SemScribe: Natural Language Generation for Medical Reports

Sebastian Varges[1] **Heike Bieler**[1] **Manfred Stede**[1]
Lukas C. Faulstich[2] **Kristin Irsig**[2] **Malik Atalla**[2]

1 Applied Computational Linguistics Lab, University of Potsdam, Germany
2 ID GmbH & Co. KGaA, Berlin, Germany

Abstract

Natural language generation in the medical domain is heavily influenced by domain knowledge and genre-specific text characteristics. We present SemScribe, an implemented natural language generation system that produces doctor's letters, in particular descriptions of cardiological findings. Texts in this domain are characterized by a high density of information and a relatively telegraphic style. Domain knowledge is encoded in a medical ontology of about 80,000 concepts. The ontology is used in particular for concept generalizations during referring expression generation. Architecturally, the system is a generation pipeline that uses a corpus-informed syntactic frame approach for realizing sentences appropriate to the domain. The system reads XML documents conforming to the HL7 Clinical Document Architecture (CDA) Standard and enhances them with generated text and references to the used data elements. We conducted a first clinical trial evaluation with medical staff and report on the findings.

Keywords: natural language generation, ontologies, applications

1. Introduction

We present SemScribe, an implemented system that automates the mapping from individual medical observations to a medical report in natural language, and thus eliminates the need for a distinct text production step on the side of the physician. The doctor enters observations into a structured entry form, and the corresponding text is produced instantaneously. In contrast to simple text block-based systems, SemScribe can systematically produce variants from the same input (verbose versus brief text), and it aims at generating more cohesive text on the basis of a well-motivated choice of referring expressions (in particular, decisions on head noun and pronominalization).

The general idea of the SemScribe approach is inspired by the SUREGEN system (Hüske-Kraus, 2003a; Hüske-Kraus, 2003b) but adds an emphasis on re-usable generation components as well as a grounding in a (pre-existing) domain ontology, which facilitates the transfer of the system to other medical domains as well as to other target languages (at present, we generate German text only).

This paper is structured as follows: Section 2 provides more details about the task. Section 3 outlines the overall architecture of the system and section 4 details the sequence of generation steps necessary for mapping non-linguistic observation-data to text. Section 5 describes referring expression generation in greater detail. In section 6 we report on findings of a first clinical evaluation. Section 7 discusses related work and provides further discussion, for example on domain adaptivity aspects of the approach (section 7.3). Section 8 concludes this paper.

2. Task Description

We address a subtask of the larger endeavor of intelligent assistance for the production of doctor's letters. Depending on the particular clinical domain, the texts to be produced are more or less stereotypical, thus suggesting different methods for creating them. Our project focuses on cardiology.

The first step was to obtain a corpus of authentic doctor's letters of a hospital in order to study the linguistic phenomena and then to devise appropriate production strategies. We selected a subcorpus of 70 texts produced by three different doctors. Overall, these texts contain 429 statements. In the various cardiological diagnoses, one general finding is a very high density of information, which leads to a relatively telegraphic style of concatenated noun phrases, rather than full-fledged sentences. Consider the example in Figure 1, taken from an echocardiography diagnosis. By 'informational density', we refer to the fact that individual units of information (measurements, observations) are typically fused into complex noun phrases, which has consequences for deciding on the best strategy of production.

The text can also be characterized as being in a largely 'context-less' style, i.e. there is relatively little reference to previous sentences. The domain is sufficiently known to the reader (fellow doctors) that the ontology concepts can be regarded as shared among participants. Many descriptions are therefore definite upon first mention (*'the left atrium'* in example 3), similar to reference to *'the moon'* in everyday language. However, it is a challenge to refer to more than one concept at the time by using the ontology – this is one of the challenges for natural language generation.

3. System Overview and Architecture

While it may be possible in principle to manage a large set of text blocks (canned text) that accounts not only for individual observations but for particular *combinations* of observations/measurements and expresses such information bundles in corresponding text bundles, this approach becomes inflexible and extremely difficult to port to other domains, where the problems are similar yet require a completely new set of text blocks. We therefore use a language generator to systematically map non-linguistic information to linguistic output, i.e., text. For architectural simplicity, we employ a 'slim' processing pipeline (Reiter, 1994; Reiter and Dale, 2000) without search or large-scale overgeneration (Langkilde and Knight, 1998; Varges and Mellish,

(1) a. Beide Ventrikel und der rechte Vorhof sind normal dimensioniert.

b. (Both ventricles and the right atrium are of normal size.)

(2) a. Sämtliche LV-Wände sind leichtgradig verdickt.

b. (All LV-walls are mildly thickened.)

(3) a. Der linke Vorhof ist mit 53 mm mittelgradig dilatiert bei unauffälliger globaler und regionaler linksventrikulärer Pumpfunktion.

b. (The left atrium is moderately dilated at a size of 53mm and exhibits an unremarkable global and regional ventricular pumping function.)

Figure 1: Example sentences from the corpus of medical reports.

2010). In order to deal with the module-interdependencies of referring expression generation (GRE) in particular, GRE is split into two parts and wrapped around sentence planning (details below). The resulting generation pipeline comprises the following modules:

1. Document planning: Serializing the information units into an appropriate order.
2. GRE I: logical inferencing over the ontology, in particular to identify generalizations of sets of concepts.
3. Sentence planning: "Chunking" the information into bundles that will later be expressed as individual sentences, deciding on words to use for conveying the information, and deciding on how to bundle information units into appropriate linguistic phrases.
4. GRE II: referring expression specification including coreference-triggered generation of anaphoric expressions.
5. Realization: the lexical items are morphologically realized and the final sentence is produced.

Sentences planned in step 3 are realized left-to-right incrementally before the next sentence is planned. This allows the system to maintain a discourse memory of the left context that can be used in the GRE II step.

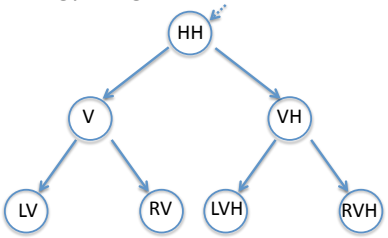
3.1. Medical Data Sources and User Interface

The input data for text generation comes from two sources:

1. Electronic data available in clinical systems such as measurements from ultrasound imaging devices or demographic patient data from a hospital information system is imported using interfaces for communication standards such as DICOM SR[1] or HL7.[2]
2. Further diagnostic observations are entered by the physician using a Web-based structured input form. A screenshot of the graphical interface for data entry is shown in figure 7.

[1]DICOM: `http://medical.nema.org/`
[2]HL7: `http://www.hl7.org/`

Ontology (fragment of ISA-relation):



Lexicon:

Node	WNC Code	German	English
HH	T002A05	Herzhöhle	heart chamber
V	T000BCF	Ventrikel	ventricle
VH	T000BB1	Herzvorhof	atrium
LV	T000BE2	linker Ventrikel	left ventricle
RV	T000BDA	rechter Ventrikel	right ventricle
LVH	T000BC8	linker Vorhof	left atrium
RVH	T000BBC	rechter Vorhof	right atrium

Figure 2: Resources for NLG (fragments).

All input data are represented as a single XML document conforming to the HL7 Clinical Document Architecture (CDA) standard.[3] The text generator reads this document and adds generated text in XML elements that link it to the data elements it is derived from. The resulting document combines structured data with generated text. It is both human-readable using a standard Web browser and machine-readable by other e-health software that supports the CDA standard.

4. Generation Pipeline

4.1. Example Input

The input can generally be described as a list of triples *(object,attribute,value)* where the object typically is an organ, the attribute a property (e.g. length) and the value the actual measurement or interpretation. Figure 5 shows as input a set of observations that describe 3 organs to be of normal size.[4] All actual input is identified by a 'WNC' code (a medical concept index). However, for readability we only show short concept labels in figure 5; the lexicon in figure 2 shows their realizations.

4.2. Document Planning

The document planner defines the overall structure of the document. This concerns paragraphs, headlines and content order. The planner can be configured individually according to the needs of the application and the preferences of the hospital or physician.
The general structure and order of the document is defined in a document template. This organizes hierarchical paragraphs, headlines and the order of the final content. To this end, abstract 'content elements' are defined as containers

[3]HL7 CDA: `http://www.hl7.org/implement/standards/cda.cfm`
[4]In contrast to the predicate-argument style notation of the example in figure (5), the actual implementation is in Java with resources in XML.

for observations, which may be arranged within the same sentence. The user controls the content of these elements by defining observation mappings using underspecified observation descriptions.
The document planner does not have to be adjusted to process other languages (section 7.3). To generate texts in another domain, the user supplies new configurations of the document template and observation mappings. The result of document planning is a document template filled with input observations.

4.3. Sentence Planning

The sentence planning task includes aggregation and lexicalization of observations. Sentence planning is performed by recursively composing partially specified syntactic trees. The sentence planner works on all observations within a content element and produces one or more sentences for these. The linguistic knowledge used for sentence planning is organized in frames and concept entries, which control the lexicalization of the input. Each frame describes a specific aspect of the domain language and contains a set of templates. These represent a syntactic tree as a possible sentence plan for the aspect considered in the frame. Templates may recursively invoke other frames, thus constructing the final sentence plan for an observation from multiple substructures.

4.3.1. Concept Lexicon

Input observations contain medical codes for attribute, object and value (see section 4.1). These codes point to concepts in the ontology. The concept lexicon represents a set of variants for each concept of the domain. Each variant represents a syntactic tree which will be adjoined to the sentence plan. The advantage of using a tree structure rather than a plain string is that different inflectional forms can be realized for the same entry.

4.3.2. Observation Aggregation

Multiple observations are aggregated by identical attributes and values (as in figure 5) or by object, i.e. different observations were made about the same organ(s). The aggregation operations map observations to observations.

4.3.3. Referring Expression Generation: Step 1

Once aggregated observations are available, the first stage of referring expression generation is invoked. This is explained in detail in section 5.1. The resulting referring expression specifications are used in the following step.

4.3.4. Frame Instantiation

The next step is to construct a sentence plan for each aggregated observation. This is done by selecting and filling *templates*. Templates are syntactic structures with variables (*slots*) to be filled. They are organized in frames; each frame describes an aspect of the domain.
The initial frame for processing an observation is determined by the observation attribute. Thus, an observation with attribute *'size'* will use the *'size'*-frame. The frame contains several templates that produce a clause for describing the size of an object. Template selection is generally controlled by the object and value of the input observation.

1. Die beiden Ventrikel und das rechte Atrium sind normal dimensioniert.
   (Both ventricles and the right atrium are of normal size.)
2. Die beiden Herzkammern und der rechte Vorhof sind normal dimensioniert.
   (Both heart chambers and the right atrium are of normal size.)
3. LV, RV und LA normal dimensioniert.
   (LV, RV and LA of normal size.)

Figure 3: Example for different text versions for the same input.

A template requires input slots and restricts them to specific values, such as a specific concept (code) or a syntactic category for realization. Next, the slots are filled with the concept's lexical entry for the corresponding input code. Input concepts of organs are marked as referring expressions, whose generation is done separately in GRE step II (see Section 5.2). The result of frame instantiation is a syntactic tree that usually represents a clause.

4.3.5. Variants Ranking

The selection of the best template or concept variant is based on a ranking of the variant's features. The linguistic resources are annotated with predefined feature-value-pairs. Currently, we use the boolean features *'short'* and *'simple'*. By default, the system produces a text with medical specialist's terms and complete sentences. In the GUI, the user has the option to either select the generation of a short, abbreviated text of compact sentences (often without a verb) or a version for the patient with understandable medical terms. The variants are ranked by weight and the best variant is used for realization. Figure 3 shows different versions for the same input.

4.3.6. Sentence Construction

Frame instantiation supplies a list of syntactic clause trees. For each clause, the system generates a simple sentence. In some cases, two or more clauses are combined.
At this stage in the pipeline we have obtained most of the final sentence plan with only the referring expressions still remaining variables. Figure 4 shows such a sentence plan for the example input of figure 5.

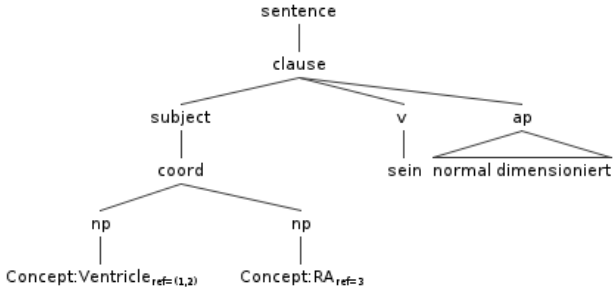


Figure 4: Sentence plan before stage 2 of referring expression generation.

4.3.7. Referring Expression Generation: Step 2

Discourse-related decision on GRE as well as the surface features of the referring expression are determined at this stage. For details see section 5.2.

4.4. Realization

In the final stage, a morphosyntactically-correct expression of the sentence plan as a sentence (even if telegraphic) in the target language (here: German) is produced. This involves fixing word order, insertion of function words, and producing proper inflection. We use a sentence realizer called XbarGen (Sawitzky, 2011) that involves the notion of Xbar-Theory and topological fields of German.

5. Referring Expression Generation

The task of generating referring expressions (GRE) can be characterized by the following question: given a domain model – in this case the medical ontology – and a set of referents, how can we uniquely identify the referent(s) using natural language? This task has often been combined with the requirement to be minimal, i.e. to use only a minimal number of properties, and with considerations of computational complexity. Early work often centered around the 'incremental algorithm' (Dale and Reiter, 1995) which incrementally 'intersects' attributes until the referent is uniquely identifiable. Due to the importance of GRE to the field of language generation in general, several alternative approaches have been developed, e.g. the graph algorithm of (Krahmer et al., 2003) or the 'overgeneration and ranking' approach of (Varges and van Deemter, 2005).

5.1. GRE I: Ontological inference

In SemScribe, where domain knowledge is available in the form of a medical ontology that includes ISA relationships between concepts, we often need to model reference to several objects by the name of a dominating concept. The resulting, possibly generalized concept can be used in combination with attributes identified with standard GRE approaches (section 7.2).

In example 1, the use of *'both ventricles'* presupposes that the objects in question are of type 'ventricle' and that there can be only two of them. Such inference is required before a syntactic frame is selected since this determines the grouping of the input into the grammatical functions of the sentence planner.

The required knowledge is part of the ISA relations of the ontology. A fragment of the ontology *ID MACS® – medical semantic network* of about 80000 concepts/nodes provided by ID is shown in figure 2 along with some lexical entries. Nodes/concepts in the ontology are concept classes (TBox), not instances of a particular patient's organ (ABox). The input to NLG, on the other hand, is interpreted as being at the instance level.

5.1.1. GRE Algorithm

In the ontological inference stage, we generally search for a concept/node in the ontology that dominates as many input concepts as possible without dominating any other nodes not mentioned in the input. Immediate dominance is not necessarily required. The ISA graph of the ontology is

Generation input:

Observation(LV, size, normal).
Observation(RVH, size, normal).
Observation(RV, size, normal).

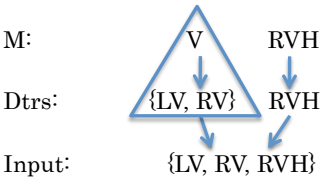
Aggregation:

observation([N4, N7, N5], size, normal).

Chart items:

	Input	M	Dtrs	Cov
e_0	LV	V	LV, RV	true
e_1	LV	HH	LV, RV, LVH, RVH	false
e_2	RVH	VH	LVH, RVH	false
e_3^*	RV	V	LV, RV	true
e_4^*	RV	HH	LV, RV, LVH, RVH	false

Structure of Referring Expression:



Realization:

both ventricles and the right atrium

Figure 5: Generation of RE for example (1).

queried by an algorithm that performs a mixed-order traversal for each input concept in turn by walking 'bottom-up' from the input concepts to higher-level concepts and then 'top down' to the sisters of that input concept (functions *Build_Chart* and *Traverse_Ontology* in the algorithm sketch in fig. 6). This is repeated until sisters not in the input are found. Since the ontology graph is large and we do not necessarily deal with terminal nodes, the algorithm keeps track of the number of levels in the bottom-up phase and uses this in the top-down phase to identify the 'level' of the daughters (counter i; the terms 'mother' and 'daughter' are understood transitively here). If a terminal node or another input concept is encountered 'early' because they are located at a 'higher' level, the traversal stops.

The results of the ontology query are stored in a chart. This enables reuse of edges at the discourse stage (section 5.2) which is important since ontology access is provided by an external service. Figure 5 shows some processing stages for the input corresponding to example (1) in figure 1. The chart edges are produced by the algorithm in figure 6 using the ontology fragment in figure 2. The edges represent two levels, the 'lower' input nodes ('*Dtrs*') and the dominating upper node ('*M*'). Additionally, it is represented whether all input nodes are present in the daughters set ('*Cov*'). If the redundancy check is used (function *Redundant* in figure 6), edges e_3 and e_4 are not produced.

Next, we search for a minimal combination of mutually exclusive and complete/exhaustive chart edges that covers the input, following the Gricean Maxim of Brevity (Dale and Reiter, 1995) (mentioned as function *Select_Edges* in the algorithm sketch). This combination may also include input concepts that could not be generalized. Each of the chart

```
def  Gre(input):
     chart ← Build_Chart(input)
     refex ← Select_Edges(chart,input)
     return refex

def  Build_Chart(input):
     chart ← []
     for el ∈ input:
        if not Redundant(el,chart):
           cov ← true, i ← 0
           while cov:
              i ← i+1
              m, dtrs ← Traverse_Ontology(i, el)
              cov ← Dtrs_Covered(dtrs, input)
              chart.append((m, dtrs, cov))
     return chart

def  Traverse_Ontology(i,el):
     m ← mother(el,i)              # walk up i levels
     dtrs ← daughters(m,i)         # walk down i levels
     return m, dtrs

def  Dtrs_Covered(dtrs, input):
     if for all d ∈ dtrs: d ∈ input: return true
     else: return false

def  Redundant(el, chart):
     if el ∈ e.dtrs for some edge e ∈ chart:
        return true
     else: return false
```

Figure 6: Sketch of algorithm for ontology-based GRE

edges so selected will give rise to a referring expression (the generalized edge e_0 and the not generalizable concept *RVH* in figure 5). The two concepts of level *M* will be realized whereas level *Dtrs* represents the extensional semantics. The union of the extensions is equivalent to the input to the generator. The concepts resulting from this phase, i.e. *V* and *RVH* in this case, will be used to determine the 'type' attribute of the referring expression to be generated (see also section 7.2).

5.2. GRE II: Co-reference based Generalization

In addition to ontology-based generalization (section 5.1), generalization may also be triggered by the discourse context: a second mention of a specific car (*'the black Mercedes'*) might just refer to *'the car'*, clearly using the ontology (rather than just performing a syntactic operation). In this type of generalization, the referents do not have to comprise all the daughters of the dominating concept at a certain level.

To keep track of previous mentions of referents, the discourse context is structured into sentence-sized units and contains their referring expressions including their syntactic features. This makes it possible to determine the 'preferred center' (Poesio et al., 2004) for possible pronominalization, for example. The discourse memory is updated after a sentence is generated. It is used to identify backwards co-reference links for the top-level expression of the input under consideration, i.e. the entire input grouped in a grammatical function (set {*LV,RVH,RV*} in figure 5) as well as for its parts (individual referring expressions such as *V* and *RVH*). This identification is required because the generation input is underspecified in this respect as it originates from a non-linguistic component. In our current model, the antecedent needs to be a top-level expression. If a coreference link is available, the use of a more general concept is triggered, based on the chart already obtained for the previous inference step (section 5.1). For example, a follow-up sentence that uses the same input as in figure 5 could use the (incomplete) chart edge e_2, i.e. *VH*, to refer to *RVH*.

5.2.1. Syntactic Features and Lexical Choice

With the syntactic constraints known, the referring expression structure can be specified further and inserted into the sentence plan. For generalized concepts, an appropriate quantifier needs to be selected based on the size of the extensions: *'both [ventricles]'*, *'all four [heart chambers]'*. Furthermore, the inner structure of the referring expression is used to determine the need of a conjunction (used in fig. 5).

6. Evaluation

We have conducted a short task-based evaluation with medical staff at a hospital.

The task was to form a diagnosis using the usual medical tools (based on MRI scans etc.) and then to create the Doctor's letter using the new text generation tool. Following each case, a detailed questionnaire was given to the doctor in which he/she made judgments about both linguistic fluency and expressibility of the generated text, and the general usability characteristics of the tool and approach. Detailed comments were asked about the realization of individual medical findings, for example the ordering of specific pieces of information.

Due to time constraints imposed by the project funding, only few cases have been evaluated, and only by a single doctor. Hence the results do not support a statistical analysis. Instead we summarize here our conclusions drawn from the doctor's judgments and comments.

The evaluation underlined, as expected, the need for a flexible document planning component since our original document plan extracted from our corpus was not entirely accepted by the Doctor. The capability of sorting, grouping, and summarizing the given observations was regarded as one of the systems greatest assets and resulted in an increased acceptance of the generated text as Doctor's letters. Various helpful suggestions regarding vocabulary and phrasing have been incorporated in a later release of the prototype. The high number of medical clarifications received shows that the customization of the NLG system needs to be carried out by domain specialists using appropriate development tools.

7. Related Work and Discussion

7.1. NLG in Medical Applications

The general idea of the SemScribe approach is inspired by the SUREGEN system (Hüske-Kraus, 2003a; Hüske-Kraus, 2003b) but adds an emphasis on re-usable generation components as well as a grounding in a (pre-existing)

domain ontology, which facilitates the transfer of the system to other medical domains as well as to other target languages (at present, we generate German text only). Our approach is also related to Cawsey *et al.* (Cawsey et al., 2000) who describe a system for generating tailored text that explains treatments, diseases etc. to patients. Moreover, Reiter *et al.* (Reiter et al., 2001) generate customized letters that encourage people to stop smoking. However, both these approaches have a different focus (patients/end users).

The BabyTalk project at Aberdeen (Gatt et al., 2009; Portet et al., 2009) investigates data-to-text generation in Neonatal Intensive Care Units. A focus of the project is the actual mining/analyzing of large amounts of numerical data. In contrast, SemScribe's input has already been analyzed either automatically or by a specialist, which is reflected in the two input sources of the system (section 3.1). The resulting texts are quite different in these two systems since SemScribe data does not reflect a time series of measurements.

7.2. Referring Expression Generation

The ontology traversal algorithm searches the ontology for dominating concepts and thus limits itself to the selection of the type attribute. The resulting, possibly generalized concept can be used in combination with attributes identified with other, more general approaches to referring expression generation; see (van Deemter, 2002; Krahmer et al., 2003; Horacek, 2004; Gatt, 2007) for references to sets of objects, for example. In the incremental algorithm of (Dale and Reiter, 1995), a generalized concept can be used as the value of the 'type' attribute, which is always chosen regardless of discriminatory power, and which is usually mapped to a head noun. In the graph-based approach of (Krahmer et al., 2003), our algorithm could provide (cheap) type edges.

In (Dale and Reiter, 1995), the importance of 'basic-level values' is emphasized, i.e. lexical preferences for specific, unmarked concepts (e.g. *'dog'* rather than *'pit bull'*). In this work, we made a related observation: the medical ontology, which was not produced for NLP applications in the first place, contains some concepts that are not required from a language perspective (and that can thus prevent successful generalization). We addressed this by introducing an 'NLG view' of the ontology which defines a subset of language-relevant concepts. The generalization algorithm only works on this subset. The NLG view currently does not express preferences for specific concepts within this subset of the ontology.

The ontology traversal algorithm assumes a single inheritance ontology. However, this requirement only applies to the ISA relation. Other relations could use multiple inheritance.

The algorithm identifies the least general (most special) generalization. In the context of logic programming, this is also called 'anti-unification' (Hinkelmann et al., 1994).

Ontological reasoning plays a big role in the semantic web and its theoretical foundation OWL/Description Logic (see (Mellish, 2010) and (Power, 2009) for applications to NLG, for example). The ontology in the presented work is maintained externally in relational form, thus requiring any language-related inferences to be performed by the NLG system.

7.3. Domain Adaptivity

A central aspect is the possibility of flexible adaptation of the generation decisions to the physicians needs and preferences. Therefore, the scenario also involves a toolbox for defining and editing the document plans (i.e., defining the ordering of information in the text), the devices ('frames') for mapping non-linguistic information to linguistic expressions (of different types) and a lexicon of concepts. Figure 8 shows a screenshot of the maintenance and adaptation tool.

The frames are used to recursively compose partially specified syntactic trees, similar in spirit to the SPUD (Stone and Doran, 1997) and PROTECTOR (Nicolov and Mellish, 1999) systems. Concepts in turn contain fully specified syntactic trees, which can be referred to in frames.

Frames and concepts may contain several realization alternatives, for example for different target audiences (patients vs specialists). The frame instantiation mechanism works language-independently. It can combine any syntactic tree of the frame and concept lexicons without making any assumptions about the grammar. Furthermore, the realization component of the system uses a dictionary of declinations containing the variations of words according to tense, case, number, gender etc. It can also be edited and extended with a customized tool developed for this project.

In contrast to document plans, sentence planning resources (frames and concept entries) and the realization dictionary are language specific. Adapting the system to other languages entails providing all resources for the new language.

The tools are still being used in the development of new text generation applications and have proven their practicability.

8. Conclusions and Future Work

We presented SemScribe, an implemented natural language generation system that produces doctor's letters, in particular descriptions of cardiological findings. The approach is adaptable through configurable document plans and sentence frames.

We evaluated the approach in a task-based user study with medical staff. Due to the cost of specialist feedback and time constraints imposed by the project funding, the evaluation was qualitative rather quantitative in nature. It underlined the need for a flexible document planning component and the importance of providing appropriate development tools for customization by domain specialists.

Regarding the use of a medical ontology, we dealt with generalizations triggered by ontological inference and by discourse. We have not yet integrated mechanisms to *specialize* the descriptions to rule out distractors. For example, a discourse-based generalization may not be desirable if this results in confusability with other referents. Such an account should integrate earlier work on context-less GRE.

Acknowledgments

The SemScribe project presented here has been funded by the German Ministry of of Economics and Technology[5]. We wish to express our thanks to Dirk Hüske-Kraus for supporting the project with valuable advice and providing examples of cardiologic discharge letters. We also thank our clinical partners from Klinikum Ernst von Bergmann, Potsdam, for fruitful discussions, example data, deployment and evaluation of the prototype: Yvon Franke, Klaus Bonaventura, Frank Suckrow, Markus Weber.

9. References

Alison Cawsey, Ray Jones, and Janne Pearson. 2000. The Evaluation of a Personalised Health Information System for Patients with Cancer. *User Modeling and User-Adapted Interaction*, 10(1):47–72.

Robert Dale and Ehud Reiter. 1995. Computational Interpretations of the Gricean Maxims in the Generation of Referring Expressions. *Cognitive Science*, 19:233–263.

Albert Gatt, Francois Portet, Ehud Reiter, Jim Hunter, Saad Mahamood, Wendy Moncur, and Somayajulu Sripada. 2009. From Data to Text in the Neonatal Intensive Care Unit: Using NLG Technology for Decision Support and Information Management. *AI Communications*, 22:153–186.

Albert Gatt. 2007. *Generating Coherent References to Multiple Entities*. Ph.D. thesis, University of Aberdeen.

Knut Hinkelmann, Manfred Meyer, and Franz Schmalhofer. 1994. Knowledge-Base Evolution for Product and Production Planning. *AI Communication (AICOM)*, 7(2):98–113.

Helmut Horacek. 2004. On referring to sets of objects naturally. In Anja Belz, Roger Evans, and Paul Piwek, editors, *Proceedings of the Third International Natural Language Generation Conference (INLG-04)*, pages 70–79, Brockenhurst,UK, July. Springer.

Dirk Hüske-Kraus. 2003a. Suregen-2: A shell system for the generation of clinical documents. In *10th Conference of the European Chapter of the Association for Computational Linguistics (EACL-03)*, Budapest, Ungarn.

Dirk Hüske-Kraus. 2003b. Text Generation in Clinical Medicine. *Methods of Information in Medicine (Methods Inf Med), Schattauer,*, 42(1):51–60.

Emiel Krahmer, Sebastiaan van Erk, and Andr Verleg. 2003. Graph-based generation of referring expressions. *Computational Linguistics*, 29(1), March.

Irene Langkilde and Kevin Knight. 1998. Generation that Exploits Corpus-based Statistical Knowledge. In *Proceedings of COLING/ACL-98*, pages 704–710, Montreal, Canada.

Chris Mellish. 2010. Using Semantic Web Technology to Support NLG. Case Study: OWL finds RAGS. In *Proceedings of the 6th International Natural Language Generation Conference (INLG-10)*.

Nicolas Nicolov and Chris Mellish. 1999. PROTECTOR: Efficient Generation with Lexicalized Grammars. In *Recent Advances in Natural Language Processing (RANLP vol.II)*, pages 221–243, Amsterdam and Philadelphia. John Benjamins.

Massimo Poesio, Rosemary Stevenson, Barbara di Eugenio, and Janet Hitzeman. 2004. Centering: A parametric theory and its instantiations. *Computational Linguistics*, 30(3).

Francois Portet, Ehud Reiter, Albert Gatt, Jim Hunter, Somayajulu Sripada, Yvonne Freer, and Cindy Sykes. 2009. Automatic Generation of Textual Summaries from Neonatal Intensive Care Data. *Artificial Intelligence*, (173):789–816.

Richard Power. 2009. Towards a generation-based semantic web authoring tool. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG-09)*.

Ehud Reiter and Robert Dale. 2000. *Building Applied Natural Language Generation Systems*. Cambridge University Press, Cambridge, UK.

Ehud Reiter, Roma Robertson, A Scott Lennox, and Liesl Osman. 2001. Using a Randomised Controlled Clinical Trial to Evaluate an NLG System. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ehud Reiter. 1994. Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible? In *Proceedings of the Seventh International Workshop on Natural Language Generation (INLG-94)*, pages 163–170, Kennebunkport, Maine, USA.

D.R. Sawitzky. 2011. Generierung deutscher Sätze nach dem X-Bar-Schema: Entwurf und Implementierung, BA thesis, University of Potsdam.

Matthew Stone and Christine Doran. 1997. Sentence Planning as Description Using Tree-Adjoining Grammar. In *Proceedings of ACL-97*, pages 198–205.

Kees van Deemter. 2002. Generating Referring Expressions: Boolean Extensions of the Incremental Algorithm. *Computational Linguistics*, 28(1):37–52.

Sebastian Varges and Chris Mellish. 2010. Instance-based Natural Language Generation. *Journal of Natural Language Engineering.*, 16(3):309–346.

Sebastian Varges and Kees van Deemter. 2005. Generating referring expressions containing quantifiers. In *Proceedings of the 6th International Workshop on Computational Semantics (IWCS-6)*, Tilburg.

[5]Gefördert durch: Bundesministerium für Wirtschaft and Technologie aufgrund eines Beschlusses des Deutschen Bundestags. Förderkennzeichen KF2604501RR0 und KF2262302RR0.

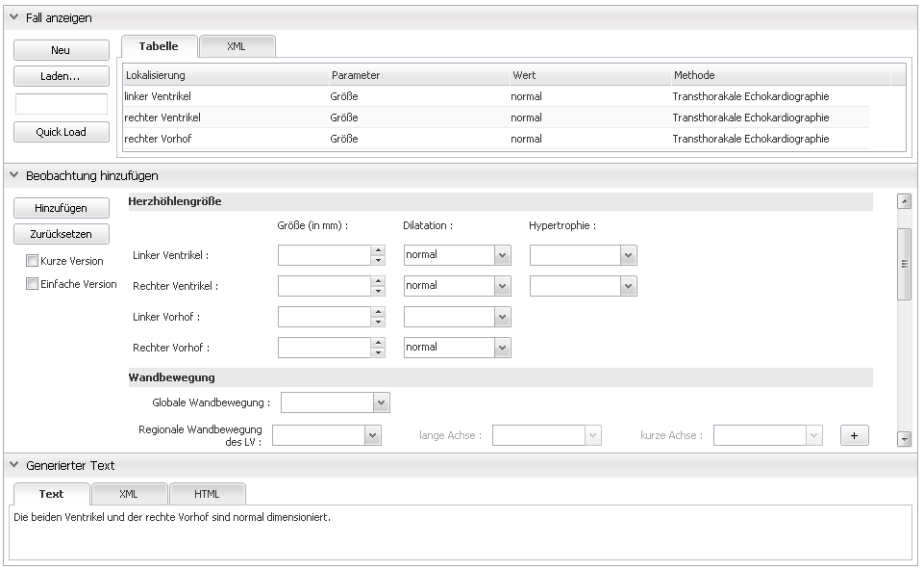


Figure 7: Screenshot of the graphical interface for data entry (output text in German).

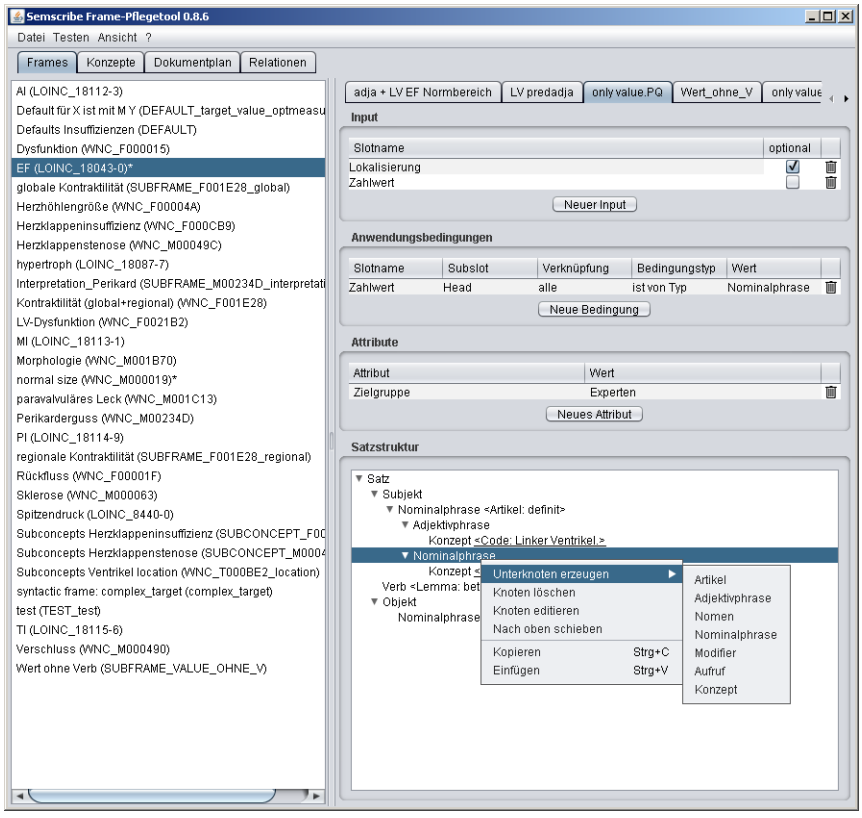


Figure 8: Screenshot of the maintenance and adaptation tool.