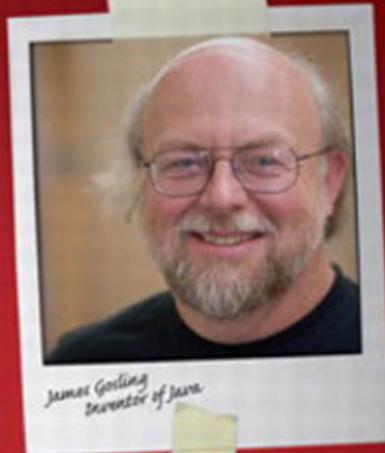
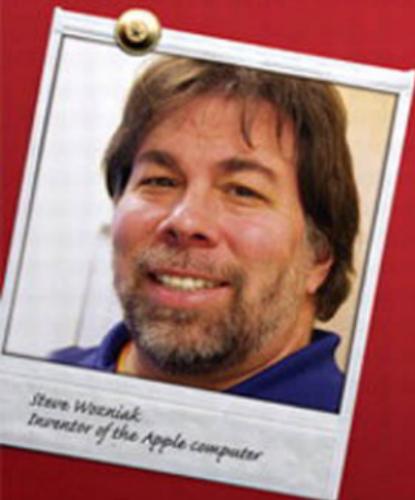


PRENTICE
HALL



Making it **Big** in Software

Get the Job. Work the Org. Become Great.



Sam Lightstone

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

Photo of Steve Wozniak by Al Luckow.

Photo of Diane Greene by Sunny Scott.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales
international@pearson.com

Visit us on the Web: informit.com/ph

Library of Congress Cataloging-in-Publication Data

Lightstone, Sam.

Making it big in software : get the job—work the org.—
become great / Sam Lightstone.

p. cm.

Includes index.

ISBN 978-0-13-705967-6

1. Computer programming—Vocational guidance. 2.
Computer software industry—Vocational guidance. 3.
Computer software developers—Interviews. I. Title.

QA76.25.L54 2010

005.1023—dc22

2010000235

Copyright ©2010 Sam Lightstone. Copyright claimed in entire work exclusive of the reproduction of US Patent No. 6,161,223

All rights reserved. Printed in the United States of America.

This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, write to:

Pearson Education, Inc
Rights and Contracts Department
501 Boylston Street, Suite 900
Boston, MA 02116
Fax (617) 671 3447

ISBN-13: 978-0-137-05967-6

ISBN-10: 0-137-05967-1

Text printed in the United States on recycled paper at
R.R. Donnelley & Sons, Crawfordsville, Indiana

First printing March 2010

Associate Publisher

Greg Wiegand

Senior Acquisitions Editor

Katherine Bull

Development Editor

Kendell Lumsden

Marketing Manager

Judi Morrison

Publicist

Heather Fox

Technical Reviewers

Joseph Hellerstein

Rasekh Rifaat

Danny Sadinoff

Managing Editor

Kristy Hart

Senior Project Editor

Lori Lyons

Copy Editor

Hansing Editorial Services

Indexer

Publishing Words

Proofreader

Language Logistics, LLC

Publishing Coordinator

Cindy Teeters

Cover Designer

Alan Clements

Compositor

Nonie Ratcliff

Manufacturing Buyer

Dan Uhrig

Preface

“Thus the sage Chaninah would say:

Much have I learned from my teachers and even more from my friends but from my students I have learned the most of all”

—Ethics of the Fathers, circa 200 CE

You went to university to study a profession, but they were hell-bent on giving you an education instead. Unfortunately, there’s a gap between the formal education we receive in school and the skills we need to build truly successful careers. Not only are many of the basic skills for professional growth not taught in school, but to a large degree they aren’t taught anywhere. The dynamic and somewhat bohemian quality of the software industry introduces unique career challenges. Our industry includes many of the trappings of corporate American culture, but many odd divergences. It’s a domain where teenage hackers compete head-to-head with MIT Ph.D. graduates and where crinkled T-shirts and unlaced running shoes coexist with stock options and executive titles. *Making it Big in Software* is my attempt to share strategies for traversing these unique dynamics and maximizing your professional potential. This book also includes interviews with some of the most influential software innovators and leaders of the past 30 years—people who literally changed the world.

In 1991, I was a fourth-year electrical engineering student at Queen’s University, trying to finish up my degree and get a job. Every Friday afternoon the electrical engineering school invited a guest speaker to inspire and enlighten our young, impressionable minds. These talks covered a wide range of technological topics; we discussed everything from high voltage transmission lines to CMOS VLSI circuit design. I confess that most of these sessions were less than inspirational for me. One day, a guest speaker arrived with a radically different message. He spoke to us about “life in the real world” and what we could expect after we graduated. I was riveted. This is the only fourth-year seminar from which I still have the notes. Most of my classmates, like me, were in the dark about what our lives might be like after graduation. Even those students who had managed to find summer positions in

engineering were limited in their experiences by the normal constraints of student positions.

I decided then that if I ever could, I would return the professional courtesy and volunteer to speak to university students. In the late 1990s and early 2000s, I began a series of career talks at leading universities. The lecture notes from my talks, often delivered to capacity-filled auditoriums, became the basis for this book. Some of the ideas are my own, but many are culled from the leading business thinkers and software development evangelists of the past three decades. Hopefully, this book succeeds in providing you what school and daily work life generally cannot: the tools with which to make it BIG.

Audience

This is a book for software professionals of all ages and levels, whether they're just trying to break into the field or have decades of experience. Writing a book with that range isn't easy; every age group and community has a different set of concerns that affect what's interesting to them. After nearly two decades of recruiting, managing, and mentoring software professionals, there are some common themes I've found that cross most boundaries of age and experience. I've tried to make those themes central to the book. I hope this book will also be of interest to students and teachers of computer science, providing a colorful look into the less mathematical aspects of the profession.

Organization of This Book

The book is divided into three major sections: fundamentals, leadership, and greatness. "Part I, Fundamentals," discusses the major building blocks of a great career in software, including the fundamentals of good software products, major skills and leading programming languages, how to land a job, and how to operate effectively within a software team (including some pitfalls to avoid). "Part II, Leadership," takes the reader into a series of topics around driving and leading change and operating in a much more harried professional environment. Leaders, almost by definition, need to multitask and drive parallel agendas forward while deflecting myriad forces of negativity. This section covers important topics related to "working an organization," how to build and pitch successful project proposals, career advancement, time

management, avoiding software project overruns, work-life balance, and higher-level management and leadership insights. “Part III, Greatness,” discusses topics around innovation—how to reach the pinnacle of the profession (becoming a software visionary or guru), how to start your own business, a review of compensation rates, and a retrospective on things I wish I had known earlier in my career.

Between the chapters you’ll find a number of interviews I did with leading personalities. These interviews deliberately include a mix of business executives, researchers, and industry leaders. I feel it’s important for readers to get perspectives broader than my own. To enhance the structure of the book, the interviews are all marked with a gray bar along the leaf edge of their pages, including the interviewee’s name. I’ve made a moderate effort to place the interviews near related chapters (for example, two startup sensations, Marc Benioff CEO and founder of Salesforce.com and Diane Greene past CEO and co-founder of VMware, frame the chapter on software startups). Some interviews didn’t have a clear correlation to a specific topic, so there’s an admittedly arbitrary aspect in the placement of a few. It’s absolutely not the case that interviews placed in the Fundamentals section are any less profound than those in the Greatness section. I think it’s fair to say that everyone I interviewed has reached the heights of the profession. After all, that’s what made them such interesting people to speak with.

Feedback

If you have feedback on the book or find any errors, please contact me. You can send feedback to feedback@makingitbigcareers.com. Although I can’t guarantee a reply to every email, I’ll make sure that every email is read.

Sam Lightstone

To learn more about *Making it Big in Software*,
including upcoming events and additional publications,
please go to:
www.MakingItBigCareers.com

CHAPTER 3

School Versus Job

“Every person takes the limits of their own field of vision for the limits of the world.”

—Arthur Schopenhauer (1788–1860)

School is very different from professional work. Some people stumble in their early professional careers by failing to transition from the school environment they’ve been immersed in for nearly twenty years into the brave new world of software professionalism. Students operate in a world of highly constrained, well-defined workloads (though it certainly doesn’t feel that way when you’re a student). As a student your scope is limited to a set of courses that are intended to address your professional needs upon graduation. In short, school is about learning. Professional life is more about getting things done, being both productive and innovative in a way that jells with the people you work with. It’s not uncommon for junior staff to treat the workplace as though it were simply the next phase of school. Few things could be more disastrous.

I also believe that a great weakness of many middle managers is failing to understand the differences between the academic and corporate skills requirements, leading them to hire the best and brightest students rather than the best and brightest professionals. A good student is usually smart, hard-working, and conscientious. But success in scholastics doesn’t guarantee an ability to innovate, learn independently, work in a team, or show leadership—which are all critical skills for professionals. Hiring the wrong skill set means hiring less effective people, and when that phenomenon is multiplied over many people in an organization, the net result is a marked reduction in the effectiveness and talent of the entire organization. Put another way, hiring the wrong people for the job is a bad way to run a company. On the flip side,

when new graduates position themselves for the workplace, assimilating the culture and needs of a professional life, and managers learn to recruit the best employees for their needs rather than the best students, the mixture forms a potent brew for the rapid success of individuals and organizations alike.

Limited Field of Vision

Whether you're in school studying the art and science of computer programming or working as a software professional, you exist and function in a largely constrained and somewhat artificial environment. Life is a kind of fishbowl. Fish in a bowl can swim and explore, up to a point, but their view of the world is remarkably limited. The fish sees almost nothing beyond the glass of the fishbowl, in part because its eyes can't focus that far and in part because of the diffraction of light as it crosses the boundary from water to glass to air. If the fish could see beyond the fishbowl, at most, its view of the world would be limited to the room the bowl resides in: a few chairs, a sofa, a bookshelf. The real world outside of the fishbowl is far larger and more profound than the tiny colored glass stones and rocks that have been artificially placed in the fish's manufactured domain. But, for better or worse, the fish is blissfully unaware. Software professionals are fish, too.

School Is a Fishbowl

School is a highly artificial environment where the workload is relatively well controlled, where all participants have similar work challenges, and where people are encouraged to do individual work. If you are a student, you probably think I've completely forgotten what school is all about because what I've described is nothing like the world you're living in. Very true, but it's a relative comment. Even though some professors demand a far heavier workload than others, the reality is that these variations are small compared to the variation that would occur without a fixed curriculum, such as in the environment outside of school. You'll protest: "But if there were no set curriculum, that wouldn't be fair!" and you'll be 100% correct. That's one reason schools must have a set curriculum—because school needs to be fair. The process of making school fair leads to a wide array of artificial constraints and behaviors that, although fair, necessarily create a framework of a highly constrained fishbowl.

Second, although schools encourage students to do their own work, on penalty of expulsion or severe reprimand, professional work is saturated with the ubiquitous mantra of "teamwork." In school, your success depends on

individual effort, whereas professional life depends frequently on your ability to work in teams.

Finally, very little that students experience in school is directly influenced by market pressures. There is some indirect influence, as the market drives new technologies to the fore, which, over time, influences curricula. Professors get involved in new technologies and introduce these themes into the courses they teach. Over time, the system of professors, industrial grants, the drive for publications, and review committees will ensure that the educational process stays reasonably connected to current industrial trends. All the while, the student body is happily oblivious.

Welcome to the fishbowl. While you're in it, you won't be able to see too far past the glass. That's fine, as long as you realize that there is a world outside the fishbowl that's a rather different place, then you're well positioned to try to learn more about it.

Industry Is a Fishbowl

How in the world could industry be considered a fishbowl? Isn't industrial work the definition of the real world? It's time for a reality check! The software industry is sensationally complex, and it requires a broad range of skills and disciplines to construct a successful business (the larger the enterprise, the more this is true). People become specialized and focus on a narrow scope of the industrial machine. Every employee has a domain. High up in the organizational hierarchy, people working on the big picture can't possibly know all the little details. At the other end, the detail workers, down in the trenches, have a very hard time understanding the broader scope of every project that is related to them.

Second, if you're like most software developers, your compensation is only loosely tied to the market success of the projects you work on, unless you are an owner in the company (such as owning stock or stock options) or have a compensation plan that is directly tied to quotas or business performance. Similarly, your compensation is only loosely connected to years of service or the technical depth of your job.

Your compensation at work will never directly increase with your productivity or the success of your product. Employees who work five times more will not get paid five times more than others. If your company revenue soars and revenues increase by tenfold, don't expect employee salaries to increase by a factor of ten. If this sounds discouraging, it shouldn't be—remember, the sword cuts both ways. Just as you are unlikely to see your salary grow 300 times in size when your product begins to sell and earn 300 times more revenue, your salary will not be cut to 1/300th if the inverse befalls you. And

although the most productive and valuable people might not get compensated proportional to their contributions in a literal measurement, they do get paid substantially more. Over time, they will accrue much more accomplishment, fame, money, and freedom of movement for their efforts.

Professional software positions almost always limit the software developer from the sales and marketing world, and in larger companies, software developers can be heavily isolated from customers, business strategy, and product planning. In short, it's a controlled and not truly reflective environment. So it's a fishbowl—what you see is not a reflection of reality.

By understanding and being sensitive to the artificial limitations of your environment, you can begin an active effort to extend your skills and expand your professional potential.

Leveraging the Differences

These fishbowls are very different both in style and in kind. The most dramatic difference is the approach to collaboration and teamwork. School teaches you technical skills in software development and software engineering and some teamwork and social skills along the way. The 18 or more years you spent in school deeply entrench within you expectations and values that are sometimes at odds with the expectations of the workplace. For example, at school we are taught that our work must be our own and that reusing the work of others is a serious crime that can lead to suspension or expulsion from the institution. Conversely, in a professional software development company, few things propel a software development project more effectively and reduce time to market than code reuse. Code reuse is an objective in the workforce but a serious liability at school. Similarly, at school, people are required to do their own work, except in the cases of a few group projects. Within corporate software projects, people are expected to work together and help others with some fraction of their time even when they are not directly assigned to a project. The scope of teamwork ranges from small groups to team efforts measured in dozens or even hundreds of people working toward a collective goal to deliver a project or a product.

Many new software developers cripple their careers by clinging to the entrenched ideology they were immersed in for years that “your work must be your own.” For good reason: From early childhood until our mid-20s, this is the message we are all consistently given and ordered to follow. Post-graduation, this edict dissolves and is replaced by the belief that work should *de rigueur* be shared and collaborative. It's critical to your success that you rapidly adopt a collective model for success, code and documentation reuse, and

skills sharing. You should collaborate, share, and team with others as much as possible and give as many people credit for that effort as is reasonable.

Table 3.1 lists some other key differences in the school versus work paradigm.

TABLE 3.1 Differences Between School and Industry

CHARACTERISTIC	DIFFERENCE
Drivers and shapers	School requirements are defined by a committee of academics who define the course curriculum. Successful businesses are always driven by market demands: customers and business climate.
Freedom of motion	Within a computer science (or related) major, the range of courses is fairly limited. You can usually take a few electives each term, but they represent a small diversion from your major. In contrast, the range of what you can work on in industry is profound, crossing deeply technical development work, research, business development, management, marketing, and sales. Even within the technical space, there is a huge range of roles, from quality assurance to development, from architecture to innovation and new product generation.
Innovation	The level playing field that school provides limits the opportunities for technical innovation. Exceptions exist, particularly in graduate school thesis programs, which allow students an opportunity to do independent research. By contrast, industry thrives on innovation; it is the distinguishing feature that separates one high-tech company from another.
Interpersonal networking	Networking is useful at school, especially if you need help from classmates to stay on top of assignments. The best and brightest students don't rely heavily on networking with others. In industry, social networking is essential because it connects you to people and ideas that are necessary to help channel the organizational energy of the business.
Leadership	School is largely a level playing field, and the institution works hard to create a common circumstance for all the students within a specific major. Although some opportunities for leadership exist within school (captain of a sports team or student council), schools generally are hierarchy free within the student body. Students do not lead other students. Within industry, leadership skills and leadership roles are prized and encouraged.
Learning and doing	School is about learning, and work is about producing. It's true that school requires students to produce and industry requires employees to learn, but the focus is heavily inverted.
Payment	You pay to go to school, but you get paid to work. That means your college or university works for you, but after graduation, you work for your employer.

TABLE 3.1 Differences between School and Industry (continued)

CHARACTERISTIC	DIFFERENCE
Productivity	Educational institutions reward students with grades. The reward for doing well is a higher grade on a project or an exam. Your GPA is the measure of your success. In industry, the measurement is "contribution," which loosely translates to "productivity" in the broad sense of producing code, designs, technical innovation, new business, improved organizational efficiency or capacity, and so on. Businesses value productivity, whereas educational institutions do not.
Recognition	In school, your evaluation is designed to be fairly quantitative and measurable. Your grades on assignments, tests, and exams define your GPA. Within a professional software development organization, people have widely varying roles, and the evaluations become much more qualitative and subjective.
Risk and impact to others	The risks of messing up your education are serious, but they predominantly affect you and your family. In industry, large projects depend on the skill of the engineering team, and if you screw up, the consequences can affect the entire corporation, many of its employees, and a broad range of customers and business partners.
Personal portfolio	Neither school nor industry requires you to have a personal portfolio of any kind. By "portfolio," I mean a collection of externally visible accomplishments. Having an impressive portfolio won't help you be a better student (though, in some cases, it can help buttress your candidacy for scholarships or graduate school admittance). In a professional setting, however, having a portfolio of major innovations, successful programming projects you've delivered on time and with high quality, trade publications, scientific papers, public speaking engagements, and positions on industrial committees can help propel your career and open new opportunities for collaboration and engagement.

The greatest challenge of your first few years in industry is to unlearn what the educational environment of the past 18 years or more has drilled into you. Understand that you have entered a brave new world with radically different rules of engagement.

Unquestionably, school and industry both require a lot of hard work and are very competitive environments. Many career paths in fields other than software have a slower pace of learning and change following graduation. The rules of the game change less frequently in most professions. The software industry is distinguished by its dynamism and rapid evolution. Everything can change in a few years. The rapid changes that characterize high tech mean that software professionals are necessarily life-long learners. One thing is certain: If you were hoping for a gently loping career that largely takes care of itself, something akin to a warm, comfortable bath, then software is the wrong place for you. Expect a wild ride.

Index

Symbols

- @ symbol, 100, 105-106
- “2008 Compensation and Entrepreneurship Report in IT,” 379

A

accomplishments

- Bentley, Jon, 38
- Booch, Grady, 238
- communicating, 179-180
- Greene, Diane, 397
- Kahn, Dr. Robert, 325
- Malloy, Tom, 251-252, 257
- Mayer, Marissa, 26
- Norvig, Peter, 123
- Russinovich, Mark, 195
- Schwarz, John, 156
- Stallman, Richard, 81
- Stroustrup, Bjarne, 63
- Tomlinson, Ray, 102
- Torvalds, Linus, 171
- Wozniak, Steve, 355

achieving work-life balance, 249

- assessing, 249
- Benioff, Marc, 377
- Bentley, Jon, 40
- Booch, Grady, 241
- defining yourself, 245
- Gosling, James, 285
- Greene, Diane, 401
- Kahn, Dr. Robert, 330

- life impacts work, 247-248
- Malloy, Tom, 256
- Mayer, Marissa, 29
- Norvig, Peter, 125
- organizational culture, 246-247
- repartitioning time, 245
- Russinovich, Mark, 197
- Schwarz, John, 159
- Stallman, Richard, 83
- stressors, 248
- Stroustrup, Bjarne, 65
- tips, 246
- Tomlinson, Ray, 108
- Torvalds, Linus, 174
- Vaskevitch, David, 217
- work environment, 247
- working backward, 249

acquisition of startups

- benefits, 392
- categories, 392-393
- getting the word out to potential buyers, 394
- growth comparison, 390-392
- problems, 394-395

adaptability (startups), 389-390

- Advanced Logistics Project (ALP), 101
- advancement
 - accomplishments, communicating, 179-180
 - credibility, 178
 - evaluations, 177
 - goal-oriented careers, 181-184
 - management peer impact, 187-188

- manager influence, 185
 - career discussions, 186-187
 - helping managers look good, 185
 - technical/organizational assistance, 185
 - promotibility inversion, 189-191
 - promoting others, 189
 - advice
 - executives
 - Benioff, Marc, 372-373
 - Greene, Diane, 399
 - field opportunities, 30
 - startups, 372
 - success
 - Benioff, Marc, 371, 375, 378
 - Bentley, Jon, 41-42
 - Booch, Grady, 242-243
 - Gosling, James, 286-287
 - Greene, Diane, 403
 - Kahn, Dr. Robert, 328, 331
 - Malloy, Tom, 257-259
 - Mayer, Marissa, 27
 - Norvig, Peter, 126
 - Russinovich, Mark, 198-200
 - Schwarz, John, 159-160
 - Stallman, Richard, 83
 - Stroustrup, Bjarne, 66
 - Tomlinson, Ray, 106, 109
 - Torvalds, Linus, 175
 - Vaskevitch, David, 216-218
 - Wozniak, Steve, 358-359
 - affiliative leadership, 292
 - affinitizing, 319
 - aggressive estimates, 228
 - agile development process methodology, 272
 - agreements, 151-153
 - All the President's Men*, 300
 - ALP (Advanced Logistics Project), 101
 - angel funding, 384
 - Apple
 - computer development, 336
 - iPhone, 310
 - Jobs, Steve
 - academics, 53
 - business fluency, 337
 - stock growth, 337
 - Wozniak, Steve
 - accomplishments, 355
 - advice, 358-359
 - biography, 352-353
 - debugging, 354
 - future of software, 357
 - inspirations, 356
 - personal future, 357
 - personal success, 356
 - pet peeves, 355
 - Tetris, playing, 357
 - time management, 357
 - trends/innovations, 358
 - viewing self as hardware person, 353-354
 - architects, 112
 - The Art of Japanese Management* (Pascale), 302
 - assertions, 89
 - assessing, work-life balance, 249
 - attracting new customers, 15
 - audiences. *See* customers
 - authority
 - leadership, 291-294
 - visionaries, 333-334
 - world-class status, 334
 - awards (resumés), 47
- ## B
- bad projects, 11-12
 - balancing work-life, 244
 - achieving by working backward, 249
 - assessing, 249
 - Benioff, Marc, 377
 - Bentley, Jon, 40
 - Booch, Grady, 241
 - defining yourself, 245
 - Gosling, James, 285
 - Greene, Diane, 401
 - Kahn, Dr. Robert, 330
 - life impacts work, 247-248
 - Malloy, Tom, 256
 - Mayer, Marissa, 29
 - Norvig, Peter, 125
 - organizational culture, 246-247
 - repartitioning time, 245
 - Russinovich, Mark, 197
 - Schwarz, John, 159
 - Stallman, Richard, 83
 - stressors, 248
 - Stroustrup, Bjarne, 65
 - tips, 246
 - Tomlinson, Ray, 108
 - Torvalds, Linus, 174
 - Vaskevitch, David, 217
 - work environment, 247
 - BARC (Microsoft Bay Area Research Center), 7
 - believing in yourself, 336-337

benefits

- big shots, 7-9
- CEOs, 379
- publishing, 342
- startup acquisitions, 392

Benioff, Marc

- advice for success, 371, 375, 378
- attraction to software as a service, 371
- biography, 369-370
- future of software, 377
- getting started, 370
- graduate degrees, 376
- listening to customers, 21
- philanthropy, 373-374
- platform independence, 375
- SaaS, 374
- software executives, 372-373
- startups, 372
- time management, 377
- trends/innovations, 376
- work-life balance, 377

Bentley, Jon

- accomplishments, 38
- advice for success, 41-42
- biography, 37-38
- future of software, 41
- getting started, 38
- personal success, 39
- pet peeves, 39
- time management, 40
- trends/innovations, 40
- work-life balance, 40, 249
- writing code, 39

best and worst programmers differences, 295

betterment of society, 4

big shots, 7-9

biographies

- Benioff, Marc, 369-370
- Bentley, Jon, 37-38
- Booch, Grady, 236-237
- Gosling, James, 281
- Greene, Diane, 396
- Kahn, Dr. Robert, 323-325
- Malloy, Tom, 252
- Mayer, Marissa, 23-24
- Norvig, Peter, 122-123
- Russinovich, Mark, 192-193
- Schwarz, John, 154-155
- Stallman, Richard, 79-80
- Stroustrup, Bjarne, 62-63
- Tomlinson, Ray, 100-101
- Torvalds, Linus, 170
- Vaskevitch, David, 214-216
- Wozniak, Steve, 352-353

bloggers, 342

bonus plans, 408

Booch, Grady

- accomplishments, 238
- advice for success, 242-243
- biography, 236-237
- future of software, 241-242
- getting started, 237
- knowledge, 239
- personal success, 239
- pet peeves, 240
- Rational Software beginnings, 238
- time management, 240
- trends/innovations, 239
- work-life balance, 241

book publishing, 342

brainstorming, 312-313

Brin, Sergey, 380

Brooks, Frederick, 225, 349

browser engines, rendering, 25

buggy code, writing, 133

building

- emotional caches, 189
- teams, candidates, 295
 - attracting, 295
 - filtering, 296
 - interviewing successfully, 296-298
 - keeping best people happy, 298-299
 - mix, 299
 - prima donnas/superstars, 300
 - trust, 140

burning bridges, 128

business of software, learning, 70

business plans, 381-382

business skills, 337-338

C

C/C++, 63, 87

The C++ Programming Language
(Stroustrup), 64

candidates

- attracting, 295
- filtering, 296
- interviewing successfully, 296-298

careers

advancement

- accomplishments, communicating, 179-180
- credibility, 178
- evaluations, 177
- goal-oriented careers, 181-184
- management peer impact, 187-188
- manager influence, 185-187

- promotibility inversion, 189-191
 - promoting others, 189
- early years
 - business of software knowledge, 70
 - career paths, choosing, 74
 - expertise, 70
 - job satisfaction, 76-78
 - low-level programming skills, 71
 - mentors, 74-75
 - networking, 72-73
 - tradescraft skills, 69-70
 - watching leaders, 72
- paths, choosing, 74
- planning/placement, 49
- roles
 - architects, 112
 - CEOs, 111
 - channel salespeople, 114
 - chief architects, 112
 - CTOs, 111
 - department managers, 113
 - direct salespeople, 114
 - directors of engineering, 111
 - directors of marketing, 111
 - fellows, 111
 - function verification testers, 114
 - product managers, 113
 - program managers, 113
 - programmers, 113
 - release managers, 112
 - research staff members, 113
 - second-line managers, 112
 - system verification testers, 114
 - technical evangelists, 115
 - technical managers, 112
 - technical salespeople, 114
 - usability engineers, 113
 - Vice Presidents, 111
 - visibility, 116
- satisfaction, 8, 76-78
- statistics, 416
- success. *See* success
- CDAs (confidential disclosure agreements), 381
- CEOs, 111
 - benefits, 379
 - salary ranges, 409
- changing career goals, 183
- channel salespeople, 114
- chief architects, 112
- choosing
 - career paths, 74
 - companies, 43-46, 411-412
- Circles of Concern/Influence, 205-206
- climbing the corporate ladder, 45
- cloud computing services, 383
- co-opetition, 115
- coaching
 - interviews, 60
 - leadership, 291-292
- code
 - debugging, 88-90
 - reuse
 - innovation, 320
 - schools versus industry, 34
 - writing, 39
- coercive leadership, 291-292
- cold calling, 50
- collaboration
 - agreements, 151-153
 - communication, 149
 - development environment, 238
 - innovation, 317
- columnists, 342
- commercially available components (COTS), 320
- communication
 - effective, 117-118
 - email, 117
 - four modes of in-person, 118-120
 - importance, 116
 - interviews, 57
 - managers, 121
 - near-real-time, 179
 - personal accomplishments, 179-180
 - personal requests, 118
 - post-factum, 180
 - working the org, 149
- companies. *See also* industry
 - choosing, 43-46, 411-412
 - communication
 - effective, 117-118
 - email, 117
 - four modes of in-person, 118-120
 - importance, 116
 - managers, 121
 - personal requests, 118
 - core competencies, 162-163
 - culture
 - core competencies, 163
 - work-life balance, 246-247
 - high/low performer differences, 115
 - impressions, 121
 - roles, 110
 - architects, 112
 - CEOs, 111
 - channel salesperson, 114

- chief architects, 112
- CTOs, 111
- department managers, 113
- direct salespeople, 114
- directors of engineering, 111
- directors of marketing, 111
- fellows, 111
- function verification testers, 114
- product managers, 113
- program managers, 113
- programmers, 113
- release managers, 112
- research staff members, 113
- second-line managers, 112
- system verification testers, 114
- technical evangelists, 115
- technical managers, 112
- technical salespeople, 114
- usability engineers, 113
- Vice Presidents, 111
- visibility, 116
- visiting, 51
- working the org
 - agreements, 151-153
 - communication, 149
 - dressng for success, 150-151
 - emotional motivators, 140-143
 - getting people on-side, 140
 - negotiations, 144-149
 - problems, 130
 - social networking, 143
- compensation
 - bonus plans, 408
- companies
 - choosing, 411-412
 - differences, 405
- financial
 - industrial, 33
 - rewards, 3-4
 - salary ranges 2010-2013, 409
 - schools versus industry, 35
- graduate degrees, 405-406
- indirect, 409-411
- larger companies, 405
- preferred stock, 407
- retirement plans, 408
- startups, 405
- stock options, 406-407
- competitive offerings, 18
- complainers, 129
- confidence in interviews, 57
- confidential disclosure agreements (CDAs), 381
- constraints (time and resource), 319

- contacts
 - building, 72-73
 - finding jobs, 49
- Convolution Principle, 227
- coop students, 266
- core competencies, 162-163
- corporate influence, 4
- costs
 - patents, 338
 - software defects, 276, 279
 - startups, financing, 383-385
- COTS (commercially available components), 320
- creating opportunities, 366-367
- credibility, 121
 - career advancement, 178
 - project proposal success, 164
- critical success factors (CSFs), 203
- cross-disciplinary studying, 334-336
- crossing the chasm, 386-389
 - customer focus, 387
 - demos, 388
 - Internet, leveraging, 388
 - looking big, 389
- Crossing the Chasm* (Moore), 17, 368
- CSFs (critical success factors), 203
- CTOs, 111, 409
- culture (companies)
 - core competencies, 163
 - work-life balance, 246-247
- curriculum, 32
- customers
 - attracting new, 15
 - criteria, 18
 - existing, 15
 - feedback, 21-22
 - focus, 16
 - listening, 21
 - product development influence, 16
 - startup focus, 387
 - types, 16

D

- databases, 10
- debugging, 88-89
 - assertions, 89
 - code inspections, 90
- Design by Contract, 89
- Wozniak, Steve, 354
- decision making, market-driven, 14
- defects, 276, 279
- delegation, 303

democratic leadership, 291-292
 demos, 388
 department managers, 113
 dependency management, 227
 Design by Contract, 89
 developing

- emotional intelligence, 98-99
- goals, 183
- growth skills, 92-94
- hard skills, 84
- soft skills, 84, 95, 99
- technical skills, 85
 - debugging, 88-90
 - programming languages, 86-88
 - reviews, 91-92
- time-management skills, 93-94

 development

- based on existing customers, 15
- customer criteria, 18
- customer focus, 16
- defects, 276, 279
- maturity, testing, 275-276
- methodologies, 270-274
- popularity, 416
- quality, 18, 279-280
- SaaS example, 19-21
- technical skills, 85
- technologies, 17-18
- user feedback, 21-22
- value, 16

 direct salespeople, 114
 directed career path example, 181
 directing others, 304
 directors of engineering, 111
 directors of marketing, 111
 disruptive technology, 18
 domain expertise skills, 86
 dressing for success, 150-151
 drivers, schools versus industry, 35

E

early adopters, 17
 early career years

- business of software knowledge, 70
- career paths, choosing, 74
- expertise, 70
- job satisfaction, 76-78
- low-level programming skills, 71
- mentors, 74-75
- networking, 72-73
- tracecraft skills, 69-70
- watching leaders, 72

early majority adopters, 17
 education. *See* school
 EI (emotional intelligence), 95, 99

- developing, 98-99
- maturity and common sense, 96
- personalities, 98
- self-assessment, 98
- side-by-side comparison example, 96-97

 Eisner, Michael, 319
 elevator pitches, 164, 381
 Ellison, Larry, 53
 email

- @ symbol, 100, 105-106
- discussions, 117
- evolution, 102
- future, 104
- invention of networked, 103-104
- length, 117
- managing, 210-213

 emotional caches, building, 189
 emotional intelligence. *See* EI
 emotional motivators, 140-143

- face time, 142
- giving something for nothing, 141
- inspiring self-worth in others, 141
- sharing glory, 142

 employment agencies, 49
 engines, rendering in browsers, 25
 enjoying work, 5-6
 enthusiasm, 8, 58
 estimates, 227-228
 evaluations (employee), 177
 executives

- advice
 - Benioff, Marc, 372-373
 - Greene, Diane, 399
- salary ranges, 409
- schedule overruns, 221

 existing customers, 15
 expertise

- defined, 190
- early career, 70

 external business plans, 382
 extracurricular activities

- resumés, 48
- value, 54-55

F

face-saving exits, 265
 face-to-face private meetings, 118
 Fagan, Michael, 90

failures
 rates, 12
 startups, 380
 success, 416
 visionaries, 348

fame, 8

feedback, 21-22

fellows, 111

financing startups, 383
 angel funding, 384
 fourth-stage, 385
 investment capital, 385
 second-stage, 384
 third-stage, 385

finding
 innovation opportunities, 311-312
 jobs. *See* job searches
 mentors, 74-75

first few months at work, learning curve,
 361-362
 colleagues, 364
 good mentors, 362
 managers, 362-363
 managing your boss, 366
 opportunities, creating, 366-367
 pushing yourself, 368

following the money, 301

formal code reviews, 90

FORTRAN, 86

four modes of in-person communication,
 118-120
 face-to-face private meetings, 118
 large group presentations, 119-120
 small group discussions, 118
 small group presentations, 118

fourth-stage funding, 385

free software, 82

freedom of motion, 35

fun work environments, 76-78

function verification testers, 114

fundamental technical skills, 85

future
 email, 104
 software, 416
 Benioff, Marc, 377
 Bentley, Jon, 41
 Booch, Grady, 241-242
 Gosling, James, 285-286
 Greene, Diane, 402
 Kahn, Dr. Robert, 330
 Malloy, Tom, 258
 Mayer, Marissa, 29
 Norvig, Peter, 125

Russinovich, Mark, 199
 Schwarz, John, 158
 Stallman, Richard, 83
 Stroustrup, Bjarne, 66-67
 Tomlinson, Ray, 108
 Torvalds, Linus, 174
 Vaskevitch, David, 217
 Wozniak, Steve, 357

G

Gates, Bill
 academic grades, 53
 business fluency, 337

Gerstner, Lou, 13, 208

getting people on-side, 140

getting started
 Benioff, Marc, 370
 Bentley, Jon, 38
 Booch, Grady, 237
 Gosling, James, 282-283
 Greene, Diane, 397
 Kahn, Dr. Robert, 325
 Malloy, Tom, 252
 Mayer, Marissa, 24
 Norvig, Peter, 123
 Russinovich, Mark, 193
 Schwarz, John, 155
 Stallman, Richard, 80
 Stroustrup, Bjarne, 63
 Tomlinson, Ray, 101
 Torvalds, Linus, 171
 Vaskevitch, David, 216

getting to revenue, 385-386

Gladwell, Malcolm, 73, 334

goal-centric time management, 202-203

goal-oriented careers, 181-184
 changing with time, 183
 developing, 183
 directed career path example, 181
 trying different jobs, 183
 undirected career path example, 181
 working backward, 184

goal-oriented project management, 261-264

goals, 226-227

good ideas versus good business, 380-381

good software
 market embracing, 12
 market-driven, 14-15

Google
 founders, 380
 successful innovation, 315
 work environment, 247

Gosling, James
 advice, 286-287
 biography, 281
 future of software, 285-286
 getting started, 282-283
 graduate degrees, 287
 Java inspirations, 283
 personal success, 284
 pet peeves, 284
 time management, 285
 trends/innovations, 284
 work-life balance, 285

gossiping, 128

government agencies, 50

GPS, value perception cycle, 314

graduate degrees
 Benioff, Marc, 376
 compensation, 405-406
 Gosling, James, 287
 Greene, Diane, 403
 Malloy, Tom, 255
 Mayer, Marissa, 28
 Russinovich, Mark, 199
 Schwarz, John, 160
 Stroustrup, Bjarne, 65
 Tomlinson, Ray, 106
 Vaskevitch, David, 216

graphical user interface, 340

Gray, Jim, 7

Greene, Diane
 accomplishments, 397
 advice for success, 403
 architecture degree, 397
 biography, 396
 future of software, 402
 getting started, 397
 graduate degrees, 403
 influences/inspirations, 399
 leading software companies, 399
 personal success, 400
 pet peeves, 397
 starting VMware, 398-399
 time management, 401
 work-life balance, 401

growth
 problems, 135-137
 skills, 92-94
 startup acquisition comparison, 390-392
 statistics, 416

gurus. *See* visionaries

H

halo-effect, 121

Hamilton, James, 12, 211-212

hard skills, 84

high performers, 115

higher-level programming languages, 87

human nature, managing, 264-266
 face-saving exits, 265
 making people feel loved, 264
 status reports, 265

humility in interviews, 57

I

IBM
 Gerstner, Lou's guiding principles, 13
 IMS, 10
 patents, 341
 successful innovation, 315

identifying successful innovations, 309

images, revitalizing, 121

impressions, 121

IMS, 10

In Search of Excellence, 303

incremental improvements, 18

indecision (time management), 207

indirect compensation, 409-411

individual work (schools), 32

industry. *See also* companies
 compared to schools, 34-36
 code reuse, 34
 drivers and shapers, 35
 financial compensation, 35
 freedom of motion, 35
 innovation, 35
 interpersonal networking, 35
 leadership, 35
 learning/doing, 35
 personal portfolios, 36
 productivity, 36
 recognition, 36
 risks, 36
 teamwork, 34

compensation. *See* compensation culture
 core competencies, 163
 work-life balance, 246-247

early careers
 business of software knowledge, 70
 career paths, choosing, 74
 expertise, 70

- job satisfaction, 76-78
- low-level programming skills, 71
- mentors, 74-75
- networking, 72-73
- trecraft skills, 69-70
- watching leaders, 72
- four modes of in-person communication, 118-120
- influence, measure of success, 4
- leaders, watching, 72
- limited field of vision, 33-34
- publications, 341
- influences
 - Greene, Diane, 399
 - managers on employee success, 185-187
 - Mayer, Marissa, 24
 - Vaskevitch, David, 216
- informal code walkthroughs, 90
- innovation, 17
 - 2005 McKinsey report on corporate success concerns, 307
 - benefits, 306
 - brainstorming, 312-313
 - business advantage, 308
 - fostering successful
 - affinitizing, 319
 - avoiding dictating solutions, 318
 - best people, 316
 - carrying through, 322
 - collaboration, 317
 - current state of the art, 321
 - data, 319
 - Google, 315
 - IBM, 315
 - iteration, 318
 - market as source, 321
 - Microsoft, 315
 - reusing existing components, 320
 - rewards, 316
 - studying other fields, 321
 - time and resource constraints, 319
 - keeping up
 - Benioff, Marc, 376
 - Bentley, Jon, 40
 - Booch, Grady, 239
 - Gosling, James, 284
 - Malloy, Tom, 255
 - Mayer, Marissa, 28
 - Norvig, Peter, 125
 - Russinovich, Mark, 197
 - Schwarz, John, 157-158
 - Stallman, Richard, 82
 - Stroustrup, Bjarne, 65
 - Tomlinson, Ray, 107
 - Torvalds, Linus, 173
 - Vaskevitch, David, 217
 - Wozniak, Steve, 358
 - opportunities, 311-312
 - resumés, 48
 - schools versus industry, 35
 - self-made innovators, 336-337
 - successful, 309-311
 - identifying, 309
 - iPhone example, 310
 - marketing, 309
 - OS/2 flop example, 310
 - value perception cycle, 313-315
 - inspections (code), 90
 - inspirations
 - Greene, Diane, 399
 - Wozniak, Steve, 356
 - interface designs (Mayer, Marissa), 26
 - internal business plans, 382
 - Internet
 - finding jobs, 50
 - frequency of top 24 programming languages and platforms, 87
 - leveraging for startups, 388
 - release cycles, 29
 - server warehouses, 30
 - internship students, 266
 - interpersonal networking, 35
 - interviews, 56-61
 - candidates, 296-298
 - coaching, 60
 - communication, 57
 - company negativity, 59
 - confidence/humility, 57
 - enthusiasm, 58
 - falsifying information, 60
 - follow-up etiquette, 60
 - interest in software, 57
 - knowing the competition, 58
 - Norvig, Peter, 124
 - personal/medical information, 59
 - positive, 58
 - research, 56
 - selling yourself, 59
 - skill-testing questions/brainteasers, 56
 - investment capital, 385
 - iPhone, 310
 - IQ, 96
 - iteration
 - development methodology, 271
 - innovation, 318

J

Java, 87. *See also* Gosling, James
 Jericho, 101
 job roles, 110
 architects, 112
 CEOs, 111
 channel salespeople, 114
 chief architects, 112
 CTOs, 111
 department managers, 113
 direct salespeople, 114
 directors of engineering, 111
 directors of marketing, 111
 fellows, 111
 function verification testers, 114
 product managers, 113
 program managers, 113
 programmers, 113
 release managers, 112
 research staff members, 113
 second-line managers, 112
 system verification testers, 114
 technical evangelists, 115
 technical managers, 112
 technical salespeople, 114
 usability engineers, 113
 Vice Presidents, 111
 visibility, 116
 job satisfaction, 8, 76-78
 job searches
 career planning/placement offices, 49
 choosing companies, 43-46
 cold calls, 50
 employment agencies, 49
 extracurricular activities, 54-55
 friends/contacts, 49
 government agencies, 50
 importance of grades, 52-54
 Internet searches, 50
 interviewing, 56-61
 coaching, 60
 communication, 57
 company negativity, 59
 confidence/humility, 57
 enthusiasm, 58
 falsifying information, 60
 follow-up etiquette, 60
 interest in software, 57
 knowing the competition, 58
 personal/medical information, 59
 positive, 58
 research, 56

 selling yourself, 59
 skill-testing questions/brainteasers, 56
 newspaper postings, 51
 professors, 50
 resource effectiveness, 51-52
 resumés, 46-48
 mailing, 51
 school performance, 46
 student positions, 55
 visiting companies, 51
 Jobs, Steve
 academics, 53
 business fluency, 337

K

Kahn, Dr. Robert
 accomplishments, 325
 advice for success, 328, 331
 biography, 323-325
 future of software, 330
 getting started, 325
 networking contributions, 326
 personal success, 327
 pet peeves, 326
 pivotal career point, 326
 software scale, 327
 technology/trends, 329
 time management, 329
 work-life balance, 330
 Kay, Alan, 350
 Kennedy, John F., 261

L

laggards, 17
 large group presentations, 119-120
 late majority adopters, 17
 lateness. *See* overruns
 Law of the Few, 143
 leadership. *See also* visionaries
 authority, 293-294
 delegation, 303
 directing others, 304
 following the money, 301
 innovation
 2005 McKinsey report on corporate
 success concerns, 307
 affinitizing, 319
 avoiding dictating solutions, 318
 benefits, 306
 best people, 316
 brainstorming, 312-313

- business advantage, 308
- carrying through, 322
- collaboration, 317
- current state of the art, 321
- data, 319
- Google, 315
- IBM, 315
- identifying valuable, 309
- iPhone example, 310
- iteration, 318
- market as source, 321
- marketing, 309
- Microsoft, 315
- opportunities, 311-312
- OS/2 flop example, 310
- reusing existing components, 320
- rewards, 316
- studying other fields, 321
- successful, 309-311
- time and resource constraints, 319
- value perception cycle, 313-315
- managing, compared, 288-290
- proceeding without formal support, 294
- resumés, 47
- rewards, 301-302
- schedule overruns, 235
 - average example, 223
 - avoiding, 235
 - causes, 224-231
 - effects, 223
 - executive perception, 221
 - handling, 232-233
 - leadership, 235
 - right people, 235
 - statistics, 222
- schools versus industry, 35
- shared values, 303
- styles, 290-292
- teams, building, 295-300
- lean development methodology, 273
- learning
 - business of software, 70
 - expertise, 70
 - low-level programming skills, 71
 - programming languages, 86-88
 - schools versus industry, 35
 - tracecraft skills, 69-70
- learning curve (first few months at work), 361-362
 - colleagues, 364
 - good mentors, 362
 - managers, 362-363
 - managing your boss, 366

- opportunities, creating, 366-367
- pushing yourself, 368
- length
 - emails, 117
 - resumés, 48
- life impacts work, 247-248
- Lightstone's Convolution Principle, 227
- limited field of vision, 32
 - industry, 33-34
 - school, 33
 - school as, 32
- Linux, 173. *See also* Torvalds, Linus
- LogAD (Logistics Anchor Desk), 101
- low performers, 115
- low-level programming skills, learning, 71

M

- mailing resumés, 51
- Malloy, Tom
 - accomplishments, 251-252
 - advice for success, 257-259
 - biography, 252
 - chief software architect role, 254
 - future of software, 258
 - getting started, 252
 - graduate degrees, 255
 - personal success, 254, 257
 - pet peeves, 259
 - risk-taking decisions, 253
 - technical accomplishments, 257
 - time management, 256
 - trends/innovations, 255
 - work-life balance, 256
- The Management of Innovation*, 303
- managers
 - communication, 121
 - employee's first few months, 362-363
 - influences on employee success, 185-187
 - managing, 366
 - peer impact, 187-188
- managing
 - dependency, 227
 - leadership, compared, 288-290
 - managers, 366
 - projects
 - defects, 276, 279
 - development methodologies, 270-274
 - goal-oriented, 261-264
 - human nature, 264-266
 - maturity, testing, 275-276
 - plans, changing, 269-270
 - quality, 279-280

- SMART, 262
- students, 266
- value, measuring, 267-269
- schedule overruns, 232-233
- marketing, 309
- marketplace
 - driving software, 14-15
 - embracing good software, 12
 - influence on schools, 33
- Master's degrees, 406
- maturity, testing, 275-276
- Mayer, Marissa
 - accomplishments, 26
 - advice for success, 27
 - biography, 23-24
 - choosing Google, 43
 - field opportunities, 30
 - future of software, 29
 - getting started, 24
 - graduate degrees, 28
 - influences, 24
 - listening to customers, 22
 - personal success, 27
 - pet peeves, 25
 - time management, 29
 - trends/innovation, 28
 - women in computing, 26
 - work-life balance, 29
- measuring
 - success, 3-6
 - value, 267-269
- mentors, 74-75, 362
- methodologies (development), 270
 - agile development process, 272
 - iterative development, 271
 - lean development, 273
 - rapid prototyping, 274
 - scrum, 271
 - waterfall development process, 274
 - XP, 273
- Meyer, Bertrand, 89
- mezzanine funding, 385
- Microsoft
 - Bay Area Research Center (BARC), 7
 - stock growth, 337
 - successful innovation, 315
 - work environment, 247
- mistakes
 - fundamentals versus incidentals, 137
 - growth
 - excessive self-promotion, 136
 - self-marketing wrong messages, 136

- technical skills, 135
- unimportant roles, 137
- people, 127
 - burning bridges, 128
 - complainers, 129
 - gossiping, 128
 - inappropriate decision making, 131
 - snitching on coworkers, 129
 - too many suggestions, 130
 - working the org, 130
- productivity, 133
 - buggy code, 133
 - lack of productivity, 134
 - missing dates consistently, 134
 - skunkworks focus, 135
 - time management, 135
- team, 131-132
- Moore, Geoffrey, 17, 386
- motivation, emotional, 140-143
 - face time, 142
 - giving something for nothing, 141
 - inspiring self-worth in others, 141
 - sharing glory, 142
- The Mythical Man Month*, 225

N

- NDAs (nondisclosure agreements), 381
- near-real-time communication, 179
- negotiating, 144
 - meeting in the middle, 145-147
 - mutual gain options, 147-148
 - outcomes, estimating, 145
 - quitting while ahead, 149
 - seeking first to understand, 144
- networked email, 103-104
- networking
 - early career, 72-73
 - Kahn, Dr. Robert's contributions, 326
 - wasting time, 209
 - working the org, 143
- new customers, attracting, 15
- new graduates
 - career paths, choosing, 74
 - finding jobs, 52-54
 - industry leaders, watching, 72
 - job satisfaction, 76-78
- learning
 - business of software, 70
 - expertise, 70
 - low-level programming skills, 71
 - tracecraft skills, 69-70

- mentors, 74-75
- networking, 72-73
- resumés, 46
- school performance, 46
- new writer advice, 344-345
- newspaper job postings, 51
- nondisclosure agreements (NDAs), 381
- North American salary ranges 2010-2013, 409
- Norvig, Peter
 - accomplishments, 123
 - advice for success, 126
 - biography, 122-123
 - future of software, 125
 - getting started, 123
 - job interview example, 124
 - personal success, 124
 - pet peeves, 124
 - time management, 125
 - trends/innovations, 125
 - work-life balance, 125

O

- OOP (object-oriented programming), 63
 - languages, 88
 - Stroustrup, Bjarne, 63
- open source, 383
- opportunities
 - creating, 366-367
 - innovations, 311-312
 - Mayer, Marissa, 30
- OS/2 flop, 310
- Outliers*, 334
- overruns
 - average example, 223
 - avoiding, 235
 - causes, 224
 - adding manpower, 225
 - aggressive estimates, 228
 - bad estimates, 227-228
 - dependency management, 227
 - fundamentals, bypassing, 230
 - human performance, 224
 - scope creep, 224
 - social dynamics, 229-230
 - strategic changes, 231
 - time away from deliverables,
 - forgetting, 231
 - unclear goals, 226-227
 - effects, 223
 - executive perception, 221
 - handling, 232-233

- leadership, 235
- right people, 235
- statistics, 222

P

- pacesetter leadership, 291-292
- Page, Larry, 380
- paralysis by analysis, 207
- Parkinson's Law, 230
- passion, 350-351
- patents, 338
 - career success, 341
 - costs, 338
 - criteria, 338
 - graphical user interface example, 340
 - names, 341
 - style-mixing pants example, 339
 - worthy ideas, 340
- Patton, General George, 207
- people problems, 127
 - burning bridges, 128
 - complainers, 129
 - gossiping, 128
 - inappropriate decision making, 131
 - snitching on coworkers, 129
 - too many suggestions, 130
 - working the org, 130
- performance (schedule overruns), 224
- Perlis, Alan, 86
- personal portfolios, 36
- personal success, defining, 415
- personal tenacity (proposals), 168
- personal walkthroughs, 90
- pet peeves
 - Bentley, Jon, 39
 - Booch, Grady, 240
 - Gosling, James, 284
 - Greene, Diane, 397
 - Kahn, Dr. Robert, 326
 - Malloy, Tom, 259
 - Mayer, Marissa, 25
 - Norvig, Peter, 124
 - Russinovich, Mark, 199
 - Schwarz, John, 160
 - Stallman, Richard, 81
 - Stroustrup, Bjarne, 65
 - Tomlinson, Ray, 109
 - Torvalds, Linus, 172
 - Vaskevitch, David, 217
 - Wozniak, Steve, 355
- Peters, Tom, 303

- Ph.Ds, 406
- philanthropy (Benioff, Marc), 373-374
- pitching proposals, 166-168
- plans (project), changing, 269-270
- platforms
 - independence, 375
 - Internet frequency of top 24 programming and platforms, 87
- popularity of software development, 416
- post-factum communication, 180
- preferred stock, 407
- presentations
 - large group, 119-120
 - small group, 118
 - summary, 381
- proactiveness (urgency), 208
- problems
 - acquisitions, 394-395
 - fundamentals versus incidentals, 137
 - growth
 - excessive self-promotion, 136
 - self-marketing wrong messages, 136
 - technical skills, 135
 - unimportant roles, 137
- people, 127
 - burning bridges, 128
 - complainers, 129
 - gossiping, 128
 - inappropriate decision making, 131
 - snitching on coworkers, 129
 - too many suggestions, 130
 - working the org, 130
- productivity, 133
 - buggy code, 133
 - lack of productivity, 134
 - missing dates consistently, 134
 - skunkworks focus, 135
 - time management, 135
- team, 131-132
- product development
 - based on existing customers, 15
 - customer criteria, 18
 - customer focus, 16
 - quality characteristics, 18
 - SaaS example, 19-21
 - technologies, 17-18
 - user feedback, 21-22
 - value, 16
- product managers, 113
- productivity
 - problems, 133-135
 - schools versus industry, 36
- professional activities, 48
- professors, 50
- program managers, 113
- programmers
 - 9-to-5 job, 108
 - high/low performers, 115
 - role, 113
- programming languages
 - Internet frequency of top 24 programming languages and platforms, 87
 - learning, 86-88
- Programming Pearls*, 38
- project management
 - defects, 276, 279
 - development methodologies, 270
 - agile development process, 272
 - iterative development, 271
 - lean development, 273
 - rapid prototyping, 274
 - scrum, 271
 - waterfall development process, 274
 - XP, 273
 - goal-oriented, 261-264
 - human nature, 264-266
 - maturity, testing, 275-276
 - plans, changing, 269-270
 - quality, 279-280
 - SMART, 262
 - students, 266
 - value, measuring, 267-269
- projects
 - failures, 11-12
 - good
 - market embracing, 12
 - market-driven, 14-15
 - managing. *See* project management
 - overruns
 - average example, 223
 - avoiding, 235
 - causes, 224-231
 - effects, 223
 - executive perception, 221
 - handling, 232-233
 - leadership, 235
 - right people, 235
 - statistics, 222
- proposals
 - connecting privately with decision makers, 164
 - converting naysayers to believers, 163
 - core competencies, 162-163
 - credibility, 164
 - elevator pitch, 164
 - example, 165

- executive commitment, 169
 - others' ideas, including, 164
 - personal tenacity, 168
 - pitching, 166-168
- skunkworks, 135
- promotibility inversion, 189-191
- promoting others, 189
- promotions. *See* advancement
- proprietary software, 194
- proposals
 - connecting privately with decision makers, 164
 - converting naysayers to believers, 163
 - core competencies, 162-163
 - credibility, 164
 - elevator pitch, 164
 - example, 165
 - executive commitment, 169
 - others ideas, including, 164
 - personal tenacity, 168
 - pitching, 166-168
- public speaking, 345-348
 - adapting to audiences, 346
 - antagonists, 348
 - clarity and pace, 346
 - engaging audience, 346
 - first person, 347
 - hand motions, 347
 - purpose, 348
 - smiling, 347
 - speaking the charts, 347
 - understanding audience, 346
- publishing, 343
 - benefits, 342
 - books, 342
 - columns/blogs, 342
 - impact, 341
 - industrial/trade publications, 341
 - new writer advice, 344-345
 - research papers, 341
 - value, 343
 - whitepapers, 342
- pushing yourself, 368

Q-R

- quadrants of time, 93-94
- quality
 - products, 18
 - testing, 279-280
- R&D, bootstrapping, 382-383
- rapid prototyping development methodology, 274

- Rational Software beginnings, 238
- recognition, 36
- recruiting candidates
 - attracting, 295
 - filtering, 296
 - interviewing successfully, 296-298
- relational databases, 156
- release cycles, 29
- release managers, 112
- remote workers, 383
- rendering browser engines, 25
- research papers, 341
- research staff members, 113
- resources
 - constraints, 319
 - finding jobs
 - career planning/placement offices, 49
 - cold calls, 50
 - effectiveness, 51-52
 - employment agencies, 49
 - friends/contacts, 49
 - government agencies, 50
 - Internet searches, 50
 - mailing resumé, 51
 - newspaper postings, 51
 - professors, 50
 - visiting companies, 51
- responsiveness (startups), 389-390
- resumés
 - awards, 47
 - details, 47-48
 - extracurricular activities, 48
 - grades, 52-54
 - innovations, 48
 - leadership roles, 47
 - length, 48
 - mailing, 51
 - new graduates, 46
 - professional activities, 48
 - technical domain skills, 47
- retirement plans, 408
- revenue (startups), 385-386
- reviews (technical), 91-92
- revitalizing images, 121
- rewards, 301-302, 316
- risks
 - schools versus industry, 36
 - toleration, 17
- roles (companies), 110
 - architects, 112
 - CEOs, 111
 - channel salesperson, 114
 - chief architects, 112

- CTOs, 111
 - department managers, 113
 - direct salespeople, 114
 - directors of engineering, 111
 - directors of marketing, 111
 - fellows, 111
 - function verification testers, 114
 - product managers, 113
 - program managers, 113
 - programmers, 113
 - release managers, 112
 - research staff members, 113
 - second-line managers, 112
 - system verification testers, 114
 - technical evangelists, 115
 - technical managers, 112
 - technical salespeople, 114
 - usability engineers, 113
 - Vice Presidents, 111
 - visibility, 116
 - Russinovich, Mark
 - accomplishments, 195
 - advice for success, 198-200
 - biography, 192-193
 - future of software, 199
 - getting started, 193
 - graduate degrees, 199
 - Microsoft recruitment, 195
 - personal success, 196
 - pet peeves, 199
 - proprietary software, 194
 - time management, 196
 - trends/innovations, 197
 - Windows incorporation of tools, 195
 - Winternals development, 193-194
 - work-life balance, 197
- ## S
- SaaS (software as a service), 19, 374
 - Benioff, Marc, 374
 - technology adoption example, 19-21
 - salary ranges 2010-2013, 409
 - Salesforce.com, 369
 - satisfaction, 76-78
 - schedule overruns
 - average example, 223
 - avoiding, 235
 - causes, 224
 - adding manpower, 225
 - aggressive estimates, 228
 - bad estimates, 227-228
 - dependency management, 227
 - fundamentals, bypassing, 230
 - human performance, 224
 - scope creep, 224
 - social dynamics, 229-230
 - strategic changes, 231
 - time away from deliverables,
 - forgetting, 231
 - unclear goals, 226-227
 - effects, 223
 - executive perception, 221
 - handling, 232-233
 - leadership, 235
 - right people, 235
 - statistics, 222
 - Schmidt, Eric, 380
 - school
 - career planning/placement offices, 49
 - compared to industry, 34-36
 - curriculum, 32
 - importance of grades, 52-54
 - individual work, 32
 - limited field of vision, 32-33
 - market influences, 33
 - performance, 46
 - student positions, 55
 - Schwarz, John
 - accomplishments, 156
 - advice for success, 159-160
 - biography, 154-155
 - career goals, 184
 - future of software, 158
 - getting started, 155
 - graduate degrees, 160
 - path to CEO position, 156
 - personal success, 160
 - pet peeves, 160
 - time management, 158
 - trends/innovations, 157-158
 - work-life balance, 159
 - scope creep, 224
 - scrum development methodology, 271
 - second-line managers, 112
 - second-stage funding, 384
 - self-made innovators, 336-337
 - server warehouses, 30
 - shapers, 35
 - shared values, 303
 - sharing glory, 142
 - Sharpen the Saw story, 92-94
 - skills
 - business, 337-338
 - communication
 - effective, 117-118

- email, 117
- four modes of in-person communication, 118-120
- importance, 116
- managers, 121
- personal requests, 118
- growth, 92-94
- hard, 84
- leadership
 - authority, 293-294
 - delegation, 303
 - directing others, 304
 - following the money, 301
 - innovation. *See* leadership, innovation managing, compared, 288-290
 - proceeding without formal support, 294
 - rewards, 301-302
 - schedule overruns, 235
 - shared values, 303
 - styles, 290-292
 - teams, building, 295-300
- low-level programming, 71
- soft
 - defined, 84
 - developing, 84, 95
 - emotional intelligence, 95-99
- technical
 - debugging, 88-90
 - development, 85
 - domain expertise, 86
 - fundamental, 85
 - growth required, 85
 - problems, 135
 - programming languages, 86-88
 - reviews, 91-92
 - tradcrafter, 86
 - time-management, 93-94
 - tradcrafter, 69-70
- skunkworks projects, 135
- small group discussions/presentations, 118
- SMART (specific, measurable, attainable, realistic, timely), 262
- SMPs (symmetric multiprocessors), 267
- socializing
 - dynamics, 229-230
 - networking, 143, 349-350
 - wasting time, 209
- soft skills
 - defined, 84
 - developing, 84, 95
 - emotional intelligence, 95-99
 - developing, 98-99
 - maturity and common sense, 96
 - personalities, 98
 - self-assessment, 98
 - side-by-side comparison example, 96-97
- software
 - as a service. *See* SaaS
 - scale, 327
- space program project management example, 261-264
- specific, measurable, attainable, realistic, timely (SMART), 262
- Stallman, Richard
 - accomplishments, 81
 - advice for success, 83
 - biography, 79-80
 - free software champion, 82
 - future of software, 83
 - getting started, 80
 - personal success, 81
 - pet peeves, 81
 - time management, 83
 - trends/innovations, 82
 - work-life balance, 83
- startups
 - acquisition
 - benefits, 392
 - categories, 392-393
 - getting the word out to potential buyers, 394
 - problems, 394-395
 - adaptability, 389-390
 - Benioff, Marc advice, 372
 - business plans, 381
 - compensation, 405
 - crossing the chasm, 386-389
 - elevator pitches, 381
 - external business plans, 382
 - failures, 380
 - financing, 383
 - angel funding, 384
 - fourth-stage, 385
 - investment capital, 385
 - second-stage, 384
 - third-stage, 385
 - getting to revenue, 385-386
 - good ideas versus good business, 380-381
 - growth versus acquisition, 390-392
 - internal business plans, 382
 - R&D, bootstrapping, 382-383
 - responsiveness, 389-390
 - summary presentations, 381
- statistics
 - job growth, 416
 - overruns, 222
- status reports, 265

- stock options, 406-407
- stressors, 248
- Stroustrup, Bjarne
 - accomplishments, 63
 - advice for success, 66
 - biography, 62-63
 - The C++ Programming Language*, 64
 - future of software, 66-67
 - getting started, 63
 - graduate degrees, 65
 - personal success, 64
 - pet peeves, 65
 - time management, 65
 - trends/innovations, 65
 - work-life balance, 65
- students
 - limited field of vision, 32
 - managing, 266
 - positions, 55
- styles, leadership, 290-292
- success. *See also* advancement
 - advice
 - Benioff, Marc, 371, 375, 378
 - Bentley, Jon, 41-42
 - Booch, Grady, 242-243
 - Gosling, James, 286
 - Greene, Diane, 403
 - Kahn, Dr. Robert, 328, 331
 - Malloy, Tom, 257-259
 - Mayer, Marissa, 27
 - Norvig, Peter, 126
 - Russinovich, Mark, 198-200
 - Schwarz, John, 159
 - Stallman, Richard, 83
 - Stroustrup, Bjarne, 66
 - Tomlinson, Ray, 106
 - Torvalds, Linus, 175
 - Vaskevitch, David, 216
 - Wozniak, Steve, 358-359
 - characteristics, 8
 - credibility, 121
 - defining, 415
 - dressings for success, 150-151
 - enthusiasm, 8
 - failures, 416
 - fame, 8
 - freedom, 7
 - fun work environment, 76-78
 - innovation, 309-311
 - affinitizing, 319
 - avoiding dictating solutions, 318
 - best people, 316
 - carrying through, 322
 - collaboration, 317
 - current state of the art, 321
 - data, 319
 - Google, 315
 - IBM, 315
 - identifying, 309
 - iPhone example, 310
 - iteration, 318
 - market as source, 321
 - marketing, 309
 - Microsoft, 315
 - OS/2 flop example, 310
 - reusing existing components, 320
 - rewards, 316
 - studying other fields, 321
 - time and resource constraints, 319
 - IQ, 96
 - job satisfaction, 8
 - luck, 414
 - management peers impact, 187-188
 - manager influence, 185-187
 - measures
 - betterment of society, 4
 - corporate/industrial influence, 4
 - enjoyment, 5-6
 - financial compensation, 3-4
 - patents, 341
 - personal
 - Bentley, Jon, 39
 - Booch, Grady, 239
 - Gosling, James, 284
 - Greene, Diane, 400
 - Kahn, Dr. Robert, 327
 - Malloy, Tom, 254, 257
 - Mayer, Marissa, 27
 - Norvig, Peter, 124
 - Russinovich, Mark, 196
 - Schwarz, John, 160
 - Stallman, Richard, 81
 - Stroustrup, Bjarne, 64
 - Tomlinson, Ray, 103
 - Torvalds, Linus, 172
 - Vaskevitch, David, 218
 - Wozniak, Steve, 356
 - positive impressions, 121
 - project proposals
 - connecting privately with decision makers, 164
 - converting naysayers to believers, 163
 - core competencies, 162-163
 - credibility, 164
 - elevator pitch, 164
 - example, 165
 - executive commitment, 169
 - others ideas, including, 164

- personal tenacity, 168
- pitching, 166-168
- qualities, 413-414
- rates, 12
- summary presentations, 381
- symmetric multiprocessors (SMPs), 267
- Sysinternals, 195
- System 360 project, 349
- system verification testers, 114

T

- task-centric time management, 203-205

- teams

- building, 295
 - attracting, 295
 - filtering, 296
 - interviewing successfully, 296-298
 - keeping best people happy, 298-299
 - mix, 299
 - prima donnas/superstars, 300
- co-opetition, 115
- problems, 131-132
- schools versus industry, 34

- technical domain skills, 47

- technical evangelists, 115

- technical managers, 112

- technical salespeople, 114

- technical skills

- debugging, 88-89
 - assertions, 89
 - code inspections, 90
 - Design by Contract, 89
- development, 85
- domain expertise, 86
- fundamental, 85
- growth requirements, 85
- problems, 135
- programming languages, 86-88
- reviews, 91-92
- tradecraft, 86

- technology. *See also* innovation

- adoption lifecycle, 17
 - customer criteria, 18
 - quality characteristics, 18
 - SaaS example, 19-21
 - technologies, 18
- disruptive, 18

- tenacity (urgency), 208

- TENEX, 102

- testing

- maturity, 275-276
- quality, 279-280

- thinking inside the box, 319

- third-stage funding, 385

- time

- constraints (innovation), 319

- management

- 1950s-1980s, 201

- 1990s, 201

- Benioff, Marc, 377

- Bentley, Jon, 40

- Booch, Grady, 240

- Circles of Concern/Influence, 205-206

- email, 210-213

- goal-centric, 202-203

- Gosling, James, 285

- Greene, Diane, 401

- indecisions, 207

- Kahn, Dr. Robert, 329

- Malloy, Tom, 256

- Mayer, Marissa, 29

- Norvig, Peter, 125

- problems, 135

- Russinovich, Mark, 196

- Schwarz, John, 158

- skills, developing, 93-94

- Stallman, Richard, 83

- Stroustrup, Bjarne, 65

- task-centric, 203, 205

- time wasting, 209-210

- Tomlinson, Ray, 107

- Torvalds, Linus, 173

- urgency, 208-209

- Vaskevitch, David, 217

- Wozniak, Steve, 357

- wasting, 209-210

- time-shared computing, 100

- The Tipping Point: How Little Things Can*

- Make a Big Difference* (Gladwell), 73, 143

- Tomlinson, Ray

- @ sign, 103-106

- accomplishments, 102

- advice for success, 106, 109

- biography, 100-101

- evolution of email, 102

- future of email, 104

- future of software, 108

- getting started, 101

- graduate degrees, 106

- invention of networked email, 103-104

- personal feelings toward email, 104

- personal success, 103

- pet peeves, 109

- programming as a 9-to-5 job, 108

- time management, 107

trends/innovations, 107
 work-life balance, 108

Torvalds, Linus
 accomplishments, 171
 advice for success, 175
 biography, 170
 future of software, 174
 getting started, 171
 personal success, 172
 pet peeves, 172
 time management, 173
 trends/innovations, 173
 work-life balance, 174

trade publications, 341

tracraft skills, 69-70, 86

trends, keeping up. *See also* innovation
 Benioff, Marc, 376
 Bentley, Jon, 40
 Booch, Grady, 239
 Gosling, James, 284
 Kahn, Dr. Robert, 329
 Malloy, Tom, 255
 Mayer, Marissa, 28
 Norvig, Peter, 125
 Russinovich, Mark, 197
 Schwarz, John, 157-158
 Stallman, Richard, 82
 Stroustrup, Bjarne, 65
 Tomlinson, Ray, 107
 Torvalds, Linus, 173
 Vaskevitch, David, 217
 Wozniak, Steve, 358

trust, 140

types
 audiences, 16
 leadership, 291-292

U-V

unclear goals, 226-227

undirected career path example, 181

urgency (time management), 208-209

usability engineers, 113

value
 extracurricular activities, 54-55
 perception cycle, 313-315
 products, 16
 publishing, 343
 shared, 303
 software, measuring, 267-269

Vaskevitch, David
 advice for success, 216-218
 biography, 214-216

future of software, 217
 getting started, 216
 graduate degrees, 216
 influences, 216
 personal success, 218
 pet peeves, 217
 time management, 217
 trends/innovations, 217
 work-life balance, 217, 249

vice presidents, 111

visibility, 116

visionaries. *See also* leadership
 authority, 333-334
 believing in yourself, 336-337
 business skills, 337-338
 cross-disciplinary studying, 334-336
 defined, 333
 failures, 348
 passion, 350-351
 patents, 338
 career success, 341
 costs, 338
 criteria, 338
 graphical user interface example, 340
 names, 341
 style-mixing pants example, 339
 worthy ideas, 340

public speaking, 345-348
 adapting to audience, 346
 antagonists, 348
 clarity and pace, 346
 engaging audience, 346
 first person, 347
 hand motions, 347
 purpose, 348
 smiling, 347
 social networking, 350
 speaking the charts, 347
 understanding audience, 346

publishing, 343
 benefits, 342
 books, 342
 columns/blogs, 342
 impact, 341
 industrial/trade publications, 341
 new writer advice, 344-345
 research papers, 341
 value, 343
 whitepapers, 342
 self-made innovators, 337

visiting companies, 51

VMware foundation, 398-399

W-Z

- wasting time, 209-210
- watching industry leaders, 72
- waterfall development process, 274
- Welch, Jack, 186
- whitepapers, 342
- Winternals, 193-194
- women in computing, 26
- work environment, 247
- work-life balance, 244
 - achieving by working backward, 249
 - assessing, 249
 - Benioff, Marc, 377
 - Bentley, Jon, 40
 - Booch, Grady, 241
 - defining yourself, 245
 - Gosling, James, 285
 - Greene, Diane, 401
 - Kahn, Dr. Robert, 330
 - life impacts work, 247-248
 - Malloy, Tom, 256
 - Mayer, Marissa, 29
 - Norvig, Peter, 125
 - organizational culture, 246-247
 - repartitioning time, 245
 - Russinovich, Mark, 197
 - Schwarz, John, 159
 - Stallman, Richard, 83
 - stressors, 248
 - Stroustrup, Bjarne, 65
 - tips, 246
 - Tomlinson, Ray, 108
 - Torvalds, Linus, 174
 - Vaskevitch, David, 217
 - work environment, 247
- working the org
 - agreements, 151-153
 - communication, 149
 - dressing for success, 150-151
 - emotional motivators, 140-143
 - getting people on-side, 140
 - negotiations, 144
 - meeting in the middle, 145-147
 - mutual gain options, 147-148
 - outcomes, estimating, 145
 - quitting while ahead, 149
 - seeking first to understand, 144
 - problems, 130
 - social networking, 143
- world-class authority status, 334
- worst and best programmers differences, 295
- Wozniak, Steve, 336
 - accomplishments, 355
 - advice, 358-359
 - biography, 352-353
 - debugging, 354
 - future of software, 357
 - inspirations, 356
 - personal future, 357
 - personal success, 356
 - pet peeves, 355
 - Tetris, playing, 357
 - time management, 357
 - trends/innovations, 358
 - viewing self as hardware person, 353-354
- writing code, 39
- XP development methodology, 273