Implementation of a System for Intelligent Summarization of Longitudinal Clinical Records

Ayelet Goldstein and Yuval Shahar

Ben Gurion University of the Negev, Beer-Sheva, Israel
{gayelet,yshahar}@bgu.ac.il

Abstract. Physicians are required to interpret, abstract and present in free-text large amounts of clinical data in their daily tasks. This is especially true for chronic-disease domains, but also in other clinical domains. In our previous work, we have suggested a general framework for performing this task, given a time-oriented clinical database, and appropriate formal abstraction and summarization knowledge. We have recently developed a prototype system, CliniText, which demonstrates our ideas. Our prototype combines knowledge-based temporal data abstraction, textual summarization, abduction, and natural-language generation techniques, to generate an intelligent textual summary of longitudinal clinical data. We demonstrate both our methodology, and the feasibility of providing a free-text summary of longitudinal electronic patient records, by generating a discharge summary of a patient from the MIMIC database, who had undergone a Coronary Artery Bypass Graft operation.

1 Introduction

Many clinical tasks require dealing with an enormous amount of time-oriented patient data. Physicians, who have to make diagnostic or therapeutic decisions regarding these patients, may be inundated by the volume of data if their ability to reason does not scale up to the amount of data.

We provide a verbal (free-text) summary of electronic patient records that include time-stamped data that have accumulated during an extended time period, such as during hospitalization, or over years of medical care. Such summaries might help care providers in their daily tasks, and support several decision-making processes.

In our previous work [1], we had proposed an architecture that supports a process of transforming longitudinal data into an intelligent, concise, text-based summary. In the current study, we have implemented the proposed architecture as a prototype System – the **CliniText** system. In this paper, we describe the inner workings of the CliniText system, using a detailed example of how each module contributes to the process of automatically transforming time-based data from a typical case in the MIMICII public database, into a discharge summary.

The input to the CliniText system includes longitudinal data and a domain-specific knowledge. The output of the system is a condensed textual summary of the patient's data. In our current prototype, the output textual summary is in the form of a

D. Riaño et al. (Eds.): KR4HC 2013/ProHealth 2013, LNAI 8268, pp. 68–82, 2013.

discharge summary, since we aim to generate text that resembles the corresponding human discharge summary of the same data, as found in the database.

2 Background

Visualization is an effective approach to facilitate analysis and presentation of high volume data, specifically when dealing with huge amounts of data [2]. However, recent findings showed that graphical representation is not always more effective than other methods, and, in the medical domain, clinical decision-making was not necessarily improved by the use of a graphical display. Another study [3] showed that when dealing with large volumes of complex clinical data, a textual presentation is even advantageous over a graphical one, for the purpose of certain clinical tasks.

Temporal abstraction (TA) is the task of producing context-sensitive and qualitative interval-based representations (interpretations). The output of the TA task can be defined as a set of time intervals, each interval representing a certain state holding over a period of time, with its respective context-sensitive concept value [4]. In the medical domain, in which data abstraction is crucial, decision making can be greatly benefited by information (trends, irregularities) derived directly from the data [5].

The *Natural Language Generation* (NLG) task deals with the generation of natural language from a machine-language form input [6]. Although the NLG task has been implemented in different domains [7], in most of the implemented systems the data are relatively well-defined, not requiring advanced data analysis techniques. Furthermore, in the case of the summary of small data sets, only brief summaries are produced, which significantly reduces the complexity of many NLG tasks. In the medical domain, existing NLG systems are far from optimal [7]. A more recent NLG system, BT-45[8], which focuses on decision support, also performs temporal data abstraction; however, the abstraction process is relatively simple, and does not consider different contexts when determining the importance of an event. In addition, it considers only short and pre-defined periods of data. Dealing with different periods of time, especially longer periods, as is common in chronic-disease patients, requires additional temporal-information handling techniques.

The main contributions of our system, are that the input data are allowed to be (1) heterogeneous, which makes the NLG task significantly more complex, (2) of high density, which means that the summaries will not be brief and (3) longitudinal over unlimited time periods. Unlike the approach we took, most existing approaches are not based on the use of complex knowledge specific to the application domain, and thus cannot automatically create any meaningful domain-specific interpretations. Furthermore, these systems cannot decide what data or interpretations are potentially redundant by using a robust domain-specific knowledge base (KB) and formal interpretation theory. Similarly, they cannot determine which facts are crucial to report in different contexts. Moreover, existing systems do not have the capability for interactive exploration of the resulting text summaries at various domain-specific, semantically meaningful and levels of abstraction, a capability for which our system provides the infrastructure.

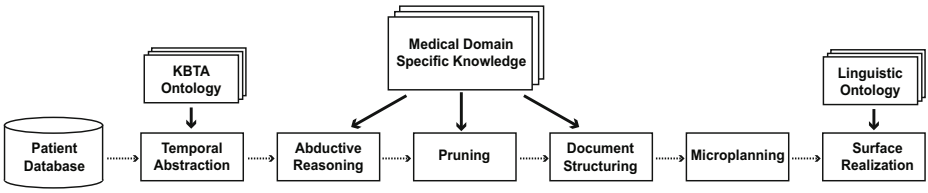

Fig. 1. The architecture of the Clinitext System. Knowledge is used by the temporal abstraction module to abstract the data. Additional knowledge is used in further modules to perform abduction, pruning and document structuring of the data. In the microplanning and surface realization modules we use standard medical dictionaries, such as UMLS to realize the text. Dashed horizontal arrows = control/data flow; Bold vertical arrows = knowledge flow.

3 Methods

Our input data arrives from a time-oriented clinical database. The required medical knowledge is specified through a graphical knowledge specification tool called Gesher [9]. We define both declarative (e.g. What Is mild-anemia in a pregnant women) and procedural (e.g. How to manage the patient, by administering two drugs in parallel) knowledge, necessary in the abstraction process. As we show, the knowledge is also exploited in other modules of the system.

We implemented the CliniText system using a framework composed of several modules, each performing a specific task. The architecture of the system, composed of six main modules, is displayed in Figure 1.

The flow of the data between the components defines the workflow of the process: Longitudinal raw data coming from the time-oriented database is abstracted by the *Temporal Abstraction* module, adding abstract data to the original raw data. Additional data are inferred from it through abduction in the *Abductive Reasoning* module. Raw, abstract and abducted data, are then pruned in the *Pruning* module, responsible to select the important and domain-relevant data, leaving only the information that will take part in the final text. The *Document Structuring* module structure these data, determining the order they should appear in the final text. The *Microplanning* component groups and prepares the structured data to the format expected by the *Surface Realization* module, which finally realizes the data into the final text.

Below we describe each component and its contribution to the process.

3.1 Temporal Abstraction (TA)

The TA step is responsible for the performance of the abstraction of time-oriented data. In the implementation of this module, we used a variation of the IDAN temporal-abstraction mediator [10], which implements Shahar's Knowledge-Based temporal-Abstraction (KBTA) Method [4]. The input to the TA module is time-stamped raw concepts (e.g., red-blood-cell "RBC-BLOOD" values). In Table 1 we list an example of input raw data and the output data, derived abstractions of the raw concepts. Note: The number of input concepts can be different from that of derived abstractions.

Table 1. A: An example of the input to the temporal-abstraction module. B: an example of the temporal-abstraction output. The first column lists the concepts; the second column specifies the number of *instances* in the database (e.g. 7 raw measurements of White Blood Cells Counts (WBC) denoted in the database as the raw concept "WBC-BLOOD"). Note: two or more instances of a raw concept can be transformed into one abstract concept interval, through the KBTA method's temporal interpolation (as is the case for the hematocrit state abstraction). Furthermore, one abstraction might be a function of two or more raw data-type instances.

A – Examples of the input of the TA process step	
Raw concept	**instances**
WBC-BLOOD	7
RBC-BLOOD	7
Hematocrit	9
Heart Rate	112

B – Examples of the output of the TA process step	
Abstraction concept	**instances**
WBC State	7
RBC State	7
Hematocrit State	8
Heart Rate State	86

The abstraction of the data is performed using the KB defined in Gesher. Figure 2 shows an example of the partial declarative knowledge definition for the derivation of the abstract concept "WBC State" from the raw concept "WBC-BLOOD".

3.2 Abductive Reasoning

The abductive reasoning step is essential, since not all important concepts, such as events or actions performed, can always be found in the input data. However, in some cases, these events can be inferred by abduction from the existing data, with a high probability, albeit not necessarily with complete certainty, using *abduction knowldge*. For example, knowing that a Swan-Ganz catheter (SG) is used to measure the pulmonary artery systolic/diastolic pressure (PAP) concept, even though the event of inserting a SG (SG-in) does not appear in the database we can hypothesize that the event SG-in occurred through an instance of the PAP concept in the database.

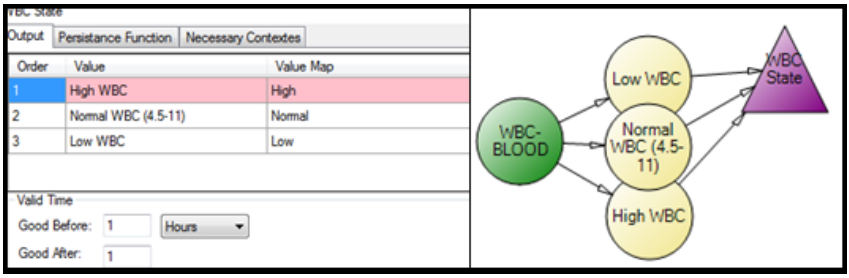

Fig. 2. The Gesher tool declarative-knowledge derivation of the abstract concept "WBC-State" from the Raw concept "WBC-BLOOD". On the right, the abstraction definition of the raw concept into the "WBC State" abstract concept: The raw-concept is initially mapped into value-abstractions which abstract the different values of the WBC-BLOOD (e.g. "Low WBC" = WBC-BLOOD < 4.5). These value-abstractions are then mapped into the different values of the State abstraction, as shown on the left. (e.g. "Low WBC" is mapped to the value "Low").

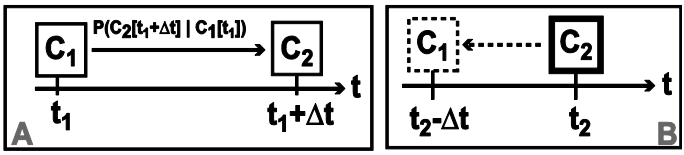

Fig. 3. A: An example of abduction knowledge: concept C_1 is followed with probability $P(C_2[t_0+\Delta t_i] \mid C_i[t_0])$ by concept C_2 within a period of up to Δt. B: The abduction process: From concept C_2 with a timestamp of t_2 we abduct concept C_1 with a timestamp between t_2 and t_2-Δt.

We can say that knowing that concept C_1 enables the existence of concept C_2, within a certain time Δt, allows through the existence of C_2 in the database, with a timestamp of t_2, the abduction of C_1 with a timestamp between t_2-Δt and t_2, as shown in Figure 3.

Formally, we compute ahead of time $\forall i\ P(C_i[t_2 - \Delta t_i] \mid C_2[t_2])$: The probability of each concept C_i, given the concept C_2, t_2 (timestamp of C_2), P_0^i (a-priori probability of each related concept C_i), $P(C_2[t_0+\Delta t_i] \mid C_i[t_0])$ (relation probability of C_2, given C_i) and Δt_i (the interval of time during which C_i can be interpreted. For simplification, we consider only a uniform distribution of probabilities.). The computation applies the standard Bayes theorem, as used in Bayesian diagnostic problem solving [11].

From a knowledge representation point of view, the abduction knowledge typically links the declarative knowledge to the procedural knowledge, although other combinations are possible. (An example of linkage between two instances of procedural knowledge could be the event "syringe-insertion" allowing the abduction of the event "skin-sterilization" 5-10 seconds before). The relationship between the concepts can be either causal (A causes B) or associative (A occurs together with B). Note that in general, the temporal relation between the two concepts might include other temporal relations, such as "overlaps". At this point, we are focusing only on the "before" temporal relation.

To perform abduction, we need to explicitly represent abduction knowledge in the KB. For example, we need to specify beforehand in the procedural knowledge for each event (e.g. SG-in) what are the declarative concepts (e.g. PAP) it enables, and what's the probability/strength of this relation (e.g. if there are other events that could also generate this declarative concept). For example, the event SG-in, with a prior probability quite low, enables PAP with P=0.95; However, in the ICU, it is the only method used to measure PAP. Thus, within the ICU context, P(SG | PAP)=1. Note that in a different context, outside of an ICU, the PAP S/D concept value would be commonly measured using an Ultrasound (U/S) procedure, and not using a SG catheter.

In our case, we enable the inference of the existence of a procedural action (e.g. SG-in) by the existence of a declarative concept (e.g. PAP). Since we deal with a medical domain, we abduct only concepts that have a probability near to certainty.

Fortunately, we usually work in domains in which, although the probability of the existence of the database item that is *accessible* (*explicit*) given the one that is *hidden* (*implicit*) is not necessarily 100% (for example, the SG catheter does not necessarily

lead to measurement of PAP S/D with certainty-that depends on the success of the procedure), the number of potential implicit concepts that can be inferred from those that explicitly exist in the database, Ci (i = 1..N), is such that N = 1 (e.g., there is only one potential enabler to the measurement of PAP S/D, as rare as that enabler might be); or there might be a huge difference in the prior probabilities, so we can still infer with certainty, or at least very high confidence, the abducted event.

If that is not the case, and there is any significant doubt, and we suspect that even a very likely implicit event or concept inferred from the explicit data is potentially hazardous to infer, we prefer to not include the inferred event in the text, recognizing the huge cost of adding a false item, compared to the omission of an inferred event.

We denote events abducted from declarative concepts as *implicit interventions*.

In table 2, we show an example of interventions inferred through abduction.

Table 2. Declarative concepts used to generate "implicit interventions". Each implicit intervention can have one or more values (e.g. Arterial Line can be "inserted" or "removed"), according to the values of the raw concept it was deducted from.

Declarative (raw) concept(s)	Abducted events (Implicit interventions)	# instances
[PAP S/D]	Swan Ganz catheter (PROC)	1
[CVP], [Arterial BP]	Arterial Line (PROC)	3
[Airway Size], [ETT Mark/Location]	Artificially breathing (PROC)	1

3.3 Pruning

After enriching our original data with abstractions and abducted interventions, we need to select which information is important and will appear in the final summary and which is unnecessary and should be removed to avoid overloading the user.

We prune the data by using general heuristics and several "independent" parameters (E.g., text length, detail level, profiles, etc). We use *pruning-heuristics* which define which data should be pruned and *maintaining heuristics*, defining which data instances should be maintained (and which override the *pruning-heuristics*).

The current *Pruning-heuristic* include: (1) Ignoring expected or "standard" values, within a specific context; during the knowledge acquisition process, we tag certain concept values as "Non-descript" (i.e., not to be mentioned without a special reason). (2) Preferring derived concepts over the concepts from which they are derived; the user can still see the raw concept from which the abstraction is derived by navigation of the text, since we keep links between the abstracted concepts and their deriving concepts. (3) When several interpretations of the same concept exist, within multiple contexts, report only on the interpretation within the most specific context.

Maintaining heuristics include: (4) Data instances that must be described; this is determined according to the domain, although certain events appear in every domain, such as death of the patient. (5) Maintaining extreme raw data values; In this case, even though they have an abstract value, we might want to maintain also the raw data.

In table 3 we show some of the heuristics used to prune the data.

Table 3. Raw and abstracted concepts before and after the pruning step, with the respective heuristic that was used in pruning: (H1) ignoring expected or "standard" values; (H2) Giving preference for derived concepts over the concepts from which they are derived; (H3) Within multiple contexts, report the interpretation with the most specific value. (H4) Data always described; (H5) Maintaining extreme raw data values

Raw concepts			
Concept	**Before**	**After**	**Heuristic**
Date of Birth	1	1	H4
Gender	1	1	H4
WBC-BLOOD	7	0	H2
Hematocrit	9	0	H2
PAP S/D	40	0	H2

Abstracted concepts			
Concept	**Before**	**After**	**Heuristic**
WBC State	7	0	H1
RBC State	7	2	H1
HematocritState	8	8	none
Ectopy status	8	2	H1
ArterialBP State	8	4	H1

3.4 Document Structuring (DS)

After considering which data will appear in the final text, we define how these data will be presented. In the DS module it is decided how much information will be expressed in each phrase, in which order the facts will appear, how they will be organized into paragraphs, etc. The structure of the final text varies according to the output format expected as well as to the domain.

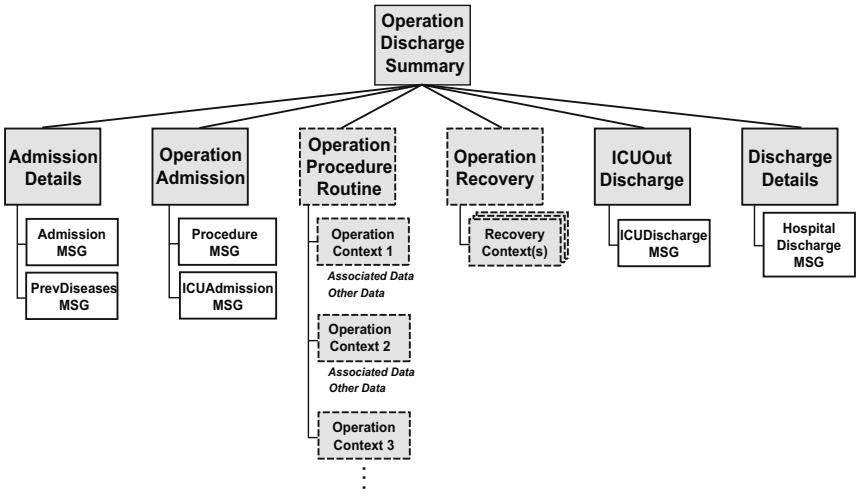

Fig. 4. The DS structures the final text into a tree. The tree defines how the data will be presented – the order, the paragraphs division and also the relation between events. (Gray rectangles are *Document-Block* objects, white rectangles are *Document-Message* objects. *Document-Blocks* can contain one or more *Document-Messages* or *Document-Blocks*.) Dashed-lined blocks represent flexible segments whose structure is defined by the data. Non-dashed blocks represent rigid segments of the text, whose structure was pre-defined.

We use two approaches for the DS: a top-bottom approach, which defines the main structure of the final text (e.g. division of paragraphs), and a bottom-up approach, which groups and combines events until all the events are linked together, by using appropriate discourse relations. The top-bottom approach works well when the structure of the text is very predictable and can be defined beforehand, but is less suitable for structures that vary and cannot be expected in advance. On the other hand, the bottom-up approach is less effective for the creation of paragraphs, and can take a lot of time which is needed to perform extensive searches and optimizations.

The output of this module resembles a tree. We prepare ahead of time, for each domain, a "stub" of the tree which includes the main textual segments in that domain, in a top-bottom approach. We build up from the pruned data a set of intermediate nodes which we try to link to the most appropriate node in the pre-prepared stub, in a bottom-up fashion. We call the leaves of the tree *Document-Messages* and the inner-nodes of the tree *Document-Blocks*. The tree has some parts that are more structured, part of the "stub" pre-defined, and others that are more flexible, with their structure affected and defined by the patient's data. In Figure 4 we show the tree used to create a Discharge Summary text. It defines for example, that each immediate child of the root ("Operation discharge Summary") becomes a new paragraph in the final text.

Document-blocks can be rigid or flexible. For example the Document-blocks: "Admission Details", "Operation Admission", "ICUOutDischarge" and "Discharge Details" have a rigid structure, which was mostly defined before hand, while the blocks "Operation Procedure Routine" and "Operation Recovery" are more flexible and their context will be determined and modeled based on the patient's actual instances of data associated with them. Document-Messages can be generic, as in the case of Drug-Message, Intervention-Message or Data-Message; or specific such as: Admission-Message, PrevDiseases-Message and ICUAdmission-Message.

Structured-blocks will be populated with the specific Document-Messages associated with it. For example, the structured section "Admission Details" will be populated with Document-Messages: Admission-Message and PrevDiseases-Message. Each specific Document-Message has specific data associated with it. Less-structured blocks may contain further Document-Blocks as well as generic Document-Messages. The creation of further Document-Blocks is affected by contexts that the data may generate. For example, the less-structured Document-block OperationRoutine will have further Document-Block children generated by the contexts associated with a specific operation event. Note that in general, the TA process applied by the KBTA method in the first step already generates, in addition to the temporal abstractions, all of the domain-specific contexts.

We denote block-context a document-block that was generated by a context. Each block-context has a start-time and end-time defining when the context starts/ends. This is defined by the start/end time of the event that generated the context and the definitions of the context duration. Figure 5 shows an example of block-contexts created by the CABG-operation event within the context of a CABG operation.

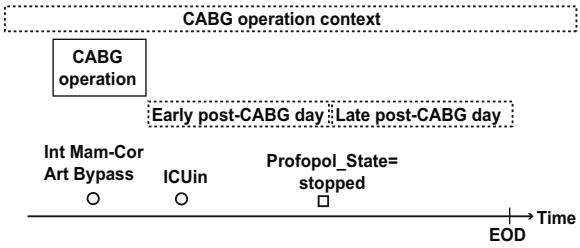

Fig. 5. Raw concepts are denoted by a circle while abstractions are denoted by a square. Contexts are denoted by dashed rectangles and events are denoted by a lined rectangle. The raw concept "Int Mam-Cor Art Bypass", in the context of a CABG operation, creates the event "*CABG Operation*". The event generates two contexts: *"Early-postCABG day"* and *"Late-postCABG day"*. The start time of the *Early post-CABG day* context is defined by *"ICU In"* and by "*Propofol drug State"*. The "*Late post-CABG day"* context has its StartTime equal to the EndTime of context "*Early post CABG day"* and its EndTime set to the "*CABG operation" event* end of day (EOD).

Data or events that occurred during the context interval are associated with the respective block-context and will become Document-Messages associated with that block.

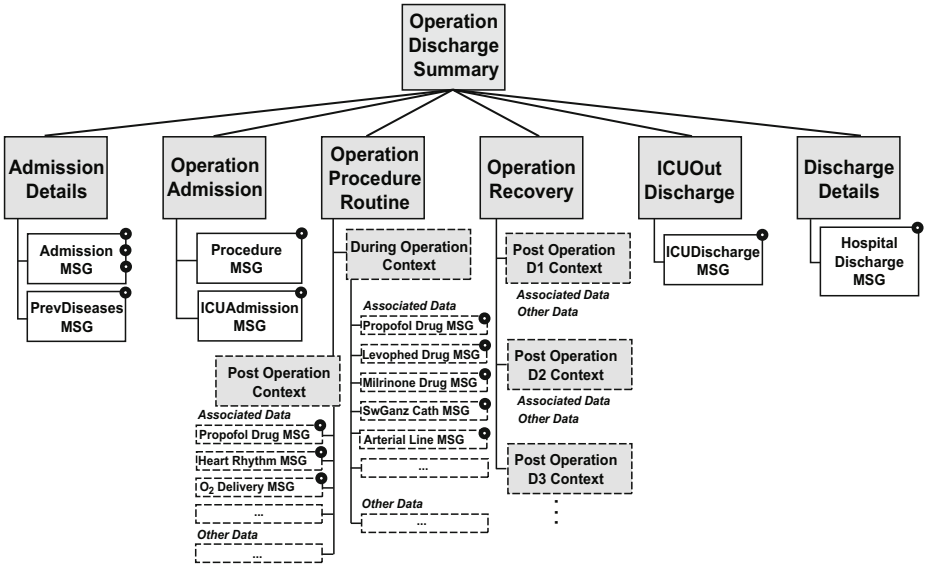

Fig. 6. A DS tree populated by patient's data. Structured blocks (e.g. AdmissionDetails) have specific messages associated to it (e.g. AdmissionMSG). Each Message block hold the patient data (circles). Less-structured blocks (e.g. OperationProcedureRoutine may have further blocks associated to it, according to the contexts generated by the data (e.g. the "*Early-postCABG day*" and "*Late-postCABG day*" context blocks, generated by the CABG operation event). Each context block has *associated* messages containing data related to the respective context interval, followed by *other* date that occurred during the interval.

Each context may have associated data/events, that we expect to occur within this context – based on what was specified in the Guideline (procedural knowledge). Within one block-context we will first add the expected associated data/events, followed by all other data that occurred within that context interval. For example, within the context "Early-postCABG day" we expect to have the events: "insertion of Swan-Ganz", and "start of artificial-breathing" between others. If the expected events indeed occurred we add respective messages to the “Early-postCABG day” block- context; then we add all additional data that took place during the interval.

The structure of the tree establishes the structure of an operation discharge summary for any given patient that has gone through an operation. To produce a Discharge Summary specific for a given patient, we need to populate the tree with the patient’s data. An example of a populated tree can be seen in Figure 6. After populating the tree, by traversing the tree in a Depth First Search, we generate the order of the events in the final text.

3.5 Microplanning

In the Microplanning step, we (1) define the words and syntax structures that are used by the final text generator, (2) perform aggregation of the data and (3) generate referring expressions.

In the aggregation process, we define how much information will be expressed in each sentence. Although aggregation was also performed in previous steps, (e.g. in the TA step, which generates abstractions from the raw data) the aggregation here is done at a sentence level. In the aggregation process the semantic order has preference over the chronological order kept until now. Taking into consideration semantic considerations, we group together messages according to their type (drug, intervention or data message types).

In the case of drug messages, we group together different (drugs) with the same value. E.g. instead of: "drug X was started, drug Y was started and drug Z was stopped" we will have: “drug X and Y were started and drug Z was stopped". Note that messages are grouped only if they occur within the same context, although not necessarily during the same time. In the case of interventions and data messages, in addition to grouping different concepts with the same value, we also group same (interventions or data) concepts appearing more than once with the same or variant values. E.g. generating instead of: "Blood transfusion was given, blood transfusion was given" the sentence: "Blood transfusion were given twice [or *n* times]"; or instead of: "Atrial wires were inserted, … atrial wires were removed" the sentence: "Atrial wires were inserted then removed".

Referring expression generation is the process of using pronouns to refer to entities. It creates a more natural and human style of the text. While the use of referring expressions can help avoiding repetition, the danger of its wrong usage is not

knowing exactly which entity we are referring to. In medical texts, ambiguous meanings can have catastrophic consequences, therefore, we opted for a very conservative and minimalist use of referring expression usage. An example of its use can be seen in the following phrase, (note also the aggregation), in which the second appearance of "the patient's" in a single phrase is substituted by the pronoun his: "The patient's hematocrit, arterial bp and spo2 were Low and **his** temperature was afebrile."

The exact format of the microplanning output depends on the expected format of the step following it: surface realization.

3.6 Surface Realization

Surface Realization is the module responsible for generating the final text. Consequently, this step is language dependent, and observes the grammar rules of the language chosen. We opted to implement the Surface realization module using the existing Java module "SimpleNLG" [12]. Since our previous steps were implemented in C#, using the SimpleNLG module required us to transform our C# objects into Java objets. (Serialize the C# object and unserialize it into a Java object). This was implemented by using Google's protocol buffer whose advantages compared to other serialization methods are described elsewhere [13].

The realization of the text is done per section. Sections were defined within the tree structure, during the DS step. When the tree is flattened, the sections are kept and the flattened list of messages is divided into these sections. Each section is realized separately and in the same order it appears in the flattened list. One section might have sub-sections (generated by contexts). For example, the OperationRoutine section has two contexts: "Early-postCABG day" and "Late-postCABG day". Each context (or sub-section) is also realized separately.

Each section has a relationType (defined in the DS step) defining how messages within the section are related. For example the "sequence" relation defines that they should be realized one after the other, while the "causal" relation defines a cause-effect relationship between the messages.

Rigid (flexible) document-blocks, become rigid-structured (flexible-structured) sections, in the microplanning phase. The realization of structured sections is usually quite straight-forward while the realization of flexible-structured sections first go through aggregation and referring expression before being realized into their final text.

Table 4 compares our generated text summary with the original human-generated discharge summary found in the database. Part of the original summary could not be reproduced, since there was no data in the database that would support it. Presumably, such parts were written using some external data source, or perhaps textual progress notes.

Table 4. Comparison between the human and CliniText generated texts. Striked-trough text between brackets [] indicate text-parts that cannot be generated, i.e., without supporting data. Bold text in the right (left) column indicate data that do not appear in the other column, although they could.

The human discharge summary	The CliniText discharge summary
(1) HOSPITAL COURSE: On **3285-5-17**, the patient was brought to the Operating Room for a redo coronary artery bypass grafting [~~times two and aortic valve replacement.~~] (2) The patient [~~tolerated the procedure well and~~] was transported intubated to the PACU **in stable condition** on a Levophed drip, Milrinone drip, and a Propofol drip. The patient had atrial wires and a chest tube. (3) On postoperative day #1, the patient was continued on his Levophed drip and Milrinone drip over night. The patient was extubated over night and was on a nasal cannula of 4 Lungs: **oxygen saturation 95% on room air**. The patient's **vital signs were otherwise stable**. (4) The patient had a postoperative hematocrit of 26.8; otherwise laboratory values were all within normal limits. [~~The patient was encouraged to be out of bed.~~] Drips were weaned. [~~The patient was started on his Plavix.~~] (5) On postoperative day #2, the patient was off all drips, [~~was started on Aspirin 325 per day and Captopril 6.25 t.i.d.~~] (6) The patient was afebrile in sinus rhythm. The patient had an **oxygen saturation of 96%** on 2 L nasal cannula. The patient had minimal output from the chest tubes. The patient's white count was 9.7, hematocrit 25.4, otherwise other labs were all within normal limits. [~~The patient's chest tubes were removed, and~~ ~~**the patient was transferred to the floor**~~.~~On postoperative day #2, the patient was also seen by Physical Therapy.~~] (7) On postoperative day #3, the patient was still in the Intensive Care Unit. [~~The patient was kept in the Intensive Care Unit due to bed availability.~~] (8) The patient was **afebrile with a T-max of 99.7??**. The patient developed atrial fibrillation over night; however, the patient was rate controlled. The patient had an **oxygen saturation of 96%** on 2 L nasal cannula. (9) The patient's hematocrit on postoperative day #3 was 23.9. Other lab values were all within normal limits. [~~The patient was started on Lopressor 12.5, Heparin drip at 600.~~] The patient's wires were removed. (10) The patient self-converted to normal sinus rhythm with **occasional premature atrial contractions.** The patient remained in normal sinus rhythm~~,[and his Heparin drip was held. The patient was continued on Lopressor and Captopril, and tolerating it well.~~] (11) On postoperative day #4, the patient's hematocrit was 22.3. The patient was transfused 2 U packed red blood cells. [~~The patient's Lopressor was increased. The patient was continued on Captopril.~~] The patient was transferred to the floor .	*(1) "On the 15/05/2005 Mr.97, a married, 82 years old patient, was admitted to the hospital. The patient's previous history includes peripheral_vascular disease(s).* *(2) On the 17/05/2005 the patient went through a "CABG" operation and was admitted to the ICU.* *(3) In the first hours following the CABG procedure, propofol, levophed and milrinone were started.* ***Swan-ganz catheter, arterial line, chest tube, atrial wires and vent wires were inserted,*** *artificially breathing was started,* ***heart rhythm was Paced.*** *The patient's hematocrit,* ***arterial bp and spo2 were Low,*** *temperature was Afebrile and his urine source was through an external catheter.* *(4) Later on, propofol was stopped. O2 delivery device was Nasal Cannula, heart rhythm was Normal sinus* ***and blood transfusion was given.*** *The patient's O2 Flow State* ***was 3*** *then 4 ; hematocrit* ***and urine output*** *were Low.* *(5) On post-oper day #1, the patient's* ***Arterial BP State remained Low*** *and Hematocrit State remained Low ;* ***o2 flow was 2****. Levophed and milrinone were stopped.* *(6) On post-oper day #2, the* ***patient's BP State remained Hypotension*** *; hematocrit was Low, ectopy status was Abnormal, heart rhythm was Atrial Fibrillation and his* ***arterial bp was High****. Arterial Line (PROC) was removed then inserted;.* *(7) On post-oper day #3,* ***the patient's BP State remained Hypotension*** *; o2 delivery device was Nasal Cannula, o2 flow was 2, hematocrit was Low, heart rhythm was Normal sinus****, rbc was Very low, ectopy status was Abnormal****, o2 delivery device was None,* ***nbp_diast was 62 and his nbp_sys was 130.*** *Atrial wires (PROC) was removed then inserted ; vent wires was inserted.* *(8) On post-oper day #4, the patient's O2 Delivery Device remained None and* ***BP State remained Hypotension ;*** *hematocrit was Low, rbc was Very low and* ***his urine source was Independent****. Blood Transfusion (PROC) was given twice ; vent wires was removed.* *(9) The patient was discharged from ICU on 21/05/2005.* *(10) The patient was discharged from hospital on 23/05/2005."*

3.7 Preliminary Evaluation

As a source for the longitudinal clinical data used in the examples, we used data from the Multiparameter Intelligent Monitoring in Intensive Care II database (MIMICII) [14]. There are two types of data in the MIMICII database: clinical data (numeric and textual) and bedside monitor waveforms. Waveforms in the Mimic DB are in fact rapidly sampled (125Hz) digitized signals recorded by the bedside monitors, such a electrocardiograms (ECG) and arterial blood pressure (ABP). Thus, to summarize MIMIC-II "waveforms", we would use the numeric representation of the signals underlying the "waveforms", in which each sample data instance would be represented by a raw data concept; and we could then abstract these data using relevant knowledge, precisely in the same way that we are already applying the temporal-abstraction process to all of the patient's other clinical data. In our current example, we used only clinical data, and mainly numeric data. Textual data was used only to compare our generated text to the original discharge summary. We focused on data of patients in the cardiac domain, more specifically on patients that went through a CABG procedure.

Our current evaluation, an example of which was shown in Table 2, is purely technical, judging the feasibility of the overall process, comparing the generated text to the MIMIC texts, without involving yet any users.

Judging by examples we have gone through such as shown in Table 2, the whole process is quite feasible and shows definite promise; but we have learned from our current preliminary experiments that much additional domain knowledge needs to be added.

Our initial analysis suggests that the amount of effort required to specify the necessary knowledge for a medical domain depends on the information expected to be included in the final summary, and on the amount of input concepts. In addition, each domain will also require a modification of the templates used to realize the text in the rigid blocks.

4 Discussion and Conclusions

It can be noted that in the CliniText summary text there are data items that do not appear in the original text. If some of these data are redundant (something to be defined by a clinical expert), we could prune them by defining additional heuristics in the pruning step, for example.

Another interesting point to be observed is that in the original text the levels of abstraction *varies* as compared to the levels of abstraction that appear in the CliniText summary, in which we always strive to have a uniformly high level of abstraction (e.g. "hematocrit=22.3" vs. "hematocrit state=low"). Allowing the user to *navigate* into the raw concepts that the abstraction was abstracted from can provide them with lower levels of abstraction when required. We are currently adding this capability.

We must take into account that the original text, because it was produced by humans, may sometimes contain wrong information. In our example (see Table 2), we can see that it's stated in paragraph #6 that the patient was discharged from the ICU,

and in paragraph #7 the mistake is corrected by saying that the patient was actually kept in the ICU due to bed availability. Such mistakes can't occur in the CliniText summary, if the data are correct. On the other hand, when there are bad data, we can't avoid generating a wrong text. An additional layer of data-validity-check (for example specifying that a dead patient can't have a heart rate) could be added to the process to clean the data before the text starts to be generated. The definition of this layer requires the support of a clinical expert, and additional knowledge.

One of the advantages of having a discharge summary text being produced automatically is that the output format is structured and doesn't depend on subjective factors (e.g. how tired is the physician writing it, or how many discharge summaries she already wrote). Furthermore, the information to be included or omitted has been defined by objective consistent criteria, resulting in an objective data-and-knowledge-based summary.

We were limited during the prototype implementation by the clinical knowledge available and also regarding the data we had access to. Having access to clinical data involves legal and privacy concerns; Furthermore, finding a database which includes data and text summaries that can be related to each other isn't always easy.

The main purpose of building the described prototype was to demonstrate the feasibility of creating a system that can generate a complex knowledge-based textual summary of arbitrarily long time-oriented clinical data. We showed that it is possible to produce a readable text, and include the main events and data, according to what was defined in our KB.

In the future, we might try different implementations, for example, using a grammar instead of a tree representation in the Document Structuring step, or other modules of Surface Realization. We intend to evaluate the CliniText system regarding the aspects of *quality* of the generated text, *usability* and *functionality* for relevant predefined tasks. The evaluation process will involve a technical evaluation as well as involve user opinions.

Acknowledgements. Part of this work was supported the EU 7th Framework MobiGuide project, grant No. 287811.

References

1. Goldstein, A., Shahar, Y.: A Framework for Automated Knowledge-Based Textual Summarization of Longitudinal Medical Records. In: KR4HC Workshop, Tallinn, Estonia (2012)
2. Combi, C., Shahar, Y., Keravnou-Papailiou, E.: Temporal Information Systems in Medicine. Springer, New York (2010)
3. Portet, F., Reiter, E., Hunter, J., Sripada, S.: Automatic generation of textual summaries from neonatal intensive care data. AI in Med., 227–236 (2007)
4. Shahar, Y.: A framework for knowledge-based temporal abstraction. AI 90, 79–133 (1997)
5. Lavrač, N., Kononenko, I., Keravnou, E., Kukar, M., Zupan, B.: Intelligent data analysis for medical diagnosis: using machine learning and temporal abstraction. AI Communications 11, 191–218 (1999)

6. McDonald, D.D., Bolc, L.: Natural Language Generation Systems. Springer, New York (1988)
7. Huske-Kraus, D.: Text generation in clinical medicine-a review. Meth. Info. Med. 42, 51–60 (2003)
8. Portet, F., et al.: Automatic generation of textual summaries from neonatal intensive care data. Artificial Intelligence (2009), doi:10.1016/j.artint.2008.12.002
9. Hatsek, A., Shahar, Y., Taieb-Maimon, M., Shalom, E., Klimov, D., Lunenfeld, E.: A scalable architecture for incremental specification and maintenance of procedural decision-support knowledge. Open. Med. Info., 255–277 (2010)
10. Boaz, D., Shahar, Y.: Idan: A distributed temporal-abstraction mediator for medical databases. AI in Medicine, 21–30 (2003)
11. Owens, D., Sox, H.: Biomedical Decision Making: Probabilistic Clinical Reasoning. In: Biomedical Informatics (Shortliffe and Cimino), ch. 3, pp. 80–129 (2006)
12. `https://code.google.com/p/simplenlg/`
13. `https://code.google.com/p/protobuf/`
14. The Laboratory for Computational Physiology, `http://www.physionet.org`