Managing Episodes, Cases, Concepts, and Rules — an Integration Approach for Medical Problem Solving

Wolfgang Oertel and Uwe Petersohn
Technical University of Dresden
Department of Artificial Intelligence
D-01062 Dresden
Germany
Email: oertel,peterson@freia.inf.tu-dresden.de

Abstract

The paper presents the knowledge organization and problem solving in MEDIUS, a medical information and support system developed for application in the domain of pain diseases. The system uses information that is permanently collected during the treatment of patients as well as generic medical background knowledge. It has been installed in a ward of a hospital and is in really practical use. The paper shows the application of case-based reasoning in combination with concept- and rule-based methods, their mapping to a data base system, as well as its integration within a network environment. The results can be generalized easily to medical treatment or problem solving at all.

Keywords: episode, case, concept, rule, reasoning, integration, data base, system architecture, medicine

Introduction

The treatment of acute and chronic pains is one of the most challenging problems in medical research and practice today (Zenz and Jurna 1993). The reasons for that are as follows.

- Pain treatment is a long-time task. That means great organizational and software-technological efforts are necessary to collect, store, distribute, and provide large amounts of data for each patient.
- Pain treatment is a very complex and heterogeneous task. Almost all diseases are accompanied by more or less strong pains. So, knowledge of all branches of medicine is necessary for an adequate treatment.
- Pain treatment is finally a psychological and neural task. But, the psychological and neural behavior of human beings is not yet well understood so that satisfying comprehensive models are not available.

It suggests itself to use computer science to give doctors working in this field any possible support. The base is a general hospital information system storing necessary data in a commercial data base distributed over a network of hospital awards an doctor's practices. A respective structure was proposed for instance in (Franczyk 1996).

The hospital information system has to be combined with a knowledge-based system which provides the medical background knowledge and respective interpreters for problem solving. But, pure concept- and rule-based methods are not enough here. Some kind of case-based technology is demanded. In (Puppe and Schewe 1997), such an approach is applied to the treatment of neural diseases.

The following facts show why case-based reasoning plays an integrative role as for computer support in the field of medical treatment.

- Case-based reasoning uses data for problem solving collected during the normal process of treating patients. So, it can be regarded as an extension of traditional hospital information systems.
- As for the existing lack of generic knowledge, case-based methods can provide first results already without or in combination with generic knowledge.
- Case-based methods can handle the problem of complexity and heterogeneity of medical treatment in a relatively simple and unique way without knowledge about particular fields of medicine.
- A typical medical treatment starts with several kinds of anamnesis and test result delivering very large amounts of data. Case-based reasoning can use these data to focus the further problem solving quickly.
- Finally, case-based methods can initiate the process of deriving missing generic knowledge by building similarity classes or statistics.

The system MEDIUS (Oertel and Petersohn 1998), tries to find solutions for a knowledge-based handling of pain diseases[1]. The proposed way is, at first, to accompany the whole medical treatment process by storing episodic data in a data base and, at second, to use case-based technologies in combination with concept- and rule-based methods for problem solving.

For this approach, we use the own extensive technical experiences in the field of integration that

[1]This research was supported by the Ministry of Economy, Middle Class, and Technology of the German Federal State of Brandenburg.

have been made during the development of hybrid knowledge-based, especially case-based, systems in several domains (Bakhtari, Bartsch-Spörl, and Oertel 1996), (Oertel and Petersohn 1996) and that have been generalized to a set of generic tools (Oertel 1994), (Oertel 1996).

Medical Treatment Cycle

Let's have a look at a typical process that occurs when a patient with pains enters a doctor's practice or a ward of a hospital (figure 1). The treatment process has a cyclic structure defined by relations on a set of treatment steps.

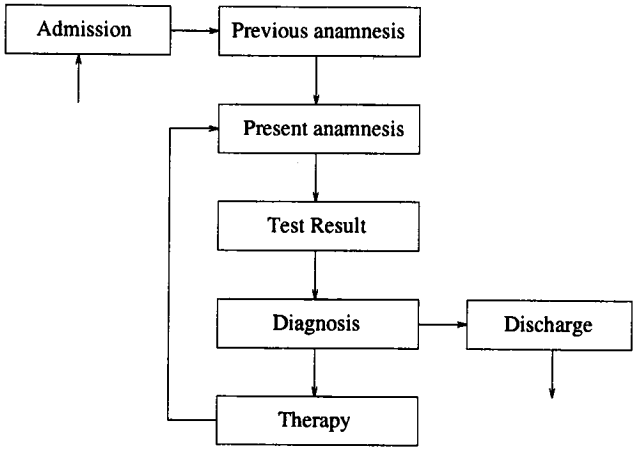

Figure 1: Medical treatment cycle

After the admission, the patient has to answer a lot of questions concerning his own history (previous anamnesis). Then he has to describe his actual state (present anamnesis). Based on the so got information, the doctor decides what tests have to be made to get certain test results. Next, the test results are used to make a diagnosis. At last, a reasonable therapy is undergone and the cycle starts with the present anamnesis again. If the diagnosis is positive or satisfying special criteria, it will come to the discharge of the patient. Sometimes, of course, single steps can be omitted so that the transitive closure is important, too.

This on the first view simple structure is essential for the static structure of the knowledge as well as the dynamic structure of its processing within the developed system.

Knowledge Organization

The knowledge necessary for supporting the medical treatment process is organized along the treatment steps and their relations in the described cycle. Each pair of directly related nodes, that means each arc, in the graph of figure 1 stands for a medical decision. And, such a decision can and must be supported by certain knowledge chunks.

The knowledge exists on different levels. A specific level contains knowledge of concrete patients. Here, a treatment step is represented as an episode. A relation of the treatment cycle, that means a pair of episodes, is called a case. It contains a problem and a solution episode. A typical episode is a filled questionnaire as a present anamnesis or a protocol of a concrete undergone therapy. A key composed of the medical personnel, the patient, and the date identifies an episode. An episode itself has a complex structure with different parameter values including texts, graphics, or pictures. An example of a case is a diagnosis directly connected with a therapy, but also a previous anamnesis together with a later therapy.

On a generic level, we find concepts that are generalizations of episodes. They contain generic descriptions in the form of logical predicates or production rules. Examples are the classification hierarchy of possible diagnoses or evaluation results of anamnesis questionnaires. Concepts are used to define several semantic views on episodes. The generic opposite of cases are rules. A generic rule is defined as connection between two concepts. If it is applied, it transforms an episode of a certain problem concept into an episode of a certain solution concept. Typical rules describe for instance the medical knowledge for deriving a diagnosis from given test results.

The dependencies between the medical treatment cycle and the knowledge elements can be summarized as follows:

- Treatment steps:
  - Specific level: Episodes
  - Generic level: Concepts
- Relations:
  - Specific level: Cases
  - Generic level: Rules

Using the relations partialization (P) and specialization (S), figure 2 shows the structure of the knowledge base on an abstract level.

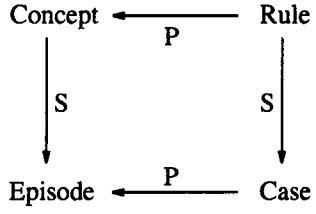

Figure 2: Knowledge elements

The central knowledge element within a knowledge base is the episode. All other knowledge elements like cases, concepts, or rules are defined directly or indirectly on episode structures for the purpose of problem solving. But, episodes don't exist only as single

knowledge elements. They are nodes connected explicitly or implicitly by different types of arcs in a graph. Such arcs stand for spatial relations (X), temporal relations (T), content-oriented relations (V) or similarities (H). An implicitly defined relation is computed by procedures depending on certain parameter values. For instance, a similarity measure is an implicitly defined arc H.

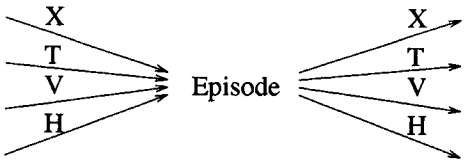

Figure 3: Episode graph

The subgraph of figure 3 established on the specific level of a knowledge base by these arcs is fundamental for information retrieval as well as problem solving in the system.

Problem Solving

Using the knowledge structure introduced in the previous section, a doctor is able to describe the whole treatment process of a patient by filling manually the respective substructures. For instance, he can give in anamnesis data, data of test results, diagnosis data, or he can protocol a therapy over a certain time.

Each relation between episodes can be used for retrieving episodes, manually. If once a new episode for a given episode has been found, immediately a node within a more or less complete treatment cycle structure is identified. From there, arcs can be used to look around — for instance to follow the way to any former or later episode. And such episodes can be considered as proposals to determine further treatment steps.

The automatic problem solving supported by the system has two sides. On the one hand, the assessment of the consistency of a stored episode can be done. And, on the other hand, the next episode for a given actual episode can be computed. So, analysis as well as synthesis tasks are supported.

The analysis of an episode is done by applying a respective concept. Its result can be interpreted as a new view on the basic episode which describes the given episode on a different level of specialization. In principle, analysis can also be done case-based by simply searching for the concept of an already analyzed similar episode.

The synthesis of new episodes for given episodes is done using either specific knowledge (cases) or generic knowledge (rules). For the case-based approach, the following steps are carried out (compare to figure 4):

1. define a problem episode **EP0** of a certain concept,
2. define an empty solution episode **ES0** of a certain concept,
3. define a relation **V,T,X** between **EP0** and **ES0**,
4. search for a similar problem episode **EP1** using **H**,
5. determine a solution episode **ES1** of a respective concept following **V,T,X** from **EP1** as part of a case **C1**,
6. fill the solution episode **ES0** by merging **EP0** and **ES1**,
7. return the solution episode **ES0**.

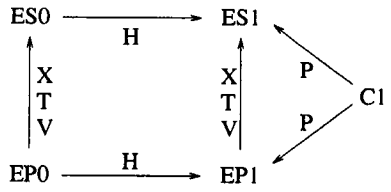

Figure 4: Case-based reasoning

This case-based reasoning can be performed syntactically on the specific level using raw episodes, simple structure-oriented similarity measures testing for equality or distances, and simple merging as adaptation. But, it can also be performed on higher, more generic levels by using concepts as pre-processing. In this way, any kind of semantical, problem-specific similarity retrieval or adaptation can be done.

Alternatively, rule-based reasoning can be performed according to the following steps (compare to figure 5). The steps 1, 2, 3, and 7 can be taken from the case-based approach:

4. search for a generic problem concept **CP0** using **S** that comprises **EP0**,
5. determine a solution concept **CS0** following **V,T,X** from **CP0** as part of a rule **R0**,
6. fill the solution episode **ES0** by specializing **CS0**.

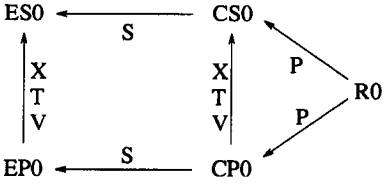

Figure 5: Rule-based reasoning

As reasoning result, the generated solution episode is given to the user as a consistent or inconsistent proposal. After an additional manual consistency check and correction, the new episode can be stored in the knowledge structure and connected there with other already existing episodes.

Because of the simple and unified structure of all knowledge elements, different problem-solving paradigms can be combined with each other for solving a concrete problem. So, problem solving can be done manually by the doctor or automatically case-based or rule-based by the system (using more or less specific

concepts), or in a combined manner. The concrete interplay between the different methods is organized by a set of alternatively applicable control structures called behaviors.

System Architecture

In the previous sections, the conceptual structure and behavior of the system MEDIUS was described. But, a good conceptual structure alone is not enough for a really working case-based support system. Other aspects must be considered, too. So, let's have a look at the software- and hardware-technical organization — the system architecture.

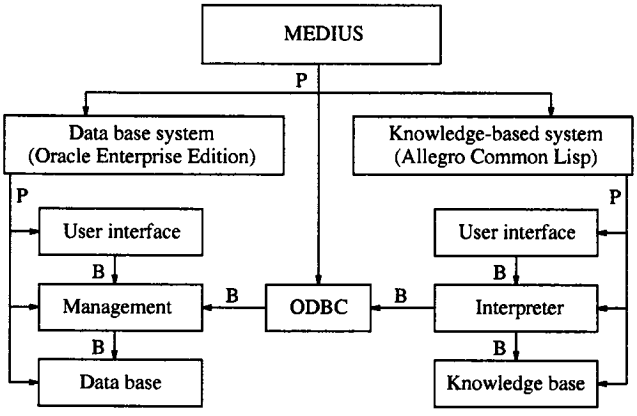

Figure 6: System architecture

The system MEDIUS is divided (P) globally into two parts, the data base system and the knowledge-based system (figure 6). The reason for this partialization is that on the one hand huge amounts of case data have to be stored and on the other hand a flexible intelligent behavior of problem solving is demanded. That's why, the proposed architecture is essential for the whole approach from a practical point of view.

The data base system stores all necessary data, especially episodes and cases. Furthermore, it performs structure-oriented query operations and supports a pre-selection of episodes and cases before the application of the final similarity measures. For that, each episode structure of the knowledge-based system is mapped to a table structure of the data base. Different tables exist for different step-specific episodes. Queries and Updates are described by the language SQL. SQL operations are also used to provide the necessary input data for each problem solving process of the knowledge-based system or to store back its results.

In the knowledge-based system, all knowledge-intensive processes are organized. Different case-based and rule-based interpreters are available. The knowledge base is initially hold within the knowledge-based system itself. If the amounts of knowledge become large, they can be stored in the data base system, too. For the whole knowledge organization and problem solving, interpreters and tools of the development system FAENSY (Oertel 1994) are used: an object system, a logic clause interpreter, a production rule interpreter, and an analogy function interpreter.

As shown in figure 7, both main subsystems have their own user interfaces to prevent loosing the advantages of each subsystem. The main control flow from the user interfaces to the other components of each subsystem is noted by the arcs (B) standing for activations with return. The activation between the knowledge-based and the data base system is done by the ODBC interface as a third component. Using additional predefined tables, views, and indexes, the data base system speeds up the problem solving processes of the knowledge-based system.

The system runs on PCs with Windows NT / 95. It is realized by the Oracle Enterprise Edition 8.0 and the Allegro Common Lisp System 3.0. The control is done manually by the user, programmed by function calls, or automatically by intelligent agents.

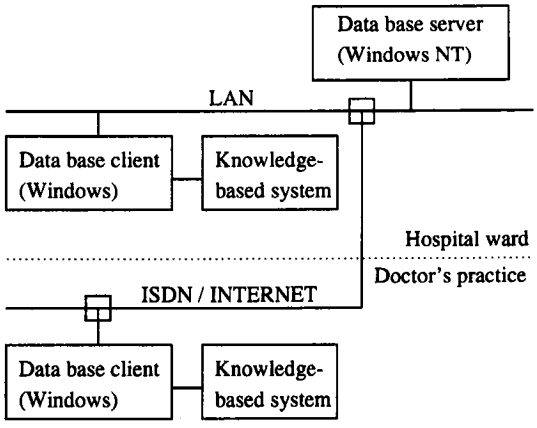

Figure 7: Network structure

Furthermore, the system architecture must be seen within a more global context. The permanent input of episodes, their actualization, and their use for problem solving are only possible if several doctors and nurses (may be also patients) have access to the system at different places at the same time. So, a network structure as shown in figure 7 is demanded.

A central sever and several clients distributed over a spatial region are connected by a network. Local area networks LAN, the INTERNET, as well as ISDN can be used. All network tasks are controlled by the data base system. The clients have their own knowledge-based system that interacts with the associated data base client or a single user data base system.

Finally, the system architecture has to provide solutions for some tasks that are essential for case-based systems, too. It must:

- allow a distributed storing of case data,
- guarantee the security and integrity of the case data

in every situation,

- define different views and rights for different users,
- guarantee an efficient retrieval and problem solving behavior.

Current State and Future Work

The presented system MEDIUS exists as a second prototype. So, it is natural that not everything has already been realized. A lot of work will have to be done in the future. The actual version contains:

- the whole system architecture including the network structure,
- the data base system with input, update, and retrieval components,
- the structure of the knowledge-based system including the interpreters,
- all data structures for anamnesis, test result, diagnosis, and therapy,
- general applicable syntactic similarity functions,
- several problem-specific semantic similarity functions,
- some proven medical algorithms to evaluate anamnesis questionnaires, and
- problem-specific concept and rule knowledge according to a reference book (Zenz 1995).

The doctor gets forms on screen and paper to feed all treatment data (more than thousand items per patient) into the system. He also can edit and retrieve them again. The input data are transformed into episodes after some consistency checks have been done by the system. The episodes are immediately available for similarity-based retrieval and application of available generic rules in order to prepare further decisions of the doctor.

The system has been installed in a ward of a hospital in December 1997 for the first time to test the principle behavior. The practical use started there in February 1998 with about two or three new patients admitted per day.

Actual problems we have to contend with are:

- still often changing medical parameters of the episodes and the relative inflexibility of the data base system,
- the integration of the system in the organizational process of patient's treatment in medical institutions,
- the small amount of time doctors have to formalize their generic knowledge, and
- some implementational errors concerning the interactions between Oracle, Lisp, and the user interfaces.

The first reactions of the doctors are as follows. The components for managing all data of patient's treatment are useful because they provide quickly certain needed information. Of course, using the system means additional work of permanent data input and update. The implemented generic medical algorithms are permanently applied because they deliver usual results. Till now, this computing had to be done manually.

Furthermore, the realized similarity functions are understood as addition to a traditional information retrieval. The problem solutions provided by the case-based and rule-based subsystems are regarded as proposals or hints for further treatment steps. Mostly, they must be put in concrete terms or adapted to the actual situation by the doctors themselves. What is missed by the doctors – and not yet realized in the system – is some kind of learning behavior.

Hence, it can be said, the system is principally accepted, first. But, the further development will depend on the facts how fast it will be possible to involve further problem-oriented medical background knowledge and how easy it will be to update this knowledge and keep it currently over the time.

From a technical point of view, the system has reached a state that would not have been possible using a single problem solving or knowledge management method alone. The integration is of the essence of a practically working case-based system.

Till the end of the project, it is planned:

- to update the basic data structures at certain times to guarantee their relevance to the present,
- to involve larger amounts of problem-specific medical background knowledge,
- to use more intensively the advantages of the data base system for better integrity, security, and performance, and
- to involve learning processes for partial deriving concepts and rules automatically.

References

Bakhtari, S., Bartsch-Spörl, B., and Oertel, W. 1996. DOM-ARCADE: Assistance services for construction, evaluation, and adaptation of design layouts. In Gero, J. S. and Sudweeks, F. *AI in Design '96*, 681–699. Dordrecht: Kluwer Academic Publishers

Franczyk, B. 1996. Adaptive Expertensystemstruktur. In Statusseminar Künstliche Intelligenz, Neuroinformatik und Intelligente Systeme. Berlin: DLR

Oertel, W. 1994. FAENSY: Fabel Development System. FABEL Report 27. GMD Sankt Augustin

Oertel, W. 1996. Knowledge Organization Using the Development System FAENSY. In Görz, G.; Hölldobler, S. *KI-96: Advances in Artificial Intelligence* 303–306. Berlin: Springer-Verlag

Oertel, W. and Petersohn, U. 1996. Hybrid Knowledge Organization within an Object Framework. In

Proceedings of the Workshop on Knowledge Representation and Configuration, 33–41. DFKI Saarbrücken

Oertel, W. and Petersohn, U. 1998. A Case-Based Support System for Treatment of Pain Diseases. In Sixth German Workshop on Case-Based Reasoning, 149–158. IMIB Series 7. Universität Rostock

Puppe, F. and Schewe, S. 1997. Mehrfachverwendung von diagnostischen Wissensbasen in der Medizin. In *KI* 3/1997: 15–23, Bad Ems: ScienTec Publishing

Zenz, M. and Jurna, I. 1993. *Lehrbuch der Schmerztherapie.* Stuttgart: Wissenschaftliche Verlagsgesellschaft

Zenz, M. 1995. *Handbuch der Schmerztherapie.* Stuttgart: Wissenschaftliche Verlagsgesellschaft

Appendix

1. **Integration name/category**: MEDIUS
2. **Performance Task**: Medical decision support
3. **Integration Objective**: Improving efficiency, solution quality, and solution extent
4. **Reasoning Components**: Case-based, concept-based, and rule-based reasoning
5. **Control Architecture**: Several behavior structures (manual, sequential, or parallel)
6. **CBR Cycle Step(s) Supported**: Retrieval, simple reuse, revision, simple retention
7. **Representations**: Data base tables for episodic knowledge; objects, logical predicates, and production rules for generic knowledge
8. **Additional Components**: Data base management system, user interaction, network
9. **Integration Status**: Applied
10. **Priority future work**: Application, evaluation, learning