The Async-First Playbook
Remote Collaboration Techniques for Agile Software Teams

SUMEET GAYATHRI MOGHE

FREE SAMPLE CHAPTER
Praise for
The Async-First Playbook

“Async is the superpower the most successful distributed teams have already realized, so understanding the playbook for implementing async-first practices is a must-do for any leader moving into this type of operating system. Sumeet has done a phenomenal job outlining that here.”

—Chase Warrington, Head of Remote, Doist

“Collaboration is going async. The advantages are just too great: global reach—which enables more inclusion—and time zone immunity. Sumeet excellently decomposes the issues. And by zeroing in on async leadership, he tackles the elephant in the async room: leadership. Yes, you can be an effective leader, asynchronously.”

—Cliff Berg, Co-Founder and Managing Partner, Agile 2 Academy

“The Async-First Playbook is a must-read for anyone part of agile teams in today’s fast-paced and distributed work environment. This insightful book shows how adopting an async-first approach can transform the way teams collaborate. The result—higher productivity, better inclusion, and most importantly, fun! The future of work is remote, and async-first collaboration is the key to unlocking its potential. Through concrete examples, this book provides a roadmap for embracing asynchronous communication, enabling teams to focus on deep work and stay in a state of flow. I recommend this book to anyone seeking to unlock the full potential of remote collaboration and achieve better outcomes for their team.”

—Iwo Szapar, Co-founder & Head of Remote, Remote-First Institute

“Asynchronous work, when implemented well, can help create a sustainable and calm way of working. It enables individuals to design their days around productivity and priorities, creating a happier and more human-centric workplace. The Async-First Playbook thoroughly examines effective asynchronous work practices’ benefits (and the challenges, too). This playbook discusses fundamental prerequisites that should be in place before transitioning to an async-first approach, including leadership practices, potential challenges, and a starter kit to help you determine your next steps. If you seek to alleviate information overload and minimize meetings, The Async-First Playbook is a valuable resource.”

—Lisette Sutherland, Director of Collaboration Superpowers
https://www.collaborationsuperpowers.com/
“At a time when organisations are experimenting with how to make the most out of remote work, Sumeet offers a step-by-step approach on how to make software development practices more inclusive and future-proof. Furthermore, the section aimed directly at managers and leaders also turns the book into a strong advocacy-tool for sustainable remote collaboration.”

—Pilar Orti, Director of Virtual not Distant

“The Async-First Playbook is a comprehensive guide for teams and leaders looking to optimize their remote work practices for a distributed world. The pandemic helped us see that knowledge work really shouldn’t have a place, and now it’s time to recognize and embrace that this work can be better accomplished asynchronously, too. Location-independent work? That’s the first step in the transformation. Have you tried time-independent work? Sumeet shows us how.”

—Tyler Sellhorn, host, The Remote Show podcast

“The Async-First Playbook is not just another book on Agile practices. It’s a game-changer for teams looking to optimize their collaboration through asynchronous communication. In this comprehensive guide, you’ll find everything you need to know about the principles, frameworks, tools, workflows, and cultural changes necessary to embrace an async-first approach. Even though I’ve been working in a remote setup with an async-first approach for many years, I still found improvements that I’ll implement with my team. I found especially valuable the chapter on “The Async-First Leadership Mindset,” and I can’t wait to share this book with my team.”

—Oana Calugar, Collaboration Consultant, Mural

“Every software company has recently been forced to adapt to remote work, and subsequently workers have awakened to the possibilities afforded by more flexible working. Companies have been experimenting and iterating to find a new equilibrium between in-office, hybrid, and fully remote work, and there have been many fits and starts. What’s been missing is a practical guide on how to make the most of this newly popular work style so that both employees and employers win. Sumeet’s book perfectly fills the void, giving teams and leaders the tools they need to escape the trap of synchronicity and make the most of their newfound asynchronous flexibility.”

—Patrick Sarnacke, Managing Director, Thoughtworks UK

“The COVID-19 pandemic has provoked many publications about remote and hybrid work models. While benefits such as improved work-life balance and diversity have been praised, concerns about weakened personal relationships and burnout are also significant. One crucial element is that, too often, the cons result from companies attempting to put in place novel models while retaining old office-centric practices. That is why I like this book. In The Async-First Playbook, Sumeet Gayathri Moghe offers a practical, hands-on approach
to making remote work effective for software development businesses. He emphasizes defaulting to action and provides actionable recommendations for optimizing distributed work arrangements. As someone who has led an organization through a similar journey and had valuable conversations with Sumeet while doing that, I can attest that this book packages knowledge from real-world techniques with a successful track record."

—Matheus Tait, Managing Director, Thoughtworks Spain

“Building on commonly established working patterns that have been adopted by many in the agile software development movement, Sumeet deftly re-examines the goals of these practices, and identifies remote-friendly alternatives that can lead your team to new heights. Of particular note is the emphasis on honing both writing and reading skills, and on guarding the time to use them both effectively. Those looking to navigate this new frontier would be well advised to set some time aside and read this first!”

—Andy Yates, Head of Technology, Corporate Strategy, Thoughtworks

“The pandemic forced the tech workforce all over the world to work remotely. And since then, the focus has been on overcoming the challenges of remote working. Sumeet argues that the opportunity is bigger—to work differently, adopting async-first approaches and practices, to increase productivity and creativity by creating space for deep work and better collaboration amongst teams. *The Async-First Playbook* is a complete practitioner’s guide to learn and implement async ways of working for yourself, your team, and your organization. In a new distributed and remote working world, this book has the potential to unlock productivity, better teamwork, and allow for improved work-life balance. A total must-read for technology or knowledge workers.”

—Sameer Soman, Managing Director, Thoughtworks India

“Want to boost productivity and cut costs in your engineering organization? Sumeet’s new book offers practical, actionable recommendations for optimizing team performance and streamlining processes via asynchronous ways of working. In today’s digital-first world, boards are taking notice of the skyrocketing cost of engineering. Sumeet's book provides a valuable perspective on this issue, offering strategies for achieving better results in less time and at a lower cost. Whether you’re a seasoned engineering leader or just starting out, this is a must-read for anyone looking to drive change and improve their organization’s performance in a highly remote setting.”

—Sagar Paul, Head of Global Solutions, Thoughtworks

*The Async-First Playbook* is the book that the distributed agile world has been eagerly awaiting for years! Sumeet’s direct and to-the-point writing style hits hard and very convincingly challenges our traditional norms. The book stays true to the format of a playbook by
providing recommendations that are highly actionable and will enable the stated impact. An amazing must-read for software professionals.”

—Santosh Mahale, Director, Engineering Group | e4r (Engineering for research)

“The COVID-19 pandemic has accelerated the adoption of remote work, and it is now here to stay. However, many companies are still struggling to adapt to this new reality. This is where The Async-First Playbook comes in. In this book, the author makes a compelling argument for why async-first practices are essential for modern knowledge work. He guides the reader through the foundational elements needed to create an async-first culture, from individual and team practices to leadership and management styles. The author also highlights the challenges that come with remote work and provides readers with practical advice on how to avoid common pitfalls. Whether you’re a developer, manager, or senior leader, this book will provide you with the tools you need to succeed in the new world of work.”

—Sunil Mundra, Org. Change and Transformation Leader, CXO Advisor, Thoughtworks
THE ASYNC-FIRST PLAYBOOK
Pearson’s Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world’s leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where:

• Everyone has an equitable and lifelong opportunity to succeed through learning.
• Our educational products and services are inclusive and represent the rich diversity of learners.
• Our educational content accurately reflects the histories and experiences of the learners we serve.
• Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview).

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

• Please contact us with concerns about any potential bias at https://www.pearson.com/report-bias.html.
This page intentionally left blank
To every worker who’s had joy sucked out of their life by a pointless meeting.
This page intentionally left blank
CONTENTS

Foreword by Martin Fowler xxii
Foreword by Darren Murph xxiii
Preface xxv
Acknowledgments xxix
About the Author xxxiii

PART I  ADAPTING TO THE NEW NORMAL  1

Chapter 1 There’s Got to Be a Better Way to Work  3
Work Deserves a New Look  5
Complex Problems Need Smarter Collaboration  6
The Promise of Flexibility  6
Escaping the Shallows  7
A More Inclusive Workplace  9
Defaulting to Action  9
Improved Knowledge Sharing and Communication  10
A Better Work Environment for All of Us  11

Chapter 2 Foster a Mindset for Change  13
Four Simple Ideas  13
Focus on Value  13
Visualize Change on a Spectrum  14
Make a Ulysses Pact 17
Appreciate the Balance 17
Go Far, Go Together 19
Ready for the First Steps 19

PART II PREPARE TO GO ASYNC-FIRST 21
Chapter 3 The Tools You Need 23
With Tools, Less Is More 23
Tools Are Everything; Tools Are Nothing 27
Chapter 4 The Biggest Async-First Superpower 29
Writing Is a Practice; Documentation a Product 30
Why Meetings Are the Last Resort, Not the First Option 30
The Benefits of Writing 31
What About the Agile Manifesto? 33
How Do We Write Better? 34
How Much More Effective Is Face-to-Face Communication? 40
The Manifesto Deserves the Benefit of Doubt 40
To Work Async-First, We Must Write 41
Chapter 5 Three More Async-First Superpowers 43
What an Async-First Superhero Looks Like 43
Block Distractions 44
Read and Comprehend 45
Work Independently 46
Personal Productivity Leads to Team Productivity 47
Chapter 6 Calm Things Down with Collaboration Protocols 49
Work Execution vs. Workflow 49
Revisit Your Workflow Statuses and Transitions 50
Make Email and IM Secondary Communication Tools 52
Streamline Your Decision-Making Process 53
Identify Channels and Their Response Times 55
Fundamentals, Fundamentals, Fundamentals 56
PART III  THE PRACTITIONER’S GUIDE  57

Chapter 7  Meetings as the Last Resort  59
Just. Too Many. Meetings.  59
  The ConveRel Quadrants  61
When You Meet, Make It Count  63
  Account for Costs of Frequent Cross–Time Zone Meetings  64
  Stay Human  66
  Async-First, with Small Shifts  66

Chapter 8  The Value of Being Face to Face  69
From URL to IRL  69
  The Time-Zone Question  69
  Face-to-Face Collaboration Isn’t Free  70
  The True Value of Being in Person  71
  Camaraderie Has Dollar Value  72
  Rethink the Onsite-Offsite  73
  Be Conscious About Building Relationships  74
  So Much for Meetings  75

Chapter 9  Micro-Moves to Shift Left  77
Small Shifts, Easy Wins  77
  Personal Shifts: Broadcast Your Commitment  77
  Team Shifts: Accelerate Leftward  81
  Build Async-First Behavioral Cues  85

Chapter 10  Write a Team Handbook  87
Distributed Teams Need a Single Source of Truth  87
  By Default, Start with Your Team  88
  What Goes into a Handbook?  89
Tools You Can Use  92
  Start Small, Own Collectively, Be Iterative  93
  Go Deep with Your Documentation Triggers  94
  Aim for a Shared Reality  95
Chapter 11  Tame the “Instant” in Instant Messaging  
Make Messaging Productive  
  If It’s Urgent, Use a Suitable Medium  
  Choose Intuitive Names for Channels and Rooms  
  Show Your Status, Respect Others’ Statuses  
  Limit the Number of Messages  
  Get to the Point  
  Target Your Conversations  
  React, Don’t Respond  
  Use Threads: One per Topic  
  Slow Things Down, and Don’t Argue on Chat  
  Make Your IM Guidelines Visible  
Messaging: Just Not Instant  

Chapter 12  Standup Meetings: An Easy Shift Left  
Distributed Standups Can Be Painful  
  Use the Project Management Tool’s Features  
  Automate the Ritual  
  Keep It Relevant  
  Make Peace with Lag  
Standups = Conveyance/Strong Relationships  

Chapter 13  Take Charge of Your Development Cycles  
Sprint Ceremonies Can Hinder Async Work  
  Approach 1: Embrace Continuous Flow  
  Approach 2: Use “Shape Up” Cycles  
The Key to Asynchrony Is a Strong Process  

Chapter 14  Run Meaningful Retrospectives  
Infrequent Retros Lead to Poor Team Health  
  View Retrospectives as a Process and Not an Event  
  Collect Inputs in Safety  
  Voting Synchronously vs. Asynchronously  
  Setting Up the Retrospective Environment  
  Retros as a Process, Not an Event  
Scrum for the 2020s
Chapter 15  Kickoffs and Desk Checks: Reduce Ritualized Interruptions  137
   How to Maintain Quality with Fewer Meetings?  138
      Approach 1: Go Async with Kickoffs and Desk Checks  138
      Approach 2: Keep the Sync Practices and Plan for Them  144
   Make Your Feedback Loops “Remote Native”  145

Chapter 16  Questions to Reimagine Your Tech Huddles  147
   The What and the Why  148
      How Much Autonomy Exists in Your Teams?  148
      How Necessary Is the “Sync Up”?  150
      What If You Defaulted to Action?  150
      If You Must Huddle, How Do You Make the Meeting Effective?  151
      Oh, and About the Regular, Scheduled Huddles…  152
   Not a Zero-Sum Game  153

Chapter 17  Pair Programming: The Elephant in the Room  155
   A Polarizing Topic  155
      Let’s Start with Why  156
      Design for Flexibility  157
      Use the Right Tools  158
      Encourage Personal Discipline  158
      Mix Pairing and Solo Work  160
   If It’s Fun for You, Pair by All Means  161

Chapter 18  Audit Trails from the Flow of Your Work  163
   The “Just Ask” Pattern Breaks Down  163
      Meeting Notes  164
      Business Decision Records  166
      Architectural Decision Records  166
      Commit Messages  168
      Pull Requests  169
   Trails as the Most Frequent Form of Documentation  171
Chapter 19  Communicate Tech and Functional Design  173
   An Agile Approach to Design  173
      Idea Papers  174
      Feature Breakdown Documents  175
      Technical Design Documents  176
   Simplify Communication Complexity  178

Chapter 20  Two Stable Pieces of Handbook Documentation  181
   Being Agile About Documentation  181
      Your Ways of Working  182
      Codebase README Files  184
      Documenting Code? Think Twice  185
   Good Documents Reduce Guesswork  186

Chapter 21  Craft an Efficient Onboarding Process  189
   Write Once, Run Many Times  189
      Preserve the Dumb Questions Budget with an FAQ  190
      Build an Onboarding Checklist  191
      Focus on Automation and Reusability  193
      Find the New Hire a Buddy  194
      Foster Strong Relationships  195
   Onboarding Efficiency = Team Efficiency  195

PART IV  ASYNC-FIRST LEADERSHIP  197

Chapter 22  The Async Leadership Mindset  199
   The Tyranny of “The Way”  199
      How Can You Get the Most Out of Yourself?  200
      How Can You Be an Example, Not a Bottleneck?  202
      How Can You Avoid Busywork?  203
      How Can You Champion Inclusion?  204
      How Can You Make Your Team Resilient?  205
   Make Time for the Essential Stuff  206
# Table of Contents

**Chapter 23**  
Manage Your People with Care  
Corrections in the Right Direction  
Manage Your Own Workload  
Meet People One-on-One  
Practice Radical Candor  
Be the Bridge Between Your Team and Your Company’s Culture  

**Chapter 24**  
Set Up Your Team for Success  
Design for Success  
Manage Team Cognitive Load Through Team Topologies  
Revisit Your Team’s Internal Structure  
Reduce Pressure, Create Calm  
Consciously Build a Team Culture  
Align on a Common Purpose  
Your Virtual Workplace Needs Configuring  

**Chapter 25**  
Farm Tacit Knowledge in Your Company  
Beyond Handbooks: Into Communities  
Create Porous Walls  
Facilitate Flows While You Manage Stocks  
From Team Knowledge to Company Knowledge  

## Part V  
**NAVIGATE THE PITFALLS**  

**Chapter 26**  
The Great Hybrid Kerfuffle  
People’s Preferences Are Heading Remote  
Hybrid Organization, Not Hybrid Employees  
Avoid a Move Backward  
Avoid New Costs for All Stakeholders  
Embrace Science, Not Superstition  
Don’t Create a Perception of Asymmetry  
Treat Everyone as “Remote”  
Choice and Autonomy Are the Key Words  

**Chapter 27**  
The Async Island  
Unpacking Organizational Inertia  
Forces for and Against Change  
Async Work Is a New Sport  

## CONTENTS

Help Your System Get Better 259
Protect and Extend the Island 260
While Being a Guerilla, Don’t Forget Advocacy 265

### Chapter 28  Toxicity in the Virtual Workplace 267
Toxicity Builds: One Benign Step at a Time 267
Celebrating the Hard Worker 267
Digital Presenteeism 268
Talking to the Document 269
Not Investing in Meaningful Synchrony 270
As a Leader, Stay Vigilant 271

### PART VI  BRING IT ALL TOGETHER 273

### Chapter 29  The Async-First Starter Kit 275
Five Stages of Sensible Defaults 275
Stage 1: Align on Goals 276
Stage 2: Tabulate Baseline Data 278
Stage 3: Agree on the Fundamentals 280
Stage 4: Clean Up Your Calendars 282
Stage 5: Build the 30-Day Program and Beyond 283
A Team Shift, Owned by the Team 285

### Chapter 30  A Brave New World of Work 287
Another World of Work Is Possible 289
Flexibility as a Desire and a Right 289
A Skills Economy 290
Digital Nomadism 291
The Four-Day Workweek 292
Gen Z and Their Sensibilities 294
A Shift for Autonomy 295
It’s Time to Sign Off 295

Endnotes 297
Index 321
Two decades ago in a workshop, we looked for a name for a new approach to software development. We settled on “agile,” but one of the suggestions was “conversational.” What I liked about “conversational” was that it emphasized how we were advocating a more collaborative approach to building software. Instead of tasks carved out by managers and architects, delivered by (usually poorly written) documents, we wanted people to talk to each other. Database engineers could collaborate with those coding business logic using that data. Programmers should talk to their users about what software would be most valuable for their work. Communication is the bloodstream of a software effort, and we wanted to recognize that and increase its flow.

A few years later, Thoughtworks, my employer, opened offices in India. We had embraced and pioneered this agile approach to software development, and it was showing great success. But how could this work with a development center in India? The close collaboration we were encouraging was one thing when it was just a walk down a hall to talk to someone; what did it mean when separated by thousands of miles and a dozen time zones?

We found ways to make things work, and in time, technology came to assist us further. By the early 2010s, we were making heavy use of video-call technologies for our meetings (a habit that served us well when COVID-19 hit in 2020).

Sumeet Gayathri Moghe has seen this activity first-hand, working with us in India through most of this time. He’s managed several teams working across space and
time zones in highly collaborative ways. As he’s done this work, he’s been reflecting on how effective our techniques are and how to improve them.

The video call has helped meetings work better when people are sitting far from each other, but even when in the same room, meetings aren’t always the best way to collaborate. Questions that come up are often not best answered off the cuff—people need time to reflect. I like writing not just to communicate, but as a tool to arrange my thinking into a coherent structure, even for something as simple as this Foreword. When working across time zones, asynchronous activities become even more valuable, as they reduce the necessity to work at inconvenient hours.

It’s a common slogan to hear “meetings are bad—get rid of them.” But that’s a simplistic reaction to the frustration of meetings that are badly run or inappropriately used. Sumeet has reflected on what situations require meetings and when they can (and should) be replaced by asynchronous techniques. This book sets out what he’s learned and is a guide to improving our collaboration even as we sit further apart.

—Martin Fowler
We’ve come a long way. Pioneers like GitLab wrote the playbook for remote work over a decade ago, yet it took a global pandemic to open the world’s eyes to the possibilities of remote work. Today, many organizations embrace what we call “location independence.” This encapsulates the magic that happens when a business decouples results from physical geography. While this has transformed millions of lives and enterprises, it’s only the start of an even greater revolution.

The future of remote work—or, dare I say, the future of work—is time independence. Since at least the advent of the internet, the majority of knowledge workers have remained bound to time. The way we design our lives and the dreams we allow ourselves to contemplate are constrained by a fixed reality: the rigidity of a workday.

What happens when we shake off that rigidity? What design principles can humans apply to their own lives when we leverage tools, software, AI, and workflows to achieve professional goals outside the strict bounds of time?

We begin to optimize for what matters most. We reprioritize our individual identity stacks—the layers that make us individually us. Businesses become stronger and more resilient to crises. They become magnets for the most talented people, those who value flexibility over all else and who generate otherworldly results as a measure of their gratitude.

The “async-first” mindset is indeed what powers this time-independent way of working.
It’s the key to saying “yes” to more midday adventures with my toddler. It’s the cornerstone of the nonlinear workday—a concept I detailed in my own book, *Living the Remote Dream*. At the same time, it’s a way to drive high-performing teams—something we’ve espoused for years as part of the “TeamOps” philosophy at GitLab.

Change, however, is hard. Concrete, tangible advice is tough to come by. We need frameworks and recipes to ensure that the asynchronous collaboration movement takes hold and scales to the ends of the earth.

*The Async-First Playbook* is one such resource. This book is the bridge between principle and execution. As opposed to being everything to everyone, Sumeet has focused this book on teams that build software. This, I’m sure you’ll notice, makes much of the advice in the book immediately actionable for you, the reader. The focus by no means alienates other readers, though. Technologist or not, everyone can learn something about working asynchronously from this book.

There’s a lot at stake. If workplace leaders get this right, we’ll create a more flexible, inclusive, connected planet. This is where this book shines. Sumeet doesn’t just focus on the everyday tasks that such teams perform, but he also sheds light on the responsibilities of effective managers and leaders in an async-first environment. His part-guerrilla, part-advocate approach is a great way for the intrepid yet enlightened leader to drive change in their realm of influence while they gather evidence and arguments for broader organizational change. And all the advice in the book comes to life with tool suggestions, examples, and resources for your team.

To be successful at asynchronous ways of working is to discover more of what makes work great—flexibility, space for deep work, frictionless knowledge-sharing, a bias for action, diversity, inclusion, and the ability to scale. By removing constraints through the power of technology, we empower society to put its focus on areas like community, service, and progress.

The plays in this playbook will empower you to lead and build differently. All that you need is an open mind and a belief that the status quo isn’t worthy of defining the future.

—Darren Murph, Future of Work Architect and VP, Workplace Design & Remote Experience at Andela
Welcome!

In March 2020, the world of work was in a frenzy. The pandemic had pushed everyone back to the safety of their homes, and it felt like our jobs and work were at risk. In the knowledge working and consulting industry, we were fortunate to have a higher exposure to modern collaboration tools. So, we coped better than some other industries that weren’t as sophisticated with tool usage. As the months rolled on, our comfort with remote work grew to where we are today—most knowledge workers want to work remotely most of the time.

My path to remote work was slightly different from that of my colleagues. In the years preceding the pandemic, I had played several roles where remote work was the norm. For example, with one of my clients, I was the only Indian consultant in a group of 50-odd technologists, most of whom worked across Central and Eastern Europe on a suite of different products. It made more sense for me to work from my home office in Pune, India, than to commute to my company office several kilometers away. Through sheer coincidence, I became proficient with remote work well before most of my company colleagues.

As you can imagine, my clients, colleagues, and I had different pathways to the remote software development experience. With trial and error and some dumb luck, I’d realized well before the pandemic that a meeting-centric approach to remote work is unsustainable in the long run. Of course, the ability to jump into conversations and to tap each other on the shoulder models the office environment
that we’re all used to, but the office was no paragon of productivity. Mimicking the “office on the cloud” with modern tools re-creates the problems of the office at a much higher rate than earlier. The casualties? Our sense of satisfaction with work and our productivity.

Yet, it wasn’t until I switched to part of the company serving North American clients from India that I could see the full extent of this “office in the cloud” dysfunction. For those of you in other countries, I want you to think of winter in the northern hemisphere. It’s a 13.5-hour time difference between India and the West Coast of the United States! There’s no overlap during standard office hours between these time zones. Even the East Coast of the United States isn’t much better. But to this day, many of my colleagues in the tech industry straddle opposing time zones to collaborate with each other. The casualty? Our work–life balance.

But we needn’t normalize these casualties. It doesn’t have to be crazy at work. The tech industry will continue to grow. And at its heart, tech needs access to talented people. And those people won’t always be in the same time zone as us. So, I believe distributed working is an inevitable part of the future of work. In recognition of that inevitability, this book aims to help software development practitioners build high-performing, distributed teams. Along the way, I hope I can show you how to make your processes less stressful, more efficient, more fun, more inclusive, and more thoughtful.

**WHO SHOULD READ THIS BOOK**

I’ve written this book for practitioners and leaders with a basic understanding of agile software development. If you’re already working on a distributed or remote team, you’ll recognize many of the challenges I describe in the book. You’ll also see how going “async-first” can address many of those challenges. You may also be a manager or a leader who wants to guide their team or organization to a remote-native way of building software. In this case, this book will provide you with a blueprint to organize your teams. If successful, your experiments may catalyze broader change for your companies and clients.

People who’ve read early drafts of this book have told me it could be useful for a wider audience as well. Many chapters of the book that deal with collaboration and management practices are domain-agnostic. Indeed, if you’re a curious knowledge worker, you needn’t be a technologist to get value out of the book. I leave it to you to find your adventure with the table of contents!
HOW I’VE STRUCTURED THIS BOOK

As the name suggests, I’ve written this book as a practical guide to remote-native software development. Asynchronous collaboration is a means to that end. However, if you care about being agile and not just doing agile, you may not follow every practice I describe in this book. So, it’s perfectly fine to read only the chapters that apply to your work. This isn’t a book that I expect you to read from cover to cover.

To present the plays of this playbook, I’ve organized them into six different parts.

- Part I, “Adapting to the New Normal,” introduces the value of asynchronous collaboration in our day and age and teaches you how to prepare your team to shift to an async-first way of working.
- Part II, “Prepare to Go Async-First,” addresses the basics you need in place through tools, skills, and protocols before you tinker with your work practices.
- Part III, “The Practitioner’s Guide,” is the chunkiest section of this playbook. It aligns techniques for remote software development, with chapters that map to common software development practices. You can read every chapter if you’re curious. But first, read the chapters that relate to practices you already follow.
- Part IV, “Async-First Leadership,” teaches you how to be a supportive leader for your async-first team. The techniques in this part of the book will apply to people who are already in management or leadership roles or those who aspire to play such roles.
- Part V, “Navigate the Pitfalls,” recognizes a few common traps when working async-first. No team is immune to these, so the mitigating techniques I describe in these chapters should interest every reader.
- Part VI, “Bring It All Together,” closes out the book with a set of tools that’ll help you guide your async-first shift.

So, dip in and dip out of the book as you shift yourself to an async-first way of working. Keep it close at hand so that when you’re ready to introduce a practice to your team, you can refer to the plays relevant to it.

THE COMPANION SITE

Writing this book began by sharing some of my ideas with a broad audience. The companion website for this book, https://www.asyncagile.org, will continue to be the place where I’ll share and discuss future musings related to this topic. As I learn, I want to learn aloud with you.
In many of the chapters, you’ll see references to resources, examples, and templates that I’ve created for you. You’ll find them all at https://www.asyncagile.org/book-resources. I encourage you to bookmark this page, so you always have the link handy.

With all that said, welcome, thank you for joining me, and let’s get started!
ACKNOWLEDGMENTS

As a first-time author, I have many people to thank for bringing this book to life. First, I must thank the OGs, particularly pioneers such as Jason Fried and David Heinemeier Hansson and companies such as GitLab, which have been passionate advocates of remote and async work. I stand on the shoulders of such intellectual giants. Of them, there are many.

Closer to home, I consider myself fortunate to have worked at Thoughtworks, a pioneer of agile software development, particularly in IT consulting. To be perfectly honest, most of the ideas in this book stem from my experiences in working at Thoughtworks. I've curated several techniques in this book that I learned only because other colleagues at the company took the pains to teach me.

I must reserve a special word of thanks to colleagues like Santosh Mahale, Sagar Paul, Sameer Soman, and Thoughtworks India's Engineering Group, who afforded me the time to work on this book. Their support represents why Thoughtworks is such a welcome environment for sharing what we learn.

I wouldn't have started on the journey of creating this book had it not been for the guidance and advice I received from Martin Fowler. Not only did he help me validate the ideas for this book, but he also took time out of his busy schedule to read my rather messy manuscript. I'll always value the feedback he shared with me. The book is richer for it. Martin has also written a foreword for this book, which I consider a massive endorsement. Speaking of forewords, this book has two. The other one is from Darren Murph, one of the most popular remote-work
leaders out there. Even while settling into a new role, Darren took the time to write a thoughtful note about the book. I can’t thank him enough for it.

I also received tons of advice and encouragement from Gayathri Mohan, Paulo Caroli, Sunil Mundra, Vinod Sankaranarayanan, and Mangalam Nandakumar. As published authors, they were generous enough to share their experiences with me, so I had a frame of reference for what writing a book is like.

Sachin Dharmapurikar, Vijay Raghavan Aravamudhan, Sunita Venkatachalam, and Allison MacQueen gave me feedback on several pieces of writing. They’ve enriched the ideas in this book with their input. And how can I forget Dr. James Stanier? He was the technical reviewer for this book, and his candid, point-by-point feedback has helped me sharpen the book into the product you see today.

When ideas collide, new ideas emerge. Amy Luckey, Daniel Pupius, Mohammed Najiullah, and Priya Darshini M G helped me out by writing about their ideas and experiences of working and leading remotely. Their thoughts have inspired a lot of the content you see in the book.

Greg Doench, my editor at Pearson, was one of the first individuals to believe in the concept of this book, demonstrating unwavering faith from the start. His contagious enthusiasm has been a driving force throughout this project. Menka Mehta from Pearson India has been my go-to person throughout the development of this book. If I had a question, she always had an answer. Dr. Chris Zahn, the developmental editor for this book, offered me insightful written feedback that deepened my understanding of style guides and the process of crafting a polished manuscript. My copy editor, Kim Wimpsett, went through my manuscript with a fine-toothed comb. Her attention to detail helped me uncover many blind spots in my writing style. Julie Nahil and Charlotte Kughen guided me through production and design activities that I’d never experienced before writing this book. Without them and the entire Pearson brigade, this book wouldn’t be in your hands.

As I take these ideas to more people, I must acknowledge the role of colleagues such as Pranav Shah, Ashwin Lal, Aneesh Chandhok, Ming Linsley, Shipra Shandilya, Jem Elias, and Jaydeep Chakrabarty for creating consulting and speaking opportunities for me. Even the idea of async-first work sometimes needs synchronous storytelling!

The last few years have taught me how valuable friends and family are. I must acknowledge those people in my life. Nagarjun Kandukuru has been a friend for the last decade and a half at Thoughtworks and an advocate for asynchronous
collaboration. His support for my work has been heartwarming. My close friends, Anuroop Krishnan, Manish Vaidya, Chetna Soni, and Surali Vaidya, have been among my biggest cheerleaders. And my lovely wife, Gayathri, has always been my first sounding board. She must be tired of listening to my ideas and reading every first draft of what I write, and yet she humors me. If you enjoy reading the book, it’s because it first passed through her filters. My kids—Avni and Vihaan—don’t know yet that this book was a labor of love for their dad. But they were understanding enough to leave me alone when I was writing. They’ll know how much that has meant to me when they grow up. For now, I put my gratitude in writing. I promise to make up for any time I lost with them.

Finally, a big shout-out to everyone who followed and supported Asyncagile.org in its infancy. You know who you are. You needn’t have patronized me at the time. And yet, you took the time to read what I had to say, and you spread the word around. Without you, I’d have never had the confidence to write this book. Thank you, whoever you are, and wherever you are.
ABOUT THE AUTHOR

Sumeet Gayathri Moghe is an agile enthusiast, product manager, and design nerd at Thoughtworks. He’s worked with a variety of clients over the last few decades, building software products and helping them improve their engineering effectiveness. During his time as a consultant, Sumeet has exposed himself to various domains such as retail, travel, telecom, payments, healthcare technology, education, and more. That, in turn, has helped him generalize his experience across industries. What you see in this book stands for what he’s learned from his colleagues, their successes, and their occasional misadventures.

Sumeet now lives in Pune, India, with his wife and two kids. When he’s not at work building software, you’ll probably see him looking through a camera’s eyepiece, photographing wildlife or a scene in the wilderness.

To get in touch, find him on Linkedin (@sumeetmoghe) or visit asyncagile.org where he blogs almost every week. If you’re curious about his photography work, check out www sumsutmoghe.com.
The COVID-19 pandemic forced the knowledge work industry into remote work. Today several companies and teams continue to work remotely, though they still follow office-centric work practices. In this part of the book, I argue that these ways of working must change so we can take advantage of our new, distributed work arrangements.

- **Chapter 1**, “There’s Got to Be a Better Way to Work,” sets the scene for how asynchronous work has the potential to transform distributed software development so it’s fun, sustainable, inclusive, and scalable.
- **Chapter 2**, “Foster a Mindset for Change,” teaches you how to prime your team for a shift to an async-first way of working. Before you explore the playbook, the ideas from this chapter will help you create a framework to align your colleagues to the change.
This page intentionally left blank
6:25 am  Nita pulls herself out of bed, having pushed snooze on her alarm five times already. Oh boy, she’s late! She shakes her son, Abin, awake. They barely have 30-odd minutes before they drive to school. Dang it—she has to drive. She’d much prefer to take the scooter, but it’s pouring outside. If only she could control the Indian monsoon.

Okay, 30 minutes. Freshen up, make some coffee, and get Abin’s lunchbox ready. Drat! She forgot to order a milk delivery. Thank heavens she has some milk powder. She hustles to get a couple of sandwiches ready as she fries some eggs so Abin can have breakfast.

7:45 am  That wasn’t the start Nita was hoping for. Life as a single mom is hard. She barely got her son to school on time. She’d have liked to wake up early and stay on top of things, but she went to bed late and didn’t get a good night’s sleep. Work doesn’t begin till noon. Maybe she can sleep now? Or maybe not. She’s got to cook, get ready, take out the trash, and put away the dishes. That won’t happen by itself, will it?

12:00 pm  Nita works a midday shift. Work at home in Bangalore begins at 12 pm and, with an hour’s break thrown in, ends at 10 pm. She works for a client in Boulder, Colorado, and some of her American teammates are based out of Boise, Idaho. This is the only way she gets an hour to overlap with them for meetings. She tried getting 30 minutes of shut-eye before starting work, but she ended up just tossing and turning in bed. The day has barely begun, and she already feels tired.
2:30 pm Time to pick up Abin. There goes 40 minutes of her break time. Lunch was already at breakneck speed—coffees and dinners will probably need to be at the worktable. Sigh.

5:00 pm As a designer, Nita is trying to build some wireframes for a new feature the team is scheduled to deliver in an upcoming sprint. But she has a meeting with the team’s business analyst in 15 minutes. On some days, Nita wonders if she’s getting any work done at all. The last four hours consisted of two hours of meetings—one meeting to onboard a new team member and another meeting to discuss the sprint planning meeting later this evening. She tried to make some progress independently, but there wasn’t enough contiguous time or enough available information for her to start.

9:00 pm Abin got back from cricket coaching, showered, and warmed up dinner. He’s a good kid. Before heading to bed, he hugged Nita and kissed her goodnight. “Tomorrow will be an on-time start,” he promised. Nita’s been powering her day with coffee. The team is now in a sprint planning meeting with the folks in Boise and the client in Boulder. One more hour to motor through. Thank heavens for that coffee.

10:00 pm Aargh! She’s not done yet. The client asked for some changes to the proposed iteration plan. Nita now has to finalize wireframes for user stories that are kind of ready but not quite. The team huddle is in 15 minutes.

11:00 pm Finally, time to get some sleep. Uh-oh! Hold on. She’s got to put the milk bag out and double-check the milk order. Abin wasn’t as good a boy as she thought, either. He didn’t clean the kitchen before heading to sleep. Time to roll up her sleeves and clean up.

She gets the job done, changes, brushes, flosses, and gets into bed. Try as she might, she just can’t sleep. Too much caffeine in the system? Too much of that bright screen? Who knows? Her attention shifts to her phone. The team is discussing the change in sprint plans. Maybe she should tune in until she feels sleepy.
Phew! That day, unreal as it may be, is commonplace for many remote-working technologists operating across continents. Nita’s character is fictional, but I know plenty of people who live and work like her. Of course, work hours can differ depending on the time zones the team is distributed across. But context-switching and highly interrupted days are common regardless of time zones.

In 2020, the global pandemic sped up the shift to remote work. To be sure, that was a change in the right direction. In a time of powerful computers in our pockets, great tools for online collaboration, and reasonably good internet connectivity in most places, it was strange to make people go through hours of commuting just to be in a noisy office every day. People made considerable sacrifices in their personal lives to have a career. For those with a single-minded devotion to that career, it was fine—but there were others who were looking for balance in their lives. And so, remote work came with the promise of restoring that balance. It gave organizations a way to tap into talent pools they hadn’t considered before, simply because they didn’t have offices in that area.

If you can now work from anywhere, an organization can employ you from anywhere. Knowledge workers are always more in demand than the supply, so being able to widen the talent pool gives employers a way to scale despite the constraints.

The purpose of this book isn’t to extol remote work, though. There are plenty of other books to do just that. The question I like to ask is about Nita—is there a better way to work? The shift to remote work in 2020 was so abrupt that many organizations never had the time to consider if the same work practices that felt effective in an office are also relevant for remote work. As a result, you see many individuals, such as Nita, continue to make compromises to their work–life balance, their mental and physical health, and their ability to do deep, meaningful work. There’s got to be a better way to work.

This book articulates a new way of working, especially for software development teams. That way is to embrace asynchronous collaboration. “What is that?” you may ask. The best way to define it is to combine a few definitions I’ve derived from Catherine Tansey and Marcelo Lebre.⁹

⁹ Citations of facts and sources appear at the end of the book. A page number and a phrase identify the passage.
Asynchronous work is the practice of working on a team that doesn’t require multiple people to be online simultaneously. You do as much as you can with what you have, write things up clearly, transfer ownership of the work to whoever needs to pick it up next, and then work on something else.

If that sounds quite different from the way you collaborate with your team today, let me explain why you should consider making the shift to this way of working.

**COMPLEX PROBLEMS NEED SMARTER COLLABORATION**

When I started my career in IT, the problems we solved for clients differed greatly from the problems we solve today. For example, few clients expect us to build simple create, read, update, delete (CRUD) applications—low code platforms have disrupted that space. How about simple storefronts? Services like Shopify have disrupted that space by simplifying it for noncoders. With services such as managing large data centers, the hyper scalers such as Azure, GCP, and AWS have made it easy for in-house IT teams to manage infrastructure.

The work we do today is far more complex—we build platforms and data meshes and deep learning and neural networks. We modernize decades-old legacy systems to give incumbents a way to compete with digital native disruptors. We explore new ways to develop the human–machine experience. It’s impossible to run these kinds of projects with any of the following problems:

- An overdependence on meetings, conversations, and tribal knowledge
- A lack of good written communication
- No commitment to deep, uninterrupted work

Asynchronous collaboration comes with the promise to meet only when necessary, to make conversations productive, and to give people time so they can actually work.

**THE PROMISE OF FLEXIBILITY**

In April 2022, I surveyed more than 450 Indian employees in a leading software development unit of a global IT firm. I asked them what they dislike about their work. These were the top two responses from more than 50 percent of the respondents:

- Long contiguous hours
- Late hours
This is almost antithetical to the promise of remote work. Luke Thomas and Aisha Samake, in their book *The Anywhere Operating System*, outline this promise:

> When coworkers say, ‘I love working from home,’ it’s about when they work instead of only thinking about where they work from. That’s the secret hiding in plain sight!

Indeed, less than 5 percent of the people I surveyed recently want to go into an office every day. Sixty percent of the respondents preferred to work “regular, daylight hours,” i.e., nine to five. Twenty-eight percent wanted a flexible schedule where they could do their eight hours, their way. And these expectations are changing. This observation bears itself out in Future Forum’s research, which surveys a broader audience than my own:

> Knowledge workers who say they have little to no ability to set their own hours are 2.6x as likely to “definitely” look for a new job in the next year (compared to those with moderate schedule flexibility).

While the job market and economic conditions at any point may or may not afford such opportunities, it’s clear where people’s preferences are. Asynchronous work allows people the autonomy and flexibility they desire by allowing them to work during the hours that are most productive for them. More important, it lets managers and employers project a shared sense of trust and empowerment.

**ESCAPING THE SHALLOW**

As a product manager and an aspiring designer, I was always at sea in an open office. If the constant white noise wasn’t bad enough, the fact that I was “right there” implicitly meant that I was available to be interrupted. I got little joy out of my working day, especially when I needed to get work done for the team and needed time with my head down. I craved what Cal Newport terms *deep work*—the ability to concentrate without distraction on a cognitively demanding task.

When we started working remotely, though, I could cut out the white noise. Suddenly, I had my corner office. Except, of course, it was a room in my house. What didn’t change were the interruptions. The across-the-table interruptions now became video meetings and all-day back-and-forth messages on our instant messaging platform. Like Nita, I’ve struggled to get time to do deep work because my day would get cut up into little one-hour chunks. Paul Graham noted this eloquently in his 2009 essay “Maker’s Schedule, Manager’s Schedule”: 

When you’re operating on the maker’s schedule, meetings are a disaster. A single meeting can blow a whole afternoon, by breaking it into two pieces, each too small to do anything hard in.

Asynchronous collaboration advocates for “meetings as the last resort, not the first option.” If your current collaboration practices need loads of meetings, I understand this sounds radical. Asynchronous collaboration advocates for using communication tools thoughtfully. The sentiment I’d like you to empathize with is that interruptions have a cost—not just in terms of the time spent in the interruption but also as a fallout impact on productivity. Mihaly Czikszentmihaly, in his bestselling book *Flow: The Psychology of Optimal Experience*, speaks of a state that most of us want to get to at work:

... a state in which people are so involved in an activity that nothing else seems to matter; the experience is so enjoyable that people will continue to do it even at great cost, for the sheer sake of doing it.

Here’s the killer, though: while 97 percent of the 450-odd people I surveyed claimed to care about “flow,” only 12.5 percent—i.e., one in eight people—claimed to achieve it with regularity. Asynchronous work promises to give people a better chance at achieving flow.

The other benefit of deep work is the ability to have deep interactions with co-workers. In his landmark book *Thinking, Fast and Slow*, Daniel Kahneman outlines two systems of thinking. System 1 is an intuitive and fast way of thinking. You call on System 1 for the answer to your parents’ names, for the result of 2+2, and for how you navigate your own home. Anything that needs slightly deep thought—double-digit multiplication, for example—needs slower thinking. Kahneman calls that kind of thinking System 2. System 1, being as instinctive as it is, can help us decide at speed but is also prone to biases and errors. An effective approach to work should combine both System 1 and System 2 thinking.

A culture that defaults to meetings as a way of working lives in the realm of System 1. There’s barely enough time to slow down, go a few levels deep, and analyze things. You lionize responsiveness and presence, but deep work becomes the casualty. Asynchronous work prioritizes good analysis and thinking. When you do meet, people have given enough thought to a topic and are meeting for a specific time-boxed purpose that needs intense collaboration. You can now make the deep work you’ve done so far deeper.
A MORE INCLUSIVE WORKPLACE

The people I surveyed were in offshore development teams in India. Their unit served clients in North America. Many people were straddling time zones with up to a 13-hour time difference. You don’t need me to tell you that this is hard. And if something is hard for most people, it’s usually harder for historically under-represented groups such as women in IT. (Even after years of conversations about the topic, women’s representation in IT was just 33 percent in 2022. In technical roles, that goes down to 25 percent.)

For example, a tight schedule with loads of meetings isn’t easy for anyone, but it’s especially hard for women, who bear a disproportionate burden of household and childcare responsibilities in most societies. Now let’s imagine a workplace without arbitrary start and end times, where people work in schedules that make sense for them and when we don’t have the pressure to cram all communication into those specific eight hours. Getting rid of those constraints can help more under-represented groups find a way into the workplace. We can’t change society overnight. What we can change is the workplace—and that’s the promise of being asynchronous.

While we’re on the topic, consider the diverse personalities in the workplace. There are introverts who will rarely be the first to voice their opinions on a topic. There are people who are non-native English speakers who may be deep thinkers but lack the confidence to articulate themselves. Asynchronous communication allows them to have the space to share their thoughts and ideas at a pace convenient for them. Modern writing tools allow non-native English speakers to correct their spelling and grammar. Introverts now don’t have to be the loudest voice in the room. If diversity is being invited to the party and inclusion is being asked to dance, asynchronous work helps that inclusion.

DEFAULTING TO ACTION

In software engineering, there’s always a choice between waiting to be perfect and being wrong at speed. When everyone was in one office, sitting at one large table, people could ask for or offer help by just tapping someone on the shoulder and talking to them. Never mind the fact that the person who needed to help you might have lost their own flow. Still, you were one step closer to being perfect. Using the same approach in a remote setup creates many problems. It’s hard to keep track of who is free, who is busy, and who you’re interrupting when you’re giving each other a shout. Remote.com speaks of the concept of “defaulting to action” in such
a situation. Sure, interrupting someone can help get your work closer to perfection, but you can’t overlook the impact it has on the person who is unblocking you. Instead, you choose to be wrong at speed. It’s okay, if you have to backtrack at a later point, but it’s better than waiting to be unblocked.

_There are many times when work isn’t ready for us to tackle, tasks aren’t planned, decision makers aren’t online, etc. In these times, successful teams execute, even if they later have to refactor and adapt, they don’t waste time “waiting.”_

—Marcelo Lebre

Defaulting to action optimizes for speed and throughput, but it also encourages thoughtful communication. For example, a well-written requirement may preempt the need for a ceremonial kickoff, where a developer, product owner, and tester discuss its implementation and testing approach. A simple screen recording attached to such a requirement can allow a developer to continue development while giving the product owner and the tester a way to review the implementation, without interrupting what those two might be up to. Sure, the trio may meet up to iron out some sticking points, but the mantra of “meetings as the last resort, not the first option” works well here. If the developer does have to backtrack, it’s also feedback for the user story. The product owner can incorporate this learning into the next set of user stories they write. When in doubt, teams that adopt asynchronous communication just execute.

**IMPROVED KNOWLEDGE SHARING AND COMMUNICATION**

Asynchronous collaboration isn’t black magic. Getting time back to do deep work isn’t a freebie. In return, people must be clear communicators—mostly through writing.

Many of us find it daunting to write. I admit we all must get better at this skill, but for now let’s focus on the potential benefits. Daily writing brings with it many advantages, including the following ones:

- A regular record of project decisions that helps anyone understand your project’s current state effectively.
- Easier knowledge sharing and project ownership transfer.
- Reducing fear of missing out (FOMO) because everyone can now know what’s happening in any part of the project. There’s no need to attend a meeting to learn what happened in the last meeting!
As projects and organizations scale, tribal knowledge and a system of “he said, she said” don’t scale. Effective writing and thoughtful curation have a better chance at helping you scale, even as tenured people leave and new people join your team.

**A BETTER WORK ENVIRONMENT FOR ALL OF US**

Like most practices, asynchronous collaboration isn’t a silver bullet. It won’t solve all your problems. Regardless, it’s safe to assume that every team that does any creative work will benefit from embracing asynchronous practices. Figure 1.1 summarizes the benefits we’ve discussed in this chapter.

**Figure 1.1 Benefits of an async-first work culture.**
CHAPTER SUMMARY

The world of work has changed considerably since the start of the global pandemic, as many of our teams are now highly distributed and remote.

- Remote and distributed teams can experience burnout if they operate primarily in a synchronous manner.
- If you’re building software or doing knowledge work in today’s day and age, you limit your productivity if you depend too much on meetings and tribal knowledge. You need effective written communication and the ability to do deep work.
- The ability to work asynchronously, where people don’t have to be in the same physical or virtual space simultaneously, has several benefits:
  - Better work–life balance
  - Higher inclusion
  - Improved knowledge sharing
  - Communication practices that support scale
  - Time for “deep work”
  - A culture that defaults to action

While the global pandemic of 2020 was a time of great pain for many of us, it also fast-tracked us into a new set of possibilities for how we work together in teams. How far you embrace these possibilities will depend on your appetite for change and your team’s context.

Throughout the rest of this book, I’ll help you identify asynchronous work techniques for your team. But before you try to change anything, you must enlist your team’s support. So, in the next chapter, I’ll describe how you can introduce this async-first shift to your immediate colleagues.
A
accelerating leftward, 81–84
access control, 92
accountability partners, 161
actions, 134
defaulting to, 9–10
inputs, 129
techn huddles, 150–151
working independently, 46–47
ad hoc conversations, 152
advocacy, 265
agendas
meetings, 63
that need meetings, 200, 201
agents, change, 255, 256
agile approaches
to design, 173–178
to documentation, 181–186
agile manifestos, 33–34, 40–41, 87
agility, 17
agreeing on fundamentals, 280–281
aligning
with common purposes, 226–227
goals, 276–278
Allen, Tim, 250
analysis of tasks, 47
Antifragile (Taleb), 205
anti-patterns, 272
The Anywhere Operating System (Thomas/Samake), 61
app blockers, 45
applications, 6
architectural decision records (ADRs), 166–167, 176, 185
artifacts
make your own templates (MYOT), 176
onboarding, 189–190
writing, 189–190
artificial intelligence (AI), 158
assistance, 231
management, 237
assessing goals, 279. See also goals
assets, 91
assistance, artificial intelligence (AI), 231
assume positive intent (API), 105
async-first
behavioral cues, 85
definition of, 16
goals of, 258–259
leadership, 197 (see also leadership)
method stacks, 264
possibilities of, 289–295
retrospective processes, 134
schedules, 287–288
starting, 275–285
strategies to protect and extend, 260–261
team ownership, 284
asynchronous audio, 38–39
asynchronous collaboration, 5–6, 8, 10–11, 17
asynchronous communication, 18, 64–66, 142
asynchronous mode, desk checks/kickoffs, 138–144
asynchronous status reporting, 124
asynchronous video, 38–39
asyncc islands
advocacy, 265
communities of practice, 262–263
forces for/against change, 256–258
goals of async-first, 258–259
improving systems, 259–260
influencing departments to join, 262
organizational inertia, 255–264
recruiting vertical slices, 261
sharing defaults/templates/tools, 263–264
strategies to protect and extend async-first, 260–261
async work
benefits of, 11, 13–14
meetings and, 17
A.Team, 291
audio, asynchronous, 38–39
auditing systems, 232–234
audit trails, 163–172, 204
architectural decision records (ADRs), 166–167
business decision records (BDRs), 166
commit messages, 168–169
documentation, 171
   examples of, 171
   just ask patterns, 163–164
   meeting notes, 164–165
   pull requests (PRs), 169–170
   replacing summaries, 164
automation, 220
   code, 186
   documentation, 181
   onboarding, 193–194
   tools, 113–115
autonomy, 7, 49, 295
   hybrid work, 253
   in teams, 148–149
avoiding busywork, 203–204
A World Without Email (Newport), 271

B
back-and-forth communication, 50
backlogs, 91, 119–120
balance
   communication, 18
   systems in, 17–18
Basecamp, 113
baselines, tabulating baseline data, 278–280
baton-pass pairing routines, 157
behaviors
   accelerating leftward, 81–84
   async-first behavioral cues, 85
   encouraging productive, 82
   shifting, 77–85 (see also shifting behaviors)
   toxicity in virtual workplaces, 267–272
benefits
   of async work, 11, 13–14
   of pair programming tools, 156–157
   of remote work, 137 (see also remote work)
   of writing, 31–32
Bezos, Jeff, 55, 64
biases, 272–273
binary judgments, 14
blocking distractions, 43, 44–45
bonding, teams, 156
book clubs, 46
Boucher, Ryan, 169
brain drains, 247–248
brainstorming, 133
breakdown documents, 175–176
broadcasting commitments, 77–81
buddy systems, onboarding, 194
budgets for meetings, 74
building relationships, 66, 71–72, 74. See also communication; relationships
business decision records (BDRs), 166
business intelligence (BI) tools, 278
busywork, avoiding, 203–204
Butler, 113

calendars. See also schedules
   cleaning up, 281–282
   free, 152
   Google Calendar, 165
   syncing, 82
calm, creating, 221–223
   camaraderie
      cost of, 72–73
      on teams, 156
   Campêre, 247
   practicing, 213–216
capabilities of systems, 231–232
capped downsides, 123
celebrating hard workers, 271–272
change, 13
   agents, 255, 256
   focus on value, 13–14
   forces for/against, 256–258
   overcoming organizational inertia, 255–264
   processes, 221
   systems in balance, 17–18
   Ulysses pacts, 17
   visualization on spectrum, 14–16
channels, 105
   identifying, 55
   naming, 99
charts
   hill, 124
   responsible, accountable, consulted, informed (RACI), 89, 90, 91
chat status, changing, 78, 99
The Checklist Manifesto (Gawande), 139
checklists
   examples, 140
   onboarding, 191–192
   requirement templates and, 139
Choudhury, Prithwiraj (Raj), 248, 249, 290, 291
cleaning up calendars, 281–282
Clear, James, 77, 224
code
   automation, 186
collaboration, 204, 205, 218
   asynchronous, 5–6, 8, 10–11, 17
   audits, 232
   costs of, 70–71
courses, 147
documents, 25, 211
guidelines, 183
online, 5
protocols (see protocols [collaboration])
strategies, 6
synchronous, 30
tools, 24–27
whiteboards, 131
collecting inputs, 129, 130
commitments, broadcasting, 77–81
commit messages, 168–169
common purposes, aligning with, 226–227
communication, 10
  asynchronous, 18, 142
  back-and-forth, 50
  balance, 18
  email as secondary tool, 52–53
  emotion, 36
EqualExpert Slack guide, 105
face-to-face, 33, 69–75 (see also face-to-face communication)
functional design, 173–178
idea papers, 174–175
instant messaging (IM) (see instant messaging [IM])
knowledge sharing, 10–11
leadership, 202
onboarding, 190 (see also onboarding)
parameters of, 34
practicing candor, 213–216
  as a process, 204–205
  progress, 47
protocols, 55
of purpose, 170
reading skills, 45–46
simplifying complexity of, 178
teams, 164
tech design, 173–178
toxicity in virtual workplaces, 269–270
urgency of, 98–99
written communication versus documentation, 29
communities
  platforms, 236
  of practice, 262–263
  tacit knowledge, 229–238
competence, 259, 260
compliance, 256
complicated subsystem teams, 218
comprehension, 43
computers, 5
concentration, 7–8
Confluence, 165, 167
connectivity, internet, 5
conscious competence, 259
conscious incompetence, 259
consent-based decision-making, 54, 142
content
  instant messaging (IM), 102
  projects, 91–92
context
documentation, 182 (see also documentation)
  projects, 89–91
  switches, 81
  in team handbooks, 89
continuous flow, 118–122
Conventional Commits guide, 168
ConveRel quadrants, 61–63, 270
convergence, 62, 129
correspondence, 312. See also
  communication; instant messaging (IM)
ad hoc, 152
targeting, 102–103
threads, 104
conveyance, 62, 128
core hours, 157
core values, 224
costs
  budgets for meetings, 74
  of camaraderie, 72–73
  of collaboration, 70–71
  for stakeholders, 246–249
COVID-19 pandemic, 1, 5, 244, 289. See also remote work
create, read, update, delete (CRUD)
applications, 6
Cross, Rob, 148
cultures
  building team, 223–225
  culture-building, 216
defaulting to action, 9–10
  meetings, 8
  onboarding, 191 (see also onboarding)
  org culture, 258
  risk-averse, 148
cycles
  development, 117–126 (see also development cycles)
  shape up, 122–125
time, 120–121
two-track approach, 124–125
  use of demos, 122
Czikszentmihaly, Mihaly, 8

D
data, tracking, 280
data-driven retrospectives, 131
David Heinemeier Hansson (DHH), 72
decentralized models, 149
decision-making
  architectural decision records (ADRs), 166–167, 176, 185
  business decision records (BDRs), 166
cost-based, 142
  meetings, 64
  processes, 53–55
decisions, events, explanations, proposals (DEEP), 94, 95
deep work, 7, 8
planning focus time, 79
value of, 14
defaults
to action, 9–10
managing, 259
sharing, 263–264
stages of sensible, 275–284
defining development environments, 193
definition of done (DoD), 139–140
definition of ready (DoR), 139–140
defragging calendars, 281–282
delivery pressures, 257
demos, use of, 122
descriptions, projects, 184
design
agile approach to, 173–178
breakdown documents, 173–176
checklists, 191–192
functional, 173–178
idea papers, 174–175
inclusion, 204
make your own templates (MYOT), 176
mini design documents, 178
pair programming tools, 157
simplifying complexity of communication, 178
for success skills, 217–227
tech, 173–178
technical design documents, 176–178
desk checks, 137–145
definition of, 138
maintaining practice of, 144–145
remote native, 145
replacing with video, 142–143
details of user stories, 141–142
dev-box tests, 138
development cycles, 117–126
definition of ready (DoR), 139
sprints, 117–125
strength of processes, 125
DevEnv, 193
digital nomadism, 291–292
directly responsible individual (DRI), 149, 159
distraction blocking, 43, 44–45
distributed standups, 108–109. See also standup meetings
distributed teams, 149
distributed work arrangements, 1
distribution of teams, 245–246
documentation, 29
agile approach to, 181–186
agile manifesto, 87
audit trails, 171
automation, 181
code, 185–186
codebases, 184–185
handbooks, 181–186 (see also handbooks)
improving, 181
onboarding, 189–190
README files, 184–185
simplifying, 92
tools, 92, 165 (see also tools)
triggers, 94–95
“Write the Docs” community, 174
written communication versus, 29 (see also written communication)
documents
architectural decision records (ADRs), 176
breakdown, 175–176
collaboration, 211
GitLab, 88 (see also GitLab)
idea papers, 174–175
meetings, 64
mini design, 178
shared, 174
six-page memo patterns, 64
technical design, 176–178
versioning systems, 93, 94
done, definition of, 139–140
dopamine, 44
doubt, strategies for, 47
dumb questions, 190–191

E
editing
team handbooks, 93–94
text, 37
effective meetings, 151–152, 201. See also meetings
Effective Remote Work (Stanier), 15, 55, 137
efficiency, 70
of sprint ceremonies, 122
of teams, 193, 195
Eisenhower matrix, 111, 112
e-mail
as secondary tool, 52–53
signatures, 78
emojis, 101. See also communication
emotion, communication of, 36
enabling teams, 218
encouraging productive behaviors, 82
team social networks (ESNs), 235
environments
projects, 183
setting up retrospectives, 131–134
variables, 217
EqualExpert Slack guide, 105
events, retrospectives as processes, 128–130, 134–135
examples
of audit trails, 171
of checklists, 140
INDEX 325

of detailed user stories, 141–142
of onboarding checklists, 192
of schedules, 3–4
of survey forms, 130
expenses, operations, 247. See also costs
time zones, 69–70

F
face-to-face communication, 33, 69–75
building relationships, 74
cost of camaraderie, 72–73
cost of collaboration, 70–71
effectiveness of, 40
hybrid meetings, 71
onsite/offsite meetings, 73
value of being in person, 71–72
facilitating knowledge flows, 234–238
facilitation (team mentoring), 219
fear of missing out (FOMO), 10
feature breakdown documents, 175–176
feedback, 170, 175, 212
loops, 145
practicing candor, 213–216
Fellow, 113, 165, 211
files, README, 182, 184–185
first among equals (FaE), 149, 220
flexibility, 6–7, 289–290
of pair programming tools, 157
schedules, 7, 138
flex work, 248. See also hybrid work flow
audit trails, 163 (see also audit trails)
continuous, 118–122
facilitating, 234–238
Flow: The Psychology of Optimal Experience (Csikszentmihaly), 8
focus
building, 46 (see also distraction blocking)
systems, 203
transactions, 203
on value, 13–14
focus time, planning, 79
focus weeks, 83
forces for/against change, 256–258
change agents, 256
external forces, 257–258
internal forces, 256–257
formatting
architectural decision records (ADRs), 167
text, 101
forums as rituals, 128
fostering strong relationships, 195
four-day workweeks, 292–294
Fowler, Martin, 87
free calendars, 152
frequency of retrospectives, 127–130
frequently asked questions (FAQs), 175, 190–191
Fried, Jason, 50, 250, 251
functional design, 173–178
fundamentals, agreeing on, 280–281
G
Gawande, Atul, 139
Geekbot, 113
Gen Z, 294–295
GitHub, 167
GitLab, 46, 88, 115
Glean, 186
global pandemics. See COVID-19 pandemic
Google
Calendar, 165
Drive, 167
Jamboard, 131
Looker Studio, 278
Workspace, 165, 230, 231
Graham, Paul, 7, 82
Granovetter, Mark, 235
Grant, Adam, 148
guidelines, 105, 283, 284
H
habits, reading, 45–46
half days, 82
handbooks
agile approach to documentation, 181–186
cost, 89–92
documentation, 181–186
iterations of team, 93–94
need for source of truth, 87–88
onboarding, 190–191 (see also onboarding)
projects, 89, 182
shared realities, 95
start with teams, 88
transactions, 204
writing team, 87–96
Hansson, David Heinemeier, 250, 251
Hari, Johann, 44
Heath, Dan, 217
Henderson, John, 166
Heath, Dan, 217
helping each other, 212

hill charts, 124
hacking keyboards, 159
Howland, Mark, 293
huddles. See tech huddles
Hugo, 165
Huston, Cate, 191
hybrid meetings, 71
hybrid work, 243–254
  autonomy, 253
  brain drains, 247–248
  costs for stakeholders, 246–249
  embracing science, 249–251
  impact on the environment, 249
  inclusion, 248–249
  organizations, 244–245
  perception of asymmetry, 251
  preferences for remote work, 243–253
  team distribution, 245–246
  treating everyone as remote, 251–253
idea papers, 174–175
identifying channels, 55
IFTTT, 113
implementation of pull requests (PRs), 170
improving
  documentation, 181
  systems, 259–260
  written communication, 36
inclusion, 204–205
  benefits of writing, 31
  design, 204
  hybrid work, 248–249
  in workplaces, 9
incompetence, 239
indexing, 32
ineffective management, 258
inline videos, 39
in-person meetings, 71–72. See also face-to-face communication; meetings
in-person training, 245
inputs, 129, 130
installation, 184
instant messaging (IM), 53, 147
  content, 102
  EqualExpert Slack guide, 105
  guidelines, 105
  limiting messages, 100–101
  managing reactions, 103
  naming channels/rooms, 99
  productivity, 97–98
  statuses, 99–100
  targeting conversations, 102–103
  threads, 104
  visibility of guidelines, 105
integration, 170
interaction patterns, 218, 219
internal forces, 255, 256–257
internal structure of teams, 219–220
internet connectivity, 5
interpreting written communication, 34, 35
interruptions, 44, 80. See also distraction
  blocking
  quick syncs, 150
  rituals, 137
iterations of team handbooks, 93–94
J
Jira, 113
journalism, 35
just ask patterns, 163–164
K
Kahneman, Daniel, 8
kickoffs, 137–145
  definition of, 138
  maintaining practice of, 144–145
  meetings, 141
remote native, 145
knowledge
  capabilities of systems, 231–232
  forms of, 230
  management, 238
  managing ecosystems of, 237
  project handbooks, 91
sharing, 10–11, 87–88, 156
tacit, 229–239 (see also tacit knowledge)
work industry, 1
kudos, 134
L
layers of companies, supporting, 261
leadership, 197
  avoiding busywork, 203–204
  communication, 202
  culture-building, 216
  effective meetings, 201
  escaping recurring meetings, 200–201
  by example, 202–203
  inclusion, 204–205
  managing people, 209–216
  mindsets, 199–207
  misaligned, 257
  one-on-one interactions, 211–212
  practicing candor, 213–216
  reducing repetitive work, 201–202
  resiliency, 205–206
  success skills, 209–210
  time management, 206
  toxicity in virtual workplaces, 267–272
  tyranny of the way, 199–200
  vigilance of toxicity in virtual workplaces, 271–272
  workloads, 210–211
  limiting messages, 100–101
LinearB, 183
links, responding with, 95
locations
  face-to-face communication, 69–70
  meetings, 64–66
logs, 168
loops, feedback, 145

M
maintaining quality with fewer meetings, 138–145
maker weeks, 83
make your own templates (MYOT), 176
management. See also leadership
  artificial intelligence (AI), 237
defaults, 259
ineffective, 258
knowledge, 238
knowledge ecosystems, 237
managing people, 209–216
project management tools, 109–112
reactions, 103
stocks, 234–238
teams, 218–219
time, 45, 206
toxicity in virtual workplaces, 267–272
mandatory meetings, 109. See also standup meetings
Manifesto for Agile Software Development, 33–34, 40–41
Markdown, 185
Martin, Liam, 220
matrices, Eisenhower matrix, 111, 112
McMinn, Keavy, 170
meaningful synchrony, 270–271
measurements
  progress, 124
  scope, 124
  sprints as denominator of, 121
  throughput, 120–121
meetings
  agendas, 63
  async work and, 17
  best practices for, 143–144
  building relationships, 66, 74
  ConveRel quadrants, 61–63
  cost of camaraderie, 72–73
cultures, 8
decision-making processes, 64
documents, 64
dollars wasted because of, 60
effective, 151–152, 201
  face-to-face communication, 69–70 (see also face-to-face communication)
hybrid, 71
kickoffs, 141 (see also kickoffs)
as last resorts, 57–67, 77
length of, 63–64
maintaining quality with fewer, 138–145
mandatory, 109 (see also standup meetings)
minutes, 165
no-meeting Fridays, 82–83
notes, 164–165
onsite/offsite, 73
planning, 74
predictability of, 82
productivity and, 59–61
pursing recurring, 83–84
recurring, 200–201
retrospectives, 131 (see also retrospectives)
schedules, 82–83 (see also schedules)
standup, 107–116 (see also standup meetings)
summaries, 165
time zones, 64–66
value of being in person, 71–72
as waste of time, 30
and written communication, 30
mentoring, 156
merge requests (MRs), 169
messages. See also communication; email;
  instant messaging (IM)
  commit, 168–169
  content, 101
  limiting, 100–101
  schedules, 78–79
metrics, 279
cycle time, 121
tracking, 183
Microsoft
  Office 365, 165
  Teams, 113, 158
  Word, 37
mindsets
  change, 13 (see also change mindsets)
  leadership, 199–207
mini design documents, 178
Miro, 131
misaligned leadership, 257
mix pairing, 160–161
models, distributed teams, 149
momentum, gaining internal, 257
Montalenti, Andrew, 160
Morris, Kief, 169
Mullenweg, Matt, 105
Mural, 131, 133, 225
Mutton, Dan, 169, 170

N
naming channels/rooms, 99
Newport, Cal, 7, 44, 49, 50, 271
new ways to work
  benefits of async work, 11
  collaboration strategies, 6
  concentration, 7–8
defaulting to action, 9–10
INDEX

flexibility, 6–7
inclusive workplaces, 9
knowledge sharing, 10–11
software development, 5
Nikoo, Hoorvash, 168
no-meeting Fridays, 82–83
notes, 164–165. See also documents
Nygard, Michael, 166

○
off-boarding, 92
Office 365, 165
office spaces, 246
onboarding, 92, 189–196
automation, 193–194
buddy systems, 194
checklists, 191–192
fostering strong relationships, 195
frequently asked questions (FAQs), 190–191
reliability, 193–194
team efficiency, 195
team handbooks, 182–183
writing artifacts, 189–190
one-on-one interactions, 211–212
online collaboration, 5
online learning, 245
onsite/offsite meetings, 73
operations
expenses, 247
project handbooks, 91
optimizing systems, 120–121
organizational inertia, 255–264
organizations, hybrid work, 244–245
org culture, 258
Otter, 165
outcomes of retrospectives, 127–130
Outlook, 165
Ouzan, Raphael, 291
overheads, communication, 190

P
page owners, 94
pair programming tools, 155–161
accountability partners, 160
benefits of, 156–157
flexibility of, 157
Microsoft Teams, 158
mix pairing/solo work, 160–161
personal discipline, 158–159
viewpoints about, 155
Zoom, 138
Pais, Manuel, 99, 218
pandemics. See COVID-19 pandemic
Parabol, 53, 131
parameters of communication, 34
partner inertia, 257
passive pairing, 159
patterns
anti-patterns, 272
interaction, 218, 219
just ask, 163–164
reputation, 231
Peel, Abby, 268
performance, 81, 209–216, 236
Perpetual Guardian (New Zealand), 292, 293
personal discipline, 158–159
personal productivity, 47
personal well-being, 212
philosophizing, 159
physical standup meetings, 108. See also
standup meetings
Pinchas, Aviva, 53
pipelines, 91
pitfalls, navigating, 241
planning
focus time, 79
meetings, 74
scrum ceremonies, 118
platforms, 165, 236. See also tools
platform teams, 218
Platt, Lew, 230
pods, 149
pointless synchrony, 30
Polaris, 183
porous walls, creating, 230–234
possibilities of async-first, 289–295
autonomy, 295
digital nomadism, 291–292
flexibility, 289–290
four-day workweeks, 292–294
Gen Z, 294–295
skills economies, 290–291
practice
communities of, 262–263
definition of, 29
predictability of meetings, 82
preferences for remote work, 243–253
Present Shock: When Everything Happens Now (Rushkoff), 271
prioritization, 119–120
backlogs, 119, 122
inputs, 129
processes
change, 221
communication as, 204, 205 (see also communication)
onboarding (see onboarding)
retrospectives as, 128–130, 134–135
strength of, 123
teams, 144
two-track approach, 124–125
productivity, 6, 70, 204. See also
asynchronous collaboration
instant messaging (IM), 97–98
meetings and, 59–61
personal, 47
remote work, 249–250
  teams, 47
tech huddles, 151–152
  tools, 26
products, definition of, 29
  progress
  communicating, 47
  on expectations, 212
  measuring, 124
  team communication and, 52
  updating, 110
  projects
  content, 91–92
  context, 89–91
  environments, 183
  handbooks, 89, 182
  management tools, 109–112
  problems with, 6
  tracking metrics, 183
  protocols (collaboration), 49
  decision-making processes, 53–55
  email as secondary tool, 52–53
  identifying channels, 55
  response times, 55
  statuses/transitions (workflow), 50–52
  work execution, 49–50
  workflow, 49–50
  proximity bias, 272–273
  pull requests (PRs), 115, 169–170
  purging recurring meetings, 83–84
  purpose, communication of, 170
  Q
  quality, maintaining with fewer meetings, 138–145
  questions
    dumb, 190–191
    frequently asked questions (FAQs), 175, 190–191
    just ask patterns, 163–164
  quick syncs
    replacing, 80–81
    tech huddles, 150
  R
  radical candor, 214, 215
  Radical Candor (Scott), 213
  Range, 113
  rationing distractions, 45
  Razetti, Gustavo, 290
  reactions, management, 103
  reading, 43, 45–46
  README files, 182, 184–185
  ready, definition of, 139–140
  reasonable lag, defining, 280
  Rebele, Reb, 148
  recordings, 38–39
  records, 91. See also documents
    architectural decision records (ADRs), 166–167, 176, 185
    business decision records (BDRs), 166
    recruiting vertical slices, 261
    recurring meetings, 83–84, 200–201. See also meetings
    reducing pressure, 221–223
    reducing repetitive work, 201–202
    relationships, 62
      building, 66, 71–72
      face-to-face communication, 74
      fostering strong, 195
      strong, 62
      weak, 62
    relevance, maintaining, 115
    reliability, onboarding, 193–194
    remote.com, 9
    remote native, 145
  Remote: Office Not Required (Hansson/Fried), 250, 251
  Remote Team Interactions Workbook (Skelton/Pais), 99
  remote teams, 50
  remote work, 1
    desire for, 289–290
    organizations, 244–245
    preferences for, 243–253
    productivity, 249–250
    remote feature of, 137
    secret sauce, 250–251
    serendipity, 250
    shift to, 5
    statistics, 59, 60
    team distribution, 245–246
  repetitive work, reducing, 201–202
  replacing
    desk checks with video, 142–143
    quick syncs, 80–81
    summaries, 164
  reporting asynchronous status, 124
  reputation patterns, 231
  requirements, 138, 139
  resiliency, 205–206
  resources, written communication, 36
  responding with links, 95
  responsible, accountable, consulted, informed (RACI), 89, 90, 91
  retrospectives, 127
  convergence and, 129
  data-driven, 131
  frequency of, 127–130
  as processes, 128–130, 134–135
  scrum ceremonies, 135
  setting up environments, 131–134
  templates, 132
  timelines, 129
  voting asynchronously, 131
  risk-averse cultures, 148
rituals
forums, 128
interruptions, 137 (see also desk checks; kickoffs)
sprints, 117 (see also sprints)
roadmaps, 91
rooms, naming, 99
Rushkoff, Douglas, 271
S
Sack, Jonathon, 71
safety checks, 132–133
Samake, Aisha, 61, 62
Satir change curve, 221
schedules, 3–4, 287–288
flexibility, 7, 138
focus weeks, 83
half days, 82
maker weeks, 83
messages, 78–79
no-meeting Fridays, 82–83
pair programming tools, 157
retrospectives, 127–130
tech huddles, 152–153
scope
measuring, 124
quantifying change, 139
score calculators, 278
Scott, Kim, 213, 214, 215
scrum ceremonies, 118, 135
searches, 32
secrets, 91
secret sauce, remote work, 250–251
self-organizing teams, 49
self-sign-up, 47
sensible defaults. See defaults
serendipitous discovery, 231
serendipity, remote work, 250
setting up retrospective environments, 131–134
shape up cycles, 122–125
sharing
defaults, 263–264
development environments, 193
documents, 174
knowledge, 156
realities, 95
templates, 263–264
tools, 263–264
shifting behaviors, 77–85
accelerating leftward, 81–84
async-first behavioral cues, 85
broadcasting commitments, 77–81
signatures, email, 78
Silva, Gonçalo, 31–32
simplifying
complexity of communication, 178
documentation, 92
Singer, Ryan, 123
sizes of teams, 164
Skelton, Matthew, 99, 218
skills economies, 290–291
Slack, 105, 113
slowing things down, 31–32, 80
social batteries, 72
social media, dopamine and, 44
software development
Manifesto for Agile Software Development, 33–34
new ways to work, 5
surveys, 6, 7
workflow, 50
solo work, 160–161
SonarQube, 183
SPACE framework, 222
special economic zones (SEZs), 290
spectrum
of synchronousness, 15, 17
visualization of, 14–16
speed, 204
sprints
continuous flow, 118–122
development cycles, 117–125
scope, 119–120
stages of sensible defaults, 275–284. See also defaults
stage 1: aligning goals, 276–278
stage 2: tabulating baseline data, 278–280
stage 3: agreeing on fundamentals, 280–281
stage 4: cleaning up calendars, 281–282
stage 5: building 30-day programs, 283–284
stages of workflow, 52
stakeholders, costs for, 246–249
standup meetings, 107–116
automation tools, 113–115
distributed standups, 108–109
lag, 115–116
length of, 108
maintaining relevance, 115
project management tools, 109–112
Zoom with video off, 108
Stanier, James, 15, 16, 55, 137, 174
starting async-first, 275–285
statistics
readability, 37
remote work, 59, 60
#status channel, 113
statuses
asynchronous status reporting, 124
instant messaging (IM), 99–100
workflow, 50–52
StepSize, 186
stocks, managing, 234–238
stories, 138
details of, 140–141
queuing up in advance, 141–142
user, 138
strategies
  collaboration, 6
  for ConveRel quadrants, 61–63
decision-making processes, 53–55
  distraction blocking, 43, 44–45
doubt, 47
to protect and extend async-first, 260–261
reading, 45–46
shifting behaviors, 77–85
statuses/transition (workflow), 50–52
Ulysses pacts, 17
  working independently, 46–47
stream-aligned teams, 218
streamlining decision-making processes, 53–55
“The Strength of Weak Ties” (Granovetter), 235
strong relationships, 62
structure
  after the fact, 232
  of text, 32
success skills, 209–210, 217–228
  aligning with common purposes, 226–227
  building team cultures, 223–225
  creating calm, 221–223
design for, 217–227
  internal structure of teams, 219–220
  reducing pressure, 221–223
  team management, 218–219
summarize
  changes to handbooks, 93
  meetings, 165
  replacing, 164
supporting scale, 10–11
surveys
  maturity, 279
  software development, 6, 7
synchronous collaboration, 30
synchronous communication, 18
synchronousness, spectrum of, 15, 17
synthesizing inputs, 129
System 1, 8
systems
  audits, 232–234
  in balance, 17–18
capabilities of, 231–232
focus, 203
  improving, 259–260
  optimizing, 120–121
T
tabulating baseline data, 278–280
tacit knowledge, 229–239.
  See also knowledge
documentation, 237–238
capabilities of systems, 231–232
communities, 229–238
creating porous walls, 230–234
facilitating flows, 234–238
knowledge management, 238
Taleb, Nassim, 205
targeting conversations, 102–103
task boards, 52, 53, 144.
  See also communication
updating statuses, 110
whiteboards as, 284	
tasks
  analysis, 47
  project management tools, 109–112
  Team APIs, writing, 183, 262
team handbooks.
  See also handbooks
  content, 89–92
documentation triggers, 94–95
iterations of, 93–94
need for source of truth, 87–88
onboarding, 182–183
shared realities, 95
start with teams, 88
tools, 92
versioning systems, 93, 94
writing, 87–96

team
  autonomy in, 148–149
  bonding, 156
  broadcasting commitments, 77–81
  clarifying norms, 145
  collaboration, 230 (see also collaboration)
  communication, 164 (see also commitments)
  creating documentation, 30 (see also documentation)
culture-building, 216
cultures, 223–225
defining values, 223
distributed, 149
distribution, 245–246
efficiency, 193, 195
goals, 283
hubs, 26
identifying values, 224
internal structure of, 219–220
management, 218–219
managing, 209–216
meetings, 66 (see also meetings)
Microsoft Teams, 113
off-boarding, 92
onboarding, 92 (see also onboarding)
ownership, 284
processes, 144
productivity, 47
remote, 50
retrospectives, 127 (see also retrospectives)
self-organizing teams, 49
sizes of, 164
stages of sensible defaults (see stages of sensible defaults)
standup meetings, 107–116
success skills, 217–228 (see also success skills)
tech huddles, 147 (see also tech huddles)
templates for team values, 225
topologies, 219, 262
treating everyone as remote, 251–253
workflow, 51
tech design, communication, 173–178
technical design documents, 176–178
templates, 165
make your own templates (MYOT), 176
retrospectives, 132
sharing, 263–264
team values, 225
Team Values Discovery, 225
tests, dev-box, 138
text
changing, 32
communication, 30 (see also written communication)
formatting, 101
interacting with, 32
structure of, 32

Thinking Fast and Slow (Kahneman), 8
30-day programs, building, 283–284
37signals, 72
Thomas, Luke, 61, 62
thrashing, 53
throughput, 120–121
ties, types of, 236
time
cycles, 120–121
management, 45, 206
pair programming tools, 157
recruiring meetings, 83–84, 200, 201
response times, 55
timelines
retrospectives, 129
scrum ceremonies, 118
time zones
face-to-face communication, 69–70
meetings, 64–66

titles, 184
tldv, 165
tools, 23
automation, 113–115
business intelligence (BI), 278
code automation, 186
collaboration, 24–25
defining development environments, 193
documentation, 165
EqualExpert Slack guide, 105
feature breakdown documents, 175–176
gaps in capabilities of, 258
instant messaging, 53
maintaining, 92
Manifesto for Agile Software Development, 33–34, 40–41
Microsoft Word, 37
pair programming, 26, 155–161 (see also pair programming tools)
productivity, 26
project management, 109–112
README files, 185
sharing, 263–264
with specific context, 26–27
task boards, 51, 52
team handbooks, 92
text editors, 37
tracking metrics, 183
whiteboards, 131
topics, threads, 104
topologies, teams, 218–219, 262
toxicity in virtual workplaces, 267–272
communication, 269–270
growth of, 267–271
leadership vigilance of, 271–272
meaningful synchrony, 270–271
tracking
data, 280
metrics, 183
training
in-person, 245
online learning, 245
transactions, 203
two-track approach, 124–125
Ulysses pacts, 17, 77
unconscious competence, 260
unconscious incompetence, 259
updating statuses, 110
Upstream (Heath), 217
usage, 184
user stories, 138, 141–142
V

Vagrant, 193

values
  alignment, 223
  asynchronous communication, 18
  of being in person, 71–72
  core, 224
  of deep work, 14 (see also deep work)
  focus on, 13–14
  identifying, 224
  synchronous communication, 18
  Team Values Discovery, 225
  of tools, 23
  variables, environments, 217
  versioning systems, 93, 94
  vertical slices, recruiting, 261
  video
    asynchronous, 38–39
    desk checks, 142–143 (see also desk checks)
    inline, 39
    replacing desk checks with, 142–143
    Zoom with video off, 108
  videoconferencing, 70, 147. See also Zoom
  virtual workplaces, configuring, 227
  virtual workplaces, toxicity in, 267–272
  visibility bias, 272–273
  visualization on spectrum, 14–16
  visuals, overviews, 184
  Visual Studio, 193

W

weak relationships, 62
weak ties, 235
well-being, 212
whiteboards
  collaboration, 131
  as task boards, 284
  wiki pages, 174
work environments, improving, 11
work execution, 49–50
workflow, 49–50
  representing practices for, 144
  software development, 51
  statuses, 50–52
  transitions, 50–52
work-from-anywhere setups, 246. See also remote work
  working independently, 43, 46–47
  work-life balance, 5–6
  workloads, 210–211
workplaces
  inclusive, 9
  toxicity in virtual, 267–272
  Workspace (Google), 230, 231
“Write the Docs” community, 174
writing. See also communication
  architectural decision records (ADRs), 166–167
  artifacts, 189–190
  asynchronous audio, 38–39
  asynchronous video, 38–39
  benefits of writing, 31–32
  checklists, 191–192
  commit messages, 168–169
  content for team handbooks, 89–92
  versus documentation, 29
  effectiveness of face-to-face communication, 40
  improving, 36
  inline videos, 39
  interpreting, 34, 35
  iterations of team handbooks, 93–94
  journalism, 35
  Manifesto for Agile Software Development, 33–34, 40–41
  meetings and, 30
  need for source of truth, 87–88
  notes (meetings), 164–165
  reading skills, 45–46
  README files, 185
  responding with links, 95
  shared realities, 95
  start handbooks with teams, 88
  Team APIs, 183
  team handbooks, 87–96
  text editors, 37

X–Y–Z

x as service, 219
Yip, Jason, 107, 108
Zapier, 113
Zoom, 69. See also face-to-face communication
  pair programming tools, 158
  with video off, 108
Zoomers, 294–295