Data Models' Flexibility in MUDRLite

Nagy Miroslav, Špidlen Josef

Department of Medical Informatics, Institute of Computer Science AS CR, Prague, Czech Republic

Abstract

The EuroMISE Centre focuses on new approaches to the electronic health record (EHR) design. One of pilot applications developed within this applied research is the MUDRLite universal EHR system. MUDRLite connects the user interface described in the XML file with data layer. The structure of data stored in the database is not fixed and is specially designed to fulfil needs of the target field of medicine where MUDRLite is going to be used. Data model describing the data structure is called the ER-model and its actual form stored in a real database is called the database schema. In the real world it is a common situation that it is necessary to modify slightly the ER-model in order to represent more precisely medical data or just to correct design or other errors. In the process of data model evolution a database schema synchronising tool is very useful. This paper deals with the description of database schema synchronisation.

Introduction

MUDRLite [1] was developed with the goal of flexibility. It is capable of being adapted to every area of medicine mainly thanks to simplicity of the solution and usage of the 2-tier architecture composed of user interface (form oriented) and data layer. In the project "*Information Technologies for Shared Health Care*" the mentioned EHR is used for collecting data mainly in the field of dental medicine via an extension module called "Dental Cross" [2]. The second proposed application of the MUDRLite EHR is indented in the field of neurosurgery.

Many computer programs, especially the successful ones, change over time; they evolve. This assertion applies for each type of application, particularly for medical ones. During the development of the ER-model for the Dental Cross component we faced several times the situation when a database schemas synchronising tool was helpful.

Synchronising problem

The main problem is when a new version of application has to replace the old one. The new version typically brings about changes in the data layer (ER-model) as well. This is the point when a synchronising tool comes in handy. The synchronising problem is connected with two database schemas. The first one is called a target schema (used by a new version of application) and the second one a source schema (old version). To synchronise two database schemas means to create a script that after being executed on the source schema causes its transformation into a schema with the same structure as the target schema.

Schemagic tool

Schemagic is a tool developed for database schemas synchronisation. It was developed as a part of master thesis [3]. During the development of this tool a big emphasis was laid on following aspects:

- Basic functionality – the tool should be capable of reading metadata of relational database schemas, compare two database schemas in order to find changes that are to be solved during the synchronisation process and generate the synchronising SQL script as its main product.
- Off-line usage – schema synchronisation capability should be available off-line mode without a direct connection to a particular database at the time of synchronisation. This feature is required for example when there is a need of synchronising the developers' and customer's schemas before the application deployment. It should be possible to store the last version of the schema deployed at the customer in a suitable format and to read it from the file later during the next deployment.
- Universality – the tool should be universal enough to be run on various operating system platforms (e.g. Microsoft Windows, Linux) and capable of co-operating with more than one database system (e.g. Oracle, Microsoft SQL Server).
- Extensibility – the tool should be extensible in different ways. The user (developer) should be able to enlarge the set of supported database objects easily. Synchronising capabilities ought to be improved or added in a convenient way without the necessity of programming Java™ classes – because the user of this tool may be familiar with other programming language than Java™. It also should be possible to add a new sophisticated features (i.e. too complicated to be implemented without programming) by adding a new plug-in.
- Usage of standards – it is required in order to ease the extensibility and make further manipulation with data, created by the tool, conveniently using third parties programmes such as editors, viewers etc. The term "standards" in this meaning represents standards in the computer programming field, not medical standards.
- Automation – as a synchronisation process is executed many times typically with little changes, i.e. schemas of various customers are to be synchronised with the current version (only one schema in the differentiation process is changed) then a command line interface of the tool becomes useful. It helps to manipulate with schemas of various customers and to synchronise schema versions developed in the course of time at one customer. The command line interface enables writing batch scripts that can be used many times or even the integration of schema synchronisation into a building process of the application via the Make utility, commonly used by Unix programmers.
- No data loss – the tool must not affect (delete or change) any data stored in the database that is being synchronised.

- Synchronising of differently named schemas – the ability to synchronise schemas that do not have equal names is required.
- Generating textual report – the tool should be capable of generating a human readable report, e.g. after the schema synchronisation process or after comparing two schemas. For example, it can contain information about the database objects, which were removed, created or modified.
- Database schema documentation – the ability of keeping documentation of the current database schema (current version in the process of its evolution) is also required.
- National languages support – correct handling of database objects named or in any other way linked with usage of language specific characters is an inherent feature.
- Open source – as the project is being developed in academic field, its source codes are written using ideas of design patterns in order to be easily modified and extended by other developers in the future. This will also prevent the tool developed within the thesis from becoming obsolete.

Features of the Schemagic tool predetermine its usage in the process of data models' synchronisation. Mentioned data models are used in data layer of MUDRLite applications. Openness of the tool enables addition of new features or modifications of the existing ones. Owing to universality of Schemagic there were no problems with the database compatibility during the tool's tests. Capability of generating the database schema documentation makes the programmer's documentation creation of MUDRLite based applications convenient.

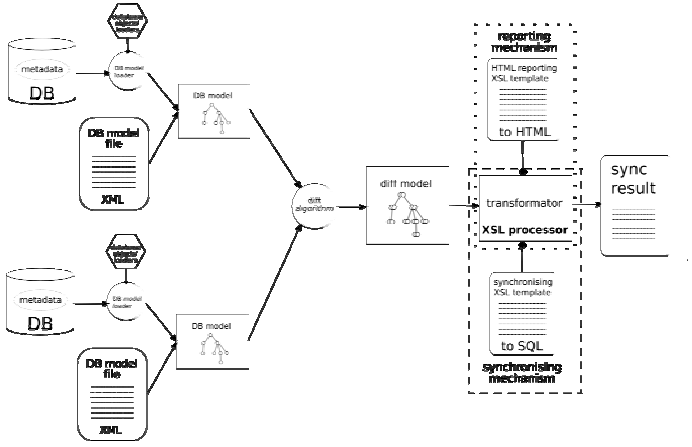

Fig. 6. Schemagic tool functionality schema.

Although the Schemagic tool architecture detailed description is beyond the scope of this paper, a short summary follows. The *Fig. 6* shows what happens with synchronised database schemas during the process until a synchronising script is generated. The schema data are gathered either from a database or a XML file. Database schema representation in computer memory is called database model. The next step of the synchronisation process is execution of the diff algorithm that compares both, the target and source, schemas and finds differences. The diff algorithm produces so called diff model that holds all schemas divergences. Final step is the synchronising script creation via XSL transformation [4] mechanism. Further information about Schemagic can be found in [3].

Conclusion

The connection of MUDRLite system and Schemagic tool mentioned above lead to a completely new approach to development of MUDRLite based applications. Proposed solution created an alternative methodology of simple medical applications development. Using Schemagic synchronising tool further multiplies the power and scalability of MUDRLite EHR system.

Acknowledgments

The work was supported by the grant number 1ET200300413 of the Academy of Sciences of the Czech Republic.

References

[1] Špidlen J., Hanzlíček P., Zvárová J.: MUDRLite – Health Record Tailored to Your Particular Needs. In Duplaga M. at al. (eds.): Transformation of Healthcare with Information Technologies, Amsterdam, IOS Press (2004), pp. 202-209.

[2] Špidlen J., Pieš M., Teuberová Z., Nagy M., Hanzlíček P., Zvárová J., Dostálová T.: MUDRLite – an Electronic Health Record Applied to Dentistry by the Usage of a Dental-Cross Component. (2005), EMBEC'05 Conference Prague.

[3] Nagy M.: Synchronisation of Relational Schemas. (2005), Master Thesis, Faculty of Mathematics and Physics, Charles University in Prague.

[4] Tidwell D.: XSLT, O'Reilly (2001), ISBN: 0-596-00053-7.

Address

Mgr. Miroslav Nagy
Department of Medical Informatics,
Institute of Comupter Science AS CR
Pod Vodárenskou věží 2
182 07 Prague 8
e-mail: nagy@euromise.cz