IT Problem Solving: An Implementation of Computational Thinking in Information Technology

Jaime L. L’Heureux
Bunker Hill Community College
250 New Rutherford Avenue
Boston, MA 02129
617-228-3229
jllheure@bhcc.mass.edu

Robert Cohen
Wellesley High School
50 Rice Street
Wellesley, MA 02481
781-446-6290
robert_cohen@wellesley.k12.ma.us

Deborah Boisvert
University of Massachusetts Boston
100 Morrissey Boulevard
Boston, Massachusetts 02125
617-287-7295
deborah.boisvert@umb.edu

Kamaljeet Sanghera
George Mason University
4400 University Drive
Fairfax, VA 22030
703-993-1547
ksangher@gmu.edu

ABSTRACT
This paper describes the implementation of information technology problem-solving constructs and scenarios designed to cultivate computational thinking in information technology education at the college level via a course entitled “IT Problem Solving.” A project of Broadening Advanced Technological Connections (BATEC), these scenarios were developed by a team of researchers under the auspices of an NSF CPATH grant focusing on adapting Computational Thinking as defined by Jeanette Wing into a more applied framework in partnership with and validated by a broad set of Information Technology (IT) professionals. The methodologies used within this highly successful course at Bunker Hill Community College may be of interest to other departments with existing IT programs that would like to take advantage of the strengths of the problem solving approach.

Categories and Subject Descriptors
K.3.2 [Computers and Education]: Computer & Information Science Education – information technology education.

General Terms
Experimentation, Design

Keywords
Information Technology (IT), Problem Solving, Computational Thinking, CPATH, Education, Curriculum Development.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIGITE’12, October 11–13, 2012, Calgary, Alberta, Canada. Copyright 2012 ACM 978-1-4503-1464-0/12/10...$15.00.

1. INTRODUCTION
Jeanette Wing, Jan Cuny and Larry Snyder have defined Computational Thinking as “the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent.” [1] Computational thinking has been traditionally defined in a very traditional computer science centric way, using terms such as recursion, abstraction, algorithms and the like. While many of our IT-centric industry partners are in fact formally trained in Computer Science, when we discussed these concepts with them, they considered this to be “education-speak”. They did not use these terms to describe similar capabilities in their typical daily routine nor did they feel that the terms reflected the broader skill set needed as an IT Professional.

We decided to work with our business partners to adapt Jeanette Wing’s framework [2] to a more information technology-oriented and business-friendly format. To accomplish this, we looked at a common business scenario and broke it down into its relevant parts from needs assessment, to process and design specifications, to implementation activities, to quality control, and finally to project evaluation. We then looked at the capabilities that were needed to achieve the outcomes desired, which resulted in the following framework of computational thinking, which we believe to be more applied and broad-based so that it can be used across computing and other disciplines.

IT Problem Solving that develops analogous computational thinking capabilities involves:

- **Logical Thinking** – Creatively develop, select and test relevant hypotheses
  - Asks probing questions to uncover details of the problem
  - Clearly defines the problem
  - Defines clear success criteria for the project including measurable objectives
2. COURSE DESCRIPTION

2.1 Course Overview

The course is offered specifically to Computer Information Technology majors as an alternative to an existing required introductory class. The course meets in two 75-minute sessions per week during a 15-week semester. The time devoted to each topic is measured in weeks consisting of two 75-minute classes and the associated time students spend working outside of the class meetings.

Specifically, this course gives students hands-on experience in the world of modern information technology (IT). Several IT concepts are introduced that provide a basis for further study in information technology. Students work on a number of problems that will give perspectives on areas of IT including but not limited to: visual programming, mobile device programming, and desktop security. Students leave the course with an understanding of PBL, the components of modern IT systems, basic programming concepts, and the scope of knowledge needed to become an IT professional. The benefit to students is that PBL enhances employability through the development of skills such as critical and analytical thinking, problem solving, communication and teamwork. These are necessary skills for the ever-changing technology driven, data-saturated world we live in.

2.2 Course Objectives

The major goal of the course is to have students acquire a technical overview of modern information technology. Students who successfully complete this course should be able to:

- Understand many aspects of Information Technology and their role in modern Information Technology implementations
- Understand Problem Based Learning (PBL) and the PBL Lifecycle
- Work in teams to produce solutions.
- Research and learn about technology.
- Present solutions in writing and spoken presentations
- Demonstrate knowledge of key IT foundational skills
- Demonstrate programming concepts through program execution

2.3 Methods of Instruction

PBL is a student-centered pedagogy in which they learn about a subject in the context of complex and realistic problems. Students identify what they already know, what they need to know, and how and where to access new information that may lead to resolution of the problem. Students “learn how to analyze, criticize and select from alternative sources of information and courses of action; how to think about problems that may have more than one viable solution; how to work together with those of differing views; and how to confront and act upon problems and situations in constructive and creative ways.” [3] As such, the instructor acts as a facilitator of learning who provides appropriate scaffolding of that process and measures student success via outcome based assessments. This methodology is highly transferable and builds the cognitive ability to apply knowledge gained to any situation a student might encounter. Such knowledge is highly valuable because it demonstrates more than rote memorization of an application’s functions.

Problems are the main focus of all course activities, including discussions and labs. There are 7 problems introduced during the semester, including a final project. A typical week includes:

- A warm up problem solving exercise
- A follow-up discussion analyzing what problem solving method steps were needed or used to solve the warm up problem
- A short discussion that introduces a problem and some brief tutorial on how to approach this sort of problem
- Lab time where students work on the problem
- A wrap-up discussion where students address their own and each others’ solutions

Grades are based on performance in the group and individual problems, and on deliverables generated during the problem.

2.4 Problem Solving Methodology

A general and widely accepted definition of problem solving is: Problem solving is (goal directed) thinking and action in situations for which no routine solution procedure is available. The problem solver has a more or less well-defined goal, but does

- Strategizing – Ability to anticipate and evaluate potential outcomes
  - Anticipates and evaluates the effects of various design options
  - Makes design decisions based on rational criteria
- Abstract Thinking – Ability to find appropriate level of detail to define and solve a problem
  - Decomposes a problem into component parts
  - Understands the relationships between components
- Procedural Thinking – Ability to select and execute appropriate steps to solve a problem
  - Identifies the steps and processes required to solve a problem
  - Identifies the sequence of steps including possible decisions and branching
  - Understands normal and exceptional behaviors of a solution
- Optimizing – Ability to analyze processes for optimal efficiency and use of resources
  - Understands available resources
  - Develops a solution that uses only available resources
  - Measures and adapts the solution to optimize resource utilization
- Iterative Refinement – Process refinement with the goal of improving quality or precision.
  - Measures and evaluates solutions against the success criteria
  - Adjusts the design and implementation as needed

We have used this definition in the design of a problem-based course; it informs both the problems that we have developed and the problem-solving methodology we teach. In addition, it is these definitional constructs that are measured at the end of the projects via a computational thinking survey that measures through self-reporting how students are building computational thinking skills while they are solving a particular problem in the course.

We believe that our course can be an alternative IT-focused way to deliver the AP CS Principles course that is under development. In section 4, we present a mapping of the projects that make up our course to the learning objectives of CS Principles.
Each individual in the group assesses themselves and the other members of the group. Grades are issued by the professor using the assessments. Deliverables for the project include:

- A Mobile Application Idea as one PowerPoint slide
- A developed prototype of a Mobile App
- A journal entry that describes their experience. In particular, answer the following questions:
  - What problem are you solving with your App?
  - What did you learn working on this project?
  - What was difficult working with App Inventor? How did you overcome these difficulties?

2.7 Problem 3: Animated Storytelling with Programming (2.5 Weeks)

Students are asked to use Alice (www.alice.org) to make an animation that completes the following story:

“Alice is shocked to find Bob doing something completely unexpected…”

Their animation should tell a story of two or more characters (at least Alice and Bob). There needs to be recognizable plot elements:

- **Exposition** – telling of the history and motivations of the characters
- **Conflict** – a problem that one or more of the characters must solve
- **Climax** - what happens when the conflict is resolved
- **Resolution** – the end of the story

To encourage students to explore Alice features, their project must meet or exceed the following technical requirements. Their world must include at a minimum...

- three objects that they place in the world
- three loops, two of them different types of loops
- camera and object motion
- run at least 60 seconds long

Deliverables for this problem include:

- A Storyboard for their animation.
- Their Alice world (the .a2w file)
- A journal entry that describes their experience. In particular, answer the following questions:
  a. What story are you telling with your animation?
  b. What are the actors? How do they interact?
  c. What did you learn working on this project?
  d. What was difficult working with Alice? How did you overcome these difficulties?

2.8 Problem 4: Green IT Research (3 Weeks)

The Green IT project has four parts. Firstly, students discuss what they know about sustainability and Green IT to establish a beginning point for student understanding of the topic. Then the students are shown a video and presented materials which seek to define Green IT for them, including introducing cloud computing.

Next, students complete a laboratory experiment in which they use watt power meters to measure the amount of energy that is being consumed by computers, monitors, printers, and other electronic devices and appliances throughout the college and their homes. This data is entered into an Excel spreadsheet from which students calculate the cost of supplying energy to these various devices on an annual basis, and generate an informative chart. In addition, students analyze the data, report their findings and
provide recommendations indicating how the college might save money and decrease energy consumption, thereby increasing the college’s commitment to become more sustainable.

In addition, students visit a local electronics recycling facility to learn about challenges and procedures specific to electronic waste, the cost and benefits to sustainability of recycling, and the impact to the earth of failing to recycle these items. After this visit, students are assigned a reflective writing piece about their experience at the recycling center. Further students are given an assignment to create a Word document recommending a policy for the disposal of outdated computer equipment for the college.

2.9 Problem 5: IT Career Exploration (1 Week)

Though students are just beginning their college career, they are reminded that each class they take and experience they have all contributes to a bigger picture. Like Randy Pausch says, they should be working toward achieving their childhood dreams. So, students are charged with finding a job posting in the Boston Area that they are interested in as a future career. They are to research the education, experience, and certifications necessary for success in this position. They need to use websites such as Monster.com, hotjobs.com, indeed.com, etc. to locate the job posting. Students are also challenged to research the company, its business sector and the job forecast for that position.

Students are then asked to complete the “Do What You Are” which can be found at: www.collegescope.com/bhcc. Students then look at the analysis combined with the job search they had just performed and consider whether they still want to pursue the same career opportunities or if their mind has been changed. Students are then encouraged to write in their journal about what they discovered about themselves and their career path as a result of the “Do What You Are” analysis and career research. Students are then told to be sure that they tie their writing to both items specifically to be successful with this writing assignment.

Working with their Success Coach, students are instructed to write an educational plan that will meet the job requirements of their selected career and then work in partnership to register for the next semester’s classes. In parallel, students continue to read and complete “The Last Lecture” by Randy Pausch: III. Adventures….and Lessons Learned; IV. Enabling the Dreams of Others; V. It’s About How to Live Your Life; and VI. Final Remarks. Students then reflect and journal on their reading and include a thoughtful personal insight as it relates to Randy’s story.

2.10 Problem 6: Desktop Security Investigation (3.5 Weeks)

In this project students are asked by the College’s IT Department to design best practice procedures for the securing of information on each student’s computer and to ensure that each student’s computer’s resources are at the correct security level to access the college’s network. The project contains four parts.

In the first part of the project, students must consider what they know about security already, what they don’t know, what they need to know, and how to fill the gaps. Students must do this by brainstorming ideas as a consulting team and use Wordle (www.wordle.net) as an analysis tool. Students must then organize and categorize those ideas into a concise, coherent document. This deliverable must be submitted to the client within the client’s timeline and delivery method, for their review and approval.

The second part of the project asks students to provide the client with a written proposal for further research and exploration in one of four categories:

- Access Control
- Protecting against Attacks
- Protecting against Data Loss
- Versioning, Patches and Updates

The class then breaks into four subgroups and aligns themselves with one of the four categories. Students develop and deliver a proposal for further research into the four categories, specifically identifying security threats in the category with prioritization of the likelihood of those threats occurring.

The third part of the project asks students to propose three viable solutions that work within constraints of the resources (time, money, personnel, network bandwidth, etc.). They are then asked to propose three solutions that the client will review.

Further, the students are informed that the client would like to see written proof of testing that shows their solution in action. The document that they will provide to the client will contain step-by-step instructions along with screenshots demonstrating that the solution is effective.

Students must also reflect on what they have learned from this task by discussing a series of questions with their team and also writing about it in their journal. These questions included aspects regarding working in teams including team dynamics and interpersonal relationships, progress and process management, decision-making, prioritization, testing and documentation.

In part four, the last and final step, students are instructed to use the vulnerability identification and testing and analysis they have done already to develop a handbook for computer security procedures. This will serve as a guide that will help students and campus IT services stay safer. Most colleges use a student handbook to inform students about appropriate behaviors on campus, to alert students to important college policies or procedures and to protect both students and the college IT systems from potentially dangerous or illegal activities. The areas that were identified earlier serve as the four sections in the handbook:

- Access Control
- Protecting against Attacks
- Protecting against Data Loss
- Versioning, Patches and Updates

Each student group takes ownership of a section. Students need to develop the content for that section and make sure that it complies with the handbook requirements.

3. LESSONS LEARNED

3.1 Computational Thinking (CT) Survey

During the course, computational thinking was measured by using a survey generated from research under an NSF grant entitled Advancing the Successful IT Student through Enhanced Computational Thinking (ASSECT - CCF 0939089). This project brought together a consortium of community colleges and associated 4-year universities spanning five states (Massachusetts, represented by Bunker Hill Community College and University of Massachusetts Boston; Virginia, represented by Northern Virginia
Community College and George Mason University; Indiana, represented by Ivy Tech Community College and Purdue University Calumet; Colorado, represented by Cameron University and California, represented by City College of San Francisco) to explore an innovative process that adapts the IT Body of Knowledge outcomes developed by the Special Interest Group for Information Technology Education (SIGITE) of the Association for Computing Machinery to Wing’s Computational Thinking taxonomy. The project team took the underlying research on computational thinking and worked with industry partners to describe behaviors that demonstrate computational thinking in the context of information technology and develop 20 scenarios that can be used to build these capabilities.

Outcomes measurement was tied to Principled Assessment Designs for Inquiry (PADI) (funded previously by NSF) which produced a body of statistically valid, evidence-centered data. Instructor training on PADI assessment was used to support or disprove the hypothesis that the treatment produces reliable improvements in computational thinking over conventional instruction. At the beginning of each term, students will be given a pretest with cognitive and computational thinking elements. At the end of instruction, students will complete a posttest and randomly selected students will also participate in a think-aloud exercise. Quantitative analyses will be conducted using exploratory factor analysis and analyses of covariance (ANCOVA) comparing teaching students on each posttest using the respective pretest as the covariate.

Further, the inability to have control and treatment groups, required utilizing a quasi-experimental design. “With quasi-experimental studies, it may not be possible to convincingly demonstrate a causal link between the treatment condition and observed outcomes. This is particularly true if there are confounding variables that cannot be controlled or accounted for.”[6] Thus, the computational thinking survey was an anonymous self report study used by students to measure computational thinking behavior. Using the descriptions of these behaviors, the project created a survey to correlate computational thinking attributes with student learning.

The computational thinking survey asked students whether or not they engaged in the behavior and to explain how this was exemplified through their work. Questions included:

1. I asked probing questions and uncovered details to better understand the problem.
2. I clearly defined the problem.
3. I clearly defined what I needed to do and how my objectives would be measured.
4. I anticipated and evaluated the effects of various solution options.
5. I made decisions about possible solutions based on relevant criteria.
6. One technique of solving problems is to identify and write down the key parts of the problem (deconstruction) and then propose actions to address those parts. I deconstructed the problem into component parts.
7. I diagrammed or otherwise recorded the relationships between the parts (from above).
8. I recorded possible strategies for addressing the parts (from above).
9. I identified and recorded the steps required to solve the problem.
10. I identified typical and exceptional (or different) solutions to the problem. I measured and evaluated my solution against the success criteria.
11. I identified available and potential resources to address the problem.
12. I developed a solution that uses feasible resources.
13. I analyzed solutions to determine the optimal resources needed.
14. I measured and evaluated my solution against the success criteria.
15. I adjusted the solution and modified the implementation as needed.

3.2 Survey Results

For each question we analyzed whether students reported strongly agreeing with or agreeing with the statements with regard to computational thinking skills employed in the problems measured (Problems 2 and 3). Please note that Problem 6 will be measured also, but that data was not available at the time of the submission deadline for this paper.

Every question asked averaged at least a 50% response rating for strongly agreed or agreed. Therefore, it is reasonable to conclude that students feel that they are using the computational thinking skills in solving the problems presented in the course. Further, Questions 3, 17, 21, and 29 received an average response for strongly agree and agree by students of over 80%. This seems to indicate that students feel the computational thinking skills of defining the problem, identifying a sequence of steps including possible decisions and branching, understanding available resources, and being given the opportunity to adjust the solution and implementation as needed are successfully being learned and employed to solve the problems presented in the course.

For Problem 2, Questions 3 and 27 also ranked remarkably high with close to 90% of students reporting that they strongly agreed or agreed with the questions. It appears that the Mobile App Development problem pushed students to clearly define what they needed to do and understand how success of their objectives would be measured, and that they felt they were given the ability to measure and evaluate their solution against the success criteria.

However, in Problem 3, these questions only achieved a result of 75% and 62% respectively. Thus, Problem 2 seemed to allow students to achieve a depth of understanding in several computational thinking skills. Whereas, it seems as if Problem 3 had a wider reach across the computational thinking skills. Specifically, in Problem 3, 11 out of the 15 questions received a strongly agreed or agreed response by more than 2/3 of the class.

4. CONNECTION TO CS PRINCIPLES

As we have demonstrated, students experience a strong foundation in computational thinking through IT Problem Solving. We also have found a strong connection between IT Problem Solving and the proposed new AP CS Principles course [7]. The aim of that course is to provide a general education introduction to the “Big Ideas” in Computing. The Big Ideas are:

I. Computing is a creative human activity that engenders innovation and promotes exploration.
II. Abstraction reduces information and detail to focus on concepts relevant to understanding and solving problems.
III. Data and information facilitate the creation of knowledge.
IV. Algorithms are tools for developing and expressing solutions to computational problems.
V. Programming is a creative process that produces computational artifacts.

VI. Digital devices, systems, and the networks that interconnect them enable and foster computational approaches to solving problems.

VII. Computing enables innovation in other fields including mathematics, science, social science, humanities, arts, medicine, engineering, and business.

While numerous variations of CS Principles courses have been created and offered as pilots across the country, all have been developed using traditional computer science constructs. There is a significant need to expand that process to a more broad-based computing approach while maintaining the integrity of the underlying constructs. IT Problem Solving takes a more novel and applied approach to achieve a comparable outcome. This course stresses the typical types of problems that IT professionals deal with on a routine basis – ones that are contextual, adaptable, involve inter-relationships, and are both systematic and strategic in nature.

<table>
<thead>
<tr>
<th>IT Problem Solving Projects</th>
<th>CS Principles Big Ideas and Learning Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eportfolio</td>
<td>I</td>
</tr>
<tr>
<td>Mobile Apps</td>
<td></td>
</tr>
<tr>
<td>Animated Story Telling</td>
<td></td>
</tr>
<tr>
<td>Green IT</td>
<td></td>
</tr>
<tr>
<td>Career Exploration/Reflection</td>
<td></td>
</tr>
<tr>
<td>Desktop Security Integration</td>
<td></td>
</tr>
<tr>
<td>Final Project</td>
<td></td>
</tr>
</tbody>
</table>

Each of the Big Ideas is accompanied by learning objectives that provide guidance for how the “Big Ideas” can be conveyed. Figure 3 shows the coverage of the 30 learning objectives by the IT Problem Solving projects [8]. It is important to note that these learning objectives are covered multiple times throughout the IT Problem Solving course. For this reason, this course should be considered as another option for developing strong computational thinking attributes in students.

The only divergence from the current CS Principles pilots is a weaker coverage of the Big Idea IV, algorithms. This is not surprising, since the focus of information technology is on the application, integration and optimization of, rather than the development of software. In order to extend CS Principles to additional audiences of students studying IT, further research is indicated to determine a balance that will achieve the necessary skill sets that will enable them to be successful in their pursuit of more broad-based computing careers.

5. CONCLUSIONS

Information Technology Problem Solving was developed to implement problem solving as pedagogy and to employ some of the scenarios developed for the cultivation of computational thinking in informational technology done through an NSF CPATH grant. This was done to begin to ascertain whether computational thinking and the computer science principles could be conveyed to, and fashioned in, information technology students. We assert that this course has served to demonstrate that these can in fact be implemented in information technology curriculum and the implementation has proven to enhance computational thinking skills as reported by the students, as well as having the added benefits of increased student engagement and connection to the material.

Further research is indicated to determine whether development of computational thinking attributes is contingent on either pedagogical or content approaches, or whether a combination of the two, in IT Problem Solving.

6. REFERENCES


