

Chapter 6

Using an Independent, Belief-Based Agent Modelling Component for Language Generation

A Feasibility Study

KALINA BONTCHEVA

ABSTRACT. This paper generalises previous work in order to establish desiderata for an independent user modelling component to be used for language generation. Within this context, we discuss the suitability of ViewGen, a powerful belief modelling system and its advantages for generation. We then conclude by outlining some future work and problems which might arise.

1 Introduction

The field of Natural Language Generation (NLG) studies the automatic production of coherent texts from some internal representation. Often these texts are made sensitive to the context and the target audience, e.g., explanations for novices should contain different information from explanations produced for experts. In this paper we will focus mostly on what features of the user have been needed by previous NLG systems and what other related problems have been tackled (Section 2). Then, we will use this as a basis for establishing the desiderata for a specialised belief modelling component if it is to be used for language generation. Based on that, an overview of ViewGen, a particular belief modelling system, will be given (Section 3). Section 4 then discusses how such a model can influence the generation process and what are the advantages offered by ViewGen in that respect.

2 Previous Work

This section aims at summarising the type of information which is encoded in user models (UM) of existing NLG systems; how it is acquired and how it

is updated and maintained. To this end, we studied the approaches taken in the following systems: PAULINE [Hovy 88], TAILOR [Paris 93], EPICURE [Dale 92], FN [Reiter 90], KAMP [Appelt 85], ADVISOR [McKeown *et al.* 85], ROMPER [McCoy 88], PEA [Moore 95], EDGE [Cawsey 92], and WISHFUL [Zuckerman & McConachy 93].

2.1 What is modelled?

The different NLG systems need a versatile user-related information in order to provide tailored utterances. The main differences seem to come from:

- system task (e.g., advice-giving, tutoring) and domain (e.g., programming, cooking);
- means of communication with the user – mainly interactivity and modality;
- generation issues addressed – focus on a (few) generation subtasks (e.g., referring expression generation) or a whole system;
- methods and approaches – task-specific and/or domain-dependent solutions vs. more general and portable ones.

Modelling knowledge and belief

Many systems have attempted to track user's *knowledge level* (e.g., PAULINE, TAILOR, EDGE (even updated dynamically)) and *known concepts and facts* – ROMPER, PEA, TAILOR, WISHFUL (concepts and *relations*, including origin of knowledge), KAMP. This also includes *domain taxonomy* (EPICURE, ROMPER); *instances of topics and subtopics* (EDGE (corresponding to instantiated content planning rules)); *mutual beliefs* (KAMP); *beliefs about other agent's beliefs and goals* (KAMP).

In addition, PAULINE explores the influence of pragmatic information on generation decisions by discrete modelling of user's social and personal aspects such as *interest*, *opinion* and *emotional state*.

Another important aspect of user's knowledge concerns *language use* (PAULINE). Also, as modelled in the FN system, some domain concepts may lack corresponding lexical realisations – a distinction which might be needed within a multilingual setting too.

Goals and plans

Further constraints on tailoring and interpretation of user's responses might come from user's *goals and plans*. There are various types of goals some of which are *social* (e.g., interpersonal PAULINE) while others are *domain- and task-related* – ADVISOR (goals inferred from discourse and plans). Some of these goals may not even have corresponding immediate plans, e.g., *high-level goals* in PEA. Related to that is modelling of user's knowledge of *methods* for achieving goals and performing acts (PEA, EPICURE).

Still another distinction that needs to be made is between *short-term* and *long-term* goals (PAULINE). The former can be achieved and discarded while the latter are never fully achieved and influence various generation decisions without having respective plans.

Separation of models

An important problem which often arises especially in tutoring and advisory systems is allowing for *differences* between the knowledge of the different agents (usually system and user). These differences can be in *user characteristics* (PAULINE); *domain knowledge* (ROMPER); and even *domain ontology or taxonomy* (ROMPER). This separation also requires special reasoning mechanisms which can handle conflict resolution.

2.2 Initial Acquisition

The initialisation of the user models very often relies on stereotypical information which is ascribed to the particular user as soon as appropriate stereotype is detected (some systems allow more than one stereotype to hold) (e.g., PEA, EDGE). Other sources of information come from *observing the user* prior to and during the interaction. For instance, PEA analyses user's code, initial request and explicitly stated high-level goals.

2.3 Update and Maintenance

Many NLG systems rely on other existing components to update and maintain the user model (e.g., an understanding component), so they are not concerned with truth maintenance and conflict resolution. Nevertheless, some do apply certain techniques to infer user-related information from the ongoing interaction. Some of the solutions are tied to the particular domain or task at hand, while others rely mostly on natural language communication.

The most simple techniques *update* user's domain knowledge by adding all newly introduced concepts (e.g., TAILOR) or at least accumulate past exchanges in a *dialogue history* (e.g., PEA). Further updates may come from the effects of realised speech acts and user's answers to explicit questions (e.g., EDGE).

2.4 Summary

To summarise, many different types of user information have been represented in various NLG systems. However, none of these systems has attempted to use a comprehensive user modelling approach that can handle *all* these phenomena in a computationally feasible manner. Moreover, each system has developed its own user model with its own representation and expressiveness. The most common representation is an em overlay model where facts from the generator's knowledge base can be marked as known by the user. Often the generation systems model and use only limited information which is sufficient for the task at hand. In general, this could lead to problems with scalability and portability.

In our view, what is needed is an independent, powerful agent modelling component which can be used to support both understanding and generation. The advantage of using such a component is that potentially it could be reused between application domains and also between generators. In addition, having the user model as an independent, general purpose component in a NLG system could facilitate the evaluation of its influence on the various generation tasks. Therefore, we will now examine an existing agent modelling approach (namely ViewGen) which we intend to integrate into a language generation system.

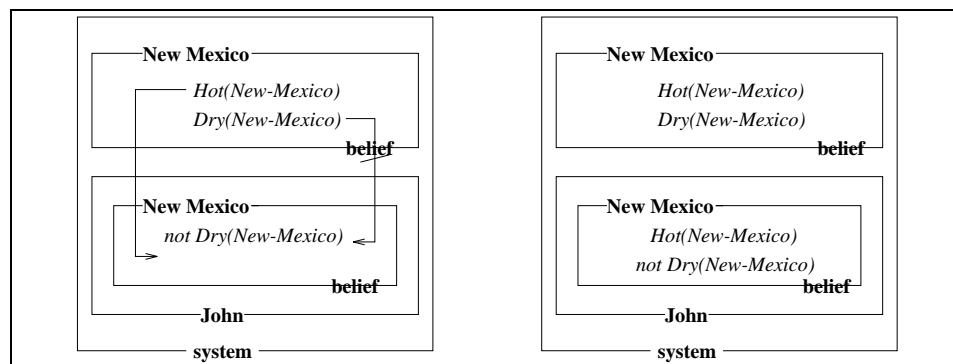


Figure 6.1: Examples of ViewGen belief environments: before ascription (left) and after ascription (right)

3 Modelling Agents Attitudes – ViewGen

ViewGen [Ballim & Wilks 91a, Lee & Wilks 96, Lee & Wilks 97] is an agent modelling system which represents beliefs, intentions and goals of dialogue participants. It represents such attitudes in *nested environments* (see Figure 6.1). There are two types of environments – *viewpoints*, which represent someone’s beliefs, goals or intentions (e.g., John’s beliefs) and *topic* environments, which contain propositions about this topic (e.g., facts about New Mexico)¹. Rather than model these statically, ViewGen constructs environments dynamically by a process of *ascription*.

Ascription assumes that attitudes held by one agent can be ascribed to others. There are two main methods of ascription – *default* ascription and *stereotypical* ascription [Ballim & Wilks 91b]. Default ascription applies to common attitudes which ViewGen assumes that any agent will hold and also ascribe to any other agent *unless there is contrary evidence* (see Figure 6.1). Stereotypical ascription applies to ‘uncommon’ attitudes which ViewGen assumes hold only for agents from a particular stereotype.

Both methods are examples of default reasoning and some mechanism is required for dealing with conflicting ascriptions. To handle such problems, ViewGen relies on a truth maintenance system (TMS) [Doyle 79]. Since ascriptions are heuristic, ViewGen has to be capable of updating its model of the attitudes of the user in the light of newly acquired information. Consequently, the TMS is also used for belief revision.

In brief, ViewGen has all the features which might be required by a language generation system (see previous section) since it:

- can represent beliefs, goals and plans of an agent;
- attitudes are grouped by topic into separate environments which can help reducing the search for relevant information;
- supports nested models which allow representation of conflicting beliefs between agents;
- utilises stereotypes and default ascription to infer additional information;
- resolves conflicts with the help of a truth-maintenance system.

Therefore, we intend to integrate ViewGen into a prototype language generation system in order to assess its suitability as an agent modelling component and identify potential problems of the approach.

¹For a description of the representation of goals and intentions, as well as the planning performed by ViewGen see [Lee & Wilks 96]

4 Generating User-Sensitive Explanations

As it has been already proven in previous research on language generation (e.g., [Paris 93]), user-related information could be used to constrain generator's decisions and to improve the fluency and tailoring of the generated text. Such an information is useful at any stage of the generation process:

- *selecting relevant knowledge* – enables explanations² with variable detail and information content which also take into account previous explanations and possible user plans and goals;
- *text organisation* – the text organisation strategies or plan operators could have preconditions dependent on user information.
- *surface realisation* – use of words which are known to the user.

In order to restrict the issues down to a manageable task, we intend to study mostly how the agent models in ViewGen could support content selection and text organisation. The resulting language explanations will be generated from that content by an existing realisation component [Bontcheva 97].

4.1 System Architecture

The tentative domain of the system is explanations about computers however, this will have to be restricted further when the experimental knowledge base (KB) is being acquired.

In order to make the system output closer to real texts, a *corpus* of target explanations is being collected and analysed. This technique has been used in other language generation systems (see [Reiter & Dale 97]) to help the identification of the necessary *types of domain knowledge* and re-occurring *text structuring patterns*.

The currently designed architecture of the language generator is shown in Figure 6.2. Basically, it is a pipeline model (see [Reiter 94]), with separate *content planning* and *surface realisation* modules. The user model will be taken into account only when selecting relevant content and organising it into a structured explanation. As demonstrated in other generation systems (see Section 2), a user model could also influence lexical choice but this will not be considered in this work³.

²Here we will discuss explanations since this is broadly the generation task we are mostly interested in. However, these points are also valid for other generation tasks such as tutorial dialogues.

³Therefore the dotted link between VIEWGEN and the surface realisation module (Figure 6.2) encodes that the information from the user model could be used *in general* by the surface realiser as well.

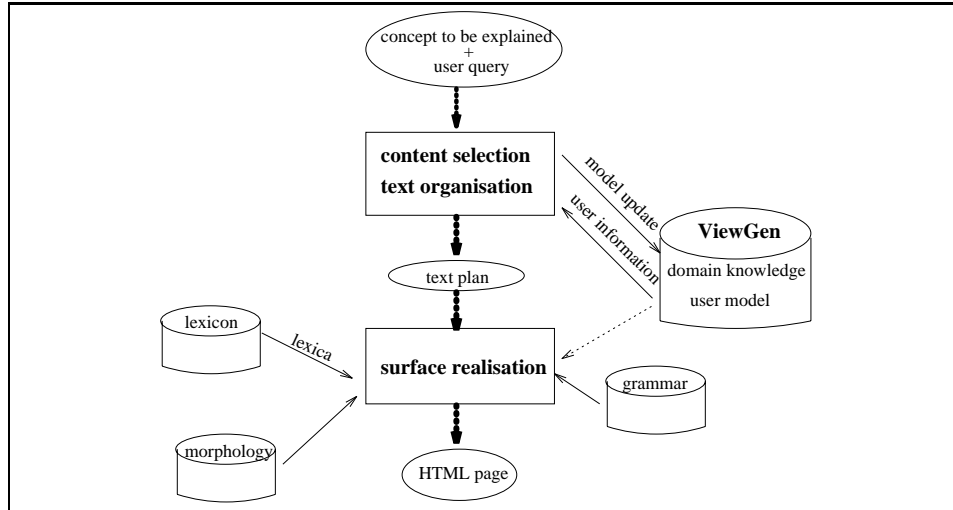


Figure 6.2: System Architecture

4.2 The role of ViewGen

An important advantage of ViewGen is its account for common knowledge (i.e., default ascription) and stereotypes, as this will allow the modelling of different classes of users and tasks and the use of the most relevant one(s). Also, to maintain efficiency, topic environments and views of each agent are built on demand. Stereotypes relevant to an agent are not applied just once in the beginning of the interaction but are *consulted* wrt a specific belief or topic only when needed. This allows flexibility in case of conflicting information, i.e., the environment from one stereotype could be preferred for a given topic, while another stereotype could be preferred for the next.

In addition, the topic environments will restrict the search for relevant information, i.e., they can act as natural *relevant knowledge pools* [McKeown 85] from where the text planning process will decide which information to include in the explanation.

An essential aspect of ViewGen's paradigm is having a main system's environment which contains environments for system's and agents' attitudes. The consequence of that for the generator's resources is that ViewGen will contain both the domain knowledge base and the user model, while they are usually differentiated in other generation approaches. However we take this to be an advantage since it facilitates the representation and maintenance of conflicting user and system knowledge. Another mostly implementational benefit is the unified interface to the agent modelling component.

The ViewGen agent model will be *updated* after each generated explanation. Since we intend to implement the explanations as hypertext, the user feedback and requests will be always expressed as following this link or another. Since links are not always sufficient to provide the user with effective means of obtaining information, we also plan a context-sensitive questions menu.

ViewGen could be also extended to model users' interests as well. A relatively reliable source of such information are the bookmarks used by the users during hypertext browsing. Therefore, bookmarks, if present, could be used to initialise the user model in subsequent interactions (see also [Bontcheva & Wilks 97]).

Using and maintaining the consistency of nested belief models could be sometimes computationally expensive. However, ViewGen's topics and default ascription limit the amount of necessary inference and keep the nestings down to the minimal possible level, which results in a computationally efficient system. Nevertheless, whether this will be sufficient for real-time generation remains to be proven in our future work. Yet, the experiments of Kobsa et al. to use a similar user modelling shell system (BGP-MS) for adapting existing hypertext have been successful [Kobsa *et al.* 97].

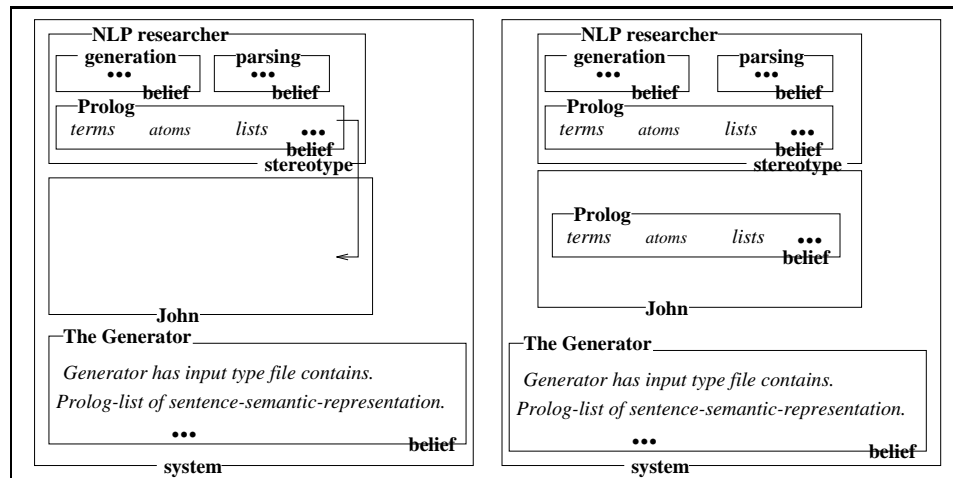


Figure 6.3: An Example: before the stereotype being considered (left) and afterwards (right)

4.3 An Example

In order to exemplify the use of ViewGen let us consider the following situation. A user (John) has requested information about the generator (if we suppose that the system domain is explanations about the system itself).

ViewGen initially (see Figure 6.3 (left)) has a stereotype of NLP researchers, has beliefs about the generator, ‘knows’ nothing about John except that he is a NLP researcher (not shown on the figure).

When the generation system has to decide what is the relevant content, it requests ViewGen what John believes about the generator (let’s call it EGEN for simplicity). ViewGen detects that it has no beliefs about John so it responds that John does not know anything about EGEN, since John’s environment and all stereotypes applicable to John do not contain such information. Then the generation system decides that the content of **The Generator** environment should be explained.

Afterwards, by looking at the facts into this environment, e.g., the fact about the generator’s input⁴, it notices that this fact mentions a *Prolog list*. Therefore, facts about Prolog could also be considered relevant and the generation system has to decide whether to include such facts in the content of the future explanation. In order to determine John’s knowledge of Prolog, it checks again ViewGen which ascribes this time all the beliefs of NLP researchers about Prolog as John’s beliefs, since the stereotype applies to John (see Figure 6.3 (right)). ViewGen also records this assumption using the truth maintenance system so that it could be retracted later if necessary.

Consequently the generation system obtains the environment containing John’s likely beliefs about Prolog and discovers that John is familiar with Prolog lists, so the Prolog environment is in fact not relevant. In this way, the generation system can check the user’s model for each concept if needed.

Now, if we assume that John is familiar with all ‘external’ concepts occurring within **The Generator** environment, the result of the content selection will be only the facts from the environment itself. If there were unfamiliar concepts, relevant information from their environments will be extracted and added to the content selection. Two possible texts produced respectively for the example above and for a novice user without knowledge of Prolog follow:

The generator expects its input as a file containing a Prolog-readable list of the semantic representation of the sentence. The format of the semantic representation...

The generator expects its input as a file which contains a list of Prolog compound terms. Compound terms, e.g., `name(e1, 'John')`, have a functor (`name`) and a sequence of terms called arguments (`e1, 'John'`). All arguments should start with a small letter or be enclosed in quotes. The format of the semantic representation...

One of the advantages with ViewGen is that if it becomes apparent from the ongoing interaction that the user does not actually have the assumed beliefs or goals, they can be retracted together with all inferences based on

⁴In order to avoid going into knowledge representation details, the fact is encoded in semi-natural language.

them. So in the example above, the beliefs about Prolog lists ascribed to John from the stereotype can be removed successfully.

4.4 Implementation and Evaluation

The system as shown in Figure 6.2 is currently under implementation. There is an early version of the generator in Sicstus Prolog, which runs on a small knowledge base in the domain of admixture separation. The text planner is schema-based (see [McKeown 85]) while the surface realiser is the one described in [Bontcheva 97], modified for English. The generated explanations are paragraph-sized and contain hypertext links to related concepts. The current generator does not use any sort of a user model, neither it takes into account previous utterances. The intention is to use this as a baseline system against which to compare ViewGen's influence and performance. In addition, this incremental approach will prevent the generator from relying too much on the user model, since, as argued by [Sparck Jones 91], tailoring is useful but the NLG systems should be able to produce acceptable texts even without such a model.

The approach could be extended further to take into account discourse structure. In order to be capable of such a reasoning, the generator will need an explicit representation of the discourse relations which is now impossible under the schema-based approach. However, we plan moving towards text planning based on the Rhetorical Structure Theory (RST) [Mann & Thompson 87].

Previous attempts at *evaluating* language generation systems (e.g., [Mellish & Dale 98]) have shown that this is quite a difficult task and the results seldom measure the performance and contribution of *each* module within the system. A measure of the overall performance of the system could be obtained by using a black-box evaluation measuring the 'quality' of the generated hypertext based on e.g., users' performance given a certain task (see [Reiter *et al.* 92]). However, since we are mostly interested in the role of the user model, we intend to run the generation system on a number of queries both *with* and *without* ViewGen and then compare execution time and output quality.

5 Conclusion

In this paper we have identified a number of user modelling features which are needed by NLG systems in order to enable the production of tailored texts. However, this is not to say that *all* these features are *always* needed – [Cahour & Paris 91] establish what kind of user-related information is

needed depending on the task to be performed (e.g. tutoring). Yet, the experiments with adapting hypertext and hypermedia based on BGP-MS [Kobsa *et al.* 97] have shown that powerful, generic user modelling shells could be used successfully even for simpler tasks thus enabling re-use, scalability and distributed organisation.

We have also argued that a language generation system can benefit from the adoption of a separate, specialised agent modelling component that can support all these features. On behalf of the generation system, this will facilitate reuse across domains and applications, whereas on behalf of the modelling system, this could be a testbed for the efficiency and scalability of the approach. ViewGen is such a model which has already been used successfully to deal with a large set of phenomena, e.g., metaphor [Ballim & Wilks 91a], conversational implicature [Lee & Wilks 97] and reference resolution [Wilks & By]. Therefore, we are now building a system that integrates ViewGen as a user modelling component for language generation.

Acknowledgements

This work has been supported by the European Community Grant LE1-2238 (AVENTINUS project) and an Overseas Research Students award ORS/97036019.

I would also like to thank Yorick Wilks for the discussions and useful comments and also the anonymous referees for the detailed feedback. I'm also indebted to all people involved in the development of ViewGen for helping me with the system and its implementation.

Bibliography

- [Appelt 85] D. E. Appelt. *Planning English Sentences*. Cambridge University Press, Cambridge, England, 1985.
- [Ballim & Wilks 91a] A. Ballim and Y. Wilks. *Artificial Believers*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1991.
- [Ballim & Wilks 91b] A. Ballim and Y. Wilks. Beliefs, stereotypes and dynamic agent modelling. *User Modelling and User-Adapted Interaction*, 1:33 – 65, 1991.
- [Bontcheva & Wilks 97] K. Bontcheva and Y. Wilks. Combining language generation and belief modelling into a flexible hypertext system. In *Proceedings of the Flexible Hypertext Workshop*, Southampton, UK, April 1997. A workshop held in conjunction with the Eight ACM International Hypertext Conference (Hypertext'97).
- [Bontcheva 97] K. Bontcheva. Generation of Multilingual Explanations from Conceptual Graphs. In R. Mitkov and N. Nicolov, editors, *Recent Advances in Natural Language Processing: Selected Papers from RANLP'95*, volume 136 of *Current Issues in Linguistic Theory (CILT)*. John Benjamins, Amsterdam/Philadelphia, 1997.

- [Cahour & Paris 91] B. Cahour and C. Paris. Role and use of user models. Technical report, ISI/RR-91-323, Information Sciences Institute, 1991.
- [Cawsey 92] A. Cawsey. *Explanation and interaction: the computer generation of explanatory dialogues*. Cambridge, Mass.; London:MIT Press, 1992.
- [Dale 92] R. Dale. *Generating referring expressions constructing descriptions in a domain of objects and processes*. MIT, Cambridge, MA, 1992.
- [Doyle 79] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231 – 272, 1979.
- [Hovy 88] E. H. Hovy. *Generating natural language under pragmatic constraints*. Lawrence Erlbaum, Hillsdale, New Jersey, 1988.
- [Kobsa *et al.* 97] A. Kobsa, A. Nill, and J. Fink. Hypertext and Hypermedia Clients of the User Modelling System BGP-MS. In M. Maybury (ed.) *Intelligent Multimedia Information Retrieval*, MIT Press, 1997.
- [Lee & Wilks 96] M. Lee and Y. Wilks. An ascription-based approach to Speech Acts. In *Proceedings of the 16th Conference on Computational Linguistics (COLING-96)*, Copenhagen, Denmark, 1996.
- [Lee & Wilks 97] M. Lee and Y. Wilks. Eliminating deceptions and mistaken belief to infer conversational implicature. In *IJCAI-97 workshop on Conflict, Cooperation and Collaboration in Dialogue Systems*, 1997.
- [Mann & Thompson 87] W. C. Mann and S. A. Thompson. Rhetorical structure theory: description and construction of text structures. In G. Kempen, editor, *Natural Language Generation: Recent Advances in Artificial Intelligence, Psychology, and Linguistics*, pages 85 – 96. Kluwer Academic Publishers, Boston/Dordrecht, 1987.
- [McCoy 88] K. F. McCoy. Reasoning on a dynamically highlighted user model to respond to misconceptions. *Computational Linguistics*, 14 (3), September 1988.
- [McKeown 85] K. R. McKeown. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press, Cambridge, England, 1985.
- [McKeown *et al.* 85] K. R. McKeown, M. Wish, and K. Matthews. Tailoring explanations for the user. In *International Joint Conference on Artificial Intelligence*, Los Angeles, CA, 1985. International Joint Conference on Artificial Intelligence.
- [Mellish & Dale 98] C. Mellish and R. Dale. Evaluation in the context of natural language generation. To appear in the *Journal of Computer Speech and Language*, 1998.
- [Moore 95] J. Moore. *Participating in Explanatory Dialogues*. MIT Press, Cambridge, MA, 1995.
- [Paris 93] C. L. Paris. *User modelling in text generation*. Francis Pinter Publishers, London, 1993.
- [Reiter & Dale 97] E. Reiter and R. Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.

- [Reiter 90] E. Reiter. Generating descriptions that exploit a user's domain knowledge. In R. Dale, C. Mellish, and M. Zock, editors, *Current Research in Natural Language Generation*. Academic Press, London, 1990.
- [Reiter 94] E. Reiter. Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible? In *Proceedings of 7th Int. Workshop on NL Generation (INLG-94)*, pages 163–170, Kennebunkport, Maine, USA, 1994.
- [Reiter *et al.* 92] E. Reiter, C. Mellish, and J. Levine. Automatic Generation of On-Line Documentation in the IDAS Project. In *Proceedings of 3rd Conference on Applied NL Processing*, Trento, Italy, 1992.
- [Sparck Jones 91] K. Sparck Jones. Tailoring output to the user: What does user modelling in generation mean? In C. L. Paris, W. R. Swartout, and W. C. Mann, editors, *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pages 201–226. Kluwer Academic Publishers, Boston, 1991.
- [Wilks & By] Y. Wilks and T. By. Protocols for reference sharing in a belief ascription model of communication. To appear in *Cognitive Science*.
- [Zuckerman & McConachy 93] I. Zuckerman and R. McConachy. Generating concise discourse that addresses a user's inferences. In *Proceedings of IJCAI'93*, 1993.