Undergraduate Topics in Computer Science

Undergraduate Topics in Computer Science (UTiCS) delivers high-quality instructional content for undergraduates studying in all areas of computing and information science. From core foundational and theoretical material to final-year topics and applications, UTiCS books take a fresh, concise, and modern approach and are ideal for self-study or for a one- or two-semester course. The texts are all authored by established experts in their fields, reviewed by an international advisory board, and contain numerous examples and problems. Many include fully worked solutions.

More information about this series at http://www.springer.com/series/7592

Natalia Silvis-Cividjian

Pervasive Computing

Engineering Smart Systems

Springer

Natalia Silvis-Cividjian
Department of Computer Science
Vrije Universiteit Amsterdam
Amsterdam
The Netherlands

ISSN 1863-7310 ISSN 2197-1781 (electronic)
Undergraduate Topics in Computer Science
ISBN 978-3-319-51654-7 ISBN 978-3-319-51655-4 (eBook)
DOI 10.1007/978-3-319-51655-4

Library of Congress Control Number: 2016963311

All figures made with MATLAB® are reprinted with permission of The MathWorks, Inc

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To my students

Foreword

Computers have been getting smaller for decades. Back in the 1960s, we had mainframes, which filled large rooms and cost tens of millions of dollars. In the 1970s, we had minicomputers, which were the size of refrigerators and cost only tens of thousands of dollars. In the 1980s, we had personal computers, small and cheap enough for a person to buy in a store and take home. In 2007, the iPhone was introduced and you could hold a full-blown computer in your hand.

Now computers are about to disappear altogether. The next phase in computing is pervasive computing, with invisible computers everywhere, doing things for you without your even being aware of them. They will be embedded in door locks, clothes, medical devices, cars, banknotes, credit cards, thermostats, roads, toys, kitchen appliances, milk cartons, guns, cameras, light bulbs, drinking cups, pets, smoke detectors, blankets, pill bottles, toothbrushes, scales, lawn sprinklers, and a vast number of other things. They will interact with people, each other, and the cloud wirelessly and seamlessly, without people even being aware of them.

The idea of pervasive computing, also called ubiquitous computing, was dreamed up by the late Mark Weiser, a visionary and friend of mine, in 1991. When he wrote his classic paper about this idea, it was pure science fiction, not unlike a 1950s science fiction author writing about a device the size of a book that could summon up all the world's information. The latter remained science fiction, until the iPad came along. And Weiser's idea is not science fiction any more, either. It is happening right now. If you want to learn about the underlying technology that is making it happen, this is the book for you.

The book covers analog and digital signals, control systems, sensors, image processing, audio signal processing, and classifying patterns. The final chapter puts all the pieces together and discusses systems engineering, especially software engineering. The appendix contains some experiments you can try with readily available off-the-shelf parts. These are ideal for a lab course for computer science or engineering students.

Although the book is intended for computer science and engineering undergraduates, it should be accessible to anyone with a science background, a command of high school mathematics, and some programming experience.

Natalia's writing style is wonderfully refreshing. So many technical books are accurate and full of information, but dry and boring to read. This one is different. Throughout the book the author talks directly to the reader, often in amusing ways. The figures are clear and often fun to look at.

If you want to learn the technical details of how pervasive computing, the Internet of Things, or whatever you want to call it actually works, this book is a great place to start.

Amsterdam, The Netherlands Andrew S. Tanenbaum

Preface

If you understand how the universe operates,
you control it, in a way.
Stephen Hawking, My Brief History

Pervasive computing describes ICT (information and communication technology) systems that seamlessly enable information and services to be made available everywhere. Other terms, used with approximately the same meaning as pervasive computing, are *Ubiquitous Computing, Internet of Things* (*IoT*), *calm computing, ambient intelligence, physical computing, smart spaces,* and *smart environments.*

You can write about pervasive computing in a million different ways. This book tells the story from a *systems engineering* perspective and focuses on smart systems that use pattern recognition to discover their context. My wish is to make you, the reader, feel confident that not only you *understand* how these systems operate, but you *can* also *build* such systems. While reading the book, you will realize, on the one hand, how simple the underlying principles are, and on the other hand, how difficult it is to implement them in practice. Learning how to ride a bicycle is also difficult. And yet, Dutch children learn this skill very early, usually before they reach four. What you have to do as a parent is to run along—over and over—and keep them in balance, until they suddenly get the trick and boost forwards, leaving you behind, out of breath, yet relieved. This book is a guide for exactly this period—of desperately running behind the bicycle. After that, I believe that imagination and talent will drive you toward building the most incredible systems.

This book is crafted around the lecture notes developed for the course *Pervasive Computing*, taught to computer science (CS) freshmen at the Vrije University in Amsterdam. Upon completion of this course, successful students will be able to:

- Design a realistic smart system with the potential to benefit human lives. The system acquires and processes data from video, audio, acceleration, or EEG sensors and uses pattern recognition to take decisions that affect the environment accordingly.

- Build a simplified version of the real system and program a software agent to control it.
- Work together in a team, collaboratively identifying not only the technical but also the safety or ethical issues with their designs, and then sharing their challenges and discoveries through reports, presentations, and in-class demonstrations.

Although mainly targeted for computer science undergraduate students, I believe this book will be interesting and readable for anyone wondering what happens behind the scenes of these fascinating systems. After reading it, the ones who dream of building their own system should feel one step closer to their goal.

Acknowledgements This book is a result of a collective effort. Maarten van Steen was the first who came up with the idea of designing a pervasive computing course. Countless brainstorm sessions on how to teach it followed, with valuable input from Melanie Rieback, Guillaume Pierre, Michel Klein, and Peter van Lith. The course followed a meandering evolution, toward the version described in this book. I am fortunate to work in the computer science department that owes so much fame to Prof. Andrew Tanenbaum and I am most grateful and honored that he kindly consented to contribute the foreword. We teachers are, in fact, story tellers. The engineering flavor in this book is inspired by my teacher, Mircea Ţăţulescu, who first told me the story of sensors, data acquisition and signal processing, and irreversibly influenced the way I see the world. Hans van Vliet pointed me out the importance of software testing, and Jaap Gordijn and Patricia Lago helped me to embed software modeling elements in this course. Many thanks go to Wan Fokkink and Spyros Voulgaris, for reviewing the original lecture notes, and to Anton Eliens and Herbert Bos for their encouragements during writing this book. Teaching this course could have been a disaster without my teaching assistants, who supported me over the years, for better or worse, with their awesome ideas, hardworking and enthusiasm. I am indebted to Alyssa Milburn, for her critical, perfectionistic eye set on my writing. Caroline, thank you for your interest in my book and for your time management lessons. I am grateful to my Springer editors, Beverley Ford and James Robinson, for their initial trust and efficient assistance during the manuscript preparation.

I collected many beautiful illustrations in this book. Grateful acknowledgment is made to all those who kindly granted me permission to use them: Roberto Brunelli, Diana Cook, Dariu Gavrila, Daniel Goehring, Horst-Michael Gross, Erico Guizzi, Alex Faaborg, Bob Fisher, Kees Hagen, Johan Hoorn, Mehdi Jayazeri, Phillip Laplante, Alisdair McAndrew, Jim McClellan, Steve McConnell, Thomas Moeslund, Niels Noordzij, Martin Pearson, Mauro Pezze, Dirk van der Pijl, Jozef Pucik, Hanna Reimers, Stuart Russell, Jan Schnupp, Chris Solomon, Ian Sommerville, Jakob Suckale, Sergios Theodoridis, Johannes Trabert, and Michal Young. Many thanks go to Naomi Fernandes and Joachim Levelt from Mathworks, Inc., for their speed in processing my permissions procedure.

Being able to teach and shape your own material is a blessing, and nobody understands this better than my parents. Thank you, *dragii mei*, for your

unconditional love and for teaching me to respect and care about my students. Undergoing this writing project, especially in its final part, put some pressure on my home front, as well. Fortunately, we survived through all this turbulence, due to Peter, my guide and companion in the journey of life, who seamlessly took over my "ubiquitous" duties. *Oma en opa Silvis, dank voor het faciliteren van al die heerlijke, broodnodige rustmomenten*. Too often, my kids had to miss me without getting too many explanations. They only knew that mama writes a book. "About an adventure?" they asked. "No? Oh, then it must be very boring…" Fortunately, their cynical optimism and young spirit prevented the worst. I hope that one day they will find more answers and be able to forgive. *Dank, mijn liefste*.

Amsterdam, The Netherlands
November 2016

Natalia Silvis-Cividjian

Contents

Abbreviations

AC	Alternative current
ADC	Analog-to-digital converter
AI	Artifical intelligence
ANN	Artificial neural network
ASR	Automatic speech recognizer
AUC	Area under the curve
B	A formal specification language
B&W	Black and white
BGC	Blood glucose concentration
BLOB	Binary large object
BVA	Boundary value analysis
ConOps	Concept of operation
DAC	Digital-to-analog converter
DC	Direct current
DSP	Digital signal processing
EBID	Electron beam-induced deposition
ECG	Electrocardiogram
EEG	Electroencephalogram
EP	Equivalence partitioning
F1, F2	The frequencies of first two formants in the spectrum
FDA	Food and Drug Administration
FFT	Fast Fourier transform
FIR	Finite impulse response
FMEA	Failure mode and effects analysis
FPR	False-positive rate
FTA	Fault tree analysis
GPS	Global Positioning System
HMM	Hidden Markov model
IC	Integrated circuit
IDE	Integrated development environment

IFFT	Inverse fast Fourier transform
i-HCI	Implicit human–computer interaction
IIR	Infinite impulse response
IoT	Internet of Things
LED	Light-emitting diode
LIDAR	Light detection and ranging
MBT	Model-based testing
MEMS	Microelectromechanical systems
MFCC	Mel frequency cepstrum coefficients
MLP	Multilayer perceptron
MOSFET	Metal–oxide–semiconductor field-effect transistor
OCR	Optical character recognizer
PD	Proportional derivative
PID	Proportional integral derivative
PIR	Passive infrared sensor
PLI	Power line interference
RFID	Radio-frequency identification
RGB	Red green blue
ROC	Receiver operating characteristics
SRS	System (or software) requirements specification
STAMP	Systems-Theoretic Accident Model and Processes
STFT	Short-term Fourier transform
STM	Scanning tunneling microscope
STREL	Structuring element
TDD	Test-driven development
TN	True negative
TP	True positive
TPR	True-positive rate
UML	Unified modeling language
V&V	Verification and validation
VDL	Vienna definition language, a formal specification language
Z	A formal specification language