Application of the Occupational Analysis of Computational Thinking-Enabled STEM Professionals as a Program Assessment Tool

Joyce Malyn-Smith
Education Development Center
55 Chapel Street
Newton MA 02458-1060
jmalynsmith@edc.org

Irene Lee
Santa Fe Institute
1399 Hyde Park Road
Santa Fe, NM 87505
lee@santafe.edu

ABSTRACT

This paper describes the application of findings from the National Science Foundation's project on Computational Thinking (CT) in America's Workplace to program assessment. It presents the process used to define the primary job functions and work tasks of CT-Enabled STEM professionals in today's scientific enterprise. Authors describe three programs developing CT skills among learners in secondary and post secondary programs and how the resulting occupational analysis was used to review these programs. The article presents ways this analysis can be used as a framework to guide the development of STEM learning outcomes and activities, and sets of directions for future work.

1. INTRODUCTION

Over the past several years, thought leaders within the computer science and education communities have defined computational thinking within their own communities of practice and discussed the importance of computational thinking as a key ingredient in technology-enabled discovery and innovation. [1, 2, 3, 9, 10, 22, 23] This national conversation has provoked questions about what computational thinking looks like in practice among scientists, engineers and technologists in various industry sectors and how core computational thinking skills might be nurtured in students as they prepare for STEM careers.

Clearly these conversations have made a significant contribution to the field by developing momentum for dialogue within the STEM education community and focusing attention on the importance of defining CT for purposes of developing programs and curricula. Thus far, however, the conversations have represented the perspectives of university-based thought leaders and others somewhat distanced from scientific, technical, and industry workplaces. And while they have succeeded in creating theoretical constructs for CT, these conversations represent only one side of the education-to-employment continuum and thus provide only an approximation of how CT is integrated into daily work activity and shapes problem solving in scientific work settings. There is an urgent need to build on and strengthen this good work by expanding the conversation to include STEM workers in a variety of settings who can ground these ongoing efforts to define CT in real work activities. Without clear, authentic examples and artifacts illustrating what CT looks like "in action" at work, educators will have difficulty building programs that lead to successful use of CT in STEM careers. Articulating authentic examples of CT in action requires the conversation to focus on expert computational thinkers who currently work in STEM fields. Expert workers can contribute accurate examples with the specificity and authenticity educators will need to integrate CT into K–20 learning.

The NSF-funded "Computational Thinking in America's Workplaces" project (NSF award #OCI 1057672 9/1/10 – 8/31/12) advances understanding of computational thinking by exploring CT as a foundational skill for STEM workers and developing a Profile that describes the ways scientists and other STEM professionals engage in CT as they carry out routine job tasks and solve problems associated with their work. Additionally, this project generated language and examples that promote new ways of talking about computational thinking and clarified the definition of CT by contextualizing it within the work of scientists and engineers. Products of the research include an occupational definition and profile of the Computational Thinking Enabled STEM Professional, and concrete examples of what CT looks like "in action" in America's scientific and engineering workplaces.

2. METHODOLOGY AND JUSTIFICATION

Building on the successes of ATE's IT Across Careers (ITAC) Project and a legacy of experience in developing national skill standards [4, 6, 7, 8, 15, 18, 20, 21], the project developed the "Learning Occupation" of a CT-enabled STEM worker then identified and validated with expert CT workers the "computational thinking" skills/competencies that are used by scientists and engineers in STEM careers. The process employed was one that had been used successfully to develop national skill standards for emerging industries (Biosciences) [11] and industries undergoing substantive changes in professional and technical job responsibilities (Human Services). [21] The Learning Occupation was used in the Bioscience and Human

Services Skill Standards Projects to represent an outcome goal for education and training designed for workers who will be able to perform a broad variety of related work tasks suitable to a large cluster of occupations. A similar situation exists in STEM, making this an appropriate application of the concept. The work process involved four distinct steps: building a team, defining a learning occupation, developing a profile of the CT-enabled STEM worker and creating examples of CT in action.

Building a team: EDC (Education Development Center, Inc.) assembled a project team that included 3 experienced, highly qualified skill standards developers; and a technical committee consisting of 4 computer scientists involved in national discussions on CT representing major universities, research institutions, and industry (University of Washington, Massachusetts Institute of Technology (MIT), Williams College, Santa Fe Institute, and Raytheon Corporation). This technical committee's role was to ensure that the project team's products would connect to the interests of national thought leaders in the field of computational thinking. A panel of 11 expert CT workers representing a range of STEM careers, occupational levels, and work settings was recruited to participate in the rigorous occupational analysis. Expert panelists included research scientists, theoretical physicists, software engineers, mathematicians, applied scientists, engineers and security specialists drawn primarily from National Laboratories.

Defining a learning occupation: The following learning occupation was developed by the technical committee and revised by the expert panel as a result of the analysis workshop:

"A computational thinking enabled STEM professional engages in a creative process to solve problems, design products, automate systems, or improve understanding by defining, modeling, qualifying and refining systems, processes or mechanisms generally through the use of computers. Computational thinking often occurs in collaboration with others."

Developing a profile of the CT-enabled STEM worker: Once the learning occupation was defined and agreed upon, the expert panel developed a profile of the CT-enabled STEM worker. The Learning Occupation proposed by the project's technical committee became the subject of a modified DACUM analysis. DACUM (Developing A CUrriculuM) [14] is an internationally-known methodology used by expert practitioners in an occupational field to identify the major areas of work and the constituent tasks that define successful job performance. The DACUM method has been used internationally for more than half a century to identify core workforce competencies. This process rests upon three basic principles:

- Expert workers can describe and define their jobs more accurately than anyone else.
- An effective way to define a job is to precisely describe the tasks that expert workers perform.
- All tasks, in order to be performed correctly, demand certain knowledge, skills, resources, and behaviors.

3. DACUM ANALYSIS

Traditional DACUM analyses invite expert practitioners representing a single occupation. The "modified" DACUM approach used successfully by EDC engaged expert workers *from a range of related occupations* who share a common core of work tasks, knowledge, and skills. The first task undertaken by this panel of experts was to discuss and refine the proposed Learning Occupation so that it captured the essence and commonalities of their own work. The ensuing guided dialogue provided descriptions of concrete, observable activities for which the panelists use CT and that met the definition of the Learning Occupation. From the set of CT activities described, the panel identified 11 large functional groupings or "job functions".

Eight job functions (A-H) were organized into four categories as follows: "A computational thinking enabled STEM professional….":

Defines:
- A. Identifies problem.
- B. Specifies constraints.

Models:
- C. Designs the model/system.
- D. Builds the model.
- E. Develops experimental design.

Qualifies:
- F. Verifies the model.

Refines:
- G. Optimizes the model and user-interface.
- H. Facilitates knowledge/discovery.

The panelists identified 68 activities/tasks in which the STEM professionals described in the learning occupation use computational thinking. Each of the 68 tasks was grouped under the job function category to which it best corresponded (see Table 1).

Three cross cutting job functions were also identified: Engages in a creative process, Collaborates, and Documents. In addition, the panelists developed lists of the Knowledge, Abilities (skills), Desirable Behaviors of CT enabled STEM professionals as well as selected Tools and Techniques used as they are engaged in the activities listed (see profile of a Computational Thinking-enabled STEM Professional).

Creating examples of CT in action: Although the profile identified the work tasks in which STEM professionals engage when they are thinking computationally, concrete examples that described what computational thinking "looks like in action" would be needed to build a strong dialog bridge between computer scientists who understood CT and non-computer scientist educators who were struggling to understand CT and connect it to learning objectives in their classes. To build this bridge project staff worked with the expert panel to draft twenty-nine examples of routine tasks and problems the expert panelists solved using CT. Two examples follow:

A computational scientist verifies the numerical convergence of a solid mechanics finite-element model, by refining the mesh associated with a mechanical assembly, for the purpose of assessing the correct implementation of the mathematical equations. [Qualifies: Verifies the model]

A nuclear engineer validates a coupled thermo-mechanical computer model, by comparing the model predictions with existing thermal stress experimental data, to assess the performance of a nuclear fuel element for the purpose of extending the operational lifetime of the fuel in the reactor. [Qualifies: Validates the model]

Although the examples of "CT in Action" emerging from this expert group were clarifying for non-computer scientists and

STEM education professionals, the technical depth and complexity of the tasks described limited the use of these examples within the K-12 curricula. Additional examples more relevant to the experiences of K-12 students, and simpler tools/strategies would be needed to help learners and their guides/teachers recognize and nurture computational thinking in the K-12. The Job Functions and Tasks identified in the DACUM analysis serve as a simple tool to bridge that gap.

4. APPLICATION OF THE CT DACUM AS A PROGRAM ASSESSMENT TOOL

The Occupational Profile (DACUM) of a computational thinking enabled STEM professional was used to evaluate the core computational thinking skills nurtured in three programs serving students ranging from the middle school and graduate school levels. The three programs, one at the middle school level, one primarily at the high school level, and one at the university level, were selected because they actively engage students in computational thinking through computational modeling and simulation. The focus on computational modeling and simulation programs was intentional as "the underlying idea in computational thinking is developing models and simulations of problems that one is trying to study and solve." [13]

The analysis was conducted by reviewing each program's curricular materials, lesson plans, student assessments and rubrics, pedagogy (as reflected in teacher professional development materials), and students' work products. Subsequently the authors interviewed individuals responsible for implementing each program's curriculum.

Santa Fe Institute's Project GUTS (Growing Up Thinking Scientifically) is an afterschool program at the middle school level that engages students in computational thinking through modeling and simulation in StarLogo TNG. In Project GUTS students actively engage in computational thinking as they design and implement models of local relevance and then use the models to run simulations. Students use the process of abstraction to narrow the problem down to something that could be implemented on a computer using StarLogo TNG, an agent based modeling tool. Students design and create models as test beds to answer questions about real-world concerns. For example, as part of the Project GUTS unit on Epidemiology, a group of students wanted to investigate whether a disease would spread throughout their school population given the layout of the school, the number of students, the movement of the students, the virulence of the disease, and the number of students initially infected. Mapping this question and scenario onto an agent based model, agents were used as abstractions or simplified representations of students and the number of agents matched the number of students in their school. Agents were given movement behaviors that were abstractions of moving from classroom to classroom, and decisions were made about which features of the school were important to take into consideration before a 3-D virtual model of the school building was created. For instance, students decided that recreating the number and location of passages and doors at the school was important. Additionally students modeled the characteristics of the contagion being spread: how often contact between students spread the disease from one to the other, and how many students were initially infected. To make the model a test-bed capable of running experiments, it was equipped with interface sliders to control individual variables. One interface element controlled the number of initially infected agents and another controlled the virulence of the contagious element.

A three-stage progression is used within Project GUTS to first engage and prepare youth in CT. This progression, called Use-Modify-Create [10], describes a pattern of engagement that was seen to support and deepen youth's acquisition of CT in the authors' NSF projects. It is based on the premise that scaffolding increasingly deeper interactions will promote the acquisition and development of CT. In the *use* stage, students run experiments using pre-existing simulations. Over time they begin to *modify* the model with increasing levels of sophistication. For example, a student may initially want to change the color of a character or some other purely visual attribute. Later the student may want to change the character's behavior in a way that entails developing new pieces of code. Modification of this kind necessitates an understanding of at least a subset of the abstraction and automation contained within a model. Through a series of modifications and iterative refinements, new skills and understandings are developed as what was once someone else's creation becomes one's own. As youth gain skills and confidence, they can be encouraged to develop ideas for new computational projects of their own design that address issues of their choosing.

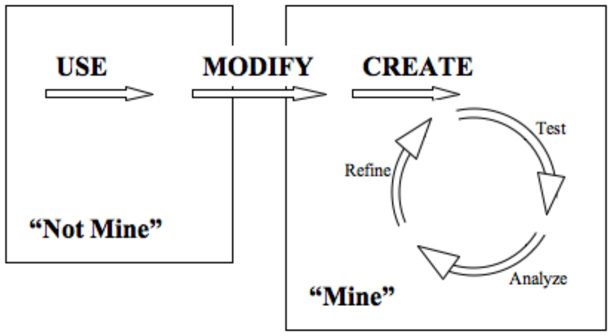

Diagram 1: Use-Modify-Create Learning Progression

Analyzing the CT learning within Project GUTS using the CT DACUM: Project GUTS engages middle school students in a creative process and encourages collaboration using pair-programming techniques, but only rudimentary coverage of the scope of tasks of a CT-enabled STEM professional are addressed. Of the 8 job functions delineated in the CT DACUM, the three primarily addressed during the course of participation in Project GUTS are: A) Identifies problem, C) Designs the model and D) Builds the model. (See Table 1.) Constraints are rarely addressed and students develop limited experimental designs. Rarely do they verify or refine a model. Note that the test-analyze-refine cycle in the diagram 1 above refers to iterative refinement in code development not model verification and validation.

The Supercomputing Challenge is a year-long program for middle and high school students culminating in a student competition. Middle and high school students are introduced to computational thinking and computational modeling at a Kickoff Conference held annually each fall. Students primarily use StarLogo TNG, NetLogo, and Java as the basis for their computational models and simulation. Teachers are prepared to sponsor and mentor student teams through the joint Supercomputing Challenge / Project GUTS Summer Teacher Institute. "Challenge" teams, working in small groups, develop computational modeling projects of their own choosing. They use a framework, called the "Computational Science Cycle" to guide them through the process of designing, implementing and analyzing a computational model. "This design-based approach has been effective in engaging learners in exploring computational ideas" [16, 17, 19]. Students are guided through the stages in the process as follows:

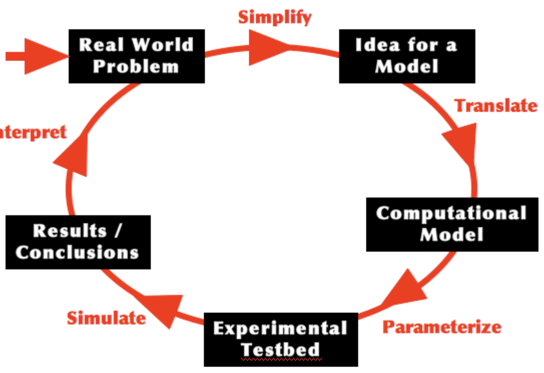

Diagram 2: The Computational Science Cycle

Stage 1: Select a real-world problem to study.
Discuss what makes a problem suitable for studying using computational methods. Describe the simplifications made in models through abstraction. Specify the measurable aspects of the problem and the questions that will be answered through modeling and simulation.

Stage 2: Simplify the scope of the model using abstraction. Specify the aspects of the problem that are important to include in the model and narrow the scope of the problem to one that can be modeled given the software and computing resources available.

Stage 3: Translate the idea for a model into a computational model. Decompose the problem. Abstract real-world objects into computational analogs. Abstract the physical behavior of the objects. Define interactions between variables, objects or elements. Choose appropriate representations. Use existing code and technology. Writes algorithms and programs. Debug and troubleshoot.

Stage 4: Parameterize the model. Discuss relevant variables and parameter and experiment design. Discuss what constitutes proof when using data output from models.

Stage 5: Simulate and collect data. Use the computational model as a test bed for running experiments. In some cases this involves writing another program that runs the model repeatedly over a set of input values; called a parameter sweep.

Stage 6: Analyze / Interpret: Discuss the limitations of the computer model, the assumptions were made, and what the model tell us, if anything, about the real world. Introduction to how models are verified and validated. Demonstrate the exploratory uses of models when no theory exists. [Verification is the demonstration that the model is logically correct and follows from the physical and mathematical laws used. Validation is the demonstration that the model correctly predicts the phenomena modeled.]

Repeat. The Computational Science Cycle is an iterative process. In evaluating the model one might find verification errors (e.g., bugs in code) or validation errors (e.g. when comparing model behavior to real-world data there are difference that suggest that the wrong assumptions or simplifications were made). In either case, the whole computational cycle repeats. It is an iterative refinement process.

Analyzing the CT learning within the Supercomputing Challenge: The Supercomputing Challenge engages middle and high school students in all three cross-cutting job functions; student teams engage in a creative process, collaborate in project work, and document all phases of their project. The Supercomputing Challenge requires student teams to submit project descriptions (abstracts), interim and final reports. In comparison to the middle school Project GUTS, a larger subset of the tasks of a CT-enabled STEM professional is addressed. Of the 8 job functions delineated in the CT DACUM, the four primarily addressed during the course of participation in the Supercomputing Challenge are: A) Identifies problem, C) Designs the model, D) Builds the model, and E) Develops experimental design. (See Table 1.) Top rated student projects address constraints and attempt at verification and validation of models however, the majority of projects do not reach this level of sophistication.

The NM EPSCoR (Experimental Program to Stimulate Competitive Research) program at New Mexico Tech offers opportunities for undergraduate and graduate students to participate on research projects in Computational Science, High Performance Computing, and Cyber-Infrastructure. Three themes, problem solving, collaboration and communication, are emphasized as keys to student success at NMT. In project-based courses, students conduct research projects as a major component of their coursework. They read and analyze project requirements, brainstorm project ideas then select and propose a project. Within each project, students working in project teams develop a solution to a real-world problem and communicate their results. Specific tasks include:

- Define and analyze requirements and specifications
- Determine feasibility and scope of problem
- Determine relevant assumptions, limitations, relationships among data
- Prototype problem to verify requirements
- Select modeling methodology, language / modeling environment, visualization
- Build model leading to hardware/software solution
- Develop experimental design (parameters & value ranges)
- Facilitate knowledge discovery (via model & vis.)
- Communicate results / accomplishment

Analyzing the CT learning within university courses at NM Tech: Undergraduate students enrolled in project-based CS courses at NM tech engage in all three cross-cutting job functions; they engage in a creative process, collaborate, and document and present their project. In comparison to the Supercomputing Challenge, a larger subset of the tasks of a CT-enabled STEM professional is addressed at the university level. Of the 8 job functions delineated in the CT DACUM, the five primarily addressed are: A) Identifies problem, B) Determines / specifies constraints, C) Designs the model, D) Builds the model, and E) Develops experimental design. (See Table 1.)

Within Senior Design projects, students are required to deliver a complete product to a customer. Students encounter tasks associated with the job function "Determines/Specifies Constraints". Senior design classes require that students work with clients/stakeholders to specify the requirements of the solution and resources, attend to stakeholder/customer communications and satisfaction, conduct a needs analysis and resolve conflicting requirements. Unlike earlier university project work, model verification and iterative refinement are necessary to improve the solution until the client/stakeholder is satisfied. Graduate research is expected to advance the state of the art in their discipline and communicate these advances to the field. This is in alignment with the expectations of a modern university.

5. FINDINGS

The DACUM occupational analysis provided a useful framework with which to evaluate the breadth and sequencing of CT instruction within computational modeling and simulation programs. Clearly students engaged in high quality, technical STEM learning are developing foundational computational thinking skills in the K-12 experience. Specifically, it was uncovered that the breadth and depth of tasks addressed corresponded with increases of grade level of the participants. Within each of the eight job functions (A. Identifies problem, B. Specifies constraints, C. Designs the model/system, D. Builds the model, E. Develops experimental design, F. Verifies the model, G. Optimizes the model and user-interface, and H. Facilitates knowledge/discovery), certain tasks were found to be appropriate for introduction at different grade levels. For example, within the "Identifies problem" job function, a middle school student in Project GUTS will identify the scope of the problem, select relevant aspects of the problem and define assumptions and limitations whereas a high school student participating in the Supercomputing Challenge additionally would be expected to identify data sources, risks of failure, and existing tools and solutions, research existing knowledge, determine if the problem has already been solved, and argue the need for a computational approach.

Across major job function categories, several entire categories were not addressed at all at the middle school level. Qualification and refinement of models based on verification and validation results is not addressed at the middle school level due to the advanced nature of these tasks. On the other hand, Documentation, a cross-cutting job function in the CT DACUM, is not emphasized in Project GUTS but would be appropriate and beneficial for middle school students to practice. At the high school level, all of the major job function categories are addressed to some extent though iterative optimization of models is rarely achieved due to constraints on students' time.

The three programs analyzed were found to prepare student for future computational thinking endeavors in STEM fields by providing student with experiences that directly mimic the work of CT-enabled STEM professionals. Several factors common to the workplace setting and the educational programs were intrinsic to offering these experiences: project-based investigations addressing real-world problems conducted by students working in teams.

6. LIMITATIONS

This study presents preliminary findings in a larger effort to define the primary work functions and tasks of computational thinking enabled STEM professional and technical workers. These would include applied and research scientists, engineers, technicians, technologists and mathematicians employed in our national STEM enterprise. Authors identify the following limitations to this study.

The sample size of computationally enabled STEM professionals was small. Although the size of the sample is consistent with literature on DACUM analyses, ongoing national validation of the importance and frequency in which CT-enabled STEM professional engage in these work functions and tasks will strengthen this work.

The sample of computationally enabled STEM professionals did not include persons describing themselves as engineers and/or technicians. As a result a second analysis is underway to align this analysis with the work functions and tasks of computationally enabled product engineers. The resulting data will be aligned and/or integrated to provide an analysis welcoming to scientists and engineers alike.

Program assessment did not take into account multiple levels of implementation of work tasks. The analysis employed a binary categorization; either the program under consideration included and promoted an activity or task or it did not. More detailed analysis would be useful.

7. IMPLICATIONS

The project's occupational profile of a CT-enabled worker and the examples of CT in action provide a common framework and an authentic structure against which CT thought leaders can test their concepts and assumptions about what CT activities, skills and knowledge STEM professionals use on the job. At the same time, these materials are designed to inform the thinking of educators. The occupational analysis provides a framework that can inform the development and sequencing of CT instruction for both academic and technical programs. Alternatively, the occupational profile provides a framework for the evaluation of programs that offer CT education. Occupational profiles have been used in the past to evaluate curricula and programs to ensure that all of the content needed to meet workplace demands/expectations were included in courses designed to lead to a specific career. [14]

In addition to providing a more inviting common language for national dialog on CT, the examples of CT are useful in helping the non-computer scientist understand the ways CT is used to perform routine tasks and solve problems in the STEM workplace. The listing of tasks can help educators understand the ways CT is applied in STEM work and analyze the evolving CT skills of their students performing routine scientific experiments/assays and solving problems in STEM classrooms and laboratories. Furthermore, the grouping of tasks into job functions, the listings of skills, knowledge and abilities as well as industry trends can help learners understand more about what it takes to succeed in America's STEM workplaces focused on discovery and innovation.

As an assessment tool, the CT DACUM provides a framework for conducting a gap analysis to assess the degree to which programs or curricula are addressing topics identified by the scientific community as important to the work of computational thinking enabled STEM professionals. Community stakeholders can use the tasks to determine how aligned STEM education programs are to workplace needs and expectations for STEM professionals work performance. Educators can determine whether specific courses or course sequences are adequately preparing students for future courses and professional endeavors. Students can analyze what skills and experiences they lack and which programs may fill those gaps. Examples of CT in Action statements can be used to guide assessment of students' CT skills.

The findings from this analysis may benefit students, educators, representatives from industry, and researchers by clarifying which computational thinking educational experiences link to workforce needs. Program managers are provided with a tool with which they can evaluate their own programs relative to preparing students for the computational thinking-enabled STEM workforce. Educators may deepen their understanding of computational thinking, its place in curricula, and its role in preparing the next generation of computational scientists.

Table 1: Job functions and tasks for the Computational Thinking enabled STEM professional annotated with three educational programs that nurture their development.

JOB FUNCTIONS and TASKS of the Computational-Thinking enabled STEM professional

Educational Programs
PG = Project GUTS (Growing Up Thinking Scientifically) middle school students using StarLogo TNG for modeling and simulation
SC = Supercomputing Challenge year long program for middle and high school students culminating in a student competition.
NMT = New Mexico Tech EPSCOR (Experimental Program to Stimulate Competitive Research) undergraduate and graduate students participate in research projects in Computational Science, High Performance Computing and Cyber-Infrastructure.

Defines			
Identifies			
A1. Identifies the scope of the problem.	PG	SC	NMT
A2. Selects relevant aspects of the problem.	PG	SC	NMT
A5. Defines assumptions and limitations.	PG	SC	NMT
A3. Identifies data sources.		SC	NMT
A4. Identifies risks of failure.		SC	NMT
A6. Identify existing tools and solutions.		SC	NMT
A7. Researches existing knowledge.		SC	NMT
A8. Determine if problem is already solved.		SC	NMT
A9. Identifies the need for a computational approach.		SC	NMT
Determines/Specifies			
B5. Specifies requirements of the solution.		SC	NMT
B6. Specifies resource requirements.		SC	NMT
B1. Determines if stakeholder has articulated the correct problem.			NMT
B2. Identifies stakeholder.			NMT
B3. Conducts needs analysis.			NMT
B4. Resolves conflicting requirements.			NMT
Models			
Designs the model			
C1. Proposes solution(s) / outcome(s) related to the problem.	PG	SC	NMT
C9. Decomposes problem / objects / processes / data.	PG	SC	NMT
C10. Abstracts the real world scenario /object into an analog.	PG	SC	NMT
C11. Abstracts physical behavior of the problem.	PG	SC	NMT
C12. Selects salient features to be included in the model.	PG	SC	NMT
C13. Designs the user interface.	PG	SC	NMT
C2. Identify why proposed solution is better than existing solutions.		SC	NMT
C3. Strategizes computational approach.		SC	NMT
C4. Identifies what modeling technique/ approach to employ.		SC	NMT
C5. Defines relationships among data (1:1, isomorphism).		SC	NMT
C6. Reverse Engineers processes and/or products.			NMT
C7. Applies systematic techniques to isolate cause & effect.			NMT
C8. Selects common properties from examples of the model / scenario / process.			NMT
Builds the model			
D1. Defines variables.	PG	SC	NMT
D2. Defines interactions among variables, objects or elements.	PG	SC	NMT
D3. Chooses an appropriate representation (e.g. data structures).	PG	SC	NMT
D4. Uses applicable existing code / technology.	PG	SC	NMT

	D5. Leverages existing solutions, algorithms.	PG	SC	NMT
	D6. Writes programs.	PG	SC	NMT
	D9. Debugs / Troubleshoots.	PG	SC	NMT
	D7. Modularizes model.		SC	NMT
	D11. Builds the User interface.		SC	NMT
	D8. Identifies sources of error.			NMT
	D10. Conducts fuzz testing (permutation testing).			NMT
Develops experimental design				
	E1. Defines parameter space.	PG	SC	NMT
	E2. Defines initial conditions under which the model operates.	PG	SC	NMT
	E4. Executes model (tests limits / sweeps parameter space) to calculate results.	PG	SC	NMT
	E5. Tests the user interface.		SC	NMT
	E3. Develops testing equipment.			NMT
Qualifies				
Verifies the model				
	F1. Verifies the model.		SC	NMT
	F2. Generates potential solutions / possibilities.		SC	NMT
	F3. Compares the behavior of the model to a known solution (or analytic solutions).		SC	NMT
	F4. Compares model with manufactured solutions.			NMT
	F5. Tests interface.			NMT
	F6. Validates the model.			NMT
	F7. Assesses the degree to which solution meets specifications / intended results.			NMT
	F8. Analyzes the sensitivity of the solution with respect to model parameters.			NMT
Refines				
Optimizes the user interface and model				
	G1. Improve input / Interface.		SC	NMT
	G2. Output / design visual representation of data.		SC	NMT
	G3. Optimize model.		SC	NMT
	G4. Use iterative refinement to focus on the problem.		SC	NMT
	G5. Propose strategies to improve solution.			NMT
	G6. Identifies risks (e.g. sub-optimal solution).			NMT
	G7. Executes improved strategies.			NMT
	G8. Selects improved solution.			NMT
Facilitates knowledge / discovery				
	H2. Observes phenomena to determine relationships (emergent behavior).		SC	NMT
	H3. Explains observed phenomena.		SC	NMT
	H6. Assesses the degree to which the solution produces new findings / knowledge.		SC	NMT
	H7. Analyzes experimental data.		SC	NMT
	H1. Generates new hypotheses that feedback to experimental design.			NMT
	H4. Discovers new relationships.			NMT
	H5. Refines experimental design.			NMT

8. CONCLUSIONS AND FUTURE WORK

The information contained in the DACUM analysis provides valuable keys to the success of the next generation of STEM innovators preparing to compete in a highly technical, global workforce. The tasks help to demystify the ways computational thinking is used in the STEM workplace by providing concrete descriptions of routine and problem solving tasks normally performed by computational thinking-enabled STEM professionals. Tying CT to concrete tasks helps educators deepen their understanding of CT and its STEM applications. This deeper understanding increases educators' ability to recognize and observe computational thinking of students in their classes. It also helps educators identify how foundational CT skills/knowledge are connected to their own curricula and addressed in K-12 both in and out of school. As we can see from the examples described herein, and from our casual observations of today's tech savvy generation, America's youth are developing valuable and marketable computational thinking skills at an early age.

The recognition of the breadth and depth of computational thinking of today's youth can result in educators purposefully cultivating computational thinking among learners of all ages both as a way to deepen the learning of difficult concepts and to prepare youth for STEM careers. As youth progress along a STEM career path the language contained in the tasks can help students articulate what they know and are able to do as they prepare for college and job interviews.

This work research also raises the following questions:

- What does the CT skills trajectory based on this work look like? What will it take to develop these skills progressions?
- Would computational thinking professionals in all career fields employ CT in similar ways? Are there core computational thinking skills that all American's should master to prepare for success in a competitive workforce focused on discovery and innovation?
- If in today's highly technical STEM workplaces, modeling and simulation are key strategies to discovery, innovation and problem solving, what role should modeling and simulation play in the education of our K-12 youth?
- If computational thinking is central to discovery and innovation in a technology rich society, how will CT be taught in K-12? Who will teach it? How will it be assessed in the K-12 system and reported?
- What does computational thinking look like in America's other workplaces?

  The profile of the Computational Thinking Enabled STEM Professional resulting from the DACUM analysis adds useful language to the ongoing national dialog on computational thinking, a framework that can contribute to the evolution of CT education at all levels.

Typically, the next steps in this process would be to work with representatives of scientific industries and the expert panel to develop rubrics for each of the 11 job functions (3 cross-cutting and 8 CT categories) that articulate current employer expectations for "proficiency" in computational thinking. Project staff would organize additionally gathered "in action" examples along with the language generated throughout the DACUM process, into levels that concretize and contextualize CT from "novice" to "above proficiency". The rubrics would be used by curriculum developers to sequence instruction, by educators as a tool to "observe" and record computational thinking in action in their classes, and by learners to self-evaluate their progress towards workplace proficiency. The rubrics could also be used by employers to guide the ongoing professional development of scientists and engineering as they move from junior to senior levels, and novice to expert in computational thinking in STEM workplaces.

9. Acknowledgements

The authors wish to thank Joe Ippolito, Dr. Lorie Liebrock, Dr. Nadine Miner, and David Kratzer for their contributions to the Supercomputing 2011 panel on linking CT educational programs in K-20 with workforce needs. We also with to thank the middle and high school teachers who have served as Project GUTS club leaders and Supercomputing Challenge team sponsors for their many contributions to support computational thinking in K-12 education.

10. References

[1] Allan, W., Coulter, B., Denner, J., Erickson, J., Lee, I., Malyn-Smith, J., Martin, F. 2010. *Computational Thinking for Youth.* A white paper of the ITEST Learning Resource Center Working Group on Computational Thinking, Unpublished manuscript, Education Development Center, Inc.

[2] Committee for the Workshops on Computational Thinking, 2010. *Report of a Workshop on the Scope and Nature of Computational Thinking*, National Research Council, Washington, D.C., National Academies Press.

[3] Cuny, J.E., Snyder, L., Wing, J.M., 2010. *Computational Thinking: A Definition*. Unpublished manuscript

[4] Dahms, A.S., Leff, J.A. Industry Expectations for Entry-Level Technical Workers, *Biochemistry and Molecular Biology Education*, 30, 4 (2002) 260-264

[5] Denner, J., Bean, S., Martinez, J., Girl Game Company: Engaging Latina Girls in Information Technology, *Afterschool Matters*, 8 (2009) 26-35.

[6] Hofstader, R., Chapman, K. 1997. *Foundations for Excellence in the Chemical Process Industries. Voluntary Industry Standards for Chemical Process Industries Technical Workers* (ERIC Document Reproduction Service No. ED405480), American Chemical Society, Washington, D.C.

[7] *Integrating IT Skills in Law, Public Safety, Corrections and Security Career Programs*, second ed. 2008. Education Development Center, Inc., Newton, MA (Accessed 1/11/12 from IT Across Careers Web site, http://itac.edc.org/)

[8] Ippolito, J., Latcovich, M., Malyn-Smith J. 2008. *In Fulfillment of their Mission: The Duties and Tasks of a Roman Catholic Priest*, NCEA Publication, New York, NY.

[9] Isbell, C., Stein, L.A., Cutler, R., Forbes, J., Fraser, L , Impagliazzo, J. et al. Re(defining) Computing Curricula by Re(defining) Computing, *ACM SIGCSE Bulletin*, 41, 4 (2009) 195-207.

[10] Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., and Werner, L. (2011). Computational Thinking for Youth in Practice, ACM Inroads Vol. 2 No. 1.

[11] Leff, J. 1995. *Gateway to the Future: Skill Standards for the Bioscience Industry*. Education Development Center, Inc., Newton, MA.

[12] Leff, J., Malyn-Smith, J., Hiles, E. 1999. *Making Skill Standards Work: Highlights from the Field*. Education Development Center, Inc., Newton, MA.

[13] Moursund, D. 2009. *Computational Thinking*, IAE-pedia.org. Available online at *http://iae pedia.org/Computational_Thinking*. Accesed August 8, 2010.

[14] Norton, R.E. 1997. *DACUM Handbook*, second ed., Publications, Center on Education and Training for Employment, Columbus, OH.

[15] *Profile of a Technology-Enabled Investigator (Duties and Tasks)*, 2009. Education Development Center, Inc., Newton, MA.

[16] Resnick, M. 2002. Rethinking Learning in the Digital Age. In *The Global Information Technology Report: Readiness for the Networked World*, G. Kirkman, Ed. Oxford University Press.

[17] Resnick, M. 2006. Computer as Paintbrush: Technology, Play, and the Creative Society. In *Play = Learning: How play Motivates and Enhances Children's Cognitive and Social-Emotional Growth*, Singer, D., Golikoff, R., Hirsh-Pasek, K. (eds.). Oxford University Press.

[18] *Rubrics to Access Basic IT User Skills*, 2006. Education Development Center, Inc. Newton, MA, 2006. (Accessed 1/11/12 from IT Across Careers Web site: http://itac.edc.org/)

[19] Rusk, N., Resnick, M., Berg, R., Pezalla-Granlund, M. New Pathways into Robotics: Strategies for Broadening Participation, *Journal of Science Education and Technology*, 17, 1 (2008) 59-69.

[20] Taylor, M., Bradley, V., Silver, J., Leff, J., Malyn-Smith, J. 1996. *Using the Community Support Skill Standards: A Guidebook for Human Service Educators & Trainers*, Human Services Research Institute, Cambridge, MA.

[21] Taylor, M., Bradley, V., Warren, Jr., R. (Eds.) 1996. *The Community Support Skill Standards: Tools for Managing Change and Achieving Outcomes. Skill Standards for Direct Service Workers in the Human Services* (ERIC Document Reproduction Service No. ED400646), Human Services Research Institute, Cambridge, MA.

[22] Wing, J., Computational Thinking, *Communications of the ACM*, 49, 3 (2006) 33-35.

[23] Wing, J.M. 2009. Computational Thinking, Presentation at *The SIGCT Forum of the ISTE Annual National Educational Computing Conference*, Washington, D.C