PERGAMON

Expert Systems with Applications 25 (2003) 231–246

Expert Systems with Applications

www.elsevier.com/locate/eswa

Building an intelligent system using modern Internet technologies

Goran Šimić[a], Vladan Devedžić[b,*]

[a]Military educational center for signal, computer science and electronic warfare, Veljka Lukića Kurjaka 1, Belgrade 11000, Yugoslavia
[b]FON—School of Business Administration, University of Belgrade, Jove Ilića 154, P.O. Box 52, Belgrade 11000, Yugoslavia

Abstract

This paper is a detailed case study of building Code Tutor, a Web-based intelligent tutoring system (ITS) in the domain of radio communications. It is ontologically founded and was built using CLIPS and Java-based expert system tools, latest integrated graphical CASE tools for software analysis and design, and Java servlets. In Code Tutor, Apache HTTP Server stores and serves static HTML pages, and Apache JServ Java package enables dynamic interpretation of user defined servlet classes and generation of active HTML pages. XML technology is used to generate files that Code Tutor uses to provide recommendations to the learners. Such a rich palette of integrated advanced technologies has greatly alleviated the system design and implementation, and has also led to interesting solutions of a number of problems common to many ITSs. The paper describes these solutions and useful design decisions, and discusses several practical issues related to architectures of intelligent Web-based applications.

Keywords: Expert systems; Intelligent tutoring systems; Knowledge representation; Ontology

1. Introduction

Using current Internet technology to support learning in the classroom is recently getting much easier and much more feasible than it used to be. If a network of computers or workstations is available in the classroom, it is easy to install and use Apache, Orion, Tomcat, or another Web server on a dedicated server machine to distribute HTML pages generated statically or dynamically by an educational application. Client computers/-workstations should only have an Internet browser. Hardware and software requirements for the client machines are minimal. We acquired experience with this technology when developing an intelligent tutoring system (ITS) called *Code Tutor*.

Code Tutor is a small Web-based tutor designed for fast students' briefing in the area of radio-communication. The learners are telecommunications college students. After they complete a course in radio-communication theory, they are supposed to exercise using expensive radio-station equipment. It is the teachers' responsibility to ensure that the equipment is always in a good condition and that it is used appropriately. There is little time for checking each student's capabilities for independent practical work when the course is over and before the exercises begin. Code Tutor is used instead.

The first version of Code Tutor has been actively used in the classroom since mid-2001. The teachers' opinion is that it is very useful, and the students favor this kind of learning. However, the first version has some limitations. Its expert module is implemented as a rule-based expert system (ES) using an ES shell that was too old, without support for network-based applications. Also, due to these weaknesses of the shell the entire system is limited to 'closed-world' standalone applications, without connections and data interchange with the environment.

These facts have motivated us to build a new version. The entire system is implemented in Java, using *JBuilder 6* tool for developing Web-based and Java-based applications (Borland Corporation, 2001). The system's mid-layer is designed using *Rational Rose* tools for software analysis and design (Rational Corporation, 2001). The new Code Tutor integrates many different current technologies:

CLIPS, a tool for building ES (CLIPS, 2002) is used to generate knowledge base files.
Java-based ES shell *Jess* is used to interpret these files (Sandia National Laboratories, 2002; Friedman-Hill, 2002).
Students communicate with the system through a standard Web browser.

* Corresponding author. Tel.: +381-11-3474123; fax: +381-11-461221.
E-mail address: devedzic@galeb.etf.bg.ac.yu (V. Devedžić).

0957-4174/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
doi:10.1016/S0957-4174(03)00049-6

Java™ Servlet technology (Sun Microsystems Corporation, 2001; Hall, 2001a,b) is used to implement the system's interactions with the students.
Apache server (The Apache Software Foundation, 2001a) is used to store static HTML pages.
Apache JServ (The Apache Software Foundation, 2001b) is used to interpret the servlets.
XML technology (The World-wide web consortium, 2001) is used to generate files that Code Tutor uses to provide recommendations to the students.

When developing the new Code Tutor, our goal was not to further the research in Web-based ITS per se. Rather than that, we wanted to re-design and re-develop a *working* system, to make it *work* in a Web-based learning context, to make it *intelligent* at least to an extent, to have both the teachers and the learners actually *use* it, to document its design (made using current Web technologies), and to acquire valuable practical experience for further research and development of intelligent Web-based learning systems. Also, with the system's purpose (students' briefing) in mind we wanted to keep the system small and simple. For the sake of all these goals, in the design of Web-based version of Code Tutor we have deliberately sacrificed some features that are certainly important in Web-based learning systems (e.g. adaptivity and integration of stand-alone domain service systems). In that sense, it should be noted that Code Tutor is actually Web-enabled and Web-ready, intended primarily for use in the classroom, rather than a full-fledged Web-based ITS built with the major goal of using it adaptively over the Web.

The paper describes the new version of Code Tutor thoroughly. The next section briefly overviews related work of other authors relevant for the design and development of the new version, providing the context within which Code Tutor is best understood. Section 3 analyzes the system's use cases and shows its architecture. Design and implementation details are discussed in Section 4, while Section 5 specifically addresses Code Tutor's intelligent behavior. Section 6 covers important issues of Code Tutor's user interface design.

2. Related work

In designing the new version of Code Tutor as a Web-based ITS, our situation was much like the one in which authors of other ITS have been already: we needed to convert a stand-alone ITS to one that operates on the World Wide Web. There were a number of architectural paths from which we might have chosen. Hence, we first studied the scope of Web-based ITS architectures. Extensive discussion on categorization of such architectures by Alpert, Singley and Fairweather (1999) and Mitrović and Hausler (2000) was our starting point. They have found out that architectures of many Web-based ITS are either *centralized* (the application server performs all tutoring functions), or *replicated* (the entire tutor resides in a Java applet that needs to be downloaded and is executed on the student's machine), or *distributed* (tutoring functions are distributed between the client and the server). Each category has some advantages and some disadvantages, described elsewhere as well. For example, Johnson, Shaw, and Ganeshan (1998) discuss feasibility of client-side tutoring deployed in their pedagogical agent called Adele.

Then we studied architectures of a number of specific Web-based ITS, in order to find out about their characteristics and to relate them to our idea of Code Tutor as a Web-based ITS. In the first version of Code Tutor we used rule-based ES reasoning in the GUIDEON style (Clancey, 1983), and we wanted to continue using it in the new, Web-based version. An example of a Web-based ITS that uses an ES is PAT Online, a model-tracing Web-based algebra tutor (Ritter, 1997). We also wanted to deploy XML technology in Code Tutor, and a good example of how to do it is ActiveMath, a generic Web-based learning system that dynamically generates interactive math courses with the content represented in XML-based format and presented to the learner via a standard Web browser (Melis et al., 2001).

The architecture proposed by Retalis and Avgeriou (2002) is important for our work in that it is based upon standards and practices from international standardization bodies, as well as on the practices of well-established software and hypermedia engineering techniques. We also studied the architectures of VALIENT, a Web-based database design learning environment (Hall & Gordon, 1998), and ILESA, a Web-based tutor for linear programming problems (López, Millán, Pérez-de-la-Cruz, & Triguero, 1998). More recently, we also found other ideas that have something in common with Code Tutor—Prentzas, Hatzilygeroudis, and Garofalakis (2002) use hybrid rules for knowledge representation in Web-based tutoring, Rebai and de la Passardiere (2002) try to capture educational metadata for Web-based learning environments, and Ahmad and Lajoie (2001) use an integrated learning model to facilitate Web-based instruction.

Theoretical work of Brusilovsky (1999), as well as ELM-ART (Brusilovsky, Schwartz, & Weber, 1996) and ELM-ART II (Weber & Brusilovsky, 2001) systems for learning programming in LISP cover a number of important issues related to adaptivity of Web-based learning environments, such as providing adaptive navigation support to the learner, links annotation, and adaptive curriculum sequencing. We are aware of the importance of adaptivity of Web-based tutors, but we haven't designed the present version of Code Tutor to have such features.

3. System analysis and architecture

Code Tutor is a client-server learning environment designed as a Web classroom (Fig. 1). Students and teachers