Learner-Centered Design of Computing Education

Research on Computing for Everyone

Synthesis Lectures on Human-Centered Informatics

Editor
John M. Carroll, ***Penn State University***

Human-Centered Informatics (HCI) is the intersection of the cultural, the social, the cognitive, and the aesthetic with computing and information technology. It encompasses a huge range of issues, theories, technologies, designs, tools, environments and human experiences in knowledge, work, recreation and leisure activity, teaching and learning, and the potpourri of everyday life. The series publishes state-of-the-art syntheses, case studies, and tutorials in key areas. It shares the focus of leading international conferences in HCI.

Learner-Centered Design of Computing Education: Research on Computing for Everyone
Mark Guzdial
2015

The Envisionment and Discovery Collaboratory (EDC): Explorations in Human-Centered Informatics
Ernesto G. Arias, Hal Eden, and Gerhard Fischer
2015

Humanistic HCI
Jeffrey Bardzell and Shaowen Bardzell
2015

The Paradigm Shift to Multimodality in Contemporary Computer Interfaces
Sharon Oviatt and Philip R. Cohen
2015

Multitasking in the Digital Age
Gloria Mark
2015

The Design of Implicit Interactions
Wendy Ju
2015

Core-Task Design: A Practice-Theory Approach to Human Factors
Leena Norros, Paula Savioja, and Hanna Koskinen
2015

An Anthropology of Services: Toward a Practice Approach to Designing Services
Jeanette Blomberg and Chuck Darrah
2015

Proxemic Interactions: From Theory to Practice
Nicolai Marquardt and Saul Greenberg
2015

Contextual Design: Evolved
Karen Holtzblatt and Hugh Beyer
2014

Constructing Knowledge Art: An Experiential Perspective on Crafting Participatory Representations
Al Selvin and Simon Buckingham Shum
2014

Spaces of Interaction, Places for Experience
David Benyon
2014

Mobile Interactions in Context: A Designerly Way Toward Digital Ecology
Jesper Kjeldskov
2014

Working Together Apart: Collaboration over the Internet
Judith S. Olson and Gary M. Olson
2013

Surface Computing and Collaborative Analysis Work
Judith Brown, Jeff Wilson, Stevenson Gossage, Chris Hack, and Robert Biddle
2013

How We Cope with Digital Technology
Phil Turner
2013

Translating Euclid: Designing a Human-Centered Mathematics
Gerry Stahl
2013

Adaptive Interaction: A Utility Maximization Approach to Understanding Human Interaction with Technology
Stephen J. Payne and Andrew Howes
2013

Making Claims: Knowledge Design, Capture, and Sharing in HCI
D. Scott McCrickard
2012

HCI Theory: Classical, Modern, and Contemporary
Yvonne Rogers
2012

Activity Theory in HCI: Fundamentals and Reflections
Victor Kaptelinin and Bonnie Nardi
2012

Conceptual Models: Core to Good Design
Jeff Johnson and Austin Henderson
2011

Geographical Design: Spatial Cognition and Geographical Information Science
Stephen C. Hirtle
2011

User-Centered Agile Methods
Hugh Beyer
2010

Experience-Centered Design: Designers, Users, and Communities in Dialogue
Peter Wright and John McCarthy
2010

Experience Design: Technology for All the Right Reasons
Marc Hassenzahl
2010

Designing and Evaluating Usable Technology in Industrial Research: Three Case Studies
Clare-Marie Karat and John Karat
2010

Interacting with Information
Ann Blandford and Simon Attfield
2010

Designing for User Engagement: Aesthetic and Attractive User Interfaces
Alistair Sutcliffe
2009

Context-Aware Mobile Computing: Affordances of Space, Social Awareness, and Social Influence
Geri Gay
2009

Studies of Work and the Workplace in HCI: Concepts and Techniques
Graham Button and Wes Sharrock
2009

Semiotic Engineering Methods for Scientific Research in HCI
Clarisse Sieckenius de Souza and Carla Faria Leitão
2009

Common Ground in Electronically Mediated Conversation
Andrew Monk
2008

© Springer Nature Switzerland AG 2022
Reprint of original edition © Morgan & Claypool 2016

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopy, recording, or any other except for brief quotations in printed reviews, without the prior permission of the publisher.

Learner-Centered Design of Computing Education: Research on Computing for Everyone

Mark Guzdial

ISBN: 978-3-031-01088-0 paperback
ISBN: 978-3-031-02216-6 ebook

DOI 10.1007/978-3-031-02216-6

A Publication in the Springer series

SYNTHESIS LECTURES ON HUMAN-CENTERED INFORMATICS

Lecture #33
Series Editor: John M. Carroll, *Penn State University*
Series ISSN
Print 1946-7680 Electronic 1946-7699

Learner-Centered Design of Computing Education

Research on Computing for Everyone

Mark Guzdial
School of Interactive Computing, College of Computing
Georgia Institute of Technology

SYNTHESIS LECTURES ON HUMAN-CENTERED INFORMATICS #33

ABSTRACT

Computing education is in enormous demand. Many students (both children and adult) are realizing that they will need programming in the future. This book presents the argument that they are not all going to use programming in the same way and for the same purposes. What do we mean when we talk about teaching *everyone* to program? When we target a broad audience, should we have the same goals as computer science education for professional software developers? *How* do we design computing education that works for everyone? This book proposes use of a *learner-centered design* approach to create computing education for a broad audience. It considers several reasons for teaching computing to everyone and how the different reasons lead to different choices about learning goals and teaching methods. The book reviews the history of the idea that programming isn't just for the professional software developer. It uses research studies on teaching computing in liberal arts programs, to graphic designers, to high school teachers, in order to explore the idea that computer science for everyone requires us to re-think how we teach and what we teach. The conclusion describes how we might create computing education for everyone.

KEYWORDS

computer science education, computing education, learner-centered design

Dedicated to Elliot, Janet, Jim,
Peter, Rich, John, Alan, and all my mentors.

Contents

Preface . . . . **xv**

Acknowledgments . . . . **xvii**

1 What Does Computing for Everyone Mean? **1**

1.1 Defining Terms: Computing Education, Computer Science, and Programming 1

1.2 Why Should Everyone Learn Computing? 3

1.2.1 Jobs 3

1.2.2 Learn About Their World 4

1.2.3 Computational Thinking 4

1.2.4 Computational Literacy 5

1.2.5 Productivity 6

1.2.6 Broadening Participation 8

1.3 How Do We Teach Computer Science Now? 10

1.3.1 Perspectives of Computer Science Teachers 10

1.3.2 Teaching for Software Development Expertise 12

1.3.3 Expertise and Authenticity are Socially Determined 13

1.3.4 The Inverse Lake Wobegon Effect 14

1.4 How Should We Teach Computing for Everyone? 15

1.4.1 Learner-Centered Design 15

1.4.2 Example: Computational Media at Georgia Tech 16

1.5 Overview of the Book 17

2 The Challenges of Learning Programming **21**

2.1 The Rainfall Problem: Programming is Harder Than We Thought 21

2.2 Multi-Institutional, Multi-National Studies 23

2.3 Measuring Computer Science Knowledge 24

2.4 How Learning Works 27

2.4.1 Assimilation and Accomodation 27

2.4.2 Transfer of Knowledge 28

2.5 Why Do Students Learn Less Computer Science Than We Expect? 29

2.5.1 What Makes Learning Computer Science Different . . . 30
2.5.2 Developing a Notional Machine . . . 30
2.6 Increasing the Value of Learning Computing . . . 33
2.6.1 Situated Learning . . . 33
2.6.2 Expectancy-Value Theory . . . 34
2.7 What Makes Learning Programming Hard . . . 36

3 Computational Thinking and Using Programming to Learn . . . 37
3.1 Computers and the World of the Future . . . 37
3.2 Computational Thinking . . . 39
3.3 Programming as a Tool for Learning . . . 41
3.3.1 Smalltalk and Boxer . . . 41
3.3.2 Students Building Software to Learn . . . 43
3.4 Programming and Problem-Solving Skills . . . 46
3.5 Learning Programming to Learn Mathematics and Science . . . 48
3.6 Learning About Computer Science When Developing Computational Literacy . . . 49
3.7 Summary: Synergy Between Computing and Other Domains . . . 50

4 Media Computation and Contextualized Computing Education . . . 53
4.1 Teaching One Course For All . . . 53
4.1.1 What's Wrong with One Course? . . . 55
4.2 The Design and Development of Media Computation . . . 56
4.2.1 Scaffolding in Media Computation . . . 58
4.2.2 Offering the First Media Computation Courses . . . 60
4.3 Impact of Media Computation . . . 62
4.3.1 Impact Within Georgia Tech . . . 62
4.3.2 Impact Outside Georgia Tech . . . 64
4.3.3 Why Does Media Computation Work, and Where Doesn't it? . . . 64
4.4 A Second Course in Media Computation . . . 65
4.5 Design of a Computing Course for Non-Computer Science Majors . . . 67

5 Adults as Computing Learners . . . 69
5.1 Graphic and Web Designers as End-User Programmers . . . 70
5.1.1 Helping Graphic Designers . . . 70
5.1.2 Starting From the Learning Challenges . . . 72
5.2 The Needs of Computing Teachers as Computing Learners . . . 73

5.2.1 Adults Learning Computer Science Online 74
5.2.2 What High School Computer Science Teachers Need to Succeed 75
5.2.3 Developing the Identity of a Computer Science Teacher 78
5.2.4 Developing Online Teacher Education 79
5.3 Designing Computing Education for Adults 80

6 Learner-Centered Computing Education for Computer Science Majors 83
6.1 Multiple Introductory Courses are Useful 83
6.2 The "Geek Gene" 84
6.2.1 Teaching to Address Differences 85
6.2.2 Diverse and Effective Teaching Methods 86
6.3 Creating Face-Saving Paths Into Computer Science 88
6.4 More Than One Kind of Computer Science 89
6.4.1 Threads 90
6.4.2 How Computer Science Students Make Choices 91
6.5 Context Helps Computer Science Majors, Too 92
6.6 The Need for Greater Exploration in Computing Education 93

7 Steps Toward Computing for Everyone 95
7.1 Revisiting Why Everyone Should Learn Computing 95
7.2 Teaching Computing in Elementary School 97
7.2.1 Why Not Teach Computing in Elementary School 97
7.2.2 Why Teach Computing in Elementary School 98
7.2.3 Research Questions for Teaching Computing in Elementary School . . 100
7.3 Teaching Computing in Secondary School 101
7.3.1 Research Questions for Teaching Computing in Secondary School . . . 103
7.4 Teaching Computing to Undergraduates 104
7.4.1 Improve Teaching About Computing 105
7.4.2 Improve the Tools of Computing 106
7.4.3 Create More Researchers in Computing Education 108
7.5 Conclusion: Learner-Centered Design of Computing Education 108

Bibliography 111

Author's Biography 141

Index 143

Preface

Some of the earliest work in computing education is about the value of computing and programming *for everyone*, or at least not just for professional programmers. The pioneers of this perspective included Alan Perlis, Seymour Papert, Alan Kay, Adele Goldberg, Cynthia Solomon, and Andrea diSessa. When those early computer scientists started talking about the value of computing for learning, there was no enormous demand for a programming labor force. Instead, they argued that computing was an important medium for learning. Today, computing has become so important for our modern society, and the need for programming labor is so great that the power of computing as a medium for expression and thought may no longer be considered.

My personal introduction to this perspective was when I first read *Personal Dynamic Media* [169] over 30 years ago, and the vision continues to inspire me. Now, I realize that there is a wide range of desired learning outcomes from computing education.

Unfortunately, most of a computer science education today is about getting better at producing software developers. The goal is greater productivity of higher-quality software developers. The annual SIGCSE Technical Symposium is mostly a meeting of over 1000 undergraduate computer science teachers, where their shared goal is to provide great teaching to contribute workers to the software industry. I share that goal, but I believe that there is a broader picture of providing access to the advantages of computing as a tool to think with to everyone who wants it.

This book is a review of the research literature on teaching computing to everyone. My goal is to be most useful to new researchers who want to understand the narrative of teaching programming to students, from the 1960's to today. Teachers may find this book useful to see different perspectives on how to design computing education for different purposes. While this book addresses the use of different teaching methods for different audiences, my goal is not to offer teaching methods. The best book I know for how to become a computer science teacher is the 2011 *Guide to Teaching Computer Science* [234]. This book focuses on understanding learners and their issues and on supporting learning by students who *are* and also others who *aren't* aiming to be professional software developers. Call this latter category "computing education *for the rest of us*."

Mark Guzdial
November 2015

Acknowledgments

The idea for this book came out of discussions with Josh Tenenberg and Jane Margolis. Josh and I talked about helping new researchers to connect computing education to other research in education and learning sciences. I hope that this book helps with making those connections. When Jane was working on *Stuck in the Shallow End* [203], she told me that she wished that there was already a book that talked about the kinds of computer science that could be taught more broadly, that argued for the importance of everyone learning about computing. Since then, Yasmin Kafai and Quinn Burke wrote *Connected Code: Why Children Need to Learn Programming* [160] which makes that argument for children better than I could. This book aims to continue that story to explore why we might want *everyone* to learn programming, from children through undergraduates and to adults, and how we might achieve that goal.

My thanks to the many colleagues who worked on the research that I describe in this book. I offer my apologies in advance for where I got it wrong.

My great thanks to those who read early drafts of this book. Peter Denning was the first of my readers, and he gave me valuable insights for which I'm grateful. Greg Wilson and Nick Falkner did an amazing job of wading through the whole document and helping me see where my words did not mean what I meant them to mean. Mike Hewner, Lijun Ni, and Lana Yarosh kindly reviewed the book to make sure that what I said about them were true (to the best of their recollection). My students at Georgia Tech were also helpful in spotting errors when we used a draft of this book in a seminar. You can be sure that mistakes and erroneous conclusions in this volume are my fault for not listening better to my readers.

I am grateful also to those reviewers who read the draft that I submitted to Diane Cerra at Morgan & Claypool. Quintin Cutts, Sally Fincher, Kathi Fisler, Alan Kay, Mike McCracken, Scott McCrickard, Chris Quintana, Mary Beth Rosson, and Elliot Soloway put in an enormous amount of time reading the whole thing, pointing out questions, and helping me understand how much a gap there was between the submitted draft and a useful book. The book is much closer to useful due to their attention and efforts, for which I thank them all.

The work at Georgia Tech was supported by several *National Science Foundation* grants, including NSF grants CNS-0512213, CNS-0618674, CNS-0634629, and CNS-0940394. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

I have dedicated the book to my mentors who have helped a learning scientist with an education degree succeed in a computing world, especially Elliot Soloway, Janet Kolodner, Jim Foley, Peter Freeman, Rich LeBlanc, John Stasko, and Alan Kay. I have been fortunate in having many mentors and advisors. When I graduated with my Ph.D. and received my offer to join Georgia

Tech, Elliot and I met and talked through strategy. Was there any way that a computing education guy could get tenured in a College of Computing? Thanks to my mentors, I have been able to succeed in a research-focused computer science department in the U.S. while doing computing education research. I needed all of their help to still be here.

Finally, my thanks to my wife, Barbara Ericson. She supported me during my Logo and Emile days, and has been my research partner for over a dozen years now. I am grateful for her support in all the crazy projects I take on, like this book.

Mark Guzdial
November 2015