Making Cognitive Summarization Agents Work In A Real-World Domain

Brigitte Endres-Niggemeyer and Elisabeth Wansorra

Fachhochschule Hannover (University of Applied Sciences and Arts)
Dept. of Information and Communication
Ricklinger Stadtweg 120, D-30459 Hanover, Germany

Abstract. The advantage of cognitively motivated automatic summarizing is that human users can better understand what happens. This improves acceptability. The basic empirical finding in human summarizers is that they combine a choice of intellectual strategies. We report here on SummIt-BMT (Summarize It in Bone Marrow Transplantation), a prototype system that applies a subset of human strategies to a real-world task: fast information supply for physicians in clinical bone marrow transplantation. The human strategies are converted to knowledge-based agents and integrated into a system environment inspired by user-centered information seeking research. A domain ontology provides knowledge shared by human users and system players. Users' query formulation is supported through empirically founded scenarios. Incoming retrieval results are first roughly checked by means of text passage retrieval before the agents apply strategies of competent human summarizers. The presumably relevant text clips are presented with links to their home positions in the source documents. SummIt-BMT has reached the state of a prototype running on a Macintosh server (http://summit-bmt.fh-hannover.de/).

1. Introduction

Summarization is a cognitive task. It means building a mental representation of a body of mostly external information, reducing it to the most relevant items, and uttering or generating the content of the reduced representation - the summary. Skilled comprehension and reduction of input knowledge are the hallmark of summarization. Recent overviews of (automatic) summarization are found in [1], [2], [3].

We are currently implementing SummIt-BMT (Summarize It in Bone Marrow Transplantation), a prototype system that applies results from earlier empirical and experimental work on human summarizing. In this paper, we report on how the findings about human summarizing are reflected in our system realization: they give rise to summarization agents in an appropriate system environment.

Endres-Niggemeyer B. and Wansorra E. (2004).
Making Cognitive Summarization Agents Work In A Real-World Domain.
In *Proceedings of the 1st International Workshop on Natural Language Understanding and Cognitive Science*, pages 86-96
DOI: 10.5220/0002662900860096

Empirical Modeling Of Human Summarizing

It is commonplace knowledge that cognitive processing reacts to environmental factors. In order to observe performance under real-world conditions and obtain realworld observations, researchers contact their field subjects in their everyday environment, visiting them at home or at office, as stipulated by the principles of field research and naturalistic inquiry [4]. Endres-Niggemeyer [5] worked with six human summarizers (abstractors / indexers - four Germans, two Americans) who delivered nine summarization processes per person under thinking-aloud conditions. The thinking-aloud data was exploited together with input documents, intermediate notes, summary drafts and final versions. Interpretation followed cognitive models about human text understanding and learning from text [6, 7, 8]. They explain how a person recognizes and learns knowledge items from texts (i.e. concepts, propositions) by referencing own prior knowledge. In computational approaches, a knowledge base, which may be an ontology, replaces the knowledge stored in human brain.
The cognitive approaches of the six professional summarizers turned out to be well organized. Overall process organizations differed, but everybody proceeded by working steps dealing with an information item like a document, a chapter, a paragraph, a sentence, a phrase or a word at a time. These working steps were reconstructed with intellectual strategies, referring to input, output, and knowledge available in the summarizer's memory. Groups of intellectual strategies were found to interact in individual working steps. For explaining all occurring working steps, some 550 different intellectual strategies were needed. The summarizers shared a considerable subset of these strategies: 83 strategies were used by everybody in the group, 60 strategies were shared by five summarizers, another 62 strategies proved to be common knowledge of four group members, 79 strategies belonged to the repertory of three of the experts, 101 were used by two of them, and 167 strategies were individual. Especially the shared strategies are assumed to be of general interest. For main strategies, a control study with non-professionals (students of linguistics) supported the claim that educated non-professionals and professionals share a core set of summarization methods [9].
Human summarizing integrates well with information seeking. Summarizers first identify items that fit the current task or interest. After focusing on one item, they scale down in the document and pick passages of manageable size that contain useful material, using all sorts of cues such as catchwords or page design features, and often refer to metadata: the table of contents or an index. After having found roughly paragraph-size text passages of high interest, summarizers read them very carefully, possibly checking and looking up what remains unclear to them. They choose sentence-size units for their own records. Summaries are mostly constructed by cutting and pasting, own formulation occurs only seldom. The resulting extract is often revised. It may need only some slight retouching to emerge as an acceptable abstract.

SimSum (Simulation of Summarizing)
After empirical and experimental investigation, a small-scale implementation called SimSum (Simulation of Summarizing - included in [5]), see also [10]) was realized in

order to demonstrate that human strategies converted to knowledge-based agents can account for real summarization sequences. We implemented some 80 agents for subtasks of summarization as needed for the simulation of selected human summarization sequences. The agents are able to deal with their restricted input, but are by no means fit for work with unknown text. However, SimSum visualizes what happens in a summarizer's mind, having little creatures acting it out on the screen.

SummIt-BMT (Summarize It in Bone Marrow Transplantation)
As soon as feasibility of summarizing according to human techniques was demonstrated, Bone Marrow Transplantation (BMT) was chosen as a first real-world application domain. Bone Marrow Transplantation is a specialized and life-critical area of hematology. It has a key function in many cancer therapies. SummIt-BMT and its agents are intended to procure fast back-up knowledge from web sources for clinical physicians in BMT. The overall system is documented at our website http://summit-bmt.fh-hannover.de/. Here, we focus on its cognitive perspective: first we briefly explain the user interface and the domain ontology. Second, we describe the agents reflecting human cognitive strategies as far as we currently have them.

2. Cognitive Agents In SummIt-BMT

2.1 Summit-BMT - The Place Where The Cognitive Agents Live

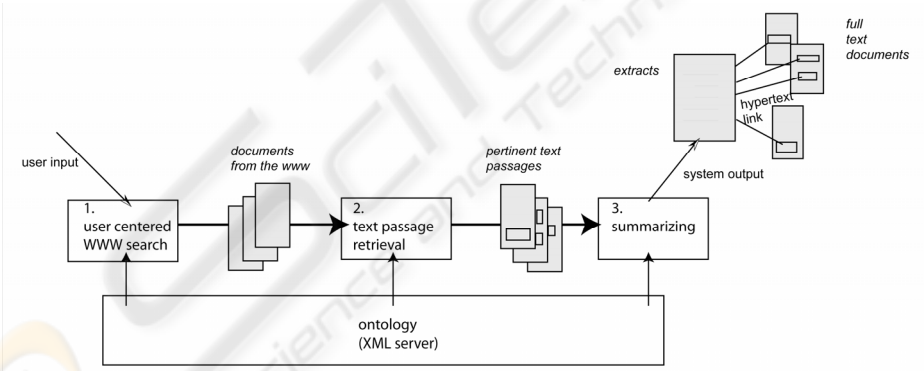

Fig. 1. Summarization process in SummIt-BMT

Inevitably, human strategies undergo changes when they are reconstructed as computerized cognitive agents. Knowledge-based agents have no real grasp of perception, but they have direct access to computerized symbols. In the system knowledge base they find the knowledge, which is the prerequisite for competent interpretation. The agents are restricted to intellectual or symbolic tasks: extracting concepts and larger knowledge items like propositions from input text, interpreting

them with reference to the knowledge base, manipulating them as requested by a task, inferencing included, and possibly some utterance or text generation.
For maintaining the essentials of human performance under changed conditions, it is important to establish a system environment (cf. Figure 1) that supports or reflects as much as possible a natural human approach to the task at hand, here to summarization and summary use. This implies the interaction of users and system players. Both parties rely on knowledge.

User Interaction
Our user interaction component heeds the claim of user-centered research on information seeking ([11], [12], [13]) that users are entitled to state their information needs in their own thinking and working framework and that the interaction with a retrieval interface is there for "helping people to find what they don't know" [14]. The SummIt-BMT user interface (more detail in [15]) presents scenarios derived from everyday working situations of clinical physicians.

Knowledge Supply
The corpus-based SummIt-BMT ontology with its annexes is the central knowledge resource for system components, agents included, and users. A systematic empirical research procedure for ontology content was applied, referring to the experience of thesaurus construction [16] and to grounded theory development [17]. The ontology supports query formulation for information retrieval, text passage retrieval, and summarization proper.
The ontology comprises circa 4500 concepts. They participate in around 4800 propositions, represented as Prolog-style Horn clauses. These propositions are combined to set up around 2500 context expressions combining a core and a context part [18]. They represent knowledge items which are too big for a single proposition.
Propositions are equipped with semi-formalized occurrence descriptions stating the surface forms propositions may assume in running text. Currently, some 1500 of them are also equipped with unifiers from a stock of some 300 unifiers.

The Summarization Process Involving Cognitive Agents
The summarization process of SummIt-BMT (Figure 1) conforms to competent human practice as described above. It integrates with information seeking activities. Before summarization agents are invoked, the retrieved documents are screened for paragraph-size units that feature some concepts used in the question. The resulting promising passages are the input for summarization proper and the agent team.
According to the online assumption of human understanding [6], the agent team treats one passage of incoming text after the other. Several agents cooperate in deciding about the relevance of an item. They check concepts and their relations referring to typical arguments used in summarization. Factual knowledge or other prior knowledge may influence them as well. When they are done, they deposit the relevant text clips in the question-answering scenario.

2.2 The Agents

In SummIt-BMT, a group of agents, which stand for cognitive strategies, reproduces a human processing model. In principle, each agent enacts a summarization strategy used by competent humans. It may invoke specialist subagents for a complex task.
All agents in the SummIt-system are implemented as Java classes. Since Java is object oriented, the agents inherit from the superior Agent class some fundamental methods such as parsing the document if special information is needed and using the knowledge base, e.g. the ontology or the unifiers. These methods provide the basic knowledge necessary for text processing and summarizing. But the agents use it in different ways depending on their task within the summarization process.

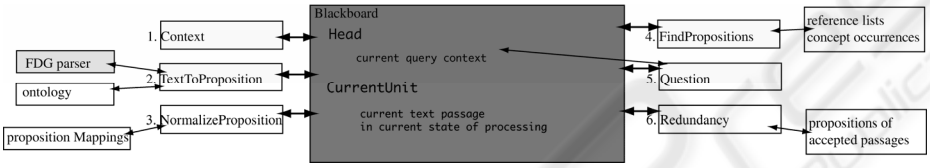

Fig. 2. The summarization agents around the blackboard

The summarization agents work around an XML-based blackboard which serves as their main communication medium (see Figure 2). Its head states the current query context as set by the query scenario. In the current unit, the passage under consideration is stored and reworked. The agent group shown in Fig. 2 is the small starter team that is currently implemented. It is far from being fully staffed. At present, the agents have not yet reached their full capacities. The whole group is reproached for being awfully slow. Their more or less pipelined organization does not yet really reflect the cognitive profile seen in human processing.

Context
The *Context* agent behaves like a human reader who superficially checks whether she is dealing with a suitable document by looking for cues such as relevant concepts. Its decision can be traced back to humans who reject a document without interest for them or for their users (strategy *relevant-in-scope* of the empirical model).
Context screens the documents found during text passage retrieval for concepts of the summarization target context from the blackboard. These concepts have not systematically participated in the web query. Human summarizers know how to profit from document structure and so does the agent. In medical articles with a known superstructure it limits its activities to sections entitled "Introduction" and "Patients and methods" or "Materials and methods", in all other papers it inspects the whole text. When *Context* starts, all texts are already converted into XML-structured documents, so the agent can easily navigate in them and does not need further knowledge about different document formats.

If *Context* discovers at least one of the terms of interest, the document is kept on the blackboard for further investigation and potential summarizing by the following agents, otherwise the whole document is thrown away and the blackboard is cleaned.

Text Interpretation Agents

The next four agents form a computerized and reduced model of task-oriented human understanding of paragraph-size text passages:

- *TextToProposition* sets up a text-driven meaning hypothesis of a clause or sentence, relying on the verbal case frame.
- *NormalizeProposition* tries to match the meaning proposed by *TextToProposition* with a proposition of the ontology, thus verifying it in the light of prior knowledge.
- *FindPropositions* corresponds to a search for meaning that first identifies interesting concepts and then tries to relate them according to a model proposition found in the ontology.
- *Question* introduces the functionality of task-oriented selective understanding. It checks which of the propositions found in the current text passage relate to the summarization target and discards propositions that do not qualify. This is a core strategy of human summarization.

Based on the knowledge represented in the ontology and its annexes, the agent group extracts propositions of interest from input and constructs the internal representation of the text. By dismissing all candidates that do not match the summarization target, they reduce the material under consideration. They enact techniques of abductive text interpretation [8].

TextToProposition

TextToProposition accepts the sentences of promising paragraphs and transforms them into text-based candidate propositions.

First, it transforms the English text on the blackboard into preliminary propositions with the aid of the Connexor Functional Dependency (FDG) parser [19] and some proposition production rules. Furthermore it embeds the statements into the conceptual background by finding out their semantic roles / ontology classes and by linking the arguments to context expressions that have these concepts in their core. Whenever possible the agent adjusts the used nouns to the preferred concepts of the ontology, replacing their textual synonyms or different spellings.

To smooth input for the parser the agent engages the technical helper agent *Sentence*, which reduces sentence length by replacing semicolons with full stops and eliminating the bibliographic references. A further helping strategy called *Morpho-syntacticChecker* invokes the parser for every text passage. It analyses all sentences of the paragraph. The resulting representations are XML structures that reflect the dependency structure of the sentences as well as morphological and syntactic information. *MorphosyntacticChecker* offers methods to use these results for recognizing e.g. the main verb, subject and object or conditionals.

Production rules are applied to the top of the dependency structure in order to find embedded clauses. For the latter, separate representations are derived. For conditionals for example, this rule says that if the dependency pattern contains the

sequence “cnd:subj:” then two propositions are constructed. The first one uses the main verb as predicate and needs one argument for the subject and one for the object role, the second predicate and its arguments are taken from the conditional phrase.
TextToProposition always uses the main verb of the investigated sentence as predicate of a preliminary proposition. The production rules enter the arguments corresponding to the dependency structure. The resulting proposition is a XML-element containing the dependency pattern, the original sentence and a short form of the proposition as attributes and element nodes for the predicate and each argument, where the dependency is stored as an attribute and all possible roles as well as context expressions that contain this concept are children of the argument node (cf. Figure 3).

```
Result of TextToProposition
<sentence>      <text>    Cidofovir (CDV) is effective against CMV in vitro and has the practical advantage of weekly administration.   </text>
                <fol>
                <prop
                     type="main"
                     dependencyPattern="subj:comp:cc:cc:NULL:NULL:NULL:"
                     verbatim="Cidofovir (CDV) is effective against CMV in vitro and has the practical advantage of weekly administration."
                     short="be effective ( cidofovir , cidofovir )">
                     <pred>be effective</pred>
                     <arg dep="subj:">   cidofovir
                                         <role>Xdrug</role>
                                         <role>Xentity</role>
                                         <role>Xobject</role>
                     </arg>
                     <arg dep="mod:">    cidofovir
                                         <role>Xdrug</role>
                                         <role>Xentity</role>
                                         <role>Xobject</role>
                     </arg>
                </prop>
```

Fig. 3. A sample from an XML structure representing a proposition

If the parser failed or if there is no production rule for the resulting dependency pattern, the agent uses a heuristic to propose preliminary propositions. It combines the verb as predicate with all nouns as arguments and puts this list on the blackboard. There, the following agents can find it and try to rework the makeshift proposition.

NormalizeProposition
The *NormalizeProposition* agent tries to prove that a preliminary proposition proposed by *TextToProposition* matches a proposition of the knowledge base. If the match succeeds, the agent performs an ersatz understanding, otherwise the proposition is probably out of range and not relevant.

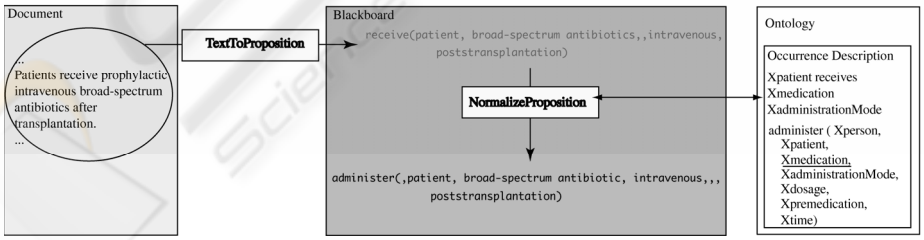

Fig. 4. How NormalizeProposition works

Figure 4 shows how *NormalizeProposition* works. The agent starts out with the predicate of the candidate proposition generated from input. It fetches an applicable proposition mapping / occurrence description from its private knowledge. Then the

agent tries to generate the target proposition of the mapping from the candidate proposition derived from input text. Normally, the name of the predicate in the candidate proposition and the ontology-accredited one will differ. As soon as a predicate and its syntax can be checked, it provides the semantic roles / ontology classes of its argument definitions. Some arguments are obligatory, others are facultative. In the case presented in Figure 4, only the third argument is mandatory. The agent tries to fill the arguments of the candidate proposition into the ontology-resident one using the semantic roles that *TextToProposition* added as child nodes to each argument. If it finds all obligatory arguments, it writes the ontology-resident proposition onto the blackboard for further use. If the agent fails, the preliminary proposition becomes input for *FindPropositions*.

FindPropositions

FindPropositions realizes a concept-driven text understanding as observed in proficient professional readers. The ontology concepts used in a candidate proposition serve as keys for finding interpretation hypotheses from the knowledge base. Predicates are of secondary interest. *FindPropositions* accepts candidate propositions and tries to match them to ontology-resident ones that share their concepts. Only if identical concepts link both of them, predicate compatibility must be checked.

In the current implementation, the agent skips all propositions that *NormalizePropositions* was able to handle, therefore its activity is somehow the last try to make text propositions usable for further processing. *FindPropositions* sifts through the arguments of the proposition and their reference lists. It gets the context expressions in the reference list attached from *TextToProposition* at each argument and counts the occurrences of terms in their core. The agent accepts core propositions sharing two or more concepts with the candidate proposition as valid justifications and interpretations of the candidate proposition. If the agent fails as well, the corresponding sentence or sentence part is not understandable to the system and therefore considered as irrelevant.

Question

The *Question* agent checks in detail whether a statement is related to the user's query. It accepts propositions that have been successfully interpreted by *NormalizeProposition* or *FindPropositions* and matches them to the propositions of the query scenario. *Question* demands that at least one proposition of a text passage can be unified with a question proposition of the current scenario. Otherwise it discards the passage because it does not relate / answer to the query.

In order to decide about the relevance of a passage to the question, *Question* compares all propositions of a passage with the core of the question context stored in the head of the blackboard. The agent takes each remaining proposition one by one, fetches the unifiers known for the current predicate from the ontology and tries to apply them to the actual candidate and the query propositions. First of all it checks whether the predicates are the same since otherwise no unification is possible. For each argument it tries unification in three steps: it takes over arguments that are equal, it fills in arguments if an argument slot is empty in one proposition and not in the other, and in

the last step the agent checks whether a unifier can resolve the difference between arguments in the two propositions.
If *Question* succeeds, the investigated proposition and the sentence it is derived from remain on the blackboard, together with all further related propositions. If no relation between at least one proposition of a sentence and the user's query can be found, the whole sentence is removed. And if *Question* cannot find any related sentence in a passage of the document, that whole passage is thrown away.

Redundancy
Redundancy weeds out doubles from the remaining propositions.
Repetitions are frequent in scientific texts. They serve reader-oriented purposes, such as recalling earlier topics before they are rediscussed or expanded. By weeding out redundant material, human summarizers reduce information size without real information loss (strategy *once* of the empirical model). *Redundancy* follows their example. It watches out for doublets and withdraws them. If all propositions of a passage are already known, the whole passage is wiped out.
When the agents have finished their work, the summary could be formulated by transforming the resulting propositions into real-language sentences. But it is also possible to take the original sentences from the document as we do in Summit-BMT, where, moreover, each text clip is linked to its position in the source document.

3. Conclusion And Outlook

Cognitively adequate automatic summarizing is possible in a real-world application. We have implemented three efficient main strategies of human summarizers: the scope decision (*Context* executing *relevant-in-scope*), selective text understanding guided by the question (*TextToProposition, Normalize, FindProposition* and *Question* executing *relevant-call*), and the reduction of redundant statements (*Redundancy* executing *once*). They were put into a system environment that enables them to demonstrate cognitively adequate behaviour as far as users can feel it. The agents work in analogy to human summarizing strategies and sort out material that does not answer the user's question. Like this the system helps to quickly find relevant information and reduces spurious material.
At the moment SummIt-BMT is far from being satisfactory in its performance. In particular, the agent team still has many shortcomings: the agents are too slow, too narrow-minded in their approaches, not really well organized, and some very useful agents are still missing. We shall improve that. Currently, anaphora resolution in the parser output by the MARS system [20] is under development.
For task- and concept-driven partial text understanding (this is the functionality of *relevant-call*) it is sufficient, cognitively adequate and faster to construct propositions only for sentences that include at least one relevant concept. We shall do so. Some relations from RST [21] will be added to *Question's* knowledge, so that relevant propositions can be linked to the query by means of discourse relations, too.

4. Acknowledgements

We gratefully acknowledge the contributions of our colleagues to system development. Implementation of SummIt-BMT is supported by the German Science Foundation (DFG) under grant EN 186/6-2. The first project phase was also funded by the German Federal Ministry of Education and Research (bmbf) under grant 1701200, and by the Ministry of Science and Culture of Lower Saxony under grant 1999.384.

References

1. Mani, I., Maybury, M. (eds.): Advances in Automated Text Summarization. MIT Press, Cambridge MA (1999)
2. Hovy, E.: Automated Text Summarization. In: Mitkov, R. (ed.): Oxford University Handbook of Computational Linguistics. Oxford University Press, Oxford (2003) 583-593
3. Afantenos, S., Karkaletsis, V.: Summarization Techniques and their Application on Medical Documents. AI Med, Spec Issue: Summarization and Information Extraction from Medical Documents (in press)
4. Lincoln, Y.S., and Guba, E.G.: Naturalistic Inquiry. Sage, Beverly Hills CA (1985)
5. Endres-Niggemeyer, B.: Summarizing Information. Springer, Berlin (1998)
6. Kintsch, W., van Dijk, T.A.: Strategies of Discourse Comprehension. Academic Press, Orlando FLA (1983)
7. Schnotz, W.: Textverstehen als Aufbau mentaler Modelle (Text comprehension as construction of mental models). In: Mandl, H. and Spada H. (eds.): Wissenspsychologie. Psychologie Verlags Union, Muenchen (1988) 299-332
8. Hobbs, J.R., Stickel, M., Appelt, D., Martin, P.: Interpretation as Abduction. Art Intelligence (1993) 69-142. http://citeseer.nj.nec.com/hobbs90interpretation.html
9. Endres-Niggemeyer, B., Waumans, W., Yamashita, H.: Modelling Summary Writing by Introspection: A Small-Scale Demonstrative Study. Text 11 (4) (1991) 523-552
10. Endres-Niggemeyer, B., Neugebauer, E.: Professional Summarizing: No Cognitive Simulation without Observation. Journal of the American Society for Information Science 49 (1998) 486-506
11. Belkin, N., Oddy, R.N., Brooks, H.M.: ASK for Information Retrieval: Part I. Background and Theory. Journal of Documentation 38(2) (1982) 61-7I
12. Bates, M.J.: The Design of Browsing and Berrypicking Techniques for the Online Search Interface. Online Rev (1989) 407-424
13. Marchionini, G.: Information Seeking in Electronic Environments. Cambridge University Press, New York (1995)
14. Belkin, N.: Helping People Find What They Don't Know. Comm ACM 8 (2000) 58-61
15. Becher, M., Endres-Niggemeyer, B., Fichtner, G.: Scenario Forms for Web Information Seeking and Summarizing in Bone Marrow Transplantation. Coling-02, Workshop Multilingual Summarization and Question Answering, Taipeh, Taiwan, 31.8. - 1. 9. 2002
16. Aitchison, J., Gilchrist, A.: Thesaurus Construction and Use: A Practical Manual. 3rd edn. Aslib, London(1997)
17. Glaser, B.G., Strauss, A.L.: The Discovery of Grounded Theory: Strategies for Qualitative Research. 11th edn. Aldine Atherton, New York (1980)
18. McCarthy, J., Buvac, S.: Notes on Formalizing Context. In: IJCAI'93 - 13th Int Joint Conf Artif Intell. Chambery, France (1993) 555–560 http://citeseer.nj.nec.com/318177.html.

19. Connexor Machinese Syntax. http://www.connexor.com/m_syntax.html
20. Mitkov, R., Evans, R., Orasan, C.: A New, Fully Automatic Version of Mitkov's Knowledge-Poor Pronoun Resolution Method. 3rd Int Conf Intelligent Text Processing and Comp Linguistics (CICLing-2002), Mexico City, Mexico (2002)
21. Mann, W. C., Thompson, S.A.: Rhetorical Structure Theory: A Theory of Text Organization. In: Polanyi, L. (ed.): The structure of discourse. Ablex, Norwood N.J. (1987)