N92-16582

PROJECTS IN AN EXPERT SYSTEM CLASS

George M. Whitson

Computer Science Department
The University of Texas at Tyler
Tyler, Texas 75701

Abstract. Many universities now teach courses in expert systems. In these courses students study the architecture of an expert system, knowledge acquisition techniques, methods of implementing expert systems and verification and validation techniques. A major component of any such course is a class project consisting of the design and implementation of an expert system. This paper discusses a number of techniques that we have used at The University of Texas at Tyler to develop meaningful projects that could be completed in a semester course.

HISTORY

As expert systems theory became a major subset of Artificial Intelligence, many universities introduced courses on his subject. Initially, there were few texts in the field and little software that was available for classroom use. Today there are many texts and inexpensive software systems to support classroom instruction. A problem in the teaching of expert systems, since such courses have been offered, is designing classroom projects that are both meaningful and that can be done in a one semester period. This paper discusses a number of projects that I have assigned over the past few years. The type of project assigned depends on the emphasis of the expert systems course, as we will illustrate in the next two sections. For those courses based on a high level shell, like CLIPS, we feel we have developed a very good method for having class projects that are both meaningful and do-able.

TYPES OF EXPERT SYSTEMS COURSES

Although the expert system field is quite young there have been a number of different types of expert systems courses. Until I started using CLIPS two years ago we tried a lot of different approaches to teaching expert systems. At our university I have taught the course with each of the following emphasis:

1. In our first expert systems course, taught in 1984, we emphasized artificial intelligence and discussed all aspects of the theory of expert systems in terms of artificial intelligence. For example, we did not look at fact

representation until we had done a complete coverage of knowledge representation.
2. In our second approach to teaching expert systems we gave a quick introduction to artificial intelligence, a solid coverage of all of the different implementation languages that were in use, including LISP, PROLOG and GOLDWORKS.
3. In our third major revision of the expert systems course we gave a brief introduction to artificial intelligence, an introduction to logic programming and spent the remainder of the course looking at techniques for implementing expert systems in PROLOG.
4. In our fourth approach to teaching expert systems we expanded the definition of an expert system to include many systems that were not rule-based. In particular, we spent a good deal of time looking at artificial neural systems and how they were related to rule-based expert systems.
5. In our final and current approach to teaching an expert systems course we quickly introduce the ideas of a rule-based expert system, get the students started on a major project and then lecture on other important topics, like fuzzy logic and artificial neural systems.

EXPERT SYSTEM PROJECTS

In each of the different types of expert systems courses we used different types of projects. It is interesting to consider how these projects have changed as the course content has changed. A sketch of the projects we gave in each of the above types of courses is:

1. When we stressed artificial intelligence we alway gavs a project to develop an expert system shell in LISP. The sketch given in [Winston, 1989] the approach we usually used. The projects were interesting from the systems point of view, but, as one can well imagine, the students spent all their time developing the shell and little time using it.
2. When we stressed a comparison of the different languages and shells used to develop expert systems, we found that we would always take a simple project so that we could implement it in several shells. Each student did the same problem in two different ways. All projects were presented to the class and then a test covering all approaches was given. As one could easily guess, we spent most of our time learning the syntax of the different tools.
3. When we based our expert systems course on PROLOG we were able to develop a fairly complex project, but fond that much explanation of logic programming was needed. Some of the specific techniques of PROLOG, especially unification, required a lot of explaining as well. While we still use PROLOG occasionally as a tool, we still find that we cannot spend as much time as we want on the real topics of expert systems, like knowledge acquisition, because of the need to explain programming language details.
4. When we emphasized artificial neural systems in our

expert systems course, the project consisted of developing an artificial neural system and an expert system to solve the same problem. While it was interesting to compare the two techniques, we realized at the end of the project that the rule-based system was not as complex as desired. This resulted from our desire to be sure we had selected a problem that had a solution in both technologies.

5. When we based our expert systems course on a shell for the project we found that with a little work we were able to develop a fairly complicated project in a single semester. While all of our previous methods of teaching the course were successful, we feel that the current approach is ideal for today in that the students spend all of their time developing and testing a real expert system rather than doing other things.

PROJECTS IN A CLIPS BASED COURSE

There are many techniques used to develop a project in an expert systems course. The one we most often hear presented at meetings is to simply require each student to develop a project on their own. The instructor provides guidance and controls the quality of the project, but the student selects the topic and finds the domain expert. While this is fine for students who are domain experts, we don't think it works very well for those who are not. We have developed a method for selecting projects that we feel is very good. This method has resulted in our developing several nice prototypes as a part of the class and has been enthusiastically received by our students. We now list some of the key components of our method of doing projects in an expert system course.

1. We define a high level tool that everyone will use to do their project. For the past two years we have used CLIPS and plan to continue to use CLIPS in the future. CLIPS is easy to use and has all the features we really want in our expert system shell. In addition, students can do much of their development work at home. We have also used GOLDWORKS and EXSYS, but prefer CLIPS.
2. I select a single project for the entire class and line up some domain experts who have an interest in developing an expert system. So far we have found domain experts who would participate in our project in several ways. In one case we found that we had four personal computer repair people in a class, in another we found several Biologists we wanted to see a diagnostic expert system for plant disease developed, and in another we found a local manager who needed to have his knowledge built into a system.
3. The class is divided into teams of about five students each. This division is dictated by our access to experts. Each team selects people to acquire the knowledge, code the system, test the system and develop the system documentation. All students are responsible for all phases of the development, but selected students are leaders in certain parts (like knowledge acquisition.)

4. The class project is described early and students begin developing the system shortly after the first week of class. This means that the expert systems theory about topics like knowledge acquisition and system verification are often covered in the lecture after students have practiced some of the ideas in their own system.
5. A major emphasis of our course is on knowledge acquisition via the traditional techniques of observation, interviewing and becoming a pseudo-expert. We spend several weeks each semester on knowledge acquisition and cover several automated techniques as well as the traditional methods.
6. Each team develops a complete prototype system. The systems are usually similar since I work with each team and present an overall suggested system design at the start of the course. But each project always has its own unique flavor. Members of the team do both verification and validation of the system as well as developing a user's manual. These prototypes have often been developed into full systems after the course is finished. At the end of the course each team demonstrates their system to the class and all systems are discussed and evaluated by the class as a whole.

SUMMARY

As one teaches an expert systems course with different emphases, projects take on different forms. Some projects develop expert system tools, some survey the current tools and some concentrate on the art of building real expert systems. After experimenting with many different types of course and projects, I have come to the conclusion that it is best to teach an expert systems course that concentrates on developing a real system. For such a course I have found that a single class project implemented by teams is best.

REFERENCES

Giarantano and Riley (1989). *Expert Systems: Principles and Programming,* PWS-KENT, Boston.

Hayes-Roth, Waterman and Lenat (1983). *Building Expert Systems,* Addison-Wesley, Reading, Mass.

Malpas, John (1987). *PROLOG: A Relational Language and its Applications,* Prentice-Hall, Englewood Cliffs, New Jersey.

Waterman, Donald (1985). A Guide to Expert Systems, Addison-Wesley, Reading, Mass.

Winston and Horn (1989). *LISP, 3rd Edition,* Addison-Wesley, Reading, Mass.