Synergy of Performance-Based Model and Cognitive Trait Model in DP-ITS

Zoran Jeremić[1], Taiyu Lin[2], Kinshuk[2], and Vladan Devedžić[1]

[1] FON – School of Business Administration, University of Belgrade, Serbia and Montenegro
jeremycod@yahoo.com, devedzic@fon.bg.ac.yu
[2] Advanced Learning Technologies Research Centre, Massey University, New Zealand
t.lin@massey.ac.nz, kinshuk@ieee.org

Abstract. Information about the student in student model is the basis for virtual learning environments to provide the necessary adaptation. Cognitive Trait Model (CTM) profiles the student based on cognitive traits, such as his/her working memory capacity and inductive reasoning ability. Performance-based adaptation can guide the student to the required concept, whereas cognitive support serves to prevent the student's cognitive overload while still representing sufficient challenges to the student. This paper describes the synergy of a performance-based student model and CTM in an intelligent tutoring system called DP-ITS.

1 Introduction

Information about the student in student model is the basis for virtual learning environments (VLEs) to provide the necessary adaptation. There are many different types of student models such as overlay model [7] and differential model [8], implemented in various VLEs (including adaptive hypermedia systems and intelligent tutoring systems). However, most of the student models in existing VLE are what is called performance-based models [4]. A performance-based model (PBM) profiles the student using his/her domain performance. The adaptive support the VLE can provide is therefore limited to what the PBM supports – the student's domain performance.

Lin, Kinshuk and Patel proposed a different kind of student model called Cognitive Trait Model (CTM) [4]. CTM profiles the student based on his/her cognitive traits, such as working memory capacity and inductive reasoning ability. Due to the nature of cognitive traits, CTM can be persistent and stay valid over a long period of time, is transferable across different domains and courses, and can provide the necessary information for the VLE to provide cognitively adapted support. Furthermore, CTM can be used together with any PBM. The information recorded in CTM is qualitatively different from that in a PBM, hence if the two models are used together the adaptive support a VLE can provide is thereby also different but complementary.

L. Ardissono, P. Brna, and A. Mitrovic (Eds.): UM 2005, LNAI 3538, pp. 407 – 411, 2005.

This paper describes both performance-based model and CTM and a synergistic combination of them in an intelligent tutoring system called DP-ITS. It starts with a detailed description of DP-ITS and its student model. Then it analyses and discusses the proposed approach, and finally summarizes its benefits and open issues.

2 Student Models in DP-ITS

DP-ITS is an Intelligent Tutoring System (ITS) for teaching design patterns for software engineering [3]. A frequently encountered issue in teaching design patterns is the organization of the learning process. With the help of DP-ITS, it is possible for both tutorial mode and self-paced mode to learn design patterns. DP-ITS provides an intelligent representation of educational material adjusted to the parameters of the students' performance, such as the background knowledge, performance in the current domain, and cognitive capacity.

The structure of DP-ITS follows the design is consisted of Pedagogical Module, Expert Module, Student Model, Domain Model, Coordinator and GUI [1]. *Domain Model* is designed as a network of concepts. A concept corresponds to a single design pattern. Each concept is decomposed in units – elementary pieces of domain knowledge. *Pedagogical Module* provides the knowledge infrastructure necessary to tailor the presentation of the teaching material according to the student model. *Student model* is explained in detail in the next section. *Pedagogical module* uses the *Expert Module* for making decisions in curriculum sequencing and evaluating the *Student Model*. Expert Module deploys Jess (Java Expert System Shell) rule-based inference engine to reason about the student model and the system's pedagogical actions. *Coordinator* controls the functionality of the whole system. HTML-based *GUI* is used on the client side and Tomcat 5.0 Web Server as JSP container on the server side.

The student performance model stores and updates data about the student's domain performance. It is essential for system operations that adapt instructional material to the student's characteristics [1] and comprises both the model of the student and the mechanisms for creating the model.

The student performance model may have any number of characteristics of the student, depending on the system requirements. In DP-ITS, three basic categories of the students' characteristics are used:

1. Personal data – personal characteristics of the students (name, ID, e-mail, etc.).
2. Performance data – information about the student's domain performance (long-term characteristics generally).
3. Teaching history – the knowledge of design patterns and attributes related to the topic in the domain model. These characteristics are related to the corresponding chapters (teaching history), but they are also used to update the overall assessment results.

When registering a new student, the system creates the student's model and populates it with XML-based data with default values. Based on the student's initial interaction with the system, the system classifies the student into one of the following categories: beginner, intermediate, advanced (expert), i.e. it classifies the student according to a predefined stereotype. Learning session then proceeds in compliance

with the assigned stereotype until the completion of the first concept, when the test for the concept is conducted. Based on the test results, the Pedagogical Module updates the "actual skill level" attribute in the Student Model. The session then develops according to the value of this attribute.

Apart from the "actual skill level", other attributes of the student model are also taken into account, such as his/her learning style and desirable level of details. The values of the student model attributes are calculated by applying groups of rules and simple functions from the *Pedagogical Module* to the group of parameters which the system gets automatically and updates during each session. At the end of the session, the student model is recorded in an XML document and read in again at the beginning of the next session. At any time during the session, the student is allowed to check his performance, so that reflective learning can take place.

2.1 Cognitive Trait Model

The modelling of individual differences in cognitive processing is one of the areas where the full potential of student modelling has not yet been achieved [6].

CTM profiles the students' cognitive traits, which are innate abilities that are more or less persistent over time and independent of the domain. Working memory capacity and inductive reasoning ability are examples of cognitive traits. CTM could enable the learning environment to provide fine-grained adaptivity that takes each individual student's cognitive abilities and resources into account. Ideally, this approach may also enable predicting individual student's performance in a new task without new parameters, presumably after deriving an estimate of each student's processing parameter from previous modelling of other tasks. The CTM offers the role of a learning-companion that the student can consult and that can interact with different learning environments. Furthermore, due to the persistent nature of cognitive traits, CTM is particularly suitable for life-long learners. The combination of CTM and

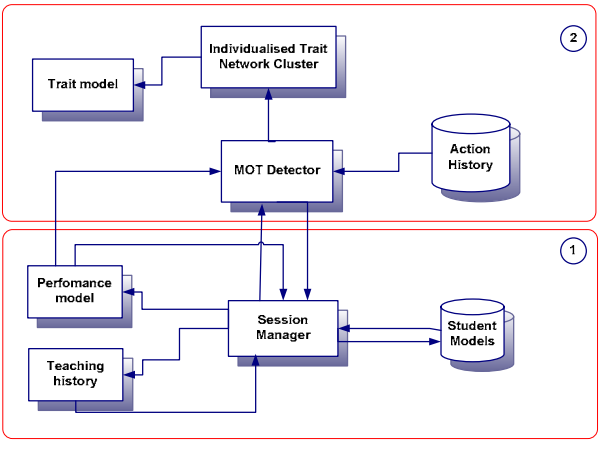

Fig. 1. Student model components: 1-Performance-based student model, 2-cognitive trait model

performance-based model allows DP-ITS to provide not only performance-based support but also support according to the student's cognitive capacity. The combined student model is shown in Figure 1.

Performance model and Teaching history are already described above, the other components are:

1. *Trait Model* – values representing the student's cognitive traits. The values are calculated from the interaction of the *Session Manager*, *MOT Detector*, and *Individualized Trait Network Cluster*.
2. *Session Manager* – manages the operation of all the other components and makes their coordination possible. It also enables reading-in the student model from the corresponding XML document at the beginning of the session writing out the student model to that XML document at the end of the learning session.
3. *Student models database* – The system creates a separate XML document for each student
4. *Action History* – The student's interactions with the system are interpreted as a series of his/her actions performed on learning objects.
5. *MOT Detector* – Various manifestations of traits (MOTs) are defined for each cognitive trait [4]. Each MOT is a piece of an interaction pattern that manifests a student's cognitive capacity. For example, Huai's experiment found that students who prefer linear navigation tend to have higher working memory capacity [2]. Therefore, the MOT of linear navigation can be used to manifest high working memory capacity. The *MOT Detector* encodes the knowledge of a number of MOTs and searches for those MOTs in a series of student's actions stored in the *Action History*. The result of the detection is the forwarded to the *Individualized Trait Network Cluster*.
6. *ITN Cluster* – The *Individualized Trait Network Cluster* in Figure 2 can have more than one individualized trait network (ITN). Each ITN is an instance of dichotomic node network [4] and represents a particular cognitive trait (e.g. working memory capacity) of the student. Each node in the ITN has a weight and corresponds to a MOT. Once a MOT is detected from the learner's actions, the corresponding node is activated, and only the activated node affects the execution of an ITN. The result of the execution determines how the nodes in the ITN should be updated. The results of the execution of all the ITNs are then saved in Trait Model. The deployment of the mechanism of dichotomic node networks ensures for an ITN to gradually grow to represent a cognitive trait of the student. When the student is using DP-ITS to learn, he/she is simultaneously training the ITNs.

3 Conclusions

DP-ITS is an intelligent tutoring system for learning about design patterns used in software engineering. It follows the ITS structure described in [1]. However, in addition to the performance-based student model, it also deploys cognitive trait model [4]. Performance-based model and cognitive trait model provide qualitatively different kinds of support to the students – both are very important in virtual learning environments. Performance-based adaptation guides the student to the required concept, whereas cognitive support serves to prevent the student's cognitive capacity

overload while still representing sufficient challenges to the student. Since initial evaluation of this synergistic approach to student modeling is encouraging, future work will include applying it to other ITS in order to gain more experience with deploying it in practical systems.

Acknowledgement

This research is partially supported by Online Learning Systems Ltd (NZ) in conjunction with the New Zealand Foundation for Research, Science & Technology, as well as by the ProLearn Network of Excellence project, funded by the Framework 6 IST (Information Society Technology) program of the European Commission dealing with technology enhanced professional learning.

References

1. Devedžić, V: Knowledge Modeling – State of the Art. Integrated Computer-Aided Engineering Vol. 8 No. 3 (2001) 257-281
2. Huai, H.: Cognitive style and memory capacity: effects of concept mapping as a learning method. Doctoral Thesis, Twente University, The Netherlands (2000)
3. Jeremić, Z., Devedžić, V.: Student Modeling in Design Pattern ITS. In Proc. of the 8th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Wellington, New Zealand (2004) 299-305
4. Lin, T., Kinshuk, Patel A.: Cognitive Trait Model - A Supplement to Performance Based Student Models. Proc. of International Conference on Computers in Education 2003, Hong Kong (2003) 629-632
5. Lin, T., Kinshuk: Dichotomic Node Network and Cognitive Trait Model. Proceedings of the 4th IEEE International Conference on Advanced Learning Technologies, Joensuu, Finland (2004) 702-704
6. Lovett, M. C., Daily, L. Z., Reder, L. M.: A Source Activation Theory of Working Memory: Cross-task Prediction of Performance in ACT-R. Journal of Cognitive System Research, Vol. 1 (2000) 99-118
7. Mitrovic, A., Ohlsson, S.: Evaluation of a Constraint-Based Tutor for a Database Language. International Journal on Artificial Intelligence in Education, Vol. 10 (1999) 238-256
8. Staff, C.: HyperContext: A Framework for Adaptive and Adaptable Hypertext. PhD Thesis, University of Sussex (2001)