

Design Pattern ITS: Student Model Implementation

Zoran Jeremić , Vladan Devedžić

FON – School of Business Administration, University of Belgrade
Jove Ilića 154, POB 52, 11000 Belgrade, Serbia and Montenegro
jeremycod@yahoo.com, devedzic@galeb.etf.bg.ac.yu

Abstract

The paper presents implementation of the student model in the Design Pattern intelligent tutoring system. The student model is created by using a model template which is filled in with new attribute values. The same principle can be applied to other ITS as well.

1. Introduction

Design Pattern is an intelligent tutoring system (ITS) learning design patterns described in pattern catalogues in the "Design Patterns" book [2]. During each session, the system permanently assesses the student's skills and adjusts the student model accordingly. The system adjusts the material to be presented to the student according to the parameters of the student model.

Fig. 1 depicts the system's architecture, composed of the Pedagogical module, Expert module, Student model, Domain model, Coordinator and GUI.

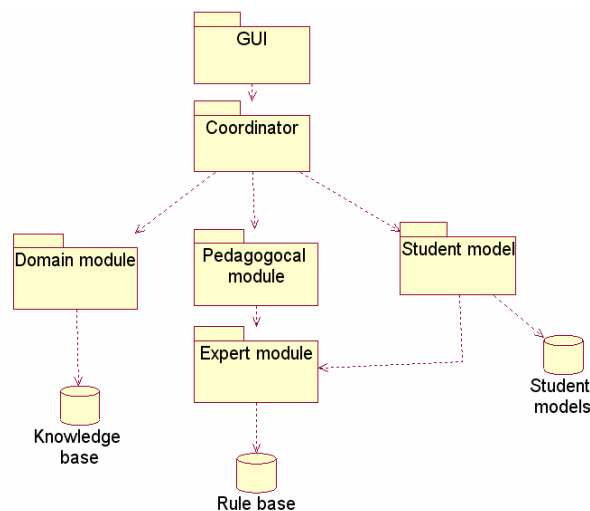


Figure 1. *Design Pattern* system architecture

Pedagogical module provides the knowledge infrastructure necessary to tailor the presentation of the teaching material according to the student model. *Expert module* uses Jess (Java Expert System Shell) as

a rule-based inference engine. *Pedagogical module* uses the *Expert module* for making decisions in curriculum sequencing and evaluating the student model. *Domain model* contains the knowledge about design patterns and the actual teaching material. *Coordinator* controls the functionality of the whole system. We use an HTML-based *GUI* on the client side, and Tomcat 4.0 Web Server as JSP container on the server side.

2. Student modeling

Student model stores details about the student's current problem-solving state and long-term knowledge progress, essential for adapting the material to the student's characteristics (attributes). In this paper, three categories of student's characteristics are considered:

- 1/ personal data - the student's personal characteristics (name, ID, e-mail ...).
- 2/ performance data - the student's cognitive and individual characteristics, as well as other general long-term characteristics.
- 3/ overlay data - the current level of mastery of design patterns and attributes related to the corresponding elements in the domain model.

There is a number of techniques for student modeling; the most frequently used ones are overlay model, stereotype model, and combination model [3]. We used the combination model in the *Design Pattern* system.

When the student registers to the system for the first time, the newly created student model is initialized with default values from a stereotype. It is the system that selects the stereotype, based on the student's initial interaction with the system. *Design Pattern* gradually introduces other characteristics into the student model based on the estimated student's knowledge, such as degree of mastery, experience level, learning style, detail level, etc.

Attribute values in the student model are calculated by applying a dedicated group of rules and simple functions from *Pedagogical module*. The values are updated throughout the session.

At the end of each session, the system stores the student model as an XML document. The next time the student logs onto the system, the data from the stored XML document are used to initialize the student model.

3. Components of the student model

Several components in the *Pedagogical module* make decisions about the student model. The decisions are related to the selection of an appropriate topic for the student, selection of an appropriate learning method, curriculum sequencing, and so on. During such processes, the *Pedagogical module* communicates with the student model in order to get relevant information.

The student model has two components (see figure 2):

- 1/ Performance model stores data related to the assessment of the student's overall skills, as well as data related to the student's previous knowledge, learning style, etc.
- 2/ Teaching history model keeps track of the material presented to the student during the session and the student's mastery of teaching units [4].

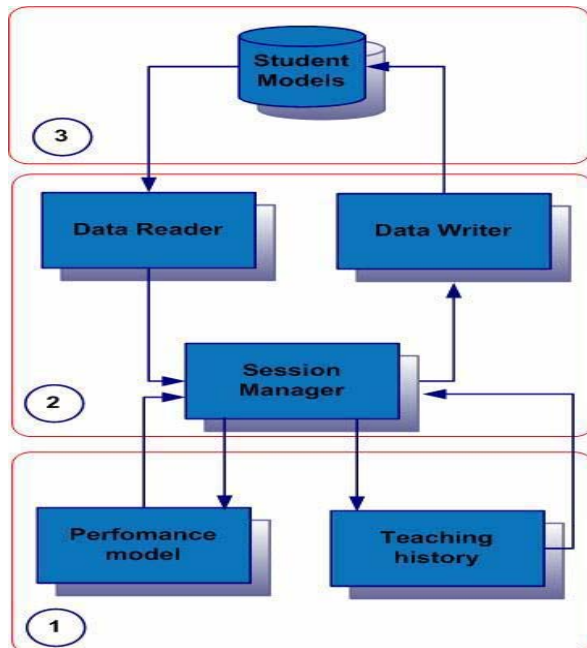


Figure 2. Student modeling component's: 1 – Student model, 2 – Student model creation and management, 3 – Student models database

Another 3 components take part in creating the student model as well (see figure 2):

- 3/ Data Reader enables reading-in the student model data from the XML document; it is based on SAX Parser, and is invoked at the beginning of the session.
- 4/ Data Writer stores the student model to the XML document at the end of each learning session.
- 5/ Session Manager coordinates all other components.

4. Conclusion

The student model of the *Design Pattern* ITS is functionally fully decoupled from the other components of the system. The feature that distinguishes the proposed model from other student models is the way it stores all the relevant data about the student – a separate XML document, created when the student registers on the system.

This approach to student modeling uses a combination of stereotype and overlay techniques. Its main advantage compared to other similar models is a universal approach to model creation. The approach is not strictly related to design patterns as the domain of teaching/learning. It can be applied to ITS in any domain without changes or with a few changes, depending on the requirements of the pedagogical module.

5. References

- [1] A. Bunt, C. Conati, "Probabilistic Student Modeling to Improve Exploratory Behaviour," *Jour. of User Mod. and User-Adapted Inter.* 13(3), 2003, pp.269-309.
- [2] E. Gamma, et.al., "Design Paterns –Elements of Reusable Object-Oriented Software," Addison-Wesley Publishing Company, USA, 1995.
- [3] O. Conlan, "Novel Components for supporting Adaptivity in Education Systems – Model-based Integration Approach," In *Proc. of the 8th ACM Int. Conf. on Mult, CA, USA, 2000*, pp 519-520.
- [4] Z. Yuijian, M.W.Evens, "A Practical Student Model in an Intelligent Tutoring System," In *Proc. of the 11th IEEE Int. Conf. on Tools with AI, USA, 1999*, pp. 13-18.