A Graph-Grammar Approach to Represent Context Knowledge in Oncological Patient Records

R. Müller[a], O.Thews[b], C. Rohrbach[a], K. Pommerening[a]

[a]*Institut für Medizinische Statistik und Dokumentation, Universitätskliniken Mainz, Germany*

[b]*Institut für Physiologie und Pathophysiologie, Universität Mainz, Germany*

Abstract: A patient's data collected during complex diagnostic decision making and long-term treatment in oncology do not simply form a chronology of events, but build up a network of various relationships between numerous units of information. Such data contexts include causal or temporal relationships, they express diagnostic inconsistencies and revision processes or describe patient-specific heuristics. Data may be grouped together according to problems, and a single data item may participate in several contexts. The knowledge of data contexts supports the retrospective understanding of the medical decision making process and is a valuable base for further treatment. Most electronic records and underlying patient models neglect this problem of meta-knowledge about the raw data or just provide additional free text attributes. In connection with the development of the knowledge-based system **TheMPO** (**The**rapy **M**anagement in **P**ediatric **O**ncology), which supports therapy and monitoring in pediatric oncology, a graph-grammar approach has been used to implement a graph-oriented patient model which allows the explicit representation of non-trivial relationships among events. For context acquisition a mouse-based tool has been developed to specify relationships in a comfortable graphical manner. Furthermore, the retrieval of patient-specific contexts is realized with graphical tools as well.

1. Objectives

The data of a patient who is undergoing complex diagnostic and therapeutic procedures in oncology do not only form a simple chronology of events, but are closely related in manifold ways. Such data contexts include causal or temporal relationships, they express diagnostic inconsistencies and revision processes or describe patient-specific heuristics. The knowledge of data contexts supports the retrospective understanding of the medical decision making process and is a valuable base for further treatment. Conventional data models often neglect the problem of context knowledge or simply use free text attributes which are not processed by the program. Several authors in the field of medical informatics have therefore claimed that a medical record should be able to represent this sort of meta-knowledge in a structured way and, in particular, should capture the medical decision making process [1,2,3,4].

This paper introduces a graph-grammar approach to represent data contexts. The approach has been used to develop the electronic patient record of the knowledge-based system **TheMPO** (**The**rapy **M**anagement in **P**ediatric **O**ncology) [5,6], which supports long-term treatment and monitoring in the domain of childhood cancer.

First of all a classification of data contexts is suggested. In section 2 the graph-grammar methodology [7] is introduced and applied to the context problem: All knowledge about a patient - and especially the context knowledge - is mapped to an attributed graph which is build up and controlled dynamically through a set of graph-grammar production rules. In section 3 the current patient graph-grammar is described, and it is shown how the contexts contribute in the reasoning process. In section 4 then a mouse-based graphical tool is introduced which enables the physician to specify and retrieve contexts among data interactively. Implementation issues are addressed in section 5, and limitations and open problems of the current approach are discussed in section 6.

1.1. Classification of Data Contexts

In the domain of cancer therapy and monitoring, the explicit representation of the following types of clinical contexts has been considered necessary:

1.1.1 Causal Contexts

Causal contexts include the justifications of diagnoses, drug administrations and drug modifications. If an electronic patient record explicitly represents the connections between a diagnosis and the establishing data, or between the discontinuation of a drug and the laboratory values indicating a serious toxicity problem, the program can quickly answer questions such as *Why has this diagnosis been established?* or *Why has this obligate cytostatic drug been discontinued after 3 days?*

1.1.2 Temporal Contexts

Data and observations are often difficult to interpret without information about their temporal context. In the domain of cancer therapy, a single laboratory value for example is useless without knowing the preceeding values and the particular temporal context (e.g., which chemotherapy or radiotherapy has been administered before or at the same time). By representing temporal contexts such as *Trend of laboratory value L* or *Chemotherapy of type T* explicitly, and not implicitly through a set of time-stamped data, the patient record can quickly generate a summary of the main periods of therapy and feed an inference engine with temporal abstractions.

1.1.3 Inconsistency Contexts

The variety of different sources of information - such as statements made by the patient and his relatives, results of several diagnostic procedures; impressions, intuitions and conclusions of the physicians - may lead to inconsistent knowledge about a patient. A new laboratory value may be inconsistent with an established diagnosis, but however, this diagnosis may be maintained because other, more plausible diagnoses are lacking. The knowledge of inconsistencies is essential to avoid diagnostic failure and therapeutic mistakes. If an electronic record explicitly represents inconsistency relationships among data which establish a conflict situation, the program for example can draw the physician’s attention directly to a laboratory value which does not conform to the actual diagnosis.

1.1.4 Substitution Contexts

In clinical care revising decisions is unavoidable. As a simple example one can imagine the case of a weakened patient with leukemia who is antibiotically treated because of a strong suspicion of staphylococcal infection (skin suppuration, phlegmon, high temperature) via vein-catheter. After 2 days without improvement the antibiotic drug is discontinued and replaced by an antimycotic drug. A physician faced with the patient’s situation at the third day of antimycotic therapy and not involved in the previous treatment may want to know why the antimycotic drug has been administered, and not antibiotics. If the record explicitly stores the information that the suspicion of staphylococcal infection has been revised and that the antimycotic drug already is the second attempt to get the persistent infection under control, the program itself can load the relevant data. Furthermore, revision and substitution contexts may range over weeks or months, and may be hidden by a lot of irrelevant data. The representation of such context knowledge therefore saves the physician from the time-consuming process of ‘‘manually’’ searching through the extensive record.

Beyond that, the problem of revising diagnoses and substituting ineffective drugs is an additional argument for the explicit representation of causal relationships in patient records. If knowledge about a patient is recognized as wrong and therefore substituted, the record itself can detect which diagnoses or therapeutic actions - based on the substituted knowledge - are possibly invalid now and have to be reviewed.

1.1.5 Patient-Specific Heuristics

In the course of long-term treatment a patient may develop individual and atypical medical behavior, e.g., in connection with drug response or disposition to infections. Such an atypical behavior may establish heuristics which are closely adapted to the patient’s individual situation and overwrite or locally complete rules of the knowledge base.

In the domain of childhood leukemia for instance, it is possible that a weakened patient repeatedly suffers from life-threatening infections during a long-term chemotherapy, the pathogene each time unexpectedly being a fungus. This would justify the individual heuristic that this patient - extremely prone to fungus infections - should be immediately treated with an additional antimycotic drug in the case of a recurring infection with similar symptoms.

If patient-specific heuristics are represented in the rule-based notation of a supporting knowledge base and stored in the patient record, they can dynamically be linked to this knowledge base with high priority each time the inference engine is reasoning about adequate diagnostic and therapeutic procedures for the patient. Furthermore, an electronic record containing patient-individual heuristics should also represent the connections among a heuristic itself and the events which led to its definition, e.g., the record should - in the case of the leukemia patient - represent the connections among the antimycotic heuristic on the one hand, and the recurring serious infection events and insufficient antibiotic administrations on the other hand. Because patient-specific heuristics usually are based on a greater amount of data and inseparable connected with these data, they can be viewed as a special type of data context.

2. Methods

To represent relationships among patient data in a flexible way, all information about a patient is mapped to exactly one labeled, directed, and attributed graph. An event such as a hemogram, a diagnosis or the administration of a cytostatic drug is modeled through a labeled, attributed node; a causal relationship among two events, for instance, is represented by a directed edge with the label *becauseOf*. Attributed graphs clearly distinguish between ‘structural’ information (e.g., relationships among events) represented by edges, and ‘value’ information encoded by attributes describing event characteristics such as the dosage and administration-time of a drug. Figure 1 and 2 list the node and edge labels which have been used, and Figure 3 shows the graph representation of a revision context.

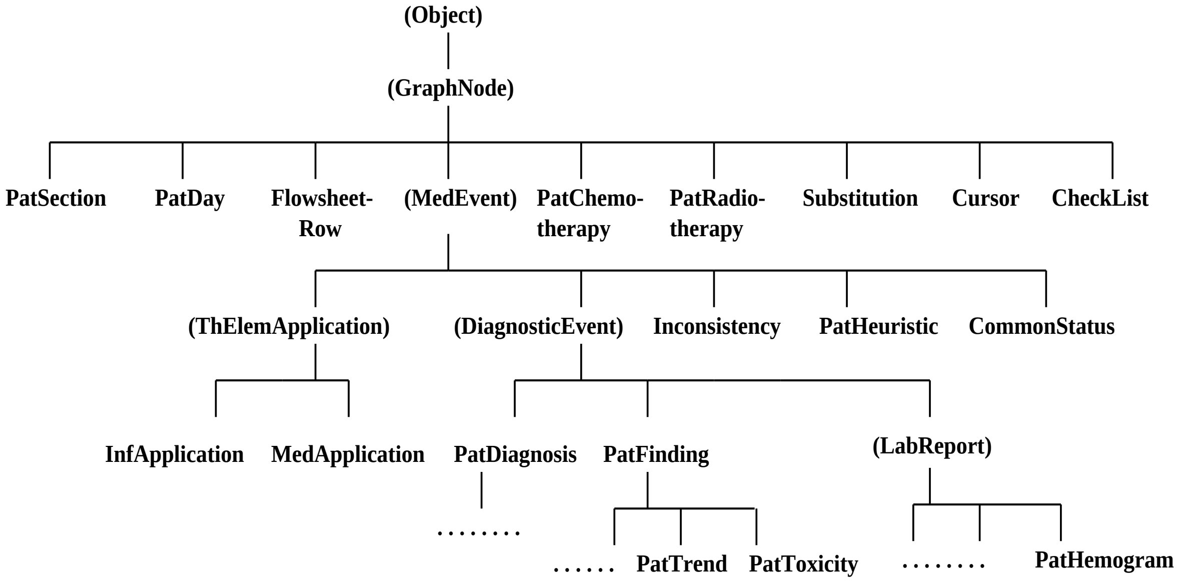


Figure 1: Hierarchy of node labels. Labels enclosed in brackets indicate abstract node labels which cannot be instantiated directly; the nodes in a patient graph must be marked with sublabels. Dots indicate subtrees which have been ommitted in this Figure. Some labels have the prefix ‘Pat’ (for ‘Patient’) to distinguish them from abstract concepts of the **TheMPO**-knowledge base. Most of the labels are derived from the generic label *MedEvent* subsuming labels for diagnostic data, conclusions, and therapeutic actions. The labels *PatChemotherapy* and *PatRadiotherapy* are not derived from *MedEvent* because nodes with these labels group several *PatDay*- and *MedEvent*-nodes.

One main problem arising in connection with graph-oriented data structures is the formal description of the class of legal graphs. A *discontinuedBecauseOf*-edge may point from an *InfApplication*-node (representing the application of an infusion) to a *LabReport*-node, but it is inadequate to run the other direction. To define the class of admissible patient graphs a *graph-grammar* approach has been used. Graph-grammars are a high-level calculus to formally describe the legal manipulations on graphs by defining a finite set of so-called graph productions [7]. In the domain of medicine, this technique has already been used to model medical dilemmas and decisions [8,9]. In the following the theoretical background of the graph-grammar approach is briefly described.

2.1 Graph-Grammars

Given a set of edge labels *E* and attributed node labels *N*, a graph-grammar is a tuple *(S, P)*, where

- *S* is a *start graph*, and
- P a finite set of *graph productions*.

Edges involved in grouping and structuring medical data:

(PatSection) - *hasSubsection* -> (PatSection)
(PatSection) - *hasFlowsheetRow* -> (FlowsheetRow)
(FlowsheetRow) - *hasEntry* -> (MedEvent)

Edges expressing temporal relationships and relating medical events to days and therapy phases:

(PatChemotherapy) - *hasDay* -> (PatDay)
(PatDay) - *hasEvent* -> (MedEvent)
(PatChemotherapy) - *hasMandatoryEvent* -> (MedEvent)
(FlowsheetRow) - *first* -> (MedEvent)
(FlowsheetRow) - *last* -> (MedEvent)
(MedEvent) - *next* -> (MedEvent)
(Inconsistency) - *knownSince* -> (PatDay)

Causal relationships connecting nodes derived from *MedEvent*:

becauseOf, discontinuedBecauseOf, reducedBecauseOf,
increasedBecauseOf, monitors

Substitution contexts:

(PatDay) - *hasEvent* -> (Substitution)
(Substitution) - *new* -> (MedEvent)
(Substitution) - *old* -> (MedEvent)

Inconsistencies, toxicity, conflict situations:

(PatDay) - *hasInconsistency* -> (Inconsistency)
(CheckList) - *check* -> (Inconsistency)
(Inconsistency) - *inconsistencyPart* -> (MedEvent)
(PatDay) - *hasShownToxicity* -> (PatToxicity)
(PatToxicity) - *hasManifestation* -> (MedEvent)
(MedApp) - *despite* -> (LabReport)

Figure 2: Edge labels. The edge labels are given in italics. The node labels (in brackets) indicate, which node types for instance can be connected with the particular edge. The entire formal description of the valid patient graphs (i.e., *which nodes may be connected by which edges under which circumstances*) is the task of the graph-grammar.

A start graph *S* is a graph built from elements of *N* and *E*, the attributes of all nodes of *S* having defined values.

Graph productions are the central constructs of a graph-grammar and describe the valid graph manipulations on a host graph *G* first being identical with the start graph *S*. A production *P* again is a graph which is divided into four parts, called *Del* (for the *deletion* of nodes and edges), *Gen* (for *generation*), *Det* (for *determined* embedding), and Ind (for *indetermined* embedding) (see Figure 4) [7]:

- *Del* and *Det* together define the *starting region* of *P*. To start *P*, a region of *G* (e.g., of the patient graph) has to be found, where the nodes and edges match the nodes and edges from *Del* and *Det*.
- *Del* then describes the nodes and edges that have to be removed from *G*.
- *Gen* denotes the nodes and edges which have to be generated.
- *Det* together with *Ind* describes the embedding of the new nodes into *G* (determined and indetermined embedding).

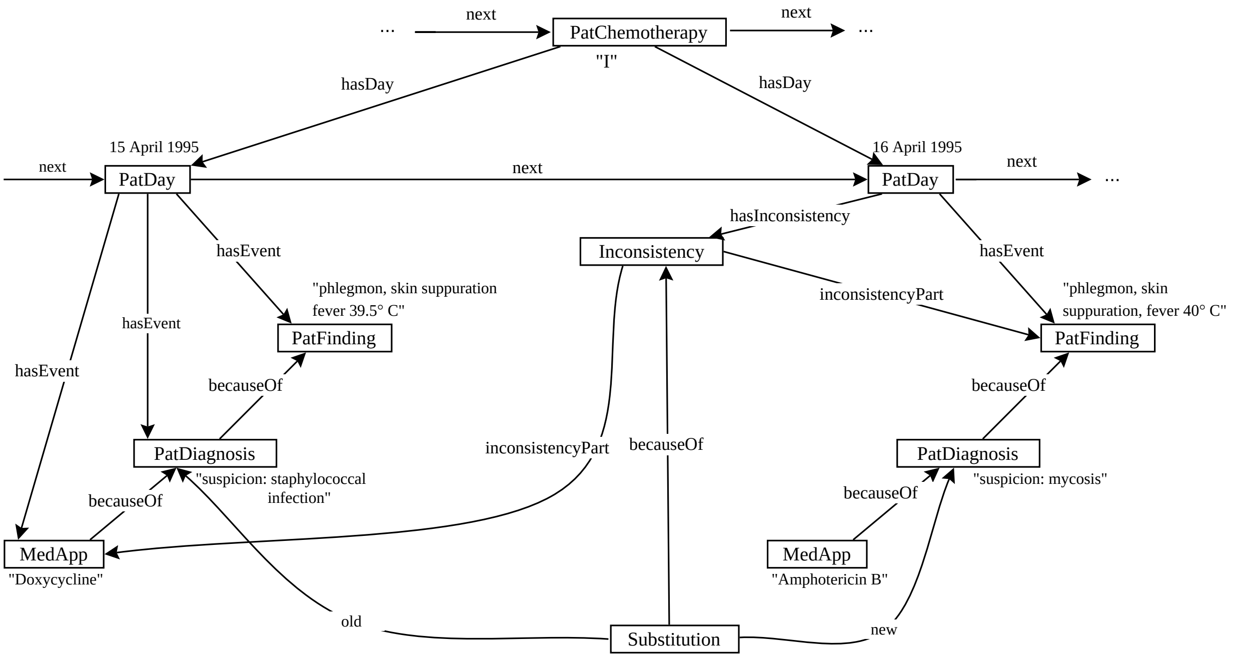


Figure 3: Representation of a revision context. On 14 April 1995 the patient first has been treated with antibiotics because of a strong suspicion of staphylococcal infection (this first day of infection is not shown in the Figure). After two days (16 April 1995) the diagnosis *suspicion of staphylococcal infection* is replaced by the diagnosis *suspicion of fungus infection* represented by the *Substitution*-node. The substitution is mainly based on the observation that the antibiotic drug could not force back the serious infection contrary to expectation. This inconsistency is represented by the *Inconsistency*-node.

The graph production of Figure 4 would be interpreted as follows:

- Determine the existence of a *PatDiagnosis*-node in the patient graph *G*, and an incoming *marks*-edge with a *Cursor*-node as its source node. (*Cursor*-nodes are often used to achieve the unambiguity of a matching region in a host graph.)
- If the structure specified by *Del* and *Det* does not exist, abort the production. Otherwise, continue with the following steps:

  - Remove the *Cursor*-node and the outgoing *marks*-edge.
  - Create an *InfApplication*-node and a *PatHemogram*-node, and connect both with a *monitor*s-edge (in the specified direction). (The attribution of the new nodes is specified in a so-called body-construct which has been omitted here.)
  - Connect the *InfApplication*-node with the *PatDiagnosis*-node through a *becauseOf*-edge (determined embedding).
  - If there is a *PatDay*-node, which is connected to the *PatDiagnosi*s-node via a *hasEvent*-edge, then connect it with the *InfApplication*-node and the *PatHemogram*-node via a *hasEvent*-edge (indetermined embedding).

Figure 5 shows a more complicated graph production mapping an inconsistency context to a patient graph. Because of the ability to specify indetermined embeddings via the production part *Ind*, graph-grammars are a powerful calculus to formally describe contextsensitive graph manipulations. Entity-relationship models, which also specify the valid structure of graphs, don’t provide a formalism to describe how a non-trivial contextsensitive graph manipulation has to be

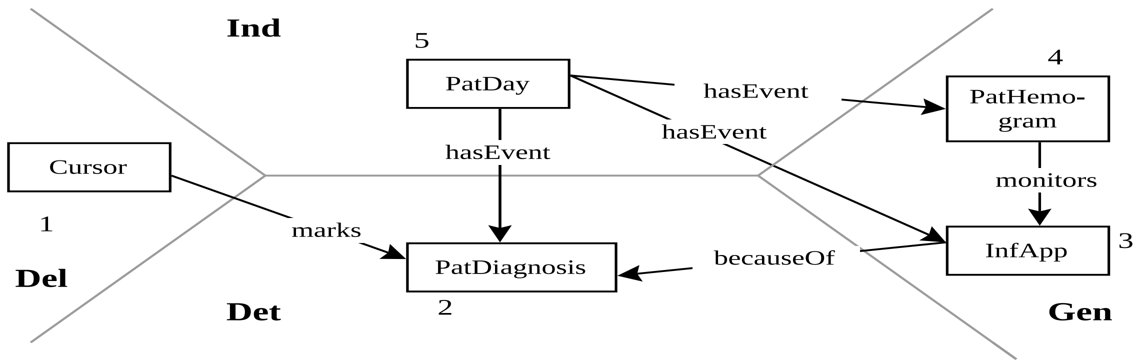


Figure 4: Sample graph production which maps a causal context to the patient graph. This production maps the administration of an infusion and the monitoring hemogram to the patient graph (see text for more details).

carried out in detail. Formally, a patient graph is valid if it can be derived from the start graph *S* by only using productions of the patient graph-grammar. In practice, the validness of the patient graph and especially the referential integrity is enforced by allowing external modules only to call public productions of the grammar.

3. Results

The graph-grammar approach introduced and illustrated above has been used to develop the electronic patient record of the knowledge-based system **TheMPO**. All together, the recent patient graph-grammar consists of 23 generic graph productions which can be parameterized with node and edge labels and attribute-values. To represent the course of treatment of an oncological patient, especially the following functionality is covered by the grammar:

- Adding new *PatDay*-nodes to the patient graph to represent days of in-patient and out-patient treatment.

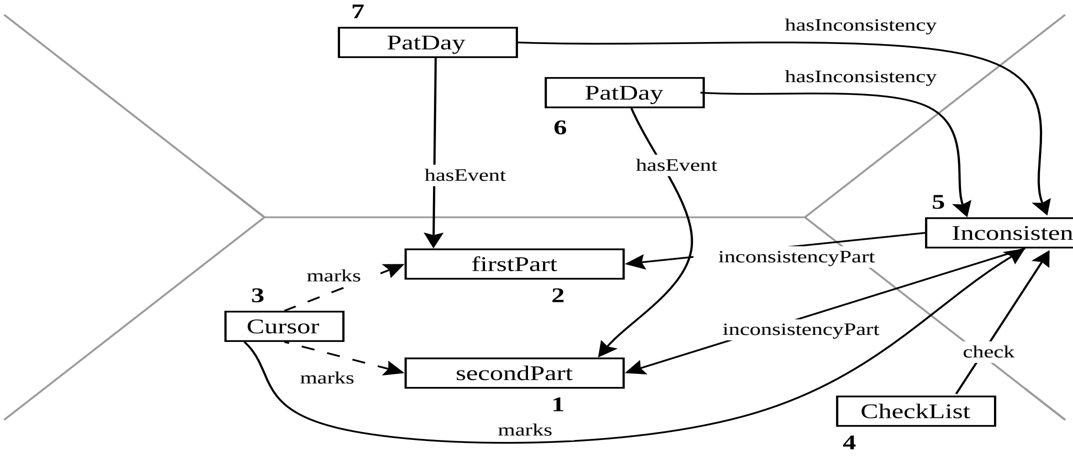
- Adding nodes derived from the node label *MedEvent* to represent diagnostic procedures and there results as well as therapeutic actions. Connecting nodes with *first*-, *last*-, *next*-, *hasDay*-, *hasEvent*-, *hasMandatoryEvent*- and *knownSince*-edges to express temporal relationships between days of treatment, phases of chemotherapy and other medical events.

- Adding *PatChemotherapy*- and *PatTrend*-nodes to represent temporal abstractions such as '*chemotherapy of type ...*' or '*significant trend of laboratory value L*'.

- Connecting nodes derived from *MedEvent* with *becauseOf*-, *discontinuedBecauseOf*-, *reducedBecauseOf*-, *increasedBecauseOf*- and *monitors*-edges to represent causal relationships between diagnostic and therapeutic data.

- Adding *Substitution*-nodes and *new*- and *old*-edges to represent substitution events such as the revision of a diagnose (see Figure 3), or the substitution of a drug *B* for a drug *A*.

```
GraphProd_EstablishInconsistencyContext(   in  firstPart:       NodeLabel,
                                           in  secondPart:   NodeLabel,
                                           in  description:  string,
                                           in time:             Time);
```



Body: 5.inconsistencyDscrpt = description; 5.contextKey = sys_contextKey();
5.eventTime = time;

Figure 5: Sample graph production *GraphProd_EstablishInconsistencyContext*. This production is parameterized with: **(1)** the labels of the conflicting nodes (*firstPart* and *secondPart*), **(2)** a string for additional comments (*description*), and **(3)** the time of detecting the inconsistency (*time*). After having found a matching region in the patient graph, the production generates an *Inconsistency*-node (node 5) and connects it with the two conflicting nodes (1 and 2) via *inconsistencyPart*-edges and the global *CheckList*-node (determined embedding). After that, the indetermined embedding is processed and the new *Inconsistency*-node is connected with the two *PatDay*-nodes 6 and 7 (6 and 7 may be identical if the conflicting nodes 1 and 2 both represent events which have occurred at the same day). The *Body*-construct at the bottom of the production specifies the new *Inconsistency*-node's values: The values of *description* and *time* are assigned to attributes of the *Inconsistency*-node; *contextKey* is an integer for indexing contexts calculated and maintained by the system. The dashed edges connecting the nodes 1, 2 and 3 indicate that these edges will be removed when the graph production has finished.

- Adding *Inconsistency*-nodes and *inconssistencyPart*-edges to represent inconsistencies. Adding *PatHeuristic*-nodes to express patient-specific heuristics.

By making contexts and patient-specific heuristics explicit, the reasoning agent (physician or machine) is supported in the following way:

- The representation of causal and substitution contexts enables the record to quickly answer questions such as *Why has this drug been administered or discontinued?* (e.g., the record can answer that the antibiotic drug has been discontinued because it has not been able to force back the serious infection symptoms, and that it has been replaced by antimycotics). Furthermore, if knowledge about a patient is revised, the inference is able to detect whether there are conclusions based on the revised knowledge that have to be reviewed.

- By making temporal contexts such as *Trend of Laboratory Value L* explicit, temporal reasoning is supported directly, as rules in the oncological knowledge base often do not refer to simple time-stamped data, but to temporal abstractions such as *Trend* or *Phase of Chemotherapy*.

- By representing inconsistency contexts, wrong conclusions can be avoided during the reasoning process. For example, whenever a finding *F* - which is involved in an inconsistency context - is used during reasoning, the record itself can point to a laboratory value which has been regarded as being inconsistent to *F* so far.

- Patient-specific heuristics can be temporarily linked to the knowledge base to provide the reasoning process with patient-adapted knowledge that is not covered by the common knowledge base and which is not valid for other patients.

4. Context Acquisition and Retrieval

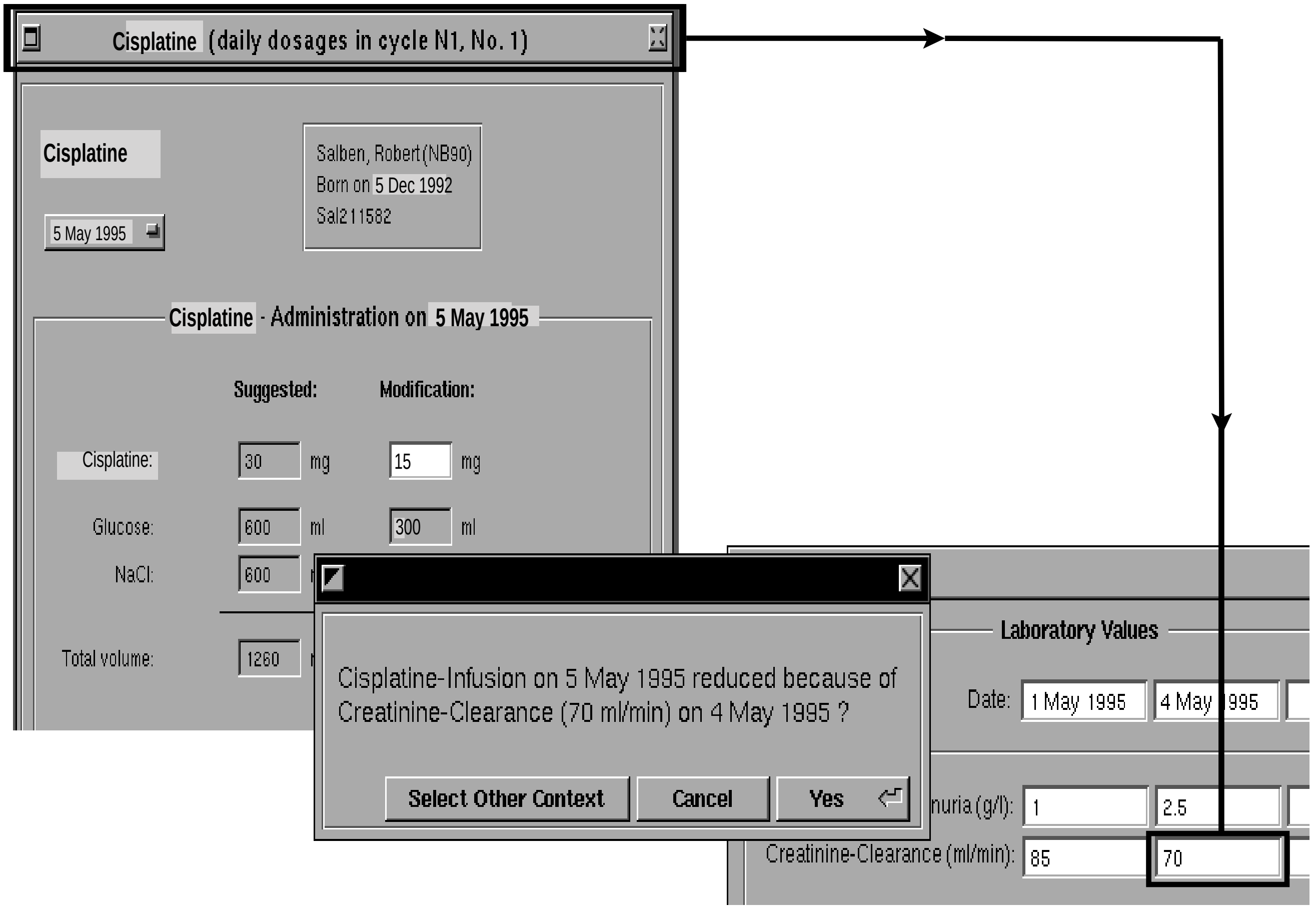
4.1 Context Acquisition

To acquire data contexts, a graphical context acquisition tool has been developed:

If the physician, for instance, wants to inform the record that he has reduced the *Cisplatine*-doses because of a decreased creatinine-clearance, he simply selects the window with the reduced drug doses, and drags from the window to the responsible laboratory value with the mouse. As illustrated in Figure 6, black rectangles then are drawn programmatically around the source and the target of the specified relationship, and a black arrow visualizes the dragging process. The program, which has recognized that the dosages have been reduced (by comparing the two dosage columns in Figure 6), assumes that the physician wants to specify a *reducedBecauseOf*-context because he has connected a drug-window with laboratory data. The program therefore generates a window asking the physician if he has reduced *Cisplatine* because of the selected creatinine-clearance. If the physician confirms, a graph production is called generating a *reducedBecauseOf*-edge between the *Cisplatine*-node with the reduced doses and the creatinine-node. Wrong specifications such as drawing a *reducedBecauseOf*-edge from a *LabReport*-window to an *InfApplication*-window will be rejected by the graph-grammar. In a similar manner, other types of contexts (e.g., temporal contexts, inconsistencies or substitutions) can be specified too.
To specify patient-specific heuristics the physician can use a graphical rule-editor. This rule-editor has been developed for the acquisition of common monitoring and stratification rules in the **TheMPO**-knowledge base, but can also be called from the electronic patient record to define rules adapted to the specific situation of the patient. These patient-specific rules are stored in the record and will be linked to the knowledge base with a high priority each time the inference engine is reasoning about this patient.

4.2 Context Retrieval

The retrieval of context knowledge which has already been mapped to the patient graph is realized with a mouse-based tool as well. If, for instance, the question arises why the dosages of the *Cisplatine*-infusion in Figure 6 are significant low, the user simply selects the *Cisplatine*-window with the mouse in a question mode. After having selected the item *Causal Context* in an appearing pop-up menu (which offers other items such as *Temporal Context* too),


Cisplatine (daily dosages in cycle N1, No. 1)

Cisplatine

5 May 1995

Salben, Robert (NB90)
Born on 5 Dec 1992
Sal211582

Cisplatine - Administration on 5 May 1995

	Suggested:		Modification:	
Cisplatine:	30	mg	15	mg
Glucose:	600	ml	300	ml
NaCl:	600			
Total volume:	1260			

Cisplatine-Infusion on 5 May 1995 reduced because of Creatinine-Clearance (70 ml/min) on 4 May 1995 ?

Laboratory Values

Date:	1 May 1995	4 May 1995
nuria (g/l):	1	2.5
Creatinine-Clearance (ml/min):	85	70

Figure 6: Screen dump of a graphical context specification (fictitious patient). The left column of the *Cisplatine*-window contains the dosages suggested by the **TheMPO**-reasoner when the chemotherapy N1 (consisting of the drugs *Cisplatine*, *Vespeside* and *Etoposide*) has been prescribed for the patient. The right column shows the dosages the physician has modified manually. Here the physician has specified that the *Cisplatine*-infusion has been reduced on 5 May 1995 because of a decreased creatinine-clearance detected on 4 May 1995 (see text for more details).

the program searches for outgoing or incoming causal edges. In this case, it discovers an outgoing *reducedBecauseOf*-edge leading to a creatinine-node, and loads the window showing the creatinine data.

On the other hand, if the creatinine-window was loaded first and the user wants to know if the abnormal clearance-value had any consequences, the program can quickly retrieve that the reduction of the *Cisplatine*-dosages has been one the actions caused by this abnormal value.

5. Implementation Issues

To implement the patient graph-grammar, the object-oriented C++-database *Poet* in the *NeXTSTEP*-environment has been used. To represent nodes and edges the abstract data types *Graph* and *GraphNode* have been implemented, encapsulating elementary graph-functionality (e.g., adding and removing nodes and edges). Graph productions were implemented as *C++*-functions using the low-level transaction-mechanism of *Poet*. To realize the graphical acquisition and retrieval tools described in section 4 the graphical libraries of *NeXTSTEP* have been used.

6. Discussion and Conclusions

This paper focused on oncological clinical contexts and their structured representation, acquisition and retrieval. Data may be related by causal or temporal relationships, they are frequently grouped together according to problems or may express inconsistencies. Diagnostic conclusions or therapeutic actions may be revised. During long-term treatment, a patient may develop individual medical behavior leading to patient-specific heuristics for instance dealing with individual drug response or disposition to infections. By making contexts and patient-specific heuristics explicit, the retrospective understanding of the medical decision process is facilitated, and a useful support for further treatment is given. To represent clinical contexts, labeled, directed, and attributed graphs have been chosen which are formally described and controlled by a graph-grammar. Mouse-based tools enable the physician to specify and retrieve contexts between the patient's data in a graphical manner.

In the current version of the **TheMPO**-record, it is up to the physician to add context or not, because we think that doctors should not be forced to do this. By providing comfortable graphical tools, which allow the specification of relationships between data with a few mouse operations, we believe that physicians can be motivated to add important context knowledge to the record. Prompts appearing whenever 'important' decisions have been made (such as the discontinuation of an obligate cytostatic drug), and asking the physician whether he wants to add causal context information to the record, could be a compromise.

However, if not all contexts are made explicit, in most of the cases reasoning will still be possible, but will take much longer, as contexts first have be to reconstructed. Of course, in complicated cases it may also occur that the unambiguous reconstruction of a context fails. An evaluation has to show to what degree users can be 'forced' to add context, and how the trade-off between context acquisition on the one hand, and efficient reasoning and retrieval on the other hand, can be minimized.

Patient-specific heuristics can be defined through a graphical rule-editor. By storing these patient-specific rules in the patient record the knowledge base can be relieved of managing rules used only in the context of this specific patient, and becoming useless when the patient is discharged from hospital.

Of course, it could be argued that whenever a patient develops a very specific medical behaviour, this alternatively could be stored as a rule in the common knowledge base, so that this knowledge is available for future patients too. However, we think that this approach is inappropriate because of the following reasons:

- Patient-specific heuristics model the way in which physicians are used to think and work in a more natural way. Physicians are not used to translate heuristics, which they have detected for a particular patient, into generalizing notations for a common knowledge base.

- As patient-specific heuristics cover very specific experiences, their translation to common rules available in the knowledge base would imply very complex premises which are difficult to acquire and represent. If specific experience is represented as a patient-specific heuristic, the premise can be held quite simple, as - roughly spoken - the record of this patient itself is the other, (fulfilled) part of the complex premise.

- Heuristics of different patients may have the same premises but different or opposite consequences, so they probably will not fit together in one knowledge base.

We therefore think that a *case-based reasoning* approach is a more appropriate way to make patient-specific heuristics available for future patients, as the reasoner then can detect similar situations, and can suggest the heuristic which has been used for the former patient. Future efforts concerning our patient record will concentrate on this topic.

However, there are some more topics which have to be addressed in the future:

• *Semantic check of contexts*

The patient graph-grammar introduced is able to reject a *discontinuedBecauseOf*-edge drawn from a *LabReport*-node to an *InfApplication*-node, but however, the current grammar allows the same edge to point from a *Cisplatine*-node to a hyperglycemia-node which does not make very much sense, as hyperglycemia is not known as a side-effect of *Cisplatine.* Future efforts therefore will have to concentrate on a more sophisticated check of contexts specified by the context acquisition tool.

• *Granularity of the patient graph*

In the recent approach, a hemogram for instance is mapped to a single *PatHemogram*-node providing attributes such as *white blood cell-count* or *hemoglobin*. If a drug is discontinued because of a serious blood toxicity, the physician will ‘draw’ *discontinuedBecauseOf*-edges between the drug node and the responsible *PatHemogram*-nodes. However, this may lead to an ambiguity because it may not automatically be evident *which* values of the *PatHemogram*-nodes were responsible for the drug discontinuation. In this case, the current graph-grammar would not provide sufficient granularity to unambiguously represent the intended relationship between specific *PatHemogram*-values and the discontinuation of a drug. The patient graph-grammar therefore is currently extended; a *PatHemogram*-node for example is fanned out to a tree, the nodes representing single blood-values which then can be connected with drug nodes via appropriate edges.

7. Acknowledgements

The **TheMPO**-project is partially funded by the *Stiftung Innovation e.V*. of the Ministerium für Wirtschaft und Verkehr des Landes Rheinland-Pfalz.
We further are most grateful to J. Michaelis, H. Göttler, H.-M. Dittrich and the two anonymous reviewers for helpful advice and valuable comments. Our special thank is to Stefan Kindel for converting the figures to the Macintosh format.

8. References

[1] Rector AL, Nowlan W, Kay S. Foundations for an electronic medical record. Meth Inform Med 1991; 30: 179-186.

[2] Dick RS, Steen EB (eds). The Computer-Based Patient Record. An Essential Technology for Health Care. National Academy Press, Washington D.C. 1991.

[3] Rector AL, Nowlan W, Kay S, Goble C, Howkins IJ. A framework for modelling the electronic medical record. Meth Inform Med 1993; 32: 109-119.

[4] Weed L. Knowledge Coupling: New Premises and New Tools for Medical Care and Education. Springer, New York 1991.

[5] Burger E, Musinszki C, Müller R, Nauerth U, Pommerening K, Sergl M, Thews O. Repräsentation therapeutischen Wissens in der Pädiatrischen Onkologie. Proceedings of the 39. Annual Conference of the Deutsche Gesellschaft für Medizinische Informatik, Biometrie und Epidemiologie. MMV Medizin Verlag, München 1994: 110-114.

[6] Müller R, Nauerth U, Pommerening K, Schurig A, Thews O. Software for protocol confirming implementation of therapy studies in pediatric oncology.Annals of Hematology 1993; 67 (Supplement): A85 (332).

[7] Göttler H. Graphgrammatiken in der Software-Technik. Theorie und Anwendungen. Springer, Berlin 1988.

[8] Egar J, Musen MA. Graph-grammar productions for the modeling of medical dilemmas. Proceedings Sixteenth Annual Symposium on Computer Application in Medical Care. McGraw-Hill, Inc, New York 1992: 349-353.

[9] Egar J, Puerta A, Musen MA. Graph-grammar assistance for modeling of decisions. Technical report. Knowledge Systems Laboratory, Stanford University 1992.