Architectures for Natural Language Generation: Problems and Perspectives

Koenraad De Smedt[1], Helmut Horacek[2], and Michael Zock[3]

[1] Institutt for Fonetikk og Lingvistikk
Universitetet i Bergen
Sydnesplass 9
N–5007 Bergen, Norge

[2] Fakultät für Linguistik und Literaturwissenschaft
Universität Bielefeld
Postfach 100131
D–33501 Bielefeld, Deutschland

[3] Langage & Cognition
LIMSI-CNRS
B.P. 133
F–91403 Orsay CEDEX, France

Abstract

Current research in natural language generation is situated in a computational linguistics tradition that was founded several decades ago. We critically analyse some of the architectural assumptions underlying existing systems and point out some problems in the domains of text planning and lexicalization. Guided by the identification of major generation challenges viewed from the angles of knowledge-based systems and cognitive psychology, we sketch some new directions for future research.

1 Introduction

The ability to carry out any complex task hinges critically upon the way in which the process for accomplishing it is organized. In current practice, the success of accomplishing the main task depends upon the decomposition of the whole into a series of manageable, flexibly related subtasks.

Language generation is a complex task that requires a considerable amount of knowledge, including domain knowledge and linguistic knowledge at various levels: pragmatics, semantics, syntax, morphology, phonology, etc. Because different kinds of knowledge are needed for different subtasks, each kind of knowledge preferably handled by its own data structures and processing mechanisms, generation does not seem to be a homogeneous process. Discourse organization requires pragmatic knowledge, lexical selection requires semantic knowledge, syntactic plans require knowledge of grammar rules, inflexion requires morphological knowledge, etc. Given this heterogeneity, the generation process is, at first sight, best viewed as a number of modules reflecting the various strata of knowledge. However, if one agrees with this

view, one must also address the question, how these various modules are to be delineated and coordinated in order to allow decision making in an effective way. As we will see, this is a crucial problem for generation architectures. In this context, the following aspects of representation and processing are relevant:

1 *Processing aspects:* How is the generation process decomposed into subtasks, modules, stages or levels? What is the control structure that coordinates the various modules and directs the information flow between them? How can we manage to be efficient while remaining flexible to handle complex demands?

2 *Representation aspects:* What information is relevant at the various levels to achieve linguistic expressiveness? What are the intermediate representations that serve as input and output at each level? What are the most adequate formalisms to represent and manage the various kinds of knowledge involved in generation?

Some important guiding criteria in addressing these issues are *efficiency* and *manageability*. However, these criteria act quite differently according to whether we are dealing with natural or artificial systems. Humans and computers are subject to different constraints, in particular with respect to attention span, memory size, and processing speed. There are also differences concerning the granularity of processes, the communication cost between processors, etc. With respect to representation, computer systems are often made efficient and manageable by a homogeneity at a high level (the representation formalism), whereas humans probably depend on homogeneity at a much lower level (the neuron). Furthermore, a given architecture may be less than optimal depending on the task. Building a system for doing a specific job is one thing, but building a multipurpose architecture is something quite different. Finally, one must keep in mind that, while language generation is similar to other cognitive processes in that it is a knowledge intensive process, it is also special in many ways, so that architectures borrowed from other cognitive domains may or may not be applicable.

McDonald, Vaughan & Pustejovsky (1987) systematically compare alternative designs for natural language generators by introducing an abstract reference model. In this paper, we will not only discuss newer systems, but also put somewhat different accents. We will first have a look at current systems and point at a number of shortcomings. Then, we will suggest how generation architectures can be improved by looking at research done in the area of knowledge-based systems as well as in psycholinguistics. Finally, we will point out some desirable characteristics of future architectures.

2 Some Current Architectures

Traditionally, the process of generation is decomposed into two stages which are realized as the main modules in a generation system. Some terms used to label the modules are juxtaposed in Table 1. One module determines the content of an utterance, or *what to say*. The other one realizes its expression in agreement with the rules of a given language: it determines *how to say* it. However, this distinction may be too simplistic. The division of labour between modules is not always exactly the same, and

the point has often been made that no clear cut line can be drawn between them (e.g. Danlos, 1984).

Table 1.
Modules of the generation process

What to say	*How to say*	
Strategy	Tactics	(Thompson, 1977; McKeown, 1985)
Speaker component	Linguistic component	(McDonald, 1983)
Goal identification and utterance planning	Realization	(McDonald, 1987)
Conceptualizer	Formulator	(Kempen & Hoenkamp, 1987; Levelt, 1989)
Deep generation	Surface generation	(McKeown & Swartout, 1988)

As a starting point, we will use the division between Conceptualizer and Formulator, without necessarily subscribing to all of Levelt's claims. The *Conceptualizer* is a nonlinguistic module which composes the content of a discourse by selecting and organizing information. In doing so, the Conceptualizer must take into account communicative requirements such as those relating to coherence, reference and focusing. The *Formulator* casts a message into a linguistic form by using the resources of a particular language, i.e. its lexicon and grammar; sentences are built by selecting appropriate words, which are put in the right order and shaped according to morphological and phonological rules. In addition to the Conceptualizer and Formulator, peripheral modules are required for the realization of speech or written text.

The modular approach is useful both from a psychological and from an engineering point of view. From a psychological point of view, there is evidence for autonomous modules: it seems that in human language generation, each module's mode of operation is minimally affected by the others (Levelt, 1989; Fodor, 1983; see also below). It has also been argued that the speed of human language generation requires an efficient system where different modules operate simultaneously (in parallel) on different pieces of the utterance. From an engineering viewpoint, decomposition into modules makes the task manageable and testable: one can concentrate on the processing aspects of a specific module without worrying too much about the internals of the others, provided that the interface between the modules is well defined. It remains to be seen, however, to what extent it is realistic to have the different modules operate independently, and how they communicate.

Various authors deal with the questions of decomposition and interaction differently. One extreme is a *sequential* architecture with a one-way information flow, where modules are maximally independent. The other extreme is an *integrated* architecture, where knowledge at all levels acts together. In between the extremes, there are architectures with varying kinds of interaction between the modules. A schematic overview of the information flow in different architectures is depicted in

Figure 1. We will now discuss different forms of decomposition and interaction in somewhat more detail.

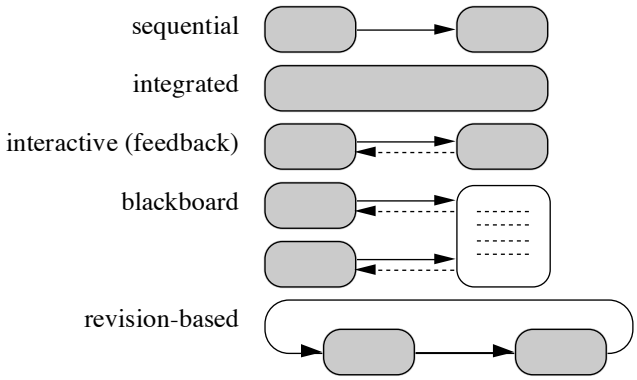

Fig. 1. Schemes for control of information flow

Sequential architectures incorporate a one-way flow of information through processing stages ordered in time. Roughly speaking, the Conceptualizer first determines the content to be uttered. It delivers this message to the Formulator, which subsequently attempts to compute an appropriate linguistic form. In a strictly sequential architecture, there is no direct feedback between these stages. The systems MUMBLE (McDonald, 1983), TEXT (McKeown, 1982; 1985), NAOS (Novak, 1987a; 1987b), and WISBER (Horacek, 1990; Horacek & Pyka, 1988) are representatives of a sequential architecture. So are the psycholinguistic models IPG (Kempen & Hoenkamp, 1987) and IPF (De Smedt, 1990; 1994). The architecture of SYNPHONICS (Abb et al., 1995) is also sequential, albeit that the sequence is not strictly linear, because phonological processing relies not only on syntactic information coming from the Formulator, but also partly on information coming directly from the Conceptualizer.

A module may itself contain other modules (submodules) of varying degrees of autonomy. In the Formulator, two components are often distinguished, the first one dealing with lexico-syntactic processing, resulting in a syntactic surface structure, the second one dealing with morpho-phonology, producing a sequence of word forms. Partitioning can go even further. In IPG, there is a processor for every major syntactic category or function; thus there is a noun phrase processor, a subject processor, an adjectival phrase processor, etc. This high degree of modularity is motivated by the benefit of having the different modules work in parallel.

A strictly sequential architecture has the advantage of simplicity. Each module is maximally encapsulated because it receives input from one source and sends output to one destination. The intermediate representation at each stage is assumed to be a complete rendering of what is known at that point about the utterance to be generated. However, a problem with a one-way flow of information between modules its need to make early commitments in the construction of an intermediate representation for the next module. The local decisions taken in each module do not necessarily sum up to a good overall result. In particular, this architecture faces the *generation gap* problem (Meteer, 1990): it cannot always be guaranteed that there are words covering exactly

the content to be expressed, nor that there are suitable syntactic patterns that fit a conceptual structure. An avoidance strategy adopted by some systems is allowing the Conceptualizer to 'think' only in terms of the words in the language, thereby trivializing the problem of lexical choice to what Marcus (1987) calls 'capital letter semantics'.

KAMP (Appelt, 1985) is the best-known example of an *integrated* system. All decisions are taken within a single, hierarchically structured process, so that generation takes place in a continuum of goal-driven and constraint-based processing. An integrated architecture allows different kinds of knowledge to be represented and processed in a uniform way, which is advantageous because of the systematicity in building the knowledge sources. However, it can also be a disadvantage if the different knowledge sources are so tightly interwoven that they can hardly be localized. This hinders maintenance, transparency, and the construction of an efficient large scale system.

Interactive architectures involve feedback between different stages of processing at certain points, in particular from the Formulator back to the Conceptualizer, as in PAULINE (Hovy, 1988a; 1988b; 1990) and POPEL (Finkler & Neumann, 1989; Neumann & Finkler, 1990; Reithinger, 1991; 1992). In principle, formulation begins with incompletely specified contents. The resulting gaps in the partially generated linguistic form trigger further planning at the conceptual level to complete the utterance. In POPEL, the Conceptualizer (POPEL-WHAT) may provide the Formulator (POPEL-HOW) only with a concept that can be expressed as a verb; the Formulator may then notify the Conceptualizer that it also needs a concept to express the subject of that verb. Which of the possible interactions are useful, and how to coordinate the interacting modules, are the main issues in an interactive architecture. POPEL uses special mechanisms, including a *request handler* that provides the necessary interface between the modules. If the different modules in an interactive architecture are well coordinated, they can operate in parallel. This is the case in POPEL, where the different modules operate simultaneously (*intercomponent* parallelism). Both modules also work on different pieces of the utterances at a time, thereby realizing *intracomponent* parallelism.

Interaction can also be realized in other ways. In Rubinoff's (1992) IGEN system, the Formulator provides feedback to the Conceptualizer in the form of *annotations* that tell it how much of the content can been covered by a particular word choice. With these annotations, the Conceptualizer can then determine which choice satisfies its secondary goals.

Another form of interaction is provided by *blackboard* architectures for generation, as in the DIOGENES system (Nirenburg & Nirenburg, 1988; Nirenburg, Lesser & Nyberg, 1989). In a blackboard architecture, modules provide information without needing to know exactly which other modules use it. Each module looks on a blackboard for the information it needs and writes its own output on the very same blackboard. An advantage of this architecture lies in the identification of a set of *knowledge sources* (this is the term for modules in DIOGENES), each of which is equipped with clearly delineated competences, thereby strengthening the system's maintainability and providing evidence about its coverage. In addition, blackboard systems allow for flexible processing: the difference in the temporal execution of the

various modules stems from the different orders in which triggering information becomes available. However, the interplay between the knowledge sources on the blackboard needs to be supported by some mechanisms to increase efficiency and to resolve conflicts. In particular, extra control knowledge is needed to assign priorities to knowledge sources, as well as a limited backtracking coupled with a simple truth maintenance system.

In order to benefit from the simplicity of a modular architecture while compensating for its limitations, several *revision-based* approaches have been proposed. These presuppose a limited form of feedback, mediated by monitoring modules that inspect intermediate structures. In the systems KDS (Mann & Moore, 1981) and YH (Gabriel, 1986) these intermediate representations are amended, whereas in WEIVER (Inui, Tokunaga & Tanaka, 1992) early decisions which led to unsatisfactory results are undone. A text specific approach using revision is proposed by Robin (1993) in a system that describes sports events. The architecture used in this system is based on a two-pass process. Basic, new information is first generated in a draft, which is subsequently revised to incorporate historical background information. An advantage of this revision-based approach is that it allows to rate communicative goals according to their relative importance.

Having surveyed the major paradigms, we are now prepared to ask the question, which architecture is the best. Even though we will go deeper into specific issues below, a preliminary answer to this question is that there is no universal solution, but the choice depends on the functionality and coverage the generator is required to have. Stratificational architectures with a sequential information flow are widely used and seem sufficient when the task does not require special capabilities. In contrast, special purposes or genres suggest the use of more complex interactions between modules. Robin's revision-based system seems suitable in cases where the genre dictates the global organization of the text, further illustrative details being added wherever they fit in. Another example with special constraints is *incremental* generation, where the demands of real-time uttering require a close coordination between the Conceptualizer and the Formulator. It is this constraint which has led some authors to adopt an interactive architecture (Finkler & Neumann, 1989; Neumann & Finkler, 1990). Another problem to be considered is the cost or effort it takes to build a system. As a matter of fact, one can afford to build a more ambitious architecture if the domain is restricted, because then the complexity of choices will not be as high.

3 Problems Concerning Text Planning and Lexicalization

Orthogonal to the enterprise of setting up a flexible control mechanism (an issue which we will revisit later) is the goal of coordination, which will be addressed in this section from the perspective of increasing a system's expressiveness. In particular, the coordination of skillful lexicalization mechanisms and text planning processes can contribute to a rich repertoire of expressive means. Despite enormous progress in both areas, however, the intended capabilities are still quite poor compared to those exhibited by human speakers and writers.

In the area of text planning, researchers have identified more problems and limitations of existing theories than they have provided new solutions. With regard to

lexicalization, only few researches have made proposals that go beyond one-to-one mappings between concepts and words (but see Horacek & Pyka, 1988; Nogier & Zock, 1992), even though the problem of *useful packaging* has been identified as a crucial one already quite some time ago (McDonald & Pustejovsky, 1985). We will now turn to the issue of text planning, then we will address the issue of lexicalization, and finally we will discuss their relationship.

3.1 Text planning

Computational approaches to text planning are among the most appealing contributions to automated natural language generation in the past decade. Important milestones on this route were the introduction of *text schemata* (McKeown, 1982; 1985) and *Rhetorical Structure Theory* (RST), as formulated by Mann and Thompson (1987; 1988; Mann, 1988). Meanwhile, RST has been operationalized by the application of a traditional top-down planner (Hovy, 1988a), and has been further refined by the introduction of intentional operators (Moore & Paris, 1993).

Several shortcomings of RST and of text planning techniques in general have been identified. In the original version of RST, there was a simplifying convention that text spans joined by rhetorical relations have to be clause-sized chunks (W. Mann, personal communication), even though there can be rhetorical relations below the clause level: the *elaboration* relation can operate as an adjective or as a relative clause (*The blue ball is heavy* vs. *The ball that is blue is heavy*). In practice, RST-based text planners widely keep this simplification, because they use rhetorical relations mainly as a basis for the insertion of appropriate connectives. Below the clause level, however, it is not primarily rhetorical relations, but rather configurational restrictions imposed by words (prominently, by verbs) which essentially determine mappings from conceptual specifications to surface structures.

Other problems have been identified as well. Carberry et al. (1993) have identified two interesting cases which are more typical for dialogs than for texts. First, elaborations of different topics may interfere with each other if they originate from alternating dialog contributions. Second, an argument may refer to a rhetorical relation itself rather than to a text span (for instance, putting doubt on the truth of the particular relation expressed). Neither possibility is supported by RST. Moreover, Moore and Pollack (1992) have presented examples of texts that clearly demonstrate that ideational and textual relations, which are two of the three main types of rhetorical relations according to the taxonomy introduced by Maier & Hovy (1993), may co-occur in a text but are not necessarily isomorphic to each other. To conclude this discussion on text planning, one can see that RST is not a panacea.

3.2 Lexicalization

Lexicalization (or lexical choice) has become a general term comprising a variety of subproblems associated with the transition from conceptual to lexical representation levels (see Busemann, 1993): definite reference, choice of open class words, collocations, connotations, and sublexical relations. As with text planning, it has proved very hard to integrate all these issues into an informed and manageable process. In

particular, correspondences between conceptual and lexical entities deviating from the simple one-to-one pattern are not frequently encountered.

One noteworthy exception is NAOS, which generates natural language descriptions of a traffic scene filmed by a video camera. Another exception is ANA (Kukich, 1983; 1988) which takes stock-market data as input. In systems like these, the increased complexity of the correspondence between conceptual and lexical representation levels is a necessity rather than a mere increase of expressiveness, because the input for natural language generation differs significantly from any linguistic terms. In NAOS, event models have been designed that enable the system to describe the relation between two trajectories associated with cars as *passing by* or as *overtaking,* depending on the commonalities and subtle differences that the concrete situation has with the relevant event models. Some other systems with nontrivial transformations from the conceptual to the lexical level are KING (Jacobs, 1987), which can generate descriptions of the same event from various conceptual perspectives, and WISBER, which supports the production of descriptions exploiting metonymy. In all these systems, the lexicalization issue is a dominating feature in comparison to their overall capabilities.

Various authors (Horacek, 1987; Zock, 1993) consider the lexical component to be a mediator between deep and surface generation. Lexical items introduce important constraints both on *what can be said* and on *how it can be said*. A clear example is found in *collocations*, where a given lexical item implies the choice of another, e.g. *ask/put a question* vs. *give an answer*. In addition, lexical items are *subcategorized* in the lexicon according to which syntactic constraints are imposed on the sentence structure. The verb *mind,* e.g., allows for a gerund: *I don't mind waiting,* whereas *want* requires a to-infinitive: *I don't want to wait*.

This pivotal role of lexicalization shows perhaps most clearly in *multi-lingual* generation. Rösner and Stede (1992; Stede, 1995) have reported about remarkable structural differences between the versions of English, French, and German in multi-lingual technical manuals. In one language, some process may be described as the reverse of an action (*unscrew*), while in another it is described in terms of the intended result (*lockern,* 'make it loose'). None of the presently known planning techniques in generation would be capable of systematically producing these variants. Current systems seem to be confined to a particular kind of sublanguage they are able to handle competently. It is an open question how the planning steps could be organized in such a way as to account for the various sorts of influences mentioned.

3.3 Coordinating Text Planning and Lexicalization

Given the broad range of problems in text planning and lexicalization mentioned above, it is hardly surprising that no serious attempts for an integrated solution exist, despite the obvious interdependence of the two subtasks. The need for taking into account this interdependence manifests itself, for instance, in the problem of applying *aggregation* to generation. One kind of aggregation is conceptual: the content of several propositions is cast into one lexical item. Another kind of aggreation is set-forming, i.e., propositions sharing information can be reduced in number. A typical case is conjunction: the two clauses *Vera cheated* and *Bernard cheated* get reduced to a single clause *Vera and Bernard cheated*. Cases requiring aggregation have been

identified and some necessary mechanisms have been proposed (Horacek, 1992; Dalianis & Hovy, 1995), yet we still lack a general methodology for rules of aggregation, so that applying aggregation systematically in lexicalization and text planning remains an unsolved problem.

Another issue related to the integration of text planning and lexicalization is the appropriate use of *implicature:* propositions implied by others need not be explicitly mentioned in the text. Implicatures are part of text planning (Horacek, 1994), but they are also relevant for the generation of referring expressions. Failure to take this kind of cross-dependence into account may lead to incoherent texts. Consider the sentence *Group leaders must be assigned to single rooms, among which only B is located close to a meeting room*, where a reference at the individual level (*which*) incorrectly has an antecedent at the generic level (*single rooms*). Here, adding the individual to the generic expression improves the sentence, yielding *John as a group leader must be assigned to one of the single rooms, among which only B is located close to a meeting room.*

Yet another example showing how lexical choices can influence text planning is the expression of attitudes. We don't know of any system capable of choosing deliberately between expressing attitudes either by lexical items carrying *connotations* and verbalizing the attitude by other textual means. A deliberate choice between the different means would require an explicit representation of the attitude towards the entity. As this is not the case in present systems, the choice of a lexical item with a connotation is usually triggered simply by a feature or a set of features. These features may be organized in quite complex ways, e.g., in taxonomies (Stede, 1993; 1995), or in systemic networks (Wanner, 1992). So far, one of the most elaborate pieces of work to express pragmatic factors adequately – which obviously includes a good deal of lexicalization and text organization issues – is Hovy's system PAULINE.

Summing up, we feel that many of the shortcomings of text planning techniques mentioned earlier (if not all of them) are caused by an insufficient account of the cross-influence of lexicalization issues. In our view, a problem that substantially hinders the integration of lexical choice and text planning is the insufficiently defined ontological state of all types of entities other than the lexical ones. In particular, RST requires that the nodes in its trees are conventionally text spans, but in the context of generation, the origin of these text spans has yet to be determined. It is unclear if the nodes in RST trees originate at some deeper level from conceptual units, and how these units relate to lexical material. Therefore, text planning and lexicalization operators do not interface well; actually, it is not even clear whether the operators are really working on the same type of items.

Continuing along these lines, it is noteworthy that in generation systems trying to choose between different linguistic resources for achieving a specific effect, the choice is often based on situation-dependent co-occurrences of sets of features. This technique is quite different from currently dominating approaches to text organization. The latter perform essentially top-down hierarchical planning in the tradition of Cohen & Levesque's (1985) theory of rational action and interaction: text planning is done by building a proof-like text plan which, once carried out, is believed to guarantee the achievement of the original communicative goal, provided the assumptions made in the course of the planning process are true. While we feel these two techniques are

complementary rather than contrastive, a suitable integration is still to be demonstrated.

Last but not least, it is interesting to investigate to what extent issues of expressiveness have implications for generation architectures. We find that few current systems break the trade-off barrier between paying attention to expressiveness and building a flexible architecture. PAULINE and Robin's (1993) system are two exceptions, but it seems very hard to adapt their techniques to different environments. PAULINE does not build any intermediate representations during the generation process. Furthermore, the use of a phrasal lexicon (Hovy, 1988c) incorporating knowledge about grammar rules, partially frozen phrases, and words, is quite different from what most other researchers do. These design discrepancies make a transfer of parts of PAULINE to other systems almost impossible. Robin's system is genre-dependent, hence a transfer of his techniques could be expected to work successfully only if the target domain is similar enough to the domain of sports reports.

In conclusion, most researchers have either concentrated their efforts on flexibility of control or on rich expressiveness, which leaves the issue of integrating the achievement in these two main streams still to be done, even though some recent work starts to address this integration issue. An example is Wanner's (1994) proposal to introduce a kind of lexical biases in discourse planning. This seems to presuppose an interactive architecture with feedback from lexical choice to discourse organization.

4 The Challenge of Flexible Architectures

When considering process control in architectures for natural language generation, it is useful to stand back and take a view from a more general perspective. Until now, such issues, even if common in Artificial Intelligence (AI) and especially in knowledge systems research, have unfortunately remained underexposed in the field of natural language processing. Although many advances in natural language generation have been motivated by relevant criteria such as efficiency, knowledge integration, etc., most of this work is formulated within the context of specific systems and is meant to overcome problems associated with the idiosyncracies of certain inputs or certain grammar formalisms. Architectures with a broader scope are scarce, and consequently most generation architectures suffer in several respects.

4.1 Problems related to Process Control

We would like to mention several problems related to process control. First, there are hardly any measures by which the search space can be characterized, in order to allow a principled choice of an adequate problem-solving strategy. One example of how to build such a measure is by letting the range of possible lexical items for a concept determine whether the lexicalization process should be carried through, or should be postponed until constraints arising from other choices make the decision easier. Hence, the overall goal is to set up an opportunistic control of the search, which can eventually be parametrized according to the needs of the task at hand. In contrast to this desideratum, sequential architectures merely perform forward pruning, cutting away all but the locally best candidate, which eventually has to be determined by some look-

ahead process. Revision-based architectures attempt to repair poor choices made at an earlier stage, but hardly ever in a principled way. Given our current experience with integrated architectures, it seems that the problem of explicitly defining the search space in these architectures is intractable. Interactive architectures are also problematic, as the flow of control is defined only implicitly.

Second, there are no explicit evaluation techniques applied, especially not in terms of global quality assessments. Search control is based on the assumption that local preference criteria accompanied by forward pruning would do the job. There seems to be an implicit agreement that these criteria comprise the acquaintance of the addressee with the terms used, as well as the unambiguity and the fluency of the resulting utterance (in decreasing order of importance). While this approach seems to be plausible in general, taking it systematically seems at least debatable in a good number of occasions.

Third, most architectures are rigid and lack flexibility because they are extremely task-dependent. Present implemented systems usually work for only one task, e.g., story generation, paraphrasing, summarizing, explanation, or the translation of a sentence. Even more specifically, the task is often restricted to a single domain and is carried out in a particular communicative setting. In addition, current generation architectures do not adapt to varying circumstances with respect to availability of knowledge, the size of the search space, the availability of time and memory and other constraints relating to process control. Finally, they do not improve by learning, neither in terms of the grammar and vocabulary of the language, nor in terms of planning and processing strategies. The premise is that adaptable systems are in principle better than rigid ones. Of course, engineering issues have to be taken into account. Clearly, the cost of developing several separate systems, each targeted at a specific application, may be less than that of an adaptable system. However, the maintenance aspect in language engineering is all too often overlooked. In this respect, adaptable systems may significantly reduce manual efforts. In the long run, we believe that adaptable systems will constitute an important contribution to both research and engineering.

Achieving flexibility is not straightforward. Given the complexity of the codependencies between the knowledge sources and the multiplicity of choices, it may be impossible to specify an optimum or even a good general control strategy for a generator in all cases. Therefore, it may be advisable to have the generator itself determine the appropriate control strategy, including the amount of feedback and correction, the depth of forward search, the relative ordering of lexical and content choice, the relative amounts of content structuring and syntactic realization, etc. The realization of such a reflexive control strategy requires a dependency on situational parameters and a declarative representation, not only of the generator's rules and criteria, but also of their complexity, their cost, the types of other knowledge they depend on, etc. None of that is clearly understood right now, but in the area of knowledge-based systems design, researchers are beginning to adress such issues.

4.2 Knowledge-Based Architectures

To help arrive at a solution to the challenges that generation systems face, we consider the transfer of architectural insights from the field of knowledge-based systems, where researchers have been concerned with architectures that increase flexibility and

efficiency. Language generation is a knowledge intensive process, and in the past few decades there have been limited efforts to apply knowledge-based techniques to solve language processing problems, in particular at the level of knowledge representation. The representation of linguistic knowledge has often been cast in terms of AI formalisms such as *rules, logic, frames* and *networks of structured objects*. Powerful knowledge representation mechanisms such as *object-oriented modelling* and *default inheritance* have been incorporated in grammar formalisms as they are in other cognitive domains (Daelemans, De Smedt & Gazdar, 1992). The premise is that general mechanisms for knowledge representation apply to many cognitive domains including language.

An example of a generator implemented in an object-oriented way is IPF. It uses an object-oriented formalism called Segment Grammar (De Smedt & Kempen, 1991), where knowledge is distributed among many computational objects, each representing a certain entity or concept playing a role in the lexicon, the grammar, or the domain of discourse. The computation of utterances arises by the exchange of knowledge among objects, which is often referred to as *message passing*. The grammar consists of objects representing the *noun phrase*, the *verb phrase*, the *noun*, the *proper noun*, the *verb*, etc. Likewise, the lexicon comprises objects for all words with their specific features. Objects are organized in hierarchies, allowing them to share knowledge efficiently by means of default inheritance (cf. also Daelemans & De Smedt, 1994). The *proper noun*, e.g., inherits much of its information from the object *noun*. The hierarchies need not be distinct, so that the same objects can play a different role in different modules (*polymorphism*). The syntactic information associated with nouns can thus be hierarchically organized way different from the morphological information associated with the same objects.

Perhaps more is to be learned from knowledge-based systems if we do not restrict ourselves to representation mechanisms, but also want to address the issue of process control. To this end, it is necessary to first cast language generation in a terminology of knowledge-based systems (e.g. Steels, 1990). In this terminology, *problem-solving methods* consult and expand initial information structures in order to arrive at a certain goal structure by means of planning and search. In the case of generation, the initial information structures are specifications of communicative goals, and an important goal structure should consist of a linguistic specification of utterances. Problem-solving methods can be divided into different types, as shown in Table 2.

Table 2.
Methods (left) and the representations they affect (right).

task decomposition and execution methods	problem case
meta-problem-solving methods	problem-solving strategy
learning methods	knowledge sources

Task decomposition methods divide a task into subtasks, and impose a temporal order of execution on them. In most generation systems, task decomposition proceeds along the fixed lines of a sequential, integrated or interactive architecture, as already

mentioned. *Task execution* methods convert a task into directly executable instructions. In the case of *unification grammar* (Kay, 1979; 1984), task execution involves only one operation (*unification*) which explores a single search space, continually expanding the initial message description until it contains a sufficiently specified linguistic description. Similarly, in *systemic grammar* (Fawcett, Tucker & Lin, 1993), there is a fixed constellation of systems which is traversed, taking decisions and expanding the structure. Indeed, Patten (1988) explicitly proposed a problem-solving approach to generation, based on systemic grammar.

Current research in knowledge-based systems is oriented toward the incorporation of more sophisticated methods that do not simply expand the problem description, but rather build up a model of the problem-solving process itself. These methods are realized as *meta*-problem solvers. Architectures with meta-problem solvers are sometimes called *introspective* or *reflective* systems, because they can reason about their own behaviour and modify it. A reflexive capacity opens up the opportunity to react to changing circumstances by changing the way the problem solver itself operates.

Maes (1986; 1987) discusses computational reflection. She gives examples of systems with reflective architectures, and presents an implementation of 3-KRS, a reflexive object-oriented language. In this language, every object has a one-to-one relation to a meta-object, that represents a meta-circular interpreter: it implements the whole computation of its object. Because the meta-objects are also objects, reflection can recurse infinitely, if needed. In systems with limited reflexive capabilities, reflection occurs in a prewired way. SOAR modifies its own computation strategy by *chunking* the results of previously experienced problems (Laird, Newell & Rosenbloom, 1987). This architecture has been applied in NL-SOAR, a model for natural language generation (Rubinoff & Lehman, 1994), as well as for comprehension (Lewis, 1993). Fully reflexive architectures have to our knowledge not been applied to natural language processing.

A second kind of sophisticated method consists of *learning* methods that build up and change the knowledge sources, which in the case of generation consist of knowledge about pragmatics, semantics, syntax, morphology, etc. Learning is a new and growing field of interest within language processing research (Daelemans & Powers, 1992), but specific needs for learning in a language generation task have not yet been addressed, and implemented models are scarce. Among those learning techniques which have been successfully applied are data-oriented and similarity-based techniques, e.g., *backpropagation learning, instance-based learning* and *analogical modelling* (Daelemans, Gillis & Durieux, 1994). These techniques, however, are performance-oriented and are not meant to produce knowledge sources for a competence-oriented model.

Despite some advances, we currently find very few language generation systems which incorporate problem-solving methods that can be called sophisticated. Considering architectures from the knowledge systems perspective, a major argument for sophisticated methods is that they increase the adaptive power of the system. This is relevant for language generation systems as well as for other intelligent systems, especially because present-day generation systems are almost without exception very rigid systems. An adaptive capacity is advantageous because actual generation of

natural language is hardly a standardized, one-shot task (Zock, 1993). On the contrary, generation systems should ideally be adaptive to a variety of tasks, each differing slightly with respect to epistemological and processing constraints depending on the context, the available knowledge sources, the importance of the goal, etc.

Let us now consider the way in which the various problem solving methods are represented and organized. In a procedural architecture, (see Figure 2), domain knowledge and a specific problem-solving method are compiled to form a domain-specific method. Kempen and Hoenkamp (1987) motivate the use of the procedural architecture in IPG on psychological grounds: human speech is an automatic activity, i.e. people do not seem to be able to consciously manipulate a model of speech. A procedural approach is comparable to the organization of first-generation expert systems like MYCIN (Buchanan & Shortliffe, 1984). However, a procedural architecture has also clear disadvantages from the viewpoint of system development and maintainability. Extensions to the language processing system may lead to discrepancies between existing procedures and new situations, which would require patches to solve the conflict.

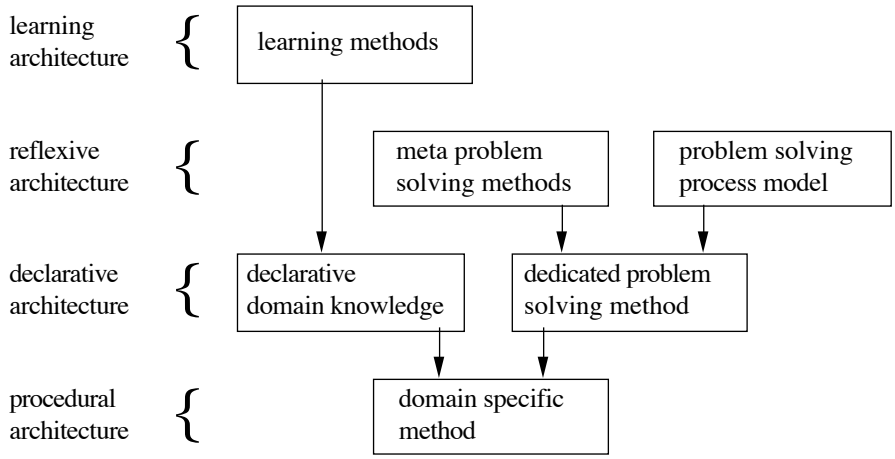

Fig. 2. Encapsulation of problem-solving methods in knowledge-based architectures.

Most present-day systems have a *declarative* architecture, where the domain-specific knowledge – which in the case of the Formulator consists mainly of the grammar and lexicon – is kept separate from a domain specific problem-solving method. Thanks to this separation, a declarative representation is easier to build and maintain. The dedicated problem-solving method may consist of a unification mechanism in the case of unification grammars, a resolution mechanism for definite clause grammars or constraint logic programming grammars (e.g. Saint-Dizier, 1992), or a kind of rule-based engine for systemic grammar.

However, purely declarative architectures sometimes lead to problems, e.g., when the chosen problem-solving methods are inefficient due to extensive backtracking. Ad-hoc solutions are sometimes pursued in the form of incorporating domain specific control methods into the grammar (e.g. Elhadad & Robin, 1992). Such approaches are effective but carry a price tag because they leave the path of declarative representation.

A completely different route offering more flexibility would consist in building a *reflexive* architecture (see Figure 2), where a separate problem-solving process model is maintained for reasoning about control. The actual linguistic problem-solving method is thus subject to manipulation depending on the situation. Although unification-based, logic-based and rule-based paradigms offer some of the flexibility needed for this purpose (e.g., by allowing forward chaining as well as backward chaining, depth-first as well as breadth-first search, or bottom-up as well as top-down processing), these possibilities have so far remained underexploited. However, one could imagine a reflexive architecture where this flexibility is exploited by meta-problem-solving methods that optimize the search path based on experience and depending on the specific generation task at hand.

An interactive generation system could, e.g., automatically be configured according to domain-relevant parameters in terms of the ingredients needed and the suitable flow of control. Also, a generator could memoize sequences of text planning or lexicalization operators explored including the resulting consequences. This information could then be used in similar situations occurring later, to quickly follow a line of reasoning that is likely to lead to a preferred intermediate result, or to avoid a potential dead end. In any case, the adequate definition of the generator's search space in terms of composable operators and globally applicable quality-assessment metrices is a crucial prerequisite for these issues.

These efforts could be complemented by the automatic acquisition and adaptation of the declarative knowledge needed for generation in a *learning* architecture, where learning methods acquire the necessary linguistic knowledge (see Figure 2). As already mentioned, machine learning of linguistic knowledge is becoming an increasingly important research area. Learning not only avoids the practically unmanageable task for humans to build full representations of the grammar, lexicon and other relevant knowledge, but also allows for the automatic *optimization* of grammar rules and other linguistic knowledge sources when performed automatically. As learning methods have not yet been applied to core issues in natural language generation, we expect that major research efforts will have to be dedicated to this goal. Learning grammars and lexica can be done by natural language understanding systems from copora. In order for generation systems to learn, they must be exposed to the correct verbalisation of a communicative goal, something true for a child, but not easy to replicate for an artificial system.

Summing up, what we envision for natural language generation architectures in the next decade is some kind of transfer of sophisticated methods from knowledge systems research, in particular the incorporation of learning and meta-methods to make generation systems more adaptive. Additional ideas from knowledge-based systems that could be also applied to language generation include competition-based models, the metaphor of the society of communicating experts, accomodation, generation as puzzle solving, and opportunism.

However, the transfer of architectural insights from knowledge-based systems is, by far, not a straightforward task. It is substantially hindered by some process-specific particularities of the generation task, which are insufficiently mirrored in knowledge-based architectures. Knowledge-based systems generally assume a homogeneous search space, where all operators are basically of the same kind (even including meta-

rules, sometimes). The formation of rule packages proposed for certain types of applications to gain efficiency is mainly motivated by the sequential flavour of the particular task to achieve, so that the order of rule applications can be guided accordingly. One could object that this methodology does not transfer well to our target domain. In generation, the search space is essentially heterogeneous, comprising, e.g., alternatives with respect to textual organization as well as with respect to lexical features. Both choices are of quite a different kind, especially with respect to the granularity of the structures affected, but they are not independent. Therefore, casting these heterogeneous operators into a homogeneous form to allow the application of knowledge-based techniques, including reflexive techniques, is likely to result in either overly complex operators or oversimplifications.

An alternative to classical rule-based techniques are constraint-based formalisms. However, propagating constraints across different levels can hardly be done in an efficient way, because in a fixed constellation of processing levels, constraints cannot be ordered in a flexible way. A pragmatic constraint aimed at a high degree of formality has consequences on various aspects of surface structure, including sentence length, constituent order, etc. (see Hovy, 1988b). According to our intuition, the softness of many pragmatic constraints, as well as the complexity of the relations among constraints of all kinds make it hard even to formulate a consistent set of constraints expressing the role of pragmatic factors in an interesting subset of natural language, let alone efficiently applying propagation techniques. Saint-Dizier (1992) describes a constraint-based approach to natural language generation, but the scope of his work is confined to syntax and only some semantic phenomena. Although he argues in favour of the general usefulness of his approach – for this and for other areas of generation –, this has not been substantiated to date, which comes as no surprise given our previous discussion.

Finally, there is an even more fundamental objection to the transfer of a knowledge-based methodology. Some researchers claim that the limitations of current knowledge-based systems cannot be remedied by the incorporation of ever more sophisticated methods. Indeed, much work on advanced knowledge systems is purely theoretical and has not been carefully evaluated. What is worse, Brooks (1991) is probably right in his claims that human level intelligence is too complex and at the moment too little understood in order to be correctly decomposed into modules. Even if we knew the modules, we still would not know the right interfaces between them. Because researchers put too much abstraction in their problem definitions, their use of internal representations interfacing between modules may be fairly different from those used by humans.

Brooks offers an alternative, namely, to build up intelligent systems that are endowed with sensors and are capable of responding to actual situations in the real world. In natural language generation, there are unfortunately only few systems that are complete in the sense that they have sensors in the real world and react by generating natural language expressions. One of those 'complete' systems is SOCCER (André, Herzog & Rist, 1988), which produces descriptions of a soccer match observed through a video camera, but it should be added that the system only works for a very narrow domain.

Taking Brooks' claims seriously would mean taking a drastically new approach to the generation problem. In line with his ideas, one could imagine separate coexisting generation strategies with different levels of refinement, ranging from pattern-driven (*shallow*) generation to grammar-based (*deep*) generation. In a Brooks-like architecture, one of these strategies could suppress the others whenever appropriate. Stereotypical situations could give rise to prepackaged responses in the form of pattern-driven generation, whereas more complex situations may require a full-fledged generation with the intervention of text planning and grammar components to provide a more refined and tailored response.

5 Psychologically Motivated Architectures

Ideas from knowledge-based systems, even if powerful, may still fall short as guiding forces in natural language generation, either because they have not reached applicability and maturity, or because they lack specificity and offer too many degrees of freedom. For inspiration, can also turn to computational architectures from psychology and cognitive science. A number of researchers have proposed models for human sentence generation which claim to be psychologically plausible (for a review of early proposals, see Valian, 1977; for a theoretical framework, see Levelt, 1989; Bock, in press; for a broad textbook in computational psycholinguistics, see Dijkstra & De Smedt, in press). These models are generally based on performance aspects like quality of generation (error rates) and its quantitative characteristics (time course and dysfluencies). The most frequent phenomena studied are speech errors (Garrett, 1980; Dell, 1986; Harley, 1982; 1984; Motley, Camden & Baars, 1982; Stemberger, 1985) and hesitations (Ford & Holmes, 1978; Ford, 1982). Among the processing assumptions we mention the notion of competing processes (Bates & Devescovi, 1989), time sharing, incremental parallel processing (Kempen & Hoenkamp, 1987; Levelt, 1989; De Smedt, 1994) and connectionism (discussed in more detail below). It should also be noted that psychologists have studied specific modes and conditions of the speaker, like writing (Flower, 1981; Flower & Hays, 1984) or agrammatic speech (Kolk, 1987).

Seminal work on the organization of the human language generation system was done by Garrett (1975, 1980). Having studied a corpus of several thousand naturally occuring errors, he noticed that many of them affected content words of similar semantic or syntactic categories (*boy* instead of *girl;* you can cut *rain* in the *trees*). Others, though affecting the content-word vocabulary, affected only adjacent words. This is the case for sound exchanges (*sh*inking *s*ips). On the basis of these errors, Garrett concluded that if two elements of a sentence are both involved in an error, then they must be simultaneously available at the stage of processing at which the error occurs. Similarly, if two types of elements are never involved in a given kind of error, then these two elements must be processed at different stages.

In Garrett's model, the transition from message to surface form is accomplished in a number of distinct, serially ordered steps, involving five levels of representation: message level, functional level, positional level, phonetic level and articulatory level. The *message* level roughly corresponds to a conceptual structure, whereas the next two levels pertain to lexical, syntactic and morphological generation in the Formulator.

Garrett makes rather precise claims concerning the stage at which words are generated. At the *functional* level, the speaker has access to the semantic representation of a content word, but not to its morphological or phonological representation. It is only at the *positional level* that phonologically specified word forms are accessed and placed in a syntactic sentence frame, to which function words are added. At a still later stage, a phonetic representation is produced, which is then transformed into a series of muscle movements.

Even though one could object that his model is underspecified, Garrett was able to confirm certain predictions concerning errors. In particular, if function words are accessed at a different moment than content words, these two kinds of words are subject to different kinds of errors. Garrett showed indeed that certain kinds of errors never occur, whereas others can be elicited in experiments. Still, his claims concerning the autonomy and serial nature of the various stages have been challenged by others (e.g., Dell & Reich, 1981), who showed that there is a large amount of data which pleads in favour of an interaction between the different levels of the language generator. E.g., substitutions at the word level are facilitated by similarity at the phonological level. Facts like this led some researchers to adopt an interactive architecture for sentence generation (Stemberger, 1985; Dell, 1986).

However, a certain degree of interaction does not necessarily preclude modularity. Bock & Levelt (1994) summarize work on lexicalization, concluding that the human lexical access system has a highly modular organization. They assume a distinction between processing at the *lemma* level, which deals with the syntactic and semantic properties of lexical items, and the *lexeme* level, which deals with their morphological and phonological forms. A substantial body of empirical work has revealed that interactions at the lemma level occur at different moments than those at the lexeme level. The conclusion is that the selection of lemmas strictly precedes and is unaffected by the computation of lexemes. Bock and Levelt suggest that there is a good reason for this modularity. Lexical selection amounts to a semantically driven search through a huge lexicon. Its incredible speed and accuracy can only be explained if it is not disrupted by phonological processes. Bock and Levelt interpret this kind of modularity as nature's protection against massive disruptions which could easily result in errors.

The prominence of lexical choice in the generation process has been shown by Bock (1986; 1987), who provides psychological evidence for the hypothesis that the syntactic structure is codetermined by the *order* in which lexical items become available. Indeed, in most psycholinguistic models of generation, lexical choice takes place prior to syntactic processing: a linguistically unspecified conceptual structure (message) guides the selection of lexical material, and subsequently, the properties of lexical material guide the construction of a syntactic structure (e.g. Kempen & Hoenkamp, 1987). However, priming experiments suggest that the choice of a particular syntactic plan can also be positively biased by prior exposure to an instance of such a plan (Bock, 1986). This seems to imply that syntactic choice is not exclusively determined by lexical choice.

Among the phenomena in human language generation that have been addressed by generation researchers, perhaps most attention has been paid to the incremental mode of sentence production, even if incremental systems are not always intended as psycholinguistic models. Human speakers are capable of starting the utterance of a sentence

before its syntactic structure or even its content have been completely specified. Several incremental models have been proposed and have been worked out in some computational detail. IPG, IPF, POPEL and SYNPHONICS, all mentioned earlier, operate incrementally by virtue of a modular approach: before the Conceptualizer has completely specified a message to be verbalized, the Formulator may already start building an utterance. Thus, these models have an interleaved (or pipeline) architecture allowing modules to operate partwise; moreover, the computations of one module may be distributed among various parallel processes, as illustrated in Figure 3. The interleaving is reminiscent of proposals motivated on linguistic, especially pragmatic grounds (Hovy, 1985; McDonald and Pustejovsky, 1985).

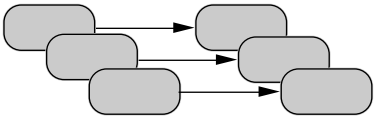

Fig. 3. Parallel and distributed operation in an interleaved architecture

Despite a common approach to modelling incremental generation, there are also differences between the various incremental models. IPG, implemented in a procedural way, and IPF, using a declarative grammar, are both confined to syntactic processing in a highly autonomous Formulator. POPEL's operation, as already discussed in section 2, is based on a mutual interaction between the Conceptualizer and the Formulator. SYNPHONICS, finally, includes a phonological component, but presupposes neither parallelism nor interaction.

Even if the incremental processing assumption is, up to now, the only psycholinguistic contribution with a major impact on generation architectures, there are still other relevant factors which have been recognized by psycholinguists. Let us mention just the following:

1 *Biases*. A number of factors like *recency* (including priming; see above), *weight* (length), and *frequency*, seem to bias the sentence generation process. This is illustrated by the fact that people do have lexical and syntactic preferences despite possible alternatives. The existence of such biases requires an architecture that allows for multiple influences in the generation process, including those from system-internal constraints concerning computing resources.

2 *Specific processing strategies*. Both overt and covert repairs suggest that the human production system is able, at least to some extent, to employ strategies that boost efficiency, such as opportunistic planning and local replanning.

3 *Conscious higher-order vs. unconscious lower-order processes*. People are generally unconscious of low-level, language-specific levels of processing, whereas they are conscious of higher-level operations dealing with message planning at the prelinguistic stage, e.g. what rhetoric devices to use in order to achieve certain communicative goals. This suggests that the Conceptualizer has much more in common with other conscious cognitive activities requiring conscious problem solving than the Formulator, which may be a more specialized module.

4 *Learning*. Human speakers achieve proficiency through a long learning process, which has specific characteristics such as overgeneralization, overdiscrimination, etc. Systems which also exhibit these learning characteristics are psychologically more plausible than those that do not.

So far, these insights have hardly been taken into account by computer models for language generation, though they could provide some guidance for cognitively plausible generation architectures. On the one hand, one should be aware that certain phenomena may strongly depend on computing characteristics of our brains. On the other hand, rather than taking the computer as a metaphor for the human mind, one may take the brain as a metaphor for the computer. This is the approach followed by *neural network* architectures or *connectionism*. The biological brain possesses a great number of interacting elements (neurons) rather than a single processing unit. Although neurons have only limited computing power, together they are highly noise-resistant and contain enough redundancy to overcome some damage due to a loss of elements. The brain has self-organizing capacities which enables the spontaneous acquisition of functional properties.

Among the many connectionist architectures which have been designed and explored during the explosive growth of the field during the last decade, we mention two main approaches: *localist* connectionism, where a 'symbol' is represented as one activated atomic element or node, and *distributed* connectionism, where a 'symbol' is represented as an activation pattern over a large number of nodes. In either case, the activation patterns that flow through the system are not necessarily interpreted by a central processor (as in a traditional Von Neumann architecture), but can directly evoke other patterns of activity through network connections. Many connections act together in parallel, while each connection between two nodes contributes a little bit to the transformation from one activation state to the next. The influence of a connection is modulated by a *weight,* which can be either preprogrammed or set automatically by one of several possible learning rules (for more details, see Chapter 3 in Dijkstra & De Smedt, in press).

The characteristics of neural networks are appealing on intuitive grounds and are also in line with emprical data concerning natural language generation. Parallelism and competition are design foundations that have been argued for by many psycholinguists (for a review in the speech error literature, see e.g. Fromkin, 1980; Cutler, 1982; Garrett, 1982; Shattuck-Hufnagel, 1979; for interpretations within the connectionist framework, see Stemberger, 1985; Berg, 1988; Garman, 1990). Connectionist architectures are highly interactive. The multi-directional spreading of activation explains how top-down constraints may bias bottom-up processes (for empirical support, see Harley, 1982, 1984; Dell, 1985, 1986; Aitchison, 1987; Berg, 1988; for complementary linguistic evidence, see Danlos, 1987; Zock, 1988, 1990; Namer, 1990).

Only recently, natural language generation research has explored the connectionist paradigm and has been able to replicate some effects of human linguistic performance, including failsoft behaviour which is characterized by specific errors rather than complete breakdown. Interestingly, despite fundamental differences between the symbolic and connectionist approaches, there is at least one important point in

common: Both accept a kind of modularity in practice, even if the big advantage of connectionism is that it allows more integrated systems. Practically all connectionist models of natural language generation (Hasida, Ishizaki & Isahara, 1987; Kalita & Shastri, 1987; Kukich, 1987; Gasser & Dyer, 1988; Houghton, 1990; Kitano, 1990; Miikulainen, 1990; Ward, 1990; 1992; 1994) have used layers or strata akin to those proposed by linguists (pragmatics, semantics, syntax, phonology) and present in most 'conventional' systems.

Fully *distributed* neural nets have seen only limited use in natural language generation. Early experiments were done by Kukich (1987), who reimplemented a stock-market report generator in a three-layer feedforward network trained with *backpropagation* (Rumelhart, Hinton & Williams, 1986). Spanning crucial steps in linguistic realization, her system demonstrates not only that lexical choice and word order can be learned in a two-stage network model, but also that the trained network performs well even with incomplete input. Indeed, the capacity of these networks to perform relatively well under noisy conditions, given appropriate input and output representations, gives them a great advantage over most present-day generation systems, which are extremely vulnerable. Despite this fact, there is no single system that has addressed all the relevant issues in a fully distributed architecture. Kitano (1990) attempts simultaneous translation in a hybrid architecture where a parallel-marker passing scheme and a connectionist network coexist. This kind of hybrid approach leads to the question whether the symbolic and connectionist approaches could be reconciled, each one being responsible for a specific part of the generation process (cf. Bookman & Sun, 1993).

A powerful, but biologically less plausible way to overcome the limitations of feedforward networks is the inclusion of feedback links in the network architecture. In a system for parsing and generating sentences with relative clauses, Miikulainen (1990) proposes a *recurrent* network architecture (Elman, 1989), where the output of the hidden layer is fed back to the same layer at the next point in time. This creates a kind of short-term memory that provides context knowledge in encoding sequences.

Other researchers have used specific localist architectures (often called *interactive activation*) as the architectural basis, showing how the joint influence of many simple activation patterns can model the generation of complex syntactic structures (Kalita & Shastri, 1987; Ward, 1990; 1992; 1994). These systems show that it is possible to generate grammatically correct sequences without any explicit representation and manipulation of grammatical structures. Instead, a continuously changing activation pattern in a network with an *unchanging* topology determines the next word to be uttered at each moment in time. A variant of this architecture consists in the use of a *competitive filter* by Houghton (1990) which produces inhibitory feedback to the winning unit at the next point in time.

Connectionist architectures represent an important effort to overcome some of the limitations of traditional processing strategies and traditional representations of linguistic knowledge. Apart from simulating learning behaviour, connectionist architectures how greater robustness in dealing with incomplete input and incomplete knowledge sources, although this has not yet been shown convincingly for a wide span of the generation process. The gradual or partial breakdown of robust neural networks corresponds to psycholinguistic phenomena observed in aphasia, amnesia, speech

impairments, speech errors, and the 'tip of the tongue' phenomenon (Brown & Mc Neill, 1966). Connectionist models also have greater sensitivity to contextual effects, which makes them suitable candidates for modelling the biases mentioned above.

It is interesting to note that those who work within the connectionist framework try to account for psycholinguistic data, which is hardly ever the case in the traditional symbol manipulation paradigm (with the notable exception of the work done on incremental generation). The translations produced by Kitano's system are supported by transcripts from real simultaneous sessions. Ward's FIG model behaves in some ways like a human, creating similar kinds of speech errors. Even if real system designers may not be looking for computational psycholinguistic models that make errors, they may still be interested in the design of more optimized systems that show desirable psycholinguistic characteristics such as robustness.

However, while these psychologically motivated approaches offer a variety of hints to generator designers, the underlying concepts are expressed in a way that makes their transfer a tough task requiring quite a bit of imagination on the part of the engineer. The proposals are generally too imprecise to allow for formalization, the models account only for a limited range of phenomena and constructions, and the overall systematicity is not well enough developed so far (but see Dijkstra & Desmedt, in press).

For instance, the description of what information is available at the different levels of processing is simply too vague, and while the results of individual experiments are often insightful and convincing, little is done to clarify the relation between the different experiments to see the full picture emerge. Moreover, pragmatic aspects are widely underrepresented in psychologically motivated approaches, which is all the more unsatisfactory, given the pressing need for evidence concerning the role of pragmatics in natural language generation.

Nevertheless, psychologically motivated approaches should be influential for the design of generation systems in at least the following way. Psychological insights into the prominence of lexicalization and the stages of access to lexical material could have a strong bearing on text planning. If taken seriously, the text planning process should be more tightly coupled with the lexical choice component (i.e. the selection of *key* lexical items, conveying core meaning and important additional information like connotations). This departs from the strategy used in many of today's generators which involves numerous syntactic commitments due to the frequently occurring requirement that individual plan steps should be realizable by clause-sized pieces of text. In the approach we propose, the syntactic constraints associated with key lexical items are readily available to guide and constrain subsequent steps in the text planning process (cf. also Gross, 1986). Moreover, the factors biasing the sentence generation process could be systematically classified for domain-specific subsets and be used as a basis for evaluation metrices.

6 Conclusion

We have analysed and criticized some architectures of natural language generation systems and their underlying assumptions.

We felt that most current research in generation is too much geared towards classical computer architectures, and is insufficiently receptive to new developments in both AI and cognitive science. Language resembles many other cognitive tasks in that it is a knowledge-intensive process. At the same time it is special in many ways. Even though advanced research in knowledge-based systems offers powerful new solutions to known problems, the methodology inherited from those domains is rather abstract and may require further refinement in its application to flexible generation. But even then, there is no guarantee that the methodology will really work to the extent needed for efficient and reliable systems.

Newer architectures, based on the work done within the connectionist framework, are scarse and limited, though quite promising for generation. Even though it is not obvious whether systems using this approach can be scaled up to a usable size, the robustness characteristics of the activation-bases selection and distributed representation and processing are not only interesting from a psychological point of view, but also offer important advantages for practical applications.

To summarize, we believe that advances in knowledge-based systems, as well as work done by psycholinguists, will have an impact on innovative architectures in natural language generation. The issues involved are anything but trivial and will have to be faced by substantial research efforts. As a parallel development, we believe that generators will be increasingly embedded into compound systems to build multimedia and hypermedia applications. This is another very appealing, new direction, with clear architectural implications. But this falls outside the intended scope of this paper.

7 References

Abb, B., Guenther, C., Herweg, M., Lebeth, K., Maienborn, C. & Schopp, A. (1995). Incremental syntactic and phonological encoding: An outline of the Synphonics formulator. This volume.

Aitchison, J. (1987). *Words in the mind: An introduction to the mental lexicon*. Oxford: Blackwell.

André, E., Herzog, G. & Rist, Th. (1988). On the simultaneous interpretation of real world image sequences and their natural language description: The system SOCCER. In: Y. Kodratoff (Ed.), *Proceedings of the 8th European Conference on Artificial Intelligence* (pp. 449–454). London: Pitman.

Appelt, D. (1985). *Planning English sentences*. Cambridge: Cambridge University Press.

Bates, E. & Devescovi, A. (1989). Competition and sentence production. In B. MacWhinney & E. Bates (Eds.), *The crosslinguistic study of sentence processing* (pp. 225–253). Cambridge: Cambridge University Press.

Berg, T. (1988). *Die Abbildung des Sprachproduktionsprozesses in einem Aktivationsflußmodell: Untersuchungen an deutschen und englischen Versprechern* (Linguistische Arbeiten, 206). Tübingen: Niemeyer.

Bock, J.K. (1986). Meaning, sound, and syntax: Lexical priming in sentence production. *Journal of Experimental Psychology: Learning, Memory and Cognition, 12,* 575–586.

Bock, J.K. (1987). Exploring levels of processing in sentence production. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 351–363). Dordrecht: Nijhoff (Kluwer)

Bock, J.K. (in press). Sentence production: From mind to mouth. In J. L. Miller & P.D. Eimas (Ed.), *Handbook of perception and cognition. Vol. 11: Speech, language and communication*. Orlando, FL: Academic Press.

Bock, J.K. & Levelt, W.J.M. (1994). Language production: Grammatical encoding. In M.A. Gernsbacher (Ed.), *Handbook of psycholinguistics* (pp. 945–984). Orlando, FL: Academic Press.

Bookman, L. & Sun, R. (1993). Integrating neural and symbolic processes. *Connection Science, 5,* 203-204.

Brooks, R. (1991). Intelligence without representation. *Artificial Intelligence, 47,* 139–159.

Brown, R. & McNeill, D. (1966). The "tip of the tongue" phenomenon. *Journal of Verbal Learning and Verbal Behavior, 5,* 325–337.

Buchanan, B. & Shortliffe, E.H. (1984). *Rule-based expert systems: The MYCIN experiments of the Stanford Heuristic Programming Project*. Reading, MA: Addison-Wesley.

Busemann S. (1993). A holistic view of lexical choice. In H. Horacek & M. Zock (Eds.), *New concepts in natural language generation: Planning, realization, and systems* (pp. 202–308). London: Pinter.

Carberry, S., Chu, J., Green, N. & Lambert, L. (1993). Rhetorical relations: Necessary, but not sufficient. In O. Rambow (Ed.), *Proceedings of the Workshop on Intentionality and Structure in Discourse Relations, Columbus, OH, 21 June 1993* (pp. 1–4).

Cohen, P. & Levesque, H. (1985). Speech acts and rationality. *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics* (pp. 49–60).

Cutler, A. (Ed.), (1982). *Slips of the tongue and language production*. New York: Mouton.

Daelemans, W. & De Smedt, K. (1994). Default inheritance in an object-oriented representation of linguistic categories. *International Journal of Human-Computer Studies, 41,* 149–177.

Daelemans, W., De Smedt, K. & Gazdar, G. (1992). Inheritance in natural language processing. *Computational Linguistics, 18,* 205–218.

Daelemans, W., Gillis, S. & Durieux, G. (1994). The acquisition of stress: A data-oriented approach. *Computational Linguistics, 20,* 421–451.

Daelemans, W. & Powers, D. (1992). Background and experiments in machine learning of natural language. *Proceedings of the First SHOE Workshop*. Institute for Language Technology and Artificial Intelligence, Tilburg University.

Dalianis, H. & Hovy, E. (1995). Aggregation in natural language generation. This volume.

Danlos, L. (1984). Conceptual and linguistic decisions in generation. *Proceedings of the 10th International Conference on Computational Linguistics, Stanford, CA, 2–6 July 1984* (pp. 501–504).

Danlos, L. (1987). *The linguistic basis of text generation*. Cambridge: Cambridge University Press.

Dell, G. (1985). Positive feedback in hierarchical connectionist models: Applications to language production. *Cognitive Science, 9,* 3–23.

Dell, G. (1986). A spreading activation theory of retrieval in sentence production. *Psychological Review, 93,* 283–321.

Dell, G. & Reich, P. (1981). Stages in sentence production: An analysis of speech error data, *Journal of Verbal Learning and Verbal Behavior, 20,* 611–629.

De Smedt, K. (1990). IPF: An incremental parallel formulator. In R. Dale, C. Mellish & M. Zock (Eds.), *Current research in natural language generation* (pp. 167–192). London: Academic Press.

De Smedt, K. (1994). Parallelism in incremental sentence generation. In G. Adriaens & U. Hahn (Eds.), *Parallel natural language processing* (pp. 421–447). Norwood, NJ: Ablex.

De Smedt, K. & Kempen, G. (1991). Segment Grammar: A formalism for incremental sentence generation. In C.L. Paris, W.R. Swartout & W.C. Mann (Eds.), *Natural language generation in artificial intelligence and computational linguistics* (pp. 329-349). Dordrecht: Kluwer Academic Publishers.

Dijkstra, A. & De Smedt, K. (Eds.), (in press). *Computational psycholinguistics: AI and connectionist models of human language processing*. Hemel Hempstead: Prentice Hall / Harvester Wheatsheaf.

Elhadad, M. & Robin, J. (1992). Controlling content realization with functional unification grammars. In R. Dale, E. Hovy, D. Rosner & O. Stock (Eds.), *Aspects of automated natural language generation* (pp. 89–104). Berlin: Springer.

Elman, J.L. (1989). Structured representations and connectionist models. *Proceedings of the 11th Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum.

Fawcett, R.P., Tucker, G.H. & Lin, Y.Q. (1993). How a systemic functional grammar works: The role of realization in realization. In H. Horacek & M. Zock, (Eds.), *New concepts in natural language generation: Planning, realization and systems* (pp. 114–186). London: Pinter.

Finkler, W. & Neumann, G. (1989). POPEL-HOW: A distributed parallel model for incremental natural language production with feedback. *Proceedings of the 11th International Joint Conference on Artificial Intelligence, Detroit* (pp. 1518–1523).

Flower, L.F. (1981). *Problem-solving strategies for writing*. New York: Harcourt Brace Jovanovich.

Flower, L.F. & Hayes, J.R. (1984). Images, plans, and prose. The representation of meaning in writing. *Written Communication, 1 (4),* 120–160.

Fodor, J.A. (1983). *The modularity of mind*. Cambridge, MA: MIT Press.

Ford, M. (1982). Sentence planning units: Implications for the speaker's representation of meaningful relations underlying sentences. In J. Bresnan (Ed.), *The mental representation of grammatical relations* (pp. 797–827). Cambridge, MA: MIT Press.

Ford, M. & Holmes, V.M. (1978). Planning units in sentence production. *Cognition, 6,* 35–53.

Fromkin, V. (Ed.), (1980). *Errors in linguistic performance: Slips of the tongue, ear, pen and hand*. New York: Academic Press.

Gabriel, R. (1986). Deliberate writing. In D. McDonald & L. Bolc (Eds.), *Natural language generation systems* (pp. 1–46). Berlin: Springer.

Garman, M. (1990). *Psycholinguistics*. Cambridge: Cambridge University Press.

Garrett, M. (1975). The analysis of sentence production. In G. Bower (Ed.), *The psychology of learning and motivation* (Vol. 9, pp. 133–177). New York: Academic Press.

Garrett, M. (1980). Levels of processing in sentence production. In B. Butterworth (Ed.), *Language production* (Vol. 1, pp. 177–220). London: Academic Press.

Garrett, M. (1982). Production of speech: Observations form normal and pathological language use. In A. Ellis (Ed.), *Normality and pathology in cognitive functions* (pp. 19–76). London: Academic Press.

Gasser, M. & Dyer, M. (1988). Sequencing in a Connectionist Model of Language Processing. *Proceedings of the 12th International Conference on Computational Linguistics, Budapest* (pp. 185–190).

Gross, M. (1986). Lexicon-grammar: The representation of compound words. *Proceedings of the 11th International Conference on Computational Linguistics* (pp. 1-6).

Harley, T. (1982). There's more than one way… In *ECAI-82: Proceedings of the 5th European Conference on Artificial Intelligence* (pp. 267–268).

Harley, T. (1984). A critique of top-down independent models of speech production: Evidence from non-plan-internal speech errors. *Cognitive Science, 8,* 191–219.

Hasida, K., Ishizaki, S. & Isahara, H. (1987). An connectionist approach to the generation of abstracts. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 149–156). Dordrecht: Nijhoff (Kluwer).

Horacek, H. (1987). HOW to say WHAT – IT or SOMETHING. In K. Morik (Ed.), *Proceedings of the 11th German Workshop on Artificial Intelligence* (pp. 320–329). Berlin: Springer.

Horacek, H. (1990). The architecture of a generation component in a complete natural language system. In R. Dale, C. Mellish & M. Zock (Eds.), *Current research in natural language generation* (pp. 193–227). London: Academic Press.

Horacek, H. (1992). An integrated view of text planning. In R. Dale, E. Hovy, D. Rösner & O. Stock (Eds.), *Aspects of automated natural language generation* (pp. 29–44). Berlin: Springer.

Horacek, H. (1994). How to avoid explaining obvious things (without omitting central information). *Proceedings of the 11th European Conference on Artificial Intelligence, Amsterdam* (pp. 520–524).

Horacek, H. & Pyka, C. (1988). Towards bridging two levels of representation: Linking the syntactic-functional and object-oriented paradigms. In J.-L. Lassez & F. Chin (Eds.), *International Computer Science Conference'88 – Artificial Intelligence: Theory and Applications, Hong Kong* (pp.281–288).

Houghton, G. (1990). The problem of serial order: A neural network model of sequence learning and recall. In R. Dale, C. Mellish & M. Zock (Eds.), *Current research in natural language generation* (pp. 287–319). London: Academic Press.

Hovy, E. (1985). Integrating text planning and production in generation. *Proceedings of the 9th International Joint Conference on Artificial Intelligence, Los Angeles* (pp. 848-851).

Hovy, E. (1988a). Planning coherent multisentential text. *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics, Buffalo, NY* (pp. 163–169).

Hovy, E. (1988b). *Generating natural language under pragmatic constraints*. Hillsdale, NJ: Earlbaum.

Hovy, E. (1988c). Generating language with a phrasal lexicon. In D. McDonald & L. Bolc (Eds.), *Natural Language Generation Systems* (pp. 353–384). Berlin: Springer.

Hovy, E. (1990). Pragmatics and natural language generation. *Artificial Intelligence, 43,* 153–197.

Inui, K., Tokunaga, T. & Tanaka H. (1992). Text revision: A model and its implementation. In R. Dale, E. Hovy, D. Rösner & O. Stock (Eds.), *Aspects of automated natural language generation* (pp. 215–230). Berlin: Springer.

Jacobs, P.S. (1987). KING: A knowledge-intensive natural language generator. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 219–230). Dordrecht: Nijhoff (Kluwer).

Kalita, J. & Shastri, L. (1987). Generation of simple sentences in English using the connectionist model of computation. *Proceedings of the 9th Annual Conference of the Cognitive Science Society, Seattle* (pp. 555–565). Hove: Erlbaum.

Kay, M. (1979). Functional grammar. *Proceedings of the 5th Annual Meeting of the Berkeley Linguistic Society* (pp. 142–158).

Kay, M. (1984). Functional unification grammar: A formalism for machine translation. *Proceedings of COLING84, Stanford* (pp. 75–78).

Kempen, G. & Hoenkamp, E. (1987). An incremental procedural grammar for sentence formulation. *Cognitive Science, 11,* 201–258.

Kitano, H. (1990). Parallel incremental sentence production for a model of simultaneous interpretation. In R. Dale, C. Mellish & M. Zock (Eds.), *Current research in natural language generation* (pp. 321–351). London: Academic Press.

Kolk, H. (1987). A theory of grammatical impairment in aphasia. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 377–391). Dordrecht: Nijhoff (Kluwer).

Kukich, K. (1983). Design and implementation of a knowledge-based report generator. *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics, Cambridge, MA* (pp. 145–150).

Kukich, K. (1988). Fluency in natural language reports. In D. McDonald & L. Bolc (Eds.), *Natural language generation systems* (pp. 280–311). Berlin: Springer.

Kukich, K. (1987). Where do phrases come from: Some preliminary experiments in connectionist phrase generation. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 405–421). Dordrecht: Nijhoff (Kluwer).

Laird, J.E., Newell, A. & Rosenbloom, P.S. (1987). SOAR: An architecture for general intelligence. *Artificial Intelligence, 33,* 1–64.

Levelt, W.J.M. (1989). *Speaking: From intention to articulation*. Cambridge, MA: MIT Press.

Lewis, R.L. (1993). *An architecturally-based theory of sentence comprehension* (Computer Science Technical Report CMU–CS–93–226). Ph.D. thesis, Carnegie Mellon University.

Maes, P. (1986). Introspection in knowledge representation. *Proceedings of the 7th European Conference on Artificial Intelligence, Brighton* (pp. 256–269).

Maes, P. (1987). *Computational reflection* (Technical report 87–2). Brussels: Free University of Brussels, Artificial Intelligence Laboratory.

Maier, E. & Hovy, E. (1993). Organising discourse structure relations using metafunctions. In H. Horacek & M. Zock (Eds.), *New concepts in natural language generation: Planning, realization, and systems* (pp. 69–86). London: Pinter.

Mann, W. (1988). Text generation: The problem of text structure. In D. McDonald & L. Bolc (Eds.), *Natural language generation systems* (pp. 47–68). Berlin: Springer.

Mann, W. & Moore, J. (1981). Computer generation of multiparagraph English text. *American Journal of Computational Linguistics, 7,* 17–29.

Mann, W. & Thompson S. (1987). Rhetorical Structure Theory: Description and construction of text structures. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 85–95). Dordrecht: Nijhoff (Kluwer).

Mann, W. & Thompson S. (1988). Rhetorical Structure Theory: Toward a functional theory of text organization. *Text, 8,* 243–281.

Marcus, M. (1987). Generation systems should choose their words. *Proceedings of the 3rd Conference on Theoretical Issues in Natural Language Processing (TINLAP-3), Las Cruces* (pp. 211–214).

McDonald, D. (1983). Natural language generation as a computational problem: An introduction. In M. Brady & R. Berwick (Eds.), *Computational models of discourse* (pp. 209-266). Cambridge, MA: MIT Press.

McDonald, D. (1987). Natural language generation. In S. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence* (Vol. 1, pp. 642–655). New York: John Wiley.

McDonald, D. & Pustejovsky, J. (1985). A computational theory of prose style for natural language generation. *Proceedings of the 2nd Conference of the European chapter of the Association for Computational Linguistics, Genève* (pp. 187–193).

McDonald, D., Vaughan, M. & Pustejovsky, J. (1987). Factors contributing to efficiency in natural language generation. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 159–181). Dordrecht: Nijhoff (Kluwer).

McKeown, K. (1982). The TEXT system for natural language generation: An overview. *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics, Toronto* (pp. 113–120).

McKeown, K. (1985). *Text generation using discourse strategies and focus constraints to generate natural language text*. Cambridge: Cambridge University Press.

McKeown, K. & Swartout, W. (1988). In M. Zock & G. Sabah (Eds.). *Advances in natural language generation: An interdisciplinary perspective* (pp. 1–51). London: Pinter.

Meteer, M. (1990). *Expressibility and the problem of efficient text planning*. London: Pinter.

Miikulainen, R. (1990). A PDP architecture for processing sentences with relative clauses. *Proceedings of the 13th International Conference on Computational Linguistics, Helsinki* (Vol. 3, pp. 201–206).

Moore, J.D. & Paris, C.L. (1993). Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational Linguistics, 19,* 651–694.

Moore, J. & Pollack, M. (1992). A problem for RST: The need for multi-level discourse analysis. *Computational Linguistics, 18,* 537–544.

Motley, M.T, Camden, C.T. & Baars, B.J. (1982). Covert formulation and editing of anomalies in speech production: Evidence from experimentally elicited slips of the tongue. *Journal of Verbal Learning and Verbal Behavior, 21,* 578–594.

Namer, F. (1990). *Pronominalisation et effacement du sujet en génération automatique de textes en langues romanes*. Doctoral dissertation, Université de Paris 7.

Neumann, G. & Finkler, W. (1990). A head-driven approach to incremental and parallel generation of syntactic structures. *Proceedings of the 13th International Conference on Computational Linguistics, Helsinki* (Vol. 2, pp. 288–293).

Nirenburg, S. & Nirenburg, I. (1988). A framework for lexical selection in natural language generation. *Proceedings of the 12th International Conference on Computational Linguistics, Budapest* (pp. 471–475).

Nirenburg, S., Lesser, V. & Nyberg, E. (1989). Controlling a language generation planner. *Proceedings of the 11th International Joint Conference on Artificial Intelligence* (pp. 1524–1530).

Nogier, J.F. & Zock, M. (1992). Lexical choice by pattern matching. *Knowledge Based Systems, 5,* 200–212.

Novak, H.-J. (1987a). Strategies for generating coherent descriptions of object movements in street scenes. In G. Kempen (Ed.), *Natural language generation: New results in artificial intelligence, psychology and linguistics* (pp. 117–132). Dordrecht: Nijhoff (Kluwer).

Novak, H.-J. (1987b). *Textgenerierung von visuellen Daten: Beschreibungen von Straßenszenen*. Berlin: Springer.

Patten, T. (1988). *Systemic text generation as problem solving*. Cambridge: Cambridge University Press.

Reithinger, N. (1991). POPEL – A parallel and incremental natural language generation system. In C.L. Paris, W.R. Swartout & W.C. Mann (Eds.), *Natural language generation in artificial intelligence and computational linguistics* (pp. 179–199). Boston: Kluwer Academic Publishers.

Reithinger, N. (1992). The performance of an incremental generation component in multi-modal dialog contributions. In R. Dale, E. Hovy, D. Rösner & O. Stock (Eds.), *Aspects of automated natural language generation* (pp. 263–276). Berlin: Springer.

Robin, J. (1993). A revision-based generation architecture for reporting facts in their historical context. In H. Horacek & M. Zock (Eds.), *New concepts in natural language generation: Planning, realization, and systems* (pp. 238–268). London: Pinter.

Rösner, D. & Stede, M. (1992). TECHDOC: A system for the automatic production of multilingual technical documents. In G. Görz (Ed.), *Proceedings of KONVENS-92* (pp. 329–338). Berlin: Springer.

Rubinoff, R. (1992). Integrating text planning and linguistic choice by annotating linguistic structures. In R. Dale, E. Hovy, D. Rösner & O. Stock (Eds.), *Aspects of automated natural language generation* (pp. 45–56). Berlin: Springer.

Rubinoff, R. & Lehrman, J.F. (1994). Real-time natural language generation in NL-SOAR. *Proceedings of the 7th International Workshop on Natural Language Generation, Kennebunkport, Maine* (pp. 199–206).

Rumelhart, D.E., Hinton, G.E. & Williams, R.J. (1986). Learning internal representations by error propagation. In J.L. McClelland & D.E. Rumelhart (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1* (pp. 318–362). Cambridge, MA: MIT Press.

Saint-Dizier, P. (1992). A constraint logic programming treatment of syntactic choice in natural language generation. In R. Dale, E. Hovy, D. Rosner & O. Stock (Eds.), *Aspects of automated natural language generation* (pp. 119–134). Berlin: Springer.

Shattuck-Hufnagel, S. (1979). Speech errors as evidence for a serial ordering mechanism in sentence production. In W.E. Cooper & E.C.T. Walker (Eds.), *Sentence processing: Psychological studies presented in honor to M. Garrett* (pp. 295–342). Hillsdale, NJ: Erlbaum.

Stede, M. (1993). Lexical choice criteria in language generation. *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 454–459).

Stede, M. (1995). Lexical options in multilingual generation from a knowledge base. This volume.

Steels, L. (1990). Components of Expertise. *AI Magazine, 11(2),* 29–49.

Stemberger, J. (1985). An interactive activation model of language production. In A.W. Ellis (Ed.), *Progress in the psychology of language* (Vol. 1, pp. 143–186). Hove: Erlbaum.

Stemberger, J. (1990). Wordshape errors in language production. *Cognition, 35,* 123-157.

Thompson, H. (1977). Strategy and tactics: A model for language production. In W. Beach, S. Fox & S. Philosoph (Eds.), *Papers from the 13th Regional Meeting of the Chicago Linguistics Society* (pp. 651–668).

Valian, V. (1977) Talk, talk, talk: A selective critical review of theories of speech production. In R. Freedle (Ed.), *Discourse production and comprehension* (pp. 107–139). Norwood, NJ: Ablex.

Wanner, L. (1992). Lexical choice and the organization of lexical resources in text generation. In B. Neumann (Ed.), *Proceedings of the 10th European Conference on Artificial Intelligence, Vienna* (pp. 495–499). New York: Wiley.

Wanner, L. (1994). On lexically biased discourse organization in text generation. *Proceedings of the 27th International Conference on Computational Linguistics (COLING-94), Kyoto* (pp. 369–375).

Ward, N. (1990). A connectionist treatment of grammar for generation. *Proceedings of the 5th International Workshop on Natural Language Generation, Dawson, PA* (pp. 15–22).

Ward, N. (1992). A parallel approach to syntax for generation. *Artificial Intelligence, 57,* 183–225.

Ward, N. (1994). *A connectionist language generator*. Norwood, NJ: Ablex.

Zock, M. (1988). Natural languages are flexible tools, that's what makes them hard to explain, to learn and to use. In M. Zock & G. Sabah (Eds.) *Advances in natural language generation: An interdisciplinary perspective* (pp. 181–196). London: Pinter.

Zock, M. (1990). If you can't open the black box, open a window! A psycholinguistically-motivated architecture of a natural language generation component. In *COGNITIVA-90, Madrid* (pp. 143–152).

Zock, M. (1993). Is content generation a one-shot process or a cyclical acitivity of gradual refinement: The case of lexical choice. In H. Horacek & M. Zock (Eds.), *New concepts in natural language generation: Planning, realization, and systems* (pp. 290–296). London: Pinter.