Practical Text Generation in Clinical Medicine

Fredrik Lindahl
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg, Sweden
lindahlf@cs.chalmers.se

January 3, 2005

Abstract

The MedView project aims at providing ways to formalize, store, and subsequentionally computationally analyze clinical data, making it more attractive for educatonal and research purposes. In our system, we use the formalized data to have the system generate various medical documents, such as patient medical records, discharge summaries, personalized patient information etc., using a generation context created by the user himself. This paper presents some of our experiences when introducing this system to users at an oral medicine clinic in Gothenburg. Furthermore, we will discuss the open-sourcing of our text generation framework, allowing others to reuse and improve the natural language generation used in our system.

1 Introduction

In the area of Oral Medicine, large amounts of clinical data are gathered every day from conducted oral examinations. Recording this data in the traditional paper-based manner presents problems when data is to be retrieved, summarized, or in other ways processed or analyzed in a time-efficient manner. As time goes by, the vast amount of accumulated clinical data becomes increasingly difficult to manage, and extracting information becomes cumbersome.

To counter this problem, the MedView project [1] has produced tools and methods of formalizing examination data, and has resulted in a large formalized knowledge base containing large amounts of examination data along with associated digital images of the oral mucosa. This formalized data can be used for several purposes, this paper discusses how to use the data to generate various clinical documents, such as personalized patient information leaflets, medical records and discharge summaries. We will also discuss the recent open-sourcing and generalization of the text generation subsystem, enabling others to re-use and improve on this part of the system.

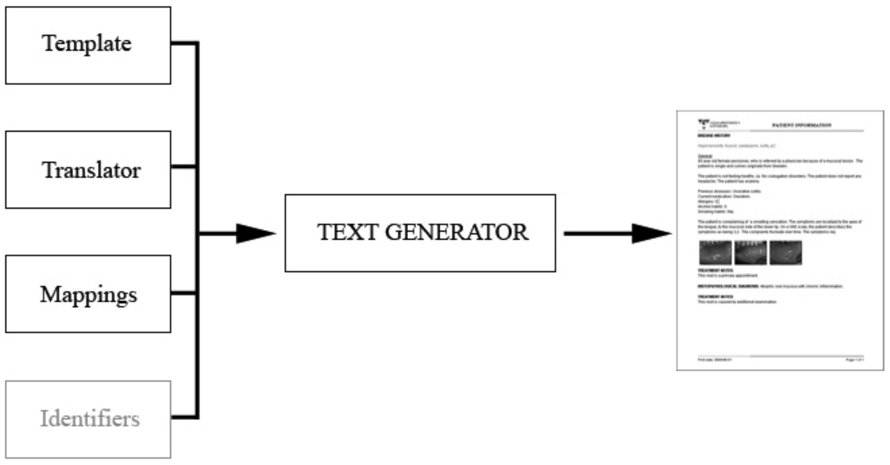


Figure 1: The four major text generation components

2 Materials and Methods

In the MedView system, we separate data input from data visualization [1]. During an examination, a clinician can thus focus on treating the patient and gathering core examination data without worrying about how it is to be presented later on. The examination data is then used to let the system generate medical documents on request, which replaces the need for dictation or for the clinicians to enter the text manually.

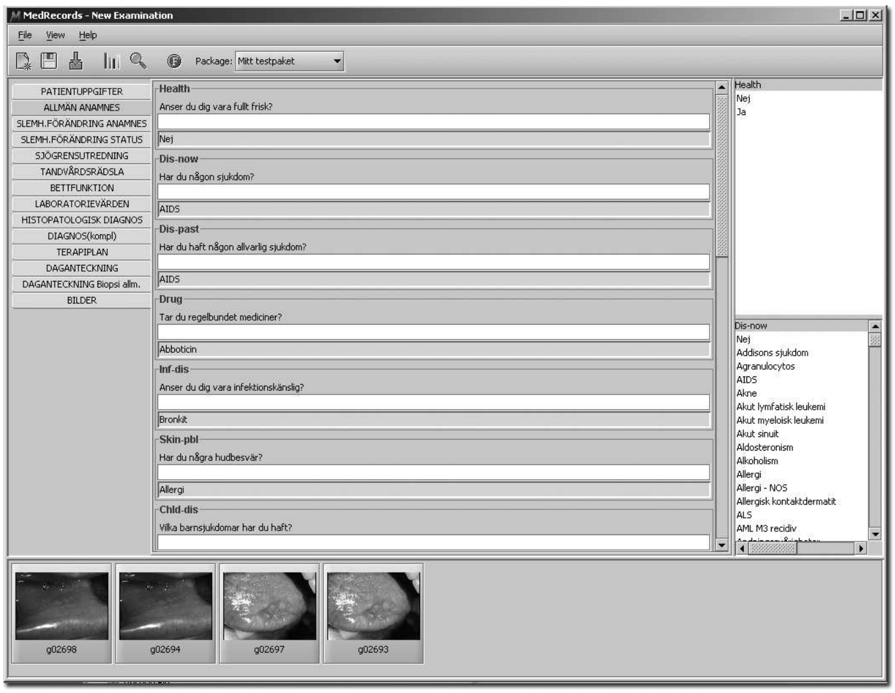
2.1 Generation Components

An idea saturating the entire development of the MedView system is user customizability and flexibility, i.e. the end user should be able to - as far as possible - customize the system so that it suits his or her personal preference. In the context of text generation, this includes the style of the generated medical documents, which can vary between clinicians. Thus, we have opted for a system in which the clinicians themselves can customize the appearance of the generated documents. For this to work, it is important that the system is not too complicated to use, and that the text generation process is understandable by a clinician.

Generation of clinical documents in our system is based on four major components, as shown in Figure 1. The fourth component is an extension of the general text generation framework discussed below, since it narrows the context to the medical domain due to its inclusion of examination data:

1. a template,

2. a translator,



Package: Mitt testpaket

Health
Anser du dig vara fullt frisk?
Nej

Dis-now
Har du någon sjukdom?
AIDS

Dis-past
Har du haft någon allvarlig sjukdom?
AIDS

Drug
Tar du regelbundet mediciner?
Abboticin

Inf-dis
Anser du dig vara infektionskänslig?
Bronkit

Skin-pbl
Har du några hudbesvär?
Allergi

Chld-dis
Vilka barnsjukdomar har du haft?

Figure 2: The feeder application used to enter instance data

3. a container of term-value mappings, and

4. an examination identifier.

The term-value mapping container contains the instance data, i.e. the core data of an examination (instance) that you wish to present to the reader of the generated document. A *term* is a concept which can have one or more *values*. An example of a term-value mapping is 'BORN = Sweden', in which case the term 'BORN' represents the concept of a patients country of birth, for this instance having the value 'Sweden'. Input of examination data is done in a feeder application, shown in Figure 2. During input, the clinician is presented with the option to preview the generated document, using a specified template and translator combination. The actual generation process consists of first processing and translating the atomic parts of the formalized instance data, followed by insertion of the results of the first step into the slots of the template.

The template is basically a text document, using a style and wording appropriate for its intended reader. It contains slots in which specific instance data, after having been processed, is inserted at generation time. Furthermore, the template is logically divided into *sections*, where each section represents a cohesive concept which the user can choose to include or exclude in the generated document. If the instance data does not contain corresponding values for a slot in the template, the line containing the slot is removed. Furthermore, certain values can trigger the removal of a line containing a slot, as discussed below.

The translator is a container for value translations, introducing the possibility for multilingual text generation, as well as for being able to incline the text towards various target readers based on the same instance data. For instance, when preparing a personalized patient information leaflet, examination data needs to be expressed in laymans terms. Preparing a discharge summary based on the same data can be expressed in terms understood by a medical professional.

The examination identifier is a domain-specific extension of the general text generation framework discussed below. It contains information about the examination during which the instance data was gathered, such as examination date, duration, location, care provider, as well as various patient information such as gender and age etc. Special terms exist whose values are derived from the value of the current examination identifier, we call these terms *derived*. For instance, the term 'Pcode(Gender)' is a derived term, with possible values 'He' and 'She'. The translator contains translation support for derived terms as well as the regular ones.

2.2 The Text Generation Framework

The text generation subsystem is comprised of two parts:

1. a general framework for text generation, currently in the process of being open-sourced to allow others to re-use and expand the ideas used in our project to their own context, and

2. specific implementations of parts of the framework to fit the context being used at the clinic, which also serves as a practical example of general framework extension.

The idea behind the general framework is that it should provide a solid base from which specific domain specific extensions can be built on, and that applications should be able to use different implementations of the framework interchangeably, thus enhancing the possibility to reuse the applications in different domains. Furthermore, graphical components used to modify the various parts of the framework are also in the process of being open-sourced, providing developers with a wide array of tools which they can use and integrate into their specific contexts.

The framework is based on object-oriented design principles for creating reusable components. A simplified view of the framework is shown in UML in Figure 3, where the extension components are filled in gray. The general framework revolves around the GeneratorEngine interface, which produces the generated document. At its disposal it has a container of term-value

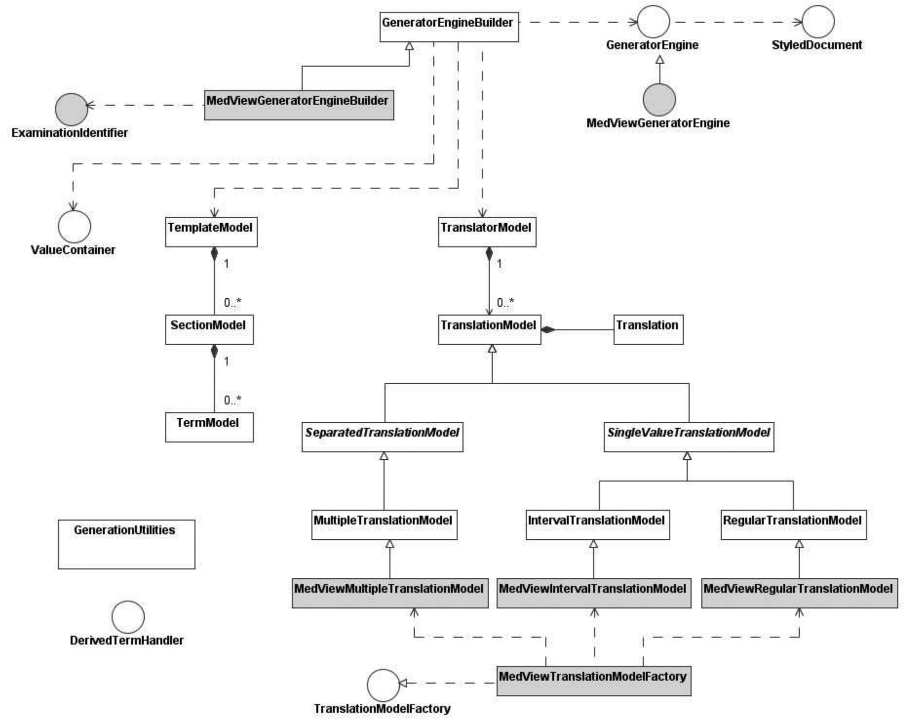


Figure 3: The general text generation framework with extensions

mappings, a template model and a translator model. These components are built upon the engine by using a builder object. As can be seen in the figure, the MedView system extends the builder object with one being able to also build an examination identifier upon the engine.

The template model is comprised of section models, each containing a number of slots where translated instance data can be inserted at generation. The generator engine takes as input the sections to include in the generated document, allowing different parts of the template to be included or excluded. Furthermore, a wide array of text processing utilities exist, as well as support for derived terms (explained above), i.e. terms whose values depend on the values of other terms.

The translator model is comprised of translation models for specific terms, each such model contains a number of translations for each value the represented term can obtain. The translation of a value can contain macros expanded at generation, and it is easy to expand the framework to include more macros if needed. Examples of some generally included macros are '?' indicating a placeholder for the instance value, and 'NOLINE' indicating that the value should cause the line containing the slot to be completely removed.

Each translation model representing a term can be of a certain type, indicating what kind of values the term can obtain. The general framework supports three types, namely:

1. terms that can obtain one value, for instance 'BORN',
2. terms that can obtain several values, for instance 'CURRENT-MEDICATIONS', and
3. terms that obtain a numerical value within an interval, for instance VAS-scales.

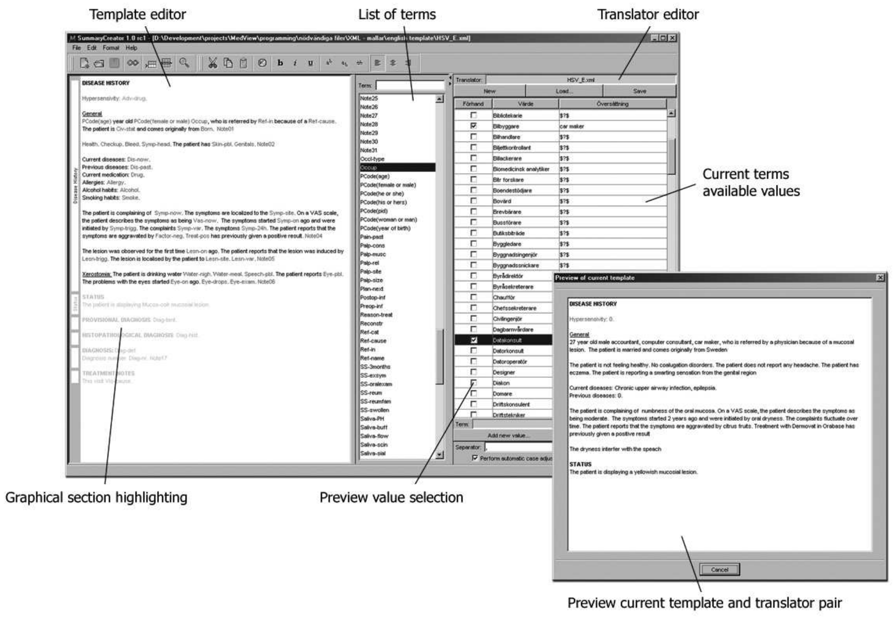
Since the translation models are created in an implementation of a factory interface, additional term types can easily be added by either implementing the interface from scratch or extending the sample implementation used in the MedView system. If additional functionality needs to be included, its easy to just extend the default models of the framework, such has been done in the MedView extension.

The described model framework is currently in the process of being open-sourced, and further work will be done to open-source higher-level GUI components utilizing the framework, providing developers with even more ready-to-use components which they can use or adapt in their own context. Figure 4 shows the application used in the MedView system to work with templates and translators, containing various pluggable components that are to be open-sourced.

3 Results

The text generation system, as described, has been in use at the clinic for several years now, and has been used to generate thousands of clinical documents. The generated documents have been printed and used as patient medical journals. Numerous enhancements have been since the system was first introduced, such as the possibility to insert images, obtaining a true WYSIWYG (what you see is what you get) display of the documents before print, text processing improvements etc. The quality of the generated text is deemed sufficient and is, as previously stated, in actual use for the daily work at the clinic. This said, there has been occasions where clinicians have wished for higher quality of the generated text.

Stylistic variance is also something clearly desired among the clinicians, and due to the large size of the templates and translators being created, a template is not being created for each clinician but rather for an entire clinic. Thus, the style of the generation components becomes one that is broadly accepted, but still leaves some to be desired on the individual level.


Template editor
List of terms
Translator editor
Current terms available values
Graphical section highlighting
Preview value selection
Preview current template and translator pair

SummaryCreator 1.0 rc1 - [D:\Development\projects\MedView\programming\nödvändiga filer\XML - mallar\english-template\HSV_E.xml]

DISEASE HISTORY

Hypersensivity: Adv-drug.

General
PCode(age) year old PCode(female or male) Occup, who is referred by Ref-in because of a Ref-cause. The patient is Civ-stat and comes originally from Born. Note01

Health. Checkup. Bleed. Symp-head. The patient has Skin-pbl. Genitals. Note02

Current diseases: Dis-now.
Previous diseases: Dis-past.
Current medication: Drug.
Allergies: Allergy.
Alcohol habits: Alcohol.
Smoking habits: Smoke.

The patient is complaining of Symp-now. The symptoms are localized to the Symp-site. On a VAS scale, the patient describes the symptoms as being Vas-now. The symptoms started Symp-on ago and were initiated by Symp-trigg. The complaints Symp-var. The symptoms Symp-24h. The patient reports that the symptoms are aggravated by Factor-neg. Treat-pos has previously given a positive result. Note04

The lesion was observed for the first time Lesn-on ago. The patient reports that the lesion was induced by Lesn-trigg. The lesion is localised by the patient to Lesn-site. Lesn-var. Note05

Xerostomia: The patient is drinking water Water-nigh. Water-meal. Speech-pbl. The patient reports Eye-pbl. The problems with the eyes started Eye-on ago. Eye-drops. Eye-exam. Note06

STATUS
The patient is displaying Mucos-colr mucosal lesion.

PROVISIONAL DIAGNOSIS: Diag-tent.

HISTOPATHOLOGICAL DIAGNOSIS: Diag-hist.

DIAGNOSIS: Diag-def
Diagnosis number: Diag-nr. Note17

TREATMENT NOTES
This visit Vis-cause.

Preview of current template

DISEASE HISTORY

Hypersensivity: 0.

General
27 year old male accountant, computer consultant, car maker, who is referred by a physician because of a mucosal lesion. The patient is married and comes originally from Sweden

The patient is not feeling healthy. No coalugation disorders. The patient does not report any headache. The patient has eczema. The patient is reporting a smarting sensation from the genital region

Current diseases: Chronic upper airway infection, epilepsia.
Previous diseases: 0.

The patient is complaining of numbness of the oral mucosa. On a VAS scale, the patient describes the symptoms as being moderate. The symptoms started 2 years ago and were initiated by oral dryness. The complaints fluctuate over time. The patient reports that the symptoms are aggravated by citrus fruits. Treatment with Dermovat in Orabase has previously given a positive result

The dryness interfer with the speach

STATUS
The patient is displaying a yellowish mucosial lesion.

Figure 4: Application used to work with templates and translators

4 Discussion

In the context of NLG systems, our system can be described as a combination of of the SDE (structured data entry), canned text, and slot-and-filler approaches [3]. The translated values can be seen as canned text segments, though they are not intended to be self-contained, i.e. they are intended to be placed in the context of a template.

The user is presented with a form during data input, but the user-entered data values should be seen as atomic values in the instance data, and not as complete phrases inserted into a generated clinical document. As discussed in [2], strict SDE systems do not allow for stylistic variance due to the rigid one-to-one mapping between entered data and phrases in the generated output. Our system decouples the user-entered instance data from the output, and provides the clinicians with the ability to create their own templates and translators, and thus to have full control over the appearance of the generated document. In practice, however, the clinicians have not worked much themselves with the text generation components, but instead an 'expert user' has been assigned to create a standard set of components that all use. We do not think this being a mere consequence of the component creation procedure being too complicated, but more an indication of a lack of time and resources for the clinicians to create their own personalized generation components.

Another drawback of a strict SDE approach is the temporal decoupling of data entry and report generation [2], not allowing the clinician the option to see how actions performed in the GUI affect generated output. In the MedRecords feeder application in which the clinician enters the instance data, we present the user with an option to preview (using a specified template and translator combination) how the generated document will look during the input process. We also have ideas on providing direct generation feedback alongside the input form.

Lack of support for certain linguistic phenomena such as aggregation, anaphora, rhetorical relations and referential expressions, along with a very basic implementation of ellipsis and enumerative phrases, leaves some to be desired in the quality and fluency of the generated text. The open-sourcing of the text generation framework, along with the sample extension used in the MedView project, provides interested NLG researchers with the opportunity to study and improve on text generation in a framework which has been in practical use for several years.

5 Conclusion

The text generation system used in MedView is deemed sufficient for daily use, though there is ample room for improvement in the quality of the generated text. It is our hope that the open-source initiative will attract researchers and projects within the NLG domain, and that our system provides an interesting platform on which to build more advanced text generation solutions.

6 Acknowledgements

I wish to thank Olof Torgersson and Mats Jontell, which have greatly contributed to this work. The work presented in this paper was supported by the Swedish Agency for Innovation Systems.

References

[1] Y. Ali, G. Falkman, L. Hallnäs, M. Jontell, N. Nazari, and O. Torgersson. Medview: Design and adoption of an interactive system for oral medicine. *A.Hasman et al, Medical Infobahn for Europe: Proceedings of MIE2000 and GMDS2000, IOS Press*, 2000.

[2] D. Kraus. Text generation in clinical medicine - a review. *Methods Inf. Med. 2003;42.*

[3] O. Torgersson and G. Falkman. Using text generation to access clinical data in a variety of contexts. *Surján, G., Engelbrecht, R. & McNair,*

P. (eds.): Health Data in the Information Society, vol 90 of Studies in Health Technology and Informatics, IOS Press, pages 460–465, 2002.