A PEDAGOGICAL FRAMEWORK TO INTEGRATE LEARNING STYLE INTO INTELLIGENT TUTORING SYSTEMS

Shahida M. Parvez and Glenn D. Blank
Computer Science and Engineering Department, Lehigh University
smp9@lehigh.edu, gdb0@lehigh.edu

ABSTRACT
Educational systems can be more adaptive and effective when they incorporate learning style models. Developing a learning style based system is not a trivial task and presents challenges such as selecting the appropriate learning style model and instrument, creating course content consistent with the various learning styles, and determining the level and degree of adaptation of domain content. We present a pedagogical framework that makes it possible to integrate learning style into an intelligent tutoring system. This domain independent framework is based on the Felder-Silverman learning style model and has been designed in the context of CIMEL-ITS, which helps students learn object-oriented design using UML.

1. MOTIVATION
One instructional environment does not fit all learners (Reigeluth, 1996). People have different learning styles as they take in and process information differently (Felder, 1996). Students learn more effectively when the instruction is matched to their individual learning styles (Rasmussen 1998).

Adaptive educational systems attempt to maintain a learning style profile for each student and use this profile to adapt the presentation and navigation of instructional content to each student. A number of these adaptive educational systems have been developed: CS383 (Carver et al., 1999), Arthur (Gilbert and Han, 1999), iWeaver (Wolf 2002), EDUCE (Kelly, D. & Tangney, B. 2002). Developing e-learning systems that adapt to student learning style is not a trivial task. Design and development challenges include selecting the appropriate learning style model and instrument, creating course content consistent with the various learning styles, and determining the level and degree
of adaptation of domain content. There is a need for a generic pedagogical methodology or framework that would allow easy integration of learning styles in an e-learning system.

We present a novel pedagogical framework that facilitates integrating the Felder-Silverman Learning Style Theory (Felder and Silverman, 1988; Felder, 1993) in an intelligent tutoring system (ITS). Designed to be domain-independent, this pedagogical framework is being developed in the context of CIMEL-ITS, an ITS for novices learning object-oriented design using UML and Java—a complex and open-ended problem solving task for novice learners. This paper will describe in detail how the Felder-Silverman learning style theory is used to create pedagogical feedback and how this feedback is dynamically assembled for a given learning style.

2. FELDER-SILVERMAN LEARNING STYLE MODEL

The Felder-Silverman learning style model categorizes a student’s learning style on a sliding scale of four dimensions: sensing-intuitive, visual-verbal, active-reflective and sequential-global. This model is accompanied by the Felder-Solomon Index of Learning Style (ILS) instrument which is used to categorize individual learning style preferences. The ILS is a questionnaire containing 44 questions, 11 questions corresponding to each of the four dimensions of the learning style model. The range of data for each dimension is from 0 to 11. This means that the learner’s preference on a given scale does not necessarily belong to only one of the poles. It may be strong, mild, or almost non-existent. We chose this particular model for the following reasons:

1. The ILS has been validated (Zywno 2003, Felder & Spurlin 2005).
2. The ILS questionnaire (Felder and Soloman, 2003) provides a convenient and practical approach to establish the preferred learning style of each student. It is simple, easy to use and the results are easy to interpret.
3. The number of dimensions of the model is constrained, improving the feasibility of its implementation.
4. The results of ILS can be linked easily to adaptive environments (Paredes and Rodriguez, 2002).
5. It is most appropriate and feasible to be implemented for hypermedia courseware (Carver and Howard, 1999).

3. CIMEL-ITS

CIMEL-ITS is an intelligent tutoring system that provides one-on-one tutoring to help beginners in a CS1 course learn object-oriented analysis and design, using elements of UML (Blank 2005). CIMEL-ITS is based on “design-first” curriculum that introduces object-oriented design using elements of UML before coding (Moritz & Blank 2005). This curriculum enables the students to understand and comprehend the problem without getting bogged down with programming language syntax.

CIMEL-ITS is composed of four distinct components. The Curriculum Information Network (CIN) consists of domain knowledge which is object-oriented design concepts. These concepts are linked together through various relationships such as prerequisite and
equivalence and assigned a measure of learning difficulty. Expert evaluator is the second component of CIMEL-ITS and it interfaces with a student through the Eclipse Integrated Development Environment (IDE) plug-in. As the student designs a solution for a given problem in the plug-in environment, EE evaluates each step of student solution in the background by comparing it with its own solution and generates a data packet for correct student action and an error packet for incorrect action. These packets are then sent to the third component of the ITS which is the student model (SM). The student model analyzes these packets to determine the knowledge level of the student for each concept and attempts to find reasons for student errors (Wei 2005). The SM uses this information to update the student model and passes the original packets along with its own analysis to the fourth component of CIMEL-ITS which is the pedagogical agent (PA).

The PA can function in two different modes: interactive or tutorial. The interactive mode is a homework helper mode where the student is working on his/her problem design and the PA is watching in the background. If student seems to be struggling, the PA will attempt to help the student by giving hints that will prompt the student to think about his problem solving action leading to a resolution. The tutorial mode provides the student with more instruction about a given domain concept. The student can access this mode either from the CIMEL-ITS pull down menu or through the interactive advice window that pops up when student is working on a design.

4. PEDAGOGICAL FRAMEWORK
4.1 Feedback Architecture

The Felder-Silverman Learning Style Model categorizes a student’s learning style on a sliding scale of four dimensions: sensing-intuitive, visual-verbal, active-reflective and sequential-global. The following table lists appropriate feedback type for each dimension, providing a guide to create the feedback architecture.

<table>
<thead>
<tr>
<th>Learning Style Dimensions</th>
<th>Feedback Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active</td>
<td>Hands on, likes to work in a group</td>
</tr>
<tr>
<td>Reflective</td>
<td>Likes to think before trying, works alone</td>
</tr>
<tr>
<td>Verbal</td>
<td>Words, written / spoken</td>
</tr>
<tr>
<td>Visual</td>
<td>Diagrams, maps, flow charts</td>
</tr>
<tr>
<td>Sensor</td>
<td>Facts, concrete, procedures, practical, conceptual</td>
</tr>
<tr>
<td>Intuitive</td>
<td>Innovative, theories, meanings, relationships</td>
</tr>
<tr>
<td>Global</td>
<td>Big Picture, large steps</td>
</tr>
<tr>
<td>Sequential</td>
<td>Orderly, linear, incremental</td>
</tr>
</tbody>
</table>

Table (2) – Felder-Silverman model dimensions / learning preferences

The feedback architecture will consist of the following components that will contain information in the form that is suitable for all these dimensions:

1. Definition – This component contains definitions of domain concepts and will be used in introducing a concept. It is useful for many learning style dimensions,
including verbal, sensor, intuitive, etc. An example would be “Attributes are characteristics of an object that persist through the life of that object.”

2. Example – This component contains examples that can illustrate a given concept. It can be used for almost any learning style, especially the sensor style which prefers a practical approach to concepts. An example of feedback in this component might be “Attributes of a car might be its color, model, make, etc.”

3. Question – This component contains questions that could serve as hints during the interactive mode. It is very useful in making the learner think about his problem solving action and very important for “reflective” type learner as it gently nudges him to reflect on his action. An example of the feedback in this component might be “Are you sure you want to set the data type of money as a string?”.

4. Suggestion – The purpose of this component is to give a suggestion to a learner who might be lost. It helps in pointing the student in the right direction. Many times it is not enough to tell a novice that his action is incorrect but one should also guide him as where to go to find the right answer. An example of this component would be something like “It is a good idea to use the tutorial to learn about ‘passing data to a method’”.

5. Picture – This component contains images that illustrate a concept for the “visual” learner. For example, when teaching concept of data type, one could create an image consisting of transparent containers marked as int, long, double, string, etc. These containers could have things such as a dollar sign in the double container, age in the int container, name in the string container, etc.

6. Relationship – This component contains information that provides the relationship of a given concept to the big picture. Often learners understand a concept but have a difficult time connecting it to the bigger picture. This component is mainly useful for “global” learners who like to know the big picture first.

7. Facts – This component contains facts about a concept that extends beyond the concept definition. It is very useful in deterring misconceptions about a given concept. An example of this component might be ‘The actual data returned by a method must match the return type of the method’. Mostly suitable for sensor learner but can also be used for other learner types.

Each of these components has the following attributes that are used by the assembly algorithm when assembling feedback for a given student problem solving action:

- Concept – the concept the information applies to
- Level – numeric feedback level
- Type – component type
- Category – the category that this component belongs to
- Actual text – the actual text of the feedback
- Relevance_factor – the applicability value of the component
- Compatible_comp – other components that this component can be combined with to generate the feedback.
- Mode – the mode the component is applicable to.
4.2 Assembly Algorithm

The feedback is a function of eight different elements: concept, action, error, student_model, learning_style, feedback_history, current_problem, mode:

Feedback : \( f(\text{concept, action, error, student_model, learning_style, feedback_history, current_problem, mode}) \)

1. **concept** – The concept that the student needs help with.
2. **action** – The incorrect problem solving action performed by the student.
3. **error** – Error that the student made while performing the above action.
4. **student_model** – The student model information such as the understanding level for each concept / prerequisite.
5. **learning style** – The learning style of the student as determined by the index of learning style that each student fills out the first time he/she logs into the ITS.
6. **feedback_history** – A detailed history of feedback provided to the student for this concept.
7. **current_problem** – The current problem the student is working on.
8. **mode** – Interactive / tutorial

The combined effect of these seven variables results in generation of feedback.

4.3 Concrete Example

Students are asked to design a simple movie ticket vending machine that sells tickets for only one movie and one show time each day. The customers can see the name of the movie, the time, and the ticket price displayed on the machine. There is a slot to insert money, a keypad of buttons to enter a number into the "Number of Tickets" field, and a "Buy" button. Printed tickets come out of a slot at the bottom of the machine.

The student creates a class diagram for the class “TicketMachine” in the Eclipse UML plug-in. Then he enters an attribute “numberofticketsrequested.” The EE diagnoses an error, because it is not an attribute of the TicketMachine. The EE generates an error packet and passes it along to the Student Model which updates the student profile with this information and passes this packet to the PA.

The PA takes concept, action, and error from the error packet, to which the SM appends information such as how well the student understands the concept and prerequisite concepts. Then the PA retrieves the student learning style information which is the result of the index of learning styles questionnaire that the student fills out the first time he logs into the system. The PA also retrieves the history of feedback provided to this student regarding the current concept or any of its prerequisite concepts. The PA sets the mode to “interactive” by default.

Let’s assume the values of variable for this example are as follows:

<table>
<thead>
<tr>
<th>concept</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>action</td>
<td>add attribute</td>
</tr>
<tr>
<td>error</td>
<td>unknown attribute</td>
</tr>
<tr>
<td>student_model</td>
<td>(object 90%, attribute 60%, data type 40%)</td>
</tr>
<tr>
<td>learning_style</td>
<td>(verbal, sensing)</td>
</tr>
<tr>
<td>feedback_history</td>
<td>None</td>
</tr>
</tbody>
</table>
The first thing the feedback assembly algorithm does is to look at the compatibility matrix of learning style / feedback component using student learning style to retrieve applicable feedback component type. Next, it creates a list of these learning style relevant components for the concept “Attribute” and “data type” with a mode “interactive.” In this case, since there is no feedback history, feedback components of all levels are considered relevant. If the student had a prior feedback history, then the algorithm will compute the “Relevance_factor” for each component that had been used before. Since the action “add_attribute” and error “unknown attribute” both indicate that the student is having difficulty identifying an attribute at this time, the feedback components for “data type” are put on a reserve list which is used to provide feedback if the student invokes the tutorial mode.

Since the student has not received any feedback for this concept before and the probability that he knows this concept is only 60%, the algorithm will choose the feedback components with the level 5 which is the highest and most detailed level. The feedback component for “Attribute” for level 1 is just a hint. For example, it might just say “Remember, an attribute is a characteristic of an object”. The actual feedback that the student will receive will be

“You added numberofticketsrequested as an attribute of the TicketMachine. Remember Attribute is a characteristic of an object that must remain for the life of the object. In other words, attributes are data that has to be maintained as long as the object exists. For example, if a car is our object, color, model, make are its attributes. Does the TicketMachine need to keep track of the number of tickets requested by each customer? It might be a good idea to learn more about Attributes through the tutorial, click on help.”

The algorithm used five different components to generate this feedback, the definition, fact, example, question and suggestion. If the student was a visual learner, the example of a car would be in the form of a picture or image. If the student was an active learner, an interactive multimedia exercise involving identifying attributes of an object would be part of the feedback.

If the student clicks on the “help” button, he will be presented with a more detailed description of an attribute along with other examples. Lessons on all relevant concepts such as “data type” will also be included in the tutorial.

5. FUTURE WORK

Currently, this feedback architecture is being implemented in CIMEL-ITS which currently provides the student feedback but without this complete pedagogical framework. CIMEL-ITS will be evaluated with this pedagogical framework during summer / fall 2006. A user interface allowing an instructor to add feedback to this framework will also be developed. This pedagogical framework will be ported and feedback will be created for a popular math puzzle game “24” (see firstinmath.com).
6. REFERENCES


