18

Principles of Programming Education

Michael E. Caspersen

Chapter outline

Chapter synopsis

The defining characteristics of the computer is its programmability, and programming is the essence of computing/informatics. Indeed, computing is much more than programming, but programming – the process of expressing one's ideas and understanding of the concepts and processes of a domain in a form that allows for execution on a computing device without human interpretation – is essential to computing.

Teaching and learning programming is not easy; in fact, it is considered one of the grand challenges of computing education. In this chapter, we describe the nature of the challenge, and we provide a dozen teaching principles to help overcome the challenge.

18.1 Introduction

Writing a chapter about the principles of teaching of programming is an intriguing task but for many reasons also challenging – an entire book could be written on the subject.

Michael E. Caspersen

Director of It-vest – netwworking universities, Denmark
Honorary professor, Department of Computer Science,
Aarhus University, Denmark

mca@acm.org

Chapter 18: Principles of Programming Education

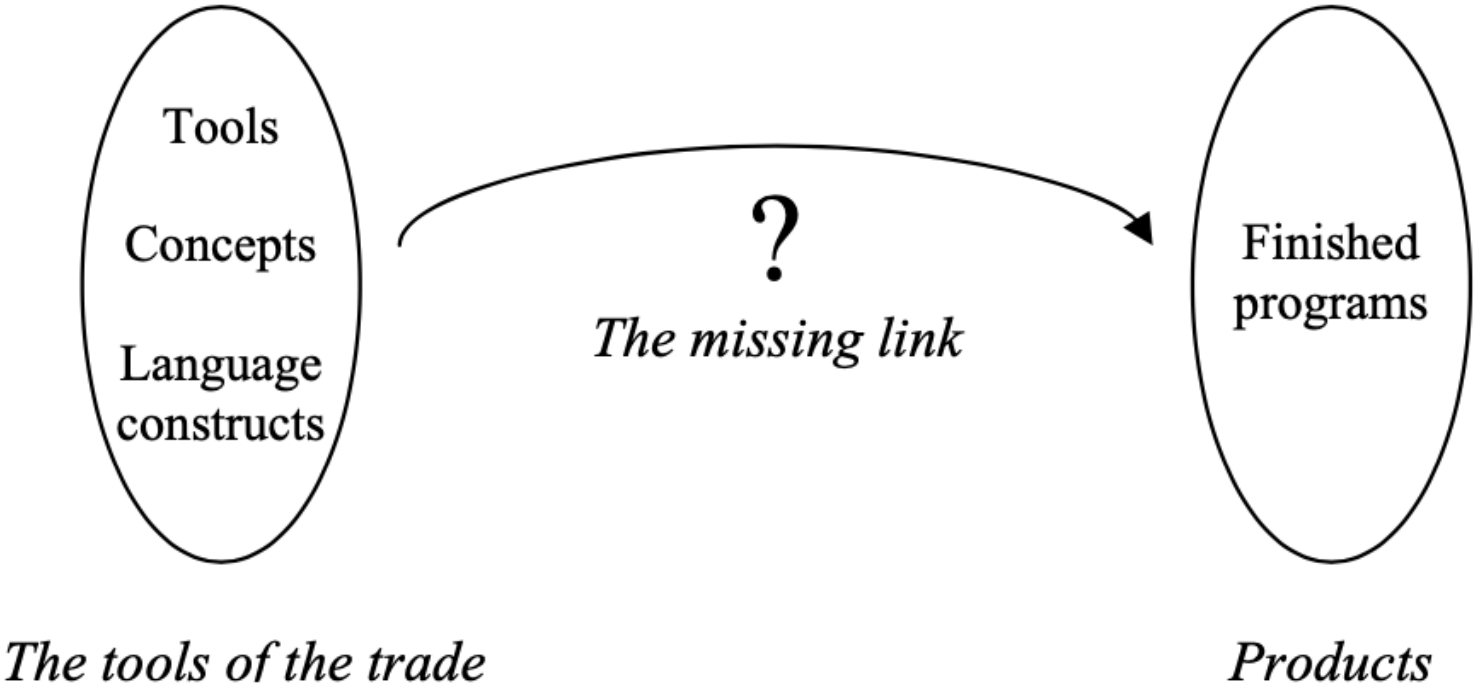


Figure 1-1: *The missing link between the tools of the trade and products*

From M.E. Caspersen (2007). *Educating Novices in the Skills of Programming*, PhD Dissertation, 2007, p. 6.

https://cs.au.dk/~mec/dissertation/Dissertation.pdf

WHAT SHOULD WE TEACH IN AN INTRODUCTORY PROGRAMMING COURSE?

David Gries
Computer Science Department
Cornell University
Ithaca, New York 14850

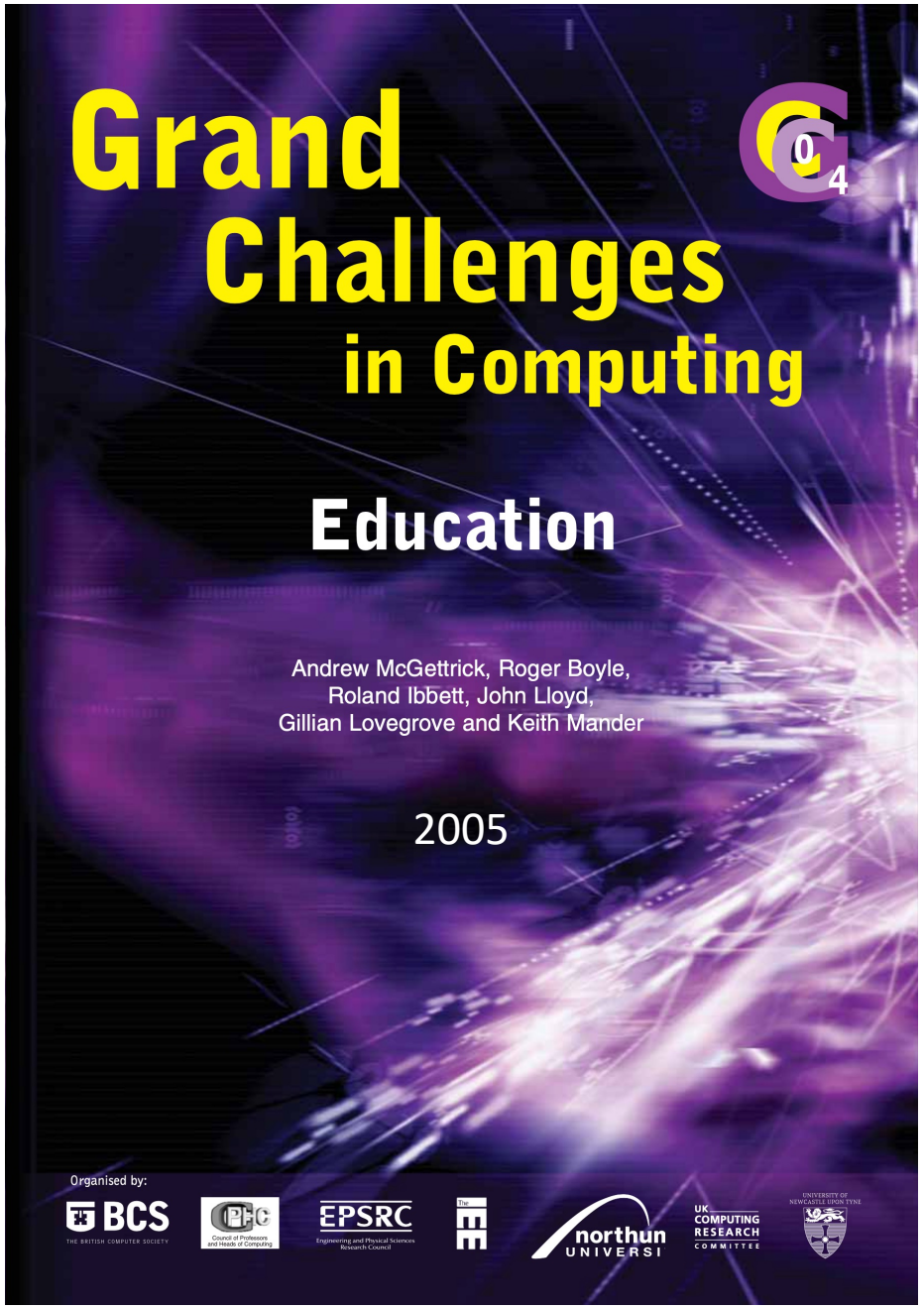
Two excerpts:

In essence, we want to teach how to solve any problem by finding an algorithmic solution to it. But what do we really teach? We describe the tools the student has at his disposal (the do-loop, goto, declarations, etc.), give a few examples, and then tell him to write programs. Almost no word on how to begin, how to find ideas, how to structure his thoughts, and how to arrive at a well-structured, well-written, readable program.

Let me make an analogy to make my point clear. Suppose you attend a course in cabinet making. The instructor briefly shows you a saw, a plane, a hammer, and a few other tools, letting you use each one for a few minutes. He next shows you a beautifully-finished cabinet. Finally, he tells you to design and build your own cabinet and bring him the finished product in a few weeks.

You would think he was crazy! You would want instructions on designing the cabinet, his ideas on what kind of wood to use, some individual attention when you don't know what to do next, his opinion on whether you have sanded enough, and so on.

https://dl.acm.org/doi/10.1145/800183.810447



However, concerns exist among the academic community internationally that when we set out to teach programming skills to students, we are less successful that we need to be and ought to be.

The particular concern is that, after more than forty years of teaching an essential aspect of our discipline to would-be professionals, we cannnot do so reliably. Indeed, there are perceptions that the situation has become worse with time.

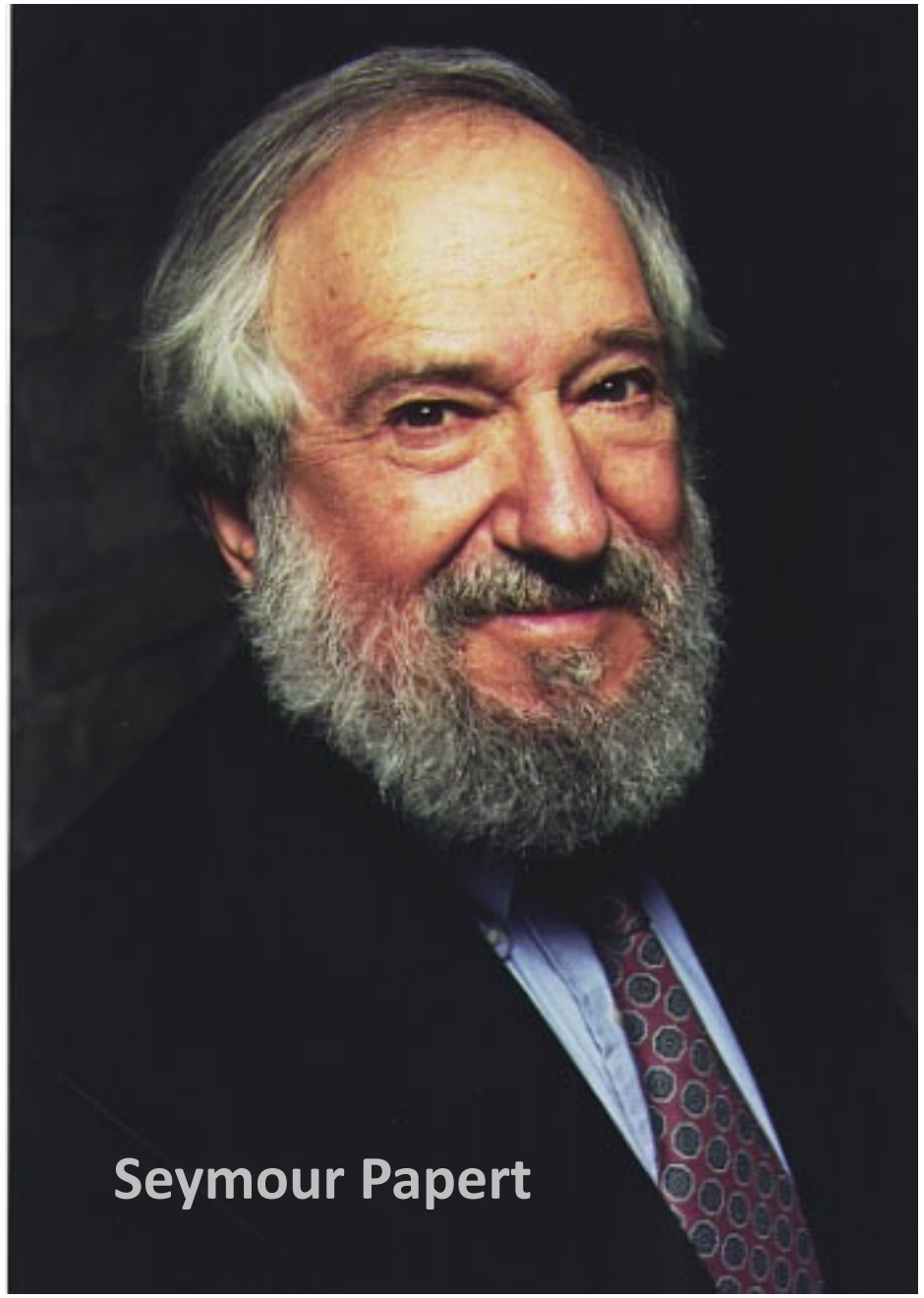
2.4.2 Challenge

Given the situation above, the computing challenge for this area is as follows:

> Understand the programming process and programmer practice to deliver effective transfer of knowledge and skills.

https://www.researchgate.net/publication/30975393_Grand_Challenges_in_Computing_Education--A_Summary

Chapter 18: Principles of Programming Education



Seymour Papert on computational thinking

Computational thinking is
the use of programming
– as an extension of our mind –
to experience and understand the world,
to manipulate the world,
and to create things
that matter to us.

Seymour Papert (1980). Mindstorms: Children, computers and powerful ideas. New York, Basic Books, Inc. p. 9.