Adapting SimpleNLG to German

Marcel Bollmann
Department of Linguistics
Ruhr-University Bochum
44780 Bochum, Germany
bollmann@linguistics.rub.de

Abstract

This paper describes SimpleNLG for German, a surface realisation engine for German based on SimpleNLG (Gatt and Reiter, 2009). Several features of the syntax of German and their implementation within the current framework are discussed, with a special focus on word order phenomena. Grammatical coverage of the system is demonstrated by means of selected examples.

1 Introduction

Surface realisation is the task of generating natural language sentences from semantic input representations. The final step in any realisation process is the mapping of representations of syntactic structures to well-formed output strings, while considering the grammar rules of the target language. This includes, but is not limited to, correctly inflecting words and applying punctuation and sentence orthography. As these tasks are rather mechanical and a necessary part of every NLG application, developers can greatly benefit from a realisation engine which specialises on this process and implements it in an easy and intuitive way. SimpleNLG, as described in Gatt and Reiter (2009), is a realisation engine for English that fulfills this description. This paper results from an effort to adapt the SimpleNLG engine to German.

In SimpleNLG, sentences are constructed by combining `LexicalItem` objects and `PhraseSpec`s, which represent various phrasal subtypes, in a modular way. Canned text can always be used interchangeably with non-canned representations, while specifics of the final realisation are controlled via features.

German, when compared to English, displays features that make an adaption of the framework a non-trivial task. First, the German inflectional system is much richer, calling for a more systematic way of describing inflection classes and generating inflected word forms. On a related note, there are many more agreement phenomena to be taken care of. As word order in German is much freer than in English, the need for reordering of constituents arises; e.g., the subject can no longer unconditionally be realised at the beginning of a sentence. This area was a special focus for this implementation.

SimpleNLG for German is a Java framework derived from version 3.8 of SimpleNLG[1]; as of June 2011, version 4.3 has been released. While there are plans to port this package to the new 4.x architecture, all claims about SimpleNLG in this paper refer to the older version.

2 Characteristics of German

This section discusses some of the changes made to the SimpleNLG system to account for German grammar rules. Section 2.1 describes the implementation of inflection, while section 2.2 discusses topics related to agreement. Section 2.3 deals with the issue of word order, which prompted fundamental changes to the system, and section 2.4 explains changes regarding modal verbs.

[1] The original SimpleNLG is available from: `http://code.google.com/p/simplenlg/`

Proceedings of the 13th European Workshop on Natural Language Generation (ENLG), pages 133–138,
Nancy, France, September 2011.

2.1 Inflection

As inflection in English is very limited, SimpleNLG properly inflects most English words using a set of regular expressions; the use of a lexicon is possible, but not required. German, on the other hand, has a rich inflectional system that requires knowledge of the inflection class for each word, thus increasing the importance of a lexicon.

In SimpleNLG for German, inflection is encapsulated in separate classes called *inflection patterns*, which largely resemble inflection classes from traditional grammars, e.g. Eisenberg (2004). Technically, an inflection pattern stores an array of suffixes and provides methods to append them to a stem; the types and number of suffixes depends on the respective part of speech. Additionally, a pattern can have a number of features which influence the inflection process. Plural umlaut for nouns is a prominent example for this, as is *'e'* elision in certain stems ending in *–el/er*:

- *sammeln* 'to collect' → *ich sammle* 'I collect'

Special consideration is required for verbs with separable prefixes, as stem and prefix can appear both joined and separated:

- *ankommen* 'to arrive' → *ich komme an* 'I arrive'

Therefore, a verb prefix is always stored separately from the base verb. The inflection class of separable verbs can always be derived from the base verb alone, without considering the prefix. This is of particular relevance for the lexicon, which only needs an entry for the base verb in order to be able to generate all combinations of separable prefixes with that verb. Separable verbs can be instantiated by placing a vertical bar between the base verb and the prefix (e.g. `an|kommen`). The boundary has to be specified manually by the user, as there are several verbs which are ambiguous in this regard: e.g., *umfahren* is separable in the meaning of 'to knock (something) over', but inseparable in the meaning of 'to drive around (something)'.

Compound nouns are a similar case: the inflection class of a compound is equal to that of its final stem. Also, as compounding is a highly productive morphological process in German, it is not feasible to list every single compound in the lexicon. Therefore, compounding has been implemented in the same way: e.g., specifying `Heimat|stadt` 'hometown' creates a compound derived from *Stadt* 'town'.

Lexicon files are currently stored in XML format; for testing and evaluation purposes, lexicon entries were imported from IMSLex (Fitschen, 2004).

2.2 Agreement

Agreement in English is mostly confined to the 3. SG. PRES. IND. suffix *–(e)s* for verbs. In German, there is a more distinct subject–verb agreement, but also other types of agreement, e.g. determiner–noun or adjective–noun. This section discusses two topics related to agreement: the problems arising with use of canned text, and the agreement of relative pronouns.

2.2.1 Canned text

In SimpleNLG, canned text can be used interchangeably with lexical items and phrase specifications. This functionality is retained in SimpleNLG for German, as it is fundamental to the "simple" aspect of the framework. Proper nouns are typical candidates to be represented by canned text, as they show no or minimal inflection and can not generally be expected to be found in a lexicon. However, in German, this approach is problematic, as the following examples show:

(1) *beim FC Liverpool*
at.the FC Liverpool

(2) *bei der Eintracht Frankfurt*
at the Eintracht Frankfurt

Here, the names of football clubs are used together with a definite article, which agrees with the following noun in gender. Note that the article in (1) is contracted with the preposition. The examples show that even proper nouns referring to abstract concepts, such as football clubs, can be assigned different genders in German. Gender information, however, is not available when working with canned text. Consequently, whenever gender information is required—for example, when combining a proper noun with a specifier, or replacing it with a pronoun—simple canned text can not be used. Instead, a new lexical item has to be manually constructed from the canned text, and assigned

the appropriate gender value. This undermines the simplicity of the system to some extent, but also highlights an intrinsic difficulty in adapting the SimpleNLG approach to languages with richer agreement morphology.

2.2.2 Relative clauses

Relative pronouns in German agree with the antecedent in gender and number, while also inflecting for case based on their function within the relative clause. To make their creation as simple as possible, explicit support for relative clauses has been added.

The `NPPhraseSpec` class now provides a method which requires a sentence (to be embedded as the relative clause) and the function of its head noun in the relative clause. The process of constructing the relative clause then consists of two steps. First, a relative pronoun is generated, referring to the head noun; this includes setting the appropriate agreement features (gender and number). Second, the relative pronoun is inserted into the sentence with the specified grammatical function; this ensures the correct case value.

(3) *die Frau, [auf die] ich stolz bin*
the woman of whom I proud am
'the woman of whom I am proud'

(4) *die Frau, [deren Kind] schön ist*
the woman whose child beautiful is
'the woman whose child is beautiful'

More complex embeddings can also be realised this way: if a preposition is given instead of a grammatical function, a new prepositional phrase is constructed, with the relative pronoun as its head. This is shown in (3). To generate (4), a noun phrase is required to which the relative pronoun is added as a specifier. All of these functions have in common that they facilitate the creation of relative clauses for the user, as they take care of mechanical steps (e.g., selecting the correct pronoun to ensure agreement), highlighting the "simple" aspect of the framework.

Relative clauses can still be constructed manually, without the use of these helper methods. This requires more code, but is actually useful, as the resulting relative clause can be embedded in phrases other than the NP containing the antecedent, thereby enabling relative clause extraposition.

2.3 Word order

German, in contrast to Modern English, has verb-second word order, i.e. the verb always has to be the second constituent in main clauses. Verb-initial and verb-final sentences are also possible, which had to be accounted for in SimpleNLG for German. More interesting, however, is the order of non-verb constituents, which is relatively free when compared to English. The examples below show that it is possible for every non-verb constituent to appear at the front of a sentence, though the order of constituents after the verb is variable, too. The preferred word order depends on many factors, which can be syntactic, semantic, or pragmatic in nature, and is therefore hard to determine automatically. Also, different word orders can, for example, be used to emphasise certain constituents. It is therefore desirable for a generation system to be able to realise these variants, which was a main focus for this implementation.

(5) *Die Frau gab dem Mann gestern*
the woman gave the man yesterday
ein Buch.
a book
'Yesterday, the woman gave a book to the man.'

(6) *Dem Mann gab die Frau gestern ein Buch.*

(7) *Ein Buch gab gestern die Frau dem Mann.*

(8) *Gestern gab die Frau dem Mann ein Buch.*

An important model in German syntax is the topological model as described in, e.g., Askedal (1986). It defines various topological fields: e.g., the first constituent of a declarative main clause is placed in the vorfeld, while the elements between the finite verb and the verb cluster constitute the mittelfeld. As we will see in the following sections, many internal representations in SimpleNLG for German correspond to topological fields in this model.

2.3.1 Subject realisation

Significant changes had to be made to the original SimpleNLG architecture to enable free constituent ordering. The most important change regards the realisation of subjects, which was moved from the sentence to the verb phrase.

front	*pre*-S	**S**	*post*-S	*pre*-I	**I**	*post*-I	*pre*-O	**O**	*post*-O	default

Figure 1: Position values for SIO word order

In SimpleNLG, subjects are always realised at sentence level, while other complements of the verb are realised in the (embedded) verb phrase. This already poses a few technical challenges for passive sentences, as a complement from the verb phrase has to be raised to subject position, while a passive complement has to be built from the subject and inserted into the verb phrase. Following a similar approach for subject movement would introduce even further complexity, but more importantly, it would imply treating subject placement differently from the placement of other constituents. Placing the subject between two VP elements after the realisation process is practically impossible, too, as the verb phrase (like all phrases) is realised as a unit and returns a single text string. For these reasons, the realisation of subjects has been moved to the verb phrase.

Although SimpleNLG for German is not built after any syntactic theory in particular, it should be noted that there are theories supporting this change: e.g., Haider (1993, p. 142 ff.) provides arguments for a VP-internal subject position, while Oppenrieder (1991) argues that a VP constituent which separates the subject from other arguments of the verb can not be justified for German.

2.3.2 Ordering the constituents

As one of the goals was to preserve the simplicity of use of SimpleNLG, the free ordering of constituents had to be implemented in an intuitive, user-friendly way. To achieve this, a two-layered system has been devised: the order of verb complements is defined through a property of the verb phrase, while the placement of other constituents is specified either relative to a complement or with an absolute value.

In SimpleNLG for German, every verb phrase has a *word order* property, which determines the order of its complements. Word order can be any permutation of subject (S), direct object (O), and indirect object (I). This is unambiguous because multiple complements of the same function are always aggregated into one coordinate phrase. Genitive objects are relatively rare and are treated like direct objects for this purpose. The default word order for new verb phrases is SIO, which is the syntactically unmarked word order in German (Eisenberg, 2004, p. 406 ff.).

Modifiers, e.g. adverbs or prepositional phrases, are realised after any complements by default. To control their placement, they can be given a *position* value. Position can be given either as an absolute value, which allows modifiers to be placed at the beginning or the end of the verb phrase, or relative to a complement, e.g. before or after the direct object. This placement specification will be obeyed even if the complement word order is later changed.

For example, assume that the variable `s` contains sentence (5) with all constituents except for the adverb *gestern* 'yesterday'. To generate (7), the adverb could be specified to be placed before the subject, and the word order must be changed to OSI:

(9)
```
s.setWordOrder(OSI);
s.addModifier(PRE_SUBJECT,
    "gestern")
```

If the word order is later changed to SIO again, the result is (8): the adverb is now realised in the vorfeld, so it still appears before the subject.

Placement specification is not restricted to modifiers, but can also be used for subordinate clauses and automatically generated passive complements.

Internally, each position can be thought of as a slot into which constituents can be placed. Complements (S, I, O) always have a fixed position slot assigned to them, while modifiers can be freely placed in any of the non-complement slots. The ordering of these slots is determined by the (complement) word order; figure 1 shows the position values for the default SIO word order. During realisation, the positions are traversed from left to right; if there is more than one constituent at any given position, they are realised in the order in which they were added. The constituent which is the first one to be realised this way is then moved to the vorfeld.

2.4 Modal verbs

In German, it is possible for a sentence to contain more than one modal verb. This necessitates the change to have a list of modal verbs for each verb phrase rather than just a single slot. Apart from that, sentences with modals have the property that the modal verb can be realised in perfect tense separately from the main verb:

(10) *Sie hat es tun können.*
she have it do can
'She was able to do it.'

(11) *Sie kann es getan haben.*
she can it done have
'She might have done it.'

In SimpleNLG for German, when a sentence is set to perfect tense, it is always the finite verb which is realised as perfect, as in (10). To be able to realise (11), a feature was added that explicitly sets the main verb to perfect tense. If no modal verb is included in the sentence, there is no difference between setting this feature and setting perfect tense in the traditional way. A combination of both settings to realise both the finite and the main verb in perfect tense is also possible.

3 Grammatical coverage

A proper evaluation of grammatical coverage is a difficult task due to the sheer number of possible constructions. In a short, non-representative survey examining five randomly selected Wikipedia articles[2], 115 of 152 sentences (75.66%) were covered by the system's grammar. Sentences were classified based on whether they could be realised within the framework using canned text only for uninflectable elements. To this end, each type of grammatical construction was recreated once within the system. The results suggest that the framework is already suitable for real-world applications.

Features and grammatical constructions supported so far include:

- morphological operations, including handling of inflection classes, separable verb prefixes, compounding, and preposition-article-contraction;
- modal verb clusters and perfect formation;
- relative clauses and relative clause extraposition; and
- constituent reordering.

However, a number of aspects remain which are not yet (fully) implemented. Verb phrase coordination is probably the most important one, as it is responsible for most of the unrealisable sentences in the above-mentioned survey. Negation is implemented only rudimentarily and is confined to the insertion of the negation particle *nicht* at a fixed position. Semi-modal verbs ('Halbmodalverben') take an infinitive with *zu*, which is not yet explicitly supported. Also, verb cluster fronting is currently not realisable:

(12) *Gesehen hatte er mich nicht.*
seen had he me not
'He had not seen me.'

In the current implementation, the position of the verb cluster is fixed, and its elements are kept separately from other sentence constituents. Therefore, sentences like (12) require further modifications to the internal representation. However, sentences of this type are pragmatically marked, so their realisation might be a peripheral problem.

In conclusion, the grammatical coverage of SimpleNLG for German is already considerable, but far from being complete. It is worth noting that some of the features mentioned above, e.g. relative clause extraposition, constitute non-trivial problems for syntactical theories of German, but are realisable in this framework in a surprisingly simple manner. The paper also highlighted several technical and conceptual problems that a realisation engine for German has to face, and offered potential solutions for some of them.

The full Java package of SimpleNLG for German will be made available online after it has been prepared for release.

Acknowledgments

I would like to thank my supervisor, Prof. Ralf Klabunde, for his guidance and support during this project.

[2] 'Josef Bartoň-Dobenín der Jüngere', 'Michael Joseph Savage', 'Saljut 7 EO-1', 'Zubringerstraße', 'Hapag-Lloyd-Flug 3378'; all retrieved on 18.05.2011.

References

John Ole Askedal. 1986. Über ‘Stellungsfelder’ und ‘Satztypen’ im Deutschen. *Deutsche Sprache*, 14:193–223.

Peter Eisenberg. 2004. *Grundriß der deutschen Grammatik. Bd. 2: Der Satz*. Metzler, Stuttgart, 2nd edition.

Arne Fitschen. 2004. *Ein Computerlinguistisches Lexikon als komplexes System*. Doctoral dissertation, University of Stuttgart.

Albert Gatt and Ehud Reiter. 2009. SimpleNLG: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, pages 90–93.

Hubert Haider. 1993. *Deutsche Syntax – generativ: Vorstudien zur Theorie einer projektiven Grammatik*. Narr, Tübingen.

Wilhelm Oppenrieder. 1991. *Von Subjekten, Sätzen und Subjektsätzen: Untersuchungen zur Syntax des Deutschen*. Niemeyer, Tübingen.