A Database Design Pattern for Structuring Hierarchical Medical Data

Oláh P[1], Măruşteri M[1], Muji M[2], Bacârea V[1], Haifa B[1], Petrişor M[1], Dobru Daniela[1]

[1] University of Medicine and Pharmacy, Tîrgu Mureş, Romania
[2] Petru Maior University, Tîrgu Mureş, Romania

Introduction: Healthcare Information Systems (HIS) are in most cases complex systems that store and manage large amounts of medical data. When such a HIS is intended to be used in medical research, it presents the system designer with a double challenge: the need for complexity and flexibility at the same time. In this paper we present a database design pattern along with a Graphical User Interface (GUI) design that empowers the researcher to build metadata structures, which are supported by relational data structures on a database server.
Method: As most of the data gathered during a medical research is viewed as having a hierarchical structure, we based our approach on modeling trees and multitrees. The system that we have developed is based on database design patterns and GUI prototypes. Issues regarding data structuring, data entry and data retrieval are addressed.
Results and evaluation: The structure and functionality of the proposed system are presented, with emphasis on three major fnctions: data structuring, data entry and data retrieval. Some considerations regarding the implementation of the system are also included in this chapter.
Conclusions: By using the presented approach, medical researchers can quickly and efficiently configure a customizable software system for recording their data in more complex structures than tables or spreadsheets, while benefiting of the consistency of a relational database.

Keywords: medical databases, healthcare information systems, design pattern, user interface

Received: 17 April 2012

Introduction

Healthcare Information Systems (HIS) are in most cases complex systems that store and manage large amounts of medical data. When such a HIS is intended to be used in medical research, it presents the system designer with a double challenge: the need for complexity and flexibility at the same time. The complexity derives from the large and highly interconnected body of knowledge developed by medical science, whereas flexibility is a specific requirement of the research process. To meet this challenge we set our goal not just to build a system according to the initial requirements of the research team and then expand it as new requirements are formulated, but rather to offer the researchers a framework in which they can design their own data structures, within some limitations, of course.

Classical electronic medical record (EMR) systems can partly address the limitations of paper-based documentation, but the uniformity of the data required in a patient-oriented clinical research can only be achieved by using a structured data entry (SDE) system [1]. Usually these systems embed the structure of the processed data in the structure of the underlying database itself. We believe that it is much more useful if the researchers themselves, the end-users of the system, have the possibility to design their own data structures as they make progress in their research.

Most of the data gathered during a medical research is viewed as having a hierarchical structure. Elementary information such as measurement results or direct observations are grouped in categories, which in turn are grouped in more general categories or domains. These hierarchical structures are best represented as trees. Sometimes the same data is relevant to two or more different categories of information. This introduces a special requirement that two or more trees have a common node (usually a leaf node but not necessarily). This type of information structuring can be represented at the data level as multitrees, a special form of directed acyclical graphs [2].

In this paper we present a database design pattern along with a Graphical User Interface (GUI) design that empowers the researcher to build metadata structures, which are supported by relational data structures on the database server. A proposal for the implementation of these structures in a relational database is also presented.

Method

The database design patterns and GUI prototypes that we present in this paper originate in the effort of providing adequate software for the research team that conducted a study in the field of gastroenterology. The data acquiring and processing requirements of the project were fairly complex and had an evolution during the life-cycle of the project, with new requirements being added as the research team progressed with their work. With the aim of developing a system that can adapt to these changing requirements without the constant intervention of the software development team, we have designed a software solution based on database design patterns and GUI prototypes.

We used a client-server system architecture, with a relational database engine on the server side. The client module was built in a software development environment which features an object-oriented programing language.

Correspondence to: Péter Oláh
Email: olah_peter@yahoo.com

Results and evaluation

In order to accommodate the requirements of the medical research team we have chosen to model the data using trees and multitrees. The majority of the medical data processed in a study can be structured as a tree, with each node representing a piece of information about the patient. The parent nodes are usually categories or general information whereas the "leaf" nodes, situated at the bottom of the hierarchy, represent values of measurements or results of direct observation. Each patient will have his own data structures, automatically generated, so the researcher only needs to update these structures with the relevant data.

We propose a database design pattern that models the medical data as a collection of trees, grouped in what we call "structure types". In this case each type of observation chart used in the study is represented by a "structure type". The advantage of this approach is that each piece of information can be accessed via its parent node but also via the structure type that it belongs to. So, for instance, we can have a report with all the measurements of a variable stored within a specific observation protocol, or observation sheet. Furthermore, the same node can be included in one or more structure types, allowing the navigation from one tree to another (multitrees). For instance if the researcher needs the same information to be included in two different observation sheets, the data must not be entered twice, but rather the same data will be embedded in the structure of both sheets. For the convenience of the user we have named this type of double belonging a "link", or a "shortcut".

The main benefit of this approach is represented by the consistency of the data while having the benefits of structural flexibility. The user can extend the structure of the registered data at any time, in any point of the structure. The user can also eliminate parts of the structure, and the system, after proper warning, will erase all the existing data that was stored in that structure. Remodeling the structure with existing data can also be done by moving an entire subtree from one position to another in the structure.

The GUI of such a system must provide three functions to the user: data structuring, data entry and data retrieval.

Data structuring

The researcher can build his own observation sheets by gradually building a tree, using a top-down approach. Concepts like "domain", "category", "subcategory", "elementary data" will be used to model the transition from general to particular as the build progresses. If the same elementary data or category is needed in two or more observation sheets, the researcher can define the node in one of the structures, and then use a function called "link copy", followed by a "link paste" in the second structure, resulting the insertion of the same node in two different trees.

A particularly challenging issue proved to be the type of the desired data at the lowest level of the tree ("leaf" or elementary data). We have identified the need for the following types:

- text – is entered directly, with some limitation to its size, derived from a limitation of the underlying database engine
- numbers – usually with two decimals
- date – as a calendar date and time stamp
- images – our approach was to record the path to an image already saved in a folder on the same disk on which the database is hosted; embedding the graphical data in the database is also possible but some performance issues must be addressed in this case
- predefined list – in this case the interface provides a special screen to define the possible values of the list. This is a particularly important type of data as it offers a rigorous structuring of the information with the possibility of processing or retrieval later on.

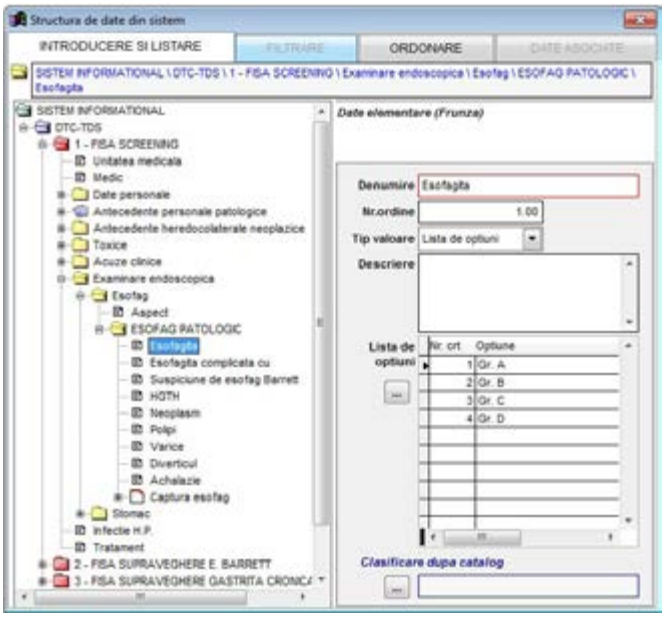

Fig. 1. Data structuring

In our approach we have limited the data types to these five in order to be able to model them in the database as columns of a table, for the first four, and respectively as a separate table for the fifth. However, in the case of the predefined lists, the number of possible values is not limited, providing the user with a great degree of flexibility and still preserving the advantages of having structured data.

Data entry

After defining the overall structure of the needed information, the researcher can collect the data for each patient using this structure. The system will automatically replicate the structure of the needed observation sheets for each new patient. This way a consistent pattern can be enforced throughout the study regardless of the operators who collect the data.

After the observation sheet has been generated, the researcher can easily navigate its structure in order to record

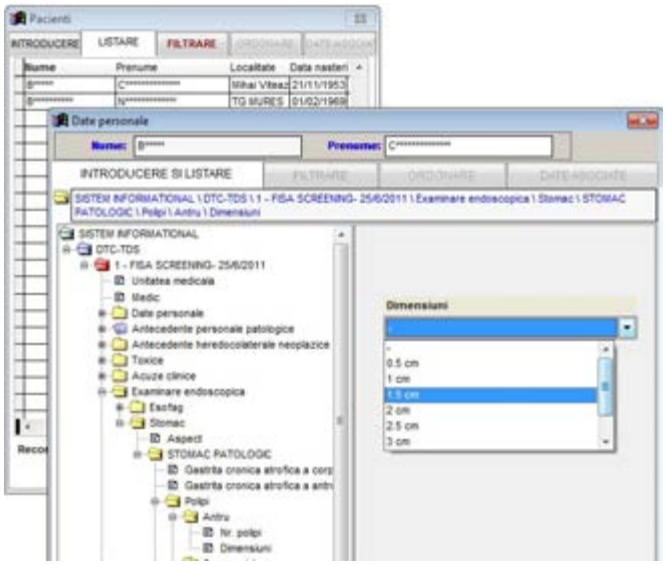

Fig. 2. Data entry

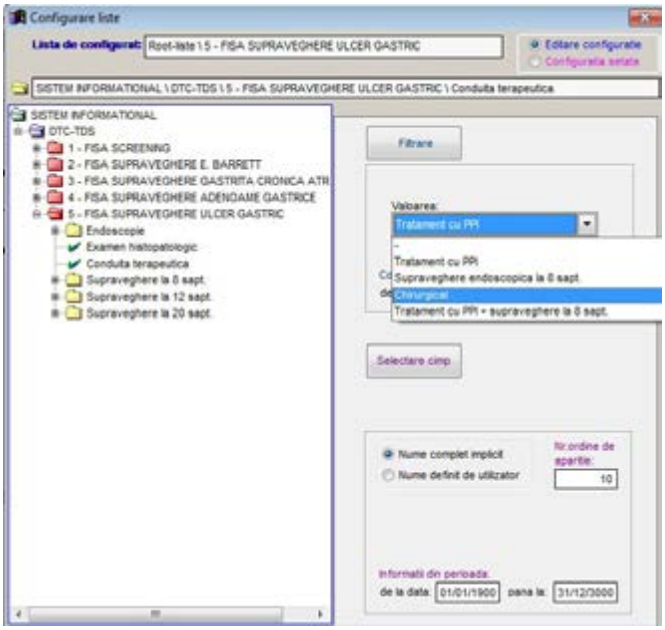

Fig. 3. Data retrieval

the necessary data. Depending on the data type of each node, the interface provides various functionalities. Text, number and dates are recorded directly into corresponding interface controls. In the case of predefined lists the user can only select one of the available options presented in a drop-down list. For images we have implemented a simple interface by opening a dialog box for the user to browse the disk to the desired image. This requires the image to be already captured and saved on the hard-drive. If the research design requires it, it is possible to link the data entry interface to an image capturing equipment for more efficient collection of the data.

If the case may be some comments can also be recorded for each piece of recorded data.

Data retrieval

As with the data structuring, the requirements for retrieving the stored information in order to be analyzed are often complex and not always fully known at the beginning of a medical research project. As we found out, a set of standard reports proved to be insufficient, so the system had to be complemented with a more sophisticated tool. Our approach was to develop a design pattern that preserves the same freedom of the researcher in querying the data as in structuring it.

To accomplish this we have devised a GUI that allows the user to define as many queries/reports as needed, grouped in custom categories for organizing purposes. For each new query the researcher can navigate the previously defined custom data structure and for each "leaf" node, or elementary information, can decide whether to include it in the report or not. In addition, for each included node, the user can specify some filtering parameters according to the data type of the node (text, number, date or list). After configuring the desired report the researcher can preview the data, print it or export it to other programs for statistical analysis.

Implementation of the system

The multitree structure is implemented using two tables, one to host the complete catalogue of nodes (CTH_TreeNode) and one to host the links between the nodes (CTH_TreeNode_Str), the actual structure of the multitree. The structure table has three foreign keys from the node catalogue table, one representing the node, one for the parent of each node and one for the structure type of each node. In addition each node has a node type that can be leaf, category or structure type.

A "structure type" is a special type of node that models in this project each type of observation sheet, determining the boundaries of each tree in the multitree structure.

The actual medical data for each patient is stored in a separate table (CTH_DatePers) that mirrors this structure and hold the values for each instance of the observation sheets. These are the actual "records" of the system from the user's point of view. For the nodes defined by the user as having the data type "predefined list" the values available in that list are stored in a separate table (CTH_TreeNode_TipVal_Lista) and can be recorded in the actual observation sheets using the foreign key migrated from this table into the "records" table.

Discussions

Healthcare Information Systems used in medical research require the capability of representing complex data structures and the flexibility to adjust these structures as the research process needs it. A data oriented [3] approach in our opinion is beneficial to build such a system by capturing the complexity of the studied domain rather in the structure of the data than in complex procedures.

By allowing the researcher to define a metastructure of the data, transparently translated into relational database structures by the system, we have provided a fairly large

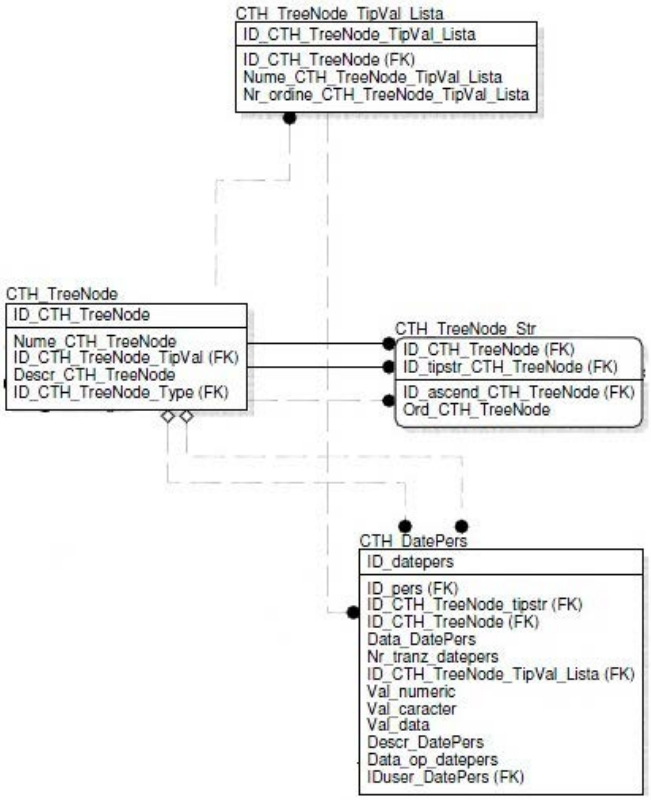

Fig. 4. Database design

amount of freedom to the user in the process of modeling through data the object of the medical research. One possible downside to this approach is the need for the researcher to shift perspective. A certain amount of training is needed before the medical researcher will get familiarized with this approach and will be able to structure the data in ways that will easily support later processing. However, an iterative process of defining a custom structure, followed by some data acquisition and then by some adjustments to the original definition proved to be useful in this regard.

Regarding the limitations of this approach we have identified a few issues. The process of data entry is not optimized for speed. The majority of the information is entered using mouse clicks, so although the system tackles complexity well it is not suitable for very large volumes of data. This model does not offer any functionality for processing the data, it merely creates a framework for structuring, storing and retrieving it. However the retrieved data can be exported to other applications in order to be further processed. There are some boundaries to structuring the data as well. The user has total freedom within the framework, but if new data types are needed, in addition to the five types presented above, some further software development is necessary in order to extend the GUI to accommodate them. The GUI features allow for efficient view traversibility [4] using the tree-view control on the left side. The right side however is developed as a collection of objects suited for the different data types that the system uses. So, if new data types are added, new interface controls must be developed in order to manage these new kinds of nodes.

Other approaches, like OpenEHR or EHR4CR, are focused on building a semantic electronic health record technology [5] that is primarily designed to improve the efficiency of conducting clinical trials [6]. The above presented approach is more general, thus lacking the specialization for clinical data. However, the archetypes used in OpenEHR can be modeled as tree types or high level tree nodes. As our approach is intended to offer the maximum amount of freedom to the researcher in structuring the data, with the express aim to also be applied in fundamental and experimental research, not just in clinical research.

The use of relational database design patterns and GUI prototype modules allows for the system to be generalized and deployed in any field that supports tree-like data structures, for instance in implementing medical ontologies.

Conclusions

By using the above presented approach, medical researchers can quickly and efficiently configure a customizable software system for recording their data. This can be done using minimal interaction with the software developers, as the main challenge in this process resides in properly structuring the research data using a multitree based metastructure. These structures allow more complex representations than two dimensional tables or spreadsheets but have a negative impact on data recording speed. The use of a relational database server as the back-end of the system ensures the consistency of the research data.

Acknowledgment

This paper is partly supported by the Sectorial Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and by the Romanian Government under the contract number POSDRU 64331.

References

1. Bleeker SE, Derksen-Lubsen G, van Ginneken AM, van der Lei J, Mol HA. Structured data entry for narrative data in a broad specialty: patient history and physical examination in pediatrics. BMC Medical Informatics and Decision Making. 2006;6:29
2. Furnas G W, Zacks J. Multitrees: Enriching and Reusing Hierarchical Structure. Proceedings of the ACM CHI 94 Human Factors in Computing Systems Conference. 1994;330–336
3. Lewis B. Data-Oriented Application Engineering: An Idea Whose Time Has Returned. The Data Administration Newsletter - TDAN.com. January 1, 2007; last accessed on April 1, 2012 at http://www.tdan.com/view-articles/4582;
4. Furnas GW. Effective view navigation. Proceedings of the ACM CHI 97 Human Factors in Computing Systems Conference 1997
5. Anani N, Chen R, Prazeres Moreira T, Koch S. OpenEHR-based representation of guideline compliance data through the example of stroke clinical practice guidelines. Stud Health Technol Inform. 2012;180:487-91
6. Ouagne D, Hussain S, Sadou E, Jaulent MC, Daniel C. The Electronic Healthcare Record for Clinical Research (EHR4CR) information model and terminology. Stud Health Technol Inform. 2012;180:534-8.