Work-in-Progress: Collaborative and Reflective Learning in Engineering Programs

Neelam Soundarajan
Ohio State University, neelam@cse.ohio-state.edu

ABSTRACT

Learning objects (LOs) were proposed as a powerful approach to building reusable chunks of educational content. But in spite of the considerable effort that has been invested by various groups, the approach has had relatively little impact on education. We argue that this is because of a fundamental weakness in the way LOs have been conceived of thus far. We show how the problem can be addressed by incorporating, into the notion of LOs, key ideas from object-oriented (OO) programming. We show that the resulting approach will not only help students in learning particular concepts but also help develop their reflective and teamworking skills. The approach is also consistent with the ideas of the *how people learn* framework.

1. INTRODUCTION

Over the years, a number of approaches to using computers as powerful tools in education, ranging from *automated tutoring systems*, to systems based on *constructionist* approaches, to systems of *multimedia learning*, etc. have been proposed. One of the most interesting ideas is that of *learning objects* (LOs). While there are several competing definitions, the consensus definition is the one offered by Wiley [2]: "[t]he main idea of LOs is to break educational content into small chunks that can be reused in various learning environments, in the spirit of OO programming". But, in spite of considerable effort by various groups, the approach has had relatively little impact on education. Our work is motivated by the observation of a fundamental weakness in the way that LOs have been conceived of thus far that seriously impacts their usefulness, especially for higher education. The problem may be less significant in corporate training which may be why LOs have found wide acceptance in that setting but few university curricula use them to any extent. In the next section, we will summarize the weakness in the current conception of LOs and present a new conception that resolves the problem. One of the important developments over the last few years is the *How People Learn* (HPL) framework [1]. HPL has identified essential features that effective learning environments should possess. According to HPL, the environment should be designed to be: *learner-centered*, i.e., account for the knowledge, skills, and preconceptions of learners; *knowledge-centered*, i.e., help students organize their knowledge around key concepts; *reflection-centered*, i.e., help students be aware of their own understanding and gaps therein; and *community-centered*, i.e., encourage students to learn from one another. So far, knowledge-centered issues have been the primary focus of most LO-based systems. As we will see, our innovative reconception of LOs allows us to naturally addresses the other essential features that HPL identifies.

2. A NEW MODEL OF LEARNING OBJECTS

Before considering the details of the problem with the existing models of LOs, it may be useful to recall some important aspects of *objects* as used in OO programming. In languages such as *Java*, an *object* is an *instance* of a *class* which is a template that specifies what information instances or objects of the class will contain; and defines a set of operations, or *methods*, that can be performed on these objects. While different objects of a class share its methods, each has its own *state*, and this state can vary over time. When a method is applied on an object, that may change the state of *that* object, but will not change the state of any other object. Consider an example. A common use of OOP is to build software simulations. Consider the simulation of a modern office. One of the *classes* in the simulation might be CopyMachine. When the simulation runs, *multiple* instances of the class would be created, each representing one of the copy-machines in the office. Suppose there are three such machines, M1, M2, and M3; when the simulation runs, three corresponding instances, I1, I2, and I3, of the class would be created. The information in, say, I2 would simulate the state of M2, such as the amount of paper left in M2, the number of copies made on it, etc. Methods defined in CopyMachine can be applied on I2. Thus applying makeCopies(10) on I2 would correspond to making ten copies on M2 and would update information about the no. of copies made, the state of the toner cartridge, etc., maintained in the state of I2. A richer version of the class could save, in I2's state, historical information about the activities on it rather than just its current condition and use this for diagnosing problems etc.

Learning objects, as currently formulated, are not like this. LOs do have something like methods but one does not think in terms of the "state" of an LO as changing with time. One

might ask, so what? If an LO defined, for example, a collection of methods designed to take a student through a set of lessons over, say, *recursion* in a *computer science* course, and contained some methods that can be used to assess how well the student had learned the topic, why should such an object change over time? The answer is that if we had a *separate* object for *each* student who goes through a particular lesson, then the state of the particular object would, if it could change with time, be a reflection of *that* student's progress through the lesson. So, for example, if a student went through part of the lesson in one session and then came back to it at a later point, the system would automatically know, by examining the state of the object associated with that student, not only where she had left off but also what difficulties she encountered when going through the earlier parts of the lesson. Hence the system could determine where the student should start, which parts of the lesson should be repeated, etc. Further, by providing, to the *student*, access to this same information, the system would enable her to reflect on her prior experiences with the activity. By contrast, existing LO-based approaches are *static* and, by ignoring the changes in the learner as she engages in the learning, in effect seem to ignore the learner!

3. REFLECTIVE, COLLABORATIVE LEARNING

Let us now consider how the new model of LOs will work in practice. Consider a student S in a course C that includes three specific topics/learning activities, A, B, C. Corresponding to these, we will have three learning *classes* La, Lb, Lc. At the start of the course, three objects, las, lbs, lcs, will be created for S. Each will contain an *empty record* since S has not yet engaged in any activity. As S engages in an activity, say, Lb, lbs will be updated to record information about this engagement. A minimal record would be the *grade* S receives in Lb; but it can contain much more. For example, suppose Lb requires students to implement an *algorithm* and test it. Students often fail to test such implementations adequately and the instructor's goal may be to help students understand the importance of adequate testing. So she may design Lb in such a way that lbs will record S's code, the test suite S used to test it, and the instructor's feedback on its adequacy. When S engages in Lb again, the system will have this information available. This will enable the system, depending on the facilities built into it, to guide the student into repeating some of the previous work, or going through the instructor feedback, etc.

In addition, we will associate a *learning history* with S. This will be a record of all the learning objects associated with S as she goes through the curriculm. When S starts out in the curriculum, we create her learning history hs, initialized to the *empty sequence* since, at this point, S would not have engaged in any activities. For each academic term t, we will append to hs an element ats recording S's learning activities during t. Suppose, in term t, S enrolls in courses C1, C2, C3. ats will be the set {c1s, c2s, c3s}, each representing the set of S's activities in that course. For example, if C3 is the course discussed in the preceding paragraph, c3s will be the set {las, lbs, lcs}. Suppose next that in a later semester, S takes a course C4 in which *recursion* is a key topic. Suppose the instructor for C4, on the basis of prior experience, knows that students often overlook subtle bugs in their recursive programs that they would have noticed had they tested them adequately. In designing the learning class Ld corresponding to this topic, the instructor can, depending on the facilities built into the system, design the class so that it provides suitable links to the topic Lb from the prerequisite course C2. When S engages in the activities that are part of Ld, the system will automatically link her to her previous work, as recorded in the object lds and saved on her history hs. This will allow the student to review the topic of adequate testing from the earlier course C3, go through the feedback she received from C3's instructor on that topic, etc. This is precisely what *reflective* learning should be: provide, at the most appropriate times, easy access to *relevant* information concerning a student's prior activities on *related* topics so that the lessons from those prior experiences are reiterated and applied in the context of the current activity. And, as just described, our approach enables us to do this *across* the curriculum, not just within the context of one course.

Further, the *course roster* for a course C will be the collection of learning histories of all students in C. This will allow C's instructor to have a good feel for the prior learning experiences of her students, both with respect to earlier activities in C as well as related activities in *earlier* courses and allow her to tailor C's activities appropriately. Moreover, by providing selective access to students engaged in a collaborative activity to each others' learning histories, the instructor will enable collaborative learning.

To summarize, our approach enables, as do other LO-based approaches, an instructor to account for knowledge-centered considerations by suitably designing the learning class corresponding to a given topic. Further, associating, with each student engaging in the activities included in the learning class, a learning object whose state can keep track of the student's engagement with the activities and the notions of learning histories and collecting these histories in course rosters allows us to account also for the reflection-centered and community-centered considerations of HPL. The key to realizing these potential benefits is building a system that provides the *right facilities* that are also easy to use: For instructors, the ability to link activities in a given course to related prior activities in the same course and in earlier courses; and the ability to allow students controlled access to each others' learning histories to permit students to learn from each other. For students, the ability to easily access and browse the relevant portions of their prior learning history as well as appropriate portions of their team-mates (as determined by the instructor). If the paper is accepted at FIE, the discussions during and after the presentation will greatly help in clarifying some of these details.

4. REFERENCES

[1] J. Bransford, A. Brown, and R. Cocking. *How people learn.* Nat. Acad. Press, 2000.

[2] D. Wiley. Connecting learning obj. to instructional des. th. In *Inst. use of learning obj.*, pages 3–23. 2002.