Computer Modelling of Storytelling and Creativity

Manfred Stede
Technische Universität Berlin
Projektgruppe KIT
Sekretariat FR 6–10
Franklinstr. 28/29
10587 Berlin
Germany
email: stede@cs.tu-berlin.de

1 Introduction

Soon after the advent of computers in the 1940s, the idea was born to have them process not merely numbers but also human language. The research field of *machine translation* was the first to attract interest, later followed other kinds of applications, among them the idea to model the processes of storytelling and the underlying creativity. Computer applications presuppose formal theories, and stories indeed seemed to be amenable to formalization, as they were often seen as instances of a small set of schematic templates. Earlier this century, for instance, George Polti had analyzed a wide range of classic plotlines in literature and proposed an inventory of thirty-six basic "dramatic situations", which included conceptions like conflict with a god, mistaken jealousy, faulty judgement, or loss of loved ones [14].

In the spirit of such schematic approaches, the concept of "story grammar" originated in the 1970s. Even though story grammars were first developed from the perspective of understanding rather than generation, the idea had influence on the efforts in story production as well. Basically, a story grammar claims that a formal structure can be assigned to a story in analogy to the syntactic structure ascribed to sentences in linguistics. Accordingly, story

grammars are represented in the form of rewrite rules; the grammar given by [17] starts with the following:

story $\longrightarrow$ setting + episode

setting $\longrightarrow$ (state)*

episode $\longrightarrow$ event + reaction

In brief, this means that a story consists of a 'setting' element, followed by an 'episode', where the setting is a series of 'states', and the episode is in turn composed of an 'event' and a 'reaction' (which are further defined by other rules). In parallel to these rules describing form, a set of semantic rules is supposed to capture the relations of meaning holding between the elements of a story.

Obviously, the notion of story grammar is attractive from the perspective of computer modelling, as it offers a clear account of "storyness" that allows for formally parsing stories in ways very similar to automatic sentence parsing. However, it was noted (see, e.g., [22]) that a grammatical approach to stories imposed very strict limitations and is not flexible enough to describe the mechanisms that make a story "work". Other theories were proposed, and in Artificial Intelligence (AI), the work on story generation focused on modelling the *reasoning processes* of the characters acting in a story, as well as those involved in story production itself.

2 Story generation systems in "Classical" Artificial Intelligence

An early storytelling program, often regarded as the pioneer of the field, was written by James Meehan for his Ph.D. thesis at Yale University in 1976 [10]. Under the title "The metanovel:

writing stories by computer" Meehan presented a program called 'Tale-Spin' that allowed the user to define a basic "setting" and a cast of characters together with a particular "problem" that one of the characters sets out to solve. The program has a problem-solving module that holds some knowledge on breaking goals into simpler sub-goals that can be solved individually; this aspect of the work is rooted in the tradition of the work by Roger Schank's group at Yale on "Scripts, Plans, Goals and Understanding" [18]. In the 1970s, this framework of research was very prominent in the research field that nowadays is often referred to as "classical" AI. In a nutshell, the underlying idea was that problem solving in terms of goal decomposition can be seen as the central ability in human cognition; and it can be simulated with computers using symbolic representations of the knowledge in conjunction with procedures manipulating these representations.

The level of problem solving employed in Tale-Spin, and thus manifested in the stories, is one of straightforward cause–effect links, such as "X is thirsty – X wants water – X does Y to get water – X drinks water – X is not thirsty anymore." To compute these sequences, a set of fairly general rules is instantiated with the specific problem and applied to find the solution. The type of tale might be said to resemble narrations produced by small children, a claim that was indeed explicitly made for a similar program some twenty years later, to which we will turn below. When reading the stories produced by Tale-Sping, one immediately notices the relatively poor "surface quality" of the text: it lacks pronouns and variation in expressions; the sentences are short and have only very few cohesive elements to glue them together. Basically, a text is a sequence of individual short statements, and thus the language sounds quite unnatural—even though one might argue that deficits of this kind are also a "feature" of child language. On this point, the author acknowledges that "the addition of personal pronouns to the output would have enhanced the quality of the output but were not

central to the problem of generating the events of a story." [11, p. 197] This issue, too, will be taken up below in the discussion of a later program. At this point, we briefly explain the inner workings of Tale-Spin in producing its stories.

The "knowledge base" coded within the program holds descriptions of eight different kinds of characters and of about fifty kinds of physical locations. Problem solving procedures have been defined for the acts of acquiring objects and information, transfering information, persuading, bargaining, and asking information. To start story production, the program user defines a list of characters and assigns a goal to one of them. Suppose the user states that "John wants to visit Mary"; then the corresponding problem-solving procedure is triggered, which sets up a transportation goal, which might in turn expand to other sub-goals, and so forth. When these procedures are executed to build an abstract "plan" describing the way to the problem's solution, the story is produced as a side-effect. On the linguistic side, the program has a vocabulary of fifty verbs and a small number of nouns and adjectives.

In the 1970s, the stories automatically produced by Tale-Spin were quite innovative examples of computer 'reasoning'. The program treated problem solving as the driving force of story production, thereby advancing the view that coherence of a story arises simply from the fact that one or several characters exhibit rational, goal-driven behavior. This was in line with the contemporary research in Artificial Intelligence, and to a good extent in Cognitive Psychology, where the capacity of solving problems by applying symbolic rules was seen as a central feature of human intelligence.

Almost twenty years after Tale-Spin, another story generator was written along the lines of the "scripts, plans, goals" research framework. 'Minstrel' by Scott Turner [19] also sees problem solving as the central ingredient of stories. To implement it, he uses top-down goal

expansion as sketched above, and in addition the more recent AI-technique of "case-based reasoning", where one problem is solved by recalling another, similar one from memory and adopting the old solution to the new problem.

Importantly, Turner places problem solving not only inside the story plots, but views the entire process of storytelling as the "purposeful achievement of author goals". Minstrel's stories are in the domain of King Arthur and his noble knights; they are little fables illustrating a morale such as "deception is a weapon difficult to aim".

As for "surface quality" the same comment made above on Tale-Spin applies here as well. Sentences are short statements and often repeat the same words, as in the schema "A loves B. Because A loves B, A wants B to love A", and so forth. By employing results from research in 'natural language generation', sketched below in the final section, the cohesion of such stories could easily be improved, but again this was not the author's main goal. Instead, the "deeper" process of solving problems on the basis of experience is treated as the central task. The program gets started when the user provides clues to the theme of the story to be developed; then, the system's memory is explored to find similar schemes that have been employed in the past. This process of adapting old solutions to new problems is put forward as a theory of creativity, which will be dealt with in Section 5.

In Minstrel, the story production task itself is seen as an instance of 'problem solving'. The story originates by succesively applying procedures that refine goals; these fall into four groups: 'Thematic goals' ensure that the story indeed illustrates the theme given, by creating an appropriate sequence of events. 'Consistency goals' supervise the resulting story to ensure its consistency; for example, they can invoke adding material covering the preconditions of some action. 'Presentation goals' are in charge of selecting and ordering the material, and

expressing it in English. 'Drama goals' serve to create simple forms of tragedy, suspense, irony, characterization, and foreshadowing.

The developer of Minstrel had the artificial stories evaluated by test subjects, and from this inquiry he concluded that the stories are comparable to those written by a younger high-school student.

In summary, storytelling in the AI tradition involves relatively deep analyses and formalizations of relatively isolated phenomena; the events of the story are actually simulated by the computer program, as a model of human reasoning. A more recent example of work in this category is the 'Brutus.1' system [5], which aims at an algorithmic formalization of betrayal and produces stories around this notion. In all these AI approaches, as a result of focusing on few specific aspects, the range of things a program can talk about is rather limited, and the stories, being produced "from scratch" by simulating the events within the program, typically sound somewhat unnatural. The storytelling systems discussed in the next section approach the topic roughly from the opposite direction.

3 Interactive fiction

In the 1980s, the so-called computer "adventure games" became increasingly popular forms of entertainment. Based either solely on text or on mixed text and graphics, these games would present an artificial world to the player, who could interact with the program by issuing simple commands as to where to go next in this world and what to do; the program, i.e. the representation of the artificial world, would react appropriately. In effect, the player was the protagonist of a story-to-unfold, and how it unfolded was to a good extent determined by the player. These programs have meanwhile improved considerably, and the field

of activities is now labelled *interactive fiction* (IF). Interactivity means that in effect every player/reader encounters a different story, depending on the choices made at the various points of interaction—it can be seen as an advanced version of a hypertext system. In contrast to classical AI storytelling, IF emphasizes breadth over depth: a program has an entire "world" to talk about, but the talk is largely not produced by deep reasoning, but by pre-stored data and chunks of text. Interestingly, though, some more recent work in IF makes contact with AI, as will be outlined at the end of the section.

The enduring popularity of text adventure games has led to the development of special programming languages for creating IF. Gradually, these were enriched with pieces of "world knowledge". An advanced language of this kind is TADS (The Text-Adventure Development System), freely available on the Internet. It supplies a range of readily defined object-types equiped with some minimal knowledge about their own behavior. For example, container and vehicle objects "know" that objects placed within them will move with them, and that they will be invisible unless the container or vehicle is transparent. TADS offers a hierarchy of object definitions along these lines, which an author developing IF can employ to efficiently build a system without worrying about programming details.

The same motivation led to the development of the "Erasmatron story engine", which is now a product on the market. Moving even further away from programming languages, it provides an authoring environment enabling users to write IF without knowing how to program. Using this system, the author creates a cast of actors and specifies their personality traits, then sets the stage on which the story will take place. Most importantly, the author has to define a web of actions defining the behaviors encountered in the story. Finally, the system takes over and the "storytelling engine" builds up a story based on the authors' pre-settings. Choice points

are calculated, where the reader can influence how the story proceeds, as in any IF system.

A much more ambitious IF-developing system is built in the long-term research project "Oz" at Carnegie-Mellon University [2]. The work in this project centers on the two issues of 'believable agents' and 'interactive drama', and both have been explored in great detail over more than 10 years. The basis of Oz is, again, a simulated physical world with characters and an interactor; then, several complex modules are in charge of producing "good" fiction.

The 'drama manager' of Oz is an attempt to overcome the fundamental theoretical problem of IF: the incompatibility between interactivity and drama. A 'conventional' story can be dramatic due to the temporal structure imposed by the author, whereas the reader's free choices in IF might very well take the dramatic coherence away from the enterprise (and the notion of 'author' becomes increasingly difficult to define).

Oz responds to this problem with a theory of "plot points": the important moments in a story that critically determine the overall course of actions. By isolating the set of possible plot points, the various permutations of the elements of this set is the theoretically possible set of all stories; most of these will make little sense, though. The drama manager module of Oz [21] uses an evaluation function that judges the quality of a particular series of plot points and thus can tell a 'good' from a 'bad' story. Story unfolding in Oz then is a process that is on the one hand driven by the user, who can in typical IF manner exercise choices on how to proceed; on the other hand, the drama manager supervises the activities and notices the important plot point transitions that occur. Using the evaluation function, the module determines a 'good' continuation of the story as originated so far, and twists the virtual world such that the events move in a direction towards a targeted subsequent plot point. This may happen by, for example, changing the physical world model, inducing characters to pursue

a certain course of action, or adding new characters. Thus, program and user effectively collaborate to produce a good story: Even though the user is choosing what to do or say, there is a type of "destiny", created by the author of the interactive drama. This destiny is not an exact sequence of actions and events, but is subtly shaped by the supervising drama manager system.

Besides the drama manager, the second major research area pursued in Oz concerns the creation of 'believable characters.' The idea is that the quality of a story depends to a great extent on the characters acting "natural" or "life-like." If the deeds or utterances of a character are un-motivated or senseless, the reader of the story will be disappointed, and rightly so. The Oz group has defined six requirements for characters to be believable:

- Personality: Characters should have unique and specific features that distinguish them from other characters.
- Emotion: Characters should exhibit emotions and respond to those of others.
- Self-motivation: Besides reacting to changes in the environment, characters ought to pursue their own goals and desires.
- Change: Depending on their personality, characters should change and grow with time.
- Social relationships: Characters should interact with others in accordance with the status of their mutual relationship. Furthermore, this relationship should change over time and in turn prompt differences in interaction.
- Illusion of life: Various basic requirements such as the capabilities of movement, perception, memory, or language.

The Oz project is a case in point to demonstrate the convergence between interactive fiction and artificial intelligence. More precisely, it is not the "classical" AI of symbolic reasoning that receives the most attention here, but the more recent branch of "behavioral AI", which emphasizes the importance of interaction between an agent and its environment: according to this research paradigm, the embedding of agents in concrete situations and the need to continuously react and act is a critical aspect of intelligent behavior. This hypothesis is at the heart of research in believable agents. When the system user engages in an interactive story, the characters encountered need to 'respond' instantly and sensibly for the whole environment to be convincing. Classical AI, on the other hand, has always focused on trying to mirror rational problem solving, by means of developing reasoning capabilities aiming at "correct" results. Thus the reliance on formal logic for implementing such systems. Clearly, the Oz interest in 'believable characters' is of a different kind: A character behaving "naturally" needs more (or other) capabilities than that of achieving correctness in rational problem solving.

Classical AI is clearly not irrelevant for the IF endeavor, though. The efforts in constructing descriptions of worlds that allow for simulating the actions of characters and their effects on the world have much in common with traditional AI approaches to knowledge representation. For instance, the CYC project [8] is a long-term and large-scale research program aiming at modeling human "common sense" as well as the basic laws governing what happens in a world. To this end, the domain of human understanding is broken into very many individual "micro-theories" in charge of modelling everyday notions like ailments, food-preparation, communication, corporate behavior, and so forth. These theories are represented in formal logic, so that programs can make use of the knowledge in order to reason about events.

The relevance of this research to IF is quite obvious: the more open-ended the process of

storytelling becomes and the more options there are for the reader to interact with the virtual world, the more world simulation is necessary on the side of the IF system, as it cannot rely on pre-stored chains of events any more. When the reader triggers an event in the world, the story must take all the effects of this event into account: the world model must be updated appropriately. This requires the kind of "deep" knowledge that the CYC team tries to formalize.

4 Natural Language Generation

We pointed out above that computer-produced stories such as those by Tale-Spin or Minstrel lack certain "surface linguistic features"—appropriate anaphoric expressions, for example—that would render the texts more readable and natural. These questions are attended to in the research field of 'natural language generation' (NLG): Given some abstract, non-linguistic representation of text content, how can it be mapped to well-formed and moreover "good" text in natural language? This includes the investigation of text structure and of thematic development from sentence to sentence, making appropriate lexical choices, selecting suitable referring expressions in the specific context (indefinite versus definite noun phrase, pronoun, elipsis), and other specific tasks.

Surprisingly, there is only little contact between NLG and the discipline of computer-storytelling (an exception is the Oz project, see [7]). Applications developed by NLG researchers are, for example, the production of reports from input "raw" data (such as weather reports or stock market reports) and the generation of instructional text such as user manuals. NLG-systems, if they are designed in a modular fashion, can to a good extent be re-used in different applications. To this end, typical text generators consist of two independent modules:

- The *text planning* module accepts data structures created by some underlying application program and converts these to an abstract linguistic specification.
- The *realization* module transforms the linguistic specification into a text in a natural language.

If the text planner is versatile enough, it can be configurated to accept different kinds of input structures from different applications; and if the interface between text planner and realizer is thoroughly defined, different realizers for different natural languages can be used with the same text planner.

It is therefore feasible to employ existing NLG technology for purposes of storytelling by designing solely the application supplying the input for a text planner; or at least it is possible to employ existing realization modules and build only a story-specific text planner on top. One system demonstrating this versatility is the SPOKESMAN generator [12], which has amongst several other purposes been used for generating narrations about characters trying to accomplish certain goals, quite similar to the scenarios in Tale-Spin or Minstrel. This particular SPOKESMAN environment was called SAGE [13] for 'Simulation and Generation Environment': Based on a represented model of a house, the program simulates the actions of a cat and a mouse following their particular goals, and simultaneously produces a verbalization of the events. For this purpose, the standard SPOKESMAN modules are employed. The resulting stories, however, are clearly recognizable as being a side-effect of a running simulation; here is an example:

> Fluffy wants to catch a mouse. He is looking for her. The mouse wants to get cheese. She is leaving a mouse-house. She is going toward it. Fluffy is chasing the mouse. He is going toward her. He caught her. The mouse didn't get the cheese.[1]

If some more *text planning* were performed prior to realizing the individual event descriptions, sentences could be conjoined to form a more cohesive text. To this end, NLG builds upon research in *discourse structure* to devise appropriate formal representations of text content. One approach that was quite influential in NLG is 'Rhetorical Structure Theory' [9], which posits that the structure of a coherent text can be characterized by the facts that adjacent portions of text are always linked by a certain coherence relation (such as condition, purpose, sequence, elaboration) and that assigning these relations applies recursively to yield a tree structure that defines the rhetorical structure of the entire text. Mann and Thompson proposed about 20 relations, defined in terms of the effects their presence has on the reader's cognitive state; NLG researchers in turn formalized these definitions in order to perform text planning: achieve a higher-level goal (e.g., persuade the reader to believe something) by breaking it up in a series of sub-goals that correspond to establishing rhetorical relations.

Following approaches of this kind, it is possible, to generate coherent and cohesive text that tries to achieve particular purposes and can be tailored to various kinds of readers to have a maximum effect. For example, the NLG program 'Pauline' [6] reports on the result of a fictitious primary election and it can be parameterized to take a particular perspective toward the results, which includes positive or negative evaluations. The parameters are a set of "rhetorical goals" whose values are set by the user of the program. In response, Pauline decides on which information is included in the text and which is left out, how the information is arranged, which words are used, etc., so that the text is as brief or verbose as desired, or conveys evaluative overtones. Thus, given the same underlying "content" (the election result), the program can produce appropriate text variants for different purposes, as represented by the parameter settings.

In conclusion, NLG research can be characterized as looking into the fine-grained details of the theoretical groundwork for automatic language production, of which computer storytelling is but one application.

5 Creativity and Humor Generation

Work in automatic story generation, being generally in its early stages, usually does not make much reference to attempts of actually explaining *creativity*, i.e. the human capacity that is generally assumed to be a driving force behind activities such as storytelling. A notable exception is the aforementioned program 'Minstrel', which was introduced by his creator as embodying a theory of creativity. As explained above, when Minstrel attempts to solve a problem, it searches memory for an episode representing a solution to a similar problem, and then tries to adapt the earlier solution to the present problem. In effect, it tries to abstract from the specific details of individual problems, and to form generalizations over classes of problems exhibiting the same structure. In the view of Turner [19], the set of rules characterizing how old problem–solution pairs can be mapped to new problems explains the nature of human creative behavior.

Under the umbrella of Artificial Intelligence, there is some other research on the nature of creativity, but this is usually not immediately related to linguistic production. One—relatively small-sized—research area, however, builds bridges between storytelling and creativity: the study of *humor* and its production, possibly with computational models. Here, it is often proposed to see *metaphor* as the link between creativity and humor. Metaphor, in the view of cognitive linguists, is at the very heart of human cognitive ability: it allows us to view one concept through the lens of another, thereby possibly uncovering analogies that were not perceived before. Two diverse situations are related and reconciled—quite similar to what

Turner [19] had described, even though he didn't explicitly refer to this process as 'metaphorical'. Humor, then, can be explained as a "perversion of an underlying metaphoric mapping." [20, p. 121] Both good metaphor and good humor rely on their ability to surprise the reader: metaphor shows that surface dissimilarities might very well conceal deeper conceptual similarities, while humor deliberately misleads the hearer along a path of convention, which then unexpectedly goes astray. The punchline of a joke very often points to an ambiguity that had so far remained in the dark.

Can the processes underlying humor be explained by computational theories and implementations? There are no very convincing joke-telling programs yet, but some initial steps are being taken. The root of some work on 'computational humor' is the "script-based semantic theory of humor" developed by [15]. It builds upon the notion of script as referenced above [18]: a representation of a prototypical flow of event, such as the series of steps involved in going to a restaurant, dining, paying, and leaving. According to Raskin, humor originates when a text is compatible with two different scripts, which—importantly—are in some way opposed to one another: normal versus abnormal, real versus unreal, etc. When being presented with the text, the hearer or reader first activates in the mind only one of these scripts, the more conventionalized one. Only the punch line of the joke triggers recognition of the other script underlying the narration; this late recognition is based on ambiguity or contradiction.

Extensions to this theory were later proposed by Attardo and Raskin [1], who proposed six levels of 'knowledge resources' needed to characterize a humorous text, which successively move from the linguistic surface to more abstract representation:

1. the text itself

2. the narrative strategy, or expository genre, underlying the text

3. the target domain of the joke

4. the situation described

5. the logical mechanism underlying the joke (such as figure/ground reversal)

6. the relation of script opposition

Such steps toward uncovering the "mechanics" underlying humorous text suggest the attempt to build computational models, but these have not been elaborated yet. Some initial thoughts on building joke-recognizing and joke-telling computer programs are given in [16].

The work of Raskin and Attardo can be seen as an attempt to find a global, all-encompassing theory of humor: what are the general principles responsible for rendering a certain piece of text humorous? This is an ambitious goal, and—not surprisingly—there have so far been no more than preliminary proposals on what a computational system implementing such a theory ought to look like. A different line of research approaches the problem from the opposite end and examines specific *kinds* of humorous texts in great detail so that a computational model can be developed for this limited domain; the hope then is that from various models of individual humorous genres, some general model can later be induced.

In this way, Binsted and Ritchie [3] investigated *riddles*: simple question/answer jokes usually based on some form of pun. The authors give the example: *What do you use to flatten a ghost? – A spirit level.* The regular structure of these jokes prompted Binsted and Ritchie to develop a schema-based approach to producing them automatically. On the basis of a (humor-independent) lexicon and a set of schemas and templates characterizing the structure of (a sub-class of) riddles, their program produces word-substitution puns that rely on the phonological identity (homonymy) of distinct words. The schemata encode knowledge about

some semantic relationships and about homophones, and the program employs these schemata to construct riddles such as: *What do you get when you cross a sheep and a kangaroo? – A woolly jumper.*

In later work, Binsted and Ritchie turned to a similar class of jokes, where the punchline is a distorted form of some popular saying (e.g., a proverb) [4]. They developed a model of producing story puns, which is not implemented yet, but it can be quickly summarized as follows. After choosing a maxim as basis for the joke, the maxim is distorted by a set of rules, and then a representation of the meaning is built up. For actually producing the story, the authors propose developing an evaluation function that can rate how well a story achieves the desired effect (recall the role of evaluation functions in the drama manager of the 'Oz' system), and for creating the candidate stories they hint at the utility of 'genetic algorithms', a class of algorithms drawing a loose analogy between computer programming and the Darwinian notion of genetic variation plus natural selection. The idea is that a system would produce stories by starting with a raw version and successively improve it in small steps, guided by the evaluation function.

From small-scope studies of the kind just described, Binsted and Ritchie hope to derive ideas about the ingredients of humor in general, and ultimately to build computer models capable of producing a broader range of humorous texts—an endeavor that can be expected to shed some more light on the nature of human creativity.

Notes

[1] Source: Ref. [13]

References

[1] S. Attardo, V. Raskin. "Script theory revis(it)ed: Joke similarity and joke representation model." In: *HUMOR: International Journal of Humor Research* 4:3–4, pp. 293-347, 1991.

[2] J. Bates. "Virtual Reality, Art, and Entertainment." In: *Presence: The Journal of Teleoperators and Virtual Environments* 1(1):133-138, MIT Press, 1992.

[3] K. Binsted, G. Ritchie. "An implemented model of punning riddles." In: *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, Seattle, 1994.

[4] K. Binsted, G. Ritchie. "Speculations on Story Puns." In: J. Hulstin, A. Nijholt (eds.): *Automatic Interpretation and Generation of Verbal Humor* (Proceedings of TWLT 12/IWCH 96, The International Workshop on Computational Humor). Enschede/Netherlands: University of Twente, 1996.

[5] S. Bringsjord, D. Ferrucci. *Brutus.1: The State of the Art in Story Generation.* Hillsdale, NJ: Lawrence Erlbaum. Forthcoming.

[6] E.H. Hovy. *Generating natural language under pragmatic constraints.* Hillsdale, NJ: Lawrence Erlbaum. 1988.

[7] M. Kantrowitz, J. Bates. "Natural Language Text Generation in the Oz Interactive Fiction Project." In: Dale, R., Hovy, E., Rösner, D., and Stock, O. (eds.): *Aspects of Automated Natural Language Generation*, pp. 13-28. Berlin/New York: Springer-Verlag, 1992.

[8] D.B. Lenat. "CYC: A Large-Scale Investment in Knowledge Infrastructure." In: *Communications of the ACM* 38 (11), 1995.

[9] W. Mann, S. Thompson. "Rhetorical structure theory: Towards a functional theory of text organization." In: *TEXT*, 8:243-281, 1988

[10] J. Meehan. "The Meta-Novel: Writing Stories by Computer." Ph.D. Thesis, Research Report Nr. 74, Computer Science Department, Yale University. 1976.

[11] J. Meehan. "TALE-SPIN." In: R.C. Schank, C.K. Riesbeck (eds.): *Inside Computer Understanding.* PP. 197-226. Hillsdale, NJ: Lawrence Erlbaum. 1981.

[12] M. Meteer. "Portable Natural Language Generation using SPOKESMAN." In: Proceedings of the 3rd Conference on Applied Natural Language Processing, Trento/Italy. 1992.

[13] M. Meteer. "Generating Event Descriptions with SAGE: a Simulation and Generation Environment." In: Proceedings of the 7th International Workshop on Natural Language Generation, Kennebunkport/ME, pp. 99–107. 1994.

[14] G. Polti. "The Thirty-Six Dramatic Situations." Franklin/OH: J.K. Reeve. 1921.

[15] V. Raskin. *Semantic Mechanisms of Humor.* Dordrecht/Boston/Lancaster: Reidel. 1985.

[16] V. Raskin. "Computer implementation of the general theory of verbal humor." In: J. Hulstin, A. Nijholt (eds.): *Automatic Interpretation and Generation of Verbal Humor* (Proceedings of TWLT 12/IWCH 96, The International Workshop on Computational Humor). Enschede/Netherlands: University of Twente, 1996.

[17] D.E. Rumelhart. "Notes on a schema for stories." In: D.G. Bobrow, A. Collins (eds.): *Representation and Understanding.* Academic Press, New York. PP. 211–236. 1975.

[18] R.C. Schank, R.P. Abelson. *Scrips, Plans, Goals, and Understanding.* Hillsdale, NJ: Lawrence Erlbaum. 1977.

[19] S.R. Turner. *The Creative Process – A Computer Model of Storytelling and Creativity.* Hillsdale, NJ: Lawrence Erlbaum. 1994.

[20] T. Veale. "No laughing matter: The cognitive structure of humor, metaphor, and creativity." In: J. Hulstin, A. Nijholt (eds.): *Automatic Interpretation and Generation of Verbal Humor* (Proceedings of TWLT 12/IWCH 96, The International Workshop on Computational Humor). Enschede/Netherlands: University of Twente, 1996.

[21] P. Weyhrauch. "Guiding Interactive Drama." Ph.D. Thesis, Department of Computer Science, Carnegie Mellon University. Technical Report CMU-CS-97-109. 1997.

[22] R. Wilensky. "Story grammars versus story points." In: *The Behavioral and Brain Sciences* 6, pp. 579–623. 1983.