An Implementation of a Framework for Representing Uncertain Knowledge in Medicine

Christoph Adl
ARC Seibersdorf research GmbH
eHealth Systems
Viktor-Kaplan-Straße 2/1
A-2700 Wiener Neustadt
Austria
christoph.adl@arcsmed.at

Günter Schreier
ARC Seibersdorf research GmbH
eHealth Systems
Reininghausstrae 13/1
A-8020 Graz
Austria
guenter.schreier@arcsmed.at

Abstract

Knowledge-based systems are already a factor to ensure quality in medical treatment and will become more important in the future. Many implementations of knowledge-based systems are isolated stand-alone solutions. On the other hand, many of the so called expert system shells do not have proper tools to represent the inherent uncertainty in medical knowledge, need a very special knowledge representation or the knowledge cannot be reused. The ARC-XS-BASE (Austrian Research Centers-Expert System-Base) represents a new approach based on a concept [6] which will overcome these drawbacks using a modern modular three tier Java architecture and a relational database for storing medical knowledge as well as patient data. Fuzzy-Set theory is used for representing uncertainty (linguistic variables based on fuzzy membership functions, fuzzy numbers, fuzzy rules, fuzzy automats). The database layer containing 123 tables, 209 foreign key relations and the corresponding 54 Java classes have already been implemented.

Keywords: knowledge representation, knowledge-based system, expert system, Fuzzy-Set theory.

1 Introduction

Knowledge-based systems are able to improve the clinical practice. This has been recently shown in a systematic review [5]. A major success factor, as shown in this review, is the integration of the system into the clinical workflow, but many knowledge-based systems are isolated stand-alone solutions that run on separate computers.

In response to this fact expert system shells or specialized programming environments have been developed like:

- CLIPS (C Language Integrated Production System, a good bibliography can be found in the CLIPS Basic Programming Guide [10]) and its extension Fuzzy-CLIPS
  
- Jess[1] and its extension FuzzyJess

- j.MD[2] (Java-System supporting Medical Diagnostic Reasoning) and its predecessor Pro.MD (Prolog-System supporting Medical Diagnostics) [12]

- Arden Syntax (a HL7 Standard, see [4, 8, 3, 9]) and its proposed extension FuzzyARDEN [11]

These systems mainly address the problem of creating a new specialised program for each

---

expert system realisation. So the aim using an expert system shell is the reuse of knowledge.

Since the "knowledge" itself is – of course – the essential basis for all knowledge-based concepts, acquiring this knowledge is a challenge all such concepts face. There are different sources and methods for knowledge acquisition:

1. As the name "expert system" suggests, the knowledge can come from a domain expert (in medicine usually a physician). However, according to the concept of evidence based medicine\[7\], the expert is on 5th level of validity (which is the lowest level) and hence not necessarily a reliable source of information.

2. Knowledge may also be acquired from the literature (scientific papers, text books, conference proceedings, patents, ...), in medicine often from publications about the outcome of clinical trials.

3. A third way to get knowledge is automated knowledge acquirement. This can be done using software like the statistical R package\[3\] or machine learning tools like WEKA\[4\]\[13\].

One last important feature for knowledge-based systems should be mentioned as well: a Web-Interface. Despite this seems to be a simple implementation feature it is very important. It allows an easy access to the system (e.g. for the healthcare personal) and makes it even possible to integrate it into other systems via XML-RPC (Extensible Markup Language - Remote Procedure Call) or SOAP (Simple Object Access Protocol).

The aim of the present development has been to provide an information processing framework for the medical domain which is able to take into account different methods of knowledge acquirement (by extending existing knowledge in the system or acquirement of new knowledge), persistent storage of knowledge, evaluation and roll-out of knowledge-based systems.

2 Material and Methods

2.1 Implementation

ARC-XS-BASE uses a standard three tier architecture (see figure 1).

It is designed to use different standards from the "Java Platform, Enterprise Edition" (Java EE\[5\]) or de-facto industry standards. These standards are used on different layers.

In the data layer Hibernate 3.0\[6\] was used to persist the data to the relational database.

2.2 Extensibility and representing uncertain knowledge

Several forms of knowledge representation can be used in ARC-XS-BASE: e.g. tables, rules, automatons. The system can be extended using new inference mechanisms (e.g. it is possible to use the table representation and a new, self programmed, inference module for generating an artificial neural network and as a knowledge representation mechanism).

\[3\]R Website, http://www.r-project.org/, checked 18.01.2006
\[4\]WEKA Website, http://www.cs.waikato.ac.nz/ml/weka/, checked 18.01.2005
\[6\]Hibernate Website, http://www.hibernate.org/, checked 18.01.2006
The first step in the inference process is the Data-to-Symbol Conversion. The acquired numerical data needs to be converted into symbolic concepts. To take care of uncertain medical terms, this can be done using linguistic variables. An example similar to the one in [6], page 29 will show that.

The numerical value "chymotrypsin level in stool" is measured and stored into the system. The value needs to be interpreted as symptom. The corresponding symptoms are: "pathological chymotrypsin level in stool", "suspect chymotrypsin level in stool" and "normal chymotrypsin level in stool". So we define a linguistic variable \(< v, T, X, g, m >\) (see table 1).

\[
v = \text{chymotrypsin level in stool} \quad \text{(name of the linguistic variable)}
\]

\[
T = \{ \text{"pathological chymotrypsin level in stool"}, \text{"suspect chymotrypsin level in stool"}, \text{"normal chymotrypsin level in stool"} \} \quad \text{(term set of } v, \text{ that is, the collection of its linguistic values)}
\]

\[
X = [0, 10] \quad \text{(universe of discourse for } v)\]

\[
g \quad \text{syntactic rules which generates the terms in } T \text{ are left unspecified because we do not want to generate linguistic terms}
\]

\[
m : T \rightarrow F(X) \text{ is specified by the membership function in figure 2 (semantic rule which associates each linguistic value } t \in T \text{ with a fuzzy subset of } X).
\]

Table 1: The linguistic variable for the interpretation of the numeric value "chymotrypsin level in stool".

Like the Data-to-Symbol Conversion, other steps in the inference process can use methods for representing uncertain knowledge as well. These methods and features are implemented in a set of "Fuzzy Classes" that make use of the Fuzzy-Set Theory:

- Type-1 and Type-2 membership functions (e.g. PI-Form, Gamma-Form, ...)
- Fuzzy Numbers
- Fuzzy Rules and inference mechanisms to derive diagnoses using fuzzy rules
- Fuzzy Automatons and inference mechanisms to derive diagnoses

A more detailed description of methods for representing uncertain knowledge used in the framework can be found in [6].

3 Results

The design of ARC-XS-BASE led to a modular architecture as depicted in figure 3. Its modules are designed to support the whole life cycle of a knowledge-based system. The modules are split into several layers (from bottom to the top):

- Layer 1: The modules below the "System Developer API" are the very core of ARC-XS-BASE. Components to connect to the database, basic user interface containers as well as web- and client-server frameworks can be found here.
- Layer 2: The modules between the "Application Developer API" and the "System Developer API". This layer contains the default mechanisms for inference and consultation that are based on the "System Developer API".
- Layer 3: The modules between the "Application Developer API" and the "Knowledge-Based System". Individually configured software components to
fulfil the needs of a single institution or group of institutions, that use ARC-XS-BASE, are present in this layer.

- Layer 4: The topmost layer shows the different users that work on a rolled-out system.

An exception to this concept is the module "Maintenance/Administrative Tools". It mainly belongs to layer 1, but may access components in higher levels as well.

The first step in the implementation of ARC-XS-BASE was the realisation of the database back-end using Hibernate. The relational data model consists of 123 database tables and 209 foreign key relations (to get a closer look on the entity "patient", see figure 4, for the whole model see figure 5).

The Java object model contains 54 Hibernate beans that are mapped to the relational model. The difference is caused by m:n relations that are represented as lists in Java, but must be resolved as tables in the relational database.

4 Discussion

Although ARC-XS-BASE has not been finished yet, the already completed components contain all necessary elements to manage all required aspects of uncertainty.

To provide for easy integration with 3rd party software components, industry standards were adopted as much as possible. An Example for this is the use of Hibernate in the Data layer. A big advantage using tools like Hibernate is, that the development process is independent of the database that will be used for production. During the implementation PostgreSQL has been used, but it is no problem to use any commercial SQL-compatible database system like Oracle or MS SQL Server later. Hibernate also has the ability to create the database from the code. This means, that no extra database maintenance code – like create/drop table scripts – need to be used.

A challenging task during the implementation was, that the object model given in [6] uses multiple inheritance – a class can have more than one base class. The programming language Java that is used for ARC-XS-BASE does not support this concept. The solution was to use so called interfaces. Every Java class can implement (not extend) several interfaces which are itself classes without implementation. The drawback of this solution is, that every class that for example extends the class "Description" in the object model need to implement the "Description" interface in the Java class model and thus need to implement its own methods for handling descriptions.
Figure 3: Components of the ARC-XS-BASE framework.

Figure 5: To get a general idea about the complexity of the ARC-XS-BASE system, the 123 database tables and 209 foreign key relations of the relational implementation are shown.
4.1 Future Work

The application layer will be implemented using Java frameworks like Spring or Struts. For the presentation layer JSP (Java Server Pages), Java Servlets and/or JSF (Java Server Faces) will be used.

Automated knowledge acquirement is planned to be used in ARC-XS-BASE. This could be realised using WEKA, because it is open source and written in Java as well, which makes integration easy. WEKA supports many different forms of classifiers – from simple decision tables or trees to complex meta classifiers. It is also easy to integrate new classification models into the existing system. WEKA also includes some tools for evaluation of the acquired knowledge (classifiers) like ten-fold cross validation or ROC (receiver operating curve) generation.

As high throughput molecular biology methods like DNA or protein microarrays become more important, the ARC-XS-BASE platform will be extended to deal with such data. These methods usually generate a large amount of data and knowledge-based systems in this so called area of bioinformatics need to be efficient. Additionally there is often not much knowledge about the meaning of most of the parameters acquired by a microarray which is in fact another source of uncertainty that needs to be dealt with.

5 Conclusion

A number of the promises of the dawning eHealth era, like safety and quality enhancements, will strongly depend on new methods of information processing and their ability to deal with incomplete or uncertain knowledge.

Uncertainty – which has always been an immanent element of the medical domain – has to find its way into automated information processing concepts. Our concept has been developed to deal with these issues and to, eventually, make eHealth systems more intelligent.

References


