SISTEMAS ESPECIALISTAS NA INTERNET: A IMPLEMENTAÇÃO DO DELEITE

José Luís Braga
Full Professor
Departamento de Informática
Universidade Federal de Viçosa – Brazil
zeluis@mail.ufv.br

Márcio Katsumi Oikawa
MS in Computer Science
USP-Instituto de Matemática e Estatística – Brazil
koikawa@ime.usp.br

RESUMO

Apresentamos nesse artigo uma descrição do sistema especialista DELEITE, construído para ser um auxiliar no diagnóstico de problemas em rebanhos leiteiros. Os diagnósticos são baseados no *intervalo entre partos*, uma medida bem aceita para avaliação inicial de produtividade de rebanhos leiteiros e na detecção de problemas com alimentação, saúde e reprodução do rebanho. O sistema foi construído evolutivamente, passando por quatro versões, cada uma melhorando a anterior, pelo uso de ferramentas e técnicas mais atualizadas e apuradas. A versão mais bem sucedida do sistema foi a cliente-servidor, extensivamente utilizada via internet. Dois especialistas no domínio do conhecimento participaram, durante todo o processo de desenvolvimento do sistema, tendo fornecido o conhecimento codificado em 104 regras contidas na sua base de conhecimento.

Palavras-chave: rebanho leiteiro, sistemas especialistas, cliente-servidor, sistema baseado em regras, base de conhecimento

EXPERT SYSTEMS ON THE INTERNET: IMPLEMENTATION OF DELEITE

ABSTRACT

In this paper, we present shortly the DELEITE expert system, built to help diagnosing milk production problems in dairy cattle herds. Diagnostics are based on *interval between births*, a common and well accepted measure of herd productivity, and problems in feeding, health and reproduction are addressed. The system evolved through four versions, each one improving over its antecedent, by using more up-to-date tools and techniques. The most succesful version was the client-server web based one, which was extensively used in Brazil. Two dairy cattle milk production experts took part in the system development, and their knowledge was transferred to the system's knowledge base, and encoded in 104 rules.

Key Words: dairy cattle herd, expert systems, client-server, rule system, knowledge-base

INTRODUCTION

In this paper, we describe shortly the history of Deleite [1], an expert system built to be an aide to decision makers in the area of dairy cattle milk production, taking into account health, feeding and reproduction problems. The forecasted needs in milk production in Brazil for the near future is around 30 million liters. Attaining this goal is a great challenge for farmers, industries and technicians in this area, and this expert system is our modest contribution towards the stated goal.

The system was first developed as a undergraduate artificial intelligence class project in 1993 at Universidade Federal de Viçosa, Brazil. We kept developing the system since then, through four main versions, each one improving formally and technologically over the former versions. Briefly, versions were:

- *first version*, developed using a DOS release of LEVEL5. This first version was a discardable prototype, built with the purpose of having a working timely knowledge base;
- *second version*, derived from the previous version, implemented using a Windows release of EXSYS. This version had a almost complete and minimal knowledge base, that was able to cover all the cases and problems that were the project's goals;
- *third version*, developed under the client-server architecture [11], available to be used through the internet. It was implemented using CGI [4] for communication between client and server, plus HTML [4] on the client side interfaces. For the server side, the knowledge base was hand-compiled into an efficient data structure, and implemented using Delphi;
- *fourth version*, a prototype using CORBA [7,8,9,11] in the middle layer, instead of CGI. For this prototype, the expert system was rebuilt using JESS [10], a Java version of CLIPS. This version was not made available for use through the internet.

In what follows, we describe briefly the second and third versions of the system, and then make a comparison between CGI and CORBA techniques for this particular case.

SYSTEM DEVELOPMENT

We adopted the knowledge systems development approach based on prototyping, and used think-aloud protocols and interviews with the specialists for knowledge elicitation [3]. The specialists took part in the whole process, with weekly meetings and discussions. In general, each week they had available a new version of the expert system, and they used it for searching for bugs and for proposing enhancements. Each session lasted from one to two hours, and they were very much effective in terms of results.

We had in mind that the system was supposed to be addressed to a large audience, ranging from dairy cattle technicians and experts, to people more related to the herd and its everyday field activities. So, we established two additional goals regarding system usage: a)the system should output a well organized and easy to understand script telling the user the current possible herd problems diagnosed;

and b)the system should be used as a helper to extension agents dealing directly with dairy herd problems.

Knowledge in the system's knowledge base was taken from three different areas: feeding, reproduction and health. Knowledge was represented using EXSYS causal rule language **if** ***<antecedents>*** **then** ***<consequents>*** where, as usual, *<antecedents>* are the conditions that must be satisfied so that *<consequents>* could be assumed true. The main decision variable used by the system was the *interval between births*, considered by the experts to carry enough information to allow a diagnostic good enough for our purposes. The use of this variable alone is disputable, but it is not a concern for the scope of this paper. The final knowledge base obtained in the EXSYS version of Deleite was composed of 104 rules. They were analised for consistency and minimality, and the system was tested by other experts and students, and was considered reliable enough to be used at least as a learning tool.

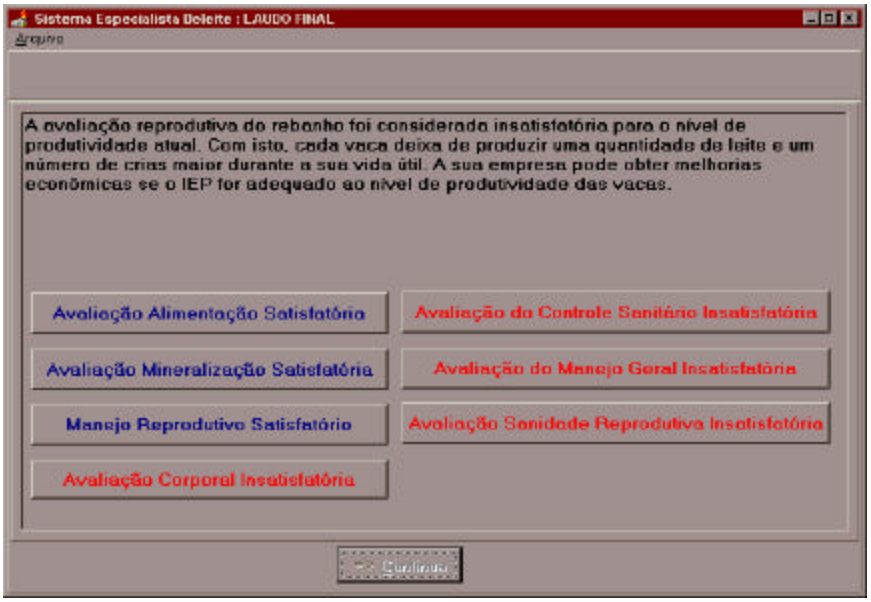

FIGURE 1: Final diagnosis client interface.

The consultation process is carried out through an interaction between the user and the system, using simple interfaces, designed to provide safe communication between the user and the system. They ask the user to input some data about the herd, and those data are then processed locally by EXSYS's inference engine, that reasons about the problem submitted returning a question to the user, or a final diagnostic.

Diagnostics are shown to the user in a format like the one shown in Figure 1. Blue text means a positive evaluation of the herd, showing that there is no problem. Red text is an indication of possible problems. The user can then have access to the diagnostic by clicking in one of the bars, and a text will

appear containing more information about the problem detected and the possible approaches to the solution.

The Internet Version

In 1997, we decided to focus our attention on a client-server [11] internet version of the system. In this new approach, we divided the project in three parts: **client part**, in which HTML forms enhanced with scripts were designed and implemented; **communication part**, or middle layer, implemented using CGI; **server part**, implemented in Delphi.

The interfaces implemented for the client side had the same content as those from the EXSYS version. They were all implemented as HTML forms stored in the application/server side, and were transferred to the client machine as requested, using CGI as the communications gateway. Example patterns for the system interfaces are shown in Figure 2.

To have the expert system available on the server side, it was fully implemented again, using Delphi as the implementation language. This was the toughest part in this version, since we had to hand-compile the knowledge base in a way that it could be efficiently used. We obtained a graph-like structure, that was then coded using conventional data structures in Delphi.

This knowledge base can not be easily augmented in this version of the system. The insertion of new rules has to be done by hand, and the base compiled again. That was a project decision, and the whole data structure was designed to allow knowledge based maintenance in the future, through the use of a special interface. When this Delphi version was finally ready, interfaces were added to the system and they were all packed together and became a standalone demo of the system, that is now freely distributed to anyone who requests an operational copy of the system.

The middle or communications layer was implemented using CGI. We overcame the problem of not having a local memory to hold the status of a consultation session by keeping a *log* of each current consultation session on the server, recorded in a global file structure. HTTP [6] was used as a forms (file) transfer protocol, and we used a free version of the server Omni HTTPd [5], that was very well suited to our needs. For FTP transfers, we used another free server, Serv-U [13].

FIGURE 2: Client interfaces – web.

The CORBA Prototype

Next step was the implementation of the system under a distributed architecture. CORBA (Common Object Request Broker Architecture) [12] was used in the middle layer, together with a Sun Microsystem's broker package included in JDK 1.1.8 (Java Development Kit 1.1.8) [11,14].

CORBA is an architecture driven to distributed objects, making the communication between client and server in a transparent way. Servers offer services (object methods) and clients request services concurrently. A scenario of the CORBA version of Deleite use can be described as follows: the user, in a client computer, enters his data using a special form (HTML or Java applet); the client machine establishes a connection with a server computer.

The client machine requests the execution of a forms data validation operation from the server; the server gets the data, and activates the Jess version of Deleite. Once the data are processed and inferences are carried out, a response is sent back to the client; the client machine gets the server response; another form is then exhibited to the user on the client machine screen, containing the next step in the consultation process; etc.

The main advantage of using CORBA instead of CGI is that the whole communication process between client and server is managed by the broker machine, that has the responsibility of message codification and decodification, and response time optimization. On the client side, operation requesting is done as if the server services were local, so relieving system developers of the burden of having to control data transmission. Another great enhancement is that Deleite could be split in logical pieces, that could be run in different and geographically distributed server machines.

CONCLUDING REMARKS

In developing Deleite, we were lucky enough to have the full cooperation of the two named experts, that worked hard as part of our crew to have a consistent and as complete as possible knowledge base.

Owing mostly to lack of funds, we had to develop the system described here using free software as much as possible, and having to deal with less than ideal servers and programming tools. The internet version of the system was very much succesfull, as we could infer from our users feedback. The project is now discontinued, although there are many technological issues that could still be addressed: building a fully operational CORBA version, building the compiled knowledge base editor, and making the system able to learn and extract knowledge from consultation logs and sessions, using artificial intelligence techniques.

REFERENCES

BRAGA, José Luís, SOUSA, Emerich Michel de, PEREIRA, Nádia Vieira Leão. *Sistema Especialista no Controle de Gado de Leite.* Agrosoft 95. 25 a 27 de outubro de 1995. Juiz de Fora – MG.

DEITEL, H. M, DEITEL, P.J. *Java: How to Program.* Upper Siddle River. USA: Prentice Hall. 2nd ed. 1998.

FIRLEJ, M., HELLENS, D. *Knowledge elicitation: a practical handbook.* London, UK, Prentice Hall, 1991.

GUNDAVARAM, Shishir. *CGI Programming on the World Wide Web: A Nutshell Handbook*. Sebastopol, USA: O'Reilly & Associates, 1996.

Omnicron Technologies Corporation. http://www.omnicron.ab.ca/httpd/download.html;

RAMALHO, José Antonio. *HTML Avançado.* São Paulo, SP, Makron Books, 1997.

Complete CORBA. http://www.iconixsw.com/CompleteCORBA.html

CORBA Tutorial Material. http://www.cs.wustl.edu/~schmidt/tutorials-corba.html

A Comparison of CORBA, DCOM and RMI. http://mordor.cs.hut.fi/~kta/corba-comparison

JESS – The Java Expert System Shell. http://herzberg1.ca.sandia.gov/jess/ ;

ORFALI, Robert, HARKEY, Dan, EDWARDS, Jeri. *Instant CORBA*. New York, NY, John Wiley & Sons, 1997.

HENNING, Michi, VINOSKI, Steve. Advanced CORBA Programming with C++. Addison-Wesley, 1999.

Serv-U FTP Server. www.ftpserv-u.com

SUN MicroSystems Inc. http://java.sun.com