Using Problems to Learn Service-oriented Computing

Sandeep Purao  
Pennsylvania State University  
spurao@ist.psu.edu

Vijay Vaishnavi  
Georgia State University  
vvaishnavi@cis.gsu.edu

John Bagby  
Pennsylvania State University  
jbagby@ist.psu.edu

Faye Borthick  
Georgia State University  
borthick@gsu.edu

Brian Cameron  
Pennsylvania State University  
brameron@ist.psu.edu

Lisa Lenze  
Steve Sawyer  
Hoi Suen  
Pennsylvania State University  
llenze@ist.psu.edu  
sawyer@ist.psu.edu  
hoisuen@psu.edu

Richard Welke  
Georgia State University  
rwelke@cis.gsu.edu

Abstract

Service-oriented computing and the ensuing science of services represent significant challenges to academia. As we come to grips with its many implications, we are slowly beginning to realize the challenges of introducing service-orientation in research. The change also requires a rethinking of strategies used for educating computing professionals. Service-orientation questions the traditional vision of the IT professional as a toolsmith. Instead, it requires shifting the role of the IT professional to that of a participant in a multi-disciplinary team of diverse professionals. We describe one specific strategy that is a core part of an ongoing experiment to support such pedagogical practices, and reflect on its usefulness and limitations.

1. Introduction

A number of characteristics of building service-oriented solutions require a rethinking of strategies used for educating computing professionals. While the traditional vision of the IT professional as a toolsmith has served computing curricula and pedagogy for the last few decades [5]; the role of the IT professional itself is now changing – from that of a toolsmith to that of a participant in a multi-disciplinary team of diverse professionals [10]. A successful IT professional must be self-directed, participate effectively in a team, be aware of information technology standards necessary for loose-coupling and interoperability [8], and be able to work at the boundary between information technology and work practice. These requirements stem largely from ideas such as co-creation of value [15] that underlie a science of services and technological platforms that implement these ideas. They also demand multiple epistemologies for learning [16], and require a significant rethinking of strategies used for educating computing professionals.

In this paper, we report on an ongoing effort where we are engaged in introducing, adapting and evaluating a number of approaches to revitalize our pedagogical practices – with a focus on educating computing professionals who will learn about and contribute to the emerging science of services. The paper outlines our efforts, starting with the multiple epistemologies we draw on, argues for and describes one core technique – problem-based learning – and discusses challenges associated with its adaptation and evaluation.

2. Multiple Epistemologies

Realizing the vision of a computing professional as a participant in a team of professionals requires an investigation into theories of learning that must contribute a multitude of perspectives to the pedagogical effort. A significant dichotomy is presented by two groups of theories beyond traditional teaching efforts that follow the stimulus-response mode [21], often rendered as the lecture method [14].

The first group of theories, called the constructivist theories [7, 17], assumes that each individual constructs his or her own reality of the objective world [14] that the educator facilitates. It interprets learning as an active process in which students construct new ideas or concepts, while engaged in new experiences,
based on their current and past knowledge [7]. Learning is seen as allowing the students to form their own abstract concepts, not feeding these externally. The second group of theories, called the social/cultural theories, (e.g. [12]) does now see the goal of learning as the formation of abstract concepts. Instead, it argues that knowledge cannot be divorced from the historical and cultural background of students [14]. An important theory in this group is situated learning [12], which argues that learning, as it normally occurs, is a function of the activity, context and culture in which it occurs. Students become involved in a 'community-of-practice', where they progress from beginner to expert. Lave & Wenger [13] call this process, ‘legitimate, peripheral participation,’ which they argue is unintentional rather than deliberate. The knowledge constructed is thus local, specific to a context, i.e. situated, and appropriate for the individual.

Historically, efficient pedagogy has been interpreted as careful and interesting presentation (primarily lectures) that enables students to absorb large knowledge bases (see, e.g. SSME-related content at IBM). These dominant, historical biases are at odds with the active pedagogy and multiple epistemologies suggested by the two theoretical groups. Nilsen and Purao [16] characterize the challenge as one that requires a balancing between (a) presenting material that students are expected to absorb, and (b) engineering experiences that promote learning of underlying principles and application to problems. Results from the use of active, problem-based learning in other domains suggests that such acquisition of rote knowledge must be maintained, and be enhanced by approaches that lead to development of expertise [20]. Our approach, therefore, includes a number of elements. In this paper, we focus on one of these: problem-based learning [18].

3. Problems as the Basis of Learning

Problem-based learning (PBL) is the simple idea that problems come before answers [1]. Instead of providing answers that students are expected to memorize, students explore what they already know and their hunches to solve a problem. In the process, they find out what they need to know, and can effectively question instructors to learn [18]. PBL involves talking: asserting, defending, and criticizing ideas, where students unlearn and learn to solve the problem (see figure 1).

An effective problem-based learning cycle, thus, begins with a well-scoped problem that the students are encouraged to explore, followed by their articulation of what they need to know, moving to provision of resources needed by the instructor, the student(s) solving the problem, and finally the instructor providing feedback by either commenting on the partial solutions developed by the students or sharing with them an idealized solutions and leading a discussion. The elements included in the discussion would be: the Problem, Proposed Solution(s), Supporting Evidence, and Conclusions.

Prior research shows that learning following a PBL strategy can be problematic, at least initially. On the other hand, PBL is shown to contribute to increased retention of knowledge, the ability to transfer learning to new domains, and increased ability to synthesize basic concepts. A more cogent outcome is that PBL enhances intrinsic interest in the subject matter, and enhances self-directed learning skills that endure.

PBL has been applied in a number of disciplines such as architecture, engineering, medicine and management. In Medical Training, PBL interventions have been shown to increase student scores on clinically oriented standardized medical exams [19]. There are also similarities to the case method attributed to Harvard and taught in business schools [11].

The idea of PBL, thus, corresponds to elements of both theoretical groups outlined earlier. If problems are designed to allow exploration, leading to the formation of abstract concepts for students, they can, arguably, belong to the first group of theories. If the problems designed are larger and contextual, they may more directly map to the second theoretical group. In addition, the discussion so far has outlined the multiple theoretical perspectives and their potential translation to PBL but has not discussed how they may be specifically operationalized for learning service-oriented computing and the science of services. We describe this in the next section.

4. Designing Problems to Learn Service-oriented Computing
There are several operational constraints in moving from the ideals to specific implementations for teaching and learning service-oriented computing. Several have been alluded to in prior research and others are germane specifically to service-oriented computing. These include: (a) balancing the need to learn fundamentals versus the context [16]; (b) devising problems that represent frames likely to engage students [2]; (c) ensuring that problem scope is sufficiently narrow for learning and feedback within assigned time; and (d) providing opportunities for reflecting on the feedback provided to encourage learning. As an example, consider the following problem (see Figure 2).

Problem Consider Maya and Oscar, and their daughter, Pooja. Maya is an IT consultant, Oscar runs a part-time business. Both worry about being good parents. They do a number of tasks such as cooking and cleaning. Things do not always run smoothly though. Just yesterday, Maya thought Oscar was picking up Pooja from school, she had to attend a meeting. Oscar never read the note on the fridge; Maya spent an extra hour at school.

- What are the ‘services’ in this Organization?
- How are the ‘services’ communicating?
- What problems do you see?
- How would you solve these problem(s)?

A Discussion following the problem would involve four elements: Problem, Proposed Solution(s), Supporting Evidence, and Conclusions to facilitate learning. The discussion would link the general ideas to underlying technology implementations.

Figure 2. An example Problem

The problem is aimed at learning fundamental concepts. The problem frame is likely to appeal to nontraditional students but may not be as persuasive for traditional students. The problem scope is sufficiently narrow to provide time for exploration and discussion within a small time. As the example demonstrates, designing a good problem may require moving away from traditional bounds. Other than the Alice and Bob problems1 used in teaching security (often standing in for the proverbial A and B) few problems are available to educators. Consider a second example (Figure 3).

The four concerns outlined above are operationalized in this problem as the following. The problem aims at learning the fundamentals with a specific example. The problem frame is simple for the students to understand and relate to. The problem scope is narrow, and directs the student to learning the underlying technology. The tutorial that the instructor eventually returns to concretizes the learning and gives the students the opportunity to test the abstract concepts they may have formed against those in the WSDL specifications.

5. Discussion

Problem-based learning is not entirely new in computer science education. Previous efforts have targeted single courses; others are attempting larger interventions [3]. Yet others have tried to use variants driven by problem categories such as the use of Computer Graphics as a driving problem in multiple courses [9]. For service-oriented computing, PBL must build on these efforts and also pay attention to efforts in allied disciplines such as business [11]. This is necessary because, as argued earlier, service-oriented computing requires a rethinking of the role of the computing professional as one in a multi-disciplinary team.

Our arguments in the paper are, therefore, aimed at (a) a new technique for pedagogy in service-oriented computing that builds on the demands it places on academia, and in the process (b) a new domain (service-oriented computing) for the application of problem-based learning techniques. Towards this end, we have begun to create a repository of problems, which are ‘field-tested.’ As we push forward in refining problems and delivery of PBL in multiple classes, the technique is being adapted in a small set of ‘pilot’ courses at another university.

One significant outcome of this effort so far has been the realization that it is difficult to step away from entrenched modes of delivery. As figure 1 shows, PBL reverses the mode of classroom delivery. This is challenging for students as well as instructors. Problems associated with this switch include devising course materials, adjusting the pedagogy, and dealing with expectations from students and administration. Prior research shows that PBL almost always leads to an initial drop in teaching effectiveness [2]; making it

---

necessary to guard against a reaction of reversing to known approaches to teaching.

To deal with these, our ongoing work focuses in several directions. The first deals with an effort to make the problems more 'generic' so that they can be transferred. As expected, making the problems generic has a cost; our current efforts are aimed at lowering this cost to produce an initial library of problems. The second set of problems deals with an approach to codify and transfer successes and lessons learned across individuals and across universities. In spite of the significant literature about PBL in several domains, there are few specific and directed instructions and 'how-to's for use of PBL (for service-oriented computing). The third set of problems deals with evaluation, which has remained a concern for PBL for a long time. Our efforts to devise evaluation mechanisms are, aimed not only at assessing summative contributions (enhanced learning) but also formative outcomes that can lead to better problems. The progress outlined in this paper reflects the current status of a project aimed at understanding, codifying, evaluating and adopting innovative techniques for learning about service-oriented computing.

Acknowledgements: The work reported has been funded by the National Science Foundation under award numbers 722112 and 722141. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

6. References