PASTEL: pattern-driven adaptive simulations

Mark K. Singley, Peter Fairweather, Tracee Wolf, and Dick Lam

IBM T.J. Watson Research Center, 19 Skyline Drive,
Hawthorne, New York, USA
{ksingley, pfairwea, tlwolf, rblam}@us.ibm.com

Abstract. We propose a new kind of learning environment called an adaptive simulation that more deeply explores and exploits the potential of simulations as pedagogical and explanatory tools. In an adaptive simulation, the simulation configuration is not fixed but rather can be modified by an instructional agent for optimal pedagogical effect. Types of adaptations include manipulations of simulation time and state, changes in representation to facilitate explanations and/or task performance, and adjustments in simulation complexity by the addition and/or removal of components. We briefly describe a system we are developing called PASTEL that is designed to enable these kinds of adaptations. Open research issues include precisely how to perform these adaptations and when to employ them for optimal effect.

Keywords: adaptive user interface, simulation, HCI patterns, systems thinking

1 Simulations as Learning Environments: Unrealized Potential?

There is a long history of interest in simulations as learning environments, primarily because of their ability to support conceptual exploration and learning-by-doing in authentic, risk-free, immersive settings. In the face of mounting global challenges requiring deep insight into the complexities of economic, biological, and social systems, interest in simulations as exploratory conceptual tools only appears to be growing [1]. Although widely regarded as having great promise, simulations can have a number of associated learning problems, primarily having to do with learner understanding, management of system complexity, and the lack of built-in instructional support [2]. In our work, we are striving for a tight linkage between the design of simulations and the design of associated instructional components. Specifically, we are considering what value may result if the simulation itself is put under much more direct and masterful control of an instructional agent.

2 Radical Adaptation

In developing our vision of adaptive simulations that can reconfigure themselves radically in service of instructional goals, we are investigating three broad categories of adaptation:

2.1 Control of Simulation Time and State

The instructional agent should be able to arbitrarily set the simulation's time and state for its own pedagogical purposes. (This goes beyond the standard simulation tactic of time compression in order to achieve future states faster than real time). We are envisioning a system that can "rewind" the simulation to an earlier critical juncture and either step through what happened with the learner as a form of review or to give the learner another opportunity to practice with either the same simulation state first encountered or a similar one. Another potential use of this capability would be to provide episodic summaries of learner behavior to an interested third party such as a teacher, mentor, or fellow learner cast in the role of critic.

2.2 Adjustment of Simulation View and/or Representation

In order to promote perceptions of authenticity as well as maximize the overlap between learning and performance environments, simulations often strive for a high-fidelity representation of the system and/or setting being modeled. However, such realistic representations might not always be best for instructional purposes. In some circumstances, it may be beneficial to expose the hidden workings of a device or the internal state of a process. Additionally, it may be beneficial to strip away what might be considered the extraneous detail of a complex situation and focus on the truly important and meaningful elements. Such representations that expose hidden elements and remove extraneous details might be termed *schematic*. The tutor should be able to change the outward view or representation of the simulation, moving along the realistic/schematic continuum to suit the current pedagogical goal.

Aside from swapping in more schematic representations of the simulation model, we are also exploring the use of scaffolded representations of the interface to provide transitional support during all phases of learner behavior. Here are some brief examples: (**a) Perceiving:** In complex situations where there are multiple information streams whose monitoring and integration is critical to understanding, a useful simplification might be to reconfigure the interface so that these sources are placed together in a prominent position. **(b) Acting:** Learner action can be scaffolded by activating only task-appropriate controls, and conveying this action space through highlighting, shading, or some other graphical technique.

2.3 Selection of Simulation Components

Finally, aside from just changing the outward representation, the instructional agent should be able to change the underlying deep structure of the simulation as well. By adding and/or removing components, the instructional agent should be able to adjust simulation complexity and focus attention on current instructional goals. The challenge here is to author the simulation in such a way as to enable this kind of decomposition and modular assembly on the fly. This is the focus of much current work on simulation in the learning objects community.

All of these factors combine into what might be called a *scaffolding gradient*, a multi-dimensional space of possibilities that can be navigated by the instructional

agent for optimal pedagogical effect. Just how, when, and where to navigate in this space is a subject of active research interest.

3 Overview of PASTEL System Components

Simulation Model. A key feature of our approach is that we make use of semantically-rich, domain-general patterns to build our simulation models. These patterns, which are better characterized as schemas or frames [3], are declarative abstractions that specify common, recurring inter-related roles and/or behaviors. An example of a schema is the SUPPLY schema, which is quite general and can be used to describe processes in many domains. A listing of the SUPPLY schema's roles and their instantiations for a simple flashlight example is shown in Table 1.

Table 1. SUPPLY schema roles and example instantiations.

Role	Definition	Flashlight Example
Producer	produces the provision for the consumer	battery
Consumer	consumes the provision	bulb
Provision	that which is supplied	electricity
Control	enables or controls the process	switch
Indicator	reveals the state of the process	bulb
Path	path by which the provision gets to the consumer	CIRCUIT schema

To build a simulation model, the user selects schemas from the PASTEL schema library (which has been stocked by mining such sources as FrameNet [4]) and instantiates their roles with elements drawn from the system being modeled. To describe a complex system, schemas have to be composed both "horizontally" and "vertically." A horizontal composition involves having a single system element fill multiple roles across multiple schemas (a system element may also fill multiple roles within the same schema, as shown by the bulb element in Table 1). A vertical composition involves having a role within a schema instantiated by another schema (again shown in Table 1, where the *Path* role is instantiated by a CIRCUIT schema). Providing these multiple, hierarchical, schematic descriptions of the system provides the leverage necessary for many of the adaptations described earlier. The schematic approach is also valuable pedagogically because it encourages the modeler to describe the system using general semantic frames. These frames activate general knowledge learners already have and use to understand the world.

User Interface. Every instantiated schema role element defined in the simulation model potentially has a representation in the user interface. Some elements (those that are typically visible in a realistic depiction of the system) will be displayed in an authentic view, whereas others (those that are either ordinarily hidden in a realistic depiction or represent abstract, latent variables that typically have no physical manifestation) may be revealed in more schematic views. It is often these hidden elements that, once exposed or reified, provide true insight into the operation of a complex system.

The creation of the user interface is coordinated semi-automatically with the creation of the simulation model. Those elements cast as *Controls* in the simulation model are initialized as actual user controls in the interface; those elements cast as *Indicators* become dynamic elements of the interface that are updated in response to changes in the internal states of the simulation. Those elements that are neither controls nor indicators become static elements in the interface.

Instructional Agent. The purpose of the instructional agent is to project different views of the simulation model onto the interface, in accordance with pedagogical goals either stated by the user (e.g. "tell me more about this element" or "explain how I got into this state") or generated by some other agent, such as an intelligent tutor. Thus, the simulation model, interface, and instructional agent can be regarded as an instance of the traditional model-view-controller pattern.

The instructional agent uses the schematic descriptions of the simulation model in combination with its own pedagogical heuristics to decide what view to project. Its operations are organized into three levels (cf. [5]) :

Operators. Each element of the interface can respond to the following basic commands, which when applied individually and in combination form the basis for realizing many of the adaptations: create/destroy, attach graphic(s), move/stop, enlarge/shrink, hide/show, dissolve/emerge, blur/glow, raise/lower, compose.

Tactics. The above operators are combined into pedagogical tactics such as *highlight* (e.g. move nearby elements aside, move and enlarge element of interest), *reveal* (dissolve occluding element and apply glow to element of interest), or *categorize* (e.g. move similar elements to the same location). There may be many methods defined within each tactic that accomplish the same goal but involve different combinations of operators.

Strategies. Finally, tactics are combined to form pedagogical strategies, such as *give overview* (traverse upper level of a functional schematic description of the system and highlight elements in turn), *disassemble* (recursively unpack elements of a spatial schematic description), and *reify hidden state* (traverse functional schematic description and reveal hidden elements).

References

1. Thompson, C.: Saving the World, One Video Game at a Time. New York Times, July 23 (2006).
2. Hayes, R., Singer, M.: Simulation Fidelity in Training System Design. Springer-Verlag, Berlin (1989)
3. Barsalou, L.: Frames, Concepts, and Conceptual Fields. In: Lehrer, A., Kittay, E. (eds.): Frames, Fields, and Contrasts: New Essays in Semantic and Lexical Organization. Erlbaum, Hillsdale, NJ (1992)
4. FrameNet website: http://framenet.icsi.berkeley.edu
5. Seligmann, D., Feiner, S.: Automated Generation of Intent-Based 3D Illustrations. SIGGRAPH Computer Graphics. 25(4) (1991) 123-132