Domain Communication Knowledge

Owen Rambow*
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104

Abstract

This paper advances the hypothesis that any text planning task relies, explicitly or implicitly, on domain-specific text planning knowledge. This knowledge, "domain communication knowledge", is different from both domain knowledge and general knowledge about communication. The paper presents the text generation system Joyce, which represents such knowledge explicitly.

The Joyce Text Generation System

The Joyce text generation system is a part of the software design environment Ulysses (Korelsky and Ulysses Staff, 1988; Rosenthal *et al.*, 1988). Ulysses is a graphical environment for the design of secure, distributed software systems. The design proceeds hierarchically by top-down refinement. A formal specification interface and a theorem prover allow the user to formally verify the flow security of the designed system.

Joyce is part of the user interface. Joyce generates different types of texts about software designs:

- It generates annotations of the design which are intended to serve as system documentation during and after the design process.
- It is used to explain the result of a heuristic security design tool, the "flow analyzer".

The texts Joyce generates are specifically conceived of as written texts: there is no feature for interactive natural-language explanation. The texts may be several paragraphs long. The text in figure 1 is an annotation of the component "Host"; the graphical representation of the first level of the design of the Host is shown in figure 2. (This picture corresponds to the first of the two paragraphs of the text.) The text annotates the software design by describing its structure and interpreting it in terms of its security characteristics.

*The research reported in this paper was carried out while the author was at Odyssey Research Associates, Ithaca, NY. It was supported by the Air Force Systems Command at Rome Air Development Center under Contract No. F30602-85-C-0098

Structure of Joyce

Joyce consists of three separate modules, which perform distinct tasks and access their own knowledge bases.

1. The *text planner* produces a list of propositions, which represents both the content and the structure of the intended text. Thus, the task of Joyce's text planner is similar in definition to TEXT's (McKeown, 1985), but different from that of Penman (Hovy, 1988), which expects the content selection task to have already been performed. Each proposition is expressed in a language-independent, conceptual frame-like formalism. It encodes a minimal amount of information, but can be realized as an independent sentence if necessary. The text planner draws on domain communication knowledge expressed in a high-level schema language (see below).
2. The *sentence planner* takes the list of propositions and determines how to express them in natural language. This task includes choosing lexicalizations and a syntactic structure for each propositions, and assembling these lexico-syntactic structures into larger sentences. It draws on knowledge captured in the conceptual/Engish dictionary.
3. The *linguistic realizer* takes the syntactic structures and produces surface sentences. It draws on syntactic and morphological knowledge, expressed in the lexicon. The linguistic component is based on Meaning-Text Theory (Mel'čuk, 1988), and is a reimplementation in Lisp of Polguère's Prolog implementation (see (Iordanskaja *et al.*, 1988a; Iordanskaja *et al.*, 1988b)).

Usually, the task of text generation is subdivided into two subtasks (planning and realization), not three. However, there is a certain amount of disagreement about where the line between the two is to be drawn. For example, McKeown's TEXT (McKeown, 1985) and Rösner's SEMTEX (Rösner, 1987) seem to consider the tasks that Joyce classifies as sentence planning as part of the realization process, whereas Meteer's SPOKESMAN (Meteer, 1989) classifies them as part of text planning. The proposed finer-grained terminology may prove useful in discussing text generation and

HOST: General Structure and Security Features
The multilevel Host is a complex component of the Station. It contains a Kernel, a TIP, a Process, a Net Handler and a group of Managers. The Process, the TIP, the Managers and the Net Handler communicate only through the Kernel. The manifestly secure Process and the Managers perform auxiliary functions. The Process is low-level. The TIP serves as interface to a User; the Net Handler handles communication with a Net. The security statuses of the TIP, the Managers and the Net Handler have not yet been specified.
The Kernel is a complex component. Its security status has not yet been specified. The Kernel contains a Message Switch, an Adress Register and a Locator. The Adress Register, the Locator and the Message Switch communicate directly with each other. The low-level Adress Register and the multilevel Locator are databases. The Message Switch handles communication with the TIP, the Process, the Managers and the Net Handler. The security status of the Message Switch has not yet been specified.

Figure 1: The HOST Text

text generation systems by avoiding ambiguity. In this paper, "text planning" will always be used in this narrower sense.

The three modules have been implemented in a pipelined manner. Execution is interleaved temporally, so that surface text is produced shortly after the generation process is initiated. However, data need only flow in one direction; each module knows what information the next module requires.

Text Planning in Joyce: the Task

Since there was no corpus of texts available for analysis prior to the design of the text planning component, the first task of the design was to assemble such a corpus. Specialists in the design of secure software were asked to produce short descriptions of their designs, concentrating on the structure and security features of the designs. This exercise provided useful insight into the problem. In particular, it became obvious that a text planner, whether human or machine, would face the following problems:

- Even if virtually unlimited domain knowledge is available (i.e, human experts), it is impossible to translate this knowledge directly into the knowledge required for writing texts about that domain. How to write about a new domain must be learned. Typically, humans do this by repeatedly going through a cycle of text production, critique and revision.
- The underlying representation in Ulysses (its domain representation) is designed in a way best suited for the formal modeling and mathematical verification of security properties, rather than for the storage and retrieval of information. Therefore, the text planner must interpret the data in order to communicate it. It is not sufficient to simply retrieve data.
- The texts in the corpus have a clear rhetorical structure, but the relations that hold between the rhetorical blocks are not very varied: using the terminology of RST (Mann and Thompson, 1987), they are by and large restricted to the **elaborate**, **background** and **sequence** relationships. This rhetorical "flatness" effectively rules out an approach to planning these texts which is based only or even largely on rhetorical considerations.
- Since there are numerous objects in the domain, with an ever larger number of interconnections between them, "paths" through the domain representation cannot be relied on for providing the organizing principle for text planning: the text would become repetitive and long. Furthermore, the decision of which of the many possible paths to take would still remain open. A "procedural strategy" (Paris and McKeown, 1986) is not sufficient to plan the text.

The question, then, is: how can text be planned in those domains and applications in which previously proposed strategies seem to fail for reasons particular to the domain or application? (This is essentially the second of Hovy's "unsolved problems in the planning of paragraphs", (Hovy, 1989)).

Text Planning and Domain Communication Knowledge

Three Types of Knowledge

Recent research in text planning has stressed the importance for text planning of what may be called "communication knowledge", general and domain-independent knowledge about how to use language in order to achieve communicative goals. Communication knowledge includes rhetoric and knowledge about thematic[1] structure. Rhetoric relates complex goals of communication to other, more elementary goals of communication. Thematic knowledge relates the thematic function of sentence elements to the thematic function of elements in adjacent sentences. Communication knowledge is independent of any particular domain knowledge. However, between domain and communication knowledge one may identify a third type of knowledge, which I will call "domain communication knowledge". Domain communication knowledge relates domain knowledge to all aspects of verbal communication, including communicative goals and function. It is necessarily domain-dependent. However, it is not the same as domain knowledge; it is not needed to reason about

[1]The term "thematic" will refer, in this paper, to the communicative structure of a sentence, and will group together the phenomena that have been identified as topic/comment or theme/rheme.

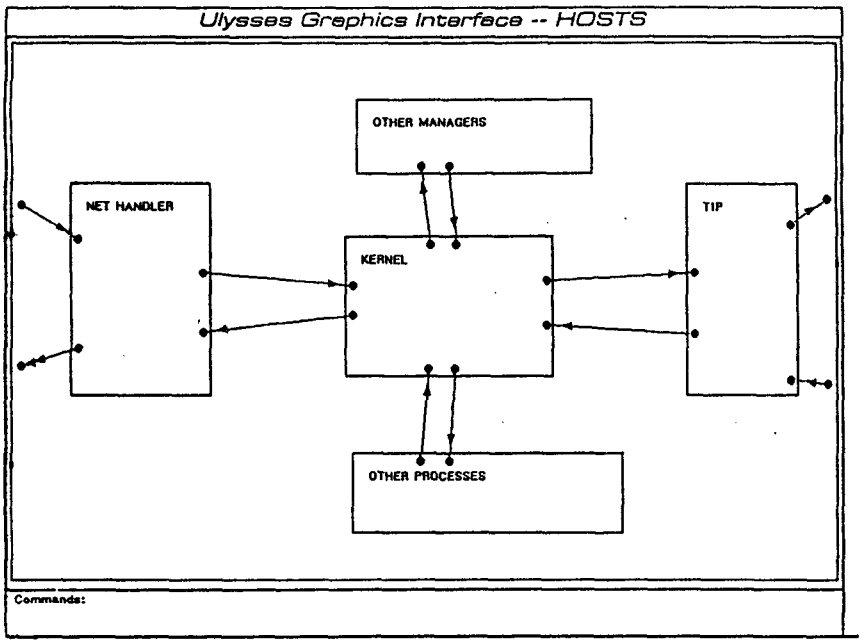

Figure 2: The HOST Graphical Representation

the domain, it is needed to *communicate* about the domain. For example, consider the task of describing some objects in some domain. Communication knowledge about thematic structure implies a strategy that describes those objects together that share some feature. Domain knowledge can supply information about which objects share which feature. But if there are many different features, the task remains of choosing the feature according to which the descriptions will be grouped together. This choice must be based on knowledge which is neither general knowledge about communication (since the choice depends on the particular features of objects in the domain), but it is not actual domain knowledge, either (since it is only needed for planning communiction).

What is the role of domain communication knowledge in text planning? Rather than trying avoid the stigma of domain-specifity, I propose to tackle the problem head-on, and posit the hypothesis that *all text planning tasks require domain communication knowledge*. The rest of this section will attempt to motivate this hypothesis by investigating two other text planning systems. The following section then discusses the representation of domain communication knowledge in Joyce.

Domain Communication Knowledge in Other Systems

Researchers whose principal interest lies in the representation and use of rhetorical or thematic knowledge have paid less attention to domain-specific problems in text planning. I will analyze two text planning systems in order to illustrate the hypothesis that any text planning task involves domain communication knowledge.

- McKeown's TEXT (McKeown, 1985) uses schemas to encode rhetorical knowledge[2]. These schemas "capture patterns of textual structure that are frequently used by a variety of people" (McKeown, 1985, p. 37). Each schema consists of a list of rhetorical predicates; the predicates can be matched against the domain representation and realized by a proposition, or they can be instantiated by another schema. The schemas/predicates represent basic rhetorical operations, such as **identification, compare and contrast** and **constituency**.

  The rhetorical schemas are domain-independent, but the text planner needs to relate them to a given domain in order to produce a text plan. This is where domain communication knowledge is needed. The domain communication knowledge is implicitly en-

[2]For reasons of space limitations, this analysis disregards TEXT's use of thematic knowledge. A more complete analysis would not significantly alter the conclusion.

coded in the "semantics" of the rhetorical predicates: "these are semantics in the sense that they define what a predicate *means* in a database system" (McKeown, 1985, p. 45). The semantics are implemented as functions that access the data base. The semantics are dependent on the structure of the data base, but independent of its domain. However, the domain of the texts is precisely the *structure* of the data base, not the *domain* of the data base. In this sense, the semantics are specific to the domain of the texts. (A similar analysis would apply to other interfaces to data bases that are based on principally on rhetoric, such as Maybury's JENNY (Maybury, 1989).)

By way of example, consider McKeown's discussion of TEXT's response to the user query "What is a ship?" (McKeown, 1985, p. 47). Using its rhetorical communication knowledge, TEXT decides that the first predicate to be matched is the **identification** predicate. Communication knowledge cannot, however, be used to interpret the domain representation in order to find appropriate information that might serve to identify a ship. Neither is domain knowledge sufficient: the domain representation encodes the facts that a ship is a water-going vehicle and that it travels on the surface, but it does not reveal that these two facts are exactly what is needed to identify a ship.

Thus the different types of knowledge have very different purposes in TEXT: rhetorical knowledge relates the discourse goal to a sequence of constituent communicative goals. Each communicative goal can in turn be related to another set of communicative goals. Once the recursive expansion of communicative goals comes to an end, domain communication knowledge (the semantics function) relates each communicative goal to domain knowledge and produces a proposition.

- TWRM-TOPOGRAPHIC, a system under development at the University of Constance (see (Sonnenberger, 1988)), produces natural language abstracts from texts. The generation component of TWRM-TOPOGRAPHIC uses a similar system architecture to TEXT, but relies on different knowledge. Its generation component, developped by Gabi Sonnenberger, uses an explicit encoding of thematic knowledge to guide the text planning process. The starting point of the abstracting and generation process is a representation of the contents of a text, the "text graph". It expresses the thematical relations between text spans. A discourse strategy is chosen on the basis of the thematic progression pattern of the text graph. The graph determines the sequence of propositions for realization.

  Thus, domain-independent knowledge about typical patterns of thematic progression guides the text planning process. Here, no semantics are needed since the information about thematic roles is already expressed by the text graph from which the discourse strategies draw their information. It is in constructing the text graph that domain communication knowledge is used so that the relevance of domain facts to the communicative process can be encoded. The use of domain communication knowledge is crucial to the text planning process, but remains implicit.

Making Domain Communication Knowledge Explicit

If the hypotheses that any text planning task requires domain communication knowledge is correct, then the question arises about how to represent and use such knowledge. Usually, domain communication knowledge is simply represented as LISP code (such as the semantics in TEXT), or implicitly encoded into the input to the text planner (such as the text graph in TWRM-TOPOGRAPHIC). Some other knowledge representation formalism is needed.

Domain Communication Knowledge Representation in Joyce

In choosing a formalism for representing domain communication knowledge, a fact becomes apparent: though certain types of texts (such as Russian folk tales) may be generated by a context-free formalism such as a "story grammar" (or even a "mildly context-sensitive" formalism such as a TAG), this is not true in general. A context-free formalism cannot capture in full generality the knowledge needed to plan texts. On the other hand, if domain communication knowledge is to be an interesting concept, then its representation must be restricted in some way; simply using, say, LISP code does not represent much insight into the process of text planning. A good approach may be, therefore, to choose a restricted formalism and then to enquire in what ways it needs to be expanded to allow for the types of operations that text planning requires.

These considerations have lead to the choice of domain-specific schemas as the basis for text planning in Joyce. These schemas are similar in form to those used by McKeown. Basically, a schema is an ordered set of instructions. The instructions can be calls to other schemas, recursive calls to the same schema, or they can produce a specific proposition and add it to the text representation. The schemas support conditional branching and iteration. In addition, two new instructions represent extensions to the context-free formalism.

- A portion of the text plan can be edited. To do this, a schema is called, but any propositions that are created (by the schema or by any schema it calls) are not added to the text representation. They are kept on a separate list in the order they are created. When the execution of the schema terminates, an editing function is applied to the list. This editing function can be a default function, or it can be explicitly named

in the call to the schema. It can delete propositions, change their order, change their contents or create new ones. The choice of an editing function depends on the domain and on the particular requirements of the text. This type of revision is different from the revision Meteer discusses in (Meteer, 1988), as it is situated at the text plan level, rather than the sentence level. The same remark holds for Gabriel's Yh system (Gabriel, 1988). Its complex architecture executes a cycle of text production, observation by "experts" and text modification. However, these experts are all situated at the sentence planning level; the text plan is produced by domain-specific text planners (such as an "array expert"), and is not itself subject to revision.

- Schemas can post to a "blackboard", and check this blackboard for messages. This allows for additional control and communication between schemas which are called at different times during the text planning process and cannot communicate with each other by passing arguments.

Instead of being templates that limit the structure of the text to certain preconceived types, the schemas are now an explicit and compact representation of domain communication knowledge.

Example

Consider the first paragraph of the sample text in figure 1. It describes the component "Host". A rhetorical schema might specify a sequence of the **identification** predicate (the first sentence), the **constituency** predicate (the second and third sentences) and several **amplification** predicates (the remaining four sentences). This analysis shows that the resulting text has a well-formed rhetorical structure. However, this analysis in itself does not constitute a text plan, since the text planner must know how to relate the rhetorical predicates to the domain knowledge, i.e., how to choose information from the domain representation to realize each predicate. The system maintains many different kinds of information about each component: its name; its parent, sibling, and subcomponents in the hierarchical model; its ports; its security level; its secutiy status; the location and size of its icon on the screen; various information relating to the formal specification of the component. Choosing different types of information to realize each predicate will result in very different texts:

- For example, the **identification** could be accomplished by reporting the name of a comonent and its location on the screen, or by describing its connectivity with its sibling components.
- In order to describe the **constituency** of a component, the text planner will most likely report the subcomponents of a component, but it could also dicuss its ports. It may or may not discuss the way the subcomponents are connected (their connectivity).

defschema describe-complex-component
1. **condition** not-atomic-component?
2. **local-variable** relevant-components
   are (get-relevant-components self)
3. **theme** self
4. **exclusive-choice** toplevel-id *or* lower-id
5. contains
6. **exclusive-choice** star-connectivity
   or complete-connectivity
   or default-connectivity
7. **exclusive-choice** enforced-security *or* not-secure
8. **edit-for-these-objects**
   objects (exclude-from coms relevant-components)
   schema security-functional-features
   edit-function join-same-concepts
9. force-paragraph
10. **for-these-objects** relevant-components: general

Figure 3: The DESCRIBE-COMPLEX-COMPONENT Schema

- The notion of **amplification** is so general as to allow any of the information in the domain representation to realize the rhetorical predicate.

The domain communication knowledge needed to make these decisions is explicitly represented in Joyce by schemas. An example of such a schema is found in figure 3. It provides the basic framework for the first paragraph of the HOST text.

In this schema, each numbered line represents an instruction. Executing every instruction in turn will accomplish the goal associated with the schema, namely to describe a comlpex component. Special operators are represented in boldface. Words in Roman font represent either calls other schemas, or variables or functions. Words and numbers in italics are comments. (The actual lisp-based representation of this schema in Joyce contains some additional parentheses, of course.)

This particular schema is only applicable if the component to which its is applied is not atomic. The Host meets this condition (line 1). Line 2 sets up a local variable, and line 3 defines the theme (topic) of the paragraph to be the Host. Line 4 identifies the component under discussion: the particular choice of how to identify a component depends on whether it is the top-level component, or whether it has a parent component. Since the Host does have a parent component, schema **lower-id** (not shown here) is chosen. Its execution generates three propositions, identifying the Host by its parent component, its complexity and its security level. The sentence planner merges these propositions into the first sentence.

In order to describe the constituents of the component under discussion, Joyce first calls the **contains** schema which lists the subcomponents of the Host (line 5). It then describes the connectivity of the subcompo-

nent. The simple solution would be to follow a procedural strategy and to list all connections between subcomponents (*The Net Handler communicates with the Kernel. The Kernel communicates with the Managers. ...*). However, for certain types of connectivity there are descriptive shortcuts: for example, all subcomponents may communicate with all other subcomponents, or all subcomponents may communicate with exactly one central subcomponent (as is the case of the Host in the sample paragraph). In fact, if no descriptive shortcut is available, it turns out that the resulting text is so cumbersome that it is better not to describe the connectivity at all (the user may consult the graphical representation). The text planner must be able to identify these special cases of connectivity and choose the applicable descriptive strategy, or initiate the default strategy if no shortcuts are available (line 6).

Joyce amplifies on the previously given information by giving some additional information about the Host (line 7; no additional security information is available for the Host, so no proposition is generated), and about each of its subcomponents. Joyce has already determined that one component, the Kernel, contains subcomponents of its own (and is thus a "relevant" component – line 2). The second paragraph of the HOST text is devoted to it, so it is not discussed in the paragraph about the Host. For the remaining subcomponents, Joyce decides to give a brief description of their function and of their security features. However, Joyce must also decide on how to order the information. No order can be specified on rhetorical grounds, since no component is more salient than the others. Joyce tries to group together those components that perform similar functions, thus avoiding the need to repeat functional descriptions: *The (...) Process and the Managers perform auxiliary functions.* This is encoded in the **edit-for-these-objects** instruction of the schema (line 8). It calls schema **security-functional-features** on the reduced set of subcomponents (without the Kernel). Instead of sending propositions to the sentence planner as they are generated by schema **security-functional-features**, the propositions are saved until comlpeteion of the iteration. Then editing function *join-same-concepts* is applied, which rearranges the propositions, and they are then sent to the sentence planner. The sentence planner can exploit the adjacency of similar propositions, and forms single sentences.

The paragraph is then completed (line 9) and the general text planner called recursively on the Kernel (line 10).

Domain Communication Knowledge and "Planning From First Principles"

If the thesis that all text planning tasks require domain communication knowledge is correct, then it would appear that "planning from first principles", i.e. text planning using only general knowledge about communication and knowledge about the domain (and a reader model), would be impossible. This conclusion is nonsensical: clearly, human text planners succeed in writing about new domains. But if text planning from first principles is possible, as anecdotal evidence suggests, then what is the status of domain communication knowledge?

Consider the following approach to text planning: Supposing that for any given domain only a finite number of facts is known, one could list all possible orderings of the elements of all subsets of these facts and use pure communication knowledge along with a reader model to evaluate each of these text plans. The communication knowledge would serve as a constraint on possible text plans, or as an evaluation metric for the quality of a text plan (the fewer violations of communicative principles it exhibits, the better). This brute-force bottom-up approach can be refined by using general communication knowledge to do some preliminary top-down planning. For example, a rhetorical predicate can refine a high-level communicative goal into a set of more easily achievable goals, as in TEXT. Nonetheless, even in this approach it is necessary to form all possible sequences of relevant domain information that together may achieve a lower-levl goal and perform extensive reasoning based on communication knowledge and the user model.

These approaches thus represent true text planning from first principles, since they do not require domain communication knowledge. But they require reasoning about an exponential number of possible text sequences. They are computationally unattractive for any true applications. In order to make text planning more efficient, domain communication knowledge is required. Thus, costly text planning from first principles can be understood as the process of acquiring or compiling domain communication knowledge. Ways in which domain facts can contribute to achieving a particular communicative goal are deduced by a complex process using only domain and communication knowledge, but once such reasoning has been performed, its results are explicitly encoded in the domain communication knowledge representation. This knowledge can then be used for subsequent text planning tasks. Intuitively, this explains why humans get better and faster at generating a given type of text about a specific domain once they have done so several times.

This view is analogous to Patten's proposal (Patten, 1988) to precompile communication planning knowledge according to the theory of register. However, his proposal is aimed at what in Joyce is classified as sentence planning knowledge, since register-related decisions are made by the sentence-planner in Joyce. Domain communication knowledge might be considered a precompilation of genre knowledge.

Further Research

A fundamental issue remains as an open research question: what is the mode of interaction between domain communication knowledge and general communication knowledge? Three views are possible:

1. Does text planning start with general communication knowledge, with the domain-specific knowledge taking over in order to relate "atomic" communicative goals to domain facts (as in TEXT)?
2. Is there a continuum from general domain-independent knowledge via specific but domain-independent knowledge to domain-dependent communication knowledge (Moore and Paris's **Motivation** plan operator for motivating replacements (Moore and Paris, 1989) is an example of a specific but domain-independent communication strategy)? This would suggest a multi-layered representation of these different types of communication knowledge.
3. Is general communication knowledge implicitly encoded into the domain communication knowledge (as in Joyce)? In this view, general communication knowledge would be used explictly only during the process of compiling domain communication knowledge.

This underlying issue affects the more particular questions that the proposed representation formalism for domain communication knowledge raises:

1. Is a schema-based approach an adequate representation for domain communication knowledge?
2. What types of editing functions are needed?
3. How do genre and user-tailoring affect domain communication knowledge?

Acknowledgments

I would like to thank Robert Dale, Richard Kittredge, Tanya Korelsky, Libby Levison and Bonnie Webber for helpful comments and discussions about various drafts of this paper.

Bibliography

Richard P. Gabriel. Deliberate writing. In David D. McDonald and Leonard Bolc, editors, *Natural Language Generation Systems*, pages 1–46. Springer Verlag, 1988.

Eduard H. Hovy. Planning coherent multisentential text. In *Proceedings of the 26th Annual Meeting*, pages 163–169, Buffalo, 1988. ACL.

Eduard H. Hovy. Some unsolved problems in the planning of paragraphs. In *Extended Abstracts Presented at the Second European Natural Langugae Generation Workshop*, Edinburgh, 1989.

Lidija Iordanskaja, Richard Kittredge, and Alain Polguère. Implementing the meaning-text model for language generation. Paper presented at COLING-88, 1988.

Lidija Iordanskaja, Richard Kittredge, and Alain Polguère. Lexical selection and paraphrase in a meaning-text generation model. Technical report, Odyssey Research Associates, 1988.

Tatiana Korelsky and Ulysses Staff. Ulysses: a computer security modeling environment. In *Proceedings of the 14th National Conference on Security and Privacy*, Baltimore, 1988. NBS.

William C. Mann and Sandra A. Thompson. Rhetorical structure theory: A theory of text organization. Technical report, ISI, 1987.

Mark T. Maybury. Knowledge based text generation. Technical report, RADC, 1989.

Kathleen McKeown. *Text Generation.* Cambridge University Press, Cambridge, 1985.

Igor A. Mel'čuk. *Dependency Syntax: Theory and Practice.* State University of New York Press, New York, 1988.

Marie W. Meteer. The implication of revisions for natural language generation. In *Proceedings of the Fourth International Workshop on Natural Language Generation*, Catalina Island, 1988.

Marie W. Meteer. The spokesman natural language generation system. Technical report, BBN Systems and Technologies Corporation, 1989.

Johanna D. Moore and Cécile L. Paris. Planning text for advisory dialogues. In *Proceedings of the 27th Annual Meeting*, Vancouver, 1989. ACL.

Cecile L. Paris and Kathleen R. McKeown. Discourse strategies for describing complex physical objects. In Gerard Kempen, editor, *Natural Language Generation*, pages 97–115. Martinus Nijhoff Publishers, 1986.

Terry Patten. Compiling the interface between text planning and realization. In *Proceedings of the AAAI Workshop on Text Planning and Realization*, St.Paul, 1988.

David Rosenthal, Tatiana Korelsky, Daryl McCullogh, Owen Rambow, and D.G.Weber. The Ulysses integrated modeling environment and its relationship to KBSA. *Heuristics*, 1(2):42–49, 1988.

Dietmar Rösner. The automated news agency SEMTEX – a text generator for German. In G. Kempen, editor, *Natural Language Generation: New Results in Artificial Intelligence, Psychology and Linguistics*, pages 138–148. Kluwer Academic Publishers, Boston, 1987.

Gabi Sonnenberger. Flexible Generierung von natürlichsprachigen Abstracts aus Textrepräsentationsstrukturen. In *4. Österreichische Artificial Intelligence Tagung: Proceedings*, pages 72–82. Springer, Berlin, 1988.