. It is assumed here that, at the beginning of the wicked creation process, the algorithms and requirements for sequence selection were not known, and therefore the first creation step employed only the directed graph.

In order to avoid ambiguities, when the directed graph of the Statechart is employed to determine the subsequent state, it will be assumed that a “Directed Graph Engine”, or simply “Graph Engine”, is activating the state. Let’s “Activation

Engine" be a term that covers both Sequencing Engines and the Graph Engine. I.e., an Activation Engine is either a Sequencing Engine or the engine of the Statechart proper, and a Sequencing Engine is an engine that actives a state of the formalism, but independently of the graph; a Sequencing Engine does not examine the transitions, but works on facultative meta-data of states and of transitions.

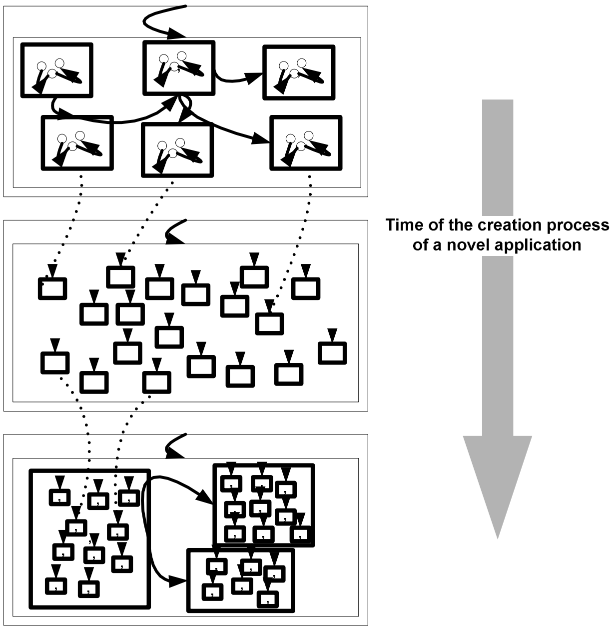


Figure 29 – On the top, a preliminary step during prototyping and experimentation. In the middle, an advanced development stage is shown, where part of the transitions were replaced by a Sequencing Engine and metadata. At the bottom, a succeeding development phase where a new grouping was found. The dotted lines show identical states, in the different stages and representations.

4.3.2 Combining Statecharts and Activation Engines

Employing a delegation pattern for the Activation Engines of a certain compound means that the Sequencing Engine will be called after the Graph Engine, i.e., a new event will be handed over to the Sequencing Engine, if no condition of the graph is evaluated to "true". Figure 30 shows the delegation pattern.

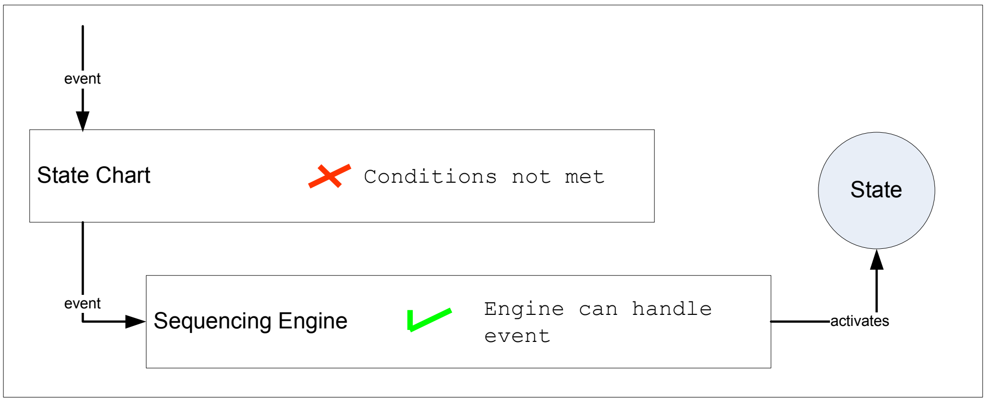


Figure 30 – An event is send first to the Statechart, which can delegate it to the attached Sequencing Engine.

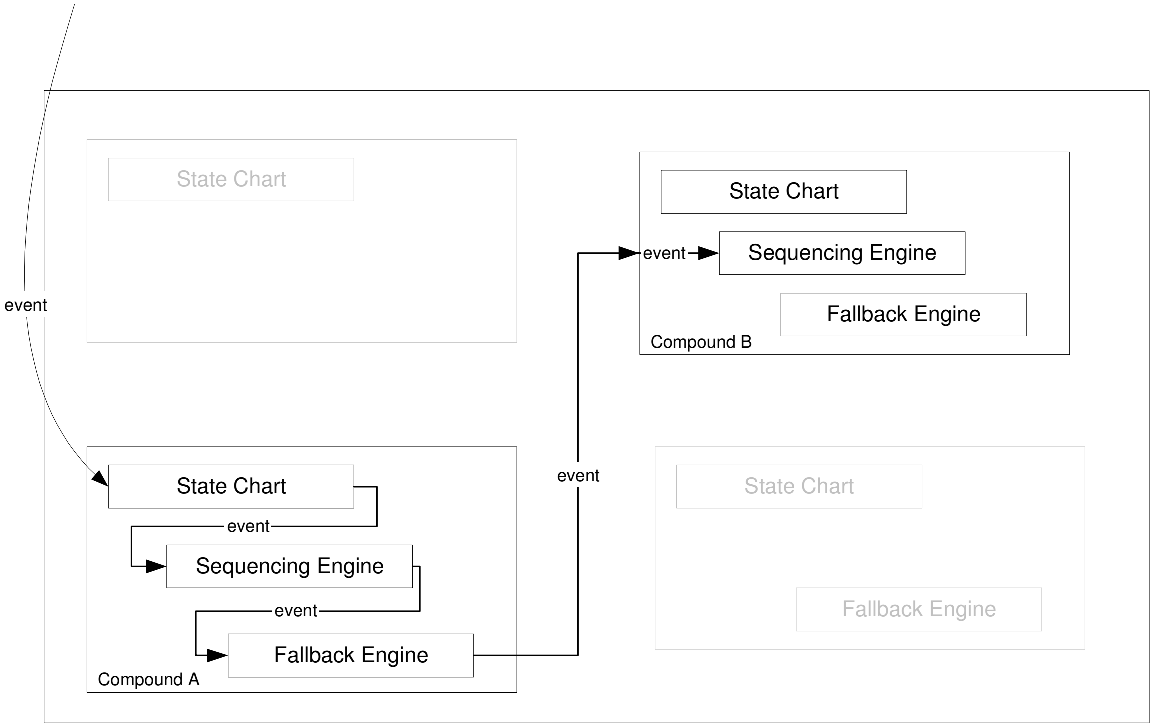


Figure 31 – In principle, every Sequencing Engine and Fallback Engine is allowed to send events to specific other compounds. In this example, the other compound, B, is not active, and thus the State-chart part cannot be called. This is intended to simulate function calls to other parts of the formalism.

A Sequencing Engine or Fallback Engine (cf. below) may send arbitrary messages and pass over events to any other specific compound. Thus, the delegation chain is just a special case, but in practice (cf. Section 6) it turned out to be yet the most important example, and is therefore on focus here. It has been facilitated by the software environment that will pass over an unhandled event to the parent compound B of compound A when an event cannot be handled by the compound A.

The more general approach of allowing Sequencing Engines and Fallback Engines to send to specific other compounds is mentioned here as part of future work that still requires some practical evaluation within concrete applications. Figure 31 shows this feature. Fallback and Sequencing Engines can consummate events, and thus, can interrupt the chain, if required.

The reason to employ a Sequencing Engine after the directed graph lies in the possibility of defining different levels of specificity. I.e., the directed graph is employed for the accurately predicted behavior, the Sequencing Engine when the graph fails. When no transition leaves an active state, the Graph Engine will not be able to activate any further state, leaving the task to the Sequencing Engine. Accordingly, if no appropriate metadata is attached to a state, within a certain compound, there is nothing to do for the Sequencing Engine. Sequencing Engines (and Fallback Engines, cf. below) are facultative elements.

The next step consists in further integrating the hierarchies. Until now, only a single compound and its internal states were regarded. Usually, in Statecharts, the hierarchies are evaluated simultaneously. In Statecharts, when a terminal state is active, every parent state is equally active; when an event occurs, the transitions of the active terminal (basic) state, and of all interrupts of its parents will be evaluated.

Employing a similar logic, every Sequencing Engine of any parent state will equally be called for handling an event. However, an order of calls shall be introduced. The inner compounds are called first. I.e., a Sequencing Engine attached to a child compound has priority over a Sequencing Engine attached to a parent compound. This again is meant to ensure that handlers of different specificity can be employed. I.e., the Sequencing Engine of a child state is more specific and thus has priority over the Sequencing Engine of the parent state, which contains more generic measures. This is the *external delegation*, in contrast to the *internal delegation* that occurs within a compound. External delegation was originally introduced to enhance the scope of the shallow NLP features, by enabling detection of thematic changes of the user. Figure 32 shows an example of an external delegation. Different usages are possible, e.g. for control of thematic ranges, or for producing typical behaviors for certain states of mind and emotions. For example, the father compound can be interpreted with focus on conversation topics, e.g. as containing response patterns for chatting about the weather, and a daughter compound contains specific patterns for chatting about some important story specific issue, e.g. marriage problems.

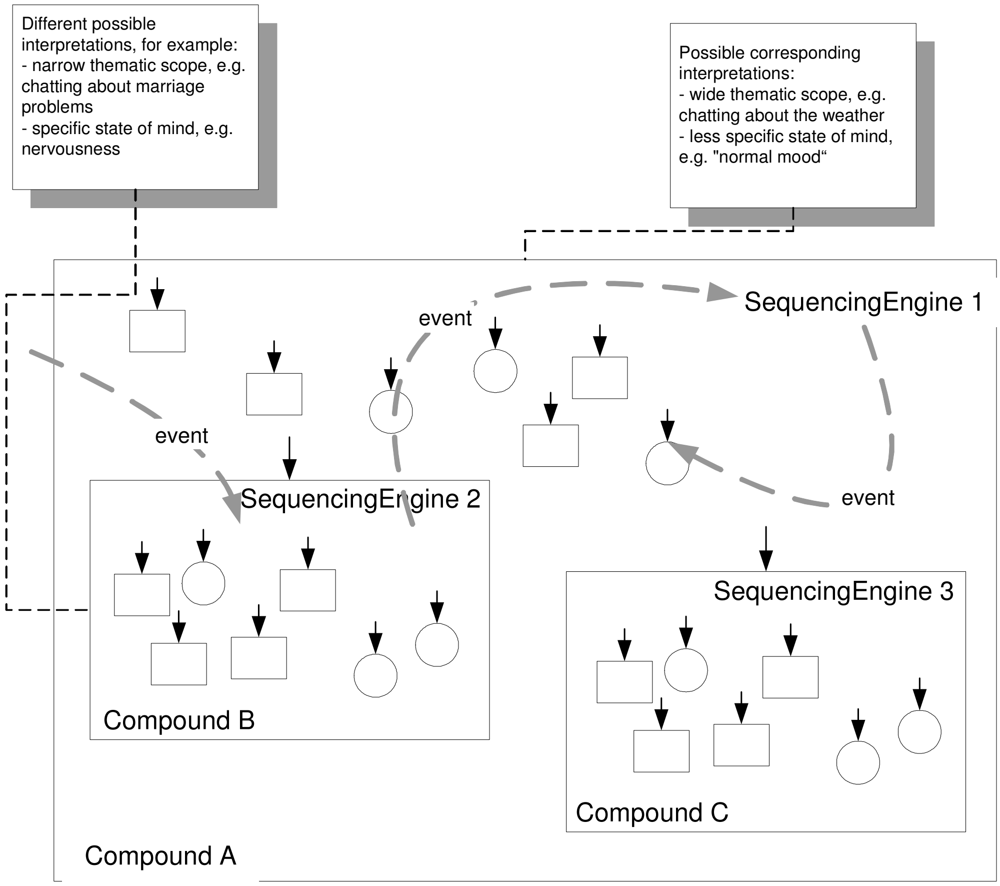


Figure 32 – Employing an external delegation chain. An event that cannot be handled by Compound B and its Sequencing Engine 2 is send to the immediate superior Sequencing Engine 1.

Now, it is necessary to define the scope of a Sequencing Engine. In the first publications on these issues, strict encapsulation was assumed, i.e., a Sequencing Engine “saw” only immediate daughter states of the compound the Sequencing Engine is attached to (cf. [Iurgel04a]). In practice, this turned out to be too restrictive. Sometimes, it is useful to reorganize the structure of the hybrid system, for example moving a set of states into a new child-compound. With strict encapsulation, these states wouldn't be any more visible to the original Sequencing Engine. Thus, every daughter state of a compound, independently of whether it is an immediate child or an intermediate descendent, is now visible to the Sequencing Engine (Figure 33).

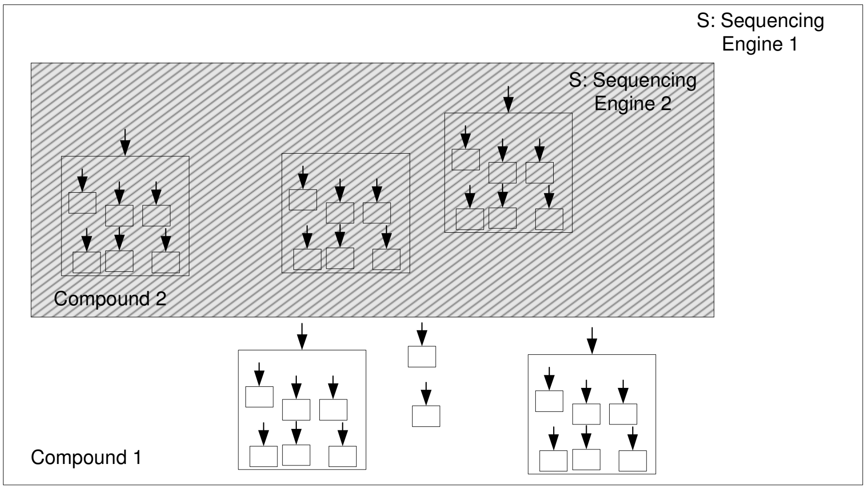


Figure 33 – The scope of a Sequencing Engine encompasses every direct and indirect daughter state with appropriate meta-data. In this example, the Sequencing Engine 2 cannot activate the direct daughter states of Compound 1.

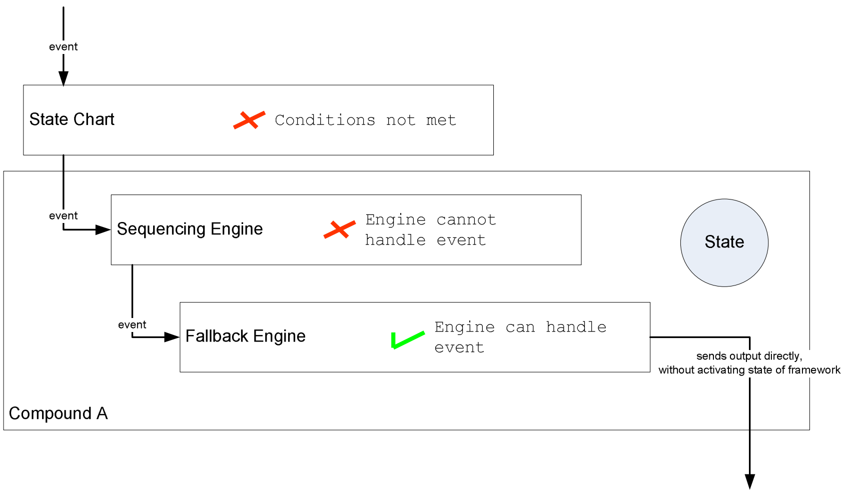


Figure 34 – The Fallback Engine sends the control message to the virtual environment directly, without activating a state of the formalism. The Fallback Engine is called after the Sequencing Engine, and is equally attached to a compound of the formalism.

Another element of the formalism is the "Fallback Engine". A "Fallback Engine" is an external module that is called when everything else fails. Like Sequencing Engines, it is facultative. A Fallback Engine outputs events directly, without activating a state of the formalism. It is a measure to allow the authoring team to integrate a generic module into an application built with the formalism that equally can control the virtual environment. In the Art-e-Fact example below (cf. Section 9.1), an external chatting module is employed as Fallback Engine, so that the available databases of an external chat engine can be reused. A Fallback Engine is evaluated last, i.e., when no transition and no Sequencing Engine could activate a state (Figure 35 and Figure 36).

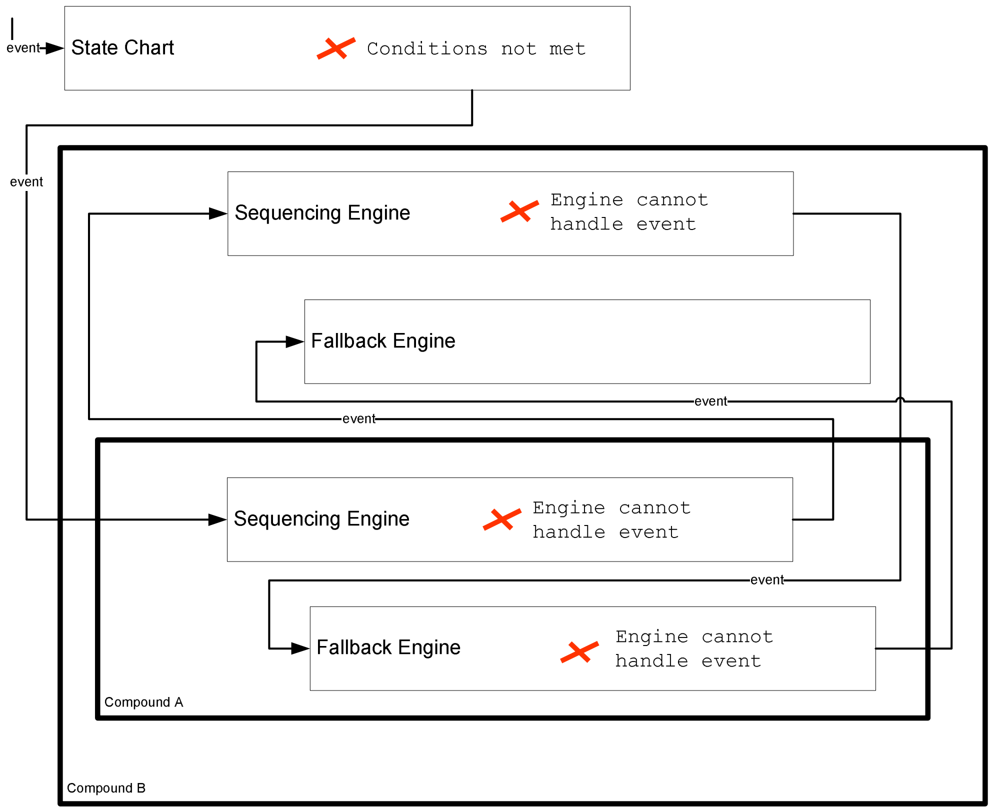


Figure 35 – Controlling the hierarchies of compounds with attached Sequencing and Fallback Engines. In this example, compound A is an immediate daughter of compound B, and compound B is the root of the hierarchy. If an event cannot be handled by a compound, it is transmitted to the father compound. Fallback Engines are served last, when no condition of a directed graph or Sequencing Engine can handle the event. Then, the hierarchy of Fallback Engines is served bottom-up.

Fallback Engines are attached to compounds, in the same way as Sequencing Engines. When a series of Fallback Engines is available in the active hierarchy, the child engine is called before the parent engine. As described in the context of the Sequencing Engines, this allows for a chain of specificity, i.e., the lower Fallback Engines are more specific and thus have priority over the engines higher in the hierarchy (Cf. Figure 35 and Figure 36).

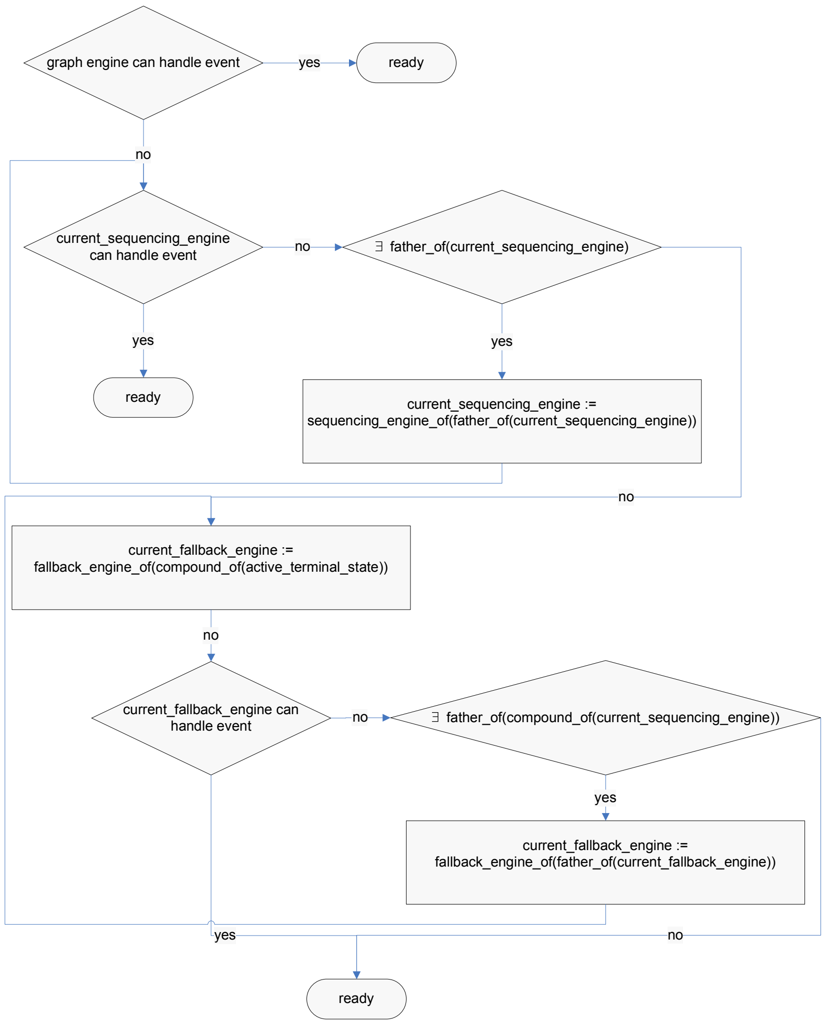


Figure 36 – A diagram representing the sequence of delegation of events to Sequencing Engines and Fallback Engines, employing pseudo-code.

In order to allow for autonomous activation by any Activation Engine, e.g. by independent timeouts, after a fixed time interval after the last event, the system will generate internal “empty” events. An Activation Engine is only allowed to activate a state if it receives an event, either empty or from the outside of the system. The Activation Engine passes over the event to the next engine of the chain, if it does not use it to activate a state and does not consummate it. Thus, “events” are employed as tokens, and an engine may only activate a state or send an action or pass over an event if it possess the “token”, otherwise it is required to pass over the token to the next engine.

Another addition to the traditional Statechart formalism are “virtual events”. *Virtual events* are events generated by some Activation Engine in order to check whether some activation formalism possesses an appropriate response to some event.

This addition is necessary for reproducing meta-communicative behavior, e.g. for a virtual character to be able to comment “OK, let’s change the theme!”, before answering to a question that indeed has changed the conversation theme. A virtual event shall only be send to an immediate daughter of the compound state the sending activation formalism belongs to.

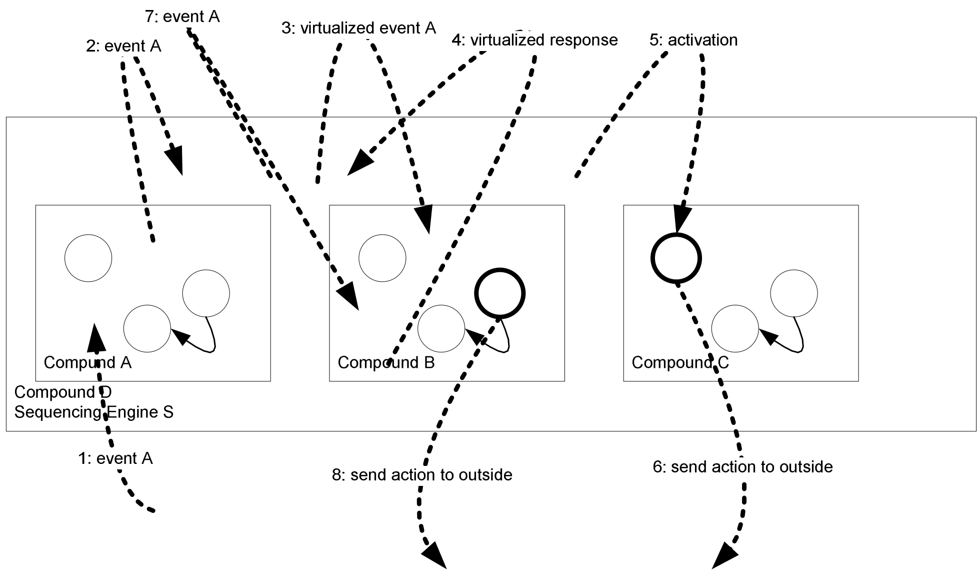


Figure 37 – Employing virtual events to check whether some subordinate compound is able to react appropriately to an incoming event. In this example, the event A cannot be handled by compound A, which is initially active; the Sequencing Engine of the father compound, S, determines that compound B would be able to react; before activating the correct state of the compound B, S activates a state of compound C, which contains e.g. orientation information. This enable a virtual character to say for example “oh, you want to change the theme!”, before reacting to the input.

An event is virtualized by being tagged as virtual, and the possible action of the compound (i.e. of the activation formalism that it contains) is not sent to the outside of the system, but as a *virtual response* to the Activation Engine that has send the virtual event. If the subordinated formalism chooses no action, the original virtual event is sent to the sink (the sending Activation Engine). The distinctive feature of virtual events is that the compound that responds to it will reset itself, including every Activation Engine and the database values, at the moment when a non-virtual event arrives at this compound. Thus, the virtual event has no consequences; it is equivalent to a “what-if” question, or a query. Figure 37 shows an example of the usage of external delegation and virtual events in order to activate orientation utterances.

This is the basic formalism that will enable incremental experimentations and combined use of different sequence selection methods.

4.4 Conclusion of the Presentation of the Hybrid Formalism

In the preceding sections the theoretical bases for controlling the flow of the formalism was presented. Harel's Statecharts form a basis of the directed graph aspects, and several dedicated extensions improve the usability of this generic formalism for use within IN.

It was also shown how a hybrid system can be constructed, reusing the hierarchic structures of the Statecharts, and introducing a delegation chain. It has become clear that employing a hybrid approach presents the following advantages:

Support of an incremental authoring process. Here, the author may start with a directed graph and gradually and possibly only partially move to an algorithmic approach, as structures, application design, and algorithms become evident and the graph approach too complicated. This implies cooperation of content creators with computer scientists.

Support of alternate use of paradigms. Assuming that an algorithmic approach will only be viable for certain parts or layers of the narrative parts, the directed graph approach can be employed for the others

Support of combined use of paradigms. It is possible to handle events with a delegation chain, where an event will be handed over to different subsystems, until some subsystem is able to handle the event. This creates a priority chain of subsystems, which allows for the creation of subsystems of different levels of precision.

Support of team work. With a hybrid approach that offers visual representations, content creators without programming background can employ the directed graph parts and the Sequencing Engines by assigning metadata to the states, with the help of a GUI. Computer specialists will program the sequencing and Fallback Engines, and the GUIs that support their use. Thus, a clear separation of roles and responsibilities is easy to achieve, so that content creators can focus on content, and do not have to program.

Support of experiments and of tinkering. Clearly defined processes and encapsulation allow for experimentations with different control strategies, including experiments with “ad hoc” algorithms, models and programs to control the flow of the story.

Thus, the afore developed Hybrid Control Formalism presents every advantage of the complex directed graph control of Statecharts, but extends the formalism of the directed graph to cope with specific demands of authoring for IN, and decisively increases the scope of applicability of the formalism, incorporating specific demands of the “wicked” situation within IN research. It is, in fact, the first formalism of its kind specifically designed for IN research.

Future work on this formalism should encompass

- Visualization methods for the structures of the Sequencing and Fallback Engines.
- Visualization methods for the runtime processes of the directed graph, the Sequencing, and Fallback Engines.
- Usability studies in the context of large projects.

5 “Cyranus”: An Extensible Framework for Visual Formalisms for Authoring

- This section presents “Cyranus”, a complete, modular software environment for authoring interactive narratives developed as part of this dissertation. Cf. also [Iurgel03d], [Iurgel04b], [IurgelHoffmann04], [Iurgel05c], [IurgelHoffmann06] for details.

5.1 Introduction

In Section 4, the principles of the Hybrid Control Formalism were developed and presented. In the present section, the software environment for IN that implements this formalism is described. The specification, specialization, and implementation of the hybrid framework is called “Cyranus”. It is a complete software environment with a complex authoring tool, with managers for several input devices (e.g. gesture recognition, text input, prop recognition, emotion recognition), animation modules for emotionally expressive character with lip-synchronized synthetic speech, different 3D and 2D viewers for displaying the virtual environment, a client-server system for multi-player internet games, and others. Its main elements will be presented in this paragraph. For some additional details on the components cf. [Iurgel03d] and [IurgelZiegler]. In the following sections, the authoring tool and the formats will be discussed in more detail. Cyranus has already proven its efficacy and generality within several projects (cf. Section 6 for some examples), and its formats and some of its principles have already served as basis for other authoring environments with different focus (within the INSCAPE and U-CREATE projects[13]).

5.2 A Reference Architecture for Storytelling

A complete flexible runtime environment is part of the framework. I.e., interactive stories can be immediately built, employing the Hybrid Control Formalism described in Section 4. A reference architecture was developed, which is depicted in Figure 38 (cf. [Iurgel03d], [Iurgel04a]). The reference architecture is the usual architecture to start with, when an application design idea is on focus. It is, with minor changes, the architecture employed for the projects described in 6.1, 6.2, and 6.3.

---

[13] Cf. www.inscapers.com and http://www.zgdv.de/zgdv/zgdv/departments/z5/Z5Projects/u_create.

The module of Cyranus that implements the Hybrid Control Formalism is called the Narrator. Note that this reference architecture does not employ autonomous virtual characters. The Narrator is the only module that controls the flow of the application. This is the intended standard usage of the Narrator. Given the generality of the module, it can be employed in different ways, e.g. as part of a virtual character's control module, and it can also direct autonomous virtual characters. In fact, this second usage was employed for the Virtual Human project (cf. below, Section 9.4, cf. Iurgel06a], [GöbelIurgel07]).

The reason to define a standard architecture and to create a fully functional platform based on this reference architecture that does not employ other control modules is the intended usage as experimentation platform. Thus, it shall not be assumed that other modules are available, or have to be created first. The creative process can start right away with the transition network (as Statechart), and further modules can be inserted afterwards, if necessary, to enhance the performance of the Statechart. If several autonomous parts are required, they can be modeled with orthogonal compounds of the hybrid framework.

The Narrator contains, as shown in the figure, as further sub-components a JessEngine, ALICE, and a CaseEngine. Those are examples for Sequencing and for Fallback Engines, respectively. ALICE is a Fallback Engine for chatting. It is a slightly adapted version of the free ALICE chatbot "Program D", which has proven to be efficient in various projects[14]. The JessEngine employs a generic rule based system[15]. Though mainly known for its forward-chaining capacities of a production system (i.e. for the possibility of employing rules that are applied one after the other, on a container of facts), it also offers powerful planning faculties. Thus, it is a suitable generic Sequencing Engine. The CaseEngine is the example of an engine that offers a Case-Based Reasoning Sequencing Engine. Section 7 will dwell on the use of employing CBR in IN.

The modules are (i) SceneObjectHandlers for handling scene objects. This hand-

[14] Cf. www.alicebot.org

[15] Cf. herzberg.ca.sandia.gov/jess/ for the underlying rule-based system

ler usually loads and unloads objects, but it can also be employed for controlling the behavior of complex objects, e.g. the autonomous movements of cameras. The (ii) DeviceHandlers offer the interfaces to the devices, and possibly carry some internal logic, for example for emotion recognition, when the input from the camera needs some filtering and analysis. The standard viewer is the Avalon[16] viewer of ZGDV e.V., which functions according to VRML and X3D specifications[17].

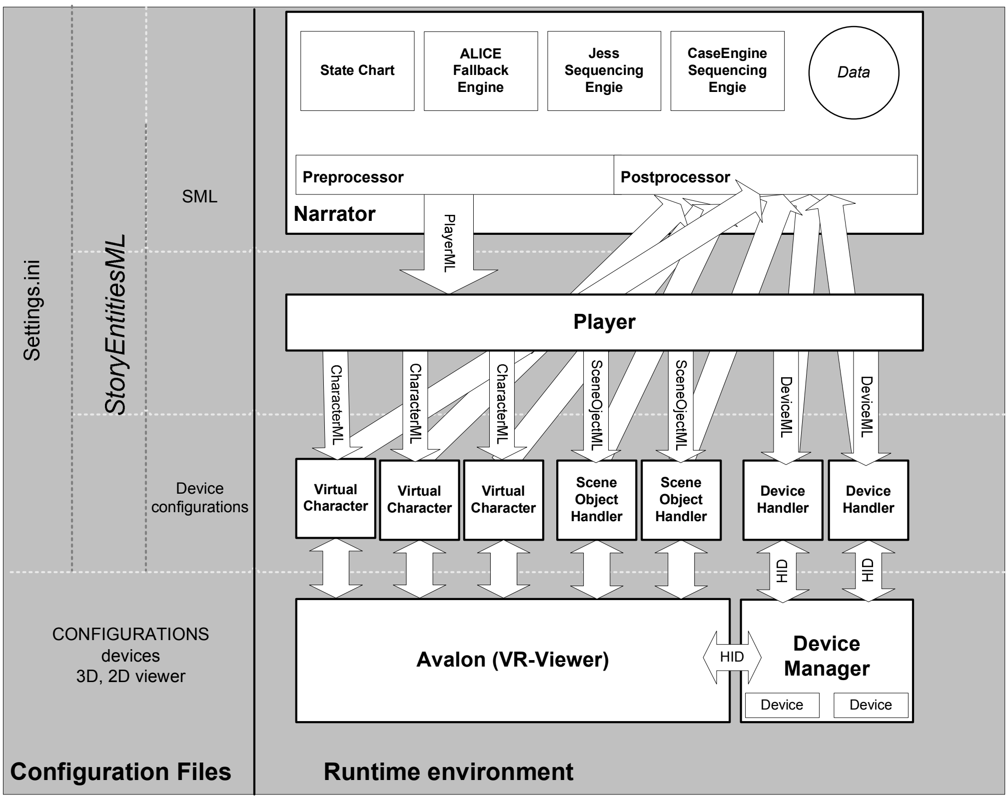


Figure 38 – Reference runtime architecture of Cyranus.

The main requirement for animation was expressive quality. The argumentation is the same as for dialogue management and interaction control: Behavior blocks that are not synthesized allow for exact control, and therefore are given priority over synthesized, modeled approaches. Thus, the problem of animation was reduced to a problem of combination of smaller, basic animation chunks. A disad-

16 Cf. www.zgdv.de/zgdv/zgdv/departments/z2/Projekte/AVALON

17 Cf. www.web3d.org/x3d/

vantage is the lesser flexibility and variability.

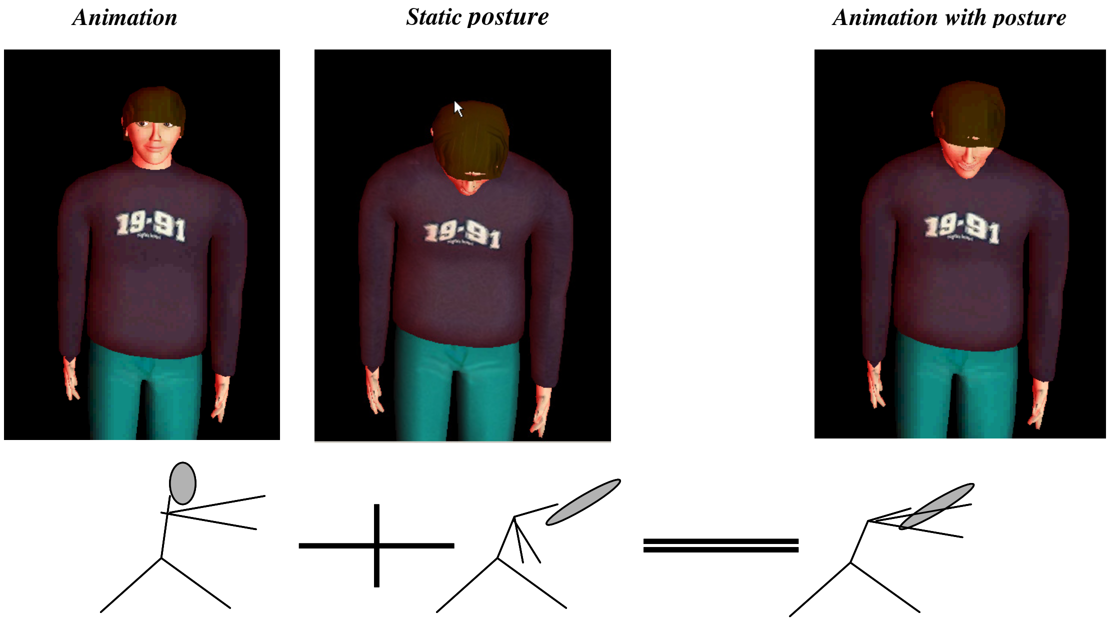


Figure 39 – Runtime adaptation of the motional e xpression of the body is achieved by adding the rotations of a static, emotionally expressive posture, to the angles of the animation.

For animating the virtual characters, therefore, a move graph was chosen. The move graph defines the animations that can follow each other. The animations are stored as joint movements of an H-Anim skeleton[18]. In order to extend the connectivity of the animation graph, slerping ("spherical linear interpolation" for the rotation interpolation of joints, cf. [Shoemake]) is employed. For each animation, metadata is defined that specify the duration of the onset and of the offset of the animation, i.e., of the parts of the animation that initiate the movement, until its peak (the "stroke"), and the finishing of it. Onset and offset are the parts of the animation employed for slerping (Cf. [Iurgel03d]).

For emotional expression of the face, morph targets were used. For emotional expression of the body, a method has been developed for using static postures to modify the postures of animations ([Iurgel03d], [ValaPaiva])[19]. An emotional static posture (e.g. sad shoulders, head down) is added at runtime to the animation, ac-

[18] Cf. www.h-anim.org

[19] Vala et al. [ValaPaiva] have published on a parallel, but independent development.

cording to the emotion of the virtual character, by adding the expressive rotations, i.e. the rotation difference between an emotional posture and a neutral posture, to the rotations of the animation (Figure 39). This method produces clear and strong non-realistic emotional expressions that respects the characteristics of the animations, and avoids the expenses and problems of the usual motion interpolation. As with motion blending based on slerping, the results must be checked visually, to avoid possible self-intrusion and other artifacts.

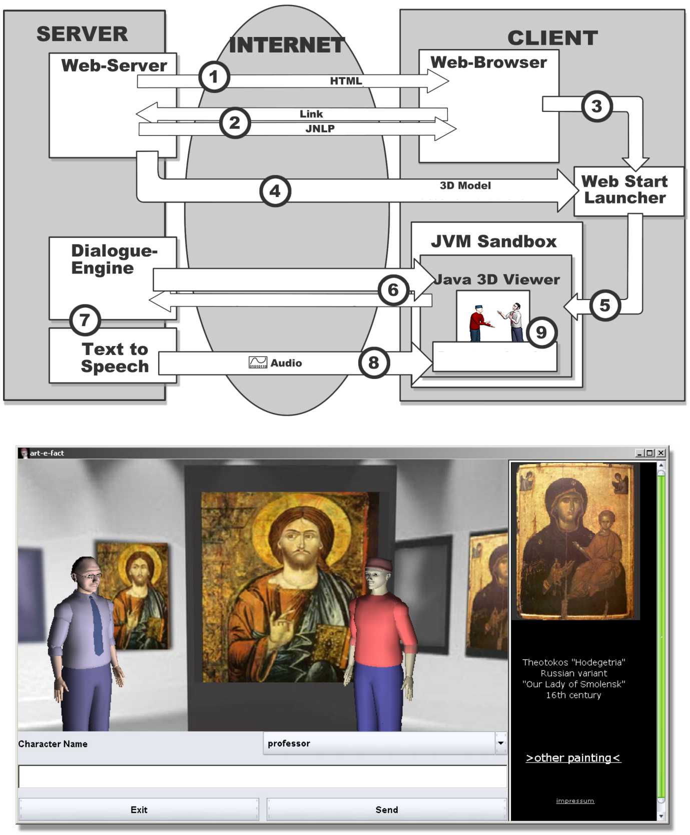


Figure 40 – The framework consists of several other modules, e.g. of a Java-Webstart client-server system for use over the internet. Its architecture is depicted in the architecture on top. Below of it is a screenshot of the Java3D-viewer employed with the Webstart framework.

The Cyranus framework consists of a series of further components and reference implementations of modules. It includes a client-server system for multiple player online games, and is modularly composed to allow e.g. for exchange of the 3D viewers and text-to-speech components. Figure 40 shows the architecture and a screenshot of the Java-Webstart version of the system.

A central component of the Cyranus framework is the authoring tool [Iurgel06a]. It contains a partial implementation of the Statechart paradigm. Parallelism is not implemented in the current version of the tool, and arrows cannot cross borders. Some workarounds are available, though.

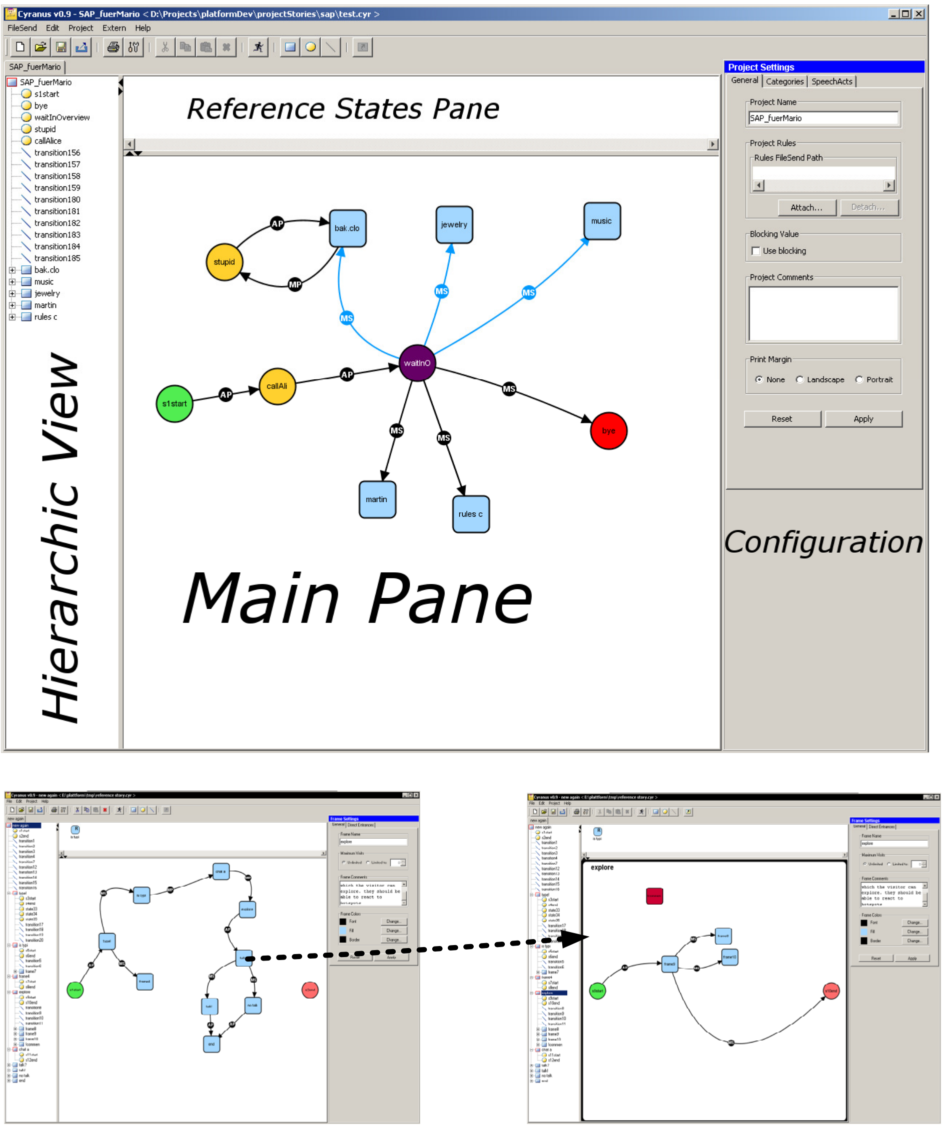


Figure 41 – On top, a screenshot of the authoring tool of Cyranus. By double-clicking on a compound, the user opens it to see the internal structures (shown below).

The prototype version of the system supports the hierarchies of the Statecharts, and the integration of Sequencing and Fallback Engines, and offers a variety of supporting features. Figure 41 depicts a screenshot. There, squares are composite states, and circles are terminal states. States that contain meta-data for a Sequencing Engine (i.e. that can be activated without a transition leading to them) possess an arrow on top, cf. Figure 42.

The reference pane on top of the main pane is employed for storing reference states; these can be drag-and-dropped into the main pane. The configuration tabs on the right side were made such as to support being exchanged on a project basis. A configuration file enables to load different tabs, depending on the requirements. Software engineering measures facilitate the exchange, preserving full modularity of the tab-modules; e.g., the controller of the tab for action definition is solely responsible for the creation and storage of action formats.

This is a software engineering based response to the “wickedness” of IN, and to the necessity to allow content creators to enter the production process without acquiring deep computer science skills: The tabs that configure metadata, actions and conditions, among others, are easy to exchange and adapt as the ideas and experiences grow, during the creation process.

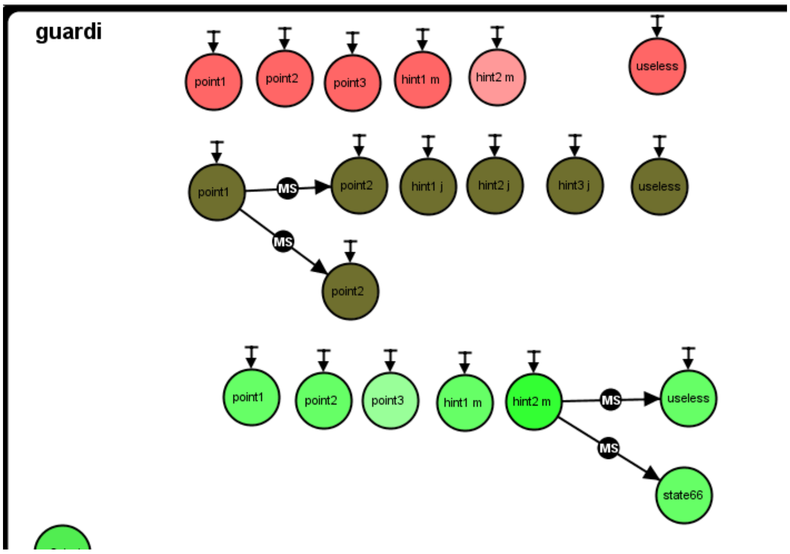


Figure 42 – States that contain meta-data posses an arrow on top.

Several different configuration tabs were made for specific designs. For example, Figure 43 depicts a script editor for scripting the virtual environment, the utterances and movements of the virtual characters.

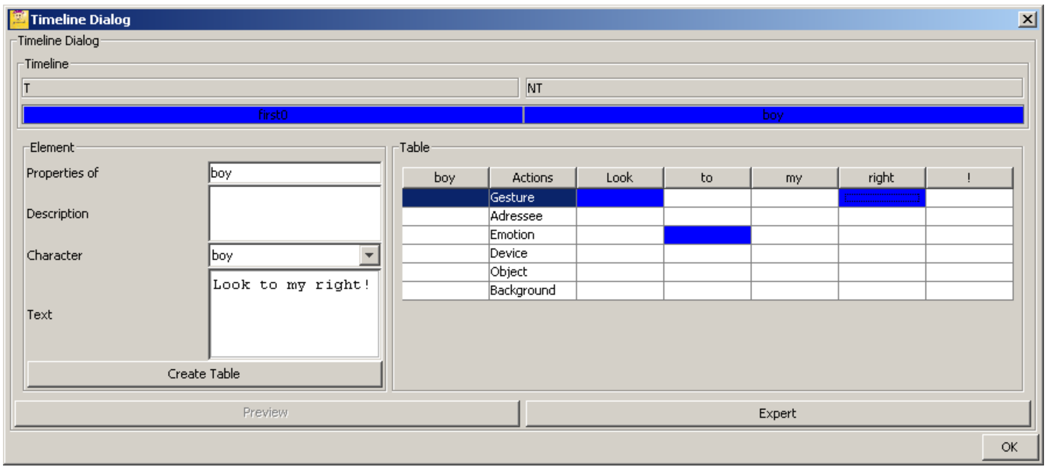


Figure 43 – A (facultative) script editor associated to the action tab.

5.3 Formats

This section describes the formats for runtime communication and for storage of stories that were developed as part of the Cyranus authoring framework. For readability, the explanation in this section will work on examples and with informal annotations. Every format employs XML.

The different modules communicate with each other employing XML. The messages use the dedicated format CML (Communication ML), which enclose the messages and contains addressing information:

```
<CML>
      <addressing from="Narrator" to="Player">
            XML-Message
</CML>
```

The Player of the reference architecture distributes the commands of the narrator among the subordinate modules. The format that the Player receives is called PlayerML, and is a script based on the format RRL [PiwekKrenn] for controlling virtual characters[20]. To the original RRL, the possibility of controlling random scene objects other than virtual characters was added, and the original RRL was simplified. The important feature of RRL to adopt are the tag for sequential and parallel script parts, which are essential for controlling the scene, and the possibility of aligning gestures to words. An example follows:

[20] Cf. NECA-Project

```
<player>
    <seq>
        <par>
            <character name="professor">
                <dialogAct>
                    <sentence id="s_1122507720459">
                        <word id="w1_1119606922599">choose</word>
                        <word id="w2_1119606922599">from</word>
                        <word id="w3_1119606922599">one</word>
                        <word id="w4_1119606922599">of</word>
                        <word id="w5_1119606922599">the</word>
                        <word id="w6_1119606922599">four</word>
                        <textSign id="ts0_1119606922599">!</textSign>
                    </sentence>
                    <animation alignTo=" w1_1119606922599"
                             alignType="starts" id="g1119606932037"
                             name="prof_bendpose_point_at_icon_0967_1003"
                             type="gesture"/>
                      <animation alignTo="w6_1119606922599"
                      name="prof_bendpose_present_icon_1003_1045"
                      type="gesture"/>
                </dialogAct>                </character>
            <device name="pointingRecognition">
                <enable id="1120041546975">
                    <configuration name="fivebuttons"/>
                </enable>
                <appear id="1120041546975"/>
            </device>
            <device name="propDetector">
                <disable id="1121958087911">
                    <configuration name="SPOT"/>
                </disable>
                <disappear id="1121958087911"/>
            </device>
            <object name="pfeil_links">
                <appear id="1122382744297"/>
            </object>
        </par>
    </seq>
</player>
```

The device tag starts devices and configures them. An example with explanations can be found at the description of the Art-e-Fact project below (cf. Section 6.1).

The object tags let scene objects appear and disappear. The name attributes denote modules.

The main storage format for stories is the SML, the Story Markup Language. It describes a particular story. It builds upon the transition network definition of W3C, XTND[21], and was chosen because of the neat integration with XEXPR[22], a functional programming language in XML that is employed both in the conditions and the actions of the network. Thus, a single, very generic and powerful subsystem can be employed for these parts of the platform, simplifying development and use (cf. [Iurgel03d]). No other format offers this generality and expressive power for condition and actions.

The original W3C proposal does not support parallelism, hierarchies, priorities, metadata, and some other specific data for the hybrid IN framework. These were added for Cyranus (cf. [Iurgel03d]). Only additions and change to the W3C specification will be explained in this section.

A snapshot of an SML-story XML is as follows:

```
<SML>
  <frames start="frame1TS1102082300687">
       <frame id="frame1TS1102082300687" name="Project1"
                      start="state1TS1102082300687" end="state2TS1102082300687"
                      concurrence="XOR">
       </frame>
       <frame id="frame2TS1102082300687" name="pointin"
                      owner="frame1TS1102082300687" start="state5TS1102082300687"
                      end="state6TS1102082300687" maxvisits="-1" concurrence="XOR">
              <metaData>
                <sequenceEngine name="jessEngine">
                      <rulePath="/../story/rules/jess">
                </sequenceEngine>
              </metaData>
              <ruleData type="scene"/>
       </frame>

(…)

  <states>
       <state id="state3TS1102082300687" name="init"
                      owner="frame1TS1102082300687" maxvisits="-1">
                      <metadata>
                             <sequencingData engine="Hot Spot Engine">
                                    <spatialOrientation priority="2"
                                     hotspot="Maria" currentPointing="immediateRight"/>

                             </sequencingData>
                      </metadata>
```

[21] Cf. www.w3.org/TR/xtnd/
[22] Cf. www.w3.org/TR/xexpr/

```
                    <preconditions></preconditions>
                    <postconditions></postconditions>
                    <prelude>
                           <send>
                                  <player>
                                         <seq>
                                                <object name="hotGuardi">
                                                       <appear id="1093620539059"/>
                                                </object>
                                         </seq>
                                  </player>
                    </send>
             </prelude>
             <postlude></postlude>
      </state>
(…)
      </states>
        <transitions>
             <transition id="transition1TS1102082300687" name="transition1"
                    owner="frame1TS1102082300687" from="state1TS1102082300687"
                    to="state3TS1102082300687" reentering="false" priority=”1”>
                    <metadata/>
                    <preconditions>
                           <event>
                                  <message value="regionChosen">2</message>
                           </event>
                    </preconditions>
                    <actions></actions>
             </transition>
             <transition id="transition2TS1102082300687" name="transition2"
                    owner="frame1TS1102082300687" from="state3TS1102082300687"
                    to="frame2TS1102082300687" reentering="false" priority=”1”>
                    <preconditions></preconditions>
                    <metadata/>
                    <actions></actions>
             </transition>
      </transitions>
</SML>
```

In this example, the generality of the format becomes apparent. There is no reference to “scenes”, “beats”, or any other story specific concept. If a version of the framework should implement such a specialization for the author, the authoring tool is the right place for the adaptation. The format and, as a consequence, the software that runs the format, remains unchanged. These results in a well-specified customization process that preserver reusability of the software modules.

Some explanations will be given on the preceding SML-example. All the explanations refer to additions that were made for the Cyranus framework, departing from the XTND-basis. A “frame” is the format’s name for a compound. The attribute “concurrence” of the frames has either the value “XOR” or “AND”, and specifies whether the children frames run exclusively or are orthogonal.

The condition in the example above,

```
<preconditions>
       <event>
```

```
            <message value="regionChosen">2</message>
      </event>
</preconditions>,
```

expresses that this condition is true when an event of the type “regionChosen” occurs, with value “2”. This Boolean “event”-function is the generic format for directly handling events in conditions:

```
<preconditions>
      <event>
            <message value="some_value">
                  some_XML_structure
            </message>
      </event>.
</preconditions>
```

Thus, the “event”-entity is interpreted as a Boolean XEXPR function, when it occurs in the condition of transitions.

The same structure is employed for the communication of external events. When employed as input format, the format is called EventML. Since “event” is both employed as Boolean function for conditions, and as communication format for input events, at runtime, a simple comparison suffices. When both are identical, the Boolean function is true.

The attributes “to” and “from” of the transition-entities allow for comma separated values, e.g. to=”state1, state2”. This is employed for entering and leaving orthogonal states with AND coupled transitions. (I.e., for a leaving multiple source transition to be followed, the “from” states must be active; a multiple sink transition activates all “to” states).

The Boolean “event”-function of a condition allows underspecification. Thus, the XML-structure within the “message”-entity is incomplete, i.e. contains fewer elements than the incoming XML-structure. In this case, the function is true when it is possible to delete elements of the structure of the input such that the structure becomes identical to the structure of the condition, possibly after reordering. For example, the function

```
<event>
	<message value="regionChosen"/>
</event>
```

employed within a condition of a transition, is true when the incoming event is a “regionChosen” event, no matter what value is has. This is an efficient measure for extending the range of input evaluation, and is especially important when the value of an input message is a complex XML-structure, as in the example of Virtual Human, cf. Section 6.4.

Some examples of basic XEXPR functions, as defined by W3C, are “not”, “and”, “or”, “random”, “get”, “set”. Any function can be easily added to the set of XEXPR functions. For example, the “event” function above is a secondary, introduced function. In the SML example above, “send” is the added function that outputs its content to the outside of the control system.

Events that enter the control system are invariably in EventML format, events that leave it are in any arbitrary XML-format.

5.4 Conclusion

The Cyranus framework is an implementation of the hybrid system described in Section 4, devised for experimenting with different solutions and ideas related to IN. The formats developed retain the generality of the approach, while remaining human-readable. The authoring tool allows for easy customization and co-development, together with the growing clarity about requirements that can only be attained during the creation process of an IN application. Expressive animation of virtual characters is achieved by employing an animation graph, morph targets, and posture adaptation. Several other components enable its usage for within a wide range of possible application design ideas.

Future work should encompass

- Mechanisms for the runtime visualization of the major processes, including the hybrid framework, preprocessing and post-processing, and data base

contents.
- Storage formats and interfaces for story states of interrupted stories, including the different engines and the data bases.

6 Applying Cyranus and the Hybrid Framework

In previous passages, the aspect of “wickedness” of IN as a research field was examined, as well as the fact that application concepts of IN are still in need of further clarification. In this section, different application concepts that exemplify and address some of the open research questions will be presented. These applications were implemented employing the Cyranus software environment, except for the “Virtual Human”-project, where the framework was employed as part of a different system (cf. Section 6.4). It is important to expose the concepts of projects and applications, partially in some detail, because: (i) The projects show that, in fact, Cyranus is an adequate framework for tackling the control issues of different IN ideas. (ii) Due to the “wicked” nature of the IN research endeavor, it is necessary to build plausible hypothesis on novel application concepts, which then also guide the technological research and the development of authoring tools and methods.

Note that none of the projects deals with pure IN. Rather, they exemplify that in practice, since IN is still a research field with many difficult open questions, most projects deal with the question of how to create and enhance applications that possess other goals with narrative elements. The project “Art-e-Fact” dwells on how to create narrative presentations of art; “Ask & Answer” has as theme narrative edutainment; “Virtual Human with Social Intelligence” contains a virtual assistant enhanced by mini-dramas. The demonstrators of “Virtual Human” do not contain narrative elements in a strict sense; but this project has important narration related results that are described in Section 6.4 and Section 7.

6.1 Art-e-Fact

Art-e-Fact’s application design relies on the following hypotheses:

- Works of art (and themes from the humanities in general) should be presented by virtual discussion groups, because art always is multifaceted and allows for different points of view (cf. [Iurgel03b], [Iurgel04b]).

- Interactive installations for art presentation should allow for different input modalities, in order to explore different aspects of the work of art (cf. [SpierlingIurgel03]).
- Stories as psychological transformations and revelation of the "inner self" of a person fit to the art presentation domain because understanding art presupposes learning how other people understand it, and thus the personalities and attitudes of these (virtual) persons become important ([Iurgel03b]).

Technologically, the present exposition of the project demonstrates the capacities of Cyranus. Cf. also [Iurgel03d], [Iurgel04b], [Iurgel05c], [Iurgel06a]. In particular it could be shown that:

- The Cyranus framework can be employed for controlling interactions with different input devices, including gesture recognition and text input.
- The Cyranus framework allows to deal with the problem of possible explosion of permutations of input, which is not possible with a pure directed graph.
- The framework can be employed for proactive, domain oriented chatting with the virtual character, by combining directed graphs with a Fallback Engine.

6.1.1 Application Design

Art-e-Fact was an EU-project under the technical and scientific guidance and partly also under the general management of the author of this dissertation. It involved five European countries and nine partners, and was successfully finished in 2004. The author was responsible, among others, for the architecture definition, the protocols, interfaces, choice of technologies, and the runtime storytelling environment in general (cf. [Iurgel05c], [Iurgel04b], [IurgelHoffmann], [Iurgel04b], and [Iurgel03d]). The project originated both in ideas developed in [Iurgel02a] and [Iurgel02b], and in distinct previous work developed under the guidance of Ulrike Spierling, on presentational talking heads and on using props to interact with informative virtual characters (cf. [SpierlingGrasbon]).

Everyday learning practice inspired much of the outline of Art-e-Fact. The normal way of things is that people grow up, acquire knowledge and form opinions and tastes essentially in groups. Solitary learning is an often painfully acquired skill, and only part of a larger essentially social learning practice. Often, learning takes place by hearing and observing controversial points of views. This dependence of learning on concrete persons and dialogue is especially important in the humanities and arts, where the concept “objective truth” may be inappropriate or very difficult to apply.

Due to this typical social context of learning, interactive discussion groups of virtual characters are ideally suited for the presentation of themes related to the humanities (cf. [Iurgel03b], [Iurgel04b]). The evolving of thoughts in discussions on humanities is not something that can be abstracted from, when it comes to presentation e.g. of philosophy. This is clearly a Socratic view of teaching that interestingly can be revived with the aid of forefront technology.

Discussion groups with virtual characters have been presented before (cf. [AndreKleasen]), but with focus on dialogue generation and not on authoring and interaction through chatting and varying props.

Besides of the alleged thematic appropriateness of the dialogic method, the second fundamental aspect is the dependence of learning from the personality of the person we learn from, in many realms (cf. [Iurgel03b]). When things are not “objective”, it is important to know the person, his life circumstances, his behavior and general understanding, to include it into our assessment of particular views, e.g. on philosophy or art. This is why employing virtual characters with clear personal stories and pronounced personalities is presumably a vigorous benefit for the didactic presentation of themes from the humanities and art, and certainly from other disciplines, too.

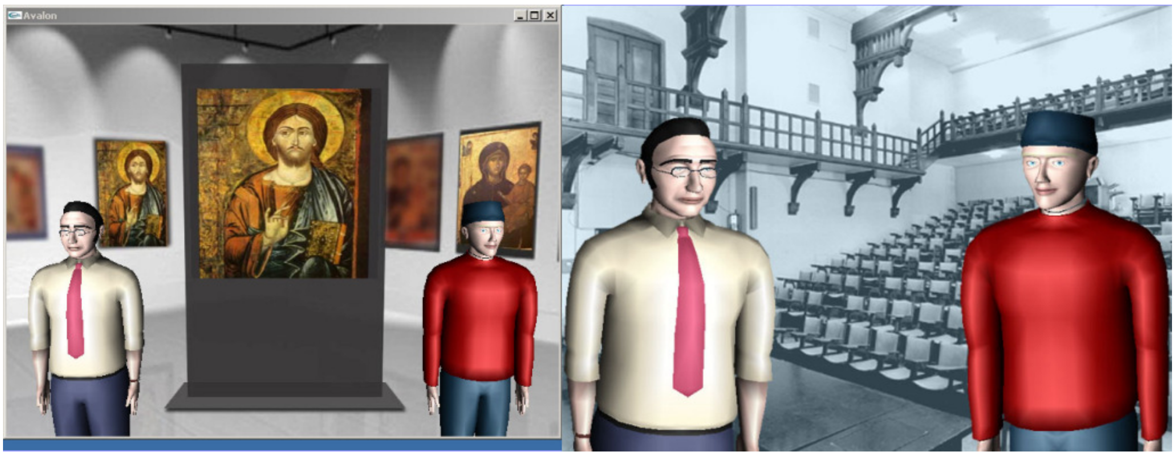


Figure 44 – Screenshot of the same two virtual characters being employed for two different Art-e-Fact installations.

The project Art-e-Fact was intended to make possible the exploration of these allegations. In the context of this project, the author of this dissertation has developed a highly generic framework for artists and content creators to devise and deploy museum installations with interactive, discussing virtual character that involve story, personality, emotions, keyboard based chatting, and props (e.g. a “magnifying glass”, which was recognized by the video recognition system), and embedded into an easy-to-use, and nevertheless generic environment. Cf. Figure 44 for a screenshot of Art-e-Fact.

Thus, with Art-e-Fact, the discussion group paradigm was transferred to computer applications. The project has employed an educational method of access to different subject matters that had already proven to be appropriate for many areas of knowledge. The main focus group consists of museum visitors who are interested in learning more about the exhibits (Figure 45). The visitors are confronted with culturally enriching, but also entertaining experiences through interactive installations related to the artworks (Figure 46). Virtual characters present aspects of the exhibits in a lively discussion from their different personal, theoretical, emotional, and cultural points of view. Their various education levels can be reflected as well as their emotions. An appropriate approximating metaphor is that of a coffee table discussion.

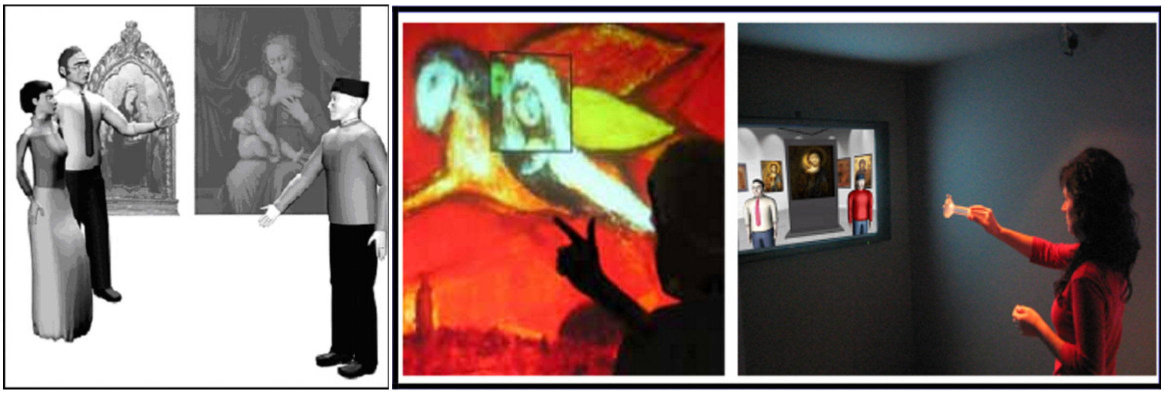


Figure 45 - To the left, a group of virtual characters discuss religious icons. To the right, a demonstration is depicted of the usage of the gesture recognition. The visitors there are exploring details of paintings, using dedicated props.

A story is employed to provide motives for controversies, emotions, and to expose personality traits of the virtual characters. Visitors interact with special devices to explore virtual exhibits. For example, they can uncover different layers of a painting with gesture recognition or chat with the virtual characters about the exhibits.

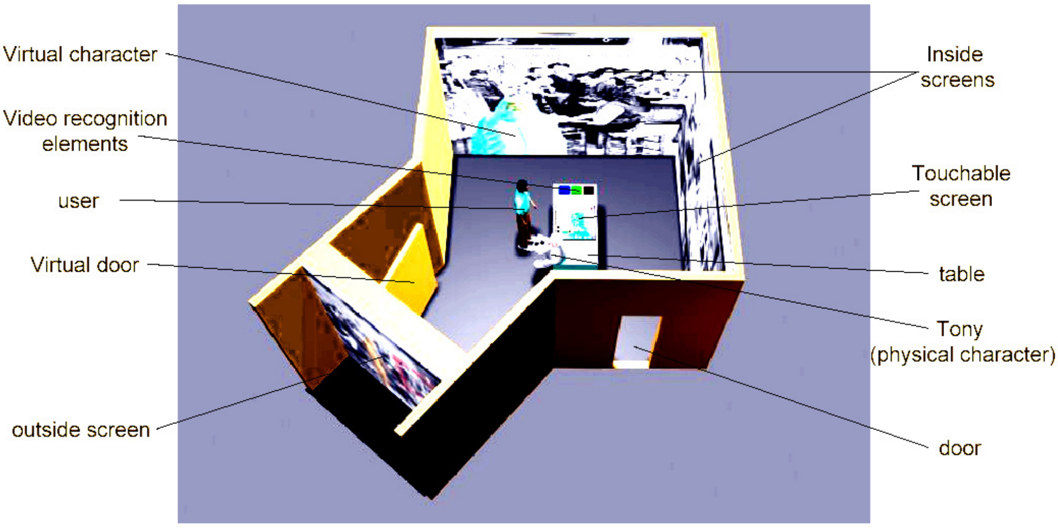


Figure 46 – A possible use of the Art-e-Fact platform with projection walls[23].

Two main exemplary public installations were created within the context of the project. One was shown in the Vienna National Gallery and was about the paintings and life of Francesco Guardi, the other took place at the Bargello Museum in Florence, and was about Renaissance sculptures of the Biblical David.

The standard interaction mode for the public installations was gesture recognition and specialized props that equally were employed with the aid of video recognition. Chatting was employed only for experimental, non-public demonstrations, due to the workload that the creation of data bases for chatting would have caused. Gesture recognition allows for navigation and choices as well as for active exploration of works of art, e.g. the visitor uses his finger as a "magic" lens that amplifies, or of a spotlight that highlights details; with the recognition, it is also possible to move and rotate 3D-objects, e.g. sculptures[24].

A major goal of the project has been to develop methods and principles that allow content creators with some talent for computer science methods, but without any special training on them, to create installations that integrate the different input

[23] Installation idea and image by M. Burgos and J. M. 'hafo', arteleku, Spain.

[24] The tasks related to props, e.g. implementing the video recognition and the choice and construction of the props, is an achievement of other project partners

modalities and encompass storytelling, art presentation, personality rich virtual characters, and the other aforementioned requirements.

An exemplary story for the Vienna exhibition involves two virtual characters, the demure art curator Dr. Hertzberg and the thieve James (cf. [IurgelHoffmann04]). James needs a value estimation of a stolen painting, and for this he needs the help of Dr. Hertzberg and of the museum visitor. The visitor should use gesture recognition devices to help the characters to uncover the hidden layer. The point of the story is to provide a narrative motivation for the visitor to examine the hidden layer of the painting, which in fact is an older painting that Francesco Guardi himself has painted over; this could be detected by X-ray analysis. The story also involves the visitor into a conflict situation: shall he betray the thief, at the end? In turns out, though, that James had a moral motivation for stealing, so the decision is not easy.

6.1.2 Employing Cyranus

In the context of the project Art-e-Fact, the main challenge was to maintain the generic faculties of the environment, since Art-e-Fact was intended as a generic platform for educational museum installations. In fact, this requirement of Art-e-Fact was the starting point for the generic authoring environment described here.

In Art-e-Fact, arbitrary input devices are allowed, and also text input as chatting. The following paragraphs described how the requirements of Art-e-Fact were met.

6.1.2.1 Handling Various Input Devices

According to the Art-e-Fact architecture, a device handler is associated to each device (e.g. gesture recognition, handler for text input, prop recognition). A device handler will translate the original input of the device (e.g. regions recognized by a video recognition system) into a format suitable for the platform, and possibly possess some kind of specific logic. The standard way of communication of the device is by the standard generic message format. As an example, the choice of regions of a painting with the use of gesture recognition will be described. This “region-choice”-device is a typical use of Art-e-Fact in museums, since it enables the interactive exploration of paintings. With this device, the user points to regions of a painting, the “hot spots” of the image, using his finger, and the system reacts to this deictic gesture. The region-choice-device can, of course, be used in a manifold way, including for navigation.

A typical output of the region-choice-device is

```
<event>
       <message value=”regionChosen”>
              pointingRecognition.MariaRegion
       </message>
</event>,
```

where “pointingRecognition.MariaRegion” is the region chosen by the user. The device manager is configured to display, in the virtual world, a certain image with specific hot spots by an action command of playerML of the kind of

```
<device name="pointingRecognition">
       <enable id="e7" >
              <configuration name="Guardi"/>
       </enable>
       <appear id="e11"/>
</device>,
```

where the mapping from the name “Guardi” to a certain image, regions and region names is defined in a “StoryEntities”-file (cf. [Iurgel03d]). The tag “appear” is a command to the device manager to show the image.

A standard condition of the Statechart would be:

```
<preconditions>
       <event>
              <message value=”regionChosen”>
                     pointingRecognition.MariaRegion
              </message>
       </event>
</preconditions>,
```

i.e. as a repetition of the event generated by the device manager. This condition is true if the device manager generates the respective event. Note that the underspecified condition

```
<event>
       <message value="regionChosen"/>
</event>,
```

is true when some “regionChosen” event is generated by the device manager, irrespective of the concrete region that was chosen.

For many contexts of museum installations, this part of the framework is already sufficient, for example when the devices are mainly employed for navigation. An example that demonstrates the use of hierarchies is provided by the presentation made for the Vienna National Gallery, where the installation was activated by mats on the floor, and the visitors employed gesture recognition for navigation.

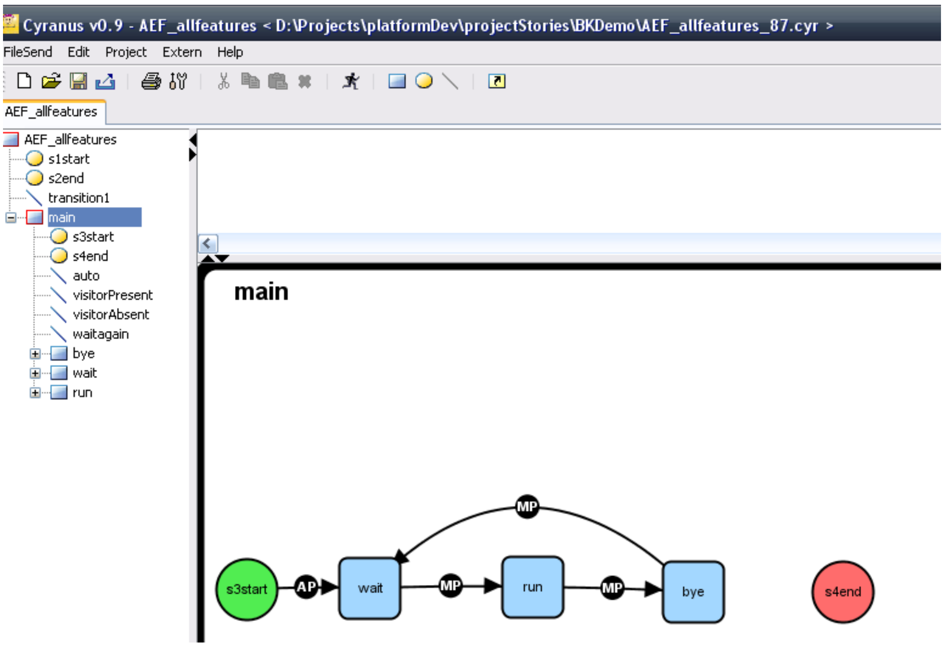


Figure 47 – A screenshot of a directed graph structure of the authoring tool of Cyranus, for the Guardi-exhibition.

Figure 47 is a screenshot of the authoring tool of Cyranus. The three composite states (the squares) and the transitions control the behavior of the system with respect to the mat: When the user steps on the mat, the mat device handler generates an event, and the transition from the state “wait” to the state “run” is followed. When the visitor leaves the mat, the state “run” is left (break condition), the virtual characters say goodbye (compound “bye”), and the state “wait” is reentered. All of the other behavior of the installation is encapsulated within these three states.

However, the directed graph approach already reaches its limits with devices that only generate a limited set of events, for example when the sequence in which the events are generated is important, and many permutations are possible. Handling this can result in forbiddingly complex graphs. For example, in the “Guardi”-installation, an important scene contains an exploration of a painting by Guardi. The visitor uses the pointing recognition to activate hot spots. A smart behavior is expected from the virtual characters: they shall comment on the hot spots without being repetitive, i.e. they shall not say the same when the visitor points twice to a spot. And they shall guide users to hot spots not yet found.

A transition network for this behavior becomes too complicated when the conditions contain the logic, because the logic then is repetitiously distributed among the transitions. For a simplified example, let’s assume the Boolean variables

> *alreadyVisitied_stateName*
> *alreadyActivated_regionName*

in the data storage, which contain the information on whether a certain state was already visited, and whether a certain region was already activated. A corresponding example condition for a “hint” state (a state that contains information that guides the user to a hot-spot not yet found) in XEXPR would read

```
<and>
	<timeout value="30000"/>
	<equals>
		true
		<get name="alreadyVisited_MariaHintState2"/>
	</equals>
	<not>
		<get name="alreadyActivated_regionName"/>
	</not>
</and>
```

Additionally, the author would have to take care always to set the variables and priorities correctly, for a large number of transitions. This example is simplified because it does not include smart location dependent guiding behavior, i.e. the user is not guided according to the area of the painting he is examining. The characters won't be able to say coherently "a bit more to the left (of where you are pointing) is something interesting".

The creation procedure can be much simplified when the logic is centralized. It is possible, within the current framework, to centralize the logic in the device handlers, similar to Gebhardt et al. (cf.). For example, the device handler can keep track of the regions already visited and of the region the user is currently pointing at. It then generates events of the form

```
<event>
	<message value="regionChosen">
		<region>
			Maria
		</region>
		<visitCount>
			2
		</visitCount>
	</message>
</event>,
```

which can be evaluated in the conditions of transitions of the Statechart. Proactivity could also be handled by the device manager, which has then to decide on when to activate a guiding state ("point to the left"). Otherwise, the problem of too complex interdependent conditions and distributed logic reappears. But this approach puts all of the control outside the system, and the author cannot decide any more on when to activate which state. Furthermore, the resulting transition pattern is trivial; it is a simple star, with a central waiting state in the middle, and transitions are in fact chosen by the device manager (Cf. Figure 48).

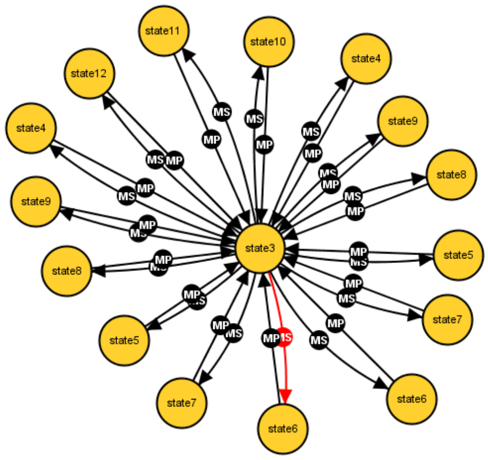


Figure 48 – Some directed graphs patterns are difficult to create and maintain, but the structure is not informative.

In this situation, employing a Sequencing Engine is a better solution, because it allows for more control by the author, and allows to get rid of unnecessary, uninformative transitions. The corresponding Sequencing Engine will choose a state according to its metadata. They are, in this example, of the form

```
<metadata >
       <sequencingData engine=”jessHotspot”>
              <hint priority=”3” hotspot=”Maria”>
       <sequencingData>
</metadata>
```

for a state that contains a hint that guides the visitor to the hotspot “Maria”, when the visitor has not yet pointed to this, and the visitor is currently not pointing to the screen, and no action is taking place. A higher priority number means that the state will be activated first, when more than one hint-state for the same hot spot is available.

For states with actions that contain spatial orientation (“point to the left”), the format is

```
<metadata engine=”Hot Spot Engine”>
       <sequencingData engine=”Hot Spot Engine”>
              <spatialOrientation priority=”2” hotspot=”Maria”
```

```
            currentPointing="immediateRight">
      <sequencingData
</metadata>
```

for specific hotspots, e.g. for the “Maria”-hotspot. The action is, in this case, an utterance of the professor

> “point a bit more to the left, and you will see an image of Maria”.

For generic orientation actions, the value of the attribute *hotspot* is changed to “generic”, e.g.

```
<metadata engine="Hot Spot Engine">
      <sequencingData engine="Hot Spot Engine">
            <spatialOrientation priority="4" hotspot="generic"
            currentPointing="immediateRight">
      <sequencingData
</metadata>
```

In this case, the action cannot be specific for this hotspot. It is

> “point a bit to the left, there is something there”.

The pointing device manager generates events of the form

```
<event>
      <message value="pointingSearch">
                  immediateRight
      </message>
</event>,
```

which are consummated by the Sequencing Engine responsible for the hot spots, the “Hot Spot Engine”. Employing the metadata of the states, the Hot Spot Engine will activate appropriate states that guide the user.

The transitions are no longer necessary, in this example. A Cyranus screenshot is shown in Figure 49. Note that the arrows on top of the states indicate that they contain metadata.

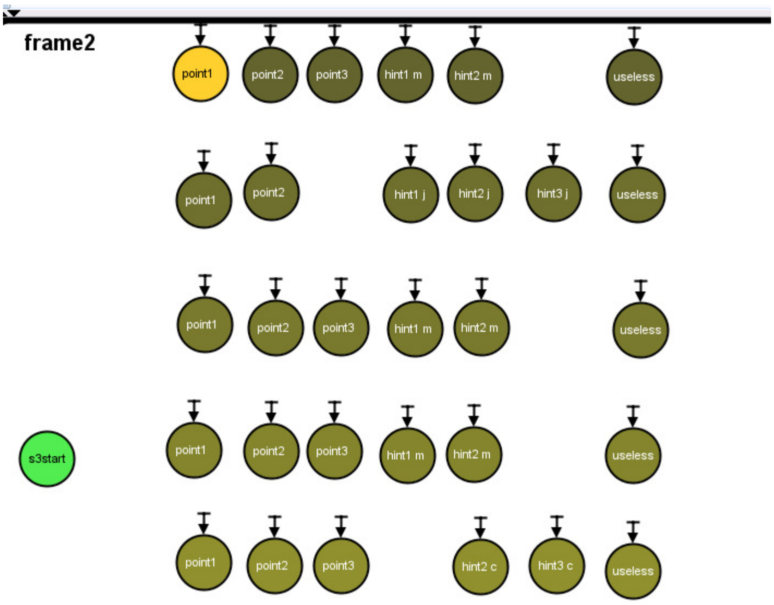


Figure 49 – These basic states can only be activated by some Sequencing Engine.

The hot spot Sequencing Engine employs a set of forward chaining rules. They consist of rules that activate the guiding and commenting states. The rules translate to plain English approximately for example to:

```
IF the visitor points at a hot spot and there are corresponding states that were
not yet activated
THEN activate the state that corresponds to this hot spot with least priority and
that was not yet activated

IF the visitor point at a hot spot, and there are no corresponding states there
were not yet activated
THEN activate a generic comment state (e.g. "you have already looked at this!");
prefer generic comments that were not yet employed

IF the visitor is pointing to a location near and to the right of a hot spot that
was not yet visited (the device handler generates a "immediateRight"-event)
THEN activate a state that informs the user of this fact; prefer states that are
specific to this hotspot, and that were not yet activated; if not available, em-
ploy generic guidance states

IF the visitor points at a hot spot and there are corresponding states that were
not yet activated
THEN activate the state that corresponds to this hot spot with least priority and
that was not yet activated
```

```
IF the visitor point at a hot spot, and there are no corresponding states there
were not yet activated
THEN activate a generic comment state (e.g. "you have already looked at this!");
prefer generic comments that were not yet employed

IF the visitor is pointing to a location near and to the right of a hot spot that
was not yet visited (the device handler generates a "immediateRight"-event)
THEN activate a state that informs the user of this fact; prefer states that are
specific to this hotspot, and that were not yet activated; if not available, em-
ploy generic guidance states
```

Figure 50 – An excerpt from the actual Jess rules.

Note that an automatic drawing of the possible trajectories would not make sense, in this example, because too many trajectories are possible, and the visible result would be a chaos. Cf. the screenshot of Figure 51, where the partial connectivity of a few states was manually drawn, according to the pseudo-rules above. These states pertain to only two distinct hot spots.

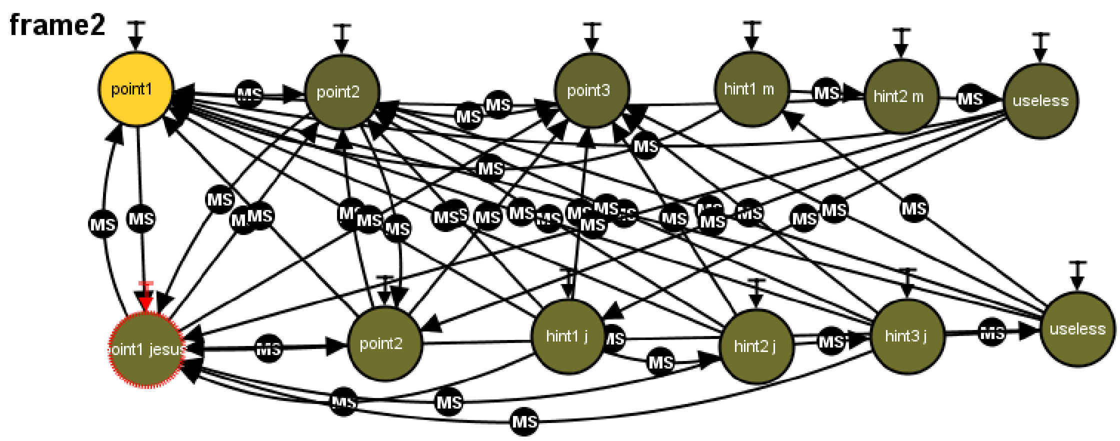


Figure 51 – The "point" states contain utterances that shall be activated when the user point directly at a hot spot; the "hint" states contain utterances that guide the visitor to this hotspot, when the user has not yet found it. The "useless" states are activated when the system gives up guiding the visitor to a certain hotspot.

Note also the importance of restricting the scope of validity of a Sequencing Engine to the children and decedents of a single compound, as described in Section 4.3: Other kinds of Sequencing Engines could have been employed for other situations that occur within the story, and a Sequencing Engine that follows some model on plot for the choice of the scenes, i.e. for the choice of compounds that themselves contain very different rules. The assignment of a Sequencing Engine to a compound ensures that the semantics of the system remains consistent, i.e., enables such combination of different sequencing algorithms. Note also the possible use of the hierarchic evaluation of input, where Sequencing Engines of internal compounds are evaluated first; with this measure, a Sequencing Engine for chatting, for example, can be easily integrated and attached to an ancestor compound, to allow the visitor to chat in the described painting exploration scene.

The workflow for the content creator consists, when the Sequencing Engine is employed, in creating appropriate utterances and annotating them with metadata. He only employs the rules, but is not expected to write any new rules, or to understand exactly how they work. If rule changes are necessary, a computer scientist would be called to assist the domain expert. This ensures a distribution of roles in the authoring process, and enables team work – an essential feature of the authoring principles.

The isolated use of a single hot spot Activation Engine solves the main problems of handling large numbers of possible permutations of input and of division of work with a generic visually supported framework. A simple example of employing the delegation pattern would involve drawing a transition from one of the states which contain information for a hot spot, and using it for a question like "do you want to learn more about this hot spot?" A more elaborate example on the use of the delegation pattern for exceptions follows.

Some hot spots are semantically related, and the author wants the virtual characters to guide the user accordingly. E.g., when the user points to the "Maria"-hotspot, the professor will first deliver the information on the "Maria"-hotspot, and then the boy will remark:

> "look a bit to the right, a bit below; you will see that Maria is holding a child".

If the user points immediately at the "child"-hotspot, the professor says:

> "You've found the child! It is Jesus, of course. He is looking at his mother, do you see?"

The challenge was to link those semantically related remarks, since the rules would dictate that a direct comment on the “child”-hotspot be activated, without the remark that establishes a link to the previous action (“You’ve found the child”). Additionally, if the visitor first points to another hotspot than to the child, the professor shall comment, instead on the hotspot, that this is not the “child”-hotspot, e.g.

“This is not the child you are searching for! Look a bit to the right and below of Maria!”

With a combination of the production system (Sequencing Engine for hot spots) with a Statechart, this behavior is easy to model, cf. Figure 52. Note that the common variable “alreadyVisited_childHotSpot” is required here to inform the Sequencing Engine that appropriate information was delivered when the user pointed at the “child”-hotspot. The Statechart has to check whether the information was already delivered before (condition A) and has to set the variable (state D). The behavior of the system when state C is active, and the visitor points at something else than at the child, according to the example graph, will be the normal Activation Engine behavior: for example, when the visitor points at the “clock”-hotspot, the boy might say: “this clock is visible on both paintings”. That is, the system has given up guiding the user to the child. This behavior also exemplifies the working of the delegation chain: the directed graph based engine, being positioned on top of the delegation chain, functions here as a handler for exceptions to the rules of the Sequencing Engine.

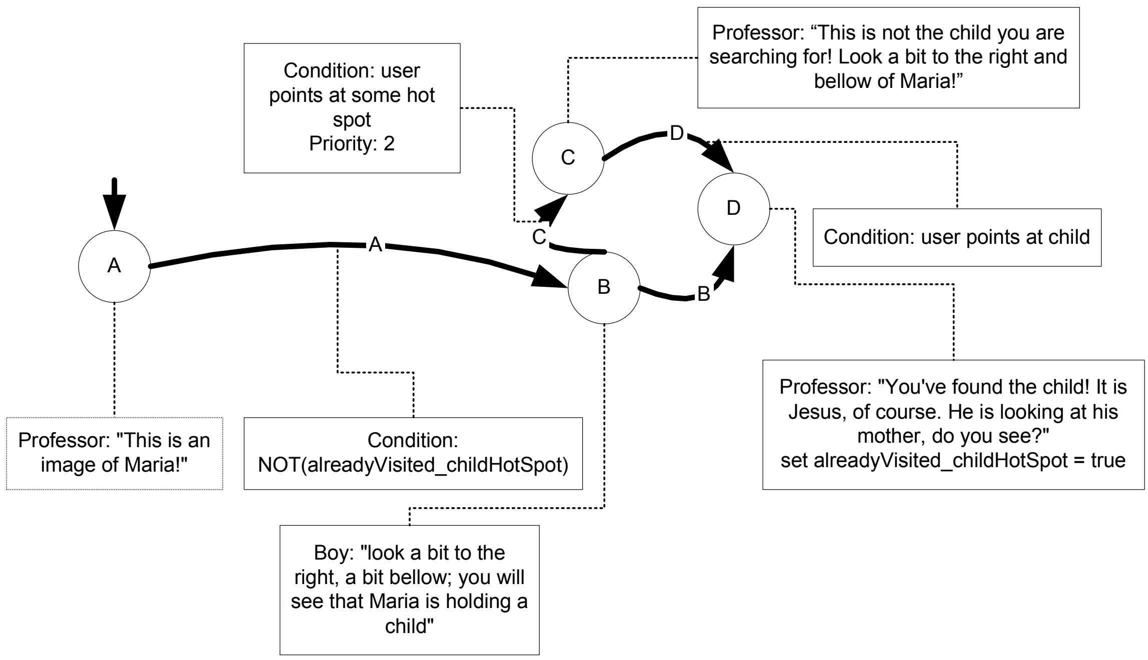


Figure 52 – Instead of changing rules, in many cases, the content creator can draw transitions that function as exceptions to the rules.

Certainly, there are many possible solutions to the problem presented here on how to model a control that guides the user from one hotspot to another, employing the present framework. It would for example also be possible to extend the rules to incorporate the exceptions, but this requires work by the computer scientist, and is visually less informative than possible. The point of this example is exactly to show that it is possible to employ the Statechart to easily and rapidly create new behaviors, even when a rule system is being employed. It is now possible to experiment with specific behaviors. It may later turn out that employing rules or some other different, non-graph based activation system is required, but the decision on this can be postponed, and the experiments can start immediately. For the museum installation that is being exposed here, only the presented example of direct guidance from one state to the other was employed. Thus, no changes of the rule set were required.

6.1.2.2 Handling Text Input

The integration of chatting with an unfolding story is achieved with the help of two measures: (i) Selective call of a chatbot with thematic constraints, as a Fallback Engine (cf. Figure 54), and (ii) use of the Statechart in order to create proactive behavior and a surface of excellent behavior patterns (Figure 55, cf. also [Iurgel05b]). No Sequencing Engine was employed for chatting within Art-e-Fact, since the chatting is used here centrally for information gathering, and only to a lesser extent as a story driving force. Thus, a database with response patterns for informative questions is on focus, and a surface behavior to guide and entertain the user.

Since Art-e-Fact installations involve normally several art exploration devices, the resulting story line is an intermitted sequence of different interaction modes, and chatting is only allowed within specific scenes, cf. Figure 53.

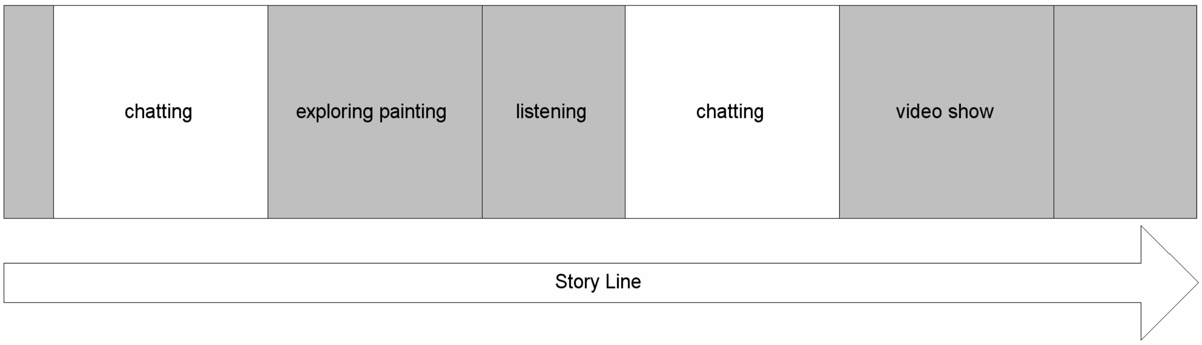


Figure 53 – Chat is allowed periodically.

A simple example on how chatting is integrated is a dialogue between the professor and the boy about the Guardi-painting. The professor announces that it is willing to answer factual questions on the painting, the corresponding chatbot databases are activated from within the Statechart, and text input of the visitor is basically passed over to the chatbot. The chatbot is embedded within a Fallback Engine (cf. Section7), the “ALICEEngine”. Since this engine is able to generate events, it can inform the Statechart on the ongoing chatting session. Within the implemented prototype, the chat engine sends information on whether the visitor is chatting at all, and whether he is talking about the picture, about another theme of the data base, or whether the theme is not being recognized at all.

The events generated by the chat engine are of the form

```
<event>
	<message value="chatinformation">
		noConversation
	</message>
</event>
```

for the case of the visitor not chatting at all. These events are generated after a timeout. With every response of the chatbot, an event is generated that informs the Statechart on the data base employed (cf. Figure 54).

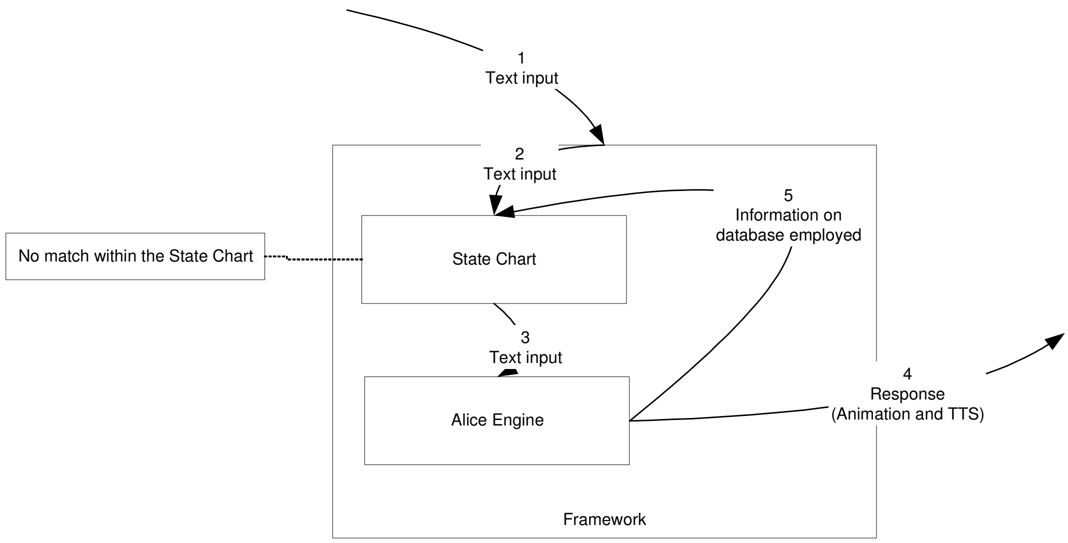


Figure 54 – The chatting of Art-e-Fact is controlled by the directed graph and the Fallback Engine (ALICE Engine). The ALICE Engine generates an event that informs the system on the data bases that were employed for responding, if any. Thus, the system can adapt to change of themes.

With these patterns, it is already possible to create the required chatting behavior. For example, a typical conversation would be:

Professor: "Now you can ask us questions on the painting!"
(No input from the visitor for 15 seconds)
Professor: "Please use the keyboard to ask questions on the painting!"
Visitor : "When was this painting made?"
Professor: "James, do you know that?"
James: "Of course, it was made in 1874!"
Professor: "More questions?"
Visitor: "How are you doing?"

Professor: “I am very well, thank you. Please stick to the theme!”

The Statechart handling of this example is depicted in Figure 55. There, the “Wrong Theme State” handles messages from the chat engine that the visitor is not chatting about the painting; The “Incentive State” produces utterances that urge the visitor to continue the chatting, after each theme related question or remark. Note that within both states, *reentering* transitions are being employed (transitions marked with an *R,* cf. Section 4). This means that, when the memory state (marked with an H) is addressed, not the last active state of this compound will be set to active, but the state to which a secondary transition starting at the last state points to. In contrast, within the “No Chat State”, no secondary transitions are used, since nothing shall happen immediately at a return to this state. The last state will be set to active in order to wait for some new event. Note that no Sequencing Engine is being employed here, but only a chat engine as Fallback Engine.

The essence of this control method, that combines a transition network and a chatbot within a framework that is also able to handle most diverse other input devices, lies in the tight control of the chatbot and guidance of the user that is enabled by generic visual means. I.e., the author can easily employ available chatbot technology, and use the visual means of the Statechart to control its behavior, to integrate it within the broader context of the installation, and to add specific faculties to the chatbot.

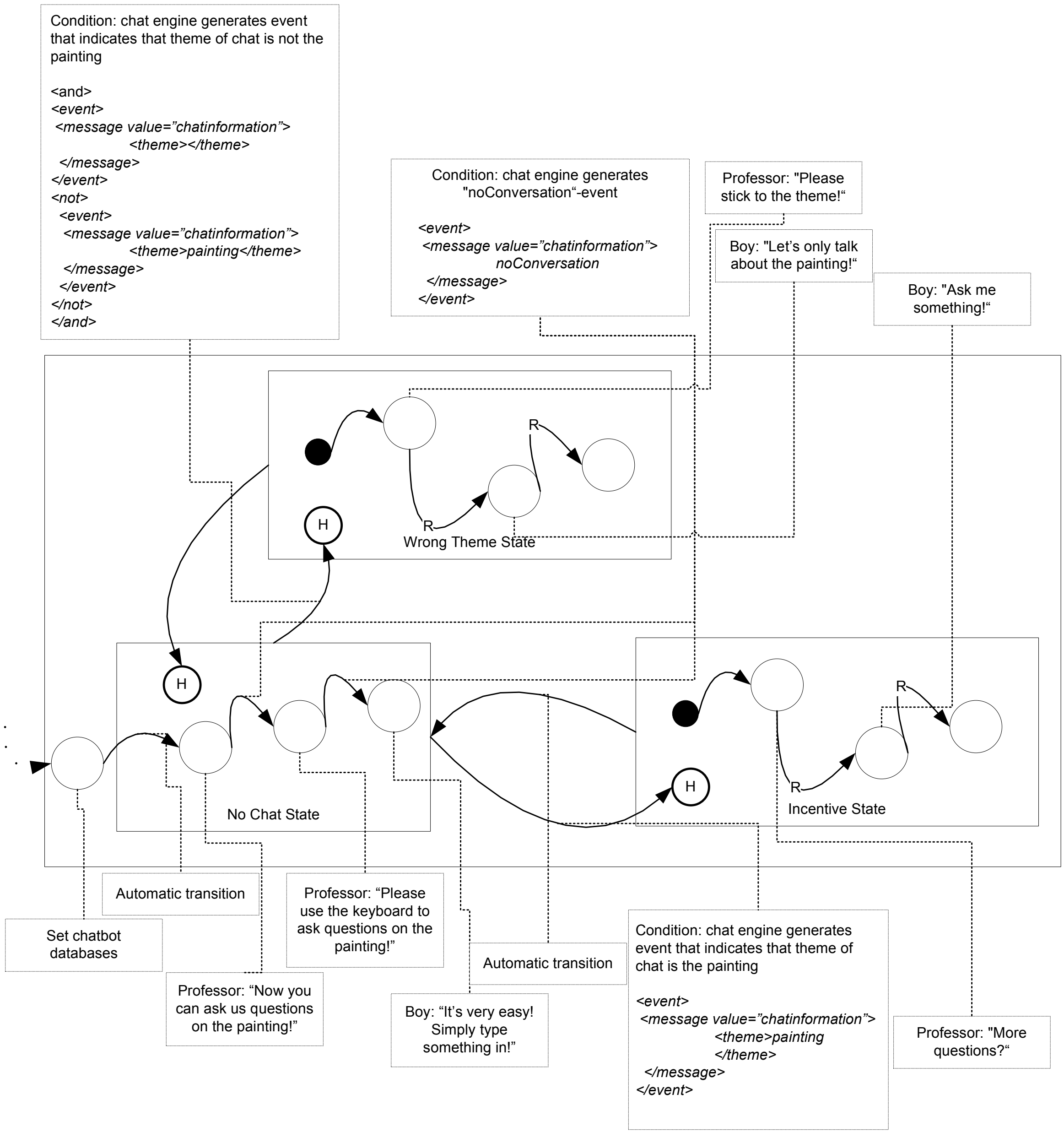


Figure 55 – Excerpt of the Statechart part of a combined approach for chatting about art.

Similar patterns were employed experimentally, within Art-e-Fact, to create more socially oriented chatting behavior, similar to [MateasStern05b]. In these social examples, the user chat influences the sequence of scenes of the story. For example, the visitor is prompted to take sides[25], and to decide on whether the professor or the boy is right. For example, in the context of the "Discovered letter story", the following dialogue occurs:

[25] A similar "taking-sides-game" is also part of Façade, cf. [MateasStern05b].

Professor: "Mh... I do not believe in that letter, that's certainly a bad joke".
Boy: "But it really looks quite old, it must be authentic!
Professor: "To whom do you think are you talking to? I am expert, I know what I say!"
Boy: "But that's arrogant! (To visitor) Don't you think that's the letter is authentic?"
(Here, the visitor is actively urged to contribute. Since the characters lead the conversation, a couple of response patterns is easily foreseen and can be integrated into the scene compound. If the visitor does not answer, the story will continue following a standard trajectory.)
VISITOR: "Yes, I think so"

The affirmative answer of the user was employed to increase emotional proximity parameters of visitor and boy. The recognized pattern is simply "yes", and the social distance is stored as variable of the common data storage. This variable later on influences the choice of appropriate scenes, appearing in the condition of transitions. More on social proximity variables will be explained in the section on “Ask & Answer”.

6.1.3 Evaluation

Art-e-Fact was evaluated with respect to several criteria of acceptance by visitors, and to its efficacy as a medium of learning. This evaluation is exemplary for the required evaluation methodology, and the necessity of an agile development process with early results, in order to assess the application design. The other projects presented in this dissertation were not evaluated formally – note that the aim of the present dissertation is not to assess effects and acceptance of particular applications, but to develop a single framework with the help of which these different applications can be first of all created. Therefore, the evaluation of Art-e-Fact will be only presented briefly, and only the most important results will be addressed.

The methodology for evaluation is work by Marc Hassenzahl ([Hassenzahl], described also in [Iurgel05c]). Its subject has been a simplified version of the Guardi-story, where the visitors only employed gesture recognition to interact. 30 volunteers were divided into two groups, where one group was told beforehand that, after the interaction session with the installation, thematic questions would be posed, and the other group was asked the questions without prior preparation. All volunteers were then presented with a questionnaire on their impressions. The division in two groups was intended to research for the efficiency of the paratelic learning of the installation, i.e. of the playful learning that occurs without any explicit intention to learn.

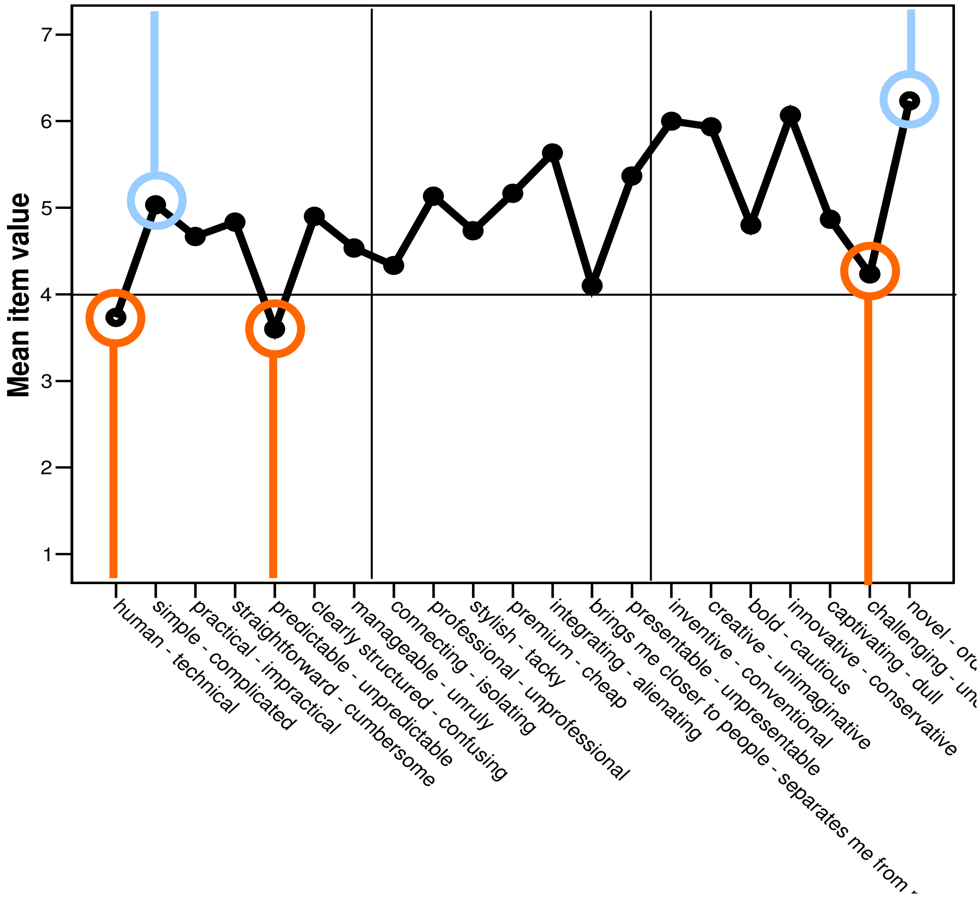


Figure 56 – Quality perception – Diagram by Marc Hassenzahl (cf. [Iurgel05c]).

Not surprisingly, the paratelic learning results were notably inferior, though some learning effects were present. This indicates that visitors learn something through a playful, hedonic interaction with the applications, albeit not as much as they would if they know that they were expected to learn and that the learning results would be evaluated. These results were not surprising because the installation that was evaluated was not fine-tuned to maximize learning results, e.g. historic dates that were inquired afterwards were only presented once and shortly, during a session. But it remains an important research question whether paratelic learning through such applications can in principle achieve similar or even superior results than telic, less hedonic learning. To what extent will applications like Art-e-Fact be able to replace traditional teaching? Up to now, we only have data that indicate that it can complement e.g. school lessons, but not sensibly replace any learning unit.

Then, the other important question refers to whether the visitors are motivated by the installation – in a learning context, this would be an important result, since motivation to learn is often difficult to achieve, and a motivating learning method will often be more appropriate than a method that is maximum efficient, on the short run, but tedious.

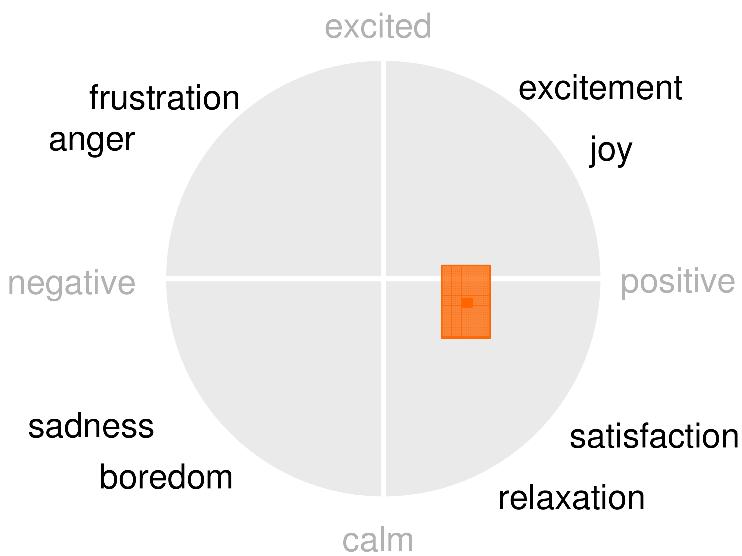


Figure 57 – The evaluation of satisfaction produced encouraging results, but also and astonishing tendency to "relaxation", in contrast to "excitement". Diagram by Marc Hassenzahl (cf. [Iurgel05c]).

The results in this respect were encouraging, since the overall emotional response to the installation was significantly positive. Interestingly, there is a certain tendency of these positive emotions towards "calm", "relaxation", "satisfaction", and not towards "excitement" and "joy" (Figure 57). This is surprising, if the very technological environment of the sessions is taken into account. There is no clear explanation for this finding, though the presence of rather non-aggressive virtual characters, their monotone voice, and the slow pacing of the concrete installation are explanatory candidates.

The only clearly negative aspect detected by the evaluation, criticized by more than 60% of all volunteers, was the speech synthesis (cf. Figure 56). This might be partly due to the fact that the installation language was English, and the volunteers were German native speakers. Though they were chosen partly for their good English skills, the synthetic voice might have posed additional difficulties to them.

In sum, the evaluation of Art-e-Fact demonstrates the need for a software framework that simplifies the task of building novel applications and of testing application ideas. The evaluation of this single application has pointed to strengths and weaknesses that must be addressed by future research and creative experimental work on interactive storytelling. The problems with speech synthesis were a surprise, since the version of the installation built for the evaluation did not contain dialogues that would require an emotional speech and prosody. The unemotional, but comprehendible voices of the TTS employed – ATT NaturalVoice – was expected to be well suited for the task. It is an open question whether a different design of the installation, e.g. with different dialogues and different, more "synthetic" looking virtual characters, would reduce the dissonance cause by the TTS. Equally important were the findings on unsatisfactory paratelic learning results, which demand for a deeper understanding of story concepts that could better foster learning. Further studies and experiments are also required to understand what exactly makes up an "exciting" installation, in contrast to a "calm" enjoyment. Since the concept of "story" is usually linked to "excitement", the bias found towards "calmness" might point to further undetected problems, when non-continuous, "exciting" narratives are intended.

6.1.4 Conclusion of Art-e-Fact

The Cyranus framework has proven to be generic enough to deal with the requirements of Art-e-Fact, and has enabled an iterative production process where content creators only have to employ the GUIs and the directed graph, and computer specialists write Sequencing Engines and devise Fallback Engine mechanisms that can then be employed by the content creators.

The Cyranus framework aims at a generic coverage. Thus, constraining the authoring tool can be useful if a very specific usage is intended. For example, for commercial creation of installations with virtual characters that guide visitors through the exploration of a painting, as described above, some more specific, list based input assistant could be useful, and visualizing the painting during the definition process of hot-spots would be helpful. This is a natural step of the joint development process of IN, where the application concepts, methods, algorithms and authoring tools are co-developed, improved and specialized together.

6.2 Ask & Answer

Ask & Answer is an experiment the application design of which addresses the following hypotheses:

- Since the user inevitably adapts to the equally inevitable shortcomings of a text based interaction, this adaptation should be motivated and rewarded, and integrated into the application design (cf. [Iurgel05b]).
- Narration, in the form of stories of virtual persons facing obstacles, can be integrated into edutainment applications with the help of virtual characters that introduce personal themes and seek personal assistance from the learners. This is intended to create fun and enhance motivation (cf. [Iurgel05b]).

From a technological point of view, Ask & Answer has demonstrated that the framework can be adapted to the task. Cf. also [IurgelZiegler], [Iurgel05a]. In particular:

- The Cyranus framework can be employed for the creation of complex educational games.
- It can deal with shallow domain oriented conversation on changing themes, and with switches to personal themes.

6.2.1 Application Design

The author of the dissertation has devised *Ask & Answer* (A&A) in order to examine the possible role of shallow natural language conversation – per keyboard – within IN applications, in particular within IN learning systems (cf. [Iurgel05a], [Iurgel05b], [IurgelZiegler]). Using natural language implies a non-trivial set of conceptual problems that are either not dependent on technology's shortcomings, or that are not likely to be eliminated by technological advances in any near future.

The non-technical, conceptual problem of employing natural language processing is related to the problem of lack of real world examples – a real actor would not know exactly how to perform his improvisational dialogue tasks, which renders the task of defining what a virtual actor extremely difficult and "wicked".

NL processing limitations and shortcomings of personality modeling prevent any communication with a virtual character from appearing natural. Thus the solution, for the time being, lies is incorporating those weaknesses into the design of the application. The following design principles for narrative applications promise a way out:

Rewards for adaptive behavior of user are built into the personality design of the virtual character

A capricious non-player character (CNPC) is defined as a virtual character that cares about how the user treats it and that expects certain linguistic and social behaviors from its human partner (cf. [IurgelZiegler]). This concept differs considerably from the "ELIZA"-tactics, which consists in hiding any problem of understanding form the user, creating a camouflage of semantically empty phrases. This approach is questionable because it is detrimental to the perception of agency. The user as player does not know whether he is really being understood and thus having an impact on the story world, or whether his input is lost and in vain. Some extra signalization of the level of understanding is likely to break the illusion of the game. The user is well advised to adapt to the expectations of the CNPC, out of utilitarian, egoistic considerations. The design of the "game", in a broad sense of the word, creates a dependency of the user from the CNPCs. Since the user depends on the CNPC, he will be motivated to adapt to the limited faculties of the technology, in order to achieve his goals. Thus, the user would try to "be nice to" and to "endear" the CNPC, e.g. he would try to make the CNPC like him, in order to be successful in the game. In this sense, the user would pay attention to his social bounds with the CNPC.

It is sufficient that the conversational behavior and the linguistic faculties of the CNPC are graspable enough to the user, so that the user can adapt its typed language to the CNPC's limited and peculiar NLP faculties. He would naturally and sensibly do this because, otherwise, a negative impact on his social bounds with this VC would result. E.g., the user would naturally avoid, after a short learning curve, difficult words and syntax, remote themes, anaphora, figures of speech, implicit speech, etc. The CNPC would normally not try to hide that it does not understand the user, but on the contrary, explicitly blame the user for this, and even get angry.

A very similar situation occurs with personality modeling. It is not necessary to create realistic psychological and cognitive models, as long as the reactions of the CNPC follow some (mostly) understandable patterns. Thus, it is of little weight if some emotional reaction of the CNPC should sometimes turn out to be unnatural for a human being. Its emotional states and patterns are important to the user because of his dependence on a good relationship to the CNPC, which forces him to stay tuned to its well being, independently of the psychological accurateness of its models.

Humor as complement to weird behavior of virtual characters

Since a CNPC will often fail and misunderstand utterances or react in an emotionally weird way, unintended comic situations will arise inevitably. Therefore, the story and the personality and appearance of the CNPCs should already create a humorous atmosphere, so that the funny fallacies will fit into this atmosphere. A dramatic story with unintended comic elements is not likely to function. The players are more likely to accept that a virtual character with a weird personality profile and funny, cartoon appearance often displays incoherent or inappropriate behavior, whereas a virtual character designed to appear realistic, both visually and behaviorally, is likely to be perceived as "buggy". For example, a virtual character designed to be an earnest person with a complicated personality structure cannot afford to enter into some fragile and complex dialogue, and then not be able even to answer reliably even simple questions.

Thus, the allegations are first (i) that a humorous game and character design is essential, in order to avoid entering the “zombie zone” of virtual character behavior; and second, that (ii) the player should be provided with a good reason to adapt to the limitations of the technology, staying in role also when it comes to these shortcomings (Cf. also [Iurgel05a]).

Ask & Answer is an educational game that exemplifies and explores these allegations. It is a game and story on its own and could be extended into a larger narrative educational game. It is also an appropriate test bed for the creation of CNPCs, because the game settings motivate the users to stay tuned to the peculiarities of the CNPC.

Ask & Answer targets pupils aged 10-15. The story that drives it goes as follows: Fritz, the only virtual character and CNPC, was the janitor of a now abandoned school, and, with the years, learnt everything that was taught at this school. Its wife quit it when the story and the game begins, and is in an instable emotional mood. Its wife reproaches it of only being interested in and speaking of school lessons and former teachers. This, Fritz itself tells the pupils, along with the conversation. At the beginning of the story, it would denies its wife allegation; but, in the course of the narration and of reflections, it recognizes that there is some truth to it, and decides to try to fix the relation to its wife and to change its spots. The story ends with this decision.

Fritz allegedly needs the players to reflect about the dramatic situation (i.e. about its marriage and its behavior), to console and advice it, and to speak about the beloved former school lessons. It is through this chat with the players and through reflection on it that Fritz will allegedly recognize that it is fixed on this part of the school’s past and not attentive enough to other persons and to its wife.

The love affair of the grumpy school janitor should appear ridiculous to pupils, and even more so their role as advisers and its need of consolation. This situation shall be funny and weird and not tragic at all.

The dramatic role of the players consists in accompanying Fritz through this inner transformation. There are many possibilities to fail, in this scenario, when chatting with Fritz, and to displease the CNPC: Not speaking about the former lessons when it wants to, not consoling it when it needs it, giving the wrong advices, being too rude or direct, using a language that it does not understand, and more.

Figure 58 – Two teams confront each other, Fritz, the Capricious Non-Player Character, is in-between.

Fritz' story only creates a background for an educational, competitive game. It adopts the formats of some TV-shows for kids and families, where two teams compete and scores are given for correct answers. Two teams of 2-8 participants play over the network. A team scores when it knows the right answer to a question asked by the system. Each participant looks at his own screen and has its own speech enabled avatar (Figure 58). The exemplary domains of learning are history and geography. The answer is usually typed in with the keyboard, though multiple-choice interfaces are also possible. Any participant may post an answer, but he risks negative scores for his team if the answer is wrong. Therefore, in order to coordinate the answers, the participants can communicate through their avatars. Avatars that are within a specific area, the territory of the group, can only be heard by members of the group that are located within this area. The players are allowed to use other material to find out the answers, e.g. textbooks.

Fritz knows in fact every answer to any question of the game. But it will only help if the player endears it enough and assists it with its story related problems. Fritz can only be heard by the team it is near to. If the team does not succeed in endearing the CNPC, it might refrain from delivering the answer, or even worse, it might leave the neglecting team and change to the other team. But even if Fritz is satisfied with some group, it might first want to talk about some favorite topic (typically additional material related to the question), before giving away any solution.

This concept of a CNPC much differs from the idea of a virtual character as believable actor or reliable assistant. Now, the virtual character must be understandable, rather than believable, so that the player may endear and manipulate it.

The game also aims at clarifying educational, technological and conceptual problems related to the creation of social bounds with virtual human-like agents, and to the use of natural language in this context. It explores the concept of profit in interactions with a virtual character. The application shall foster our understanding of intelligent virtual characters in situations where there are strategically rational reasons for the user to pay attention to the moods and wants of the intelligent agent. Fun and motivation are expected to last longer if they are grounded in utility, even if artificially as part of a game. Coping with the virtual character's peculiarities and establishing social bounds with it thus become part of the challenge of the game. Many edutainment and entertainment applications could be conceived from this starting point: The focus need not be in human-like behavior of virtual characters; rather, the focus can be on the human-like situation where it is profitable to adapt to the social expectancies and emotional peculiarities of the (virtual) other.

In general, this implies that a capricious non-player character must

1. possess resources that it controls and also retains;
2. that these resources are desirable to the user, and
3. that through skilful social and emotional interaction, the user can obtain these resources.

Thus, interaction with a virtual character is most entertaining and most useful not necessarily if its psychological model or its NLP is accurate, but if it pays off for the user to maintain a successful interaction. This much simplifies the Herculean tasks of NLP and psychological and cognitive modeling:

In short, the expectations are that

1. the creation of social bounds with Intelligent Virtual Agents as learning companions can foster the learning experience;
2. interaction through (typed) natural language is useful, provided that it is possible and pays for the user to adapt his language use to the limited faculties of the system;
3. emotional expressions and models of virtual characters need not necessarily be psychologically accurate, as long as the behavior patterns are understandable, and as long as it pays for the user to understand those patterns and to stay tuned to the emotions of the virtual companion.

Thus, this game involves social elements in a complex way:

- members of a team have to cooperate to acquire knowledge and coordinate answers, in order to defeat the other team;
- a team has to develop a common strategy on how to endear the CNPC, for example by nominating a CNPC-officer to exploit this source of knowledge, and
- the relationship to the CNPC is a social relation it pays off to maintain.

The CNPC evaluates, based on the technical framework described in Section 5 and [IurgelZiegler], whether it “likes” the team it is interacting with. The CNPC Fritz wants

- to talk constantly;
- to be praised for its vast knowledge;
- not to be insulted;
- to talk about certain themes, and not about others, and
- to understand the user.

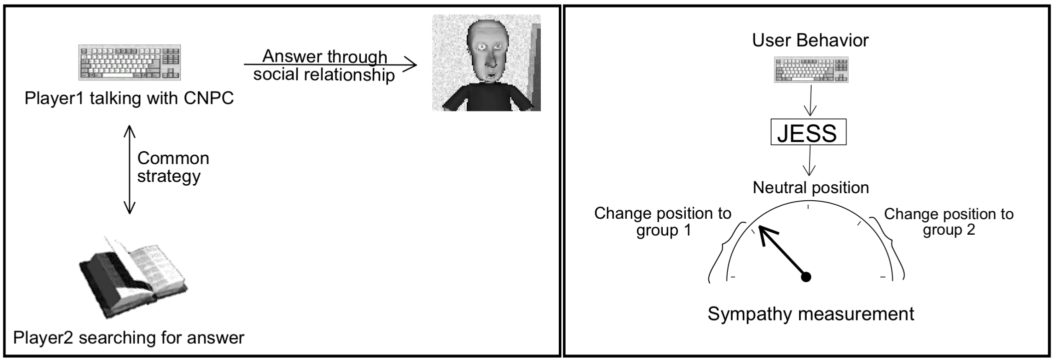


Figure 59 – To the left: Acquiring the right answer implies team collaboration. Player1 and Player2 have assumed separate roles to acquire the correct answers. Here, Player1 is responsible for endearing the CNPC. To the right: The CNPC has the tendency to move to the adversary team, if it does not feel enough sympathy.

The emotions of the CNPC towards a team – its sympathy for this team– depends on the fulfillment of these wants. If it is dissatisfied, it may leave the team he was talking to.

Now, using adapted social and linguistic skills when interacting with the virtual characters is clearly beneficial for the player (cf. also Figure 59):

- The CNPC possesses resources that the user wants, namely the right answers.
- The user can access those resources through social interaction.
- Attentiveness to the emotional state of the CNPC pays off, because its emotions are indicators of how successful the user is being in acquiring the resource – an unsatisfied character may even change the team, causing the loss of the resource.
- It is beneficial for the user to adapt its language to the limited faculties of the CNPC, because it tends to become unsatisfied when it does not understand.

If the themes the CNPC wants to speak about are chosen by the author to be those relevant for the curriculum, important additional learning effects can be achieved, since the conversation with the CNPC will be about things that have to be learnt.

Heuristics sum up the interaction and assign numeric values that are directly mapped to the valence of the emotional state of the CNPC (i.e. positive or negative; Ask & Answer won't consider directional aspects that would allow to discriminate between e.g. “angry” and “sad”).

Further rules express

- that the CNPC will tend to join the team which promises the better emotional outcome;
- that the CNPC will try out, more frequently at the beginning of the game than later on, which team is the most agreeable for it, and
- that the CNPC prefers the company of the loosing team (this shall diminish a bit its influence on the final outcome).

6.2.2 Applying Cyranus

The major challenge of Ask & Answer stems from the elaborated chatting requirements. An important feature is the integration of a model of social distance that determines Fritz' behavior. In short, it is be necessary to employ the Cyranus framework to compute

- the social distance between Fritz and each of the two learning groups
- the desire of Fritz to speak about his personal drama with his wife
- the interest of Fritz to chat about things related to the learning theme
- the ability to take initiative and to chat coherently
- the decisions of Fritz to change theme, to give hints, and to change the group it is chatting with, depending on social distance and score of a group

The state structure of Cyranus employed is composed of several compounds, cf. also Figure 60:

- Curriculum compounds for handling conversation on the topic of the current question that the system has posed to the teams
- Personal compounds for handling conversation on the marriage and the feelings of Fritz

- Orientation compounds with dialogue acts suitable for providing the users with an orientation on thematic changes, misunderstandings, and explanations of actions
- A compound for moving to the opposite group

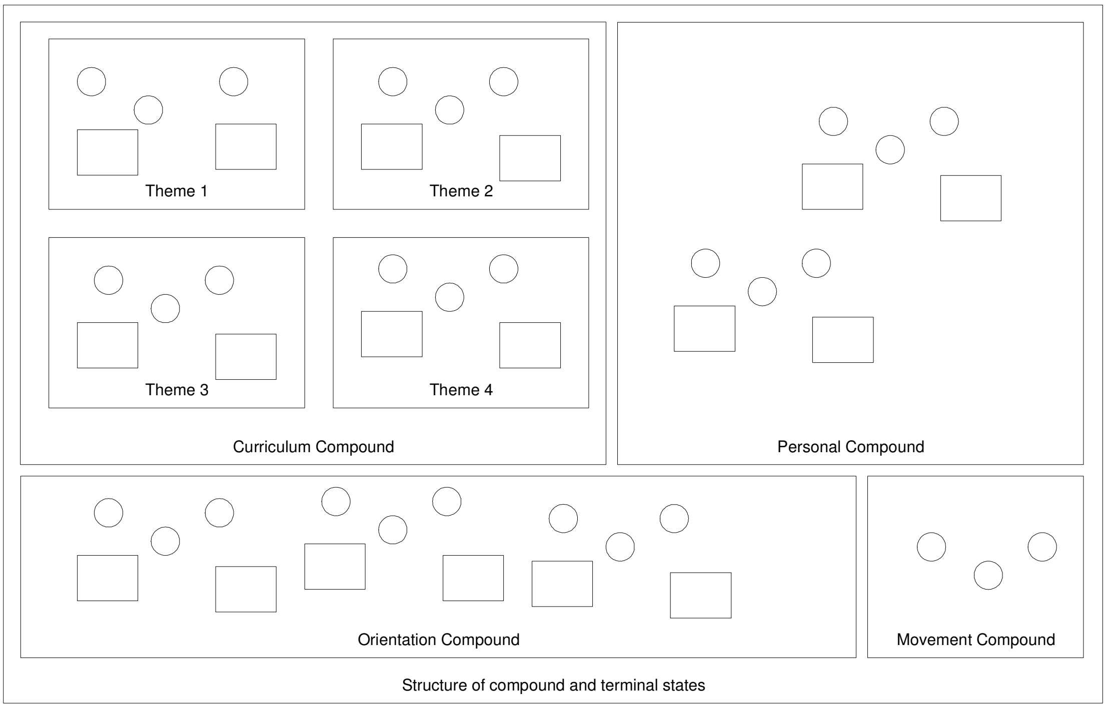


Figure 60 – An overview of the types of compounds employed for Ask & Answer.

A single Sequencing Engine, attached to the top-level frame, was employed. This Sequencing Engine has access to any basic state and compound of the system. This Sequencing Engine navigates through the different thematic compounds, searching for some state containing metadata that matches some text input of a user. The Sequencing Engine follows fixed search paths (cf. [Iurgel05a], [Iurgel05b]). For example, when the current active state lies within a thematic compound of the curriculum, e.g. on the French Revolution (cf. below), then first the other compounds of the curriculum that belong to the same current question will be searched, and then the personal compounds.

If no matching state is found, default states of the personal compound are activated – remember that Fritz, the virtual character, interprets input that it does not understand as insults.

It is necessary to cope with the presence of two teams. Because a team cannot hear Fritz when it is in the territory of the other, it is possible to reuse the states, i.e., Fritz can repeat things it has already said, when the other team is being addressed. Since memory states and other variables are dependent on the current team, they have to be reset when Fritz changes its position. The resetting of states and other variables is also a task of the Sequencing Engine. This is the single adaptation of the Hybrid Control Formalism, on a software engineering level, that has shown to be necessary in order to implement Ask & Answer.

Fritz' sympathy to a team is also stored in the data base. It is calculated by a XEXPR function "addSympathy(x)", where the range of x is [-1..1]. This function is invoked by the actions portions of the transition network. Display of emotions was scripted. E.g., should a participant insult Fritz, it would change its sympathy count and reply with some scripted utterance and dissatisfied emotional display. With the next scripted utterance, the unsatisfied emotional expression "disappears", and is overwritten by the new expression. This keeps the initial system simple, and is accurate enough to show to the players the consequence of their input, e.g. of their offence. Certainly, this is an example of an "understandable" reaction that is not "believable", in the sense that humans wouldn't react this way. The integration of some emotional model would not have been a major technological challenge, but would have increase authoring expenses and probably wouldn't improve the results. In fact, there is no emotional and social model available that would allow for an accurate control of such situations.

For measuring social distance, two variables, *sympathy_to_team_A and sympathy_to_team_B* are employed. The range is [0..1]. The function for sympathy does not distinguish between different aspects of sympathy, e.g. different levels of sympathy in different respects. The value of sympathy, if positive, decreases linearly with time until the value 0. This forces the participants to keep constantly talking to Fritz, attempting to please it.

The decision of Fritz to move to the other team is dependent on the variables score_of_team_A, score_of_team_B and the sympathy variables. The Sequencing

Engine is responsible for the decision on moving to the other team; for these calculations, heuristics are employed.

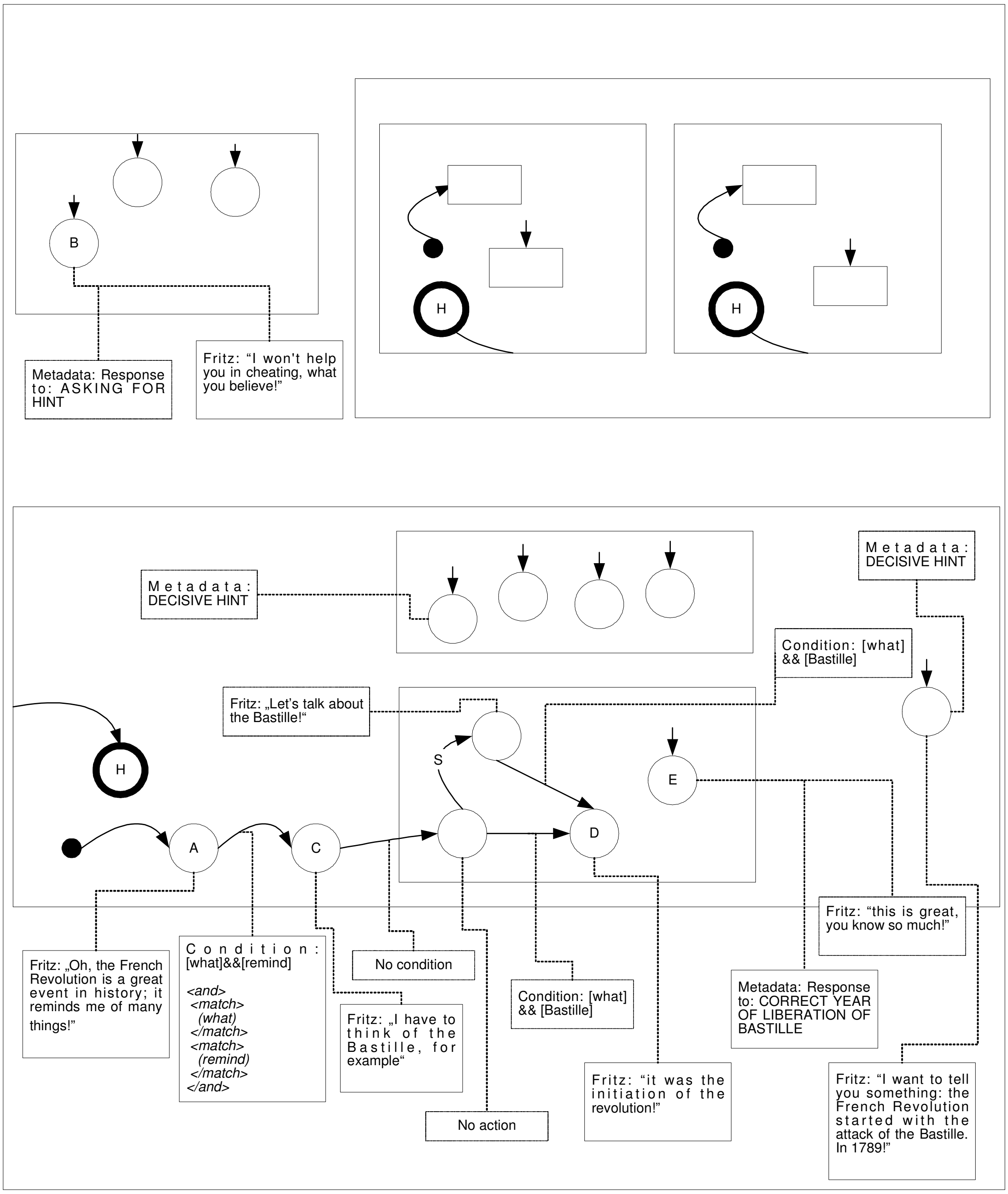


Figure 61 – An example of the hybrid framework, employed for guiding Fritz' chatting behavior.

Equally, heuristics decide on whether Fritz shall give the decisive hint to the participants, i.e. the hint that leads to the correct answer. Basically, it gives the hint if sympathy is above 0.5, and if some fixed time n has passed since the current game question was announced. After a time m>n, Fritz gives the hint proactively, i.e. without being asked for it. Thus, it is a good strategy to ask from time to time for the hint.

The running example will deal with the French revolution (Figure 61). Fritz has now joined the team B. The system has asked the game question: “what is the year of the French revolution?” Fritz immediately assumes the initiative, and remarks:

State A → Fritz: “Oh, the French Revolution is a great event in history; it reminds me of many things!”

This is an example of a sentence that was elaborated with some predictable reaction in mind. The player could predictably ask “what does it remind you of?” This expectation can easily be formulated with two regular expressions that combine to an “AND”-Boolean expression: [what] && [remind]. This expression is true if and only if the input of the user contains at least one “what and one “remind”. This is a simple, put practicable example. The principle of dealing with expected input wouldn’t change if some deep processing NLP were employed here, e.g. if some external NL-analysis module would allow for the more exact Boolean expression “The user asks what Fritz is reminded of by the question about the French revolution”.

Any other text input of the user would be dealt with generic mechanisms. For example, if the user asks directly the year of the French revolution. Fritz then answers:

State B → “I won’t help you in cheating, what you believe!”

For handling this, the Sequencing Engine will check that the user-utterance is a direct question about the correct answer, employing meta-data of the state that contains an utterance of Fritz that comments on the premature attempt of the player. Fritz won’t answer the question because, in this example, none of the preconditions are met: the minimum offset for answering has not passed, and Fritz does not feel obliged enough to the team to give a hint. The fact that this was an attempt to get the answer from Fritz is recognized by the global Sequencing Engine. It then activates the denial state with the utterance mentioned above. Then, the state A is set active again, so that the user can directly now ask about what Fritz is reminded by the question, continuing the thematic chat. A clever player would now ask: “What does the question remind you of?”. Fritz then opens the thematic dialogue with

> State C → “I have to think of the Bastille, for example”

This remark opens the field for different possible remarks by the user, either instigating Fritz to continue its talk –

> User: “what do you think of the Bastille?”
> State D → Fritz: “it was the initiation of the revolution!”

– or by inputting own knowledge about the theme:

> User: “the Bastille was liberated in 1789”
> State E → Fritz: “this is great, you know so much!”

These answers close the theme “Bastille”, and Fritz would go on with a new theme: “Do you know what the August Degrees are?”. The thematic dialogues would thus go on in a similar way for some time.

Suddenly, Fritz “changes his mind”, due to heuristics of the Sequencing Engine that guide when to enter the more personal theme:

> Fritz: “my wife left me!” (This is said with a very sad expression)

This enters the thematic compound of the personal affairs of Fritz, namely his marriage. It is not represented in the figure any more.

6.2.3 Conclusion of Ask & Answer

Ask & Answer is representative for the intended use and for the power of the presented system, and for the kind of questions that it shall enable to solve. Ask & Answer demonstrates that the system can be employed for shallow, but structured natural language processing; the heuristics that were mentioned are examples of the "procedural literacy" that is required, when creating such experimental IN applications; but once the heuristics are implemented, the content creator can rely on them, and add content without any programming literacy.

Ask & Answer contains also typical open questions that can only be addressed experimentally – which principle of controlling emotional expression is the most appropriate? How to deal technologically and conceptually with pitfalls of NLP? The only way of finding out whether a certain IN application idea in fact fulfills the initial expectations is by building it after solving the detail problems that were not initially foreseen. With the Cyranus framework, a sensible tradeoff between rapid development and complex control is available that allows for implementing novel IN ideas.

A conclusive assessment on the conceptual ideas and further technical requirement of Ask & Answer is not yet possible. Much more investment on content and experimentation with control strategies, and involvement of actual pupils is required to attain some conclusion. In particular, fine-tuning of the different parts of the system still poses considerable difficulties.

6.3 Virtual Human with Social Intelligence

The application design of Virtual Human with Social Intelligence is based on several novel assumptions (cf. [IurgelMarcos06], [KrämerIurgel05]). In particular, it assumes that

- A virtual shopping assistant can sensibly, proactively introduce personal themes, as long as the users normally perceive this as funny and entertaining.
- A virtual shopping assistant should actively guide the user, and not only be

reactive.

- Mini-dramas can be employed to create entertaining suspense during a shopping session. The mini-dramas are short story elements that involve the user into a personal problem with the shopping assistant.

Technologically, the project shows that

- Guiding behavior can be very easily combined with information seeking queries, reactions to mouse clicks on hyperlinks and text input can be integrated, and mini-dramas (cf. also [IurgelHoffmann06b], [IurgelMarcos06]).

6.3.1 Application Design

The project “Virtual Human with Social Intelligence” was commissioned by SAP AG. The task was to develop novel ideas on how to employ virtual assistants. As example, a shopping assistant should be employed. The author of this dissertation is the originator of the technological and conceptual aspects of the project that are described and examined in this section.

A first conceptual quest was to understand the role of soft skills of the virtual shopping assistant. In daily life, interacting with a person that is fulfilling a certain function – a functionary or employee, a sales person, a teacher, a work colleague – normally involves more than purely accomplishing rationally “hard” transactions. “Soft”, emotional factors play an eminent role for the success and the stability of the processes. E.g., a teacher should expect better learning results and continuity of work if the students feel at ease with him; a sales person is more successful if the customer likes him and laugh at his jokes. In short, “soft”, social and emotional skills matter very much. Their importance for the efficient operating of organizations is well known and studied.

The term “virtual assistant” is employed in a wide sense that covers any kind of virtual functionary, e.g. teachers or sales persons. Currently available embodied virtual assistants show only very rudimentary soft skills and personality traits. E.g., a virtual shopping assistant usually displays a constant smile as the only sign of emotional involvement, remaining otherwise passive and emotionally shallow[26] (cf. Figure 62).

Figure 62 - IKEA's Anna is an example of a tedious virtual shopping assistant.

But what exactly are the soft skills required for a technically feasibly virtual agent? Certainly, an exact simulation of human soft skills should not be intended. It is important to keep in mind that a virtual agent remains a metaphor; there is a certain analogy between a virtual and a human agent, but there are also essential differences. First, it is not possible to succeed in simulating exactly human social in virtual agents, in any predicable future, because of technological limitations. E.g., virtual agents are far from being able to understand subtleties of emotional and verbal expression of a human counterpart, and thus cannot be very precise when employing social skills. It is necessary to take those technological limitations into account, when designing social skills for virtual agents; otherwise, the task of really building them would be postponed into an indefinite future.

Second, social skills for virtual agents must differ from human skills because virtual agents are perceived and evaluated differently. E.g. a charming smile of a virtual assistant won't touch a human user as deeply as the smile of a sales person; but behavior that would be incontestable for a real functionary, e.g. lamenting about an unhappy love affair, might be funny and entertaining when presented by a virtual agent.

Because the required social skills and personality traits of virtual agents are not

[26] Cf. www.ikea.com/ms/de_DE/local_home/homeshopping.html for an example of a standard shopping assistant

yet well understood, the Cyranus platform was employed exemplarily for the creation of a virtual agent with specific soft skills, as an example on how to employ the platform for this exploratory task. The rationale is that content creators need to experiment with different ideas and intuitions, thus incrementally developing technically feasible and appealing virtual behaviors.

A. Gulz [Gulz] provides an excellent overview and discussion of social skills in the realm of virtual assistants, especially concerning virtual tutors. Though some very sophisticated systems were developed (e.g. [Bickmore], [BaylorKim]), no system was built from scratch on with emphasis on efficient authoring and on fostering experiments by content creators. Astonishingly, none of the agents was built with the expressed intention of being entertaining and funny, and no behavior patterns inspired by drama were employed. Nijholt [Nijholt05] has discussed humorous acts of embodied virtual agents to smoothen human-computer interaction problems. Most current virtual shopping assistants are commercial products, and their details are not published. However, no one seems to possess the guiding and pro-active faculties presented in this paper. The technology discussed in this paper is based on the work described in [Iurgel05a].

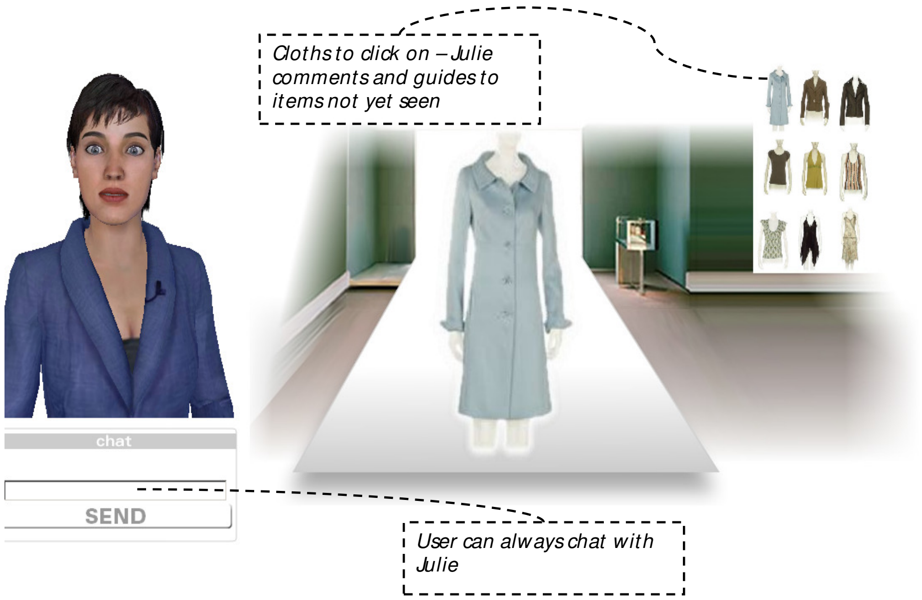


Figure 63 – This is a typical view of the shopping assistant Julie, on the IE. The customer can always ask questions about the shopping items and chat with Julie, or click on some cloth to have a closer look at it. Julie will always be responsible and smart, and also pro-active.

Basically, the exemplary scenario developed for SAP "decorates" minimal, functional behavior of a virtual agent with entertaining, relaxing, or motivating additional elements. Therefore, the soft skills of the agent Julie must be interwoven with rational and functional faculties. The goal is devise ideas and finally to understand what assisting virtual characters can sensibly do besides fulfilling their duties. In particular, the focus laid in short remarks and short emotional dialogue games that the virtual agent can throw in whilst providing information and, in short social games that enclose the functional, assisting behavior. I.e., object of the study was how to maintain the assisting functions while at the same time adding more social and emotional factors. E.g., Julie shall always say the price of an item, when the user requests it, thus staying within its role, but that also throws in emotional remarks and interjections, and that plays engaging "mini-dramas". Figure 63 shows a screenshot of Julie.

Mini-dramas are temporal structures that create suspense and end with a resolution. It is where interactive narrative enters, experimentally, into the project. A mini-drama is a minimally short narrative structure[27]. It aims at providing additional structuring of time for the user, who should feel compelled to learn the solution. Mini-dramas shall enhance motivation, enjoyment and fun. Also, the social and emotional dimension of the communication shall be enhanced, in an analogy to the creation of social bounds and reciprocity in real human interaction.

The suspense of a mini-drama shall provide additional reasons for the customer to remain at the shopping site, or, with respect to learning applications, for the pupil to continue his e-learning session. E.g., Julie will begin throwing in unhappy and nasty remarks which she first refuses to comment (conflict), increase their intensity until a climax is reached, and then confess that she desires to become friend with the user, but doubts whether this wish is reciprocal (resolution). We expect the users to be pleased and laugh at this turn around, and not to be deeply touched by this virtual advance. Following drama theory, this mini-drama has revealed a hidden, unexpected trait of Julie's "personality", namely her (alleged) affection towards the user (cf. Section 3.1). As in cliff-hangers of TV-soaps, mini-dramas could also be used to create continuity, suspense and curiosity when returning to a learning session or to an internet shopping site.

It is not common for e.g. real sales persons or teachers to actively employ mini-dramas to engage the counterpart. But this is an example for the major difference between virtual and real agents, and clarifies why we need a framework for experimenting with the specifics of virtual assistants: A real teacher could lose his face and his privacy, when uncovering hidden personality layers and feelings; a real sales person could strategically flirt with a customer, but would appear aggressive and loose his credibility when playing with deep feelings. Those are no concerns, when it comes to a virtual teacher or shopping assistant. The really challenging interaction design questions in the virtual realm are related to balancing and estimation of effect.

"Balancing" means that, in applications that employ virtual assistants, some hard criterion for success exist and that extra-functional behavior of the virtual agent must, at the end of the day, have contributed to this goal. As in any real working context, we are better of with social skills, but we won't achieve the targets if we only focus on socializing. Some difficult balancing is required. E.g., the funny behavior of a virtual shopping assistant must not attract totally the focus of the customer. It should be useful in attracting more site-visitors and motivate them to buy. A virtual tutor in a business context must not consummate too much time with creating emotions; the emotions must be beneficial for the final learning success.

"Effect" is a problematic design aspect because the impact on a particular user is still very difficult to foresee. Here, again, the analogy between a virtual and a real human can be misleading. If the user laughs at a joke told by a virtual character, how much is he laughing about the joke, and how much about the fact that a soulless thing is pretending to tell a joke, employing a dry synthetic voice and imprecise mimic? In the later case, would the user still laugh at the n^{th} joke? The effect of a behavior well known on humans is not easy to foresee, when simulated by a virtual human.

Though balance and effect can be evaluated formally, the process of optimizing them is essentially an intuitive, experience driven, and even artistic process, in the same vain as screenwriting. Hence the importance of a system that supports experiments and that can be authored by content creators that are not computer specialists.
The exemplary cloths shopping site is intended for young women and girls. We expected that a considerable part of this group would react to attributes as “cool”, “trendy”, “funny”[28]. The visual design of the site is inspired by real high class cloth shops, in order to create a young, sophisticated and demanding atmosphere that differs from usual internet shopping sites.

The shopping assistant *Julie* should be able to guide the customer through her shopping experience. Julie should comment on mouse clicks of the user, e.g. saying “If I were not caged in this computer, I would myself buy this jacket!”, or initiating short dialogue games, e.g. “do you want me to tell what I really think about this cloth?” – “yes, please” (typed) – “it will make you look overweight”. Julie should not be repetitive when the user clicks more than once on an item, and should actively guide the user to items not yet examined, e.g. saying “look at the jacket on the top right corner. It will certainly fit you!” – (user clicks at item) – “And? What do you think? Am I not right?”

Julie’s remarks are not only informative, but sometimes teasing, joking or even bitchy. Of course, this is intended to create fun. Julie is pro-active, and will e.g. sometimes try to instigate some off-topic chat, e.g. on the party of last night. Julie also can comment on the overall session and the reactivity of the user, e.g. saying “you are wasting my time if you do not do anything”. At any time, the user can freely click on the items and chat with Julie.

These examples already indicate the personality profile we have created: Julie is much more active, demanding, direct, and even nasty and aggressive than other virtual assistants.

[28] The choice of a scenario for young girls and of a cloth shop is joint work of me and of Anja Hoffmann.

Julie is intended to be a rather extreme example of a personality rich, active, emotional assisting agent. This shall enable further examinations on whether and by whom this is perceived as distracting and offensive, or funny and attracting. For commercial use, marketing considerations would certainly require the reduction of the intensity some parameters.

Experimentally, mini-dramas were devised for Julie. The mini-drama “I want to be your friend” was implemented. Julie will begin by throwing in some nasty remarks like

“I don’t think that we are going to get along with each other” – “why?” (typed) – “I do not want to tell”.

A bit later, Julie will resume the topic:

“Do you really want me to tell you what the problem between us is?” – “yes” (typed) - “I think I really want to be your friend, but this is not possible…” (makes a very sad face).

This mini-drama shall, at its beginning, create suspense, because the user shall be intrigued to know Julie’s reasons. It shall after its resolution initiate some more intimate exchange, and the creation of some sort of social bounds between user and Julie.

6.3.2 Applying Cyranus

Within the project Virtual Human with Social Intelligence, two major technological tasks had to be solved: (i) Enabling easy to author proactive, smart chatting behavior of Julie (the virtual shopping assistant); (ii) combining chat and mouse clicks; and (iii) enabling experiments with mini-dramas.

The chatting was again implemented with the help of an ALICE-Fallback Engine, as in Art-e-Fact (cf. Section 6.1). Equally, the Sequencing Engine of Art-e-Fact was employed for guiding the user through the cloth selection. Thus, the user is guided to cloth that he did not yet look at, and “intelligent” remarks are made if the user clicks more than once at a cloth. Additionally to the guiding behavior, Julie also initiates conversation themes, e.g. about a party it was at last night. This initiating follows time based heuristics, e.g. when the user is inactive for a certain time n. Short dialogue games are represented as directed graphs; when the user types something that does not match any of the conditions of the graph, the text input is send to the ALICE-engine, which provides a generic feedback.

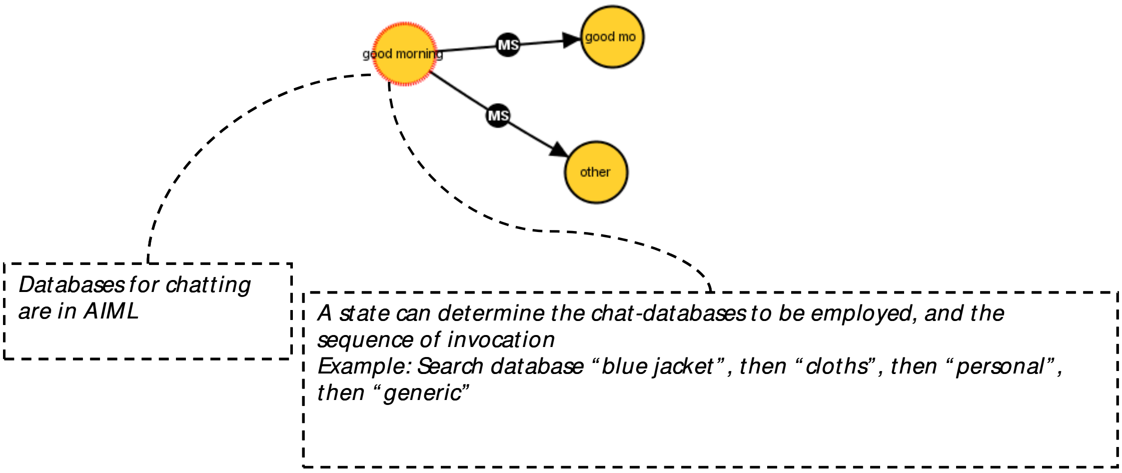


Figure 64 – When chatting, the ALICE-Fallback Engine can be configured to employ certain data bases in a certain sequence, in order to generate the best possible response.

The single ALICE engine employed here can be configured with dedicated XEXRP-expressions, in order to allow for a controlled search of the chatting data bases (Figure 64). E.g. a greeting dialog game is modeled with directed graphs as a predictable sequence of Julie saying "hello", and the user answering a typed "hello". If the user types in a question on the price of the visible cloth instead, Julie will be able to answer, because the text input is send to the ALICE data bases, and ALICE was configured to search the data bases in a sequence that prioritizes the data based on this particular cloth. Employing the event forwarding chain (cf. 4.2), both the directed graph and the Fallback Engine can be made to answer. For example, Julie could remark "Oh, you are not polite", employing a transition of the directed graph that matches any text input other than "hello", before responding with the price.

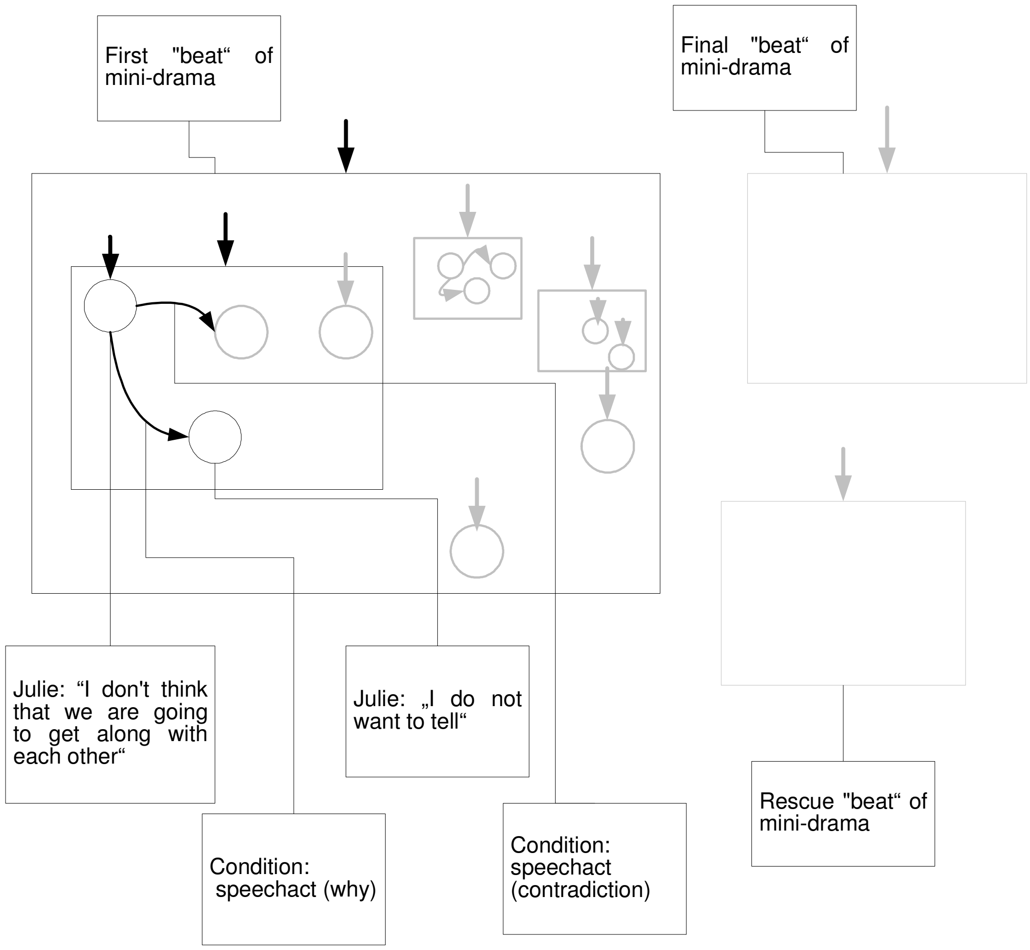


Figure 65 – Structure of the "beats" of a mini-drama. Each beat is organized as a compound of the hybrid framework. At the core of a beat are short dialogue games that can be modeled employing directed graphs, and which Julie initiates proactively. The sequence of the dialogue games, and the sequence of the beats, is controlled by "ad hoc", experimental Sequencing Engines. Cf. [IurgelMarcos06] for more details.

Following a strategy similar to that employed by Mateas and Stern (cf. [MateasStern06]), concerning their use of beats, "mini-dramas" of this project are encoded as compounds; each drama part that shall be played as a whole, without interruptions, is organized in a compound. For example, the announcement of Julie that it does not like the user, and a short conversation hereafter on the topic, until Julie's refusal to continue dwelling on it, is a "beat" of a mini-drama, i.e. a compound. The initial, experimental structure consists of only three beats. The second and final beat-compound of the drama contains the revelation of the deep sympathy that Julie feels towards the user, and that lies behind the apparent repulsion of the first beat. A "rescue" beat is employed when the first beat is not successful, i.e., when the user does not react appropriately to the first attempt at dwelling on the theme of sympathy. The “rescue” beat is a second attempt to save the mini-drama. Figure 65 shows an example of the beat structure.

Other than at Ask & Answer, where the user is expected to adapt to the limited faculties of the system and thus no generic, Eliza-style responses are allowed, in the context of Julie’s behavior modeling, a Fallback Engine is employed (Figure 66). This is because of the application design of a shopping site, where the user is not expected to spend much efforts trying to “understand” Julie.

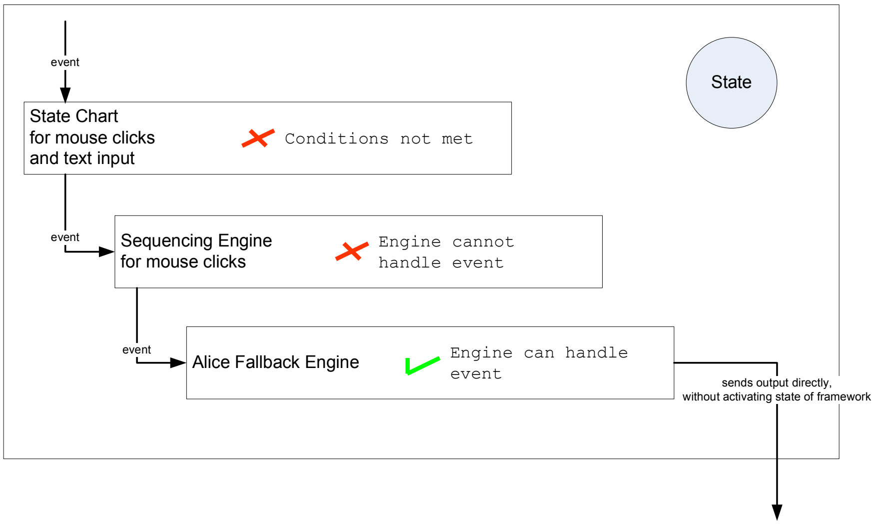


Figure 66 – The project “Virtual Human with Social Intelligence” employs a fallback engine in order to guarantee a sensible response to unexpected text input.

6.3.3 Conclusion of Virtual Human with Social Intelligence

A commercial version of Virtual Human with Social Intelligence would necessarily require extensions and changes:

- Incorporation of large and changing databases of e-commerce systems
- use of templates and of a knowledge base, instead of fixed sentences, in order to employ the databases as story sources

6.4 Virtual Human

The project Virtual Human introduces partially autonomous virtual characters. Important results of the project are

- The development of an appropriate communication language for control of the partially autonomous virtual characters by the Cyranus framework. The communication language can be derived formally from a given ontology, and thus allows for authoring tool support (cf. [GöbelIurgel]).
- Development of measures to integrate the ontology into the framework
- The Cyranus framework could be employed for the requirements of the demonstrators, which demonstrates its efficacy and generality (cf. also [Iurgel06c]).
- Theoretical and conceptual insights into how to author partially autonomous virtual characters (cf. [Iurgel06a]). These are described in Section 7.

6.4.1 Application Design

Virtual Human (VH) is a German research project that was successfully accomplished in December 2006. VH has produced demonstrators, but, comparing to the other projects described here, there was less focus on application design issues. The aim was the creation of a generic framework (in this respect much resembling Art-e-Fact) that employs virtual characters with an increased level of autonomy – "Virtual Humans", in this sense. The main research topic of the author of this dissertation, within VH, was related to the architecture of the system, to the applicability and extensions of the generic hybrid framework described below, and to the study of novel authoring principles required by the increased autonomy of the virtual humans (cf. Section 7).

The global application design of the two demonstrators was joint work of the consortium under the lead of partner RMH. The first demonstrator was a learning application, where a real pupil could interact with a virtual pupil and a virtual teacher, which spoke about the planet system and asked questions on it.

The second demonstrator (Figure 67) was a soccer quiz simulation with two games. In the first game, two players competed against two virtual characters. A video on a soccer game was shown to the participants. The video was stopped, and the participants were asked, sequentially, how they thought that the video would continue, e.g. whether a goal or a foul would follow. The possible answers were presented by a moderator, and the players could answer referring to the moderator, e.g. saying “first option”, or by mentioning the possible outcome, e.g. saying “a goal”[29].

The human winner of this first round moved forth to the second game. Here, the player had the task of choosing and positioning the German soccer team against some adversary, e.g. choosing the best soccer players and their positions against Italy. He had a limited amount of time for this, and his choices lead to a final score. A virtual player form the first round and the moderator were also present, during these choices, and the human player could ask these both for advice, and they commented on the moves of the player.

Figure 67 – A screenshot of the second demonstrator of the Virtual Human project.

Both demonstrators aimed at proving the generic quality and feasibility of the system: Quite disparate applications can be built with the same system. VH aims much more at a simulation of humans than the other projects presented in this dissertation. But this approach remains a questionable promise when the authoring process does not offer a content creator the methods to employ the modules. Thus, a main research challenge of VH has been devising appropriate authoring

[29] The language of the application was German.

methods and processes. A contribution to this challenge is partially also described in Section 7.

6.4.2 Applying Cyranus

The project Virtual Human (VH) adds new challenges to the authoring framework, but also allows for reusing many of the central elements of Cyranus. The architecture could be adopted from the initial architecture of the Art-e-Fact project (cf. Section 6.1, [Iurgel03b], [IurgelHoffmann04]), as far as the usage of a Narrator as central story control instance is concerned. The new feature is the high level of autonomy of the virtual characters; In VH, the agents are able to accept parallel high level goal directions. The partially autonomous virtual characters are called CDEs (Conversational Dialogue Engines). The Virtual Human architecture also contains a central ontology in XML that represents, among others, any entity that can appear in a certain story, the commands that the virtual characters can accept, and their hierarchical and slot structure (Figure 68).

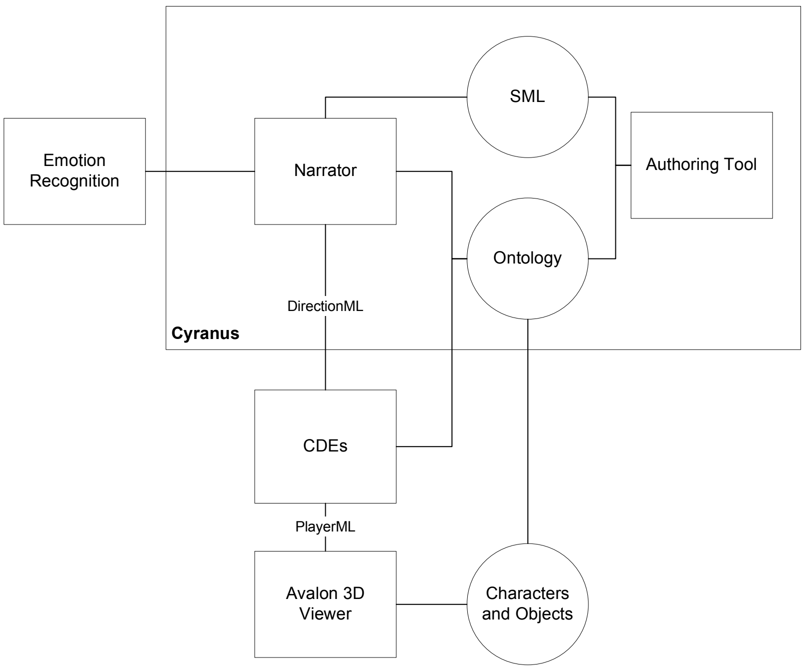


Figure 68 – The basic architecture of Virtual Human. The ontology is the common data storage for the authoring tool, the narrator, and the virtual characters.

Thus, a major task of VH was the development of a command and communication language for directing autonomous virtual characters. Because of the complex structure of this communication language, special XEXPR functions had to be developed in order to enable the access of every feedback of the virtual characters by the Narrator.

The communication language for autonomous virtual characters is called DirectionML. The author of this dissertation intended DirectionML to be a formal mapping from the ontological structures, because this is a prerequisite for creating an authoring module for DirectionML. Thus, an authoring tool is able access the ontology and to offer the possible commands to the author, based on the ontological information. This link of the declared faculties of the virtual characters through an ontology to the command language makes DirectionML unique among direction languages for virtual characters. The details of DirectionML were developed in discussions with partners of the project.

Since DirectionML is derived from an ontology, it is impossible to provide a standard XML format definitions. Therefore, a non-standard, “macro”-terminology for expressing the structure of DirectionML was developed, which is used in the Annex B.

An example of DirectionML follows:

```
<setGoal identifier="play">
             <Participant identifier="Fritz"/>
             <Participant identifier="Hans"/>
             <Goal>
                    <PlayRoundOfFirstGame
                           identifier="play2">
                           <has_theme>
                                  <Video identifier="Video WM 1974"/>
                           </has_theme>

                           <has_moderator>
                                  <Character identifier="Hans"/>
                           </has_moderator>

                           <has_assistant>
                                  <Character identifier="Fritz"/>
                           </has_assistant>
                    </ PlayRoundOfFirstGame >
             </Goal>
</setGoal>
```

In this example, the Narrator is setting the goal “Introduction”, which is sent to the virtual story participants “Fritz” and “Hans”; the ontology describes three slots for the goal “introduction”: “theme”, “moderator”, and “assistant”, these roles of the goal are assigned to the appropriate ontological objects, i.e. “Video WM 1974”, “Hans”, and “Fritz”, respectively. The structure below the “Goal” tag is a valid instantiation of the ontology.

The participants report on their success or failure equally in DirectionML. An example follows:

```
<GoalFeedback identifier="play">
             <Status>
                    <Success/>
             </Status>
             <has_source>
                           <Character identifier="Hans"/>
             </has_source>

<result>
                    <Character identifier="Fritz">
                           <has_ranking>
                                         bad_player
                           <has_ranking>
                    </Character>
      </result>
<GoalFeedback/>
```

In this example, the virtual character Hans is reporting on having succeeded with the task, and it delivers additional information on the character “Fritz” being a “bad_player”, in this round. This feedback can be directly evaluated in the conditions of transitions; irrelevant substructures can be omitted in the condition, cf. Section 5.

For example,

```
<event>
      <Status>
                    <Success/>
      </Status>
             <has_source>
                           <Character identifier="Hans"/>
             </has_source>
</event>
```

is a valid Boolean function of the Cyranus framework; it evaluates to *true* when the example feedback arrives as input.

Sometimes, it is not possible to evaluate conditions by simply matching them with incoming events. For example, the current game score of a player may be required to decide on the transition to follow. For these cases, the Narrator has full access to the ontology. For accessing the ontology, the XEXPR-function “RDQL” was introduced30, which can be employed in the conditions and actions of states, also within a DirectionML command or GoalFeedback-condition. Within the <RDQL>-tag, any RTF-ontology query can be expressed, and the return values can be used in Boolean expressions of conditions, or as instantiations of variables for actions.

Thus, the main concrete adaptation requirement of the Cyranus system to the Virtual Human project lies in devising means for integrating the ontology, and in devising an appropriate high-level command language. The required extensions to the authoring tool involve only the implementation of a DirectionML and of a RDQL editor for use in the configuration tabs (cf. above, and Section 5).

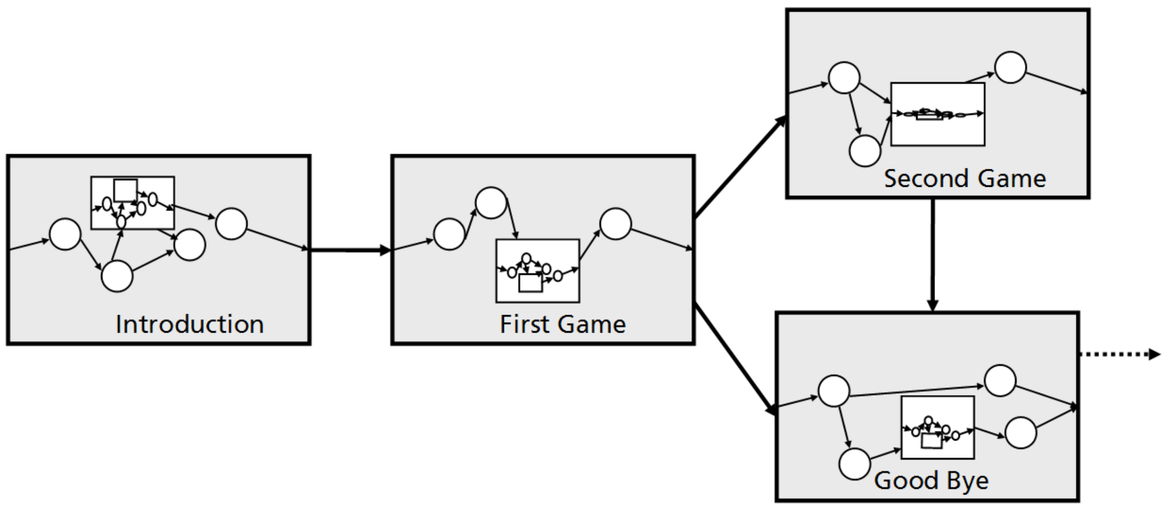


Figure 69 – The structure of the control graphs for the Virtual Human project could be easily mapped to story elements. In this example, the compounds represent phases of the quiz-simulation of the second demonstrator.

The demonstrators of the project were created employing strategies and patterns that were already described above, in the context of other projects (cf. also Figure 69).

[30] Cf. http://www.w3.org/Submission/RDQL

6.4.3 Conclusion of Virtual Human

This overview shows how it was possible to extend the Cyranus system to guide autonomous virtual characters by stressing on the formal properties of the direction language. Apart from these proven and implemented solutions, the Virtual Human raised very fundamental questions concerning the authoring process of autonomous virtual characters. This future work is addressed in Section 7.

6.5 Lessons Learnt by Applying Cyranus

The examples have shown the flexibility and aptness of the Cyranus framework for creating interactive narratives in spite of the "wickedness" of the research area. In fact, a variety of different application with IN aspects have been implemented and put into practice. The Cyranus framework and the Hybrid Control Formalism have proven to be efficient and generic.

Future research on Cyranus must address:

- Visualization of the run-time processes.
- Instantaneous authoring at run-time.
- Visualization of the structure of the different engines, in dependence of the compounds.

Concerning the application design, further authoring and production efforts are necessary and formal evaluations are required to guide future steps.

To be sure, for a commercial scale usage, customizations and adaptations can be advantageous or mandatory, depending on the concrete goal. Cyranus' focus lies on experimentation and research oriented development rather than in commercial production – assuming that in commercial production it is usually clear what the outcome shall be and the technologies that shall be employed.

7 Creactor – Principles for Authoring of Autonomous Characters

7.1 Introduction

This section deals with the question of how to enable concrete authoring of autonomous virtual characters. It presents general concepts. These are the central ideas:

- **Concrete Authoring.** Authoring of partially autonomous virtual characters should be based on actual story situations. Only when dealing with an actual example can the content creator decide whether the autonomous behavior of the virtual character is appropriate, or whether any correction is necessary.
- **Assisted Generalization.** The authoring system should assist the content creator in the generalization of the concrete case.
- **Management of Side Effects**. Side effects of generalizations and changes to the system must be made immediately visible to the author
- **Learning and Adaptation.** The framework must provide graphical and software interfaces for experimenting with learning algorithms, i.e. automated, assisted generalization.

The fact that learning algorithms, and especially Case-Based Reasoning (CBR), are most promising for the progress of IN research, was already recognized within the research area of IN, cf. for example [FaircloughCunningham]. But no attempts have been made to create veritable virtual actors (partially) based on CBR, or in the context of an abstract authoring framework for IN. The ideas presented below on assisted generalization and concrete authoring are inspired by the work of [GervásDíazAgudo] on employing CBR as basis for story plot of new linear stories. However, CBR has never been adapted for character-based interactive storytelling. My preliminary studies on these issues are described below, and partially in [Iurgel06a] and [Iurgel06c]. The central ideas that are presented in this section of the dissertation are also described in [IurgelMarcos07].

The Virtual Human project has shown how to integrate autonomous virtual characters into the Cyranus framework (cf. Section 6.3). As described by [Aylett05], authoring of autonomous agents considerably diminishes the control of the author on what will happen in an interactive session. To be sure, the author is in charge of some abstract commands or constraints and can change some parameters; but if the virtual character in a given story situation does not act as the author expects, the author may have no means to bring the virtual person to follow his intentions. For example, the author may want a virtual character to show a specific emotion, and a specific gesture, in the concrete situation.

This is related to the knowledge acquisition bottleneck of authoring virtual characters, and also to the “wickedness” of IN creations: Only when seeing a concrete reaction of the virtual character, in a concrete situation, will the author know what he really expects the character to do. What is more, even as an experienced and talented IN developer he may not be able to express his intuitions in form of rules.

In Sections 2 and 3, the hypothesis was defended that character focused IN has an imminent counseling and even therapeutic component: interacting with other people, while they are undergoing difficult inner transformations posed by obstacles and stakes, naturally leads to counseling and even therapy. The author of this dissertation believes this to be a very central answer to the question of what IN can be: An essential design category for interactive narrative is exactly a sort of counseling training simulation, with a “dramatic” exposition of the problems and the consequences of the influence of the user.

With an approach based on sequencing, as e.g. in Cyranus and Façade (cf. Section 2.2), the “therapeutic” interaction itself must, however, remain shallow, because only a very limited set of reactions is available, and the story must go on when reactions are consummated, or when the user input is unexpected. Also, the quantity of the required authoring efforts required from the content creator can be huge.

Autonomous virtual actors with autonomous language reactions, emotions, and action choices could help alleviate the authoring burden. But the models available for emotion and personality are currently not “deep” enough for modeling interesting psychological transformations and counseling dialogues (cf. Section 3.2).

Thus, for creating psychologically sensible INs with autonomous virtual characters, a different approach to authoring is required. The focus must be much more on the exemplary case; the author should be able to determine, on different abstraction levels, what the character shall do *now*¸ if he is not satisfied with the autonomous behavior. The authoring framework then would support the process of generalizing from this particular case.

In this section, the components of this future work will be exposed and analyzed. In the next section, a short example of an exemplary application design for a psychologically "deep" narrative will be provided, in order to clarify the appropriate context of this research.

7.2 Psychological Intricacies for Virtual Characters – Exemplary Application Design

This is the sketch of an application design that exemplifies the requirements on virtual characters that shall function within psychological "deep" IN.

In Capote's "Other Voices, Other Rooms" [Capote], the boy Joel, the protagonist of the story, feels repudiated by the behavior of a girl, Idabel. The initial repudiation that Joel feels towards Idabel is due to the fact that both are outcasts ("freaks", in Capotes words) within a conservative rural scenario. Thus, Joel is seeing in Idabel a mirror of his own position, and he initially dislikes the implications – and therefore feels repudiation. This is an example of the intricacies of the soul that must enter IN research. The sense of such psychologically deep INs is to confront the user and enable him to deal with such non-obvious psychology and rules of social life, presented in a way that is touching and elucidating. In [Iurgel03a], the author has argued that a rather fixed story line without many bifurcations should be sufficient for creating such an application, if the virtual actors are intelligent enough to catch interaction deviations. Figure 70 shows the corresponding architecture.

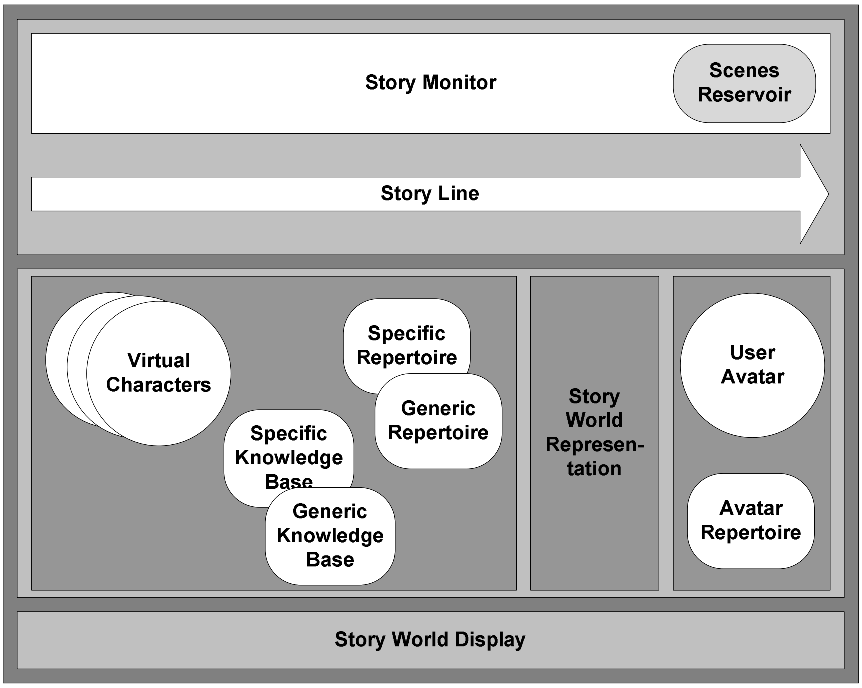


Figure 70 – With very smart virtual characters, the story line does not need much adaptation, because the virtual characters bring the story autonomously back to the main plot.

Section 3 has stressed on the importance that "deep" psychology can play in IN, and that talking to a virtual protagonist can be made part of the game challenge (cf. also the motivation for Ask & Answer). The hypothesis is that psychologically deep interactions are an important missing conceptual link between "interactivity" and "storytelling". Following McKee's theory of story as revelation of "deep layers of personality" (cf. Section 3.1), a huge potential for interesting and successful IN-applications lies in learning how to devise and implement psychologically "deep" and elucidating interactions with virtual characters. Thus, this sketch represents a guideline for future research and creativity.

Employing autonomous virtual characters can be decisive for the creation of the impression of psychological depth, but control of what happens in interactions must be put in the hands of the gifted content creator, who requires a concrete authoring process.

7.3 Authoring on the Situation

Concrete authoring is authoring on a certain situation: "Now, do that!" "Now" refers to the problem of generalization, "do that" to the problem of adaptation. When a framework exists for concrete authoring of autonomous virtual characters, this framework can be very useful for a varied kind of applications, including linear animation. A finite state based approach fosters concrete authoring, in the sense that the active state defines the current situation. The author of this dissertation believes that, for non-finite state based technologies, authoring technologies should be developed that equally foster a concrete creative process.

A lesson learnt from the Cyranus framework is that the authoring system must be co-developed with the application design and the control technologies; in the same sense, an authoring tool for concrete authoring of autonomous characters should provide a framework for a stepwise development of the technologies. For simplicity of reference, the model of this authoring framework will be called “Creactor”.

The concrete interactions that fit the intuitions of the content creator can be compared to the facts that a theory shall explain. The domain models and their parameters can be compared to the theory that shall explain the facts. Creactor can foster generalization in three ways: (i) by assisting the user in developing the rules of the model, (ii) by automatically verifying whether a human made “theory” is correct, given a set of “facts”; and (iii) by automatically assuming generalization tasks, i.e., by creating a “theory” over the set of facts.

The problem of automated verification will be addressed under the heading of a “set of endorsed stories”:

7.3.1 Employing a Set of Endorsed Stories

Changing models, technologies and parameters that have a global impact on the behavior of virtual characters is problematic, when the content creators have a certain behavior in mind. A change in the algorithm that is intended to produce a certain desired effect at a certain situation S may also cause an undesired effect at situation S’. Without a dedicated process for handling side-effects, this derogation of its behavior cannot be systematically checked, and may remain unnoticed for a long time. Again, repairing a faulty reaction can produce unnoticed side-effects, and so on. This cycle is depicted in Figure 71.

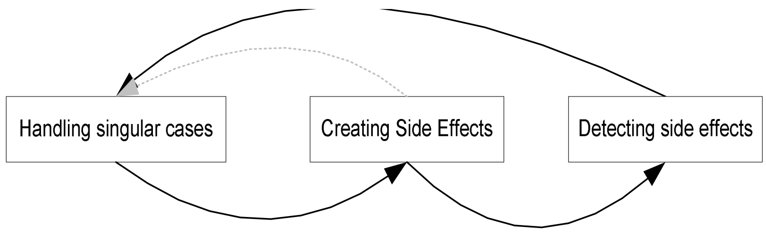


Figure 71 – The problematic cycle of fine tuning the system. The desirable process leads directly back from the side effects to the handling of the case, whereas a lazy detection is unproductive and unmanageable.

This problem can be alleviated with the use of a repository of endorsed stories or story fragments, i.e. of stories (or fragments) which the author thinks to be already finished, and to be in accordance with his intentions (Figure 72). When some globally affecting change on models occurs, the system shall “recompile” the repository of endorsed stories, and visually indicate side-effects. The author then might endorse the change, or further modify the algorithm in question. “Recompile” means here exactly that the stories of the repository will be checked with respect to possible effects of changes of algorithms; this is equivalent to automatically “playing again” the same story in an “express mode”, so that modifications become rapidly visible.

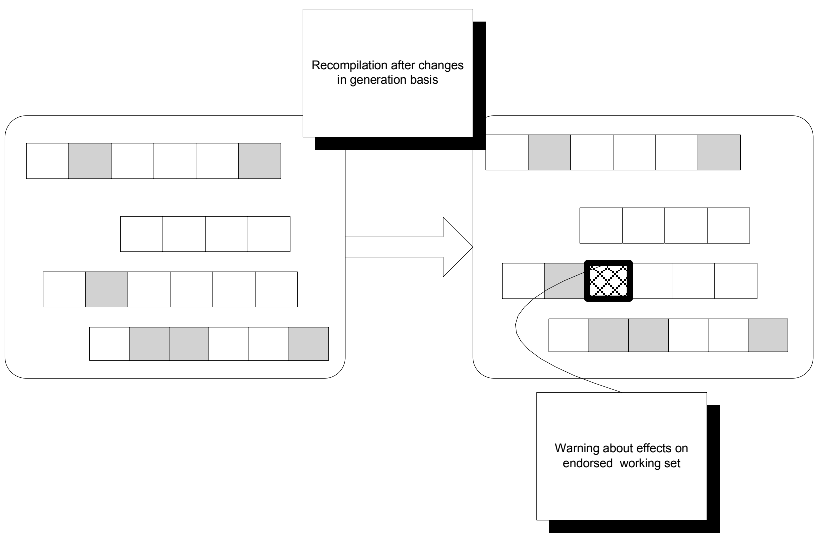


Figure 72 – Employing a set of endorsed stories to verify consequences of changes to the system.

Depending on the algorithm, recompilation in this sense can be a non-trivial algorithmic task. Creactor must contain the necessary interface specifications and software-engineering framework that allows for recompilation. To the best of my knowledge no attempts had been made yet to keep modifications of control systems for multimedia applications (including interactive narratives) under such a tight and systematic control.

“Rollback” is a sibling concept to “recompilation”.

7.3.2 Rollback

Where at recompilation the modifications under observation are made on the control system, e.g. by exchanging models, at rollback, the differences at hand are related to external input, or to modifications of parameters. In rollback, we want to examine a “what-if” question: what would happen if a certain input or parameter were different, at time t, in the context of a specific narrative N? To check this, the author will want to determine, during a second run, a certain time t of N to which the narrative shall roll back, i.e. continue at exactly the same state as during the first run. He then can experiment with different inputs. As a part of the concept of endorsed stories, rollback too is a combined software engineering and user interface task that defines the framework for concrete implementations.

7.3.3 Assisted Modeling

Further on, the framework should support the creation of models by providing specific assistants. This assistance depends on the implemented module. But the framework can support it by providing interfaces and specifications for the data of the current world state, implemented as ontology, e.g. by providing an interface to a blackboard that contains the world data. The rationale behind this interface is to allow the assistant modules to provide the data on which to generalize (Figure 73). For example, when the author wants a virtual character to react angrily, at a certain situation, the system provides a description of the current world state, which contains the description that the character is a friend of the user, and that the user has insulted the character. The user then can employ these predicates to express rules as “whenever the character is friend of the user, and the user insults the character, the character shall become angry”. Allowing for this assisted modeling feature requires the building of a specific ontology. Most important is its combination with the set of endorsed stories: the framework should provide the modules with a difference set, i.e. with a set of the differences between the ontological descriptions between two concrete stories of the endorsed set. Thus, for example, the user can see immediately the differences between an endorsed story where the rule he has generated is correct, and another story where the rule produce faulty results; e.g., that the rule mentioned above is not correct when the virtual character is tired.

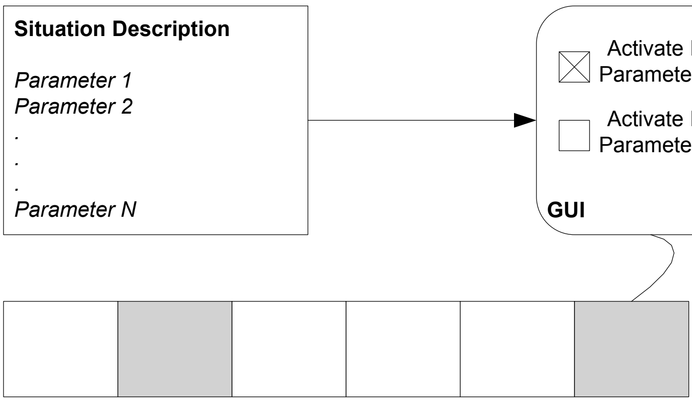


Figure 73 – A simple example of an assistant assisted modeling, where the author marks the parameters of the situation that shall be employed on the left side of the rule that defines when to use a concrete behavior.

The features described so far shall enable ad-hoc and experimental modeling and parameterization. They facilitate a neat combination of top-down (abstract) and bottom-up (concrete) character behavior definition steps. Given the restricted context of a particular story, and of a particular character, it is possible to invent "deep" models that possibly are not as generically valid as scientific models, but that create a sufficient illusion for the particular application.

7.3.4 Machine Learning

A learning module would much facilitate the authoring process, since explicitly modeling of behaviors would interrupt, complicate, and endanger the creative process. Optimization over a set of parameters, e.g. over a set of personality traits to create the emotions defined in the endorsed story set, is a very promising approach, but limited to cases were such parameters are available and the model sufficiently covers the desired behaviors (e.g., the author might want a combination of emotional reactions that cannot be defined by the personality parameters).

Developing a learning module is a feasible task. Certainly, only knowledge-rich methods can be employed, since any statistical method requires a too large database of endorsed stories. The ideal authoring method would be achieved if the author only defines a limited set of *typical* stories, actions and reactions, and the system employs these typical cases as guidelines for controlling behavior during further interactions.

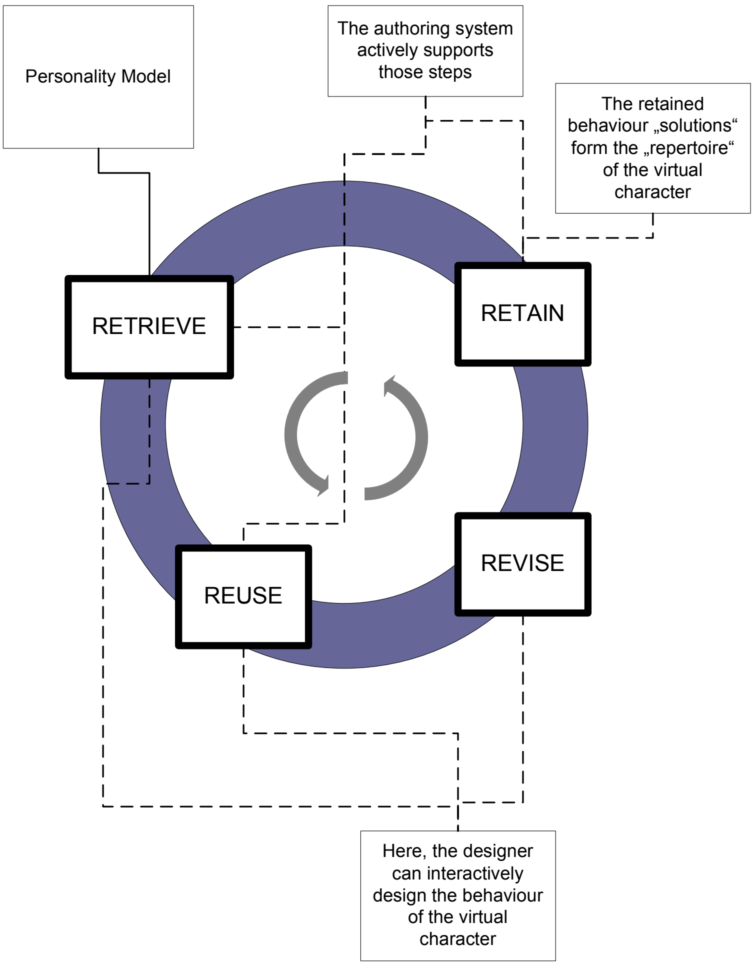


Figure 74 – The CBR-cycle applied to designing behaviors for virtual characters.

The most suited technology for behavior teaching of virtual characters is probably Case-Based Reasoning. Some storytelling systems have already started experiments on CBR, with adaptation of the story structure being in focus (cf. [FaircloughCunningham], [Swartjes]). In my view, CBR is promising in the storytelling domain because of the notion of “analogy”, which translates easily into “typicality”. CBR is the technology that relies on the storage of concrete cases and of reusing them, with the help of adaptation algorithms and a similarity function, to find the most appropriate case out of the data base of cases. Cf. Figure 74 for a diagram of the CBR-cycle, adapted to virtual characters in IN. It is known for its appropriateness to domains where the problem is badly describable and structured, as in IN (cf. [AamodtPlaza]).

Ideally, the author of a virtual character in IN would do nothing but define exemplary cases, possibly assisting the system with some metadata that explain why he is choosing a particular behavior, employing GUI-based dialogues for this explanation. All of the knowledge of domain experts and of ad-hoc algorithms remains hidden in the similarity, adaptation, and storage functions.

The advantage of employing analogy over direct modeling lies (i) in the possibility of reusing structures that are not modeled, but that contribute to the desired effect, and to (ii) approximate the envisaged solution by increasing the data base of cases. In fact, the "set of endorsed stories" mentioned about maps directly into a set of successful cases, though it is not necessarily identical with the cases of the data base, which may for example also contain failure cases, i.e. cases that did not please the content creators.

Figure 75 exemplifies the advantage of reusing structures that are not modeled. The content creator wants here Joel (cf. the application sketch of Section 7.2) to react angrily, when the player first speaks about a certain story person. The system reuses the peculiar facial expression that the author wants the system to display, in a similar situation, where the only difference is that player and Joel have become friends. But, according to the system's adaptation rules, an angry expression is not appropriate between friends, and thus removes the angry component of the face, but does not destroy the peculiarity of the intended expression.

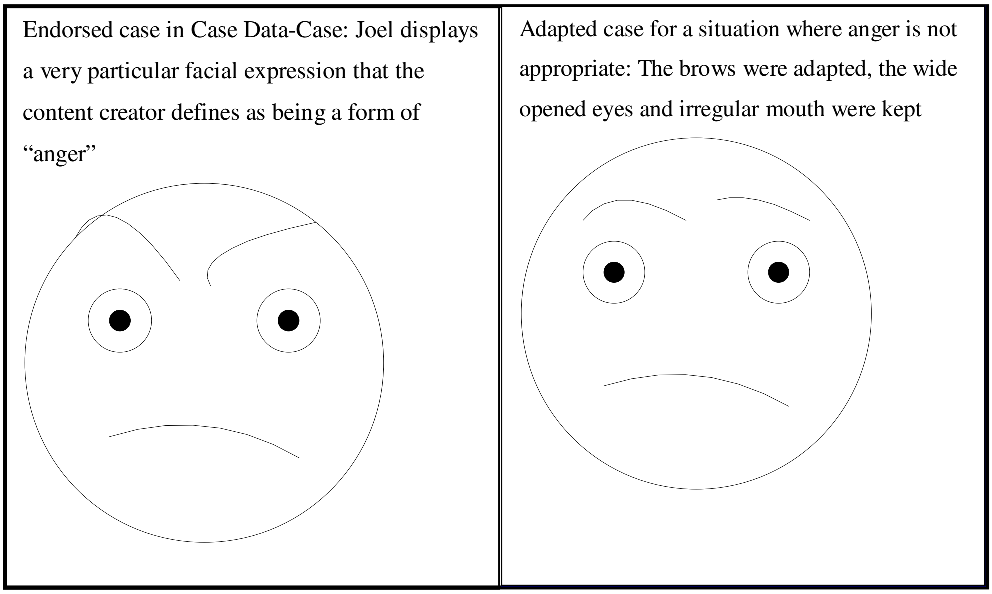


Figure 75 – The left image depicts the face of Joel, as defined by the content creator. Joel shall show this expression in a situation where it is asked first about Idabel and the player has not yet acquired Joel's friendship. The right side shows how the system can adapt the face by simply removing the "angriness" of the brows, employing generic adaptation rules. The peculiarity of the facial expression remains visible, without explicit modeling of expressions like "astonishment" or "doubt".

If natural language interaction is employed, integration with the CBR is a difficult task. The most promising approach consists in a combination of CBR and modeled solutions; the stored cases being the important exceptions to the regular behavior that is modeled. Figure 77 shows this structure. Following McKee's analysis, a story is composed of sequences of "beats", i.e. of important events that describe some change in story value; what is not important does not belong directly to the story.

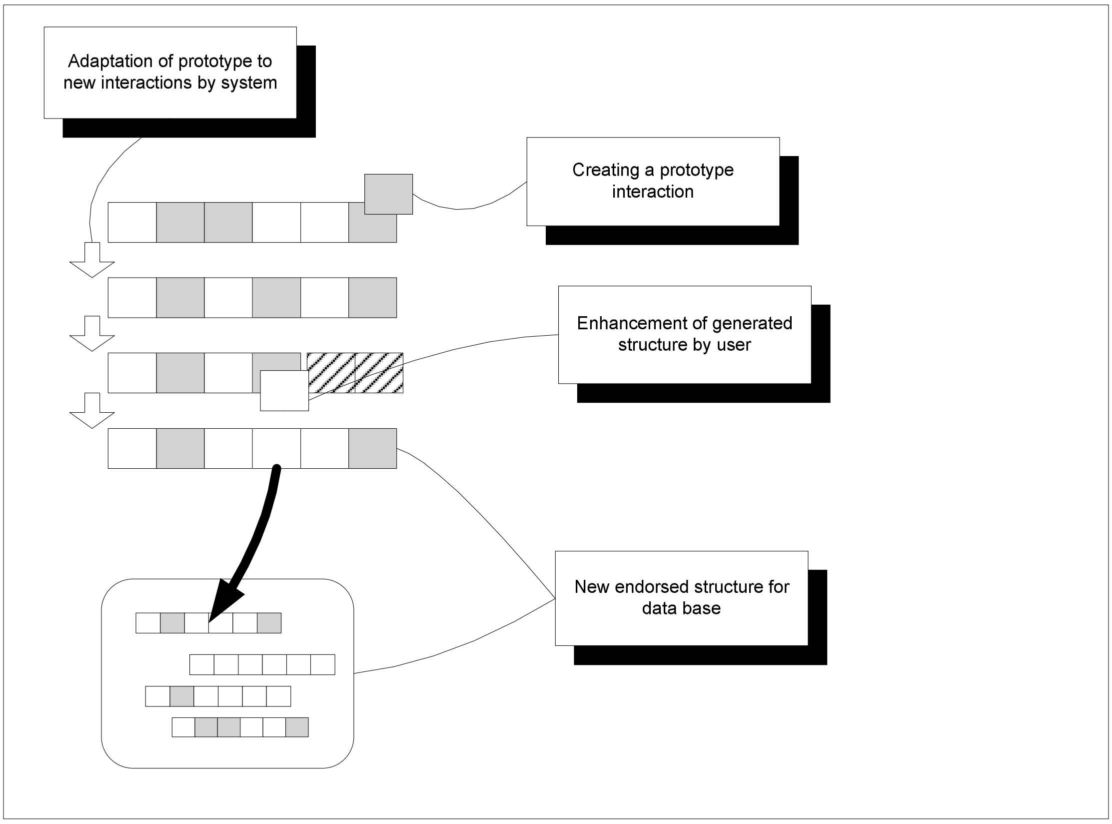


Figure 76 – The production of a typical story involves correcting and enhancing the system, until it produces the desired story without violating stories that are already in the data base of endorsed stories.

For example, chatting with the virtual character about the weather is normally not important for the story. Therefore, generic natural language and emotion modules can be employed. But speaking with it about his relationship to another important story person is important, and it may be essential to have the character (e.g. Joel, cf. above) react angrily when the theme comes to this person (e.g. Idabel). Thus, a set of important behavior patterns can overlay the base system of modeled behaviors. The important cases can be added iteratively, until a sufficient approximation of the desired behavior is achieved. A specific feature of an approach that employs typical interactions as basis is then, apart from the adaptation of the cases, the faculty of “bringing the interaction back” to the typical course and to the important beats; e.g., a virtual character should bring a conversation on the weather back to the more important relationship to other story characters.

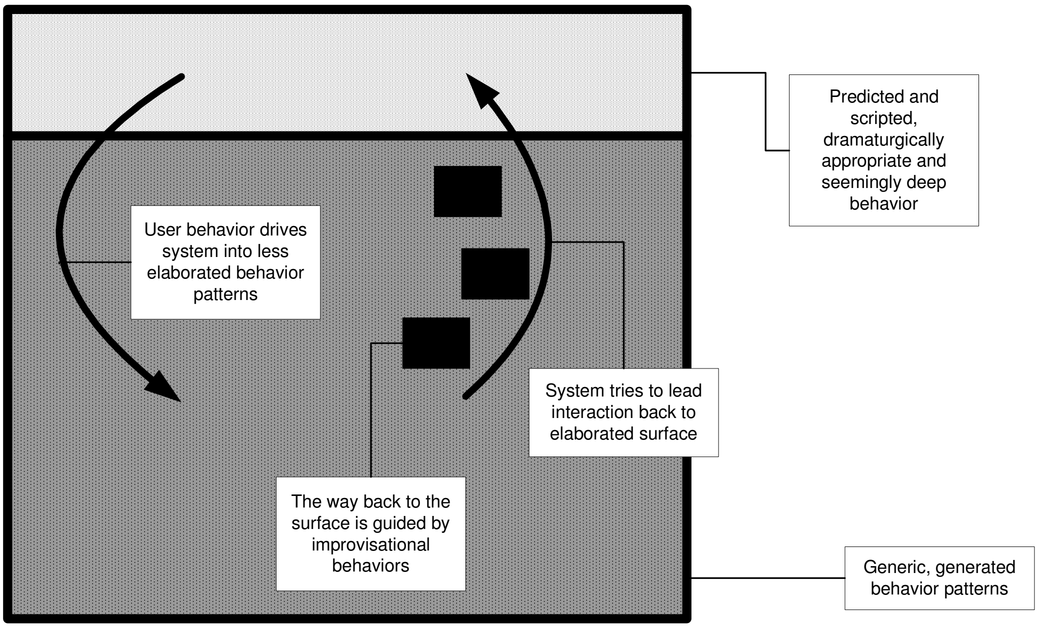


Figure 77 – The dramaturgically important interactions build normally only a thin surface, comparing to the whole body of possible interactions. The large part of the body can be generated by generic modules and algorithms, while the important surface requires, to a large extent, authoring efforts. The virtual actors must be able to bring a generic interaction back to the important dramatic layer. The surface layer can be approximated by adding CBR-cases to the system.

There is no need to restrict a CBR-case to a single instance of a story fragment. Departing from the experiences acquired with Cyranus on the usefulness of employing directed graphs to have control over precise reactions, it is sensible to allow for the use of this formalism also in Creactor. Thus, a content creator should be able to define a case also in terms of bifurcations. Then, this specific CBR-case would function like a composite state of the Cyranus-framework, the main difference being the possibility of adaptation, while the CBR-module represents a "Sequencing Engine" of the framework.

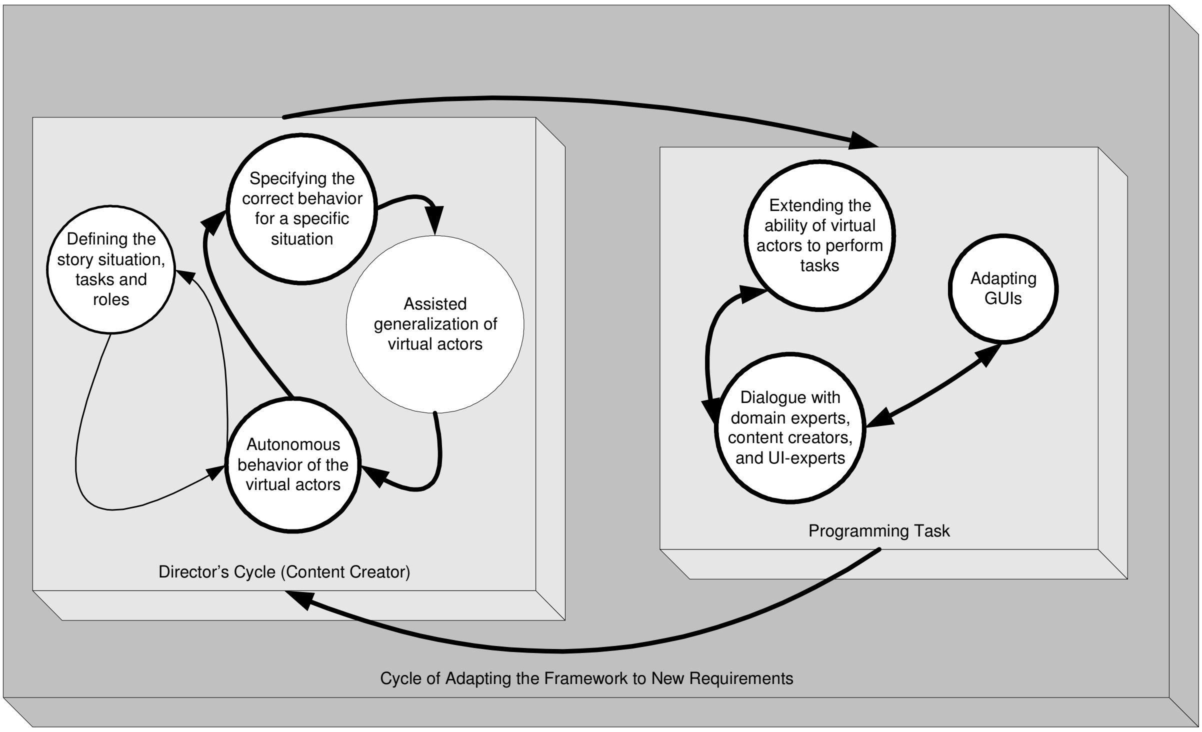


Figure 78 – The major cycle of the adapting the framework includes the minor cycle of the content creator approximating the desired behavior of the virtual actors, without changing the framework itself.

Figure 78 shows the resulting creation cycle of the approach, which involves computer scientists together with domain experts and content creators making deeper changes to the system in order to meet demands that emerged during the work of the content creator, and the minor cycle of the content creator who, through teaching, correcting, explaining and fine tuning, approximates the behavior that he has in mind, until it is sufficiently close to the intended result, or until the system is no longer capable of approximating the target, and entering into the major cycle becomes necessary. The work represented within the square labeled with "Programming Task" is not clearly a cycle, but implies rather a constant dialogue between the parties involved, and also joint work. An example of the production cycle of the content creator is depicted in Figure 76.

7.4 Summary of Creactor

The conception of Creactor, an authoring framework for concrete authoring of autonomous virtual characters, exemplifies once again the principle already employed in the creation of Cyranus, of devising right at the beginning of the production process a generic framework that offers interfaces and structures which facilitate experiments with content, authoring and technologies. Creactor could thus enable concrete authoring, especially when learning modules are integrated, in spite of the autonomy of the virtual characters.

8 Conclusion

This dissertation has described the conceptual foundations and the implementation of an authoring framework for creating experimental interactive, narrative applications. The Hybrid Visual Formalism developed within this work combines the advantages of complex directed graph notations with the flexibility and power of algorithmic control strategies, thus enabling a controlled, agile, iterative, collaborative creation process. With the help of this formalism, it is now possible to define the behaviour of interactive narration applications employing visually intuitive authoring methods.

Directed graphs can be used at parts and abstraction layers of the application that are only moderately complex, and they can be used for prototyping highly complex parts that are possibly not well-defined at the beginning of the production process. Thus, the main advantage of the Hybrid Visual Formalism over other known methods is the fact that it allows to employ directed graphs at parts and layers of the applications where this is appropriate, combining the directed graphs with other control methods within a single applications, and that it allows for the use of directed graphs in the prototyping phase of the application; in the later case, a smooth transition from the directed graph representation to different control strategies is possible, because the visual representation of the states can be reused.

The Cyranus authoring framework, a concretization of the hybrid formalism tailored to the requirements of experimental IN-research, has been presented. The developments of the hybrid formalism and of Cyranus are the responses to the "wicked" situation of interactive narrative research, where the design ideas of the novel applications cannot be fully specified beforehand or are susceptible to considerable changes during the creation process. They enable the practical implementation and experimentation steps towards exploitation of the chances that the field of "interactive narration" contains.

Several novel concepts for employing interactive narrative within different domains were presented in this dissertation, together with patterns of implementation that employ the Cyranus framework, thus demonstrating its appropriateness for the task.

The final section of this dissertation has provided a model for an authoring framework for partially autonomous characters, named “Creactor”. Creactor is the sketch of an authoring system for partially autonomous virtual characters built upon a “set of endorsed stories”, i.e. a set of stories or story parts that function as fact basis. Other modules can then be verified manually or automatically with the help of the fact basis of endorsed stories. The model of Creactor has shown that an intuitive and iterative authoring process is possible also for partially autonomous virtual characters, in particular if machine-learning algorithms such as Case-Based Reasoning are employed.

It is to be hoped that once some more interactive narrative applications have been built and have been acclaimed within and outside the research community for an incontestable experience design, other parts of the research endeavor for interactive narration will become clearer, as well as the specific technological and authorial requirements necessary for their concretisation. I believe that the continuing intermingling of practical development and theoretical clarification of the production process is among the most important further research and development issues in the field of interactive narrative.

In particular, the concepts of Creactor must be further developed and understood, i.e. concepts for authoring of partially autonomous virtual characters, as well as the integration of these concepts with the Cyranus framework, i.e. with a framework for control of correct sequences of elements. The result of the combination of both approaches should be a most powerful and yet intuitive, generic tool for a large variety of different interactive applications.

References

Aamodt, A.; Plaza, E. (1994): Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications, 7(1), March 1994.

Aizpurua, I.; Ortiz, A.; Oyarzun, D.; Arizkuren, I.; Andrés, A.; Posada, J.; Iurgel, I. (2004): Adaptation of Mesh Morphing Techniques for Avatars Used in Web Applications. In: Francisco J. Perales López; Bruce A. Draper (Eds.): Articulated Motion and Deformable Objects, Third International Workshop, AMDO 2004, Palma de Mallorca, Spain, September 22-24, 2004, Proceedings: Springer. Lecture Notes in Computer Science, p. 26–34.

André, E.; Klesen, M.; Gebhard, P.; Allen S.; Rist, Th. (2000): Exploiting Models of Personality and Emotions to Control the Behavior of Animated Interface Agents. In: Jeff Rickel (eds.), Proceedings of the workshop on "Achieving Human-Like Behavior in Interactive Animated Agents", in conjunction with the Fourth International Conference on Autonomous Agents, Barcelona, June (2000), p. 3-7.

Albee, E. (1962): Who's afraid of Virginia Woolf? 1st ed. New York: Antheneum.

Anton, A.; Carter, R.; Dagnino, A.; Dempster, J.; Siege, D. (2001): Deriving Goals from a Use-Case Based Requirements Specification. In: Requirements engineering, Nr. 1, Vol. 6, p. 63–73.

Aristotle (1962): Aristotle's Poetics. S. H. Butcher (Translator): Hill and Wang.

Aylett, R. (2005): FearNot! - An Experiment in Emergent Narrative. In: Lecture notes in computer science (LNCS), p. No. 3661 (2005),305-316.

Aylett, R.S; Louchart, S; Dias, J; Paiva, A; Vala, M; Woods, S. & Hall, L. (2006): Unscripted Narrative for affectively driven characters. IEEE Journal of Graphics and Animation. May/June 2006 (Vol. 26, No. 3) ISSN: 0272-1716. pp. 42-52.

Bartneck, C. (2002): Integrating the OCC Model of Emotions in Embodied Characters: Workshop on Virtual Conversational Characters: Applications, Methods, and Research Challenges, Melbourne.

Baylor, A.; Kim, Y. (2005): Simulating instructional roles through pedagogical agents. International Journal of Artificial Intelligence in Education 15, p. 1-18.

Bernsen, N.; Dybkjaer, L. (2004): Domain-Oriented Conversation with H.C. Andersen.: Proc. of the Workshop on Affective Dialogue Systems, Kloster Irsee, Germany, 2004, p. 142–153.

Bickmore, T. (2003): Relational agents: effecting change through human-computer relationships. PhD Thesis, MIT.

Bourg, D.; Seemann, G. (2004): AI for Game Developers. Creating Intelligent Behavior in Games. O'Reilly Media.

Capote, T. (1948): Other voices, other rooms. New York: Random House.

Cavazza, M.; Charles, F.; Mead, S. (2002): Sex, Lies, and Video Games: an Interactive Storytelling Prototype.: AAAI Spring Symposium on AI and Interactive Entertainment, Stanford University, California, USA, p. 13–17.

Cavazza, M.; Pizzi, D. (2006): Narratology for Interactive Storytelling: a Critical Introduction. 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment, Darmstadt, Germany, December 2006.

Crawford, Ch. (2005): Chris Crawford on interactive storytelling. Berkeley, Calif.: New Riders.

Clarke, A.; Mitchell, G. (2001): Film and the Development of Interactive Narrative. In: Virtual Storytelling. Using Virtual Reality Technologies for Storytelling : International Conference ICVS 2001 Avignon, France, September 27-28, 2001, Proceedings. Lecture notes in computer science, Nr. 2197, p. 81.

Conde, T.; Tambellini, W.; and Thalmann, D. (1993): Behavioral Animation of Autonomous Virtual Agents Helped By Reinforcement. Herausgegeben von T. Rist et al.: Springer (LNAI, 2792), p. 175–180.

Conde, T.; Thalmann, D. (2006): Learnable behavioral model for autonomous virtual agents: low-level learning: AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, p. 89–96.

Fairclough, C.; Cunningham, P. (2003): A Multiplayer Case Based Story Engine. In: Quasim H. Mehdi; Norman E. Gough; Stéphan Natkine (Eds.): 4th International Conference on Intelligent Games and Simulation (GAME-ON 2003), 19-21 November 2003, London, UK: EUROSIS, p. 41-51.

Fauconnier, G.; Turner, M. (2002): The Way We Think. Conceptual Blending and the Mind's Hidden Complexities. New York: Basic Books.

Gebhard, P.; Kipp, M.; Klesen, M.; Rist, Th. (2003): Authoring Scenes for Adaptive, Interactive Performances. In: Proc. of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'03), Melbourne.

Gervás, P.; Díaz-Agudo, B.; Peinado, F.; Hervás, R. (2005): Knowledge-Based Systems, Volume 18, Issues 4-5, August 2005, AI-2004, Cambridge, England, 13th-15th December 2004, p. 235-242.

Glassner, A. (2004): Interactive storytelling. Natick Mass.: A.K. Peters.

Göbel, S.; Schneider, O.; Iurgel, I.; Feix, A.; Knopfle, C.; Rettig, A. (2004): Virtual Human Storytelling and Computer Graphics for a Virtual Human Platform. In: Lecture notes in computer science (LNCS), Nr. 3105, p. 79–88.

Göbel, S.; Iurgel, I.; Rössler, M.; Hülsken, F.; Eckes, Ch. (2007): Design and Narrative Structure of the Virtual Human Scenarios. International Journal of Virtual Reality, 2007, 5(3):1-10. *To appear.*

Grasbon, D.; Braun, N. (2001): A Morphological Approach to Interactive Storytelling. In: Proceedings of on Artificial Intelligence and Interactive Entertainment, Cast'01, Living in Mixed Realities, Sankt Augustin, Germany, 2001, p. 337–340.

Gratch, J. (2005): Lessons from Emotions Psychology for the Desing of Lifelike Characters. In: Applied artificial intelligence : AAIan international journal, p. Vol. 19, No. 3/4 (2005), p. 215-234.

Grützmacher, B.; Wages, R.; Trogemann, G. (2003): "An Authoring System for Non-Linear VR Scenarios." hybridreality03: 9th International Conference on

Virtual Systems and Multimedia 2003. Montreal, Quebec, Canada, October, p. 15-17.

Gulz, A. (1995): Social enrichment by virtual characters – differential benefits. Journal of Computer Assisted Learning 21, p. 405-418.

Harel, D. (1988): On Visual Formalisms. In: Commun. ACM, Jg. 31, Nr. 5, p. 514–530.

Hassenzahl, M. (2006): User experience – a research agenda. In: Behavior & information technology : an international journal on the human aspects of computing, p. Vol. 25, No. 2 (2006), p. 91-97.

Hoffmann, A.; Iurgel, I.; Becker, F.: Interactive Drama and Learning Experiences (2005): M. Burmester, D. Gerhard, F. Fhissen (Eds.). Digital Game Based Learning. Proceedings of the 4th International Symposium for Information Design. 2nd of June 2005 at Stuttgart Media University, p. 149–157.

Hoffmann, A.; Göbel, S.; Schneider, O.; Iurgel, I. (2006): Storytelling-Based Edutainment Applications: Tan, Leo (Ed.) et al.: E-Learning and Virtual Science Centers. Hershey et al. : Information Science Publishing, 2005, p. 190–214.

Howard, D.; Mabley, E. (1993): The tools of screenwriting. 1st ed. New York: St. Martin's Press.

Iurgel, I. (2002a): Modellierung einer Gruppe virtueller Charaktere mit verschiedenen Persönlichkeitsmerkmalen. Master Thesis. Supervised by J. Encarancao, U. Spierling, M. Alexa. Darmstadt. TU Darmstadt, IGD.

Iurgel, I. (2002b): Emotional Interaction in a Hybrid Conversation Group: PRICAI-02. Workshop on Lifelike Animated Agents. Tokyo, Japan. JSPS.

Iurgel, I. (2003a): Virtual Actors in Interactivated Storytelling. In: Thomas Rist; Ruth Aylett; Daniel Ballin; Jeff Rickel (Eds.): Intelligent Agents, 4th International Workshop, IVA 2003, Kloster Irsee, Germany, September 15-17, 2003, Proceedings: Springer. Lecture Notes in Computer Science, p. 254–258.

Iurgel, I. (2003b): Experiencing Art on the Web with Virtual Companions: Proceddings of Museums and the Web 2003, 20-22 March 2003, Charlotte, USA.

Iurgel, I. (2003c): Automated Emotional Immersion in Interactive Storytelling: Technologies for Interactive Digital Storytelling and Entertainment. Proceedings : TIDSE 2003. Stuttgart : Fraunhofer IRB Verlag, 2003, pp. 403-407 (Computer Graphik Edition 09), p. 351–356.

Iurgel, I. (2003d): Technical Specification of the Storytelling Runtime Environment of Art-e-Fact project, Deliverable D 1.3, www.Art-e-Fact.org. Available also at ZGDV e.V.

Iurgel, I. (2004a): Narrative Dialogues for Educational Installations: Brna, Paul (Ed.): Narrative and Interactive Learning Environments. NILE 2004 Proceedings. p. 9–16.

Iurgel, I. (2004b): From Another Point of View: Art-e-Fact. In: Stefan Göbel; Ulrike Spierling; Anja Hoffmann; Ido Iurgel; Oliver Schneider; Johanna Dechau; Axel Feix (Eds.): Technologies for Interactive Digital Storytelling and Entertainment, Second International Conference, TIDSE 2004, Darmstadt, Germany, June 24-26, 2004, Proceedings: Springer. Lecture Notes in Computer Science, p. 26–35.

Iurgel, I.; Hoffmann, A.; Spierling, U. (2004): Wissensvermittlung durch interaktives Erzählen - die Plattform Art-e-Fact: Keil-Slawik, Reinhard (Ed.) et al.; Gesellschaft für Informatik (GI) Fachausschuss Mensch-Computer-Interaktion et al.: Mensch & Computer: Allgegenwärtige Interaktion. München : Oldenbourg, p. 147–156.

Iurgel, I.; Ziegler, M. (2005): Ask&Answer: An Educational Game Where It Pays to Endear Your Capricious Virtual Companion. In: Themis Panayiotopoulos; Jonathan Gratch; Ruth Aylett; Daniel Ballin; Patrick Olivier; Thomas Rist (Eds.): Intelligent Virtual Agents, 5th International Working Conference, IVA 2005, Kos, Greece, September 12-14, 2005, Proceedings: Springer. Lecture Notes in Computer Science, p. 15–24.

Iurgel, I. (2005a): Beneficial Dependencies: Design Principles for Narrative Games. In: Gérard Subsol (Ed.): Virtual Storytelling, Using Virtual Reality Technologies for Storytelling, Third International Conference, ICVS 2005, Strasbourg, France, November 30 - December 2, 2005, Proceedings: Springer. Lecture Notes in Computer Science, p. 209–218.

Iurgel, I. (2005b): Concepts for Employing Natural Language Processing for Entertainment Applications: In: Masuch, Maic (Ed.) et al.; Gesellschaft für Informatik (GI): Proceedings of the Computer Science and Magic 2005: GC Developer Conference Science Track. Leipzig, 2005.

Iurgel, I. (2005c): Final Management Report of Art-e-Fact project, www.Art-e-Fact.org. Available also at ZGDV e.V.

Iurgel, I. (2006a): Cyranus - An Authoring Tool for Interactive Edutainment Applications. In: Zhigeng Pan; Ruth Aylett; Holger Diener; Xiaogang Jin; Stefan Göbel; Li Li (Eds.): Technologies for E-Learning and Digital Entertainment, First International Conference, Edutainment 2006, Hangzhou, China, April 16-19, 2006, Proceedings: Springer. Lecture Notes in Computer Science, p. 577–580.

Iurgel, I., Hoffmann, A.; Linge, M.: (2006b): Final Report of the Project “Virtual Human with Social Intelligence”. Delievered to SAP AG in September 06. Available also at ZGDV e.V.

Iurgel, I. (2006c): Interim Technical Report of the Project “Virtual Human”, www.virtual-human.de, 8/2006. Available also at ZGDV e.V.

Iurgel, I.; Marcos, A. (2006): Engaging Virtual Agents: Interacção 2006 : Actas da 2a. Conferencia Nacional em Interacção Pessoa-Máquina. Lisboa : Grupo Portugues de Computacao Grafica (CPCG), p. 123–131.

Iurgel, I.; Marcos, A. (2007): Employing Personality-Rich Virtual Persons – New Tools Required: Computers and Graphics. Special Issue on Digital Arts. *To appear.*

Johnstone, K. (1999): Impro for storytellers. New York: Routledge/Theatre Arts Books.

Krämer, N., Iurgel., I.; Bente. G. (2005): Emotion and motivation in embodied conversational agents: Canamero, L. (ed.), Proceedings of the Symposium “Agents that Want and Like”, Artificial Intelligence and the Simulation of Behavior (AISB). Hatfield: SSAISB, p. 55–61.

Laurel, B. (1986): Toward the design of a computer-based interactive fantasy system: Ph.D. dissertation, The Ohio State University.

Lazarus, R. (1991): Emotion and adaptation. New York: Oxford University Press.

Leake, D. (1996): CBR in Context: The Present and Future. In Leake, D., editor, Case-Based Reasoning: Experiences, Lessons, and Future Directions. AAAI Press/MIT Press, p. 1-30.

Louchart, S. (2004): Narrative theory and emergent interactive narrative. In: International journal of continuing engineering education and life-long learning : IJCEELL, p. Vol. 14, No. 6 (2004), p. 506-518.

Machado, I. (2004): 1, 2, 3… Action! Directing Real Actors and Virtual Characters. In: Lecture notes in computer science (LNCS), Nr. 3105, p. 36–41.

Magerko, B. (2005): Story Representation and Interactive Drama. In: AIIDE Conference June 1-3, 2005, Marina Del Rey, CA, p. 87–92.

Mateas, M.; Sengers, P. (1999): Narrative Intelligence. In: Working Notes of the Narrative Intelligence Symposium, AAAI Fall Symposium Series.

Mateas, M.; Stern, A. (2000): Towards Integrating Plot and Character for Interactive Drama. In Working notes of the Social Intelligent Agents: The Human in the Loop Symposium. AAAI Fall Symposium Series. Menlo Park, CA: AAAI Press.

Mateas, M.; Stern, A. (2003): Façade: An Experiment in Building a Fully-Realized Interactive Drama: Game Developers Conference, Game Design track, March 2003.

Mateas, M.; Stern, A. (2004): Natural Language Understanding in Facade Surface - Text Processing: Lecture notes in computer science (LNCS) 3105), p. 3–13.

Mateas, M.; Stern, A. (2005a): Build It to Understand It: Ludology Meets Narratology in Game Design Space.: Digital Games Research Conference 2005, Changing Views: Worlds in Play, June 16-20, 2005, Vancouver, British Columbia, Canada.

Mateas, M.; Stern, A. (2005b): Structuring Content in the Façade Interactive Drama Architecture: Proceedings of Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2005), Marina del Rey, June, 2005.

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

Mateas, M.; Stern, A. (2006): Facade Architecture and Authorial Idioms for Believable Agents in Interactive Drama. In: Lecture notes in computer science (LNCS), Nr. 4133, p. 446–448.

McKee, R. (1997): Story. Substance, structure, style and the principles of screenwriting. 1st ed. New York: ReganBooks.

Moreno, J. (1947): The theatre of spontaneity. New York: Beacon House.

Mott, B. W.; Lester, J. C. 2006: U-director: a decision-theoretic narrative planning architecture for storytelling environments. In Proceedings of the Fifth international Joint Conference on Autonomous Agents and Multiagent Systems (Hakodate, Japan, May 08 - 12, 2006). AAMAS '06. ACM Press, New York, NY, 977-984.

Müller, W.; Spierling, U.; Alexa, M.; Iurgel,I. (2001): Design Issues for Conversational User Interfaces: Animating and Controlling 3D Faces. In: Nadia Magnenat-Thalmann; Daniel Thalmann (Eds.): Deformable Avatars, IFIP TC5/WG5.10 DEFORM'2000 Workshop, November 29-30, 2000, Geneva, Switzerland and AVATARS'2000 Workshop, November 30-December 1, 2000, Lausanne, Switzerland: Kluwer. IFIP Conference Proceedings, p. 115–125.

Murray, T. (1997): Expanding the knowledge acquisition bottleneck for intelligent tutoring systems. In: International Journal of Artificial Intelligence in Education, Jg. 8, Nr. 3-4, p. 222–232.

Murray, J. (1997): Hamlet on the holodeck. New York: Free Press.

Nelson, M. J.; Ashmore, C.; Mateas, M.: Authoring an Interactive Narrative with Declarative Optimization-Based Drama Management. Proceedings of the Second Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE-06). June 20–23, 2006, Marina del Rey, California, USA.

Nijholt, A. (2005): Conversational agents, humorous act construction, and social intelligence. In K. Dautenhahn (Ed.), Proceedings AISB 2005: Social Intelligence and Interaction in Animals, Robots and Agents. Symposium on Conversational Informatics for Supporting Social Intelligence and Interaction, 1–8. Hatfield, England: University of Hertfordshire.

Ortony, A.; Clore, G.; Collins, A. (1988): The cognitive structure of emotions. Cambridge England ;;New York: Cambridge University Press.

Paiva, A.; Dias, J.; Sobral, D.; Aylett, R.; Woods, S.; Hall, L.; Zoll, C. (2005): Learning By Feeling: Evoking Empathy With Synthetic Characters. In: Applied Artificial Intelligence, Jg. 19, Nr. 3-4, p. 235–266.

Piwek, P.; Krenn, B.; Schröder, M.; Grice, M.; Baumann, S.; Pirker, H. (2002): RRL: A Rich Representation Language for the Description of Agent Behavior in NECA. Proceedings of the AAMAS workshop on "Embodied conversational agents - let's specify and evaluate them!", Bologna, Italy.

Prensky, M. (2001): Digital game-based learning. New York: McGraw-Hill.

Prensky, M. (2006): Don't bother me Mom, I'm learning! 1st ed. St. Paul Minn.: Paragon House.

Propp, V. (1971): Morphology of the Folktale. (Publications of the American Folklore Society): University of Texas Press.

Rao, A.; George, M. (1991): Modeling rational agents within a BDI-architecture. In J. Allen, R. Fikes, and E. Sandewall, editors, Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann Publishers, San Mateo, CA.

Riedl, M. (2003): Character-Focused Narrative Generation for Execution in Virtual Worlds. In: Lecture notes in computer science (LNCS), p. No. 2897 (2003),47-56.

Riedl, M.; Stern, A.; Dini, D. (2006): Mixing Story and Simulation in Interactive Narrative: Proceedings of the 2nd Annual Conference on Artificial Intelligence and Interactive Digital Entertainment, Marina del Rey, California.

Riedl, Mark O.; Young, R. Michael (2006): From Linear Story Generation to Branching Story Graphs. In: IEEE Computer Graphics and Applications, Jg. 26, Nr. 3, S. 23–31.

Rittel, H.; Weber, M. (1973): Dilemmas in a General Theory of Planning. In: Policy Sciences 4, Elsevier Scientific Publishing, Amsterdam, p. 155–159.

Saucier, G.; Goldberg, L. (2003): The structure of personality attributes: M. Barrick & A. M. Ryan (Eds.), Personality and work. San Francisco: Jossey-Bass-Pfeiffer., p. 1–29.

Shoemake, K. (1985): Animating rotation with quaternion curves. International Conference on Computer Graphics and Interactive Techniques. Proceedings of the 12th annual conference on Computer graphics and interactive techniques, p. 245 – 254.

Seger, L. (1990): Creating Unforgettable Characters: Owl Books.

Sobral, D.; Machado, I.; Paiva, A. (2003): Managing Authorship in Plot Conduction. In: Olivier Balet; Gérard Subsol; Patrice Torguet (Eds.): Virtual Storytelling; Using Virtual Reality Technologies for Storytelling, Second International Conference, ICVS 2003, Toulouse, France, November 20-21, 2003, Proceedings: Springer. Lecture Notes in Computer Science, p. 57–64.

Spierling, U.; Grasbon, D., Braun, N.; Iurgel, I.: (2002): Setting the scene playing digital director in interactive storytelling and creation. In: Computers & graphics : C& G. An international journal of applications in computer graphics, p. Vol. 26, No. 1 (2002),31-44.

Spierling, U., Iurgel, I. (2003): Just Talking about Art - Creating Virtual Storytelling Experiences in Mixed Reality. In: Lecture notes in computer science (LNCS), p. No. 2897 (2003), p. 179-188.

Spierling, U.; Müller, W.; Vogel, R.; Iurgel, I. (2004): Digital Conversational Storytelling Elements for Teaching Mathematics in Primary School: Cantoni, Lorenzo (Ed.) u.a.; Association for the Advancement of Computing in Education (AACE): Proceedings of ED-Media 2004 : World Conference on Educational Multimedia, Hypermedia & Telecommunications. Norfolk, VA, USA : Association for the Advancement of Computing in Education (AACE), p. 4191–4196.

Spierling, U.; Iurgel, I. (2006): Pre-conference Demo Workshop Little Red Cap: The Authoring Process in Interactive Storytelling. In: Stefan Göbel; Rainer

Malkewitz; Ido Iurgel (Eds.): Technologies for Interactive Digital Storytelling and Entertainment, Third International Conference, TIDSE 2006, Darmstadt, Germany, December 4-6, 2006.: Springer. Lecture Notes in Computer Science, p. 193–194.

Struck, H.-G. (2005): Telling Stories Knowing Nothing: Tackling the Lack of Common Sense Knowledge in Story Generation Systems. In: Gérard Subsol (Ed.): Virtual Storytelling, Using Virtual Reality Technologies for Storytelling, Third International Conference, ICVS 2005, Strasbourg, France, November 30 - December 2, 2005, Proceedings: Springer. Lecture Notes in Computer Science, p. 189–198.

Swartjes, I.M.T.; Vromen, J.A.F.; Bloom, N.E. (2007): Narrative Inspiration: Using Case Based Problem Solving for Emergent Story Generation. In: Proceedings of the Fourth International Joint Workshop on Computational Creativity, 17-19 June 2007, London, United Kingdom. pp. 21-28. Goldsmiths, University of London.

Szilas, N.: (1999): "Interactive Drama on Computer: Beyond Linear Narrative". AAAI Fall Syposium on Narrative Intelligence. Tech. Rep. FS-99-01. p. 150-156.

Szilas, N. (2003): Authoring Highly Generative Interactive Drama. In: Lecture notes in computer science (LNCS), p. No. 2897 (2003),37-46.

Szilas, N. (2004): Stepping into the Interactive Drama. In: Lecture notes in computer science (LNCS), p. No. 3105 (2004), p.14-25.

Theune, M.; Slabbers, N.; Hielkema, F. (2007): The Narrator: NLG for digital storytelling. In: ENLG-07 11th European Workshop on Natural Language Generation, 17-20 June 2007, Germany. pp. 109-112. DFKI Document Series (D-07-01). DFKI (Deutsches Forschungszentrum für Künstliche Intelligenz GmbH).

Vala, M.; Paiva, A.; Gomes, M. (2003): Happy Characters Don't Feel Well in Sad Bodies! In: Thomas Rist; Ruth Aylett; Daniel Ballin; Jeff Rickel (Eds.): Intelligent Agents, 4th International Workshop, IVA 2003, Kloster Irsee, Germany, September 15-17, 2003, Proceedings: Springer. Lecture Notes in Computer Science, p. 72–79.

von der Beek, M. (1994): A comparison of Statechart variants. In Formal Techniques in Real-Time and Fault-Tolerant Systems, L. de Roever and J. Vytopil, Eds. Lecture Notes in Computer Science, vol. 863. Springer-Verlag, New York, p. 128–148.

Glossary

Activation Engine: Either the Sequencing Engine, or the Graph Engine.

BDI: Believe-Desire-Intention architecture for intelligent agents, cf. [RaoGeorge]

CBR: Case-Based Reasoning, a learning method based on analogy, cf. [AamodtPlaza]

Creactor: The model of an authoring environment for autonomous virtual characters.

Cyranus: The software environment for interactive narration, with all of its modules and formats, including the authoring tool, and the implementation of the hybrid formalism.

Graph Engine: The engine of the extended Statechart part of the framework, i.e. the engine that interprets transitions and activates states.

Holodeck: "In the Star Trek fictional universe, the holodeck is a simulated reality facility, generally on starships and starbases."
(Source: http://en.wikipedia.org/wiki/Holodeck, 03/15/06)

Hybrid Formalism: The extension of Harel's Satecharts, and its integration with Sequencing and Fallback Engines.

IN: Interactive Narrative

MMORPG: Massive Multiplayer Online Role-Playing Game

NLP: Natural Language Processing

OCC: The emotion theory of Ortony, Clore and Collins [OrtonyClore]

Sequencing Engine: The engine within the Hybrid Control Formalism that can activate a state directly, without traversing a transition.

TTS: (Synthetic) Text-to-Speech

Index of Figures

Akademischer Lebenslauf

Geboren am 01.06.67 in Sao Paulo/Brasilien.

02/74 – 12/75	Ginasio I.L. Peretz – Grundschule in Sao Paulo, Brasilien	
02/76 – 05/79	EEPG Aristides de Castro – Grund- und Mittelschule in Sao Paulo, Brasilien	
08/79 – 07/82	Gesamtschule der Stadt Mülheim a. d. Ruhr	
08/82 – 07/87	Karl-Ziegler-Gymansium, Mülheim a. d. Ruhr	Abschluss mit *Abitur*
10/87 – 09/88	Studium der Medizin an der U-GH Essen	
10/88 – 06/96	Studium der Philosophie, Sozialpsychologie und allgemeinen Sprachwissenschaft an der Ruhr-Universität Bochum	Abschluss mit dem Titel *Magister Artium*
01/91 – 01/95	Studentische Hilfskraft am Sprachwissenschaftlichen Institut der Ruhr-Universität Bochum	
02/93 – 05/95	Zusammenarbeit mit dem Labarq (Computergraphikabteilung der Architekturfakultät der Universität von Sao Paulo) beim Marketing von Informatik-Produkten	
07/96 – 09/97	Wissenschaftliche Hilfskraft am Philosophischen Institut der Ruhr-Universität Bochum	
10/97 – 09/98	Mitglied des Graduiertenkollegs „Phänomenologie und Hermeneutik“	
10/98 – 01/02	Studium der Informatik an der TU Darmstadt	Abschluss mit dem Titel *Dipl.-Inform.*
03/02 – 12/06	Wissenschaftlicher Mitarbeiter in der Abteilung Digital Storytelling am ZGDV e.V., Darmstadt	