Work in Progress: A Generic Model for Interactivity-Intense Intelligent Tutor Authoring Tools

Danilo L. Dalmon and Leônidas O. Brandão
Department of Computer Science, University of São Paulo
{ddalmon,leo}@ime.usp.br

Guilherme M. Gomes and Anarosa A. F. Brandão
Escola Politécnica, University of São Paulo
guilhermeg@usp.br, anarosa.brandao@poli.usp.br

Seiji Isotani
Institute of Mathematics and Computer Sciences, University of São Paulo
sisotani@icmc.usp.br

Abstract — Intelligent Tutoring Systems (ITS) provide many features that improve learning and teaching experiences. ITS are usually interactivity-intense and content-specific. Interactivity-intense assignments are recommended for scaffolding learning, while content-specific systems can offer low flexibility regarding its possible pedagogical approaches and its uses by teachers. In order to overcome this limited flexibility, there are systems which let content-specific interactivity aside to provide authoring tools, with which teachers can author intelligent tutored assignments without programming. The generic model proposed herein intends to address this problem providing flexible authoring tools for interactivity-intense assignments with tutoring features, letting teachers benefit from the flexibility of content authoring tools as well as the interactivity usually restricted to content-specific ITS. We introduce an application framework which implements this model, which is available as free software.

Keywords — Authoring Systems, Computer Assisted Instruction, Electronic Learning, Intelligent Tutoring Systems.

I. INTRODUCTION

Intelligent Tutoring Systems (ITS) are educational software that provide intelligent feedback in order to improve learning experiences for students [1]. ITS main benefits are, for instance, recognizing misconceptions, guiding them through the study, enhancing the speed of learning procedures and helping or preventing teachers from repetitive work.

In our point of view, there are two important aspects of educational systems, including ITS: interactivity and flexibility of use. In order to guide the use of interactive mechanisms on user interfaces of software systems, some conceptual frameworks were developed [2]. Using this approach, it is possible to highlight types of interactivity that enhance learning of specific domains, such as sciences and mathematics. We define interactivity-intense assignment as the one in which user interfaces adopt different types of interactivity in order to enhance learning. Flexibility consists of allowing teachers to adapt the system behavior in order to better suit his/her pedagogical approach [3].

Interactivity is an aspect well explored by ITS, which is not the case for flexibility of use. Most ITS provide assignments for students in a fixed set of problems or they are created automatically using templates. There are authoring tools that intend to facilitate the programming of tutored assignments [4]. Authoring tools for teachers are presented by example-tracing tutors [6]. By analyzing existing systems, we found no ITS with interactivity-intense assignments and authoring tools for teachers, but systems with one feature or other.

Our goal is to provide means to facilitate the development of ITS that present the two features together. This paper proposes a generic model of ITS in order to develop systems with interactivity-intense assignments and authoring tools, which we call Interactivity-Intense Intelligent Tutor Authoring Tools.

II. A MODEL FOR INTERACTIVITY-INTENSE INTELLIGENT TUTOR AUTHORING TOOLS

In this section, we present the model for developing educational software with functionalities of: (i) intelligent tutoring, (ii) interactive assignments and (iii) authoring tools for teachers. The model is presented as a generic system architecture with its description and the structure for each functionality it provides. By using it, one can develop an ITS with the desired features following a cookbook and reusing the proposed architecture. Fig. 1 presents this model with four components: structural, assignment, domain and tutoring.

The communication flow depicted in Fig. 1 works as follows: (a) structural component initializes other components; (b) use of domain-independent features by them; (c) assignment component integrates domain and tutoring actions to create and manage assignments; and (d) tutoring component reacts to domain actions. A system built with this model can be used in two ways, by teachers that author and edit assignments and students that solve these assignments. We describe the four components of the proposed architecture as follows.

The structural component has three goals: (i) providing basic structure to the system and (ii) to the domain-independent
features; and (iii) initializing and setting communication among other components. Its basic architecture is defined by its conditions of use and its final user, such as Web-based, mobile, or desktop applications. As these specific aspects of the system are essential to designing its architecture, the proposed model is restrained to define this component role.

The assignment component has three main goals: (i) modelling an assignment; (ii) allowing teachers to author assignments; and (iii) allowing students to solve assignments. This modeling must be respected by other components for compatibility. Fig. 2 shows how an assignment is structured: it has a proposition, a configuration, three states (current, initial and expected) composed of domain objects. Domain objects are elements specific to the educational concepts used by the system, such as points in geometry. The student starts an assignment defined by the initial state, interacts with the current state to solve it, and the system compares the solution to the expected state through automatic assessment.

The domain component provides means for the domain-specific features to be compatible with other features. Domain features are functionalities that are specific to the educational domain, such as creating a line in geometry. These features are the essence of the system interactivity-intense assignments. The internal structure of this component is shown in Fig. 3. It uses the Command pattern [6]. On the left, the graphical user interface (GUI) has a list of domain actions, which connect an assignment state and a list of domain objects (its parameters) to the domain model, the class responsible for executing the operation.

The tutoring component provides the intelligent features for the system. Its main goals are: (i) providing intelligent tutoring actions; (ii) managing the way these actions are being accessed; and (iii) allowing the definition of the tutor behavior. Its structure is depicted in Fig. 4. On the left, domain actions are observed by a tutoring sensor, which is associated with the intelligent module, in the same way as the help button. This module has a list of tutoring actions that are selected and executed depending on the assignment state and its behavior, defined by the authoring user interface on the right.

![Diagram](image1.png)

**Figure 2. The internal structure of assignment component.**

The domain component provides means for the domain-specific features to be compatible with other features. Domain features are functionalities that are specific to the educational domain, such as creating a line in geometry. These features are the essence of the system interactivity-intense assignments. The internal structure of this component is shown in Fig. 3. It uses the Command pattern [6]. On the left, the graphical user interface (GUI) has a list of domain actions, which connect an assignment state and a list of domain objects (its parameters) to the domain model, the class responsible for executing the operation.

![Diagram](image2.png)

**Figure 3. The internal structure for domain component.**

The tutoring component provides the intelligent features for the system. Its main goals are: (i) providing intelligent tutoring actions; (ii) managing the way these actions are being accessed; and (iii) allowing the definition of the tutor behavior. Its structure is depicted in Fig. 4. On the left, domain actions are observed by a tutoring sensor, which is associated with the intelligent module, in the same way as the help button. This module has a list of tutoring actions that are selected and executed depending on the assignment state and its behavior, defined by the authoring user interface on the right.

![Diagram](image3.png)

**Figure 4. The internal structure of the tutoring component.**

**III. AN APPLICATION FRAMEWORK IMPLEMENTING THE MODEL AND CONCLUSIONS**

After the creation of the model, we developed an application framework for developing Interactive Learning Modules (iLM) [7]. This framework uses a component architecture based on the four components of the proposed model. Currently, the tutoring features are being implemented. The development of iLM using it has resulted in the refactoring process of four existing systems, which facilitated this process. Future work includes adding more features to the framework according to the proposed model and using it to create other systems. This will also serve to evaluate its impact on the development tasks.

The main contribution of this paper is considering the flexibility of use and interactivity features in designing educational systems in general and ITS in particular. Also, by providing abstract reusable concepts and architectures such as the proposed model, the development effort and time of these systems are reduced. The model presented centralizes many concepts and design decisions that serve as a basis for developing educational systems such that the design effort can be focused on the instructional dimension.

**ACKNOWLEDGMENTS**

Danilo L. Dalmon is supported by FAPESP under grant 2010/06805-2. This work was partially supported by FAPESP (2011/10926-2) and CNPq (550449/2011-6).

**REFERENCES**


