Braided Teaching in Secondary CS Education: Contexts, Continuity, and the Role of Programming

Arno Pasternak and Jan Vahrenhold
Faculty of Computer Science
Technische Universität Dortmund, Germany
Overview

1. Introduction
2. Theoretical Framework and Related Work
3. Practical Background and Implications
4. Conclusions
Observations

Role of Computer Science in (Lower) Secondary Education:
- Diminishing (or non-existing) curriculum credit for CS.
- “Information and Communication Technology vs. CS” debate.
- Slight variations across states and countries, yet same general theme.

Breadth of Computer Science as a Subject:
- Ranging from (Electrical) Engineering to (Discrete) Mathematics.
  - Can reach out to students with varying aptitudes and interests.
  - Wealth of subject matters and application contexts.
- One of the distinguishing features (not only in secondary education).

Alas, hard (if not impossible) to teach along a spiral curriculum.
Spiral Curriculum & Computing Mechanics

Spiral Curriculum [Bruner, 1960]:
- “Any subject can be taught effectively in some intellectually honest form to any child at any stage of development.”
- Multiple iterations needed to reach an understanding at adult level.
- Obvious “depth-versus-breadth” problem...

Computing Mechanics [Denning, 2003]:
- Structure and operation of computations.
- Windows to look at core technologies:
  “Although the views through the edges of windows overlap, the view through the centers is distinctive.”
Observations:

- Majority of courses in secondary education focus on programming.
- Tremendous progress with (meta-)microworlds for outreach/teaching.
  - Alice, Greenfoot, Scratch, [insert your favorite here] (to name a few).
- Community (still) struggling with “OO first/late/never”.

Recent Debate in CACM on the “CS = Programming” Portrait.

What the ACM K-12 Model Curriculum says:

“While programming is a central activity in computer science, it is only a tool that provides a window into a much richer academic and professional field.” [Tucker et al., 2004, p. 6]
Intermediate Summary

Observations:

- Multiple responses to diminishing interest in/credit for CS in secondary education.
- Breadth of the subject matter versus spiral curriculum.
- Multiple viewpoints w.r.t. the role of programming.

Objectives:

- Propose a new way of thinking about implementing CS curricula...
- ...that leaves as much leeway to educators as possible
- ...and that is compatible with established procedures.
- Finally: give evidence for why it should be more than just “Philosophy”.

Arno Pasternak and Jan Vahrenhold  
Braided Teaching in Secondary CS Education
Overview

1. Introduction

2. Theoretical Framework and Related Work

3. Practical Background and Implications

4. Conclusions
Strands

Definition 2.1
A strand is a sequence of items addressed in class that satisfies the following criteria:

1. The items can be assigned to a well-defined subject matter (by their structure or their content). → Coherency
2. The subject matter is identifiable and recognizable to the students throughout the sequence. → Transparency
3. The subject matter is being presented from more than one point of view or embedded in more than one context. → Variance
4. The sequence of items is addressed in more than one teaching unit. → Redundancy

Note: A strand does not require a particular order to be imposed on its items.
Objective and Subjective Criteria:

1. “The items can be assigned to a well-defined subject matter.”
   - Purely objective criterion; can be verified easily.

2. “[…] identifiable/recognizable to the students throughout sequence.”

3. “[…] presented from more than one point of view […].”

4. “[…] addressed in more than one teaching unit. […]”
   - In the responsibility of the educator; situated. Cf. this SIGCSE’s keynotes.
   - Subject to structural and pedagogical considerations by the educator.
   - Subject to external factors such as curriculum or “teaching tradition”.

Implications:

- (3) → Aptitudes/interests, cf. “windows of computing mechanics”.
- (4) → Spiral curriculum, cf. this morning’s keynote (“spaced in time”).
Examples

Operating Systems
- Basic functions
- GUI-based applications
- Shell
- Remote access

(Semi-)Structured Data
- (X)HTML
- XML
- SVG
- XSLT
- CGI
- XPath

Stand organized by content.
Strand organized by structure.

A strand is different from the following:

- ...“Fundamental idea” [Schwill, 1997]
  - Mastery allows for transfer of principles (“(Semi-)Structured Data”).
  - Organization exclusively from the perspective of the educator.

- ...“Threads™” [Furst et al., 2007]
  - Collection of sequences of courses to fulfill college(!) curriculum.
  - Organization by the educators, selection by the students.
Why Contextualized Teaching?

- Increased motivation & effectiveness [Tew et al., 2008].

“On the Nature of ‘Context’ in Chemistry Education” [Gilbert, 2006]:

- There is no single notion of “context”.
- Higher secondary education and college:
  - “Reciprocity between concepts and applications.” (MediaComputation)
- Lower secondary education:
  - “Topics and [...] activities [...] [important] to [...] the society.”

Contexts Compared to Strands:

- Coherent time frame for presenting items from (more than) one strand.
Definition 2.2

Braided teaching is the process of covering the contents of a given curriculum by a collection of strands that are interlaced wherever appropriate.

Observation:
- By definition (of a strand): no single “programming” unit allowed.
Students, Computers, and Programming

Observation:

- Programming ≈ using the computer to solve otherwise seemingly infeasible problems.

In the Good Old Days™:

- In-depth treatment of a small number of topics possible.
- Few (if any) software available in the public domain.

Today:

- Increasingly complex contexts covered breadth-first only.
- Much free and possibly open source software available; can download a program for seemingly any problem touched upon in high school.
Considering “Programming” as a Strand:

- Admittedly, not every concept can be taught from multiple viewpoints.
- Attributes the role delineated in the ACM K-12 Curriculum.

> “While programming is a central activity in computer science, it is only a tool that provides a window into a much richer academic and professional field.” [Tucker et al., 2004, p. 6]

- “One among many”-status prohibits “CS = Programming” image.
- Ultimately, allows an educator to choose a programming language based upon its usability for a particular course design.

Disclaimer: This may or may not be possible in a particular country / state / school district and/or lead to more philosophical discussions...but where to present such considerations if not in a “Philosophy” session?
Overview

1. Introduction
2. Theoretical Framework and Related Work
3. Practical Background and Implications
4. Conclusions
Professional Background

Bio in a Nutshell:

- Teaching qualification for Computer Science, Physics, and Mathematics.
- Almost 30 years of teaching experience on secondary level.

Fritz-Steinhoff-Gesamtschule Hagen:

- Secondary school (grades 5-13) with 1,500 students.
- CS has been taught at FSG Hagen since (at least) 1983.
  - Unfortunately, CS is neither mandatory nor (in general) worth full credit.
  - Fortunately, the latter is going to change at FSG :-).
Observations:

- Breadth of topics increased; course volume did not.
- Common component: Programming.
Observation:

- Programming ≈ using the computer to solve otherwise seemingly infeasible problems.

Today’s Classroom Situation:

- Increasingly complex contexts covered breadth-first only.
  
- In lower and middle secondary education:
  - Students very well aware of “Computers and the internet”.
  - Programming (in the above sense) increasingly difficult.
  - Still possible: Working with “elementary” tasks and tools.

- Need to relate subject matters to (the students’) everyday life.

- Present meaningful, yet feasible programming tasks.
  - Remember: Programming is a tool, not a goal in itself.
### Example: Contexts for Programming Tasks

<table>
<thead>
<tr>
<th>Context</th>
<th>Subject Matter</th>
<th>Strand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remotely maintain a client or server</td>
<td>Scripting &amp; animated GIFs</td>
<td>Multimedia</td>
</tr>
<tr>
<td>Discuss security of online shopping</td>
<td>XHTML</td>
<td>(Semi-)Structured Data</td>
</tr>
<tr>
<td>Create guest book for school's web server</td>
<td>XML data base</td>
<td>(Semi-)Structured Data</td>
</tr>
<tr>
<td>Maintain a database for an MP3 collection</td>
<td>CGI scripting</td>
<td>(Semi-)Structured Data / Networks</td>
</tr>
<tr>
<td>Create web page for field trip</td>
<td>Cryptology</td>
<td>Networks</td>
</tr>
<tr>
<td>Produce flip book animations</td>
<td>Shell scripting</td>
<td>Operating Systems / Networks</td>
</tr>
</tbody>
</table>
Pedagogical Imperative: “Programming as a Tool”
- Learning a particular language is not a first-order objective.
- Choose subject matters first, then a fitting language.
- In our situation: Contexts at times interactive and/or graphical.

One Possible Language of Choice: Tcl/Tk\(^\ast\) [Ousterhout, 1994]
- Highly interactive language; integrates with OS’s shell.
- Allows for realization of CGI scripts and (Java-)applet-like scripts.
- Allows for (does not enforce) object-orientation via graphics toolkit Tk.

\(^\ast\): Based on your experiences, you may prefer X over Tcl/Tk. This is perfectly fine!
From Theory to Practice:
- New course “Computer Science and Physics”.
  - Target group: Grades 6 to 10; 3 hours/week.
  - Elective course, yet worth full curriculum credit.
  - Start: Summer 2010.
- Course designed according to the “braided teaching” philosophy.
- Touching points with contexts from Physics.

Project Evaluation:
- Evaluation done by fellow PhD student (I’m teaching, after all).
- Control group design possible (→ elective course).
Overview

1. Introduction
2. Theoretical Framework and Related Work
3. Practical Background and Implications
4. Conclusions
Summary

Braided Teaching:

- Organization along strands.
  - Coherency, transparency, variance, redundancy.
- Programming as a tool → strand.
- Potential:
  - Teaching along a spiral curriculum.
  - More leeway for educators’ decisions.

Future Work:

- Transform philosophy into practice and non-useless truth.
- Investigate extension to higher secondary and college education.
Bibliography


Example Course Sequence

Course for Grades 9 and 10:

- Strands interlaced throughout the course:
  - Operating Systems, Multimedia, (Semi-)Structured Data, Networks.

Sequence in a Nutshell:

- Using shell commands for working with files and directories.
- Exemplary installation of a small software package.
- Automating tasks by using shell scripts.
- (X)HTML for describing web pages and representing structured data.
- Uploading and maintaining pages on a web server.
- Using CGI for automated document creation.
- Representing and creating animated graphics with SVG.
The Flipbook Example
Situation in Germany:

- Pretty much the same as everywhere:
  - CS education in each state’s responsibility.
  - Diminishing curriculum credit, only few fully qualified teachers per school.

- But there is a twist: “Educational Standards for Computer Science in Lower Secondary Education” (see, e.g. [Brinda et al., 2009]).
  - Outcome-based standards, modeled after NCTM’s standards for Mathematics.
  - Result of a multi-year effort of 70+ educators and researchers.
  - Standards (not curricula nor their implementation) defined first!
  - Assumption: CS taught at least one session per week starting in fifth grade.

- Opportunity: rethink what and how to teach in secondary CS courses.