https://doi.org/10.48009/1_iis_2012_391-401
Issues in Information Systems
Volume 13, Issue 1, pp. 391-401, 2012

ENABLING INNOVATIVE COURSEWORK THROUGH INCREMENTAL PROBLEM-BASED LEARNING

Amanda M. Holland-Minkley, Washington & Jefferson College, amh@washjeff.edu
Samuel B. Fee, Washington & Jefferson College, sam@washjeff.edu

ABSTRACT

Problem-Based Learning (PBL) organizes student learning around ill-structured, realistic problems that lead students to become self-directed learners and sophisticated problem-solvers. Though this pedagogy has its roots in medical education, it is gaining traction in computing education as well. We describe some of the reasons that PBL and computing and information technology curricula fit together well, describe an incremental approach to deploying PBL across a curriculum, and give specific examples of the course structure and assignments that might be used at various points in an incremental PBL curriculum. We conclude that the incremental PBL approach is not only effective in producing positive student outcomes, but it supports instructors in teaching courses involving emerging technologies and up-to-the-minute research by preparing students to take an active role in the learning activities of a course.

Keywords: IT Pedagogy, Problem-based Learning, Computing Education, Curriculum Development

INTRODUCTION

In teaching computing and information technology, there is a tension between teaching tools and teaching problem solving. Our courses focus around the idea of "doing," and our learning outcomes reflect a demand that students not only be able to describe how something works, or how something might be done, but to actually carry out the task and show themselves to be effective IT practitioners. However, this does not mean that we want our courses to simply be tutorials in the latest tools or checklists for potential resume items. While students' final products are tangible evidence that they have learned, it is equally important to us how they have created that artifact, and not just that it exists. We teach process, teamwork, project management, ethics, technical writing, effective communication, and above all problem solving skills, and we expect to see evidence of these skills in the work that our students do.

However, it is easy for our students to become overly focused on the tangible product - to argue that so long as the database they build, program they write, or website they create acts in the expected manner, they have done their work satisfactorily, regardless of the quality of their documentation, elegance of their code, or the ethics of their process. Despite what a project specification might say or the items listed on a grading rubric, a fair portion of our students persist in believing that these issues are ancillary to the true goal – using a tool, creating an artifact, making something "do what it is supposed to."

What to do in the face of this inevitable student focus on the tangible and active? We believe that Problem-Based Learning (PBL) offers a strong pedagogical foundation for the breadth of abilities that we expect students to exhibit upon leaving our courses. With its focus on engaging students in ill-structured, real-world problems and building robust problem-solving skills and self-directed learning, it is designed to encourage exactly those abilities that we find our students are often least interested, or able, to engage with. Further, because we are often already drawing on real-world problems in our courses, for those of us in computing fields the transition from our existing courses to a PBL version of the course may be easier than for colleagues in other fields considering adopting the pedagogy.

We believe that PBL offers another unique benefit to our discipline. Information technology and computing faculty and departments find themselves in a constant process of updating – their course, their curriculum, and the technologies they use. Students push departments to stay current, and most faculty are happy to try to oblige. For example, departments are finding themselves with an opportunity, and also an obligation, to incorporate mobile computing. But with cutting edge technologies such as mobile computing come inconsistent standards, incomplete development tools, and frequent changes coming out of up-to-the-minute innovation.

We observe that the characteristics that we wish to develop in our students, and which PBL helps us grow, are also characteristics that make students more able to work with emerging technologies in the classroom. This makes sense – we tell our students that these skills are necessary because, in their chosen field, it is unlikely that the technologies they are using today will be the same as the technologies they will be using in their jobs ten or even five years from now. We tell them that they must learn to understand the problems before them, to teach themselves new tools, techniques and systems, and to learn how to solve their problems creatively, because without these skills they will be ineffective as computing professionals. But we can also take advantage, as instructors, of the fact that these skills make our students more effective students, and that by growing these skills in them early, we are establishing habits that will allow them to take greater responsibility for their role in our courses as they progress through their degree.

In the remainder of this paper, we will describe the PBL approach, how it supports the goals of our curriculum, and how existing courses and assignments may be adapted to a PBL form. We will argue that by adopting a PBL approach thoughtfully, and in multiple courses across a curriculum, students will exhibit the features that we described above, and also be better prepared to engage in more sophisticated coursework and authentic learning.

PROBLEM-BASED LEARNING

Problem-Based Learning, in its simplest form, is a pedagogy that engages students in solving authentic, contextualized problems. The instructor's work focuses on crafting real-world problems that draw the students through the learning outcomes of the course [4]. It privileges "meaning-making over fact-collecting"[17] by requiring students to engage in the process of unpacking the problem before them and understanding for themselves what it would mean to solve the problem before approaching the task of creating a solution. Problems must be sufficiently ill-structured that this type of engagement is possible, and in fact necessary. Students, often in teams, take part in realistic problem-solving that guides them to understand the types of knowledge they require to solve a problem, and then to come together, either within groups or as part of the course as a whole, to gain that knowledge, motivated now by the task ahead of them.

We take as a given here that PBL can be effective, and our concern is with how to make use of the pedagogy in the face of the acknowledged difficulties that can occur, particularly when transitioning from traditional instruction to a PBL approach. We make this assertion with considerable confidence since the literature has yielded multiple examples of success in PBL environments [3, 8, 16]. The effectiveness of PBL in developing problem-solving skills and encouraging self-directed learning has been well studied and described [19]. It has been shown that PBL results in improved knowledge retention as well, and can be used effectively with a variety of types of problems [11]. The earliest of these studies focus on medical education, as PBL was developed within medical schools, but it has been adopted more widely in recent years, including within computer science [9, 13]. There are also some similarities to the advantages seen in test-driven development [12], requiring students to write the test cases their code must pass before they write the code, though PBL takes student responsibility for defining success as just one of its tenets.

It is important to note that PBL is not just about the use of projects in a course. Having students work on a project, even a real-world project, is a key characteristic of PBL, but it is not sufficient. The projects that we are talking about in PBL courses are ill-structured problems where students are given responsibility for some of the outcome definition themselves. Students must take ownership of defining what success will be for them. Once students have done this, they have created the targets that they have to meet, and the content and skills they have to learn to meet those targets are thus also created by them. However, as noted before, teaching courses anchored by realistic projects does offer a significant advantage in adopting a PBL approach in one's courses, as part of the work, and for some instructors, the hardest work, is already completed. It is further worth noting that by adopting a project to take on PBL characteristics as well, each student or team of students will engage with the project in a unique way, which can have the added happy benefit of giving projects greater longevity as unique student interpretations of the project will also ensure that unique, new work must be done to complete each project. We will further discuss how PBL can work within a course with specific examples below.

ADOPTING A PBL APPROACH

Though we are making a case for adopting a PBL approach, we must acknowledge that there are some common problems that can arise when doing so, many centered on the issue of student resistance. Often, students treat a course as a challenge set by a professor, to be mastered (or not) by the student. This is, of course, exactly what PBL is attempting to battle – this remove that students experience from their education. We require that students take part in defining their tasks, because we want to engage them and ask them to motivate themselves. But for most college students, our course would be the first time they would encounter this, and they may not be equipped to handle it. Some students may fear that there is a secret "right answer" – that despite what they wish to work on, they will be judged against the "right project" with the "right goals" that the instructor has set. And students may be frustrated that they are no longer told precisely what to do. This is, of course, also the point. But, it can also lead to students shutting down, disengaging, or deciding there is no way to succeed – particularly if the problems set before them are too large in scope.

These problems have been commonly acknowledged, and the usual advice is to start small, with very limited-scope problems, and work one's way up from there. We support doing this by implementing an incremental version of PBL on a curricular scale, not just within the context of a single course [7]; this has also been referred to as a gradual PBL elsewhere [2]. It is overly ambitious to expect a student with only moderate problem solving skills or experience with PBL to go from expecting thorough guidance and instruction to taking on authentic, ill-structured problems entirely independently in just one semester. This is a life-long skill they must develop, and we recommend that departments treat this as a skill that will develop across a curriculum, with each level of courses preparing students with an incrementally fuller PBL approach.

PBL at the Introductory Level

What this means is that in introductory courses, we focus on showing students the process of taking a problem, working to understand it and establish our success criteria, and then breaking it down into a set of sensibly-sized tasks. These may not look like PBL courses to some as they do tend to include constrained assignments with clearly set objectives and grading criteria, and little student involvement in the shape of the course. But at this level, the point is to give students a good, strong model of what an appropriate problem-solving process looks like, that they will be able to refer back to in their subsequent courses when they are given more freedom. We begin by modeling good behavior for them in the assignments we set and the way we talk about them.

Consider the example of an introductory programming course where we wish to have students illustrate that they understand objects and the basics of how to build them in Java. Suppose we want to assess a set of learning outcomes about implementing objects ("be able to create instance data," "use encapsulation appropriately," "be able to write a method declaration," "be able to have a method update the state of an object," "be able to use the methods of a created object appropriately"). A common assignment would give students a more-or-less real-world problem of implementing an object meeting some specification, set such that for the student to succeed they must have met the learning outcomes. The assignment might look as follows:

Implement an object in Java that that represents a Book, such as could be used in a Library program where the user wants to keep track of their progress through the books they are reading. Specifically, a Book should know its title, how many pages it has in it, and what page the reader is currently on in the book. Assume that all books start with none of the book being read yet. In addition to setting up an appropriate, minimal set of instance data and any needed constructor(s), provide a full set of accessor methods for the object, a method that allows the book to be updated to show that the reader has read some number *n* more pages in the book, and a toString() method for printing out the book's status at any time. Be sure to follow all course style guidelines, including an appropriate use of encapsulation. You should create a Library driver class as well to test each component of your Book object thoroughly.

Figure 1. Traditional Object Creation Assignment

Though this does describe a realistic, contextualized problem, under a traditional PBL approach, this assignment is overly constrained. All of the creative choices and problem solving have already been done for the student. They have been told what to assume about Books and how they work, and this has robbed them of the process of attempting to write the program and reflecting for themselves on what information they must know in order to solve the problem, and then searching it out or determining it for themselves. Thus, in a PBL setting, our problem might better appear as follows, where the student is given both a problem statement and then some supporting guidance:

Problem: Implement an object in Java that represents a Book, such as could be used in a Library program where the user wants to keep track of their progress through the books they are reading. A likely user group would be students wanting to track their course readings to easily see that they are on track and getting all of their work done.

Guidance: You will submit code for both the Library and the Book classes. At the top of your Library class, provide documentation of the ways in which the user may interact with the program, and at the top of your Book class, detail all assumptions you have made about how Book objects behave with brief justification of why these are sensible assumptions.

Figure 2. Problem-Based Learning Object Creation Assignment

This, obviously, is a more creative and authentic problem-solving scenario. The student must think about what the problem really means. Given in this form, some students may decide that keeping track of some sense of the rate at which the user reads different books would be helpful. Others may want to provide graphical output of the relative progress through all of the books in the collection. They may disagree in how much and what types of bibliographic information should be stored, but given the context of the problem, they all ought to decide that the user will alternate between reading different books, and read different amounts of a book at different times. They will have to make these decisions for themselves, with the support of classmates, teammates and their instructor, and thus will be thinking about what it means to even solve a problem like this.

It is likely if you have taught an introductory programming course, you can list a number of problems that you would encounter in distributing this assignment as given in Figure 2. Some students would respond with "what do you want" and want to check every choice and every line of code with you to be sure it is right. Others would ask how to even get started, complaining that you "haven't told them what to do." And you yourself may be asking "how can I grade this assignment?" – both in terms of the time required and concerns about fairness and consistency. What if some students decide to make simple choices, decide that nothing more than current page number is important, and generally do not rise to the challenge?

These are legitimate concerns all rooted in a single problem – that these introductory students simply are not prepared to handle such an ill-structured problem. They have not been asked to solve problems like this before. They are used to their assignments, and their grades, being focused around clearly specified targets, defendable through detailed and often pre-articulated rubrics. This is the exact antithesis of authentic, real-world problem solving, but it is what they have come to understand problem-solving to be. They have no experience, and they more than likely have never even seen somebody go through such a process. Before presenting students with full PBL-style problems we must show students carefully what it is to solve a problem: what it is to think about what a problem means, break it into sensible questions, and see how our decisions raise new questions or shape what options are open to us. Most importantly, we must show our students examples of good solutions to problems. Only then can we fairly expect them to give good, robust solutions to problems in their upper-level courses, and fairly assess them based on their ability to do so. Our introductory courses should lay the groundwork for a more robust PBL approach to be used in more advanced courses.

So, for our running example, we will offer a middle ground between the two versions seen so far. We will reintroduce the specificity of the first version of the assignment; it will be clear what must be done and even to an extent how it must be done. But the difference will be, we will discuss within the assignment that these are choices and expose to the students the design choices we ourselves made while creating the assignment, modeling for them how we hope they will come to approach a problem like this for themselves. This may give us a version of the assignment that now looks like follows:

This problem will walk you through the process of developing a Book object, such as could be used in a Library program where the user wants to keep track of their progress through the books they are reading, such as a student wanting to keep track of their progress in their course readings.

1. To start, we have to set up the basic information about a Book. We know Book objects have titles and a number of pages, and a page at which you have left off reading. Create a new class definition for a Book and declare the needed instance data to support this description.

2. We know that every object class must have a constructor whose role is to set up the object upon instantiation, specifically initializing the instance data. Given the instance data you declared in step one, write the necessary supporting constructor.

3. Now you are ready to start your driver class. For now, just have it create a single Book object, perhaps one for the 306 page book "Free Culture." Be sure to compile your code before proceeding; now that you have both an object and a driver class you will be able to compile and debug at each step.

4. Program output will help us debug; for this step, create a toString() method that, when called, prints out the current status of a Book object. Call it from the driver class on your Book. A possible output at this point might be:

   ```
   You are on page 0 of the 306 page book "Free Culture."
   ```

5. Now, add a read() method to the Book class that takes in a number of pages read and updates the object. Call it from the driver class, and use output statements to test it. If the method is called with the arguments 54 and then 30 and the object is then printed out, our running example might output:

   ```
   You are on page 84 of the 306 page book "Free Culture."
   ```

6. Next, add methods that return specific numeral values describing the state of the object. A getPage() method should return the page number the reader is currently on. A getRemaining() method should return the number of pages left to read in the book. And a getPercentDone() method should return the percent of the book that has been read so far. Calling these three methods in turn on the Book object in our running example should allow us to print out the following from the driver class:

   ```
   You are on page 84 of the book.
   You have 220 pages left to read in the book.
   You are 27.5% of the way through the book.
   ```

7. Finally, create a second Book object and call the same range of methods on it, showing that your code works for multiple instantiations.

Figure 3. Adapted Object Creation Assignment for Incremental PBL

What we have ended up with here is not just more comfortably specific for the students, but it should also be clear that each learning objective we set easily corresponds to a step in our assignment, and in fact our grading should become simple again. But the student has begun to begun to read about, talk about, and think about what it is to solve a problem, and is becoming familiar with the types of problem-decompositions that are required to truly pull apart and understand a problem before starting to solve it, preparing them to encounter PBL in their next courses.

We recommend proceeding to adopt PBL incrementally through the curriculum. At the intermediate level, you may want to hit a midway point between this type of assignment and a full-PBL version. Often, for us, this involves setting a partial specification for a problem, giving students a very specific set of open questions about the problem to determine for themselves, and then asking them to submit their choices as a separate part of the assignment that we explicitly grade, comment on, and if necessary ask them to revise to ensure that they are on a good track and solving the problem well and in a way that is consistent with the goals of the course. This also gives students gentle practice with and feedback on their burgeoning problem defining and solving skills at this intermediate level. It gives the instructor confidence that the student actually is developing improved problem-solving skills or the opportunity to work more closely with those students who continue to struggle.

PBL in Advanced Coursework

The payoff comes in the upper level courses, where we are prepared to fully engage in the PBL approach. In this setting, we can return to ill-structured, real-world questions, formulated more like the example in Figure 2. At this point, students are prepared to do the work of defining interesting problems and articulating success criteria. They are also ready to face the challenges of self-guided education to meet their individually customized goals. To illustrate this, we look at two upper level courses with fairly different content that are both able to take advantage of the incremental work of our curriculum to engage students in full PBL style projects: an advanced digital media course in Mobile Application Development and an advanced computer science course in Artificial Intelligence.

Mobile Application Development is an upper level course designed with two complementary learning outcomes. The first is for students to reflect on the particular usability problems within mobile applications and what does and does not work in these interfaces, particularly as compared to other contexts like web applications. The second is for students to understand the complexity of the toolsets available for mobile app development and be able to select the appropriate tools for a given application. The professional relevance of both of these outcomes reflects the newness of this area of development, with few standards set and new best practices coming out every month or week. Because of this, even experts in the field find themselves constantly learning new tools and skills to keep up. In order to cope with this uncertainty and change, there seem to be two approaches to teaching such a course. One is to select out a single set of tools such as App Inventor or Xcode as a development platform and then teach the course within that context. While this allows students to carry out the design and development process, it limits the ability to tackle our second learning outcome of understanding how to select the appropriate technologies for app creation. It is not authentic to the types of work those professionally engaged in mobile application development are doing nor does it anticipate the likely radically different environment that students will encounter in mobile application design and development in five years. Furthermore, in some cases, the appropriate technologies may emerge during the teaching of the course and need to be learned by the instructor and student alike. In our own most recent offering of the course, the planned usage of HTML5's WebSQL specification had to be replaced by other methods such as using localStorage when WebSQL was deprecated in HTML5. This adjustment was possible because of our alternate approach to teaching the course.

Instead of selecting a single set of tools, we structure our Mobile Application Development course around introducing students to a range of tools, and then reinforcing the need to make good choices in their individual projects. The first week of the course is spent surveying the broad range of technologies available for development, with the next significant portion of the course focusing on the students learning and then using specified tools to solve small problems that fit well with the strengths of those tools. The second half of the course is then dominated by students developing an application to solve the following problem:

Problem: You have been tasked with developing a mobile application for entering data into a pre-existing database. The database already holds a year's worth of data, so you must be careful to preserve what is there while also successfully appending the new data to the existing structure. In addition, a clear and concise front-end interface must be designed to effectively limit possible user errors.

Guidance: Each individual will begin by writing a short text description of their proposed product, including multiple images of potential interface designs and features. Additional online readings relevant to the common technical challenges faced by students will be provided via the course web page. After beginning the coding for the project, students will articulate, through conversation with the instructor, the success criteria for the product. Final success for the project will be assessed by a comparison of that initial conversation regarding the product to the final form of the project.

Figure 4. Problem Description, Mobile Application Development

Given this problem and guidance, students begin by describing the details of their app, such as a stats tracking app for an athletic team or a data collection tool for ecologists collecting data in the field. The needs of these different apps and their different users lead them to develop their apps in very different ways. The course readings and class activities are shaped by the needs of the specific projects. The assignment and our chosen course outcomes allow students to engage with a more accurate reflection of app design and development work is done in the real-world, but also reflects exactly the PBL philosophy of granting students the authority of setting their own goals and expecting them to then guide their education in the directions that allow them to achieve those goals. It also accommodates changes in tool usage, such as our switch to localStorage. These are also only feasible because the instructor is able to press the students to take on significant responsibility for the shape and content of the course, and to have the confidence that their previous coursework has taught them not only the needed coding skills but also the needed independent learning and problem solving skills to rise to the challenge.

Perhaps our most dissimilar upper-level course to Mobile Application Development, in terms of content, is our Artificial Intelligence course. If Mobile Application Development explores an emerging field with students tackling bleeding edge tools, our Artificial Intelligence course is a more traditional and theoretical offering, based on Russell and Norvig's text [18], and with a mix of project work and problem sets. However, here too we follow a PBL approach. While AI is an old field compared to mobile application development, it is also exhibiting yearly if not monthly advances, with similar pressure to keep course content up-to-date with the most recent innovations. For example, in our Spring 2011 offering, it would have been impossible to discuss information retrieval, natural language processing, or even learning without talking about Watson, actively competing on Jeopardy during that semester, and ideally to allow student interest in that project to motivate their work in the course.

In this setting, our PBL approach is reflected in two ways. The first is in how the content of the course is selected. The first half of the course covers a set of core concepts in AI that we wish every student to see. But it is impossible for a single undergraduate course to cover the entirely of the field of AI (something Russell and Norvig even grant in the preface to their text). Instead of the instructor deciding for the students which of the "optional" topics will then be included, this choice is thrown open to the students. Pairs of students are assigned to skim each of the chapters of the text that are not planned to be covered in the first half of the course and write up a paragraph summary of the content of each chapter. These summaries are merged, and students then give ranked votes of the five additional chapters they would most like to see covered. This is then used to determine the course content of the second half of the course. PBL is also reflected in the course project used. Students are asked to propose a topic they want to explore more deeply, and then develop either a literature-survey project or an implementation project applying a technique studied in class. Students are given broad freedom to choose a topic and are encouraged to find a project that complements their other coursework or career goals, but are given the following guidance on what they must be able to do by the end:

Problem: Identify a topic within Artificial Intelligence that relates to your personal curricular or career goals, and develop an understanding of the current abilities of AI within that area so as to be able to assess when this sort of AI may or may not be useful.

Guidance: Each individual/team will negotiate the specifics of their report with me on a case by case basis. However, each report will at the least do the following: (1) Describe clearly the problem being looked at. (2) Explain how the problem falls within the discipline of Artificial Intelligence. (3) Describe clearly what would be considered success. (4) Discuss what makes the problem hard and where there remains work to be done. (5) Draw a comparison to the successes of AI (either theoretically or in your own work) to human performance at the problem in question. Additional questions will need to be considered for each project as agreed upon in later conversations.

Figure 5. Problem Description, Artificial Intelligence

Essentially, students are told the learning outcomes of the project and are then expected to meet them. The instructor is available throughout the process to offer feedback, both on the formal project proposal and informally during class time and office hours to discuss both the direction of the project and get assistance with the content of the project as well. The end result of implementing PBL in this way is that the course is able to be customized to fit the interests of the students currently in the course, and to adapt to incorporate new developments even as the course is taking place. This particularly helps us given the diversity of students who take the course. As an interdisciplinary computing program, the majority of students taking artificial intelligence are interested in understanding how AI may inform the work of their chosen career path, how it is actually used in practical products, and for some, how it meshes with their interest in psychology or neuroscience. The PBL approach permits students the flexibility to make the course useful to themselves.

Student Outcomes with Incremental PBL

Adjusting pedagogy, particularly for preexisting courses, can be time-consuming and may not receive formal consideration in faculty review processes. Our purpose here is to describe methods for transitioning to a PBL approach in a manner that makes effective use of prior course development work and that has the potential to support faculty in innovative advanced courses. We have seen when adopting these approaches ourselves that the expected benefits from PBL do appear to carry through in our courses. We have reported elsewhere [6, 7] on our particular application of incremental PBL across our entire curriculum, with a more detailed description of the learning objectives and level of instructor guidance appropriate at each level of the curriculum in our incremental approach. We also provide qualitative data based on alumni surveys showing a consciousness of problem-solving processes and ongoing ability to tackle real-world problems in the workplace, consistent with the general findings in the PBL literature [6]. Responses consistently identified the problem-solving aspects of their education, and not just the content-knowledge, as essential to their current career progress. One student specifically identified that their introductory courses taught them what an acceptable and complete answer to a problem looked like, preparing them for advanced coursework and eventually their career.

We have also looked at the effectiveness of our incremental approach in supporting student success on later PBL-style problems. Informally, the fact that students successfully craft and complete advanced course projects such as those described for our Mobile Application Development and Artificial Intelligence course give us confidence that our lower-level courses with their modified PBL style are being effective. Looking internally to our introductory programming course in particular, we see that adopting modified-PBL assignments, such as those in Figure 3 above, has a direct impact even within the course on student problem solving abilities. The course is structured around six short homework assignments (one week, 20-40 lines of code) focusing on practice of core programming skills, followed by a significant two-part project (four weeks, 300-500 lines of code) focused on combining skills and applying more advanced design and problem solving. Comparing our three offerings of the course since adopting the incremental PBL approach to the two prior traditional offerings, we see that students performed very similarly on the homework assignments in both cases, with the average grade differing by less than 2 percentage points between types of offerings. However, when looking at the projects, where students must do more sophisticated problem solving in creating their programs, the average project grade in the PBL offerings was 8.9 percentage points higher than in the previous offerings (83.6% versus 74.7%) and the standard deviation was significantly lower in the PBL offerings (12.1 versus 20.7). We conclude, from this then, that the modified introductory assignments, designed to support students in later PBL-style courses, are effective in improving the problem-solving skills of the students. This also supports our belief that, even if a department is uninterested or unable to adopt an incremental PBL approach across their entire curriculum, adopting the adjustments suggested for introductory courses alone may bring positive benefit.

Avoiding PBL Pitfalls

If our goal here is to describe effective methods for adopting PBL, we should also warn instructors considering the transition of potential pitfalls they will want to avoid. On a broad scale, it is important to plan for the time it will take to develop good problems, as well as for the increased time required to support student exploration of the problems. This will almost certainly include adjusting how class time is used, to enable collaborative work and active instructor feedback during the problem-solving process.

Instructors new to PBL also often struggle in crafting problems that are as ill-structured as desired. For example, consider the Mobile Application Development example from Figure 4. A traditional project of this sort, even one that allows students to select the audience and type of data to be collected, would likely include constraints describing what must and must not be considered when solving the problem. When these constraints relate to ensuring students follow a single, desired approach (e.g., "must incorporate the jQuery library") or artificially imposing breadth to force all students to have uniformity the challenges they face (e.g. "at least one piece of data being collected must be numeric "), they are inconsistent with the ill-structured form of PBL problems, and detract from the problem-solving process for the student [5, 4]. This is in contrast to restrictions that are authentic to the problem being posed, (e.g. requiring new data collected be integrated with existing data in a database, as in the provided example); it is not the case that ill-structured problems must lack all structure [14]. The onus is on the instructor to determine the minimum set of features that a problem solution must have, and craft a problem that can organically require those features. The desire to add details in order to ensure easy comparability between student solutions must be resisted, and grading rubrics must be written to reflect the deeper underlying skills, such as making sure to implement responsive design in the final product. Notice that in both Figures 4 and 5, it is explicitly stated that students will establish their problems and success criteria and have them approved by the instructor. This is the opportunity to take inappropriate projects – whether because they lack ambition or, as often, are too ambitious – and re-craft them into a more suitable form for the goals of the course, in conversation and collaboration with the students. Hung [10] provides additional information on helpful methods for crafting good problems, and is recommended as a starting place for those looking for more detail.

Even in the modified-PBL style problems we suggest for introductory courses, such as is shown in Figure 3, we recommend keeping these principles in mind. In that example, we do describe the set of methods that will be needed, because at the introductory level, we are still providing students with examples of object design rather than expecting them to be able to create such designs for themselves. But notice that we still leave details, such as the number of arguments methods take, or the return types of methods, unspecified. At each stage in crafting this type of problem specification, the instructor should ask themselves if they are providing a detail that a student in the course could reasonably be expected to determine for themselves with reflection on the problem posed, and if so, that detail should be removed. In general, instructors new to writing PBL problems will provide significantly more detail and constraint than recommended, and should be aware of that tendency as they adapt to the pedagogy.

CONCLUSIONS

In both our example of the Mobile Applications Development course and the Artificial Intelligence course, we see that problem-based learning is not just a theoretically appealing pedagogy for our discipline, with its focus on real-world, independent problem solving, but can also be used to enable instructors to teach courses that cover a breadth of content, to incorporate current advances even as the course progresses, and to include content that is still in the process of being developed during the course, by involving students in the responsibility of creating and then mastering coursework to meet the set objectives.

We believe there is an additional advantage within computing disciplines to taking an incremental approach to PBL. The field continues to work actively to increase diversity, and one of the best-practices currently encouraged is to ensure that course content feels authentic to real-world problems students may wish to solve, and not focused on toy problems or experimenting with technology its own sake [1]. PBL clearly encourages the correct types of problems. However, adopting a PBL approach without the incremental adoption we suggest runs the risk, we believe, of actually contributing to diversity problems not remedying them. It is well understood that women and under-represented minorities may come to computing courses with interest, but without significant experience. In particular, they may come without the experience of experimenting with technology or "playing" to figure out how things work. Even in a traditional course, this can lead to retention problems as these students are assumed to be less interested because they do not engage in these practices, or less skilled because they do not have a habit of "messing around" with technology until it works [15]. For this reason, if one is going to follow a PBL approach, it is important to do so in a way that helps these students develop their skills in independent learning rather than assuming that a "good" or "hard-core" computing student would have been doing this already. Our incremental approach will bring these interested but less-experienced students along in the process.

It must be acknowledged that the PBL approach we have described represents a significant time investment on the part of the instructor, just as any PBL approach does. While much of a course can still be prepared in advance (any offering of our Mobile Application Development course will require lectures on responsive design and open standards such as HTML5 and any offering of our Artificial Intelligence course will surely cover search and learning), the instructor must be prepared to guide students through learning new content on the fly, and often to learn new content themselves as they teach the course. Assignments and projects may take less time to prepare in advance, as they need not come in formally-packaged formats, but this is counterbalanced by time spent as students develop their projects, and in then creating fair assessments from the range of final products. An instructor must be prepared for this shift in their workload, and it is vital to have the support of one's department in taking it on.

Even if the full PBL style upper-level courses we endorse are not appropriate for a particular program, whether due to resource limitations, the size of the courses, or other factors, or if an entire curriculum cannot be adapted, benefits can still be found in adapting introductory courses to the model we suggest. There is no course within a computing curriculum that does not benefit from students who enter with a stronger appreciation for robust problem solving. Further, the capstone experiences common in computing majors often hold the flavor of our upper-level PBL-style offerings, with their frequent focus on teams of students solving large and often real-world problems with significant independence. In these courses alone, the benefits of explicitly developing problem-solving and independent-learning skills in students ought to be observed, as we have found they are in our own service-learning capstone.

Ultimately, we hope we have argued that adopting a PBL approach incrementally across multiple courses or an entire curriculum has a variety of payoffs. Some are to the students, as they develop the skills, as well as mastering the content, that will prepare them for graduate study or careers in a rapidly-evolving field. But we have also illustrated that there are payoffs to the instructors themselves. By developing the skills in our students to enable them to take significant responsibility for their own education, we can take on courses that incorporate authentic problems and include up-to-the-moment content and tools. Ultimately, we develop students who are prepared sooner to go beyond their coursework and become independent scholars and our research collaborators.

REFERENCES

1. American Association of University Women (2000). *Tech Savvy: Educating Girls in the New Computer Age*. Washington, DC: AAUW Educational Foundation.
2. Angeli, C. (2002). Teacher's practical theories for the design and implementation of problem-based learning. *Science Education International*, 13(3), 9-15.

3. Belland, B., Glazewski, K., & Ertmer, P. (2009). Inclusion and problem-based learning: Roles of students in a mixed-ability group. *Research in Middle Level Education*, 32(9).
4. Duch, B., Groh, S., & Allen, D. (2001). *The Power of problem-based learning: A practical "how to" for teaching undergraduate courses in any discipline*. Sterling, VA: Stylus Publishing.
5. Fee, S. & Belland, B. (2012) . Understanding Criticism and Problem-Based Learning: An Introduction. *The Role of Criticism in Understanding Problem Solving, Explorations in the Learning Sciences, Instructional Systems and Performance Technologies 5*. New York, NY: Springer.
6. Fee, S. & Holland-Minkley, A. (2010). Teaching Computer Science through Problems, not Solutions. *Computer Science Education*, 20(2).
7. Fee, S. & Holland-Minkley, A. (2012). Correlating Problems throughout an Interdisciplinary Curriculum. *The Role of Criticism in Understanding Problem Solving, Explorations in the Learning Sciences, Instructional Systems and Performance Technologies 5*. New York, NY: Springer.
8. Finkelstein, N., Hanson, T., Huang, C., Hirschman, B., & Huang, M. (2011). *Effects of problem-based economics on high school economics instruction*. NCEE Report number 2010-4002rev.
9. Garćıa-Famoso, M. (2006.) Problem-based learning: a case study in computer science. *Proceedings of the Third International Conference on Multimedia and Information & Communication Technologies in Education*.
10. Hung, W. (2006). The 3C3R model: A conceptual framework for designing problems in PBL. *Interdisciplinary Journal of Problem-based Learning*, 1(1), Article 6. Available at http://docs.lib.purdue.edu/ijpbl/vol1/iss1/6.
11. Jonassen, D. & Hung, W. (2008). All Problems are Not Equal: Implications for Problem-Based Learning, *Interdisciplinary Journal of Problem-based Learning*: Vol. 2: Iss. 2, Article 4. Available at: http://docs.lib.purdue.edu/ijpbl/vol2/iss2/4
12. Jones, C. (2004). Test-driven development goes to school. *Journal of Computing Sciences in Colleges*, 20, 1 (October 2004), 220-231.
13. Kay, J. Barg, M. Fekete, A. Greening, T. Hollands, O. Kingston, J. H. & Crawford, K. (2000). Problem-based learning for foundation computer science courses. *Computer Science Education*. 10(2).
14. Kirschner, P., Sweller, J., & Clark, R. (2006). Why minimal guidance during instruction does not work: An analysis of failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2), 75-86.
15. Margolis, J. & Fisher, A. (2003). *Unlocking the Clubhouse: Women in Computing*. Cambridge, MA: The MIT Press.
16. Pedersen, S. & Liu, M. (2002-2003). The transfer of problem-solving skills from a problem-based learning environment: The effect of modeling an expert's cognitive processes. *Journal of Research on Technology in Education*, 35, 303-320.
17. Rhem, J. (1998). Problem-based learning: An introduction. *The National Learning and Teaching Forum, 8*(1), 2-4.
18. Russell, S. & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall.
19. Savery, J. (2006). Overview of problem-based learning: Definitions and distinctions. *The Interdisciplinary Journal of Problem-Based Learning, 1*(2), 9-20.