Supporting Assessment Practices in Secondary Computer Science Education

2014 CSTA Conference
St. Charles, IL

Eric Snow, Marie Bienkowski, Daisy Rutstein
Slides and handouts available:

http://pact.sri.com/?page_id=28
Overview

- **Assessment Argument**
  - Evidence-Centered Assessment Design
  - Assessment Design Patterns

- **Context: Principled Assessment for Computational Thinking (PACT)**
  - Designing, developing and validating assessments for Exploring Computer Science
Overview

- Handouts
  - Stylized Evidence-Centered Assessment Design Flowchart
  - Computational Thinking Practices and ECS Design Pattern Examples

Slides and handouts available:

http://pact.sri.com/?page_id=28
Teaching Questions

- What are the main knowledge and skills students should learn?
- What classroom activities will help students learn the desired knowledge and skills?
- What evidence from classroom activities will help best determine how well students are learning the desired knowledge and skills?
Assessment Questions

- What complex of knowledge, skills, or other attributes should be assessed?
- What behaviors or performances should reveal those constructs?
- What tasks or situations should elicit those behaviors?
Assessment Questions

- What complex of knowledge, skills, or other attributes should be assessed?
- What behaviors or performances should reveal those constructs?
- What tasks or situations should elicit those behaviors?
Operationalizing the Assessment Argument

How to we get from an assessment argument to an assessment?

What complex of knowledge, skills, or other attributes should be assessed?

What behaviors or performances should reveal those constructs?

What tasks or situations should elicit those behaviors?
Operationalizing the Assessment Argument

What complex of knowledge, skills, or other attributes should be assessed?

What behaviors or performances should reveal these constructs?

What tasks or situations should elicit these behaviors?

Evidence-Centered Assessment Design (ECD)
Evidence-Centered Assessment Design

- ECD is a framework for assessment design and development:
  - Views assessment as a process of gathering evidence to support an argument about what a student knows and can do
  - Provides a structure for an approach that incorporates validity evidence into the assessment design process
  - Documents what decisions have been made with regards to the assessment and the justification for those decisions
From Mislevy & Riconscente, 2006
Domain Analysis

What is important about this domain?
What work and situations are central in this domain?
What KRs are central to this domain?

- From Mislevy & Riconscente, 2006
Domain Analysis

- What is important about this domain?
- What work and situations are central in this domain?
- What KRs are central to this domain?

Domain Modeling

- How do we represent key aspects of the domain in terms of assessment argument?
From Mislevy & Riconscente, 2006
From Mislevy & Riconscente, 2006

- Domain Analysis
  - What is important about this domain?
  - What work and situations are central in this domain?
  - What KRs are central to this domain?

- Domain Modeling
  - How do we represent key aspects of the domain in terms of assessment argument?

- Conceptual Assessment Framework
  - Design structures: Student, evidence, and task models.

- Assessment Implementation
  - Manufacturing “nuts & bolts”: authoring tasks, automated scoring details, statistical models.

© 2013 SRI International - Company Confidential and Proprietary Information
From Mislevy & Riconscente, 2006

Assessment Delivery
Students interact with tasks, performances evaluated, feedback created.

Assessment Implementation
Manufacturing “nuts & bolts”: authoring tasks, automated scoring details, statistical models..

Conceptual Assessment Framework
Design structures: Student, evidence, and task models..

Domain Modeling
How do we represent key aspects of the domain in terms of assessment argument..

Domain Analysis
What is important about this domain? What work and situations are central in this domain? What KRs are central to this domain?
From Mislevy & Riconscente, 2006
Domain Analysis: Computational Thinking Practices

- What is important about the computational thinking practices domain?
- What work and situations are central to the computational thinking domain?
- What KRs are central to the computational thinking domain?
Domain Analysis Resources for Computational Thinking

Literature
- National Academies Report: Computer Science: Reflections on the Field, Reflections from the Field
- SIGCSE, CSTA, ITiCSE, Journal of Computing in Higher Education, Educational Researcher
- Jeanette Wing & others; National Academies Workshop on Pedagogical Aspects of Computational Thinking

Standards/Curriculum
- CSTA (2011). CSTA K-12 Computer Science Standards
- Exploring Computer Science
- NGSS, CCSS
Computational Thinking Practices

New high school curricula (e.g., CS Principles, ECS) emphasize “computational thinking practices”.

This reflects an orientation toward not just an internal, individual “thinking” but “ways of being and doing” that students should demonstrate when learning and exhibiting computer science knowledge, skills, and attitudes.
Computational Thinking Practices

The **Common Core State Standards** include standards related to computational thinking practices in mathematics such as problem solving and abstraction.

The **Next Generation Science Standards** include standards dealing with engineering design and describe “using mathematical and computational thinking” as an essential practices for modeling and analyzing and interpreting data.
CSE & CSE Literature / Standards & Curriculum

CS Concepts
# Computational Thinking Practices

<table>
<thead>
<tr>
<th>Example CS Concepts</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td></td>
</tr>
<tr>
<td>Programming</td>
<td></td>
</tr>
<tr>
<td>Recursion</td>
<td></td>
</tr>
<tr>
<td>Abstraction</td>
<td></td>
</tr>
<tr>
<td>Debugging / Testing</td>
<td></td>
</tr>
<tr>
<td>Variables</td>
<td></td>
</tr>
</tbody>
</table>
CSE & CSE Literature / Standards & Curriculum

- CS Concepts
- Inquiry Skills
# Computational Thinking Practices

<table>
<thead>
<tr>
<th>Example CS Concepts</th>
<th>Example Inquiry Skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Evaluate</td>
</tr>
<tr>
<td>Programming</td>
<td>Explore</td>
</tr>
<tr>
<td>Recursion</td>
<td>Analyze</td>
</tr>
<tr>
<td>Abstraction</td>
<td>Explain</td>
</tr>
<tr>
<td>Debugging / Testing</td>
<td>Elaborate</td>
</tr>
<tr>
<td>Variables</td>
<td>Model</td>
</tr>
</tbody>
</table>
Computational Thinking Practices

CSE & CSE Literature / Standards & Curriculum

- Inquiry Skills
- CS Concepts
- Noncognitive Skills
## Computational Thinking Practices

<table>
<thead>
<tr>
<th>Example CS Concepts</th>
<th>Example Inquiry Skills</th>
<th>Example Noncognitive Skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Evaluate</td>
<td>Communication</td>
</tr>
<tr>
<td>Programming</td>
<td>Explore</td>
<td>Teamwork/collaboration</td>
</tr>
<tr>
<td>Recursion</td>
<td>Analyze</td>
<td>Leadership</td>
</tr>
<tr>
<td>Abstraction</td>
<td>Explain</td>
<td>Self-efficacy</td>
</tr>
<tr>
<td>Debugging / Testing</td>
<td>Elaborate</td>
<td>Self-concept</td>
</tr>
<tr>
<td>Variables</td>
<td>Model</td>
<td>Persistence</td>
</tr>
</tbody>
</table>
### Computational Thinking Practices

<table>
<thead>
<tr>
<th>Example CS Concepts</th>
<th>Example Inquiry Skills</th>
<th>Example Noncognitive Skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Evaluate</td>
<td>Communication</td>
</tr>
<tr>
<td>Programming</td>
<td>Explore</td>
<td>Teamwork/collaboration</td>
</tr>
<tr>
<td>Recursion</td>
<td>Analyze</td>
<td>Leadership</td>
</tr>
<tr>
<td>Abstraction</td>
<td>Explain</td>
<td>Self-efficacy</td>
</tr>
<tr>
<td>Debugging / Testing</td>
<td>Elaborate</td>
<td>Self-concept</td>
</tr>
<tr>
<td>Variables</td>
<td>Model</td>
<td>Persistence</td>
</tr>
</tbody>
</table>

Analyze their computational work and the work of others
### Computational Thinking Practices

<table>
<thead>
<tr>
<th>Example CS Concepts</th>
<th>Example Inquiry Skills</th>
<th>Example Noncognitive Skills</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>Evaluate</td>
<td>Communication</td>
</tr>
<tr>
<td>Programming</td>
<td>Explore</td>
<td>Teamwork/collaboration</td>
</tr>
<tr>
<td>Recursion</td>
<td>Analyze</td>
<td>Leadership</td>
</tr>
<tr>
<td>Abstraction</td>
<td>Explain</td>
<td>Self-efficacy</td>
</tr>
<tr>
<td>Debugging / Testing</td>
<td>Elaborate</td>
<td>Self-concept</td>
</tr>
<tr>
<td>Variables</td>
<td>Model</td>
<td>Persistence</td>
</tr>
</tbody>
</table>

Collaborate with peers on computing activities
## Computational Thinking Practices

<table>
<thead>
<tr>
<th>Exploring Computer Science</th>
<th>Computer Science Principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyze the effects of developments in computing</td>
<td>Analyzing Problems and Artifacts</td>
</tr>
<tr>
<td>Design and implement creative solutions and artifacts</td>
<td>Developing Computational Artifacts</td>
</tr>
<tr>
<td>Apply abstractions and models</td>
<td>Abstracting</td>
</tr>
<tr>
<td>Analyze their computational work and the work of others</td>
<td>Analyzing Problems and Artifacts</td>
</tr>
<tr>
<td>Connect computation with other disciplines</td>
<td>Connecting Computing</td>
</tr>
<tr>
<td>Communicate computational thought processes, procedures and results to others</td>
<td>Communicating</td>
</tr>
<tr>
<td>Collaborate with peers on computing activities</td>
<td>Collaborating</td>
</tr>
</tbody>
</table>
Context: Principled Assessment for Computational Thinking (PACT)

Domain Modeling: Computational Thinking Practices

How do we conceptualize and represent key aspects of the computational thinking domain in terms of an assessment argument?
What are Design Patterns?

- Solution to a problem that occurs repeatedly in our environment
- Specified at a level of generality that the underlying approach can be applied across many situations while adapting to the particulars of each case
- Shows general relationships and interactions without specifying details
What are Design Patterns?

- Design Patterns in Computer Science & Software Engineering

© 2013 SRI International - Company Confidential and Proprietary Information
What are Design Patterns?

In this computer science and software engineering, design patterns:

- Help programmers tackle complex problems that recur in different guises
- Provide structured insights into conceptual problems and solutions above the level of specific programming languages and implementation environments
- Object-oriented design patterns
Domain Modeling & Design Patterns

- Specifies and organizes assessment argument in narrative form based on information from Domain Analysis
- High-level representation of assessment argument
- Transition point between specialized knowledge about the domain to the specialized knowledge about the more technical machinery of assessment
Motivation for Assessment Design Patterns

• Serve as an interstitial document that allows different assessment stakeholder groups to understand important aspects of assessment
• They lay out a design space for developers
  – Choices, connections, coherence, tradeoffs, examples
• Attributes reflect assessment argument structure
• Can improve both efficiency & validity
Developing Design Patterns

- An iterative, interdisciplinary process requiring:
  - Content experts
  - Educators
  - Assessment experts
  - Practitioners
  - Multiple sources of information (e.g., education research, curriculum examples, existing standards, industry trends, policy documents)
Assessment Design Pattern Attributes

Overview

• Description of construct being modeled in design pattern.

Focal Knowledge, Skills & Attributes (KSAs)

• The primary KSAs targeted by the design pattern. What we want to make inferences about.

Additional KSAs

• Other KSAs that may be required for successful performance on the assessment tasks.
Potential Observations

• *Features* of the things students say, do, or make that constitute the evidence.

Potential Work Products

• Some possible things one could see students doing that would give evidence about the KSAs.

Characteristic Features

• Aspects of assessment situations that are likely to evoke the desired evidence.

Variable Features

• Aspects of assessment situations that can be varied in order to shift difficulty or emphasis.
Example Design Patterns

- Which type of design pattern is best depends on a number of factors:
  - Number of domains being assessed
  - Complexity of inquiry practice(s) being modeled
  - Degree of scaffolding needed by item writers
Context: Principled Assessment for Computational Thinking (PACT)

- Developed design patterns for:
  - Six computational thinking practices (CTPs)
    - Analyze the effects of developments in computing
    - Design and implement creative computational solutions and artifacts
  - Design and apply abstractions and models
  - Analyze computational work (both own and others)
  - Communicating computational thought processes, procedures and results to others
  - Collaborate with peers on computing activities
Context: Principled Assessment for Computational Thinking (PACT)

- Developed design patterns for:
  - ECS units 1-4
    - Human-computer interaction
    - Problem solving
    - Web design
    - Introduction to programming
Context: Principled Assessment for Computational Thinking (PACT)

<table>
<thead>
<tr>
<th>ECS Units</th>
<th>Computational Thinking Practices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit 1: Human-Computer Interaction</td>
<td>• Analyze the effects of developments in computing.</td>
</tr>
<tr>
<td>Unit 2: Problem Solving</td>
<td>• Design and implement creative solutions and artifacts.</td>
</tr>
<tr>
<td></td>
<td>• Apply abstractions and models.</td>
</tr>
<tr>
<td></td>
<td>• Analyze their computational work and the work of others.</td>
</tr>
<tr>
<td>Unit 3: Web Design</td>
<td>• Design and implement creative solutions and artifacts.</td>
</tr>
<tr>
<td></td>
<td>• Analyze their computational work and the work of others.</td>
</tr>
<tr>
<td></td>
<td>• Connect computation with other disciplines.</td>
</tr>
<tr>
<td>Unit 4: Introduction to Programming</td>
<td>• Design and implement creative solutions and artifacts.</td>
</tr>
<tr>
<td></td>
<td>• Apply abstractions and models.</td>
</tr>
<tr>
<td></td>
<td>• Analyze their computational work and the work of others.</td>
</tr>
</tbody>
</table>
## Context: Principled Assessment for Computational Thinking (PACT)

<table>
<thead>
<tr>
<th>ECS Unit / CTP</th>
<th>Example Unit FKSAs</th>
<th>Example CTP FKSAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit 1: Human-Computer Interaction</td>
<td>• Students are able to explain why an object is or is not a computer.</td>
<td>• Ability to describe the characteristics of a computer (including what it means for a computer to be “intelligent”).</td>
</tr>
<tr>
<td>Analyze the effects of developments in computing.</td>
<td>• Students are able to evaluate the implications of a form of data exchange on social interactions.</td>
<td>• Ability to analyze the effects of computing on society within economic, social, and cultural contexts.</td>
</tr>
<tr>
<td></td>
<td>• Students are able to explain how computing innovation has led to new types of legal and ethical concerns.</td>
<td>• Ability to evaluate legal and ethical concerns raised by computing-enabled innovations.</td>
</tr>
</tbody>
</table>
Example Design Pattern, Unit 1: Human-Computer Interaction

Overview (from curriculum)

In Unit 1 students are introduced to the major components of the computer, including: input, output, memory, storage, processing, software, and the operating system. Students consider how Internet elements are organized, engage in effective searching, and focus on productive use of email. Fundamental notions of Human Computer Interaction (HCI) and ergonomics are introduced.

Students learn that “intelligent” machine behavior is not “magic” but is based on algorithms applied to useful representations of information. Students learn the characteristics that make certain tasks easy or difficult for computers, and how these differ from those that humans characteristically find easy or difficult. Students gain an appreciation for the many ways (types of use) in which computers have had an impact across the range of human activity, as well as for the many different fields in which they are used. Examples illustrate the broad, interdisciplinary utility of computers and algorithmic problem solving in the modern world.
Example Design Pattern, Unit 1: Human-Computer Interaction

Example Focal Knowledge, Skills & Attributes (KSAs)

- Students are able to explain why an object is or is not a computer.
- Students are able to evaluate the implications of a form of data exchange on social interactions.
- Students are able to explain how computing innovation has led to new types of legal and ethical concerns.
### Example Design Pattern, Unit 1: Human-Computer Interaction

#### Example FKSA

Students are able to explain why an object is or is not a computer.

#### Example Potential Work Product

An explanation of why an object is or is not a computer.

#### Example Potential Observations

Appropriateness of the explanation of why an object is or is not a computer.

- Did the student correctly identify aspects of the object that relate to aspects of a computer?
- Did the student correctly identify aspects of a computer that the object lacks?
## Context: Principled Assessment for Computational Thinking (PACT)

### Example Design Pattern, Unit 1: Human-Computer Interaction

<table>
<thead>
<tr>
<th>Example FKSA</th>
<th>Example Characteristic Features</th>
<th>Example Variable Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Students are able to explain why an object is or is not a computer.</td>
<td>The student must be presented with an object</td>
<td>Whether the object could be considered a computer or not.</td>
</tr>
<tr>
<td></td>
<td>The object must have clear characteristics that allow the evaluation of whether it is a computer.</td>
<td>Whether students would be able to argue either way if the object is a computer or not.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The degree to which the important characteristics are explicitly stated in the problem or must be inferred by the test taker.</td>
</tr>
</tbody>
</table>
You and your friend are debating about whether or not a microwave is a computer.

Explain why you think a microwave IS or is NOT a computer. In your response describe at least TWO characteristics of a computer that support your explanation.

A microwave (check one) _____ IS a computer _____ is NOT a computer

Explain:
You and your friend are debating about whether or not a microwave is a computer. Explain why you think a microwave IS or is NOT a computer. In your response describe at least TWO characteristics of a computer that support your explanation.

A microwave (check one) _____ IS a computer _____ is NOT a computer

Explain:

**FKSA:**

Students are able to explain why an object is or is not a computer.
Example Assessment Task, Unit 1: Human-Computer Interaction

You and your friend are debating about whether or not a microwave is a computer.

Explain why you think a microwave IS or is NOT a computer. In your response describe at least TWO characteristics of a computer that support your explanation.

A microwave (check one) _____ IS a computer    _____ is NOT a computer

Potential Work Product:

An explanation of why an object is or is not a computer.
Example Assessment Task, Unit 1: Human-Computer Interaction

You and your friend are debating about whether or not a microwave is a computer.

Explain why you think a microwave IS or is NOT a computer. In your response describe at least TWO characteristics of a computer that support your explanation.

A microwave (check one) _____ IS a computer   _____ is NOT a computer

Explain:

**Characteristic Features:**

The student must be presented with an object

The object must have clear characteristics that allow the evaluation of whether it is a computer.
Example Assessment Task, Unit 1: Human-Computer Interaction

You and your friend are debating about whether or not a microwave is a computer.

Explain why you think a microwave IS or is NOT a computer. In your response describe at least TWO characteristics of a computer that support your explanation.

A microwave (check one) _____ IS a computer _____ is NOT a computer

Explain:

**Variable Features:**

The degree to which the important characteristics are explicitly stated in the problem or must be inferred by the test taker.

Whether students would be able to argue either way if the object is a computer or not.
**Context: Principled Assessment for Computational Thinking (PACT)**

<table>
<thead>
<tr>
<th>Example Scoring Rubric, Scoring Guidance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Points Possible for a): 2</td>
</tr>
<tr>
<td>The points are given based on the explanation – the explanation should relate what a microwave can do to an aspect of a computer (the aspect could be something other than what we specified here).</td>
</tr>
<tr>
<td>Defining aspects (characteristics) can include things such as reducing human effort, taking inputs, giving outputs, stores data, processes information</td>
</tr>
<tr>
<td>1 point for listing 2 valid aspects of a computer. Response must name specific terms, such as input, output, process, data, programming, instructions, etc.</td>
</tr>
<tr>
<td>1 point for relating microwave to the aspect(s) they identify.</td>
</tr>
<tr>
<td>If they only name one aspect and relate it to the microwave then they should be scored 1 point.</td>
</tr>
<tr>
<td>Misconceptions about microwaves and how they work don’t count against score.</td>
</tr>
</tbody>
</table>
### Context: Principled Assessment for Computational Thinking (PACT)

<table>
<thead>
<tr>
<th>Example Scoring Rubric, Scoring Guidance w/ Potential Observation</th>
</tr>
</thead>
</table>
The points are given based on the explanation – the explanation needs to relate what a microwave can do to an aspect of a computer (the aspect could be something other than what we specified here).

Defining aspects (characteristics) can include things such as reducing human effort, taking inputs, giving outputs, stores data, processes information.

1 point for listing 2 valid aspects of a computer. Response must name specific terms, such as input, output, process, data, programming, instructions, etc.

1 point for relating microwave to the aspect(s) they identify.

Did the student correctly identify aspects of the object that relate to aspects of a computer?
Context: Principled Assessment for Computational Thinking (PACT)

Example Scoring Rubric, Exemplar Responses

Example 1 point response:
“It is programmed to heat up or unfreeze food.” (+1 programmed to heat up or unfreeze)

Example 2 point responses:
"The microwave has data and it does have a processor because when you push the time (numbers) show up on the screen and when I push start, it started the time starts and the food starts cooking".

“Yes, it is a computer because it is given command when we press buttons on it (+1 for has input by pressing a button) and put a timing on the food.” (+1 for has output by timing food)

“A microwave is a computer because its programmed (+1 program to heat up food) to help us heat up our food when its cold we can also program the time of day to let us know what time it is.” (+1 program the time)
Assessment Validity

Extent to which assessment argument is supported by different types of evidence:

• Test content
• Student response processes
• Internal structure
• Relationships with other variables
• Consequences of test use
## Closing Comments

<table>
<thead>
<tr>
<th>Teaching Questions</th>
<th>Assessment Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>What are the main knowledge and skills students should learn?</td>
<td>What complex of knowledge, skills, or other attributes should be assessed?</td>
</tr>
<tr>
<td>What evidence from classroom activities will help best determine how well students are learning the desired knowledge and skills?</td>
<td>What behaviors or performances should reveal those constructs?</td>
</tr>
<tr>
<td>What classroom activities will help students learn the desired knowledge and skills?</td>
<td>What tasks or situations should elicit those behaviors?</td>
</tr>
</tbody>
</table>
Closing Comments

- Assessment arguments and evidence-centered assessment design.
- Assessment design patterns.
- Slides and handouts available: http://pact.sri.com/?page_id=28
Thank You!

© 2012 Principled Assessment of Computational Thinking (PACT). Produced by the Center for Technology in Learning at SRI International with support from the National Science Foundation under contract numbers, CNS-1132232 and CNS-1240625 and CNS-0943507 to the University of Oregon. Any opinions, findings, conclusions, or recommendations expressed are those of the authors and do not necessarily reflect the views of the National Science Foundation.