Story Generators: Models and Approaches for the Generation of Literary Artefacts

Birte Lönneker *(birte.loenneker@uni-hamburg.de)*
Universität Hamburg
Jan Christoph Meister
(jan-c-meister@uni-hamburg.de)
Universität Hamburg
Pablo Gervás *(pgervas@sip.ucm.es)*
Universidad Complutense de Madrid
Federico Peinado *(fpeinado@fdi.ucm.es)*
Uni Madrid
Michael Mateas *(michaelm@cc.gatech.edu)*
Georgia Institute of Technology

This session is concerned with the automated creation of fiction or "literary artefacts" that might take the form of prose, poetry or drama. Special focus is placed upon those approaches that include the generation of narrative structures and therefore use some kind of story model. First attempts in automated story generation date back to the 1970s, with the implementation of Meehan's *TALE-SPIN* (1977) based on the achievement of character plans and Klein's automatic novel writer (1973/1979) that simulates the effects of generated events in the narrative universe. Currently, story generators enjoy a phase of revival, both as stand-alone systems or embedded components. Most of them make reference to an explicit model of narrative, but the approaches used are diverse: they range from story grammars in the generative vein to the conceptually inspired engagement-reflection cycle. Real-life applications include the generation of a set of plot plans for screen writers in a commercial entertainment environment, who could use the automatically created story pool as a source of inspiration, and the generation of new kinds of interactive dramas (video games).

The creation process of literary artefacts is of particular relevance to Literary and Humanities Computing. Not only does it provide methods of simulating and modelling narrative processes, but it identifies basic and combinatory elements of story and discourse. The definition of these elements can in turn help scholars involved in the analysis of narrative to produce annotation (mark-up) that might be re-used in the story generation models.

Research into Story Generators is by nature an interdisciplinary project and as such constitutes an exemplary case for Humanities Computing efforts that are of a more speculative kind, as opposed to application oriented approaches. While the humanities – and more particularly, narratology – are called upon to clarify and systematize basic concepts and theoretical models of narration, computer scientists and AI researchers try to translate these models into workable system architectures and processes. Accordingly, our session brings together one theoretical and two applied approaches to the generation of literary artefacts. The theoretical paper presents an attempt towards formulating an 'ideal' Story Generator Architecture based on a narratological model of story generation. The applied papers discuss the role of the story model in a stand-alone fairy tale generator (*ProtoPropp*), and the problem of story management in games, using *Façade* and other story (drama) management architectures as examples.

"Dream on": Designing the ideal Story Generator Algorithm

Birte Lönneker and Jan Christoph Meister

*Deep Blue*may have beaten Kasparov at chess —but whether computers will ever be able to generate well-formed and aesthetically pleasing narratives is still subject to dispute (Bringsjord/Ferrucci; Pérez y Pérez/Sharples). Most AI researchers have come to the conclusion that the generation of natural language narratives that are both domain independent and Turing test compliant is a 'killer application': it defines the outer limit of computational creativity.

The current paper reports on our research into Story Generator Algorithms (SGAs), that is, computational systems designed to generate natural language narratives. Though attempts at designing and implementing SGAs date back to the early 1970s they have not received a lot of attention in Humanities Computing (HC) circles. The relevance of these experimental and speculative approaches seemed rather limited in the light of practical HC desiderata such as the definition of mark-up conventions, document type definitions, and standards for digital resource building, to mention but a few. However, contrary to this pragmatist line of reasoning we would like to argue that SGAs in their abstract models make explicit some of the cardinal assumptions underlying our intuitive human models of narrative, which in turn have filtered down into the practice of humanities computing in whose object domain narratives play a dominant role. Our own research methodology is therefore in part empirical - we aim to survey and classify the types of SGAs developed thus far - and in part theoretical. The current paper focuses on one of the theoretically oriented tasks: the design of the architecture for a hypothetical 'ideal' SGA that would be able to emulate advanced, aesthetically validated human storytelling capability.

The system architecture of this ideal SGA is derived from advanced models of storytelling developed in narratology, i.e. the humanities methodology dedicated to the scientific study of narratives. Figure 1 shows the architecture with its four domains:

1. the goal domain, in which several kinds of story-telling goals are offered, for a random or user selection;
2. the knowledge domain, in which static knowledge is represented in concepts and their interrelations in an ontology, to which language-specific lexica as well as a case-base of previous and system-generated stories are related;
3. the *histoire* domain, containing three modules concerned with the question: "what happens?", or with the production of the content of the story;
4. the *discours* domain, combining two modules that aim to answer the question: how is the content presented?

The two system complexes labelled *histoire domain* and *discours domain* mirror the two main 'levels' of narratological description introduced by structuralist scholars, *histoire* and *discours*(Todorov). However, the 'level'-metaphor used by narratologists —a residue of the structuralist 'deep layer' vs. 'surface layer' dichotomy – misleadingly implies a generative hierarchy which is at the same time uni-linear and strictly bottom-up, starting at the *histoire* level. We prefer instead to use the non-hierarchical 'domain' metaphor because it is better suited to accommodate backtracking procedures. Those procedures are necessary during the generation of the final product that should eventually reflect the intertwined results of constrained operations on knowledge pertaining to both *histoire* and *discours*; therefore, the backtracking is possibly recursive and iterative within and between both domains. With respect to implemented models for the generation of natural language artefacts, this view is more in line with the engagement-reflection model of the story generator *MEXICA*(Pérez y Pérez/Sharples) than with those of the story generator *BRUTUS*(Bringsford/Ferrucci) or of generators for technical texts (cf. Reiter), all of which basically use a unidirectional pipeline model. In the two remaining sections, we will briefly and exemplarily present key elements of our model.

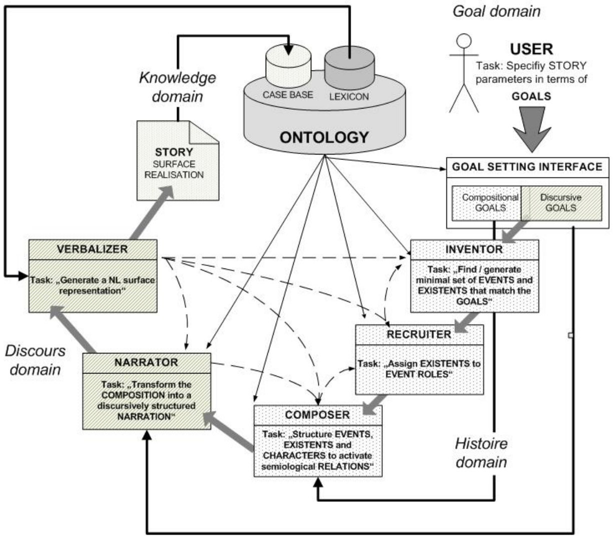

Goal domain
USER
Task: Specifiy STORY parameters in terms of GOALS
Knowledge domain
CASE BASE
LEXICON
ONTOLOGY
STORY SURFACE REALISATION
GOAL SETTING INTERFACE
Compositional GOALS
Discursive GOALS
VERBALIZER
Task: „Generate a NL surface representation"
INVENTOR
Task: „Find / generate minimal set of EVENTS and EXISTENTS that match the GOALS"
RECRUITER
Task: „Assign EXISTENTS to EVENT ROLES"
Discours domain
NARRATOR
Task: „Transform the COMPOSITION into a discursively structured NARRATION"
COMPOSER
Task: „Structure EVENTS, EXISTENTS and CHARACTERS to activate semiological RELATIONS"
Histoire domain

Figure 1: Architecture of an ideal SGA

The histoire domain

According to a minimal consensus definition, the *histoire*(story) is a chronological sequence of narrated events, together with their participants. Some narratologists have identified actions as the constitutive elements of stories, where an action is a special type of event, intentionally caused by at least one of the participants. Often it is also claimed that in order to form a story, the events need to be causally related (cf. Rimmon-Kenan16–17). However, narratology and related approaches have identified further types of relations between elements in the story domain that might as well contribute to storiness. For example, our previous formulation of a computational model of 'episode'(Meister) has shown that the episode is an interpretive construct of several events (four in that model) based on the activation of various semiological (semantic) relations, including contradictories.

In the system architecture shown above, events are represented as classes or frames and have at least the following slots (properties): A *follows* slot, the filler (value) of which points to the previous event; the slots *parallels* and *causes*, pointing to the respective events; the slots *involvesExistent* and *causedByExistent*, pointing to the respective participants (modelled as existents); the slot *changesStateInto*, which as a filler has an attribute-value-pair of one of the involved existents (the effect of the event is the resulting state of the affected existent); and, finally, an *isIntentional*slot to indicate the intentionality of the event. Some of the fillers are mandatory, others not; some slots can host a list of fillers. Existents like characters and objects (cf. Chatman) are also modelled as concepts with slots (attributes) and fillers (values).

The model defines further narratological notions in the *histoire* domain in terms of events and their relatedness; in particular, *story schema*, *closed event sequence* and *story*(cf. Forster93-94) can be defined as follows:

1. A story schema applies to a group of stories whose *histoire* is similar, for example fairy tales, legends, or detective stories. It constitutes a predefined closed set of existent classes and event classes together with their relations. The *ProtoPropp* generator presented in another paper of this session constitutes an example for a story schema based approach.
2. A closed event sequence is a temporal succession of events with a constant (sub)set of participants and a start event and end event. The temporal succession can be obtained from the fillers in the *follows* slots of the events.
3. Instead of explicitly defining what a story is, the model allows a comparison of different event sequences with respect to their storiness which is measured in terms of a) the numerical ratio between all events and action events in the considered event sequence; b) the numerical ratio between the events and the causal relations in the event sequence.

The discours domain

The *discours* domain should take into account several *discours* parameters, or aspects of text description defined in structuralist narratology. Such *discours* parameters include for example the **Order** of the presented events (which might differ from the order in which they actually occur in the story) and the **Frequency**with which the same or similar event(s) are presented (Genette), or the **Mediation-of-relatedness**, to name but a few. Currently, we work with a set of twelve *discours* parameters.

Every parameter has a list of subconcepts, which represent the actual phenomena they subsume. For example, the Order parameter has the subclasses *anachrony* and *synchrony*, with anachrony comprising the phenomena *flashback*, *flashforward*, and *achrony*. One of the aims of the project is to identify a standard phenomenon (default) in every parameter class. For example, in unmarked narrative texts, events are most likely to be presented in their chronological order (Order-subclass: *synchrony*) and motivation relations such as 'causality' are most likely to be left implicit (Mediation-of-relatedness subclass: *implicit*).

Discours parameters operate on, or apply to, specific types of elements belonging to the *histoire* domain. For example, the Order parameter operates on events and event sequences. Furthermore, each *discours* parameter supplies a modification rule that states in which way it affects the story representation during the preparation of the *discours*. Thus the *flashforward* parameter states that the affected story element (event) be moved one element (event) back in the sequence of presented events, i.e. towards the start of the story.

Future work

Future work will include the representation of the aesthetical or communicative effect of each of the *discours* parameters and the study of application restrictions of these parameters and combinations of them. It is important to note that those effects and restrictions are not absolute, but depend on the types of events and existents and their relations used in the *histoire* domain. Therefore, the 'communication' between operations performed in both domains, described as backtracking above, will be necessary in the generation process.

While the implementation of the hypothetical 'ideal' SGA outlined in the above may seem practically impossible, it proves to be a dream that raises fundamental questions — if nothing else by challenging our HC methodologies which, by and large, have hitherto concentrated on managing static humanist data, yet shied away from tackling the conceptual threshold of dynamism and recursivity inherent in most semantic artefacts.

Bibliography

Bringsjord, Selmer, and David A. Ferrucci. *Artificial Intelligence and Literary Creativity. Inside the Mind of BRUTUS, a Storytelling Machine.* Mahwah: Lawrence Erlbaum, 1999.

Chatman, Seymour Benjamin. *Story and discourse.* Ithaca: Cornell University Press, 1978.

Forster, Edward Morgan. *Aspects of the Novel.* London: Arnold, 1974.

Genette, Gérard. *Narrative Discourse.* Ithaca: Cornell University Press, 1980.

Meister, Jan Christoph. *Computing Action.* New York/Berlin: de Gruyter, 2003.

Reiter, Ehud. "Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible?" *Proceedings of the Seventh International Workshop on Natural Language Generation (INLGW-1994).* Kennebunkport, Maine, USA, 1994. 163–170.

Rimmon-Kenan, Shlomith. *Narrative Fiction.* Contemporary Poetics. London/New York: Routledge, 1983.

Todorov, Tzvetan. "Les catégories du récit littérarie." *Communications* 8 (1966): 125–151.

A generative and case-based implementation of Proppian morphology

Federico Peinado and Pablo Gervás

Automatic construction of story plots has always been a longed-for utopian dream in the entertainment industry, specially in the more commercial genres that are fueled by a large number of story plots with only a medium threshold on plot quality, such as TV series or video games. Although few professionals would contemplate full automation of the creative processes involved in plot writing, many would certainly welcome a fast prototyping tool that could produce a large number of acceptable plots involving a given set of initial circumstances or restrictions on the kind of characters that should be involved. Such a collection of plots might provide inspiration, initiate new ideas, or possibly even include a few plot sketches worthy of revision. Subsequent selection and revision of these plot sketches by professional screen writers could produce revised, fully human-authored valid plots. By making such a collection of tentative plots available to company screen writers, a smaller number of writers might be able to provide the material needed to keep the technical teams in work.

*ProtoPropp*is a software application that can generate new stories. The domain of the application is the Russian Folk Tale, and we have used Vladimir Propp's formalization of the structure of such tales (Propp) as a guideline for the generation process.

The particular method employed for plot generation is case-based reasoning (Riesbeck & Schank). Case-based reasoning (CBR) is an AI method that relies on reusing solutions to problems solved in the past to solve problems in the present. The concept of case identifies a pair formed by a problem and its solution. The main idea is to store past problems and their associated solutions as cases, in what is called a *case base*. Whenever a new problem has to be solved, the case base is searched for similar problems faced in the past (retrieval step), and the solution of the best matching past case is adapted to the new problem (adaptation step), to account for differences between the new problem and the old one. The full CBR cycle involves two additional steps (revision and update) which provide the means for enriching the case base with the results of solving new problems using this method, but those steps are not relevant to the current endeavour. Traditional CBR is very useful in domains where the information to be handled is too complex to model explicitly, and in which there is an easily accessible store of previously solved problems and their solutions. The ProtoPropp application considers a plot generation problem in terms of Proppian functions, and a solution to that problem in terms of the assignment of a conceptual representation of the plot of a story. This involves transcribing existing folk tales into conceptual representations of their contents and associating them with elements of Proppian morphology. This is done by resorting to a formalized knowledge base of concepts, organised into a taxonomy, which explicitly includes the relations between them. Such a knowledge base, following current terminology in AI, is referred to as an *ontology* (Gruber 1994). The use of explicit conceptual knowledge to guide the CBR process characterises *Knowledge Intensive* CBR (Díaz-Agudo & González-Calero 2003).

In this project we propose a Knowledge Intensive Case-Based Reasoning (KI-CBR) approach to the problem of generating story plots from a case base of existing stories analyzed in terms of Proppian functions. A case-based reasoning process is defined to generate plots from a user query, with two important phases: retrieval of old stories, and adaptation to build a new one. The user query specifies an initial setting for the story, and the ontology is used to measure during the generation process the semantic distance between concepts specified by the user and those that appear in the texts.

ProppianOnto: the formalized knowledge

ProppianOnto is the name of the ontology developed to implement the formalization of Propp's theory (see Figure 1). It is built using OWL description logic according to current standards (Bechhofer et al. 2004). It includes concepts like PlotCase, Character or ProppianFunction that our system needs to reason about stories.

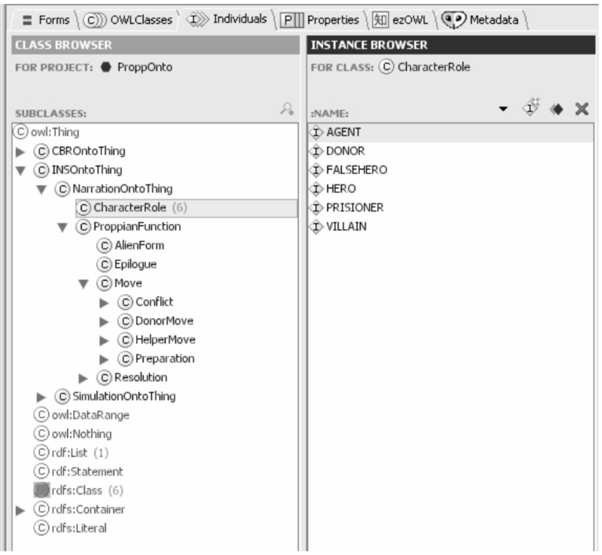


Figure 1: Implementation of ProppianOnto

The cases are plots taken from the Alexander Afanasiev's collection of Russian Fairy Tales. According to our interpretation of Propp, a plot case can have structures of two types.

PlotCase	PlotCase
1..n Character	1..n Character
1..n MoveCase	1 AlienForm
1 Resolution	0..1 Epilogue
0..1 Epilogue	

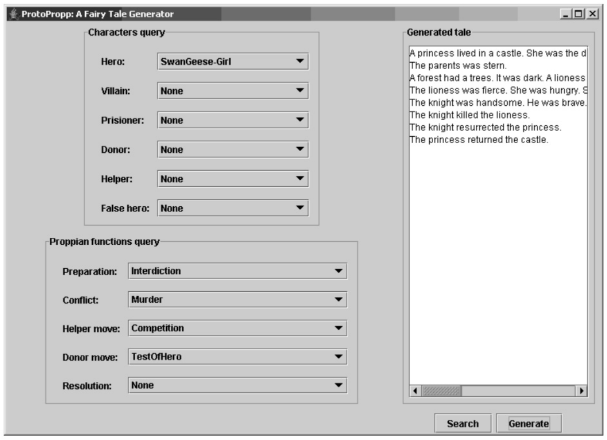
The elements in these structures can themselves have structure. The structure of a move case and a Proppian function are shown below.

MoveCase	ProppianFunction
1..n Character	1..n Character
1..n ProppianFunction	1..n Event
	1..n Place
	1..n Item
	before: 0..1 ProppianFunction
	after: 0..1 ProppianFunction

The structure of resolutions, epilogues and alien forms are similar to the move case.

ProtoPropp: the generation process

The generation of a new tale starts with a query presented by the user which represents the constraints that the desired tale should fulfil. The query is composed by filling in a form created which has six boxes for the main character roles (hero, villain, prisoner, donor, helper and false hero) and five boxes for the main Proppian functions (preparation, conflict, donor move, helper move and resolution), as it can be seen in Figure 2.



Hero: SwanGeese-Girl
Villain: None
Prisioner: None
Donor: None
Helper: None
False hero: None

Preparation: Interdiction
Conflict: Murder
Helper move: Competition
Donor move: TestOfHero
Resolution: None

```
A princess lived in a castle. She was the d
The parents was stern.
A forest had a trees. It was dark. A lioness
The lioness was fierce. She was hungry. S
The knight was handsome. He was brave.
The knight killed the lioness.
The knight resurrected the princess.
The princess returned the castle.
```

Figure 2: Author interface of ProtoPropp

Based on this query, the system attempts to find the most similar tales in the case base.

Because of the hierarchical structure of the conceptual taxonomy of the ontology, the given constraints can be fulfilled either in terms of strict similarity — the requested concept occurs in the tale — or in terms of conceptual similarity - either a specialization or an abstraction of the requested concept occurs in the tale.

1. *There is (at least) one retrieved tale with all the constraints in the query.* The system gives the option of adding more constraints.
2. *There are no retrieved tales fulfilling all the given constraints but at least one retrieved tale fulfils some of them.* The system selects the tale fulfilling the highest number of stated constraints. A second retrieval process is initiated, searching for a *different tale* satisfying the failed constraints. In this second process, the system can relax the similarity to ensure that some tale is found. That tale is mixed with the first retrieved tale to create a new one, adding or changing restrictions to the first tale, depending on the inter-relations between Proppian functions and other logical considerations, and generating from scratch the elements that the system was not able to find in the case base.
3. *There are no retrieved tales satisfying any of the query restrictions.* The system offers the possibility of adding more restrictions or changing the current restrictions to create a new story.

Whenever more restrictions are needed, the system always has the default option of filling them in randomly.

The resulting tale is obtained in an 'abstract form', as a conceptual representation of the ingredients that make it up. The last step is to use a natural language generator - cFROGS (García-Ibánez) - to convert this conceptual representation to a simple text that can be easily read by the user.

Conclusions and future work

This system shows how narrative structure theories can be implemented in a computational generator of stories. It is a long road, but every step we take will help us to learn more about the way stories are built. The current version of the prototype is restricted to single move plot cases. Complex stories made of more than one move case need more control over the dependencies between structural elements, and their construction should be directed by appropriate inference over the concepts represented explicitly within the system. For this strategy to succeed, the existing knowledge base must be extended to include any narratological concepts that may play a role in the adequate resolution of any conflicts arising from the existence of complex dependencies between the structural elements of a plot.

Bibliography

Bechhofer, S., et al. *OWL Web Ontology Language Reference.* W3C, 2004. Accessed 2005-04-06. <http://www.w3.org/TR/2004/REC-owl-ref-20040210/>

Díaz-Agudo, B., and P.A. González-Calero. "Knowledge Intensive CBR through Ontologies." *Expert Update* 6.1 (2003): 44–54.

García Ibáñez, C., et al. "Una arquitectura software para el desarrollo de aplicaciones de generación de lenguaje natural." *Procesamiento de Lenguaje Natural* 33 (2004): 111–118.

Gruber, T.R. "Towards Principles for the Design of Ontologies Used for Knowledge Sharing." *Formal Ontology in Conceptual Analysis and Knowledge Representation.* Ed. N. Guarino and R. Poli. n.p.: Kluwer, 1994. n. pag..

Propp, V. *Morphology of the Folktale.* Austin and London: University of Texas Press, 1968.

Riesbeck, C.K., and R.C. Schank. *Inside Case-Based Reasoning.* Evanston, Illinois: Lawrence Erlbaum, 1989.

Beyond Story Graphs: Story Management in Game Worlds

Michael Mateas

Bringing truly interactive story structures to computer games is hotly debated topic within the worlds of computer game design and academic game studies. For some designers and theorists, interactive story worlds are a holy grail of game design (e.g. Murray, Crawford), while for others narrative is antithetical to interactive experiences, destroying the high-agency, procedural potential of games (e.g. Eskelinen, Frasca). The heart of the tension between games and narrative lies in player agency. A player is said to have agency when she can form intentions with respect to the experience, take action with respect to those intentions, and interpret responses in terms of the action and intentions. Those who argue against narrative games point to the predetermined or predestined nature of narrative; strong narrative structures have complex sequences of cause and effect, complex character relationships and sequences of character interactions. Since player interaction can at any moment disrupt this narrative structure, the only way to maintain the structure is to remove or severely limit the player's ability to effect the structure. This eliminates so-called 'global' agency, forcing the player down a predetermined path. Thus ludologists argue that if narrative must inevitably mean a diminishment in player agency, it should not be used in game design.

Contemporary games do seem to support the ludologist position. In all contemporary story-based games the story structure is completely fixed, or has an extremely simple branching structure. The player has local agency, that is, can move around the environment and interact with objects and non-player characters, but the narrative structure is a linear sequence of cut scenes unlocked during the gameplay. In order to provide global narrative agency, computational and design methods must be devised that can incorporate player interaction into large scale story structures. This problem can be best understood by contrasting it with story graphs.

The standard 'best practice' in interactive narrative is the story graph, where each node represents a story event and each arc represents player actions. In a story graph the author has manually unwound all possible paths through the narrative space. However, the manual authoring overhead of story graphs means that, in practice, they tend to have a very small branching factor (quasi-linear) and a small number of nodes (limited story-level variation), leading precisely to the lack of global (story) agency evident in the contemporary game scene.

Story management

The alternative to the story graph is story management. The first story manager was proposed by Brenda Laurel in her thesis on interactive drama (Laurel), and further developed by AI research groups exploring interactive narrative. A story manager replaces the graph structure with a policy for story event selection. The author still creates the nodes of the story graph, where nodes represent story events such as scenes or individual character actions (depending on the granularity of global agency). However, rather than manually linking the nodes, that author instead creates a selection policy for story events; story events are activated as a function of the history of the story so far and the actions performed by the player. The story policy implicitly defines a story graph; theoretically, one can imagine unrolling the policy into a graph by recording the story function's response to all possible inputs (story histories + player action). The whole point of the story management approach, however, is to keep the graph implicit. By implicitly specifying graphs via a story policy, authors can create interactive stories that would be impractical to explicitly specify as graphs, and can thus create experiences with rich global agency.

In order to define a story policy, the author must specify:

- A representation of the desired story. In order for the story policy to select the next story event, it will need some model of the desired story (what a good story looks like within the domain) so as to decide what direction the story should move in given the story history plus player actions.
- A collection of story events. The story events may correspond to discrete units of storyness, such as scenes or dramatic beats, or may be more abstract story moves that manipulate the world in such a way as to make a desirable story happen in the future.
- A function that, given a model of the desired story, the story history, and the player actions, selects a story event.
- When story event selection happens. In general, a game world presents the player with a continuous, real-time experience while story guidance only happens at discrete

points. This presents the design problem of deciding when guidance should happen.

Example story managers

In this paper I survey three approaches to story management, the beat-based drama manager of the interactive drama Façade (Mateas & Stern 2003), Magerko and Laird's IDA (2003), and the search based drama manager (SBDM) first defined by Bates and Weyhrauch (1992; 1997), describing the different design decisions made by each approach with respect to the four design questions above.

In the *Façade*drama manager the story events are inspired by dramatic beats (McKee), the smallest units of dramatic value change. The desired story is modeled by one or more story value arcs (in *Façade*, the tension story value), and by declarative knowledge represented on each beat. This declarative knowledge includes:

- one or more preconditions, tests over facts pertaining to the episodic memory of the story-so-far that must be true for the beat to be potentially selectable;
- one or more priority tests that, given a satisfied precondition, boost the importance of a beat being selected;
- one or more weight tests that, given a satisfied precondition and highest-priority, boost the probability of a beat being selected;
- one or more effects that describe how the beat, assuming it is successfully executed in the world, will change the story values.

This knowledge, plus the desired story value arc(s), is used to compute a probability distribution over possible next beats; beats are selected by drawing from this changing distribution. When a beat is selected it activates a collection of behaviors that support the autonomous characters in carrying out the beat. These character-specific behaviors, which model the intentional structure of the characters, are written so as incorporate the player's moment-by-moment activity into the performance of the beat. If the player's activity deviates too far from the context assumed by the beat, the beat is aborted and a new one selected. Beat selection occurs on beat success (the beat successfully accomplishes the drama value change) or failure (the player's activity violates the beat context).

In SBDM, a player's concrete experience in the world is captured by a sequence of Player Moves, abstract plot points that a player's activity can cause to happen. A single Player Move may encapsulate 5 or 10 minutes of concrete player activity in the world - moving around, picking up objects, interacting with characters and so forth. When the concrete activity accomplishes a plot point, then a Player Move is recognized. A SBDM has a set of System Moves available that can materially alter the world (e.g. move objects around, change goals in characters' heads, etc.) in such a way as to encourage or obviate a Player Move. System Moves give the SBDM a way to warp the world around the player so as to make certain Player Moves more or less likely. Besides the System Moves, the author also provides the SBDM with a story-specific evaluation function that, given a complete sequence of Player and System Moves, returns a number indicating the "goodness" of the story. Whenever the drama manager recognizes a Player Move occurring in the world, it projects all possible future histories of Player and System moves, evaluates the resulting total histories with the evaluation function, and backs these evaluations up the search tree (in a manner similar to game-tree search) to decide which system move to make next that is most likely to cause a good total story to happen.

The future of story management

Story management virtualizes the links of a story graph; while the nodes of the graph must be authored, the possible paths through the graph remain implicit. The future of story management is to virtualize the nodes of the story graph as well; the nodes (story events) will be dynamically constructed as needed. The challenge will be to adapt algorithmic story generators to incorporate interaction. All artificial intelligence-based models of story generation, including story grammars, character modeling, and author modeling, assume that all elements of the story are under the complete control of the generator. In interactive narrative, however, the player can perform actions at any time that may compromise the current causal structures established by the generator. In the context of story management, generation must be able to dynamically adapt to player action.

Bibliography

Bates, J. "Virtual Reality, Art, and Entertainment." *Presence: The Journal of Teleoperators and Virtual Environments* 1.1 (1992): 133–138.

Cavazza, M., F. Charles, and S.J. Mead. "Sex, Lies and Videogames: an Interactive Storytelling Prototype." *Proceedings of the AAAI 2002 Symposium on Artificial Intelligence and Interactive Entertainment.* Ed. K. Forbus and M.S. El-Nasr. 2002. 13–17.

Crawford, C. *Chris Crawford on Interactive Storytelling.* n.p.: New Riders, 2004.

Eskelinen, M. "Towards Computer Game Studies." *Proceedings of SIGGRAPH 2001, Art Gallery, Art and Culture Papers.* Ed. K. Forbus and M.S. El-Nasr. 2001. 83–87.

Frasca, G. "Ludologists love stories too: notes from a debate that never took place." *Proceedings of Level Up 2003, Utrecht,* *The Netherlands.* 2003. Accessed 2005-04-06. <http://ludology.org/articles/Frasca_LevelUp2003.pdf>

Lamstein, A., and M. Mateas. "Search Based Drama Management." *Working Notes of the AAAI Workshop Challenges in Game AI, AAAI.* 2004.

Laurel, B. *Towards the Design of a Computer-Based Interactive Fantasy System.* Ph.D. Dissertation., The Ohio State University, 1986.

Magerko, B., and J. Laifo. "Building an Interactive Drama Architecture with a High Degree of Interactivity." Paper delivered at the 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment (= TIDSE '03), Darmstadt, Germany, March 2003. 2003.

Mateas, M., and A. Stern. "A Behavior Language for Story-Based Believable Agents." *Working notes of the Artificial Intelligence and Interactive Entertainment Symposium, AAAI Spring Symposium Series.* AAAI Press, 2002.

Mateas, M., and A. Stern. "Integrating plot, character and natural language processing in the interactive drama Façade." Paper delivered at the 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment (= TIDSE '03), Darmstadt, Germany, March 2003. 2003.

Mateas, M. "An Oz-Centric Review of Interactive Drama and Believable Agents." *AI Today: Recent Trends and Developments. (= LNAI 1600).* Ed. M. Wooldridge and M. Veloso. Berlin, New York: Springer, 1999. n. pag..

McKee, R. *Story: Substance, Structure, Style, and the Principles of Screenwriting.* New York, NY: HarperCollins, 1997.

Murray, J. *Hamlet on the Holodeck.* Cambridge, MA: MIT Press, 1998.

Pérez y Pérez, Rafael, and Mike Sharples. "Three computer-based models of storytelling: BRUTUS, MINSTREL and MEXICA." *Knowledge-Based Systems* 17 (2004): 15-29.

Weyhrauch, P. *Guiding Interactive Drama.* Ph.D. Dissertation, Tech report CMU-CS-97-109, Carnegie Mellon University, 1997.