

## Oxford Handbooks Online

### **Natural Language Generation**

John Bateman and Michael Zock

The Oxford Handbook of Computational Linguistics 2nd edition (2 ed.)

*Edited by Ruslan Mitkov*

Subject: Linguistics, Computational Linguistics Online Publication Date: Apr 2016

DOI: 10.1093/oxfordhb/9780199573691.013.010

### **Abstract and Keywords**

Communication via a natural language requires two fundamental skills: producing 'text' (written or spoken) and understanding it. This chapter introduces newcomers to computational approaches to the former—natural language generation (henceforth NLG)—showing some of the theoretical and practical problems that linguists, computer scientists, and psychologists encounter when trying to explain how language production works in machines or in our minds. The chapter first defines and illustrates the abstract components of the NLG task and their distinctive roles in accounting for the coherence and appropriateness of natural texts and then sets out the principal methods that have been developed in the field for building working computational systems. Current problems, new proposals for solutions and potential applications are also briefly characterized.

Keywords: natural language generation, deep generation, surface generation, rhetorical structure, lexicalization, text planning, reference expression generation, statistical generation

---

## **1 General Introduction: What Is NLG? (Cognitive, Linguistic, and Social Dimensions)**

## 1.1 NLG—a knowledge-intensive problem

Producing language is a tightly integrated cognitive, social, and physical activity. We speak to solve problems, for others or for ourselves, and in order to do so we make certain choices under specific space, time, and situational constraints. The corresponding task of natural language generation (NLG) therefore spans a wide spectrum, ranging from planning some action (verbal or not) to executing it (verbalization). This can be characterized in terms of mapping information from some *non-linguistic* source (e.g. raw data from a knowledge base or scene) into some corresponding linguistic form (text in oral or written form) in order to fulfil some non-linguistic goal(s). This ‘transformation’ is neither direct nor straightforward, and bridging the gap between the non-linguistic input and its linguistic counterpart involves many decisions or choices.

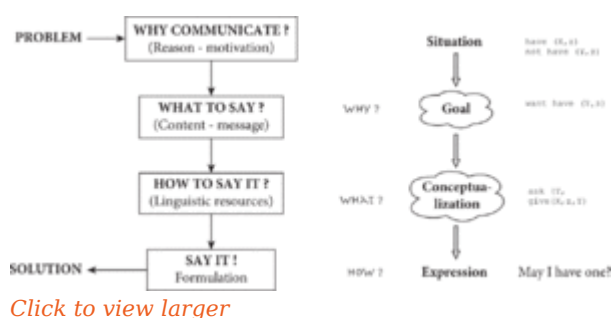


Figure 1 NLG: A simplest view

The field of NLG typically considers these choices to include the determination and structuring of *content* (i.e. choice of the message) and that content’s presentation in terms of *rhetorical organization* at various levels (text, paragraph, sentence) as

well as the choice of the appropriate *words* and *syntactic structures* (word order, constructions, morphology), and the determination of *text layouts* (title, headers, footnotes, etc.) or *acoustic patterns* (prosody, pitch, intonation contour). Providing theoretical and computational architectures in which these diverse decisions can be orchestrated so as to yield natural-sounding/reading texts within a reasonable amount of time is rightfully regarded as one of the major challenges facing the field. An initial schematic view of this process and its various stages is suggested in Figure 1.

While everybody speaks a language, not everybody speaks it equally well. There are substantial differences concerning its speed of learning and its ease and success of use. How language works in our mind is still in many respects a mystery (Levelt 1989), and some consequently consider the construction of NLG systems as a further useful methodology for helping to unravel that mystery. Others see NLG as an approach to solving practical problems—such as contributing to the synthesis side of *machine translation* (cf. Chapter 32), to the production side of *spoken dialogue systems* (cf. Chapter 41), to *automated writing assistance* (cf. Chapter 44), to *text summarization* (cf. Chapter 37), to *natural-language processing for educational applications* (cf. Chapter 43), and to *multimodal systems* (cf. Chapter 42). That our understanding of the process remains fragmentary is largely due to the *quantity*, the *diversity*, and the *interdependence* of the choices involved.

While it is easy to understand the relationships between a glass falling and its breaking (causal relationship), it is not at all easy to understand the dependency relationships holding in heterarchically organized systems like biology, society, or natural language. In such systems, the consequences of a choice may be multiple, far-reaching, and unpredictable. Moreover, there will be multiple and apparently flexible interdependencies holding across choices. In this sense, then, there are many points in common between speaking a language well, and hence communicating effectively, and being a good politician. In both cases one has to make the right choice at the right moment. Learning *what* the choices are, and *when* they should be made, is then a large part of the task of building effective NLG systems.

### 1.2 What is language?—a functional perspective

Characterizing more finely just what the decisions for mastering language production are is therefore a further major challenge of its own. NLG needs to combine many sources of knowledge: *knowledge of the domain* (what to say, relevance), *knowledge of the language* (lexicon, grammar, semantics), *strategic rhetorical knowledge* (how to achieve communicative goals, text types, style), *knowledge of how to achieve discourse coherence*, and much more. Moreover, building successful NLG systems requires *engineering knowledge* (how to decompose, represent, and orchestrate the processing of all this information) as well as *knowledge* about the *characteristics, habits, and constraints* of the end user (listener, reader) in order to determine just what kinds of produced language will be appropriate.

The complexity of the knowledge-intensive, flexible, and highly context-sensitive process evidently constituting NLG is revealed particularly clearly when we consider the production of connected texts rather than isolated sentences. Producing connected texts, rather than isolated sentences, is a core activity defining NLG as a field and leading to particular styles of approaches distinguishing it from other areas of computational linguistics.

Consider the following example. Suppose you were to express the idea of some population of people leaving the place where they live. First, to describe this state of affairs more independently of language choices, we might employ a logical expression of the form: [LEAVE (POPULATION, PLACE)], i.e., a simple predicate holding over two arguments. It is then instructive to watch what happens if one systematically varies the expression of the different concepts *leave*, *population*, and *place* by using either different *words* (*abandon*, *desert*, *leave*, *go away from* in the case of the verb, and *place* or *city* in the case of the noun), or different *grammatical resources*: a *definite description* ('the + N'), *possessives* ('yours', 'its'), etc. Concretely, consider the five continuation variants given here.

“X-town was a blooming city. Yet, when the hooligans started to invade the *place*

- (a) the place was abandoned by ((its/the) population)/them.
- (b) the city was abandoned by its/the population.
- (c) it was abandoned by its/the population.
- (d) its/the population abandoned the city.
- (e) its/the population abandoned it.

The place was not liveable anymore.”

The interested reader may perform all the kinds of variations mentioned above and check to what extent they affect *grammaticality* (the sentence cannot be uttered or finished), *clarity* (some pronouns create ambiguity), *cohesion*, and *rhetorical effect*. In particular, while all the candidate sentences we offer in (a)–(e) are basically well-formed, each one has a specific effect, and not all of them are equally felicitous. Some are ruled out by virtue of poor textual choices, others, because of highlighting the wrong element, or because of wrong assignment of the informational status (given-new) of a given element. For example, in (a) ‘the place’ is suboptimal, since it immediately repeats a word, and in (d) ‘the city’ is marked as ‘minimal’ new information, while actually being known, i.e. old information. Probably the best option here is (c) since this preserves the given-new distribution appropriately, without introducing potentially ambiguous pronouns (see Chapter 27 and our section discussing ‘reference generation’).

Getting the text ‘right’ therefore constitutes a major problem. Even though we have considered rather few of the options available, the apparent independence of the choices involved leads directly to a combinatorial explosion. Thus, even were we to have a reasonably well-developed grammar, it is the *use* of that grammar that remains crucial. This means that the notion of ‘grammaticality’, central to formal approaches to language (see, particularly, Chapters 2 and 4) is on its own by no means sufficient and many other factors, such as social, discourse, and pragmatic constraints (Chapters 6 and 7), have to be taken into account in order to assure *successful communication*. In short, effective NLG calls for some way of taking account of the *effects* of individual linguistic choices—not only locally, but also globally for the text to which they are contributing. Both choices and effects need to be orchestrated in order to collectively achieve the speaker’s overall goals. Considering communication as a goal-oriented activity embedded in some context is one of the defining characteristics of **functional theories** of language (e.g. Halliday

and Matthiessen 2004). Since NLG is clearly of this sort (it is a means towards an end), we can see why such theories have had a far stronger influence on the NLG community than in most other areas of computational linguistics.

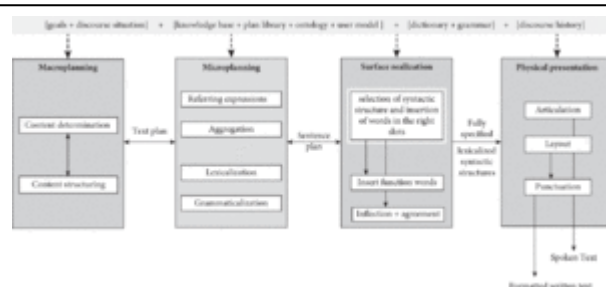
### 1.3 Decomposing the NLG task: Architectures for NLG

Architectures, both abstract and implemented, deal with the functional relations between components (dependencies, control of information flow) or the problem of *what* to process *when* (order). Given the number and diversity of components, and given the complexity of their interactions and dependencies, finding optimal computational architectures for NLG is difficult. Implementation decisions here revolve around problems of *order* (uni- vs bi-directional generation), *completion* (one pass vs several; revisions), *control* (central vs distributed), etc. and have employed sequential, parallel, integrative, revision-based, blackboard, and connectionist methods. Detailed discussions as well as some attempts at standardization are offered by de Smedt et al. (1996) and Mellish et al. (2006).

As a practical solution, however, many systems designers opt for a unidirectional flow of information between components (the *pipelined* view of Reiter 1994). This offers considerable engineering benefits (e.g. ease of implementation, speed of generation) but nevertheless also poses a number of problems. Most prominent among these is the **generation gap** (Meteer 1992). This is the danger of not being able to find resources (e.g. words) to express some message because of some suboptimal decision made prior to this point. Put differently, a local decision may well lead to a potential deadlock at the global level, preventing you from expressing what still needs to be said because of incompatibility between the choices (e.g. not all verbs can be passivized) or because of lack of available grammatical and lexical resources. In a pipelined unidirectional system there is then no opportunity for recovering from this situation (backtracking) and so typically the entire generation process will either need to be restarted or some other error-handling mechanism invoked.

A more general way of characterizing the range of options available when addressing NLG system design is to consider orientations to the NLG problem as a whole. The diversity of interdependencies between choices and components has also led to differing conceptualizations of what NLG is, resulting in at least three kinds of definition:

1. NLG as a *mapping problem*
2. NLG as *problem of choice*
3. NLG as a *planning problem*



[Click to view larger](#)

Figure 2 Commonly assumed components/tasks of the NLG process

All of these involve some decomposition of the problem space into different kinds of representational layers, the most common of which are those shown in Figure 2. Here the four broad tasks of the NLG process (macro- or text-planning, micro- or sentence-planning, linguistic realization, presentation) are further divided into a series of subtasks. At the top of the figure there are reoccurring situational constraints and knowledge sources (dictionary, grammar) that commonly apply at several stages in processing; the boxes in Figure 2 represent the (sub)tasks at the various levels.

These four main tasks may be described as follows:

- **Macroplanning** comprises *content choice* and *document structuring*. The former decides what information to communicate explicitly in the text, given the interlocutors' goals, knowledge, beliefs, and interests; the latter deals with message clustering and message order to produce a thematically coherent whole that does not give rise to unwanted inferences ('She got pregnant, they married.' vs 'They married. She got pregnant.'). Cue words (*because*, *nevertheless*, etc.) may subsequently be added to reveal or clarify the rhetorical role (*cause* vs *concession*) of the various conceptual fragments, i.e. clauses: 'He arrived just in time (*because/despite*) of the heavy traffic'. The result of macroplanning is generally represented as hierarchical tree structures in which the leaves represent the messages to be expressed (clauses or sentences) and their communicative status (primary/secondary; see *nucleus/satellite* in section 2.3), and the arcs represent the *rhetorical relations* (cause, sequence, etc.) holding between them.
- **Microplanning** covers the generation of referring expressions, lexicalization, and aggregation. **Reference generation** involves producing descriptions of any object referred to in such a way as to allow the hearer to distinguish that object from potential alternatives (the big car, the truck, it). **Lexicalization** consists in finding the form (lemma) corresponding to some concept (DOG: canine, puppy) or in choosing among a set of alternatives. More sophisticated lexicalization approaches attempt to segment semantic space (e.g. a graph representing the messages to convey) so as to allow integration of the resulting conceptual fragments within a sentence or paragraph without falling foul of the **generation gap**. And, finally, **aggregation** is the process of grouping together similar entities or events in order to minimize redundancy of expression.
- **Realization** consists in converting abstract representations of sentences into concrete text, both at the **language** level (*linguistic realization*, involving grammar,

lexical information, morphology, etc.) and the layout level (*document realization*), with abstract text chunks (sections, paragraphs, etc.) often being signalled via mark-up symbols.

- **Physical presentation** then finally performs final articulation, punctuation and layouting operations as appropriate for a selected output medium.

## 2 Content Selection and Discourse Organization: Macroplanning

### 2.1 Content planning

Before starting to talk, one generally has something to say—at least at some level of abstraction and with some communicative goal. The crux of the problem is then to find out how this ‘something’—the conceptual input to be expressed by the surface generator—is determined. Obviously, given some topic, we will not say everything we know about it, neither will we present the messages in the order in which they come to mind; we have to perform certain organizational operations first: for example, we must elaborate (underspecified messages), concentrate (generalize specific messages), focus (emphasize or de-emphasize), and, if necessary, rearrange (change order). But what guides these operations?

Suppose you had the task of writing a survey paper on NLG or a report on the weather. You could start from a knowledge base containing information (facts) on *authors, type of work, year, etc.* or *meteorological information* (numbers). Since there is no point in mentioning all the facts, you have to filter out those that are irrelevant. *Relevance* here means being sensitive to readers’ goals, knowledge, preferences, and point of view. Moreover, because the knowledge to be expressed (message) may be in a form that is fairly remote from its corresponding linguistic expression (surface form)—for example, raw numerical data or qualitative representations—it may first need to be *interpreted* in order to specify its semantic value: e.g. a sequence of decreasing numeric values might be **expressed** as ‘*a drop*’, ‘*a decrease*’, or ‘*the falling of temperature*’. The knowledge to be expressed will also generally require further organization since raw data, even in a knowledge base, rarely has the kind of structure that can be used directly for structuring a text.

Data can be organized in many ways, alphabetically, chronologically, etc., while many types of texts exhibit functional, topical, or procedural structures: for our ‘survey paper’ task that may be: deep generation, surface generation, lexical choice, etc., or for the ‘weather report’: the state of the weather today, predictions about changes, past statistics, etc. This requires further information, either explicitly (e.g. an *ontology*



classifying and interrelating the objects, processes, and qualities in some domain; cf. Chapter 20), or implicitly, that is, via inference rules (which in that case are needed in addition to the knowledge base).

An important function of these latter kinds of organizational knowledge is to enable the text producer to select and structure information according to criteria that may not be explicitly represented in the data being expressed. For example, in our survey, we may organize according to *pioneering work*, *landmark systems*, etc. Such criteria then need to be operationalized in order to play a role in an NLG system: for example, one could define the notion of ‘landmark’ as work that has changed the framework within which a task is being carried out (e.g. the move from *schemata* to *RST* in discourse planning as we discuss in section 2.3), or as a paradigm shift that has since been adopted by the field (as in the increased use of statistical methods that we consider in section 4), and so on. Precisely which information is extracted also then correlates with issues of topic structure and thematic development. This close link between *content selection* and *discourse structure* is a natural one: information is generally only relevant for a text given some particular communicative goals. For this reason, these two tasks are often addressed by the same component.

2.2 Text planning: The schema-based approach and the TEXT system

Name of predicate	Function	Example
IDENTIFICATION	identifies the object as a member of a class	<i>A catamaran is a kind of sailboat.</i>
CONSTITUENCY	presents constituents of the entity	<i>A sailboat has sails and a mast.</i>
ILLUSTRATION	provides an example	<i>The Titanic is a boat.</i>

[Click to view larger](#)

Figure 3 Examples of rhetorical predicates

To illustrate text planning in more detail, we briefly consider one of the earliest systems to automatically produce paragraph-length discourse: **TEXT** (McKeown

1985). McKeown analysed a large number of texts and transcripts and noticed that, given some *communicative goal*, people tended to present the *same kind of information* in a canonical *order*. To capture these regularities, McKeown classified this information into **rhetorical predicates** and **schemata** (Figure 4)). The former describe roughly the semantics of a text’s building blocks; examples of predicates, their function, and illustrative clauses are shown in Figure 3. The latter describe the legal combination of predicates to form patterns or text templates. Following McKeown, Figure 4) illustrates several ways of decomposing the *Identification* schema. Figure 4) then gives an example text; connections between the sentences of the text and the text schema elements responsible are indicated by sentence numbering. Schemata are thus both *organizational devices* (determining *what* to say *when*) and *rhetorical means*, i.e. discourse strategies for achieving some associated goals. Since schemata are easy to build and use, they are still commonly used in text generation.



(A) Identification ( <i>class &amp; attribute</i> (1) / <i>function</i> )	(1) A Hobie Cat is a <i>brand of catamarans, manufactured by the Hobie Company.</i>
(B) [ <i>Analogy/Constituency/Attributive</i> (2) / <i>Renaming</i> ]*	(2) Its main attraction is that it's <i>cheap.</i>
(C) <i>Particular-illustration</i> (3) / <i>Evidence</i> +	(3) A <i>new one</i> goes for about \$5000.
(D) [ <i>Amplification/Analogy/Attributive</i> ]	
(E) [ <i>Particular-illustration/Evidence</i> ]	
Key: {} optional; / alternative; * optional item which may appear 0 to n times; + item may appear 1 to n times. The underlined predicates are the ones that are actually used in the text produced (see below). (a): Identification schema	(b): A corresponding text

[Click to view larger](#)

Figure 4 (a) Identification schema and (b) a corresponding text

Once the *building blocks* and their *combinations* are known, simple text generation (i.e. without microplanning) is fairly straightforward. Given a goal (define, describe, or compare), the system chooses a schema that stipulates in abstract

terms *what* is to be said *when*. Whenever there are several options for continuing—for example, in the second sentence one could have used *analogy*, *constituency*, or *renaming* instead of the predicate *attributive* (cf. Figure 4), schema B)—the system uses **focus rules** to pick the one that ties in best with the text produced so far. If necessary, a schema can also lead on recursively to embedded schemata. Coherence or text structure as a whole is then achieved as a side effect of choosing a goal and filling its associated schema.

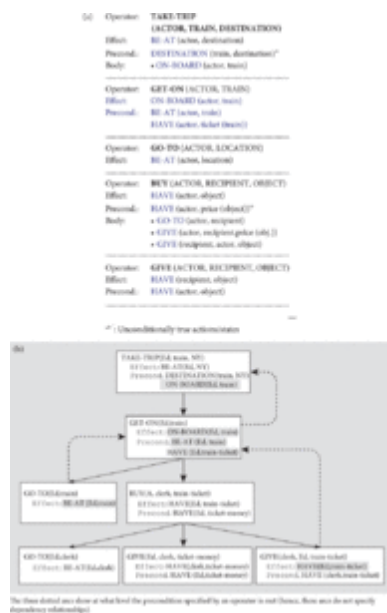
Not all information stipulated by the schema appears in the final text since only schema parts (A) and (C) are compulsory. A given predicate may also be expanded and repeated. Optionality, repetition, and recursion together make schemata very powerful, although they also have some recognized shortcomings—the most significant of which is the lack of connection between the *components* of an adopted schema and the goals of a text. The goals associated with a schema specify the role of the schema as a whole, but they do not specify the roles of its parts: i.e., why use a specific rhetorical predicate at a given moment?

This is particularly problematic in the case of failure: for example, if the text is being generated as part of a dialogue system, or if information required subsequently is not found. The system cannot then recover in order to offer an alternative solution because there is no linking between schema parts and any subgoals for the text as a whole: the schema-driven system will invariably produce the same answer, regardless of the user's expertise, informational needs, or problems, unless such constraints are explicitly built into the schema selection process. Another problem with schemata arises from the flexibility of natural-language use: in many communicative situations language producers diverge considerably from a straightforward scheme and this then also needs to be dealt with when considering language generation that is to be judged 'natural'.

2.3 Text planning and Rhetorical Structure Theory (RST)

**Rhetorical Structure Theory** (Mann and Thompson 1988) was adopted in NLG partly to overcome the problems of schemata just mentioned by providing a more flexible mechanism that closely links communicative goals and text structure. According to RST, any coherent text is decomposable into a recursive structure of text 'spans' (usually

clauses) related via a small set of rhetorical relations ('cause', 'purpose', 'motivation', 'enablement', and so on). The relations between the spans are often implicit, dividing each text span into at least two segments: one that is obligatory and of primary importance—the **nucleus**—and one which plays a supportive role—the **satellite**. The existence of a single overarching rhetorical structure for a text is taken as an explanation of that text's perceived coherence. RST received its initial computational operationalization in the late 1980s and has since been incorporated in a wide range of NLG systems (Hovy 1993); probably the most widespread version of operationalized RST is still that presented in Moore and Paris (1993), many variations of which have been produced since.



[Click to view larger](#)

Figure 5 (a) Library of planning operators; (b) A partial plan to achieve the goal 'get to NY'

RST in this form makes essential use of the *planning paradigm* from AI (Sacerdoti 1977).

**Planning** here means basically *organizing actions rationally* in order to *achieve a goal*. Since most goals (problems) are complex, they have to be decomposed. High-level, global goals are thus refined to a point where the actions associated with them are primitive enough to be performed directly. Planning supposes three things: a **goal** (problem to

be solved), a **plan library** (i.e. a set of plans, each of which allows the achievement of a given goal), and a **planning method** (algorithm). A **plan** is a schema composed of the following information: an **operator**, that labels the particular kind of change on the 'world' carried out; an **effect**, i.e. a state which holds true after the plan's execution, the goal; optional **preconditions**, which must be satisfied before the plan can be executed; and a **body**, i.e. a set of actions, which are the *means* for achieving the goal. Somewhat simplified, then, a hierarchical planner works by decomposing the top-level goal using plan operators whose effects match the goals given and whose preconditions hold. These plan operators may introduce further subgoals recursively. Planning continues until 'primitive actions' are reached; i.e., actions that can be directly performed without further decomposition. Several examples of such planning operators, their definitions, and their use in constructing a plan are given in Figure 5.

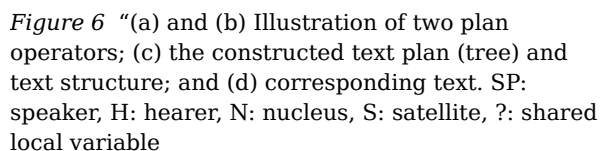
Let us take an example to illustrate how this works. Suppose 'Ed' wanted to go to 'New York'. We assume that this can be represented logically in terms of some goal such as: [BE-AT (ACTOR, DESTINATION)], with the arguments instantiated as required for our particular case. Now, rather than building a plan that holds only for one specific problem, one usually appeals to a generic plan—e.g., a set of plans and solutions for a related body of problems. In Figure 5), for example, we show a **library** of plan operators appropriate for the planning of trips for different people to go to different places. Given our present starting goal of wanting to get Ed to New York, the planner looks for operators to achieve this, i.e. operators whose *effect* fields match the *goal*.

In our case the top-level goal can partially be achieved in a number of ways: for example, via the TAKE-TRIP or the GO-TO operator; however, there are additional dependency relationships and states of affairs that need to be fulfilled and these restrict the ordering of applicability of the plan operators. Thus, we cannot take a trip unless we get on a train, etc. Assuming that the text planner has got as far as proposing the taking of a trip, then the body of the operator is *posted* as a further subgoal: GET-ON. Associated with this further operator are two conditions: [BE-AT (ACTOR, TRAIN)] and [HAVE (ACTOR, TICKET)] which can be satisfied via the GO-TO and BUY operators respectively. The first one is considered unconditional—it can be directly achieved—while the second one decomposes into three actions: *go to the clerk*, *hand over the money*, and *receive the ticket*. It is this last action that ensures that the traveller (actor) has finally what is needed to take the trip: the ticket (precondition of the GET-ON operator). The process terminates when all goals (regardless of their level) have been fulfilled, which means all effects or preconditions hold true, either unconditionally—the action being primitive enough to dispense with any further action (see the GO-TO operator)—or because there is an operator allowing their realization. Figure 5) shows the partially completed hierarchical plan for achieving the final goal.

The use of this planning paradigm for operationalized RST is quite natural: rhetorical relations are modelled as plan operators and their effects become communicative goals. Text planning then operates in precisely the same way as general planning as just illustrated. To show this, we take an example slightly adapted from Vander Linden (2000) concerning a system providing automated help or documentation for a computer program. Several NLG systems have been built for this kind of scenario. Suppose then that a user would like to know how to save a file. For the NLG system to generate an appropriate text, it is first given a top-level goal, for example:

(COMPETENT H (DO-ACTION SAVING-A-FILE))

That is: enable the Hearer (H) to achieve the state of knowing how to 'save-a-file'. Planning then proceeds as above, but employing a plan library in which the operators implement the definitions of rhetorical relations as offered by RST. As before, the planning process 'bottoms out' when it reaches **primitive planning actions**, i.e. actions that can be directly performed. In the NLG case, such actions are assumed to be simple utterances, i.e. individual messages associated with simple speech acts, such as 'inform',



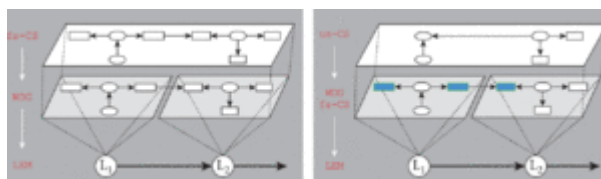
‘saving a file’; then the constraints slot specifies that this action must be decomposable into some substeps and that this sublist should not consist of a single action. If we assume that this is the case for our present example—saving a file will typically require several substeps—then the operator will apply. Note that this process not only checks constraints but also *retrieves* relevant information from the system knowledge base via matching and binding of values for the local variable, thereby combining content selection with its textual ordering.

The subgoals specified in the nucleus and satellite slots are then posted and must both in turn be satisfied for planning as a whole to succeed. The satellite subgoal succeeds immediately, since it calls for the system, or speaker (SP), to inform (a primitive act) the hearer (H) about some propositional content (<DO 'save a file'>). Moreover, this entire content is embedded as a satellite under an RST 'purpose' relation, which then constrains its possible linguistic realizations; one possible realization of this branch of the text plan, provided by the surface generator on demand, is given in (1) in Figure 6). The goal posted under the nucleus requires further expansion, however, and so the planning process continues (cf., e.g., Figure 6): 'expand sequence'. The result of the process is a tree whose *nodes* represent the rhetorical relations (effect), and whose *leaves* represent the verbal material allowing their achievement. This result is then handed either to the microplanner (e.g. for aggregation, which may eliminate redundancies) or to the surface generator directly. Figures 6(c, d) show a final text plan constructed by the planning operators and a possible corresponding text respectively.

## 3 Microplanning and Realization

Having sketched in section 1 some of the subtasks of microplanning, we will specifically address the central task of **lexicalization** in more detail, since without succeeding in this task, no text whatsoever can be produced.

### 3.1 Lexicalization



[Click to view larger](#)

Figure 7 (a) The lexicon as mediator between the conceptual level and word level; (b) The lexicon as a means for refining the initial message

Somewhere in a generation system, there must be information about the words that would allow us to express what we want to convey, their meanings, their syntactic constraints, etc. This kind of information is stored in

a **lexicon** (see Chapter 3) whose precise organization can vary considerably (for examples, see Chapter 17). Dictionaries are *static knowledge*, yet this knowledge still has to be used or ‘activated’: i.e., words have to be accessed and, in case of alternatives (synonyms), some selection has to be made. All this contributes to what is called *lexicalization*. There are two broad views concerning this task: one can conceive it either as *conceptually driven* (meaning) or as *lexicon-driven* (see Figures 7a and 7b).

In the former view, everything is given with the input (i.e. the message is *complete*), and the relationship between the conceptual structure and the corresponding linguistic form is *mediated* via the lexicon: lexical items are selected, provided that their underlying content covers parts of the conceptual input. The goal is to find sufficient, mutually compatible lexical items so as to completely cover the input with minimal unwanted additional information. In the latter view, the message to be expressed is incomplete prior to lexicalization, the lexicon serving, among other things, to refine the initially underspecified message (for arguments in favour of this second view, see Zock 1996). One begins with a skeleton plan (rough outline or gist) and fleshes it out progressively depending on arising needs (for example, by providing further information necessary for establishing *reference*).

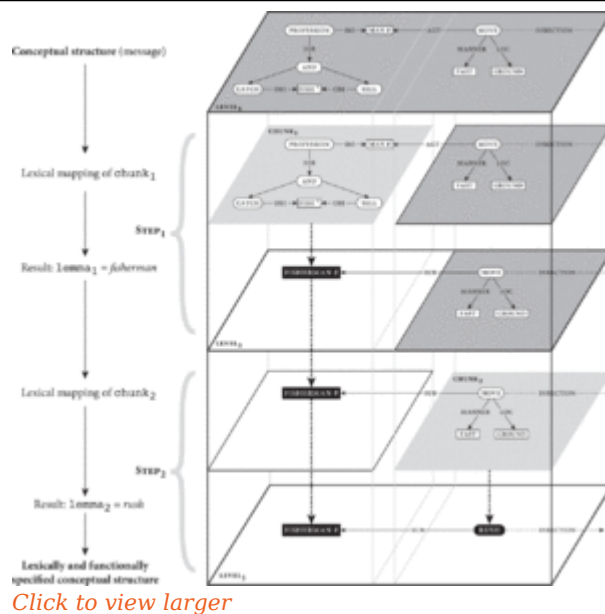


Figure 8 Progressive lexical specification

To illustrate this first approach (conceptually driven lexicalization), we adapt an example from Nogier and Zock (1992). Suppose your goal were to find the most precise and economic way for expressing some content, say, ‘the fisherman rushed to the canoe’; the problem is to decide how to carve out the conceptual space such as to cover everything planned. We will show here only how the first two open-class

words (*fisherman* and *rush*) are selected, suggesting by the *direction* link that more is to come. Lexicalization is performed in two steps. During the first, only those words are selected that pertain to a given semantic field (for example, movement verbs). In the next step the lexicalizer selects from this pool the term that best expresses the intended meaning, i.e. the most specific term (maximal coverage). Thus we need to map the two conceptual chunks (input) *a man whose profession consists in catching and selling fish* and *move fast on the ground* into their linguistic counterparts: *fisherman* and *rush*. The process is shown in Figure 8; for a sophisticated proposal of what to do if one cannot find a complete covering, see Nicolov et al. (1997).

Lexicalization is one area where the work done by psychologists is particularly relevant for knowledge engineers. Psycholinguists have run a considerable number of experiments over the years to study lexical access, i.e. the *structure* and *process* of the mental lexicon. There are several approaches: while connectionist approaches (Levelt et al. 1999; Dell et al. 1999) model the time course of getting from ideas to linguistic forms (lemma), lexical networks like WordNet (Miller 1990) deal more with the organization of the lexicon. The insights gained via such studies are relevant not only for the simulation of the mental processes underlying natural-language production, but also for improving navigation in electronic dictionaries (cf. Zock et al. 2010).

### 3.2 Surface generation

The final stage of NLG proper consists in passing the results of the microplanner to the **surface generator** to produce strings of properly inflected words; while some surface generators produce simple text strings, others produce strings marked up with tags that can be interpreted for prosody control for spoken language, for punctuation, or for layout. Unlike in situations of theoretical grammar development, the primary task of a surface generator is to find the most appropriate way of expressing the given communicative content rather than finding all possible forms. Hence the solution found should be in line not only with the rules of the language, but also with the text genre, the speakers' goals and the rest of the text generated (the *co-text*).

Surface generation has traditionally been the area of NLG that overlaps most with the established concerns of 'core' linguistics—particularly structural linguistics aiming for accounts of sentences (see Chapters 2–5 devoted to morphology, syntax, semantics, and the lexicon). Although nearly all linguistic theories have been implemented in trial generation and analysis systems, the number of frameworks actively employed in generation systems is much more restricted. For extended text, where functional control of the textual options is central, accounts based on *systemic-functional grammars* (SFG: Halliday and Matthiessen 2004) have often been employed; for dialogue systems, where not only real-time but also *bidirectional* resources are an issue (i.e. grammars that can be used for both generation and analysis), *combinatory categorical grammars* (CCG: Steedman 2000) with their close coupling of semantics and syntax have established themselves, taking over a role previously enjoyed by *tree-adjoining grammars* (TAG); for general grammar engineering, *head-driven phrase structure grammars* (HPSG) continue to receive considerable attention; and for speedy development, *template-based grammars* of various kinds remain common since they require little effort to write—although at the cost of lack of reusability and scalability (Becker and Busemann 1999; van Deemter et al. 2012).

Several large-scale generation systems providing reusable technology for NLG systems have been constructed drawing on these and other frameworks. Freely available functionally based tactical generation is provided by systems such as KPML (Bateman 1997), while CCG-based development is provided by the OpenCCG toolset (<http://openccg.sourceforge.net/>). Pointers to all of these frameworks and systems are provided in the NLG Systems Wiki (see the section Further Reading).

## 4 Current Issues, Problems, and Opportunities

In certain respects, the field of NLG has reached a limited state of maturity. The techniques for producing natural language in particular contexts of application are reasonably well understood and there are several, more or less off-the-shelf components











that can be employed. None of these components, however, provides capabilities that allow the developer to simply ‘generate’: some fine-tuning of components and resources is always necessary. Debate continues, therefore, concerning how to achieve such fine-tuning. The large-scale, purpose-independent generation components, particularly for surface generation, achieved during the 1990s and developed further in the 2000s, require a certain degree of grammar engineering for each new application they are used for. This requires that the developer extends grammatical capabilities, provides more lexical information, performs domain-modelling tasks, and so on in order to make the kinds of planning and surface generation techniques described above work. Common criticisms of such approaches are therefore that one requires expert (computational) linguistic knowledge to apply them and that their performance is restricted to the areas for which they have been developed.

As an alternative, many researchers now attempt to replace or augment the functionalities described above with techniques relying on the increasingly sophisticated statistical methods that have been developed in natural-language processing since the late 1990s (cf. Chapters 11–12). Although approaches of this kind are being explored for most of the component tasks of the NLG process (cf. for text organization: Wang 2006; Bollegala et al. 2010; Zock and Tesfaye 2013), the primary domain of application for statistical methods until now has been tactical generation. Such approaches work by first deriving distributions of grammatical and lexical resources from corpora of naturally occurring language data exhibiting the kind of linguistic constructions or linguistic variability desired for a generation system, and then using the resulting language models to avoid much of the fine-grained decision-making traditionally required. One of the first systems of this kind was described in Knight and Hatzivassiloglou (1995), who proposed a basic framework for statistical generation still employed today. The basic idea of such systems is to allow the tactical generator to radically overgenerate—thereby reducing the load of making possibly computationally expensive ‘correct’ decisions—and then to rely on statistically derived language models to select the best alternatives.

Knight and Hatzivassiloglou offer a simple illustration of this process by considering preposition choice. In more traditional, non-statistical tactical generation the information concerning the choice of preposition in the contrasting phrases ‘She left *at* five’ / ‘She left *on* Monday’ / ‘She left *in* February’ would require detailed functional motivations or dedicated lexical or semantic information in order to select the appropriate preposition. In the statistical approach, the tactical generator simply produces all of the alternatives with varied prepositions (and many other options as well, generally represented as a word lattice rather than explicitly producing each of the strings) and then consults its statistical model of likely word strings in order to select the statistically most probable. In many cases, this already gives sufficient information to weed out inappropriate selections, making the detailed decision process unnecessary. As we saw at the outset, however, the number of choices available in natural language is considerable and so overgeneration brings its own range of problems for managing the large number of strings potentially produced. Standard linguistic problems, such as ‘long-distance dependencies’ (cf. Chapter 4), also present issues for statistical models. As a

consequence, most statistically based systems developed since also consider syntactic information so that the main challenge then becomes how to combine syntactic constraints with statistically derived language models (cf. DeVault et al. 2008). Just what the best mix of statistical and non-statistical methods will be is very much an open issue.

Statistical approaches have also been applied to counter the pipeline problem introduced in section 1.3. One particularly effective approach here is that of pCRU (Belz 2007). In this framework, an entire generation system is seen as a collection of choices irrespective of the components those choices are drawn from. The choices are then ranked with probabilities drawn with respect to a designated target corpus. This is beneficial when generating texts of a particular style and at the same time avoids some of the problems inherent in pipelined approaches since there is no pre-set order of decisions. The pCRU approach can also be combined with a variety of generation techniques and so is a promising direction for future investigation; Dethlefs and Cuayáhuitl (2010), for example, describe an architecture combining pCRU, reinforcement learning, and SFG-based generation with the KPML system.

Object of thought	Alternatives	Linguistic expression	Discrimination factor
		the white one	COLOUR
		the round one	SHAPE
	  	the round white one	COLOUR + SHAPE

[Click to view larger](#)

Figure 9 Examples of referring expression generation

Statistically driven systems have moved NLG towards potential applications in many areas and this has made it natural that attention within the NLG community turn more to questions of *evaluating* alternatives (cf. Chapter 15). The

considerable diversity of approaches and theoretical assumptions and lack of agreed input and output forms have made this a complex issue within NLG. However, certain generation tasks have now become *de facto* ‘standards’ that any NLG system can be expected to address and so have provided the first candidates for **shared tasks** around which evaluation procedures can be built. The longest established of these is the generation of **referring expressions** (Dale 1992), where the task is one of providing an appropriately discriminating referring expression successfully identifying a referent from a collection of ‘distractors’. This task can either be text-internal with respect to some knowledge base or draw on external stimuli, such as a set of objects in the visual field. Some examples are given in Figure 9; several workshops have been devoted to evaluation of this kind (see, for example, <<http://bridging.uvt.nl/news-events.html>> or <<http://www.macs.hw.ac.uk/InteractionLab/refnet/>>), as well as open contests comparing systems and approaches (see, for example, <<http://www.abdn.ac.uk/ncs/departments/computing-science/tuna-318.php>>) For surveys from a computational linguistic or psycholinguistic point of view, see Krahmer and van Deemter (2012) or van Deemter et al. (2012). Other tasks being explored include scene description, question generation (cf. the INLG Generation Challenges 2010: Belz, Gatt, and Koller 2010), and word access (Rapp and Zock 2014). Krahmer and Theune (2010) provide a quite extensive collection of

approaches to NLG that draw on corpus data in order to derive ‘empirically’ motivated, or *data-driven*, NLG accounts. The problems being addressed in this way are now beginning to range across all the levels of the NLG task introduced in this chapter. This includes the use of natural interaction data to derive NLG components specifically appropriate for building natural language dialogue systems, an area where considerable flexibility is required. The deployment of NLG techniques in the context of interactive systems more generally is thus currently of considerable interest, as reported in detail for several application domains by Stent and Bangalore (2014).

Finally, there are several further application directions currently being explored where NLG capabilities play a central role and where we can expect substantial developments. One area receiving growing attention is that of generating language for the Semantic Web, i.e. from linked data (Duma and Klein 2013) or from information maintained in **ontologies** (cf. Chapter 20) within the Semantic Web (Androutsopoulos et al. 2013; Mellish and Pan 2008; Schütte 2009). This is actually a natural goal for NLG since one of the bottlenecks in applying NLG technology has always been the lack of suitably structured and rich knowledge sources from which to generate challenging texts. The provision of large-scale ontologies within the Semantic Web solves this problem automatically, although there are still significant issues to be faced brought about by the mismatch between the kind of knowledge organization available within the Semantic Web and that required for the NLG architectures.

## Further Reading and Relevant Resources

There are several excellent introductory articles to NLG that take more technical views of both the tasks of NLG and the approaches that have attempted to deal with them; see, for example, Vander Linden (2000), McDonald (2000) and the many references cited there, as well as the textbook on building NLG systems from Reiter and Dale (2000). A detailed account of text planning using rhetorical relations is given by Hovy (1993). Stede (1999) provides an excellent covering work on lexical choice. For relevant psychological approaches at the sentence and discourse level, see Levelt (1989), Bock (1995), and Andriessen et al. (1996). For access to recent publications, reference lists, free software, or to know *who is who*, etc., the best way to start is to go to the website of the ACL’s Special Interest Group for Generation (SIGGEN: <<http://www.siggen.org/>>). There are a number of freely available generation systems and generation grammars that can be used for hands-on exposure to the issues of NLG. For multilingual generation, the KPML generation system at <<http://purl.org/net/kpml>> includes an extensive grammar development environment for large-scale systemic-functional grammar work and teaching, as well as a growing range of generation grammars: e.g. for English (very large), Spanish, Chinese, German, Dutch, Czech, Russian, Bulgarian, Portuguese, and

French. Finally, an almost complete list of NLG systems with linked bibliographical and web-based information is maintained in a wiki at <http://www.nlg-wiki.org/systems/>.

## References

Andriessen, Jerry, Koenraad de Smedt, and Michael Zock (1996). 'Discourse Planning: Empirical Research and Computer Models'. In T. Dijkstra and K. de Smedt (eds), *Computational Psycholinguistics: AI and Connectionist Models of Human Language Processing*, 247-278. London: Taylor & Francis.

Androutsopoulos, Ion, Gerasimos Lampouras, and Dimitrios Galanis (2013). 'Generating Natural Language Descriptions from OWL Ontologies: The NaturalOWL System', *Journal of Artificial Intelligence Research* 48: 671-715.

Bateman, John A. (1997). 'Enabling Technology for Multilingual Natural Language Generation: The KPML Development Environment', *Journal of Natural Language Engineering* 3(1): 15-55.

Becker, Tim and Stephan Busemann (eds) (1999). 'May I Speak Freely? Between Templates and Free Choice in Natural Language Generation'. In *Proceedings of the Workshop at the 23rd German Annual Conference for Artificial Intelligence (KI '99)*, 14-15 September, Bonn, DFKI Document D-99-01.

Belz, Anja (2007). 'Automatic Generation of Weather Forecast Texts Using Comprehensive Probabilistic Generation-Space Models', *Natural Language Engineering* 14(4): 431-455.

Belz, Anja, Albert Gatt, and Alexander Koller (2010). 'Section: INLG Generation Challenges 2010'. In *Proceedings of the 6th International Conference on Natural Language Generation (INLG)*, 7-9 July, Dublin, 229-274. Stroudsburg, PA: Association for Computational Linguistics.

Bock, Kathryn (1995). 'Sentence Production: From Mind to Mouth'. In J. Miller and P. Eimas (eds), *Handbook of Perception and Cognition, xi: Speech, Language, and Communication*, 181-216. Orlando, FL: Academic Press.

Bollegala, Danushka, Naoaki Okazaki, and Mitsuru Ishizuka (2010). 'A Bottom-Up Approach to Sentence Ordering for Multi-Document Summarization', *Information Processing & Management* 46(1): 89-109.

Dale, Robert (1992). *Generating Referring Expressions: Constructing Descriptions in a Domain of Objects and Processes*. Cambridge, MA: Bradford Books and MIT Press.

Deemter, Kees van, Albert Gatt, Roger van Gompel, and Emiel Krahmer (2012). 'Towards a Computational Psycholinguistics of Reference Production', *Topics in Cognitive Science* 4(2): 166-183.

Dell, Gary, Franklin Chang, and Zenzi M. Griffin (1999). 'Connectionist Models of Language Production: Lexical Access and Grammatical Encoding', *Cognitive Science* 23: 517-542.

Dethlefs, Nina and Heriberto Cuayáhuitl (2010). 'Hierarchical Reinforcement Learning for Adaptive Text Generation'. In *Proceedings of the 6th International Conference on Natural Language Generation (INLG)*, 7-9 July, Dublin, 34-46. Stroudsburg, PA: Association for Computational Linguistics.

DeVault, David, David Traum, and Ron Artstein (2008). 'Making Grammar-Based Generation Easier to Deploy in Dialogue Systems'. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue*, Columbus, OH, 198-207. Stroudsburg, PA: Association for Computational Linguistics.

Duma, Daniel and Ewan Klein (2013). 'Generating Natural Language from Linked Data: Unsupervised Template Extraction'. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, Potsdam, 83-94. Stroudsburg, PA: Association for Computational Linguistics.

Halliday, Michael and Christian Matthiessen (2004). *An Introduction to Functional Grammar*, 3rd edition. London: Arnold.

Hovy, Eduard (1993). 'Automated Discourse Generation Using Discourse Structure Relations', *Artificial Intelligence* 63: 341-385.

Knight, Kevin and Vasileios Hatzivassiloglou (1995). 'Two-Level, Many Paths Generation'. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL95)*, 252-260. Stroudsburg, PA: Association for Computational Linguistics.

Krahmer, Emiel and Kees van Deemter (2012). 'Computational Generation of Referring Expressions: A Survey', *Computational Linguistics* 38(1): 173-218.

Krahmer, Emiel and Mariët Theune (eds) (2010). *Empirical Methods in Natural Language Generation: Data-oriented Methods and Empirical Evaluation*. New York: Springer Verlag.

Levelt, Willem (1989). *Speaking: From Intention to Articulation*. Cambridge, MA: MIT Press.

Levelt, Willem, Ardie Roelofs, and Antje Meyer (1999). 'A Theory of Lexical Access in Speech Production', *Behavioral and Brain Sciences* 22: 1-75.

Mann, William and Sandra Thompson (1988). 'Rhetorical Structure Theory: Toward a Functional Theory of Text Organization', *Text* 8(3): 243-281.

McDonald, David (2000). 'Natural Language Generation'. In R. Dale, H. Moisl, and H. Somers (eds), *A Handbook of Natural Language Processing Techniques*, 147–299. New York: Marcel Dekker.

McKeown, Kathleen (1985). 'Discourse Strategies for Generating Natural-Language Text', *Artificial Intelligence* 27: 1–41.

Mellish, Chris and Jeff Pan (2008). 'Natural Language Directed Inference from Ontologies', *Artificial Intelligence* 292: 1285–1315.

Mellish, Chris, Donia Scott, Cahill Lynne, Roger Evans, Daniel Paiva, and Mike Reape (2006). 'A Reference Architecture for Natural Language Generation Systems', *Natural Language Engineering* 12(1): 1–34. Cambridge University Press.

Meteer, Marie (1992). *Expressibility and the Problem of Efficient Text Planning*. London: Pinter Publishers.

Miller, George (1990). 'WordNet: An On-Line Lexical Database', *International Journal of Lexicography* 3(4): 235–312.

Moore, Johanna and Cécile Paris (1993). 'Planning Texts for Advisory Dialogs: Capturing Intentional and Rhetorical Information', *Computational Linguistics* 19(4): 651–694.

Nicolov, Nicolas, Chris Mellish, and Graeme Ritchie (1997). 'Approximate Chart Generation from Non-hierarchical Representations'. In R. Mitkov and N. Nicolov (eds), *Recent Advances in Natural Language Processing*, 273–294. Amsterdam: John Benjamins.

Nogier, Jean-François and Michael Zock (1992). 'Lexical Choice by Pattern Matching', *Knowledge-Based Systems* 5(3): 200–212.

Rapp, Reinhard and Michael Zock (2014). 'The Cogalex-IV Shared Task on the Lexical Access Problem'. In Michael Zock, Reinhard Rapp, and Chu-Ren Huang (eds), *Proceedings of the 4th Workshop on Cognitive Aspects of the Lexicon*, 23 August, Dublin, Ireland, 1–14. Stroudsburg, PA: Association for Computational Linguistics.

Reiter, Ehud (1994). 'Has a consensus NL generation architecture appeared, and is it psychologically plausible?'. In David McDonald and Marie Meteer (eds), *Proceedings of the 7th. International Workshop on Natural Language Generation (INLGW '94)*, Kennebunkport, Maine, 163–170. Stroudsburg, PA: Association for Computational Linguistics.

Reiter, Ehud and Robert Dale (2000). *Building Natural Language Generation Systems*. Cambridge: Cambridge University Press.

Sacerdoti, Earl (1977). *A Structure for Plans and Behavior*. Amsterdam: North Holland.

Schütte, Niels (2009). 'Generating Natural Language Descriptions of Ontology Concepts'. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLWG)*, 30–31 March, Athens, Greece, 106–109. Stroudsburg, PA: Association for Computational Linguistics.

Smedt, Koenraad de, Helmut Horacek, and Michael Zock (1996). 'Architectures for Natural Language Generation: Problems and Perspectives'. In G. Adorni and M. Zock (eds), *Trends in Natural Language Generation: An Artificial Intelligence Perspective*, 17–46. New York: Springer Verlag.

Stede, Manfred (1999). *Lexical Semantics and Knowledge Representation in Multilingual Text Generation*. Dordrecht: Kluwer.

Steedman, Mark (2000). *The Syntactic Process*. Cambridge, MA: MIT Press.

Stent, Amanda and Srinivas Bangalore (eds) (2014). *Natural Language Generation in Interactive Systems*. Cambridge: Cambridge University Press.

Vander Linden, Keith (2000). 'Natural Language Generation'. In Daniel Jurafsky and James H. Martin (eds), *Speech and Language Processing: An Introduction to Speech Recognition, Computational Linguistics and Natural Language Processing*, 763–798. Englewood Cliffs, NJ: Prentice Hall.

Wang, Yang-Wendy (2006). 'Sentence Ordering for Multi-Document Summarization in Response to Multiple Queries'. Master thesis, Simon Fraser University.

Zock, Michael (1996). 'The Power of Words in Message Planning'. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING 1996)*, 5–9 August, Copenhagen, 990–995. Stroudsburg, PA: Association for Computational Linguistics.

Zock, Michael, Olivier Ferret, and Didier Schwab (2010). 'Deliberate Word Access: An Intuition, a Roadmap and Some Preliminary Empirical Results', *International Journal of Speech Technology* 13(4): 201–218.

Zock, Michael and Debela Tesfaye (2013). 'Automatic Sentence Clustering to Help Authors to Structure their Thoughts'. In *Proceedings of the 10th International Workshop on Natural Language Processing and Cognitive Science (NLPCS 2013)*, 15–16 October, Marseille, France, 35–49. Stroudsburg, PA: Association for Computational Linguistics.

### John Bateman

John Bateman is Professor of Applied Linguistics at the University of Bremen, Germany. Since obtaining his Ph.D. from the Department of Artificial Intelligence at the University of Edinburgh in 1985, he has worked in natural language generation and functional linguistics, applying the latter to the former, on projects in Scotland,



Japan, California, and Germany, as well as in a variety of European cooperations. His main research focuses are multilingual NLG, multimodal document design, discourse structure, and the application of all areas of systemic-functional linguistics.

### **Michael Zock**

Michael Zock is Emeritus Research Director of the CNRS and is currently affiliated with University of Marseille, France (LIF-AMU). He holds a Ph.D. in experimental psychology. Having initiated (1987, Royaumont) the European Workshop Natural Language Generation workshop series, he has edited five books on language generation. His major research interests lie in the building of tools to help humans to produce language (speaking/writing) in their mother tongue or when learning a foreign language. His recent work is devoted to the navigation in electronic dictionaries, aiming at overcoming the tip-of-the-tongue problem, and facilitating the access, memorization, and automation of words and syntactic structures.



Access is brought to you by