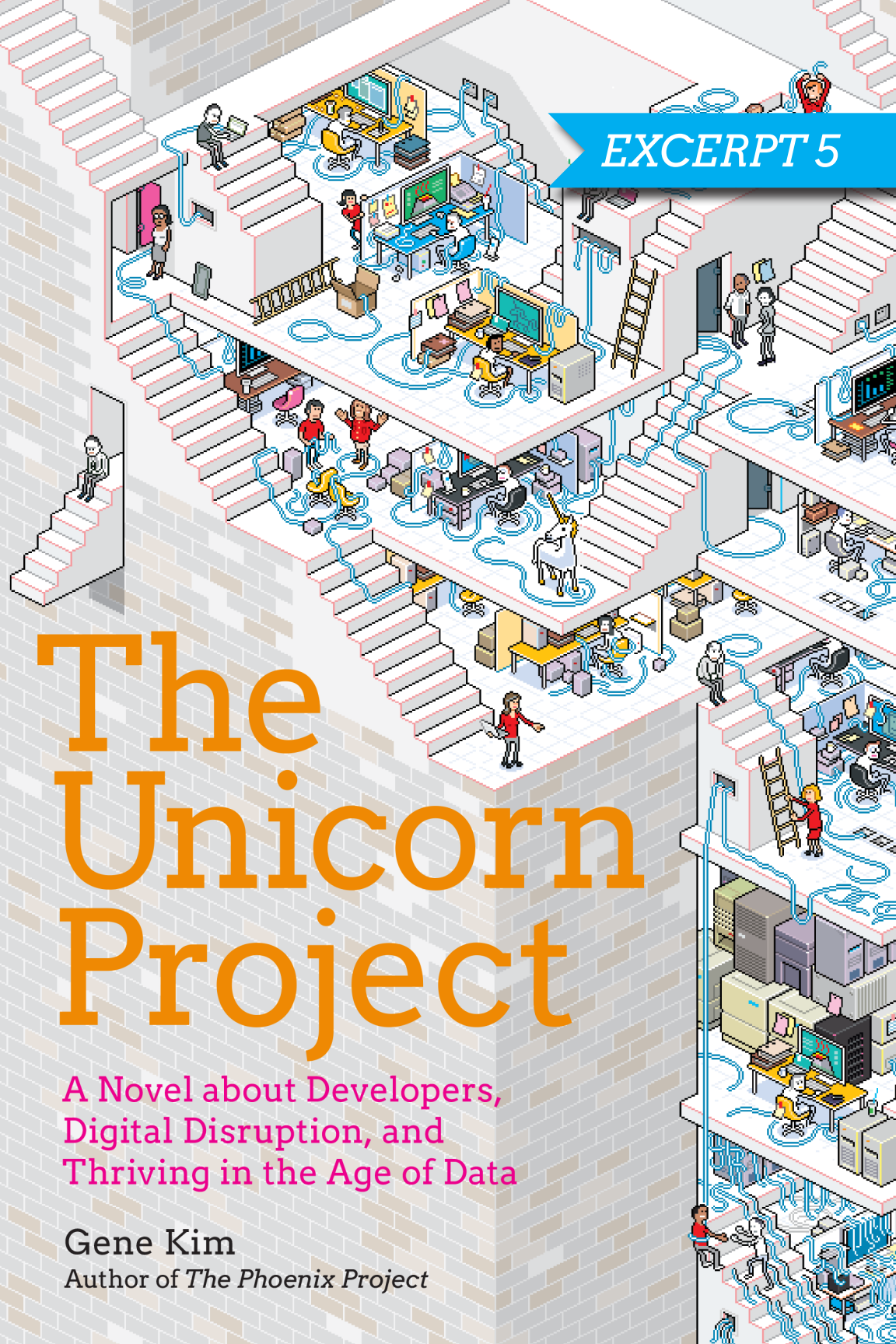


The Unicorn Project

A Novel about Developers, Digital Disruption, and Thriving in the Age of Data

Gene Kim
Author of *The Phoenix Project*

EXCERPT 5

The Unicorn Project

A Novel about Developers, Digital Disruption, and Thriving in the Age of Data

Gene Kim

IT Revolution
Portland, OR

IT REVOLUTION
25 NW 23rd Pl, Suite 6314
Portland, OR 97210

First Edition
Printed in the United States of America
24 23 22 21 20 19 1 2 3 4 5 6 7 8 9 10

Book design by Devon Smith Creative
Cover image by eBoy
Cover design by Joy Stauber, Stauber Brand Studio
Author Photograph by Anna Mayer Photography

Library of Congress Catalog-in-Publication Data
Available Upon Request

ISBN: 978-1942788768
eBook ISBN: 978-1942788775
Kindle ISBN: 978-1942788782
Web PDF ISBN: 978-1942788799

Publisher's note: Portions of this book were based on talks and articles by different thinkers and industry leaders with their permission and are included in the references of this book.

For information about special discounts for bulk purchases or for information on booking authors for an event, please visit our website at www.ITRevolution.com.

THE UNICORN PROJECT

Dedication

To my dad, Byung Kim (1937–2019), who really, really wanted me to get this book done.

To the loves of my life: my wife, Margueritte, and our three sons, Reid, Parker, and Grant, who also really, really wanted me to get this book done.

To the achievements of the DevOps Enterprise scenius, which this book is inspired by and celebrates.

Note to the Reader

The Unicorn Project takes place "in the present day," and is a companion novel to *The Phoenix Project* (which also takes place "in the present day"). The events from both novels take place concurrently, although certain situational elements of *The Unicorn Project* have been altered to account for changes in our industry.

While both books are about Parts Unlimited, *The Unicorn Project* was written to be a standalone book—there is absolutely no need to read or re-read *The Phoenix Project* first! (You may recognize some characters from *The Phoenix Project*—but then again, don't worry if you don't!)

Because the two books were written six years apart, there may be some suspension of disbelief required—for example, everyone's awareness of the Retail Apocalypse and the use of ride-sharing (Uber, Lyft) is much higher now than it was when *The Phoenix Project* was written.

For those who need some concrete waypoints, the characters who appeared in *The Phoenix Project* are indicated as such in the cast of characters, and there is a rough timeline of the two books provided as an endnote (beware, there may be spoilers!).

Parts Unlimited Employee Directory

REDSHIRTS

Maxine Chambers, Developer Lead, Architect
Kurt Reznick, QA Manager
"Cranky Dave" Brinkley, Developer Lead
Shannon Corman, Security Engineer
Adam Flynn, QA Engineer
Dwayne Cox, Lead Infrastructure Engineer
Purna Sathyaraj, QA and Release Manager
‣Brent Geller, Ops Lead

JUNIOR OFFICERS

Randy Keyes, Dev Manager
Rick Willis, QA Manager
‣William Mason, Director of QA
‣Wes Davis, Director of Distributed Technology Operations
‣Patty McKee, Director of IT Service Support

BRIDGE CREW

‣Steve Masters, CEO, Acting CIO
‣Dick Landry, CFO
‣Sarah Moulton, SVP of Retail Operations
‣Chris Allers, VP of Application Development
‣Kirsten Fingle, Director of Project Management
‣Maggie Lee, Senior Director of Retail Program Management
‣Bill Palmer, VP of IT Operations
‣John Pesche, Chief Information Security Officer (CISO)

STARFLEET COMMAND

Alan Perez, New Board Director, Operating Partner, Wayne-Yokohama Equity Partners
‣Bob Strauss, Lead Director, Former Chairman, Former CEO
‣Erik Reid, Prospective Board Director

‣Indicates characters who appear in *The Phoenix Project*.

PROLOGUE

• *Tuesday, September 2*

From: Steve Masters (CEO, Parts Unlimited)
To: All Parts Unlimited Employees
Cc: Dick Landry (CFO, Parts Unlimited),
Laura Beck (VP Human Resources)
Date: 11:50 p.m., September 2
Subject: Payroll Failure

To fellow employees of Parts Unlimited,

Early this morning, several thousand timecards were corrupted due to a technical failure, mostly affecting employees and contractors in our manufacturing facilities and retail stores.

My goal is to ensure that everyone gets paid as soon as possible. Anyone who was underpaid should get a check in the next twenty-four hours.

As CEO, my job is to ensure that we fulfill our obligations to our employees, who make the daily work of this organization possible. Without you, we would not be able to serve our customers, who depend on us to keep their cars running to conduct their daily lives.

I apologize to you and everyone who depends on you for the problems and inconveniences this payroll issue causes. I commit to you that we will provide all necessary help, including communicating with any bill collectors, banks, etc.

At the bottom of this email you will find a list of Frequently Asked Questions from HR and Business Operations. If you are not getting help quickly enough, please email me or call me on my office phone anytime.

In the meantime, our top priority is to understand what factors led to this failure, and we will do whatever it takes to make sure that it doesn't happen again.

Steve Masters,
CEO, Parts Unlimited

From: Chris Allers (VP Dev, Parts Unlimited)
To: All IT Employees
Cc: Bill Palmer (VP IT Ops), Steve Masters (CEO), Dick Landry (CFO, Parts Unlimited)
Date: 12:30 a.m., September 3
Subject: Corrective actions for the payroll failure

All—

Because of the high-profile nature of the payroll outage, we have conducted a thorough root-cause analysis. We have concluded that it was due to both human error and a technology failure. We have taken decisive actions to ensure that it will not happen again. The person responsible has been reassigned to a role where they can no longer affect production outcomes.

If you have any questions, please email me.

—Chris

Elkhart Grove Herald Times

Parts Unlimited Flubs Paychecks, Local Union Leader Calls Failure 'Unconscionable'

Automotive parts supplier Parts Unlimited has failed to issue correct paychecks to some of its hourly factory workers, and others haven't received any compensation for their work, according to a Parts Unlimited internal memo. The company denies that the issue is connected to cash flow problems and instead attributes the error to a payroll system failure.

The once high-flying $4 billion company has been plagued by flagging revenue and growing losses in recent quarters. These financial woes, which some blame on a failure of upper management, have led to rampant job insecurity among local workers struggling to support their families.

According to the memo, whatever the cause of the pay roll failure, employees might have to wait days or weeks to be compensated.

"This is just the latest in a long string of management execution missteps by the company in recent years," according to Nestor Meyers Chief Industry Analyst Kelly Lawrence.

Parts Unlimited CFO Dick Landry did not return phone calls from the Herald Times requesting comment on the payroll issue, accounting errors and questions of managerial competency.

In a statement issued on behalf of Parts Unlimited, Landry expressed regret at the "glitch," and vowed that the mistake would not be repeated. The Herald Times will continue to post updates as the story progresses.

PART ONE

September 3–September 18

CHAPTER 1

• *Wednesday, September 3*

"You're doing what?" Maxine blurts out, staring in disbelief at Chris, VP of R&D at Parts Unlimited.

Chris smiles weakly from behind his desk. *Even he realizes how absurd he sounds*, Maxine thinks.

"Maxine, I'm really sorry about this. I know it's a terrible way to come back from vacation, but this payroll outage created an incredible crap storm. The CEO and CFO wanted heads to roll. We agonized about this for days, but I think we came up with a pretty good solution . . . after all, no one is getting fired."

Maxine slaps the printed copy of his email onto his desk. "You say right here that it was caused by 'human error and a technology failure.' And now you say that I'm the 'human error'? After all that time we spent together deciding how to resolve that compliance finding, you're placing all the blame on me? What sort of bullshit is this?" She glares at him furiously.

"I know, I know . . . It's not right," Chris says, squirming under Maxine's intense gaze. "Everyone here values your incredible skills and talents and your fantastic contributions to the company over the last eight years—no one actually believes it was your fault. But the payroll issue was front-page news! Dick had to give a quote to keep the unions from filing a grievance! Given all that, I felt like we came up with the best solution in a pretty awful situation."

"So you blame the person who was on vacation because that person couldn't defend herself?" Maxine says in disgust. "That's really admirable, Chris. Which leadership book did you get that from?"

"Come on, Max, you know I'm your biggest fan and biggest defender. In fact, take this as a huge compliment—you have one of the most stellar reputations of anyone in IT," Chris says.

Blaming someone for a payroll outage is a strange way of appreciating someone, she thinks.

He continues, "*Everyone* knows that this isn't actually your fault. Just think of this as a vacation—you can work on anything you want, and you won't have any real responsibilities if you don't want."

Maxine is about to respond when she thinks about what she just heard. "Wait, treat exactly *what* like a vacation, Chris?"

"Uh…" Chris stammers, buckling under her stare. Maxine let's him squirm. As a woman in what remains a largely male dominated profession, she knows her directness might be contributing to Chris' discomfort, but she will always stand up for herself.

"…I promised Steve and Dick that I'd put you in a role where you couldn't make any production changes anymore," Chris says, squirming. "So, uh, effective immediately, you're moving from the manufacturing plant ERP systems to help with documentation for the Phoenix Project…"

"You're sending me to…" Maxine can't breathe. She can't believe what she's hearing.

"Look, Max, all you have to do is lie low for four months. Then you can come back and have your choice of any project you want to work on, okay?" Chris says. Smiling weakly, he adds, "See, like a vacation, right?"

"Oh, my God…" she says, finding her voice again. "You're sending me to the Phoenix Project?!" she nearly yells. Maxine immediately kicks herself for this brief moment of weakness. She takes a deep breath, adjusts her blazer, and pulls herself together.

"This is bullshit, Chris, and you know it!" she says right into his face, pointing her finger at him.

Maxine's mind races, thinking about what she knows about the Phoenix Project. None of it is good. For years, it's been the company death-march project, having ensnared hundreds of developers, achieving unprecedented levels of notoriety. Maxine is pretty sure that the reason nothing is going right is simply because they're not doing anything right.

Despite the Phoenix Project's obvious failures, it keeps going. With the rise of e-commerce and the decline of physical stores, everyone knows something has to be done to ensure that Parts Unlimited stays relevant in the increasingly digital age.

Parts Unlimited is still one of the largest players in the industry, with nearly a thousand stores across the nation. But there are times when Maxine wonders how the company will fare beyond its hundredth anniversary, which wasn't that long ago.

The Phoenix Project is supposed to be the solution, the shining hope that will lead the company into the future. It's now three years late (and counting) and $20 million has disappeared, with nothing to show for it except developer suffering. It stinks of impending failure, which will have grave implications for the company.

"You're going to take one of your best people and exile her to the Phoenix Project because you need a fall guy for the payroll outage?" Maxine says, her frustration boiling over. "This is not a compliment—this is the best way that you can say, 'Screw you, Maxine!' Hell, there's probably nothing in Phoenix that is even worth documenting! Unless it's to document incompetence? This is like labeling all the deck chairs on the *Titanic*. Have I said that this is bullshit already, Chris?"

"I'm sorry, Maxine," Chris says, throwing up his hands. "It's the best I could do for you. Like I said, no one is actually blaming you. Just do your time and it'll all go back to normal soon enough."

Maxine sits, closes her eyes, takes a deep breath, and steeples her hands in front of her, trying to think.

"Okay, okay…" she says. "You need a fall guy. I get it. I can take the blame for this whole fiasco. That's cool, that's cool…that's how business is done at times, right? No hard feelings. Just…put me to work in the cafeteria or in vendor management. I don't care. *Anywhere* but the Phoenix Project."

Listening to herself, Maxine's aware that in less than two minutes she's moved from denial to anger and is now in full-blown bargaining mode. She's pretty sure she's missed a step in the Kübler-Ross grief cycle, but at the moment she can't think of which one.

"Chris," she continues. "I have nothing against documentation. Everyone deserves good documentation. But there are tons of places that need documentation *way* more than Phoenix does. Let me go make a bigger impact somewhere else. Just give me an hour or two to come up with some ideas."

"Look, Maxine. I hired you eight years ago because of your amazing skills and experience. Everyone knows you enable teams to do the impossible with software," Chris says. "That's why I fought for you, and why you've led the software teams that are responsible for all our supply chains and internal manufacturing processes for all twenty-three manufacturing plants. I know how good you are…But, Maxine, I've done everything I can. Unfortunately, the decision has already been made. Just do your time,

don't rock the boat, and come back when everything blows over," he says, looking so remorseful that Maxine actually believes him.

"There are executives being shot left and right, and not just over this fiasco," Chris continues. "The board of directors just stripped Steve Masters of the chairmanship, so now he's just CEO. And both the CIO *and* VP of IT Operations were fired yesterday, no explanations given, so Steve is now acting CIO too. Absolutely *everyone* is worried that there is going to be even more blood in the streets…"

Chris looks to make sure the door is closed and, in a lower voice, says, "And there are rumors of potentially *even bigger and more sweeping changes coming*…"

Chris pauses, as if he might have said too much. He continues, "Look, whenever you're ready, go get yourself set up with Randy, the Phoenix development manager—he's a good guy. Like I said, think of this as a four-month vacation. Seriously, do whatever you think will be helpful. Heck, you don't need to do anything at all. Just keep your head down. Don't rock the boat. And whatever you do, just stay off Steve and Dick's radar. Sound good?"

Maxine squints at Chris as he name-drops Steve Masters and Dick Landry, the CEO and CFO of Parts Unlimited. She sees them every other month during the company Town Halls. How did she go from a two-week vacation seeing the wondrous sights of Kuala Lumpur to having Chris dump all this crap on her?

"Maxine, I'm serious. Just lie low, don't rock the boat, stay clear of outages, and everything will be fine, okay? Just thank your lucky stars you weren't fired for the payroll issue like the two other people were last year," Chris implores.

"Yeah, yeah. Don't rock the boat," she says, standing up. "See you in four months. And *you're welcome* for helping you keep your job. Super classy, Chris."

Chris is actually getting more spineless each year, she thinks, storming out of the room. She considers slamming the door, but instead closes it…decisively. She hears him say, "Please don't rock the boat, Maxine!"

When she's out of sight, she leans against the wall. Tears well up. Suddenly she remembers the missing step in the Kübler-Ross cycle after bargaining: depression.

Maxine slowly makes her way back to her desk. *Her old desk. Where she* used *to work.*

Maxine can't believe this is happening to her. Trying to counter all the negative self-talk flying through her head, she reminds herself of her qualifications. She knows that for the past twenty-five years, her job has been to bend technology to do her bidding—efficiently, effectively, precisely, with creativity and flair, and most importantly, competence.

She knows she has unmatched real-world experience building systems that run under adverse and even hostile environments. She possesses a fantastic intuition about which technologies are best suited to achieve the mission at hand. She is responsible, meticulous, and careful about her work, and she insists on the same level of excellence and diligence from everyone around her. *After all, dammit, I was one of the most sought after consultants at the top* Fortune *50 companies*, Maxine reminds herself.

Maxine stops mid-stride. Even though she is a stickler for details and doing things right, she has learned that mistakes and entropy are a fact of life. She's seen the corrosive effects that a culture of fear creates, where mistakes are routinely punished and scapegoats fired. Punishing failure and "shooting the messenger" only cause people to hide their mistakes, and eventually, all desire to innovate is completely extinguished.

During her consulting days, she could always tell, usually within hours, whether people were afraid to say what they really thought. It drove her crazy when people were careful about how they phrased things, speaking obliquely and going to extreme lengths to avoid using certain *forbidden* words. She hated those engagements and would do everything she could to convince the client to end the project, saving them time, money, and suffering.

She can't believe she's starting to see these red flags at Parts Unlimited.

Maxine thinks, *I expect leaders to buffer their people from all the political and bureaucratic insanity, not throw them into it*.

Only yesterday, she and her family were getting off of a nearly twenty-hour flight back from Kuala Lumpur. When she turned her phone on, it nearly melted from all the incoming messages. While Jake and her two kids went to find food in the airport, she finally got ahold of Chris.

He told her about the payroll failure and filled her in on the mayhem. She listened carefully, but her heart stopped when she heard Chris say "…and we discovered that all the Social Security numbers in the payroll database were corrupted."

She broke out in a cold sweat, her hands tingled, and there was ice in her blood. For what felt like a lifetime, she couldn't breathe. She knew. "It was the tokenization security application, right?"

She cursed loudly. Parents all around herded their young kids away from her on the airport concourse. She heard Chris say, "Yep. And there's going to be hell to pay. Get into the office as soon as you can."

Even now, she's still in awe of the scale of the carnage. Like all engineers, she secretly loves hearing disaster stories…as long as she doesn't have the starring role. "Stupid Chris," she mutters as she thinks about dusting off her résumé, untouched for eight years, and putting out feelers for any job openings.

By the time Maxine reaches her work area, whatever equanimity she had managed to muster is gone. She stops before she walks in. Her armpits are sweaty. She smells them to make sure she doesn't stink of the humiliation she feels. She knows she's being paranoid—she put on so much deodorant this morning her armpits were chalky-white. She was glad she did.

She walks into the work area. Everyone knows she is being reassigned but are trying not to let on. Glenn, who has been her manager for three years, comes up and squeezes her shoulder, a pained expression on his face. He says, "Don't worry, Maxine. You'll be back here before you know it. None of us are happy with the way things went down. A bunch of people wanted to throw you a big party, but I was pretty sure that you wouldn't have wanted to make a big scene," he says.

Maxine says, "Damned right. Thanks, Glenn."

"No problem," he says with a wry smile. "Let me know how I can help, okay?"

With a forced smile, she says, "Come on, it's not like I'm dying or being sent into outer space! I'll be closer to headquarters, which is where all the action is. I'll send updates to all you ignorant villagers who aren't good enough to be in the thick of things!"

"That's the spirit. We'll see you back here in four months if all goes well!" he says, giving her a playful jab. Maxine's brow furrows slightly at the "if all goes well" bit. That was news to her.

As Glenn heads to a meeting, Maxine goes to her desk to start packing up. She picks the most critical things she'll need during her exile: her

carefully configured laptop (she is very picky about keyboards and amount of RAM), pictures of her family, her tablet, and the USB and laptop chargers carefully selected and accumulated over the years, along with the big sign that hangs over them: "DO NOT TOUCH, under penalty of death!"

"Hi, Maxine! Why are you packing up?" she hears someone ask. Looking up, she sees Evelyn, their promising young computer science intern. Maxine recruited her. All summer long, Evelyn has dazzled everyone with how quickly she picks things up. *She'll have her pick of jobs when she graduates*, Maxine thinks. Which is why all summer long Maxine has been relentlessly selling Parts Unlimited as a great place to work and learn. Which she herself believed, until this morning. *Maybe this isn't such a great place to work after all*.

"I've been temporarily re-assigned to the Phoenix Project," Maxine says.

"Oh, wow," Evelyn says. "That's awful…I'm so sorry!"

You know you're in real trouble when even the intern feels sorry for you, Maxine thinks.

She leaves the building, carrying her plain cardboard box, alone. She feels like she's reporting to prison. *Which is basically what the Phoenix Project is*, she tells herself.

It's a four-mile drive to the corporate headquarter campus. While she drives, she thinks about the pros and cons of staying at the company. Pros: Her husband is a tenured professor, which is why they moved to Elkhart Grove in the first place. Her kids love their schools, friends, and activities.

She loves her work and all the challenges; she loves interacting with the countless and complex business processes that span the entire company—it requires an understanding of the business, incredible problem-solving skills, patience, and the political sophistication to work with sometimes Byzantine and occasionally incomprehensible processes that every large organization seems to have. And the pay and benefits are great.

Cons: The Phoenix Project. Working for Chris. And the feeling that the corporate culture is changing for the worse. *Like how I just got scapegoated for the payroll outage*, she thinks.

Looking around, she sees buildings designed to exude status and success. Parts Unlimited earned that level of prestige by being one of the largest employers in the state, with seven thousand employees. They have

stores in almost every state and millions of loyal customers, although every metric shows those numbers declining.

In the age of Uber and Lyft, the younger generation is more often choosing not to own cars at all, and if they do, they sure don't fix their cars themselves. It doesn't take a strategic genius to realize that the long-term prosperity of the organization requires something new and different.

As she drives deeper into the corporate campus, she can't find Building 5. When she circles around for a third time, she finally sees the sign to the parking lot. Her heart sinks. The building's a dump compared to the others. *It even looks like a prison*, she thinks.

Building 5 used to be a manufacturing plant, just like MRP-8, her "old" building. But where MRP-8 is obviously still the pride of the company, Building 5 is where they dump misbehaving IT people like her and throw away the key.

If the Phoenix Project is the most important and strategic project for the company, don't the teams who work on it deserve a better building? Maxine wonders. But then again, Maxine knows that in most organizations, corporate IT is rarely loved and is often parked in the least attractive properties.

Which is odd. At MRP-8, the ERP technology teams work side-by-side with the plant operations people. They're viewed as partners. They work together, eat together, complain together, and drink together.

On the other hand, corporate IT is usually viewed as ranks of nameless faces whom you call when there's something wrong with your laptop or when you can't print something.

Staring at Building 5, Maxine realizes that as bad as the reputation of the Phoenix Project is, the reality is probably much, much worse.

Everyone tells Maxine that one of her most endearing qualities is her relentless and never-ending optimism. She keeps telling herself that as she walks toward Building 5, carrying the cardboard box full of her belongings.

A bored security guard inspects her badge and recommends she take the elevator, but Maxine chooses to go up the stairs instead. She wishes she had a more cheerful bag to carry all her things in instead of lugging this dumb box around.

As she opens the door, her heart sinks. It's a vast cubicle farm with drab gray partitions separating each work area. The maze of cubicles reminds her of the old computer text game *Zork*—she's already lost in a series of twisty passages, all alike.

It's like all the color has been drained from the building, she thinks. Maxine is reminded of her parents' old color TV set when her brother had fiddled with the brightness, contrast, and color dials to make everything look a sickly gray and green.

On the other hand, Maxine is delighted to see that each desk has two massive LCD screens. She's in the right place—these are developers. The new monitors, open code editors, and the high percentage of people wearing headphones are dead giveaways.

The room is so quiet you could hear a pin drop. It's like a university library. *Or a tomb*, she thinks. It doesn't look like a vibrant space where people work together to solve problems. Creating software should be a collaborative and conversational endeavor—individuals need to interact with each other to create new knowledge and value for the customer.

In the silence, she looks around, feeling even worse about her fate.

"Do you know where I can find Randy?" she asks the person nearest her. He points to the opposite corner of the room without even taking off his headphones.

Walking through the hive of silent cubicles, Maxine sees whiteboards and people huddled in groups, speaking in hushed tones. Along one long wall are enormous Gantt charts easily four feet high and thirty feet across, assembled from what looks like more than forty sheets of paper taped together.

Alongside the Gantt charts are printouts of status reports featuring lots of green, yellow, and red boxes. Standing in front of the charts are people dressed in slacks and collared shirts. Their arms are crossed and they look concerned.

Maxine can almost feel the people mentally trying to compress the bars closer together so that they can hit all those promised dates. *Good luck*, she thinks.

As she walks to the opposite corner where she was told to find Randy, Maxine suddenly smells it: the unmistakable smell of people who have slept in the office. She knows this smell. It's the smell of long hours, inadequate ventilation, and desperation.

In technology, it's almost a cliché. When there's a need to deliver capabilities to the market quickly, to seize a market opportunity, or to catch up with the competition, long hours become endless hours, where it's easier to sleep under your desk than to go home only to come right back. Although long hours are sometimes glorified in popular culture, Maxine views them as a symptom of something going very wrong.

She wonders what is happening: Too many promises to the market? Bad engineering leadership? Bad product leadership? Too much technical debt? Not enough focus on architectures and platforms that enable developers to be productive?

Maxine notices that she is wildly overdressed. She looks down at the suit that she's worn to work for years, realizing that she sticks out like a sore thumb. In this building, T-shirts and shorts far outnumber the collared shirt crowd. And *no one* is wearing a jacket.

Tomorrow, I'm going to leave the jacket at home, she thinks.

She finds Randy in a corner cubicle, typing away and surrounded by huge stacks of paper. Randy is a redhead, wearing the management khaki-land uniform—a collared, striped white shirt and khaki pants. Maxine guesses he's in his late thirties, probably ten years younger than her. Judging from his low body fat, he probably runs every day. But he looks stressed in a way that no amount of running can take away.

He gives her a big smile, standing up to shake her hand. She puts her big cardboard box down and realizes how tired her arms are. As she shakes his hand, he says, "Chris told me about how you ended up here. I'm sorry to hear about all that. But trust me, your reputation precedes you, and we're so excited to have someone of your experience on this team. I know it's not the best use of your skills, but I'll take any help we can get. I think you can make a real difference here."

Maxine forces herself to smile because Randy seems nice enough, even earnest. "Happy to help, Randy. What do you need to get done?" she asks, trying to be equally earnest. She does want to be useful.

"I'm in charge of documentation and builds. In all honesty, things are a mess. We don't have a standard Dev environment that developers can use. It takes months for new developers to do builds on their laptops and

be fully productive. Even our build server is woefully under-documented," Randy says. "In fact, we've had some new contractors on site for weeks, and they can't even check in code yet. God knows what they've actually been doing. We're still paying them. To do nothing, basically."

Maxine grimaces. She hates the idea of paying expensive people to just sit around. And these are developers—it deeply offends her sensibilities when willing developers are prevented from contributing.

"Well, I'm happy to help out wherever I can," she says, surprised at how much she means it. After all, making developers more productive is always super important, even those working on the Phoenix Project in its fiery, meteoric descent.

"Here, I'll show you where we've got you set up," Randy says.

He leads her past more rows of cubicles, showing her an empty desk, a filing cabinet, and two large monitors connected to a laptop. It's plainer and smaller than she'd like, she thinks, but it's fine. Especially since she'll only be here for a few months. *One way or another, I'll be out of here soon,* Maxine thinks. *Either my prison term will end or I'll get another job somewhere else.*

"We got you a standard developer setup, just like any developer who starts at Parts Unlimited," he says, gesturing at the laptop. "You've got your email, network shares, and printers set up with your existing credentials. I'll send out an introduction email this afternoon. And I've assigned Josh to help you get everything set up."

"That's great," Maxine says, smiling. "I'll take a look at what you have in terms of Dev onboarding and maybe come up with some recommendations. I'd love to get a Phoenix build going on my laptop too."

"That would be great! Wow, I'm so excited, Maxine," Randy says. "I never get senior engineers to work on these problems. Any engineers I have that are any good are always poached away by other teams. They're lured away by feature work that customers see instead of working on boring infrastructure…Now, where is Josh?" he mutters, looking around. "There are so many contractors and consultants running around here that sometimes it's hard to find the actual employees."

Just then, a young kid carrying a laptop walks by and sits down at the desk next to them. "Sorry I'm late, Randy. I went to go check on last night's build failure. Some developer broke the build when they merged their changes in. I'm still looking into it."

"I'll help you in a second, Josh. In the meantime, meet Maxine Chambers," Randy gestures at Maxine.

Maxine does a double-take. He looks barely older than her daughter. In fact, they could be classmates at the same high school. Randy wasn't kidding when he said he had junior people on his team.

"Maxine is a senior engineer in the company, and she's been assigned to us for a couple of months. She's the lead architect for the MRP system. Can you show her what she needs to know to get productive around here?"

"Uh, hi, Ms. Chambers. Nice to meet you," he says, holding out his hand and looking puzzled. *He's probably wondering how he ended up being responsible for someone who could be his mom*, she thinks.

"Nice meeting you," she says, smiling. "Please, just call me Maxine," she adds, even though it usually irks her when her daughters' friends call her by her first name. But Josh is a work colleague, and she's glad to have a native guide who can show her around. *Even if he's not old enough to drive*, she jokes to herself.

"Okay, let me know if there's anything you need," Randy says. "Maxine, I'm looking forward to introducing you to the rest of the team. Our first staff meeting is next week."

Randy turns to Josh. "Tell me more about the build failures."

Maxine listens. All those stories about caveman technical practices in the Phoenix Project are actually true. She's learned over her entire career that when people can't get their builds going consistently, disaster is usually right around the corner.

She looks around at the entire floor. Over a hundred developers are typing away, working on their little piece of the system on their laptops. Without constant feedback from a centralized build, integration, and test system, they really have no idea what will happen when all their work is merged with everyone else's.

Josh spins his chair around to Maxine. "Mrs. Chambers, I've got to go show Randy something, but I just emailed you what we've got in terms of documentation for new developers—there are wiki pages where I've assembled all of the release notes we've written and the documentation from the development teams. There's also links to the stuff we know we need to write. Hopefully that will get you started?"

Maxine gives him a thumbs up. As they leave, she logs in with her new laptop and is able to get in and open her email, miraculously working the

first time. But before looking at what Josh sent, she pokes around to see what else is on her new laptop.

Immediately, she is mystified. She finds links to HR systems, network shares to company resources, links to the expense reporting system, payroll, timecard systems… She finds Microsoft Word and Excel and the rest of the Office suite.

She frowns. *This is fine for someone in finance*, she thinks, *but not a developer*. There are no developer tools or code editors or source control managers installed. Opening up a terminal window she confirms that there aren't any compilers, Docker, Git… nothing. Not even Visio or OmniGraffle!

Holy cow! What do they actually expect new developers to do? Read emails and write memos?

When you hire a plumber or a carpenter, you expect them to bring their own tools. But in a software organization with more than one developer, the entire team uses common tools to be productive. Apparently here on the Phoenix Project, the toolbox is empty.

She opens her email to see what Josh sent. It takes her to an internal wiki page, a tool many engineers use to collaborate on documentation. She tries to scroll up and down the wiki page, but the document is so short there isn't even a scroll bar.

She stares at the nearly empty screen for several long moments. *Screw you, Chris,* she thinks.

Driven by morbid curiosity, Maxine spends the next half hour digging. She clicks around and finds only a handful of documents. She reads PowerPoint slides with architecture diagrams, lots of meeting notes and Agile sprint retrospectives, and a three-year-old product management requirements document. She is excited when she finds tantalizing references to some test plans, but when she clicks on the links, she is prompted by an authentication screen asking for her login and password.

Apparently she needs access to the QA servers.

She opens a new note file on her laptop and types a note to herself to find someone who can give her access.

Giving up on documentation for the moment, she decides to find the source code repositories. *Developers write code, and code goes into source*

control repositories. There are developers working on Phoenix, ergo, there must be a Phoenix source code repo around here somewhere, she thinks.

To her surprise, despite almost ten minutes of searching, she can't find it. She adds to her notes:

> Find Phoenix source code repo.

She finds links to internal SharePoint documentation servers, which may have more clues, but she doesn't have accounts on those servers either.

She types another note:

> Get access to DEVP-101 SharePoint server.

For the next hour, so it goes—Search. Nothing. Search. Nothing. Search. Click. Authentication screen. Click. Authentication screen.

Each time, she adds more notes to her growing list:

> Get access to QA-103 SharePoint Server.

> Get access to PUL-QA-PHOENIX network share.

> Get access to PUL-DEV-PHOENIX network share.

She adds more notes and to-dos, accumulating a list of more user accounts that she needs, adding the QA wiki server, the performance engineering wiki server, the mobile app team wiki, and a bunch of other groups with acronyms she doesn't recognize.

She needs network credentials. She needs installers for all the tools that are mentioned. She needs license keys.

Maxine looks at her watch and is surprised to see that it's nearly one o'clock. She's achieved nothing in two hours except document thirty-two things she *needs*. And she still doesn't know where the development tools or the source code repositories are.

If the Phoenix development setup were a product, it would be the worst product ever.

And now she needs food. She looks around, and seeing the nearly empty floor, realizes that she missed the lunch rush.

It would have been nice if she had followed them, but she had been too engrossed in digging through the labyrinth of Phoenix docs. Now she doesn't know where people find food. She wonders if she should add that to her list too.

Right after "update and send out my résumé."

From: Alan Perez (Operating Partner, Wayne-Yokohama Equity Partners)
To: Steve Masters (CEO, Parts Unlimited)
Cc: Dick Landry (CFO, Parts Unlimited), Sarah Moulton (SVP of Retail Operations), Bob Strauss (Board Chair, Parts Unlimited)
Date: 6:07 a.m., September 4
Subject: Go Forward Options, January Board Session **CONFIDENTIAL**

Steve,

Good seeing you two days ago in Elkhart Grove. As a newly elected board director, I've been learning a lot and appreciate the time being invested by the management team to get me up to speed. I've been especially impressed with Dick and Sarah (CFO and SVP Marketing, respectively).

Although I'm new, it's clear that Parts Unlimited's failed efforts to increase shareholder value have raised questions of confidence and created need for action. We must work together to break the string of broken promises repeated quarter after quarter.

Given how essential software is to your plans, your decision to replace your CIO and VP of IT Operations seems proper—hopefully this will restore accountability and increase urgency in execution.

To reiterate my motivation for reviewing strategic options at the board level: revenue growth isn't the only way to reward shareholders—we've

put so much focus into forcing Parts Unlimited to become a "digital company" that I believe we've lost sight of low risk ways to unlock value, such as restructuring the company and divesting non-core, poor-performing assets. These are just two obvious ways to increase profitability, which increases shareholder value and provides working capital for transformation.

We need to quickly assemble options for the board to review and consider. Given how much time management is spending on the current strategy, the board chair asked me to work with a few key members of the executive team to generate options for the board to discuss. I will work with Dick and Sarah, given their tenure and breadth of experience at the company. We'll have bi-weekly calls to discuss and assess ideas, and we'll be ready to present strategic options to the entire board in January.

Our firm bought a significant interest in Parts Unlimited because we believe there's considerable shareholder value that can be unlocked here. I look forward to a productive working relationship and improved outcomes for Parts Unlimited that we can all be proud of.

Sincerely, Alan

CHAPTER 2

• *Friday, September 5th*

Maxine scans her to-do list, slowly shaking her head in frustration. It's been two days and she is determined to perform a Phoenix build on her laptop, like any new developer should be able to do. This has become her mission. But according to her list, there are over a hundred items that she's missing, and no one seems to know where to find them.

She has done nothing on her list. Except for updating and sending out her résumé. Many friends replied to her right away, promising to look for positions she might be interested in.

Maxine asked her guide, Josh, about all those missing build items, but he didn't know anything about them. The build team used to know these things, but the details are either out-of-date or missing entirely, the knowledge scattered across the entire organization.

She is frustrated, every turn she takes leads to a dead end. There is nothing fun about this challenge. What she is doing, she's pretty sure, is the exact opposite of fun.

She's an engineer at heart, and she loves challenges and solving problems. She's been exiled smack in the middle of probably the most important project in the entire company's history. And somewhere, there's code—almost certainly millions of lines of code written by hundreds of developers over nearly three years. But she can't find any of it.

Maxine loves coding and she's awesome at it. But she knows that there's something even more important than code: the systems that enable developers to be productive, so that they can write high-quality code quickly and safely, freeing themselves from all the things that prevent them from solving important business problems.

Which seems to be completely missing here. Maxine is one of the best in the game, but after four days she still has almost nothing to show for it. Just endless clicking around, reading documents, opening tickets, scheduling meetings with people to get things she needs, trapped in the worst scavenger hunt ever.

For a moment, Maxine wonders if she's the only person having this problem. But she sees developers all around her struggling, so she quickly pushes away any feelings of self-doubt.

Maxine *knows* her kung fu is amazing. Many times in her career she's had to solve problems that seemed hopeless and impossible. Often in the middle of the night. Sometimes without any documentation or source code. One of her most famous escapades is still known as the "Maxine Post-Holiday Save," where all the in-store systems that handled refunds crashed spectacularly on the Friday after Christmas. It's one of the busiest shopping days of the year as people come in to return gifts from loved ones so they can buy something they actually want.

With her team, Maxine worked into the wee hours of Saturday morning to fix a multi-threading deadlock in a database vendor's ODBC driver. She had to manually disassemble the vendor library and then generate a binary patch. By hand.

Everyone said it couldn't be done. But she pulled it off, to the amazement of the scores of people who had worked that outage for over seven hours. The database vendor professional services team was in awe and immediately offered her a job, which she politely declined.

The legends about her kept growing after that. She's classically trained as a developer, and in her career, she's written software to stitch together panoramic graphic images and chip layout algorithms for CAD/CAM applications, back-end servers for massive multi-user games, and, most recently, the ordering, replenishment, and scheduling processes that orchestrate thousands of suppliers into a plant production schedule for their MRP systems.

She routinely lives in the world of NP-complete problems that are so difficult to solve they can take more than polynomial time to complete. She loves the *Papers We Love* series, revisiting her favorite academic papers from mathematics and computer science.

But she has never seen her job as just writing application code, working only pre-deployment. In production, when theory meets reality, she's fixed wildly misbehaving middleware servers, overloaded message buses, intermittent failures in RAID disk arrays, and core switches that somehow kept flipping into half-duplex mode.

She's fixed technology components that were spilling out their guts in the middle of the night, having filled up every disk and log server, making

it impossible for teams to understand what was actually happening. She led the effort to systematically isolate, diagnose, and restore those services based on decades of intuition and countless production battles.

She's deciphered stack traces on application servers that were literally on fire, racing to get them safely backed up before the flooding water, halon extinguishers, and emergency power shutdowns destroyed everything.

But deep down, she's a developer. She's a developer who loves functional programming because she knows that pure functions and composability are better tools to think with. She eschews imperative programming in favor of declarative modes of thinking. She despises and has a healthy fear of state mutation and non-referential transparency. She favors the lambda calculus over Turing machines because of their mathematical purity. She loves LISPs because she loves her code as data and vice versa.

But hers is not merely a theoretical vocation—she loves nothing more than getting her hands dirty, creating business value where none thought it could be extracted, applying the strangler pattern to dismantle decades-old code monoliths and replacing them safely, confidently, and brilliantly.

She is still the only person who knows every keyboard shortcut from *vi* to the latest, greatest editors. But she is never ashamed to tell anyone that she still needs to look up nearly every command line option for Git—because Git can be scary and hard! What other tool uses SHA-1 hashes as part of its UI?

And yet, as awesome as she is, at the height of her soaring powers and skills honed over decades, here she sits in the middle of the Phoenix Project unable do a Phoenix build, even after two days. She found where two of the four source code repositories are, and she's found the three installers for some of the proprietary source code management (SCM) tools and compilers.

However, she is still waiting for license keys for the SCM, and she doesn't know who to ask to get license keys for the two other build tools. She needs credentials for three network shares and five SharePoints, and no one knows where to get the ten mysterious configuration files mentioned in the documentation. When she emailed the person who wrote the docs, it bounced. They had long left the organization.

She is stuck. No one responds quickly to her emails, her tickets, or her voicemails. She's asked Randy to help, to escalate her requests, but everyone says it will take a couple of days because they're so busy.

Of course, Maxine never just takes "no" for an answer. She has made it her mission to do whatever it takes to get a build running. She's hunted down almost all of the people who have promised her something. She's found out where they sit and has pestered them, even camping out at their desks, willing to stay there until they get her what she needs.

Sometimes she got what she needed: a URL, a SharePoint document, a license key, a configuration file. But more often than not, the person she hunted down didn't have what she needed—they would have to ask somebody else, so they would open up a ticket on Maxine's behalf. And now they were both waiting.

Sometimes, they had a promising lead or clue on who or where Maxine needed to go to next. Most times, though, it was just a dead end, and she was right back where she started.

Trying to get a Phoenix build going is like playing *Legend of Zelda,* if it were written by a sadist, forcing her to adventure far and wide to find hidden keys scattered across the kingdom and given only measly clues from uncaring NPCs. But when you finally finish the level, you can't actually play the next level—you have to mail paper coupons to the manufacturer and wait weeks to get the activation codes.

If this really were a video game, Maxine would have already quit, because this game sucks. But Phoenix isn't a game—Phoenix is important, and Maxine never quits or abandons important things.

Maxine sits at her desk looking at the calendar she's printed out and pinned to the wall.

She turns back to her computer and runs her finger down her ever-growing list of to-dos again—each item a dependency she requires to get her build going.

She just added two more SharePoint credentials she needs to get from two different Dev managers who, for some reason, run their own Active Directory domains. They're rumored to contain some critical build documentation with some of the information she seeks.

Randy sent her a ton of Word docs, Visio diagrams, and marketing PowerPoint presentations, which she quickly skims for clues. They may be helpful to marketing people and architects, she supposes, but she's an

engineer. She doesn't want to see brochures for the car they've promised to build—she wants to see the engineering plans and the actual parts that they're going to assemble the car from.

These documents might be useful to someone, so she posted them on the wiki. Moments later, someone she doesn't know asks her to take them down because they might contain confidential information.

Looking further down her to-do list, she reads:

> Find someone who can give me access to Dev or Test environments.

These were referenced in some of the documentation she read yesterday, but she has no idea who to ask to get access.

She has crossed off one item:

> ~~Get account for integration test environment.~~

This was less satisfying than she had hoped. She poked around the environment for two hours, trying to gain an understanding of the giant application. But in the end, she found it too bewildering—it was like trying to picture the layout of an enormous building by crawling around air ducts without a map or a flashlight.

She types out a new to-do:

> Find someone who's actually doing integration testing so I can shoulder-surf while they work.

Watching someone use the Phoenix applications might help orient her. She's baffled that no one knows of an actual person who uses Phoenix. *Just who are they building all this code for?*

Scanning her to-do list again, she confirms that she in fact has nothing to actually do—she has already pestered everybody today and now she's just waiting for people to (not) get back to her.

It's Friday, 1:32 p.m. Four and a half hours to go until five, when she can finally leave the building. She tries hard not to sigh again.

She looks at her to-do list. She looks at the clock.

She looks at her nails, thinking that she needs a manicure.

She gets up from her desk with her coffee mug and walks to the kitchen,

passing by groups of people wearing hoodies, huddled together, talking in hushed urgency. Just to have something to do, she pours herself another cup of coffee. She looks down at her mug and realizes that she's already had five cups today, to satisfy the need to be *doing something*. She pours her coffee down the drain.

Along with her ever expanding to-do list, Maxine has kept a daily work diary on her personal laptop for the last decade. In it, she tracks everything she's worked on, how much time she spent on it, any interesting lessons she learned from it, and a list of things to never do again (most recently, "Don't waste time trying to escape spaces in file names in Makefiles—it's too difficult. Use anything else instead.").

She stares at her huge to-do list and her recent work diary entries in disbelief. She has never met a system she couldn't beat. *Is it possible that the utterly mediocre Phoenix Project, totally incapable of anything, is actually defeating me? Over my dead body,* she vows silently, then turns back to her work diary entries.

WEDNESDAY

4 p.m.: Waited around for Josh this afternoon who was supposed to walk me through his setup so I can replicate it. He's dealing with more nightly build problems.

I have a ticket to get access to build server, but was told by security that I need authorization from my manager. Sent email to Randy.

I'm reading every developer design doc I can find, but they're all starting to look the same to me. I want to see the source code, not read design docs.

4:30 p.m.: In one of the design docs I found the most succinct description of Phoenix: "Project Phoenix will close close the gap with the competition, allowing our customers to do the same things online that they can do in our 900 stores. We will finally have a single view of our customers, so in-store employees can see their preferences and order history, and enable more effective cross-channel promotion."

The scope of Phoenix is a little frightening. It needs to talk to hundreds of other applications across the enterprise. What could go wrong?

5 p.m.: Calling it a day. Chris stopped by, reminding me to not rock the boat or call too much attention to myself. And to not deploy anything into production.

Yeah, yeah. Sheesh. I can't even get a build going or log into network shares. How would I be able to push anything into production?

Bored out of my skull. Going home to play with new puppy.

THURSDAY

9:30 a.m.: Yes! They've given me accounts on a couple more wikis. I'm eager to dig in. This is progress, right?

10 a.m.: Seriously? This is it? I found some QA docs, but this can't be all of it, right? Where are all the test plans? Where are the automated test scripts?

12 p.m.: Okay, I met William, the QA director. Seems like a nice guy. We were able to meet just long enough to get me user accounts for their network share. Millions of Word docs filled with manual test plans.

I emailed William asking whether I can meet some of his test team. How do they execute all these tests? Seems like they need a small army. And where do they put the test results? I'm on his calendar. In two weeks. Madness.

3 p.m.: I found out where the big daily project stand-up is held: it's 8 a.m. by the whiteboards. I missed it today, but I won't miss it tomorrow.

5 p.m.: I've gotten so little done in two days. Everything I try to do requires an email, a ticket, or trying to find someone. I'm now resorting to asking them out for coffee. Maybe I'll get more responses.

FRIDAY

10 a.m.: The "15-minute stand-up meeting" went for almost 90 minutes because of all the emergencies. I don't know how I missed this meeting yesterday—seems hard to miss because of all the yelling. Wow.

OMG. Almost no one else can build Phoenix on their laptops, either. They're supposed to deploy this into production in TWO WEEKS! (No one is worried. Crazy. They think it will be delayed again.)

If I were in their shoes, I'd be losing my shit. Oh, well.

2 p.m.: I found a bunch of contractor developers brought in two months ago. They can't do builds, either. Shocking. I took them out to lunch. What a disappointment. They know even less than I do. At least the salad was okay.

I shared everything I know with them, which they were extremely grateful for. Always good to give more than you get—you never know who can help you in the future. Networking matters.

Note to self: I must curb my coffee intake. Must have drunk 7 cups yesterday. This is not good—I think I'm getting heart palpitations.

At 4:45, Maxine packs up her things. There's no chance anyone will get her anything this late on a Friday.

She almost makes it to the staircase when she runs into Randy.

"Hi, Maxine. Bummer we couldn't get further on the Dev environment. I escalated a bunch of issues and will make some phone calls before I leave today."

Maxine shrugs. "Thanks. I hope that lights a fire under some people."

"Whatever it takes, right?" Randy smiles. "Uh, I have one more thing that I need?"

Uh oh, thinks Maxine. But she says, "Sure, what's up?"

"Umm. Everyone on the Phoenix Project is required to submit timecards," Randy says. "We've got to show utilization levels or the project management people take our people away. I sent you the link to our

time-carding system. Could you fill it out before you leave? It should only take a couple of minutes."

He looks both ways before whispering, "I especially need *your* hours because when it comes to budgeting next year, it will help me backfill your position."

"No problem at all, Randy. I'll take care of it right now before I leave," Maxine says agreeably, but she is not happy. She understands the budgeting games that need to be played, but that's not what bothers her. Instead, it's that she already knows exactly what she worked on all week, because she keeps such meticulous notes. Absolutely zero real outcomes accomplished. Zilch. Nothing. Nada.

Back at her desk, she logs into the time-carding system. By her name are hundreds of project codes. They're not the project names. Instead, they're project codes that all look like airline reservation numbers—ten characters long with capital letters.

Looking at Randy's email, she copies the project code he gave her (PPX423-94-10) into the field and then dutifully puts in eight hours into each field from Wednesday to Friday and hits submit. She frowns. It won't let her submit until she describes what she did each day.

Maxine groans. She writes something for each day, basically saying some variant of "Working on Phoenix builds but waiting for entire universe to get me something." She spends five minutes modifying the text so that each entry is sufficiently different from each other.

It felt bad enough to be sitting on her butt getting so little done this week, despite her very best efforts, but it feels far worse to have to lie about it in writing.

Over the weekend, Maxine continually scans her phone for updates on her tickets, but only sees them being transferred from one person to another. When her husband, Jake, asks her why she is brooding, she refuses to admit that it's because of the timecard she filled out—it was like rubbing salt in the wound of non-productiveness. She allows herself to be distracted by their new puppy, Waffles, and thoroughly enjoys seeing her kids play with him.

By Monday morning, Maxine has successfully convinced herself to be cheerful, upbeat, and optimistic as she files into the big auditorium

for the Town Hall that the CEO of the company, Steve Masters, hosts every other month. She's enjoyed attending these ever since she joined the company. Her first one made a big impression on her because it was the first time she had seen executives directly address an entire company, taking questions from any of the nearly seven thousand employees.

Steve usually presents with Dick, the CFO. About a year ago, Steve also started co-presenting with Sarah Moulton, the SVP of retail operations. She has profit-and-loss responsibility for retail, the second largest business unit generating over $700 million in revenue per year. While Steve and Dick exude a certain amount of trust and authenticity, Sarah seems less trustable and believable. Last year, she had a different pitch every Town Hall, promising a totally different transformation then what had been presented before, causing lots of confusion, organizational whiplash, and eventually ridicule.

Maxine sees Steve preparing off stage, writing last-minute notes on a folded sheet of paper. Someone hands him a microphone, and he walks onto the stage to polite applause. "Good morning, everyone. Thank you so much for joining us today. This is sixty-sixth Town Hall that I've had the privilege of hosting.

"As you know, for almost a century, our mission has been to help our hard-working customers keep their cars running so that they can conduct their daily lives. For most of our customers, that means driving to work so they can collect a paycheck, take their kids to school, and take care of their loved ones. Parts Unlimited helps our customers in many ways. We are one of the world's most admired manufacturing organizations, making the high-quality and affordable parts that our customers need to keep their cars running. We also have over seven thousand world-class employees directly helping our customers in nearly one thousand stores around this great nation. We are often the only thing that keeps their cars out of expensive service stations."

Maxine has heard this almost fifty times from Steve at these sessions—it's obviously important to him to remind everyone who their customers really are. When things go wrong with Maxine's car, she usually takes it to her car dealership because it's still under warranty. But the vast majority of their customers don't have that luxury. Their cars are older, sometimes older than her kids—in fact, their customers may be driving the same make, model, and year of the car she drove as a teenager. They

often have little discretionary income. When something goes wrong with their car, it can wipe out whatever savings they have (if any). And when their car is at a repair shop, they not only deplete their savings but they also can't drive to work. And that means they can't provide for their families.

Maxine appreciates these reminders about their customers—when engineers think of "the customer" in the abstract instead of as a real person, you rarely get the right outcomes.

Steve continues, "For almost a century, that mission has remained unchanged, although the business environment certainly has. On the manufacturing side, we now have fierce overseas competitors that undercut our prices. On the retail side, our competitors have opened up thousands of stores in the very same markets we serve.

"We are also in a time of incredible economic disruption. Amazon and the other e-commerce giants are reshaping our economy. Some of the most famous retailers that many of us grew up with are going out of business, such as Toys"R"Us, Blockbuster, and Borders. Just down the road here from corporate headquarters, many of us drive past that space where the old Blockbuster used to be, and that space has remained vacant for over a decade.

"We are not immune to this. Our same-store sales are continuing to decline. Many of our customers would rather order their windshield wipers from their phone from someone else than go into one of our physical stores and talk to someone.

"But I believe people don't want just automotive parts; they want help from people they trust. And that's why our store associates are so important. That's why we invest so much in training. And the Phoenix Project will allow us to bring that expertise and trust to our customers in the channels they want to use, whether it's our physical stores or online.

"Sarah will talk about the progress of the Phoenix Project later, and how it supports the three metrics I care most about: employee engagement, customer satisfaction, and cash flow. If all our employees are excited to come to work each day, and if we're delighting our customers through constant innovation and great service, cash flow will take care of itself.

"But before we go through our top annual goals, let me first talk about something that is probably on all your minds," Steve pauses for

several moments. "Recently, I sent out an email that Bob Strauss is taking over as chairman of Parts Unlimited. As many of you know, I've been here for eleven years, and for the first eight of those, I had the privilege of working for Bob. He was the person who hired me back when I was head of sales at another manufacturer. I'll always be grateful to Bob for giving me a chance to be COO of this company and for mentoring me over the years. When he retired, I took over for him as CEO and chairman.

"Effective last week, the board of directors has re-appointed Bob to be board chairman," Steve says, his voice starting to quaver. Maxine watches with amazement as he wipes a tear from his eye. "Of course, I support this move and I look forward to working with Bob again. I've asked Bob to come out and share some words with us and tell us what it means for the company."

Until this moment, Maxine hadn't realized how much of a setback this was for Steve. She had heard that it was a demotion, but to be honest, she didn't really understand or care much about these types of changes at the executive level. Executives came and went, often without much impact on her and her daily work. But she's riveted by the drama unfolding in front of her.

A slightly stooped older man with white hair and a wry smile walks on stage and stands next to Steve.

"Hi, everyone. It's great to be here in front of you after so many years. I even see some familiar faces, which makes me very happy. For those of you who don't know me, my name is Bob Strauss. I was CEO of this company for fifteen years, back when dinosaurs roamed the earth. And even before that, I was an employee at this great company for nearly thirty years. As Steve mentioned, it was with great hope and pride that I recruited him away from another company many years ago.

"Since retiring, I've continued to serve on the board of directors. The job of the board is very simple: to represent the interests of the company shareholders, which includes almost all of you. We want to ensure that the company's future is secure. If you have a pension or have been part of our employee retirement stock purchase plan, this is probably as important to you as it is to me.

"We do this primarily by keeping company executives accountable, by hiring, and, umm, occasionally firing the CEO," he says plainly. Maxine's

breath catches slightly—until that moment, Bob seemed like a friendly grandfather. Apparently, he has a stricter side.

"Just by looking at the stock price, you know that the markets don't think we are performing as well as we should be. When our company's stock price goes down while our competitors' shares are going up, something has to change.

"I like to think that companies have two modes of operations: peacetime and wartime. Peacetime is when things are going well. This is when we are growing as a company and can continue business as usual. During these times, the CEO is often also the board chairman. However, wartime is when the company is in crisis, when it is shrinking or at risk of disappearing entirely, like what's happening to us now.

"During wartime, it's about finding ways to avoid extinction. And during wartime, the board will often split the roles of CEO and chairman." Bob pauses, squinting into the bright lights, looking across the entirely silent audience. "I want everyone to know that I have complete confidence in Steve and his leadership. And if all goes well, we'll figure out how to get him the chairmanship again so I can go back into retirement where I belong." The crowd laughs nervously as Bob waves and makes his exit.

Steve steps up to the front of the stage and says, "Could everyone give a round of applause for Bob Strauss?"

After a muted applause, Steve resumes, "The company goals this year were to stabilize our business. Our manufacturing operation makes up two-thirds of our revenue, which has remained flat but still profitable. This has been the mainstay of our business for almost a century, and we've been able to fend off our very fierce Asian competitors.

"However, our retail operations continue to underperform. Our revenue is nearly five percent lower than last year," he says. "Our biggest quarter is still coming up, so there's hope. But hope alone is not a strategy, and you can see how Wall Street has reacted to our performance so far. However, I remain confident that the Phoenix Project will help us adapt to these new market conditions.

"So, without further ado, I'll turn it over to Sarah Moulton, our SVP of retail operations, to describe why the Phoenix Project is so important to the future of the company."

Sarah walks onto the stage wearing a strikingly beautiful royal blue business suit. Whatever Maxine's opinion of Sarah, she grants that Sarah

always looks fabulous. In fact, she would look right at home on the cover of *Fortune*—intelligent, aggressive, and ambitious.

"As Steve and Bob mentioned," Sarah begins, "we are in a time of incredible digital disruption in retail. Even *our* customers order online and through their phones. The goal of the Phoenix Project is to enable our customers to order however they want, whether it be online, in our stores, or even through our channel partners. And wherever they order, they should be able to have their product delivered to their homes or to pick it up in one of stores.

"This is what we've been trying to do for years. Right now, our stores are still in the dark ages. That was Parts Unlimited 1.0. The Phoenix Project will create Parts Unlimited 2.0. There are so many efficiencies we can create to help us compete against the e-commerce giants, but we must innovate and be agile. In order to remain relevant, people need to view us as a market leader creating new business models—what worked for our first century may not work for our second."

As always, there's some validity to what Sarah says, Maxine grudgingly acknowledges, but she can be so condescending.

"The Phoenix Project is the most important initiative for our company, and we're betting our survival on it. We've spent nearly $20 million on this project over three years, and customers still haven't seen any value," she continues. "I've decided that it's time for us to finally get in the game. We will be launching the Phoenix Project later this month. No more delays. No more postponements."

Maxine hears an audible gasp from the whole audience and a loud buzz of urgent murmuring. Sarah continues, "This will finally give us parity with the competition, and we will be poised to regain market share."

Maxine sighs in frustration. She understands Sarah's urgency, but it doesn't change the fact that there are over a hundred developers who are nowhere near as productive as they should be, struggling to perform routine builds, spending too much time in meetings, or waiting for things they need. Sarah's speech sounds like listening to a general tell you how important winning the war is and then finding out that all the soldiers have been stuck in port for three years.

On the other hand, at least Sarah hasn't pitched something completely new today.

Steve thanks Sarah and then quickly reviews the company financials, and an injury that happened in one of the manufacturing plants last month. He talks about Hannah, who had her finger crushed by a stamping machine, and how they've replaced that machine with one that has a sensor preventing the plates from closing when anyone is the danger area. He applauds the team for not waiting for budget to act on this, "Remember, safety is a precondition of work."

Maxine loves these report-outs and has always been impressed and emotionally moved by how much Steve cares about employee safety.

He says, "That almost concludes our report-out. We have about fifteen minutes for questions and answers."

Maxine's attention wanders as people ask Steve questions about the revenue forecast, the performance of the physical stores, the recent issues in manufacturing…But when someone asks about the payroll outage, she's jolted alert before shrinking back into her seat while straining to hear every word.

"I apologize to everyone who was affected by this," Steve replies. "I understand how disruptive this was to everyone, and rest assured that we've taken very specific actions to make sure that it never happens again. It was a combination of technical problems and human error, and we think we've remedied both of them."

Maxine closes her eyes, feeling her cheeks turn bright red, hoping no one is looking at her. She can't see how her exile to the Phoenix Project could possibly be considered a remedy.

CHAPTER 3

• *Monday, September 8*

After the Town Hall, Maxine returns to her desk. She looks at her calendar. It's day four of her imprisonment and her quest to perform a Phoenix build, but it feels like it's been nearly a year, the hours passing like molasses.

She gets a notification on her phone, startling her back to reality:

> Phoenix Project: stakeholder status update (starting in 15 minutes).

This is a new meeting for her. To further her quest, she has asked everyone to feel free to invite her to any meetings. It beats sitting at her desk, and she's still trying to get the lay of the land. She's hoping to find someone who can get her some of the things she needs. She's been careful to avoid getting assigned any action items or to volunteer to work on fun-sounding features—she cannot be distracted from the Phoenix build.

Everyone around here thinks features are important, because they can see them in their app, on the web page, or in the API. But no one seems to realize how important the build process is. Developers cannot be productive without a great build, integration, and test process.

She arrives early and is surprised that there's no room except in the back. She stands against the wall with five other people. Looking around, her eyes widen—all the movers and shakers in the company are here. Maxine smiles when she sees that Kirsten Fingle is leading the meeting. She heads up the Project Management Office. Maxine loved working with her when she was supporting a major program that had several of Kirsten's project manager ninjas assigned to it—they were typically reserved for the most important projects that required lots of coordination across many groups within the company. They're aces at making things happen. They can escalate and resolve issues quickly, often with a single text message.

At the front of the room is Chris, who gives her a terse nod—he oversees the efforts of over two hundred developers and QA people,

dominated by the Phoenix Project. Chris is glaring at someone across the table from him who looks like Ed Harris from *Apollo 13*. When she quietly asks the person next to her who he is, he responds, "Bill Palmer, the new VP of IT Operations. Promoted last week after the big executive purge."

Great, Maxine thinks. But she enjoys seeing the seniority of the people in the room. It's like being up on the bridge of the starship *Enterprise* and watching the officers interact with each other.

She enjoys the first fifteen minutes of the meeting. It's chaos. Everyone is trying to decipher what exactly Sarah meant at the Town Hall when she said that the launch was "later this month." Kirsten says emphatically, "The date is still being negotiated and nothing specific has been communicated to me yet." *Could it really be another false alarm*? Maxine thinks, incredulous.

Maxine guesses that there is extra urgency as they review the business priorities, the top issues needing escalation and attention, and the priorities that need deconflicting. She doesn't know what all the acronyms mean, but she adds the ones that she thinks are actually important to her list and leaves out the insipid corporate argle-bargle.

As the meeting drones on and against all her expectations, she grows bored as the focus turns toward meaningless minutia, passionately pushed by…she honestly has no idea. *Does OEP stand for "Order Entry Protocol" or "Order Entry Program"? Or were they talking about the OPA again? Or maybe they were the same thing? Do I really care?*

Forty minutes later, her eyes are glazed over—it's the "task status" phase of the meeting, and Maxine has lost all interest. If she had anything else to work on, she would have left by now.

Her feet hurt from standing so long, and she is reconsidering her decision to stay in the meeting when she hears someone complaining about how long they've been waiting for something they need. She smirks as she thinks, *Join the club. That's what I do all day long.*

One of the Dev managers responds from the "junior people" side of the table. "Yes, we're definitely behind, but we have a couple of new developers starting this week to help, and they should be up to speed and productive in a week or two."

Ha. I'm really good at this stuff and I've made almost no progress, she thinks, looking at the floor. She smirks to herself. *Good luck, chumps.*

There is a long, awkward silence in the room. Maxine looks up. To her horror, everyone is looking at her—she realizes that she must have said something aloud.

She looks at Chris, who has a stunned expression on his face and is wildly gesturing with his hands at her in a "no, no, no" kind of way.

From the front of the room, Kirsten quickly says, "Great to see you, Maxine! I had no idea you were on Phoenix. We're glad to have someone of your experience helping this effort—you couldn't have shown up at a better time!"

Chris buries his face in his hands. If Maxine hadn't already been standing against the wall, she would have been backpedaling. Mimicking Chris, she waves her hands in front of her. "No, no, no… Sorry, I've only been here a few days. You're all doing an amazing job. Please carry on—I'm just here to help with documentation and builds."

Kirsten, with the earnestness that makes her so effective, doesn't let it go. She leans forward. "No, really. I think you said, 'Good luck, chumps.' I'm always interested in your perspective, given your extensive success in plant operations. I'd love to better understand what made you laugh."

"I'm sorry I laughed," she starts. "It's just that I've done nothing except try to get Phoenix builds going on my laptop since last Wednesday, and I'm basically nowhere. I'm waiting for credentials, license keys, environments, configuration files, documentation, you name it—I know everyone has a lot on their plate and I know that Phoenix is such a large application that getting all the pieces together to do a build must be a pretty immense undertaking, but if we all want our developers to be productive, they need to be able to perform builds on Day One. Ideally, they should be writing code in a production-like environment so they can get fast feedback on whether the code they write works with the system as a whole. After days of trying, I still don't have anything resembling the whole system—I have a box of subassemblies, with a whole bunch of missing parts. And I'm really, really good at this stuff."

She looks around the room and gives a half-hearted shrug to Chris. She really needed to get that off her chest. Chris looks aghast.

"I'm just hoping these new engineers that you've been hiring have better luck than me, that's all," she concludes quickly.

There's a long awkward silence. Randy nods emphatically and crosses his arms, looking smug. Someone from across the table laughs

loudly. "She is so right! They'll need a lot more than luck! Getting a Dev environment around here is like going to the local DMV to renew your driver's license—take a number, fill out a bunch of forms, and wait. Hell, I can get a driver's license in a day . . . it's more like trying to get a permit to start a new construction project—no one knows how long it'll take."

Half the people in the room laugh unkindly, while the other half of the room is clearly offended.

Maxine looks at the wise-cracking person who spoke—he's about forty-five, slightly overweight in an "ex-athlete" sort of way. He's square-jawed, oddly clean-shaven, with big, square glasses. He's wearing a skateboarding T-shirt, and his face has a permanent scowl.

Based on his crankiness, Maxine bets that he's a senior developer—being stuck in an environment like Phoenix for a long period of time must take a toll on people.

Someone from the front of the room starts responding—she recognizes William, the *super nice* Director of QA, who has gone *out of his way* to help her. "Look," he says, "our teams are getting further behind on testing, so we all agreed that in order to hit our dates we would deprioritize environment work—shipping fully tested features would take priority. We all knew that this would increase the lead times for getting environments to our teams. Trust me, my teams are being hit just as hard as yours—QA needs environments to test in too."

The cranky developer immediately responds, "William, you got suckered. That was a terrible decision. This is a disaster. Maxine is right—developers need environments to be productive. You should have an entire *team* of people assigned to fix the environment creation process. I'm on three projects that need staging environments, all of which have been waiting months. In fact, this is so important, I'd like to volunteer to help," he says.

"Denied," says Chris wearily from the front of the room. "Stay in your lane, Dave. We need you focused on features."

William says, "Wait, wait . . . I'll have you know that we're not actually the bottleneck for environments—we have several environments that are ready to go, but we still need login accounts from Security and storage and mount points from Ops. I've escalated it but haven't heard anything yet."

Chris points a finger accusingly at Bill and turns to Kirsten, "I need help on escalating our needs to Operations."

Bill quickly responds. "If we're the bottleneck, I need to know. Let's figure out how to get William what he needs."

Kirsten nods, appearing slightly exasperated. Maxine assumes it's because more and more dependencies are surfacing. "Yes, good idea, Bill. Alright, let's move to the next milestone on the list."

As Kirsten talks, Chris turns to look at Maxine, his expression screaming, *Which part of "lie low" did you not understand, Maxine?* Maxine mouths the word sorry.

Out of the corner of her eye, she sees a younger man kneel by Kirsten, whispering in her ear while gesturing at Maxine. Instead of wearing khaki pants, he's wearing jeans and is holding a black Moleskine notebook.

Kirsten nods and smiles at him, points at Maxine, and whispers a couple of sentences in return. The young man nods, furiously taking notes.

Maxine decides to make a beeline for the door, leaving as quickly as possible before she does something else stupid.

She makes it into the cool hallway, relieved to be out of that hot, stuffy room. She heads to the kitchen, where it is even cooler. She's thinking about getting a mug of coffee, maybe her fifth today, when she hears someone say behind her, "Hello, you must be Maxine!"

She turns around. It's the young man from the meeting who was talking to Kirsten. He smiles broadly and extends his hand, saying, "Hi, I'm Kurt. I'm one of the QA managers who works for William. I heard in the meeting that you need license keys and environments and a bunch of other things to get a build running? I think I can help."

For a moment, Maxine just stares back at him, not sure if she heard him correctly. For days, her life has been to search every nook and cranny for the components needed to build Phoenix. For days, she's been submitting ticket after ticket into an uncaring and faceless bureaucracy. She's stunned that someone actually seems to want to help her.

Maxine catches herself staring at Kurt's outstretched hand, snaps back to reality, and shakes it. "Nice meeting you. I'm Maxine, and, yes, I'll take any help I can get to get a Phoenix build going!"

She adds, "I hope that I didn't step on anyone's toes back there. I'm sure everyone is doing their best, you know, given everything that is going on…"

He smiles even more broadly, pointing his thumb back toward the conference room that they were in. "Those folks? Don't worry about it. They're in such deep trouble that they're all covering their asses and

throwing each other under the bus. I doubt they'll even remember what you said by the end of the day."

Maxine laughs, but Kurt is all business. "So, you need to get Phoenix builds going. How far have you gotten and what do you still need?"

Maxine slumps. "Not nearly as far as I want, and it's not for a lack of trying." She describes in considerable detail what she's done so far and all the steps that still remain. She opens her checklist on her tablet, showing him all the open to-dos, pointing out everything she's waiting for.

"Wow, most people give up long before getting as far as you have," Kurt says. "May I see that?" he adds, gesturing to her tablet.

"Sure thing," she says, handing it to him. Kurt runs his finger down the list, nodding and appearing to compare it with another list in his head.

"No problem, I think I can get you almost all of these things," he says. And with a smile, he adds, "I'll throw in a couple other things that I'm guessing you'll need later. Don't worry, you couldn't have known. We had to learn the hard way too. No one around here documents the build environment very well."

Kurt takes a picture of her list with his phone and gives it back to her. "You'll hear from me in a day or two," he says. "The Phoenix Project is in the Stone Age. We've got hundreds of developers and QA people working on this project, and most can only build their portion of the code base. They're not building the whole system, let alone testing it on any regular basis. I keep pointing this out to the powers that be, but they tell me they have everything under control."

He looks pointedly at her. "You wouldn't put up with that back in your old MRP group supporting the manufacturing plant, right?" he asks.

"No way," she responds quickly. "It's like that guy said in the meeting—developers need a system where they can get fast and continual feedback on the quality of their work. If you don't find problems quickly, you end up finding them months later. By then, the problem is lost in all the other changes that every other developer made, so the link between cause and effect disappears without a trace. That's no way to run any project."

Kurt nods. "And yet, here we are, running the Phoenix Project, the most important project in the company, like we would have run a program in the 1970s. Developers coding all day and only integrating their changes and testing at the end of the project. What could go wrong?" he adds with smirk. "They keep telling me these decisions are above my pay grade."

They both laugh.

Kurt doesn't seem bitter or cynical. He radiates a good-natured vibe with an easy acceptance of the way the world works. He continues, "I envy how much your manufacturing team got done and how many platforms you supported. We've got ten times as many people on Phoenix, but I suspect your old team gets a lot more done than we do."

Maxine nods. She definitely misses her old team.

"Oh, and by the way, there's a rumor that might interest you," Kurt says, looking around as if afraid of being overheard. "Word is that Sarah pushed for Phoenix to launch this week, and that Steve just approved it. All hell is about to break loose. Let me know if you want to tag along as they assemble the release team. That'll be super fun to watch."

After that strange interaction, Maxine sits back at her desk and realizes that she is waiting again. Absentmindedly, she looks at a quote she always has taped to her desk from one of her favorite Dr. Seuss books, *Oh, the Places You'll Go*.

The book describes the dreaded Waiting Place, where people wait for the fish to bite, wind to fly a kite, for Uncle Jake, for pots to boil, or a better break…Everyone is just waiting.

> NO!
> That's not for you!
> Somehow you'll escape
> all that waiting and staying.
> You'll find the bright places
> where Boom Bands are playing.

Everyone on the Phoenix Project is stuck in the Waiting Place, and she is determined to rescue everyone from it.

It's 11:45 a.m. Maxine looks at her calendar. It's only day four of her forced exile. While she hadn't hear back from Kurt, she did manage to get access to the third of four source code repositories. Today, she decides that she cannot wait for other people any longer.

She is going to build something.

Over the next four hours, she tries running every Makefile, maven POM, bundler, pip, ant, build.sh, build.psh, and anything resembling a build script she can find. Most fail immediately when she runs them. Some barf out alarmingly long error messages.

She pores through all the error logs, scanning for any clues on how to get something to actually run, sifting through all the poo for anything resembling a peanut—laborious and unpleasant work. She identifies at least twenty missing dependencies or executables that she needs. Several times she asks around to see if anyone knows where to get these things, opening tickets and sending out emails, but no one knows. She spends three hours searching around the internet for clues, pouring through Google and Stack Overflow.

In a moment of very poor judgment, she decides to try building some of the missing components from scratch, from similar sounding components that she finds on GitHub. Five hours later she's in a terrible mood—exhausted, frustrated, irritated, and absolutely positive that she has just wasted an entire day going down rabbit holes she never should have gone down.

Basically, she tried to forge missing engine parts by melting down aluminum cans. *That was really dumb, Maxine*, she thinks.

That night when she gets home, she realizes that she has brought all the frustration from work with her. She warns her husband and kids that she is currently incapable of conversation and finds two mini-bottles of Veuve Cliquot rosé in the fridge. When her teenage kids see her, they know immediately to avoid her. She is wearing her "Mom is in a super bad mood" face.

While they prepare dinner, she crawls into bed and watches movies.

What an utter waste of a day, she fumes.

She reflects upon the difference between a great day and a bad day. A great day is when she's solving an important business problem. Time flies by because she's so focused and loving the work. She's in a state of flow, to the point where it doesn't feel like work at all.

A bad day is spent banging her head against a monitor, scouring the internet for things she doesn't even want to learn but needs to in order to solve the problem at hand. As she falls asleep, she tries not to think about how much of her life is spent searching the internet for how to make error messages go away.

It's a new day. Fresh from a good night's sleep, she sits at her desk, intent on *not* repeating yesterday's mistake. Just because she felt busy doesn't mean that she actually did anything meaningful. In a terminal window, she pulls up her work from yesterday and, without looking at it, deletes it all.

Then she pulls up all of her open tickets in the ticketing system. She refuses to feel powerless, at the mercy of distant powers, trapped inside a cold bureaucracy actively impeding her goals, aspirations, desires, and needs.

Maxine has a long and complicated history with ticketing systems, both good and bad.

Last year, she funded a Kickstarter project for a mug that promised to keep her coffee or tea at whatever temperature she wanted for hours. It even had a Bluetooth connection so that she could see and set the temperature of her drink from her phone. She loved the idea of it and quickly paid five hundred dollars to help the inventor.

She was thrilled every time she got a notification: when the inventor reached her funding goals, when a manufacturer was selected, when the first production run started, and, most importantly, when her mug had been shipped. It was so satisfying to be a part of the collective journey and to eventually hold one of the first five hundred mugs made.

The Dev ticketing system felt totally different. It was the opposite of the joy and anticipation she had felt with her magic mug. Instead, it reminded her of the horrific experience of getting her first high-speed DSL broadband package working back in the 1990s. Although she received the DSL modem right away, she then had to deal with the internet service reseller (who sold her the DSL service) and the phone company (who owned the copper wires to her house).

Whoever did the installation at her house must have screwed up because nothing worked—and when she called either organization, they told her that it was the other's responsibility to fix it. Sometimes they could find a ticket related to her previous conversations, sometimes not. She was trapped in a cruel, uncaring, Kafka-esque bureaucracy. For four weeks, her awesome DSL modem did nothing except flash a red blinking light. It was as useless as a brick, and she had countless open tickets with both organizations.

One day, Maxine decided to take an entire day off of work just to tackle activating her DSL line. After three hours, she finally managed to talk her

way up the chain to a Level 3 escalation support person who had access to both ticketing systems. He was amazing and obviously incredibly savvy, able to thread Maxine's request to the right department, using the right keywords to get the two supervisors from both companies on the phone to line up all the necessary work. An hour later, she finally had her 64 Kbps broadband connection working.

Even decades later, she remembers how grateful she was to that support person. She had told him, "I'd love to talk to someone that matters about what an amazing job you did and how grateful I am for your help." She happily stayed on hold for ten more minutes to talk to a supervisor and then spent ten minutes gushing in great detail about all the help that she had received.

It was so important to Maxine to describe how extraordinary and even heroic everything that Level 3 support rep did and how much she valued his help. Maxine was gratified to hear that he was being considered for a promotion and that her phone call likely sealed the deal.

Afterward, Maxine spent an hour watching the blinking green light and savoring the blazingly fast download speeds.

Remembering that savvy support rep, Maxine reminds herself that she loves solving problems, that she loves challenges, and how important her work is. It will help every developer become more productive.

Taking a deep breath, she summons the relentless optimism she's been accused of having, and carefully scans her email for any new ticket status changes. She ignores all the team status updates, except for the ones where people are yelling at each other in ALL CAPS. She wants to know who the hotheads are so she can avoid them.

Continuing to scroll, Maxine's heart leaps when she sees the subject line:

> Notification: change to ticket #46132: Phoenix Dev environment.

Is her Dev environment finally ready?

She tries not to get excited, because she's been fooled before. Twice yesterday she got a notification, but it was only for the most minor status

changes on her ticket. The first was when someone finally looked at the ticket; the second was when it was reassigned to someone else.

Maxine clicks on the link in the email, which pulls up the entire history of the ticket on her browser. She squints and leans closer to her screen. She opened the ticket six days ago (if she counts the weekend), and there have been seven status changes as different people worked on her environment request. As of 8:07 this morning, the ticket is marked closed.

She hoots loudly. At long last, the Ops people are done! She is in the build business!

But she's confused. Where's her environment? How does she log in? What's the IP address? Where are the login credentials?

She scrolls to the very bottom to the notes and comments, reading what each person typed in as they were worked on her ticket. It got bounced from Bob, to Sarah, to Terry, back to Sarah, then finally to Derek. At the very bottom of the notes, Derek wrote:

> To get a Dev environment we need approval from your manager. The correct process is documented below. Closing ticket.

Maxine's face turns hot and bright red.

Derek closed my ticket?! After all that waiting, he just closed my ticket because I didn't have an approval from a manager?

Who the hell is Derek?! Maxine yells to herself.

She moves her cursor around the ticketing screen, trying to find anything to click on. But the only clickable link is the policy document that Derek provided. She can't find any way to find out who Derek is or how to contact him. She finds a button to reopen the ticket, but it's grayed out.

Thanks a lot, Derek, *you shithead,* thinks Maxine angrily.

Fuming, she realizes she needs a break. She stomps out of the building and sits on a bench in front of the office, she takes a deep breath, closes her eyes, and counts to fifty. Then she walks back into the office and sits down at her desk.

She clicks the button to open a new ticket. When the blank ticket appears, with the countless fields that need to be filled in, she almost gives up and goes home. Almost. Instead, she forces herself to smile and summons up her friendliest self:

> Hi, Derek. Thank you so much for working on my environment, which we badly need for the Phoenix Project—please see Ticket #46132 (link below). I'm assuming I can paste my manager's approval (Randy Keyes) in this ticket? I'll get an email from Randy with his approval for you in the next 30 minutes. Can I call you to make sure this gets processed today?

She clicks the Submit button, writes a short email to Randy asking him to okay the creation of a Dev environment, and runs to his desk. She's relieved to see that he's there and not in a meeting. She tells him what she needs and then stands over him as he sends a reply that merely says "Approved."

By the time she gets back to her desk, she feels a sense of grim determination and relentless focus. She will do whatever it takes to get her Dev environment. She sits down, copies and pastes Randy's approval from her email into the service desk ticket, and adds the note:

> Derek, thanks so much. Here's my manager's approval. Can I still get this environment today?

She hits submit.

She pulls up the corporate phone directory and scans for all the Dereks in the IT organization. There's three. The one in the helpdesk department is the most promising.

She emails him a nice, friendly note, CCing Randy, to thank him in advance for all his help and to let him know how grateful she'll be to get builds going for Phoenix, practically begging him not to put her ticket at the bottom of the queue where it will take another week to process.

She hits send. Five seconds later:

> This is an automated response—please put all service desk related tasks into the service desk. I will do my best to read all my emails and respond in 72 hours. If this is an emergency, please call this number . . .

She curses. She imagines Derek sitting with his feet up on his desk, guffawing at her misery. She prints out everything related to Ticket #46132, her emails, and the names of the three Dereks, looking up

where each one of them sits. Helpdesk Derek is two buildings over, lower level.

She steps out of the elevator on Derek's floor and sees rows of small cubicles with people wearing headsets in front of computers. There are no windows. The ceilings are surprisingly low. She can hear the electrical buzz of florescent lights. It's oddly quiet. *There should be more fans running to make the air less stale*, she thinks. She hears people typing and a few people talking politely to people on the phone.

Seeing all this, she feels suddenly very, very guilty about her previous anger toward Derek. She asks someone where he sits. After walking through the maze of cubicles, she finally sees Derek's nameplate, printed by an ink-jet printer without enough ink. Then she sees Derek.

He's not a hardened bureaucrat at all. Instead, he's in his early twenties. He's Asian, with a remarkably earnest expression as he scans something on a small LCD screen. Maxine has had laptops with bigger screens than this budget PC setup. She feels even worse about all the bad things she's thought about him.

There are no extra chairs, so Maxine kneels down beside him. "Hi, Derek," she says in her most friendly voice. "I'm Maxine, the person who submitted the ticket about the Dev environment last week. You closed it this morning because I didn't have a manager's approval. I just got it. And because you closed the ticket, I had to open a new one. I'm wondering if you can help me get this through the system."

"Oh, golly, sorry about closing the ticket. I'm new to all of this!" Derek replies earnestly, obviously distraught that he might have screwed up. "All I know is that I'm supposed to make sure that certain requests that need approvals have them. I can't reopen tickets. Only supervisors can do that. And all new tickets go into the queue where they get assigned to the next open person. Maybe my supervisor can help?"

Maxine slumps, dreading what might be ahead. But looking around at the sea of people, she realizes that if she doesn't get this straightened out now, she will never get her Dev environment.

"Absolutely. That would be great, Derek." He smiles and they walk toward one of the outer offices.

Over the next fifteen minutes, Maxine watches Derek's supervisor expertly navigate through the extensive ticket trail, examining its history. In the time since Maxine left her desk, someone named Samantha has already

closed her new ticket, pointing out that approvals cannot be submitted in the "Notes" field.

Maxine refuses to lose her cool. These people are trying to help her. The supervisor apologizes about how inconvenient this is. She merges Maxine's two tickets together, puts Randy's name in the approver field, and resubmits the ticket. "Now Randy just needs to hit one button in the tool and you're good to go! Sorry we can't actually authorize requests—only designated managers can."

"Can he approve it from his phone right now?" she asks, with forced cheer. Apparently not—the helpdesk tool was written before smartphones existed and mobile phones were probably still the size of suitcases capable of showing only seven LED digits.

Maxine sighs, but she thanks them effusively because she is certain she is finally close to achieving her goal. As she turns to leave, Derek asks tentatively, "Do you mind if I ask a stupid question?"

"Of course. There are no stupid questions. Fire away," she smiles, trying not to look manic.

"What's a Dev environment? I've dealt with laptop issues, password resets, and things like that. But I've never heard of an 'environment' before."

And there it is, Maxine thinks, thoroughly humbled. *This week's lesson on patience, kindness, and empathy, brought to you by Derek and the helpdesk department*.

Maxine is proud that she has earned a reputation for being level-headed, compassionate, and having empathy for others. But right now, she feels like she demonstrated none of those things. Is being assigned to the Phoenix Project making her a bad person?

She realizes how misdirected her anger at Derek was. This poor guy didn't hold anything against her because she was a developer. He didn't even know what she was asking for, let alone understand how important it was. In his inexperience, he had merely closed her ticket by following the rules set for him. He was only trying to do his job the best he'd been shown how.

Maxine returns to her desk two hours later. She had taken Derek and his supervisor to lunch—out of appreciation for their help and to atone for thinking the worst of Derek. She got the chance to explain the world

of development to him, and his earnest curiosity was infectious. She described all the exciting career tracks available for technical people outside of the helpdesk, hoping it might encourage him to explore some of the options available to him.

She goes to find Randy to make sure he approves her request. He's not at his desk. She calls him on her cell phone right away.

"I can't approve it until I'm back at my desk," Randy tells her. "I absolutely promise that I'll approve it when I'm out of meetings. It'll be done before five o'clock."

Maxine goes back to her desk, feeling conflicted. She understands the need for automated workflows. On the manufacturing plant floor, the MRP systems she wrote control everything that thousands of people do virtually every minute of the day. You can't manufacture products in large volumes, costing thousands of dollars, without a rigorous process.

Even the helpdesk process, whether here or at the organization she had to deal with to get her DSL model installed decades ago, is a pretty good way to provide consistent customer service most of the time, even when it's delivered through thousands of call center staff.

So why does the ticketing system here feel so awful? We're all part of Parts Unlimited, so why does everything feel like I'm dealing with a government bureaucracy or an uncaring vendor? Maxine ponders. *Maybe it's because when friends do favors for friends, we don't require them to open a ticket first.*

The next day, Maxine sees that Randy delivered as promised—he approved the Dev environment ticket, but it was too late for anyone to start work on it.

Despite this breakthrough, Maxine is still waiting for a Dev environment. Disappointed, she wanders aimlessly from meeting to meeting, not wanting to be idle at her desk.

Killing time in the kitchen waiting for another pot of coffee to brew, her phone buzzes. Screenful after screenful of email notifications about changes to Ticket #46132 appear. A request to get a virtual machine assigned to the distributed systems group, a request for storage from another group, an IP address from yet another group, a network mount point from another group, application installs from three more groups…

Maxine whoops in delight—progress, at last! Santa just mobilized his magical elf army to start building the Dev environment that she so desperately needs. The cavalry is finally coming!

Exhilarated, she reads through all the tickets. So much work is being dispatched to what appears to be the entire Ops organization. Maxine suddenly becomes alarmed at just *how many people* are required to create one environment.

She works at her desk, planning out what she'll do first with her Dev environment, when her phone starts buzzing incessantly. Opening her emails, her jaw drops at the forty notifications in her inbox. At the top of her screen are a flood of new notifications marking all her tickets as closed.

"No, no, no," she moans, starting to trace through the ticket history from the beginning. She sees that the user accounts were created, the mount points configured, but then she sees a note from one of the storage administrators:

> I'm sorry to have to close your ticket. Believe it or not, we've been out of storage space for the last three months. We have a big order for more storage that we can't expedite until January, and worse, all the controllers are already at capacity. Purchasing says we can only order twice per year to get the best quantity discounts with our vendors. You are near the top of the list, so we'll get this scheduled for February.

Maxine blinks.

It's September.

Phoenix is the most important project in the company. They've spent $20 million over three years. And yet, here she is, trying to help, and they won't spend $5,000 on more disk space. And now she won't get a Dev environment for five months! She buries her head in her hands and silently screams down at her keyboard.

Completely defeated, she takes another walk, to nowhere in particular. It's two thirty p.m. None of the meetings on her calendar seem interesting anymore. It's just people complaining about waiting. Waiting for something. Waiting for someone. Everyone is just waiting. And she wants no part of it right now.

She returns to her desk, picks up her jacket and laptop bag, and decides to leave. She doesn't know where she'll go, but she just can't stay here today.

It isn't until she's behind the steering wheel that she decides to drop by her kids' old school. Not to see her kids—they're at that age where they don't want to be seen with their parents anymore. No, she's going to the lower school where the fifth- and sixth-graders meet for after-school activities. She is proud of having helped the teachers create a coding club three years ago, which has become wildly popular. And she is delighted they've found so many students who are having fun with science, technology, engineering, and math before they go into high school.

Maxine believes learning these skills is incredibly important, because coding is a proficiency that every profession is likely to need in the next decade. It won't be just for developers anymore.

She walks into the classroom and immediately sees Maia and Paige, two of her favorite kids to work with. They're best friends but also fierce competitors, sometimes even archrivals. They're both smart, ambitious, and have a gift for problem-solving.

This is the first time this school year that Maxine has visited. She's surprised at how much older everyone looks and how much their coding skills have progressed. Some are writing what look like games in JavaScript, some are working on web servers, and two students look like they're writing mobile phone apps.

She spends the next hour learning what each group is working on, laughing in delight as they show off what they've created, and loving it when they ask her questions. When Maia and Paige ask her to help solve a problem, she quickly pulls up a chair.

They're trying to complete a classroom exercise to compute the mean, mode, and interquartile ranges of an array of numbers in Python. She immediately sees they've made the same indentation mistake over and over again.

Of course, when they try to run their program, the Python parser vomits at all the indentation errors. They've clearly been struggling to figure out what exactly they did wrong, trying everything they can to make the errors go away.

"May I make a suggestion?" Maxine asks, taking a more active role.

"Of course, Mrs. Chambers," Maia says. Maxine sighs, still not sure how she wants to be addressed by teenagers.

Maxine explains to them how Python indentation works and how it differs from most other programming languages. "But whatever language you're using, the most important thing is to run your program all the time," she says. "When I'm doing something for the first time, I run my program every time I change *anything*, just to make sure it still compiles and runs. That way, I don't make the same mistake for hours without knowing. Better to catch the mistake the first time you make it, right?"

She directs them on how to correctly line up some of the indentation. "Let's see if that fixes the first error…"

She scans the buttons on their editor. "It looks like you can run your program just by hitting Control-Enter. Ah, looks like there's one more little change needed. Yep, you've now fixed that first error. Now fix up the next error until you're back into a working state. If you keep checking after each small change, you'll never have something big you need to fix…"

Reflecting on what she said, she adds, "One of the really nice things about running your program frequently is that you get to see it running, which is fun, and that's what programming is all about."

The girls smile in understanding and stomp out the rest of their errors in quick succession. Maxine grins seeing them use the keyboard accelerator she showed them.

The girls are smiling because their program is actually running now. Looking at the output, Maia notices that something seems wrong. Their computed average is way off.

"Hmm…I think this is an 'off by one' error," Maxine says. "This is one of the most common errors that developers make. It often happens when we are looping through every element in an array, and we miscompute which one is the last element. And that's what's happening here—we missed the last element…and believe it or not, when you accidentally process one element too many, that can cause the program to crash or even be exploited by a hacker."

Maxine can't stop smiling. She's so excited to share this lesson, because over the decades, she's learned that state mutation and looping is so very, very dangerous and difficult to get right. That crashing ODBC database driver she fixed in the middle of the night a decade ago, which made her so notorious, was caused by this type of problem.

This is why Maxine is so dedicated to applying functional programming principles everywhere. Learning Clojure, her favorite programming language, was the most difficult thing she had ever done, because it entirely removes the ability to change (or mutate) variables. Without doubt, it's been the one of her most rewarding learnings, because she's found that about ninety-five percent of the errors she used to make (like the ones the girls just made) have disappeared entirely.

Functional programming is truly a better tool to think with.

"You want to see something cool?" Maxine asks the girls. When they nod, she says, "Here's what I would do. It looks a little strange, but you can get rid of loops entirely by using iterators—it's an easier, and much safer, way to write a loop."

She scans the Python documentation she finds on the internet, and types one line of code in their editor, hitting Control-Enter a couple of times and converging on the right answer.

"Voila! Look at this. It does the same thing as what you wrote, but with no loops or conditional logic, such as checking for the end of the array. In fact, it's only one line of code, with no risk of an 'off-by-one' mistake!" she says, actually feeling proud of what she just wrote.

Maxine is rewarded by "oohs" and "ahs" as the girls' eyes widen at what Maxine shows them. Maxine is pleased, because even this small exercise has banished some unnecessary complexity from the world. It might save the girls from decades of frustration and make the world a safer place.

She spends the next hour floating between the teams, having fun watching kids solve problems and teaching them little tricks here and there to make them more productive and joyful. When the kids adjourn, packing up their huge, heavy backpacks, Maxine realizes that she's in a very good mood.

The joy of teaching people something that they want to learn is awesome. Plus, these are really great kids. She thinks about how easy everything is here. You hit Control-Enter and the program builds and runs. If there is an error, you fix it, hit a key, and try again.

In her current work hellhole, it's the opposite. She still can't build any part of the Phoenix system. Somehow builds have ceased being a part of everyone's daily work.

Maia and Paige made the same indentation mistake for a half hour. At Parts Unlimited, there are a hundred developers probably making bigger

mistakes, and it'll be months before they realize they've done something wrong.

Everyone waves goodbye to Maxine, thanking her for all her help. She climbs into her car and slumps in her seat. To her surprise, she feels sad and dispirited—at work, there is none of the joy and learning that she just experienced here. She wonders if this is how everyone on the Phoenix Project feels all the time.

Maxine is about to start her car when her phone vibrates. It's a text message from one of her collaborators on an open-source project she wrote to help her with personal task management. She started this project over five years ago to help her keep awesome work diaries. She's always been a manic record-keeper of how she spends her time.

Initially, it was just to help her be more productive, to triage her incoming tasks from email, Trello, Slack, Twitter, reading lists, and what seemed like a gazillion other places where work is generated. Her app let's her easily push work into GitHub, JIRA, Trello, and the many other tools where she interacts with other people and teams.

Over the years, she's used this program every day to help her run most of her professional and personal life. It's where she sends all her tasks. It's her master inbox, where she can see all her work and move work between all the systems that she works with.

Many other people use her application too, and some have written adapters to other tools they need to connect to. She's constantly amazed that thousands of people around the world use it every day, with over twenty active contributors writing code for it.

She looks at the new pull request from the text message—someone has created a new adapter to the task manager. The proposed change looks fantastic. She's wanted to do it for years. Their change is quite clever, and she approves of the way the automated tests were written to show that his change works without breaking anything else. He also documented his work, writing several paragraphs on what he did and why. She approves of the way he's turned it into a tutorial, so others can do something similar.

She loves seeing other people's ingenuity and their willingness to make the app better. As the project owner, she sees it as her primary responsibility to ensure that any contributor can be productive.

A couple years ago, there were over twenty active pull requests, but for a variety of reasons, she couldn't merge them in—sometimes the changes were conflicting, sometimes her API couldn't quite accommodate what they needed. She knows it's dispiriting when you submit a change to someone's project, but no one ever looks at it or they tell you that it can't be integrated. If that happens enough, people eventually give up or fork your project and splinter the community.

So, when this started happening to her project, she spent every evening for weeks rearchitecting her system so that people could quickly, easily, and safely make the changes they wanted. It took a lot of effort, and she personally rewrote every pull request so that the contributors didn't have to redo their work. But everyone was delighted and grateful when their changes made it in—but not as delighted as Maxine.

Maxine knows that agility is never free. Over time, without this type of investment, software often becomes more and more difficult to change. There are exceptions, like floating-point math libraries that haven't changed in forty years—they don't need to change, because math doesn't change.

But in almost every other domain, especially when you have customers, change is a fact of life. A healthy software system is one that you can change at the speed you need, where people can contribute easily, without jumping through hoops. This is how you make a project that's fun and worthwhile contributing to, and where you often find the most vibrant communities.

She drives home and is delighted that her husband has already taken care of dinner. She regales her kids about her last-minute decision to visit their old middle school and the exciting new generation of geeks.

When they disappear to do their homework, she grabs her laptop and brings up the exciting new pull request. She pulls the code in and spins up the new version on her laptop. She logs in and clicks around, testing some of the corner-cases to make sure he got the details right.

Smiling, she brings up the pull request in her browser and clicks a button, which merges it into the code base. She writes a thank you note to the submitter, complimenting him on his cleverness and initiative.

Before hitting send, she notices something he wrote: "Maxine, I'd throw you a huge party if you could display a desktop notification whenever someone modifies this property…"

Good idea, she thinks. She pulls up her code editor, and in the next fifteen minutes takes a stab at implementing this idea. When it works the

first time, she grins and laughs out loud, clapping her hands in gleeful selfcongratulation. She's in a great mood. You can do so much with so little effort because of all these miracles of technology.

She resumes her note to the submitter:

> Again, really nice work. I'm sure everyone is going to love it as much as I do. Thank you! (And I just added your notification feature. Your offer for throwing me a party is accepted.)

Hitting send, she wonders if the universe is sending her a message. Her afternoon with the middle schoolers and the ease with which she added functionality to her application (which is years older than the Phoenix Project) shows her what coding *should* feel like.

She is able to build things with focus, flow, and joy. She had fast feedback in her work. People were able to do what they wanted without being dependent on scores of other people. This is what great architecture enables.

She has been exiled to the most strategic initiative of the company, on which the survival of the entire company hinges. And yet, hundreds of engineers are paralyzed, unable to do what needs to be done.

In that moment, Maxine decides she must bring this level of productivity that she's helped create for middle-schoolers and her open-source project to the Phoenix Project, even if it means personal suffering in the short term.

CHAPTER 4

• *Thursday, September 11*

The next morning, Maxine still feels triumphant from yesterday's numerous victories. But as Kurt predicted, everyone is freaking out. To everyone's shock and disbelief, the launch was not going to be called off or delayed. Instead, the Phoenix Project is going to be launched tomorrow at five.

Captain Kirk apparently hit the warp speed button, despite Engineer Scotty telling him the dilithium crystals were about to blow. So, no boring status meetings today. Instead, every meeting is in a genuine shitstorm, with people on the verge of panic. One meeting quickly devolved into bedlam and pandemonium, full of questions, objections, and shocked disbelief. People were furiously typing away on their phones and laptops, and a third of people in the room were making phone calls. It was like watching an old movie from the '40s, with reporters racing out of the courtroom to the payphones or back to their offices, frantic to break the news first.

Maxine turns to the person next to her, loudly yelling, "Has Phoenix ever been deployed into production before?!"

"Nope," he yells back.

"Is there a release team yet?" Maxine asks.

"Nope. Chris, Kirsten, and Bill are mustering up a formal release team today, but I have no idea who's going to be in charge," he responds and mimics biting his fingernails in nervousness and fear.

Maxine looks back at him, speechless.

Maxine does not delight in the suffering of others, but watching the fireworks surrounding Phoenix is much more exciting than waiting for people to work on her tickets. She groans, realizing that given this crisis no one will be able to work on her tickets now.

Later that morning, Chris announces that William, the QA director, is in charge of the release team. His goal: get everything into a releasable state and to coordinate with Ops, who were blindsided too.

Poor bastard. She knows that they are in big trouble. The Phoenix developers can't even merge their own code together without accidentally leaving parts of it behind or blowing up the build. Pulling off a successful production deployment seems wildly optimistic. *Or plain bat-shit crazy*, she thinks.

"William, when is your release team meeting?" Maxine asks him as he jogs by. She runs to keep up. "Can I help?"

"First meeting is in one hour. We need all the help we can get," he says, not even breaking stride. Maxine is delighted. Finally, a chance to actually use her skills and experience.

This will be an interesting meeting, she thinks. Maxine has seen how Dev and Ops interact around Phoenix. Instead of acting like an actual team, they act more like sovereign states on the brink of war, with diplomats trying to patch together an uneasy peace, complete with embassies, protocols, and official formalities. Even scheduling a meeting between these two groups requires a summit and lawyers present.

Regardless, she's excited to be in the game. In a perverse way, this is the most fun she's had on the Phoenix Project so far. She realizes she's grinning from ear to ear. *Does this make me a bad person?* she wonders. She grins again, not caring.

Despite trying to arrive early, Maxine is late to the war room. They had to move the meeting twice because the crowd kept outgrowing the space.

It's fifteen degrees hotter in the room than in the hallway, and the air is stale. Nearly fifty people are crammed into a space designed for half that number. She sees Chris, Kirsten, William, and a bunch of the lead developers and managers. Kurt, sitting next to William, waves to her.

On the other side of the table is Bill Palmer, surrounded by a phalanx of faces she doesn't recognize. She notices that there's something…different about them.

The largest one of them to the left of Bill has his arms crossed and a huge, unhappy scowl on his face. He shakes his head in disbelief. "What is wrong with you people? You're telling me you don't know how many Windows servers you need, on top of the handful of Linux servers…Tell me again, how many exactly is a 'handful of servers?' Is that in metric or

imperial handfuls? While we're at it, you have any Kumquat boxes you need, or perhaps a Tandem?"

Flanking him are a woman and a younger man. The way they snicker makes Maxine immediately think of Crabbe and Goyle, the two mean-spirited goons who were best friends with Malfoy, Harry Potter's rival from Slytherin house.

"Uhh…" says one of the Dev managers. "Actually, there is *one* component that can only run on Kumquat servers. It's an extension we had to build off of the existing message bus. But it's only a small modification. It shouldn't cause any problems, and it should add negligible load…"

Maxine hears groans from around the room, and not just from the Slytherins on the opposing side of the table. The younger man sitting next to the large man, whom Maxine is already thinking of as Big Malfoy, sighs. "Technically, there's nothing wrong with Kumquats—we have over a decade of experience running production workloads on them, and we understand their characteristics pretty well. The problem is that the reboot time for that cluster is almost eight hours. We just need to be careful about anything that could involve restarts, like security patching. I'm concerned that certain changes will require multiple reboots, which could mean a day of downtime…or that they'll never come back at all…"

These are all the Ops people, Maxine realizes. No wonder she hasn't seen them around.

"Wes, trust me, we are as afraid of that scenario as you are," the Dev manager replies from across the table. "We've been trying for three years to get this application re-platformed, but it always takes a backseat to something more important."

"Yeah, you developers always make sure features take priority, and you never clean up all the technical debt you create…typical crap," Big Malfoy says, gesturing angrily.

Bill says to Big Malfoy, without even turning his head, "Stow it, Wes. Work the problem. Stay focused."

"Yeah, yeah. Got it, boss," Wes (Big Malfoy) says. "Handful of Linux servers, handful of Windows servers, and one Kumquat server. Got it. Now who can define a 'handful?'"

Maxine watches all the Dev managers put their heads together, tabulating the compute needs for each of their components. It's clear they're just going on gut, not any sort of thoughtful capacity-planning exercise.

Maxine realizes that this release is in even more trouble than she thought. The developers still haven't merged all their code together. And they haven't defined the production environment that the application needs to run in—describing your environment in "handfuls" definitely does not cut it.

Raising her voice, she asks, "How many transactions per second are we expecting for product displays and orders? And how many transactions per second are the current builds capable of handling right now? That will tell us how many servers we need for the horizontally scalable portions, as well as how far we're off for the vertically scaled components, like the database."

The room falls silent. Everyone turns toward Maxine. They seem startled by her common sense question. The woman sitting to the left of Wes says, "Thank you! That is precisely what we need to know!"

Maxine gives a small nod and winks.

Chris stands up. "This is the highest publicity release in the entire company's history. Marketing has pulled out all the stops. They're going to spend almost a million dollars getting the word out about the Phoenix launch. All the store managers have been given instructions to tell every customer to download the app and hit the website Saturday—they're even having contests to see which stores register the most new mobile customers. They're hitting all the industry and business press. They're trying to get either Sarah or Steve on all the news shows—even *Good Morning America*.

"Here are the best calculations that I've been able to get from Marketing," Chris continues, flipping through his notebook. "Expect one million people to come banging on the Parts Unlimited website and mobile apps. If all goes well, we should be prepared to sustain at least two hundred orders per second."

Maxine hears mutters and curses from all around the room.

Wes scans the room and finally turns to Chris, all sense of jocularity gone. "Okay, that's good to know." He gestures at Maxine, "Our smart architect just asked how many transactions Phoenix can handle right now. Well?"

Chris looks to William, who pulls out a printout. "Hot off the press from this morning. In our tests, Phoenix currently handles about five transactions per second. Anything over that causes the database clients to

start crashing due to timeouts, including the mobile apps…I think we're missing a bunch of database indexes, but we haven't figured out which ones yet…"

William looks up. "It's very, very not good, Chris."

Wes sits in stunned silence for a moment. And then in a blunt, world-weary voice, he says to Chris, "We're not going to make it, are we?"

No one says anything. Eventually, Bill asks, "Wes, what help do you need?"

"…I don't even know," he replies. "Maybe just give the teams some air cover so they can stay focused."

At that moment, Maxine hears a loud voice from the doorway. "For the survival of Parts Unlimited, we have to make this work, so of course we're going to make it."

Oh no, Maxine thinks. It's Sarah Moulton.

She's dressed in a bright, expensive looking yellow suit, and her face is so radiant that Maxine wonders how it's even possible. The fluorescent lights in the office usually make people look ghoulish and devoid of color. Maxine wonders if maybe she adds radium to her makeup to make herself glow like a 1950s bedside clock. Sarah has a certain dangerous glamour about her, and everyone in the room seems similarly rapt.

"We are in a market that is shrinking, with fierce competitors taking market share away from us," Sarah says. "Not to mention tech giants like Amazon and twenty new startups that are coming in to disrupt this entire category. As Steve said at the Town Hall, we've had three years to prepare for this. Now it's time for us to go to war and defend what is rightfully ours."

She looks around the room, scanning for signs of resistance or rebellion. "This is the strategy the executives of this company have decided on. Anyone have a problem with that?" she challenges.

Incredibly, Maxine hears herself laughing. Horrified, she covers her mouth. *Keep it together, Maxine!* Quickly, she wipes all expression from her face, like a student caught doing something bad in high school. *Since when have you ever cared what people in authority think about you?* she wonders.

Ever since Chris warned me to keep my head down, she realizes. Maxine forces herself to look calmly at Sarah with her best Lieutenant Saavik expression, radiating only cool, dispassionate logic.

"Something seem particularly funny to you…umm, sorry, what is your name?" Sarah asks, looking at Maxine coolly.

"Maxine," she replies, calmly. "I was laughing because you were talking about *why* you think Phoenix is important. But in this room, we're just trying to figure out *how* to get Phoenix deployed."

"Which is not going terribly well, either," Wes mutters loudly to some nervous laughter.

"I can see that some of you have not bought into our mission," Sarah says, appraising everyone in the room. "Well, as I mentioned in the Town Hall, the skills that got us here are not necessarily the same skills that will take us to where we need to go. As leaders, we need to figure out if we have the right people on the bus. I'll be sure to keep Steve apprised. I know this release is personally very important to him."

Upon hearing Steve's name, Chris looks at Maxine with disbelief and then covers his face with both hands. *Nice job keeping a low profile,* Maxine tells herself.

"Okay, Sarah, that's enough," Bill says, standing up. "Let's go apprise Steve about some of these problems and let the team figure out how to execute the release. We're only getting in the way here."

"Yes, Steve needs to hear about this," she says. Sarah turns to leave, but then looks back at Maxine. "I like that you say what you think. If you're available sometime this week, let's get lunch. I'd like to get to know you better."

What the… Maxine freezes like a deer in headlights.

"As women, we really need to stick together, don't we?" Sarah says with a wink.

With a frozen smile, Maxine says, "Uhh…thank you—I…I'd love to." Immediately, she hates herself, embarrassed that so many people just witnessed her lying so baldly.

"Let's make it so," Sarah replies with a warm smile. "And if you need someone to mentor you, I'd be happy to." She looks at her phone and says, "That's Steve. He needs something from me. I'll leave you to it. Remember, we all need optimism."

When Sarah is gone, Maxine lets out a long breath, not quite believing what happened. She knows how important it is to have a great network, able to find people who can help get important things done. But she's not terribly excited to be associated with Sarah, no matter how influential she is. Maxine is very picky about who she associates with.

For the next hour, Maxine drifts between the various groups as the huge release team tries to fully understand what is required to support the Phoenix launch. There are at least twelve different technology stacks that need to be deployed, more than Maxine had estimated during her build archaeology.

She knew of the various application servers on Windows and Linux, and the front-end applications that run on the web, but she totally forgot about the two mobile applications (one for the iPhone and one for Android), and those all collectively hit at least ten different back-end systems from across the business, all of which required changes in order to accommodate Phoenix.

She had also forgotten that when you throw the Operations teams into the mix, the number of teams involved more than doubles, because getting all those applications running in production requires the server administration teams, virtualization teams, cloud teams, storage teams, networking teams…

All this reminds Maxine of why production deployments are some of the most complex activities of any technology organization, because they require so much coordination between so many different parts of the organization. And Phoenix wasn't just any deployment—it was designed to change how almost every part of the organization interacts with the customer.

The more Maxine hears, the worse she feels. It seems impossible that they can get everything right the first time with this many moving parts. Getting an environment required Maxine to open up scores of tickets, and she still wasn't successful. She's guessing that deploying Phoenix will require hundreds, or maybe even thousands, of tickets.

The project manager in the group she's sitting with says, "Won't we need a bunch of firewall changes too? Not just to external traffic. I don't think some of these systems have ever talked to each other…"

Maxine raises an eyebrow. She hears more groans around the group. "Oh, great. The firewall teams usually need at least four weeks to get change requests through," says the woman who Maxine learns is Patty. "You think our change management process can be slow? We're speed demons compared to Information Security."

Suddenly, Maxine hears a door slam open behind her, and Patty looks up. "Well, speak of the devil. Here's John, our chief information security officer. This should be fun…" she says.

John is in his late thirties. He's about twenty pounds overweight, but his clothes are still baggy on him. Like in an old Western, John is flanked by two people—one male and one female engineer, who looks vaguely familiar. "At last, I've found you all," John sneers, looking around as if he were a sheriff who had hunted down a group of outlaws. "I'm here about this mad plan to deploy the Phoenix application. This deployment will only happen over my dead body."

The woman behind John suddenly looks embarrassed, as if she's seen John say this before. John continues, "The Phoenix Project has millions of lines of new code, and we cannot responsibly deploy it without my team testing it for vulnerabilities. We just came out of a very, very interesting meeting with the auditors, and trust me, they aren't going to take it very kindly if we put something into production that jeopardizes our compliance posture.

"I have it on *pretty* good authority that the CIO and VP of IT Operations were just fired over some compliance audit findings that were no longer tolerable," John continues. "Let that be a warning to you that compliance is not just a moral obligation or a set of contractual obligations…it's also the law."

Maxine wonders how many times John's rehearsed that line. *It's a pretty good line*, she acknowledges.

Kirsten says from the front of the room, "As you know, the decision to ship Phoenix came straight from the top—Steve Masters, the CEO, and Sarah Moulton, the SVP of retail operations. In fact, Sarah was just here, reminding us of that. The release is scheduled to start at five tomorrow so that everything is live when stores open on Saturday morning."

"That's what you think, Kirsten," John says. "I'm going to go speak with Steve right now. Rest assured that I *will* stop this madness."

He turns to Wes. "You were there at the meeting with the auditors—tell them how serious this is and why there is no way the production release can happen tomorrow!"

Wes quickly replies, "No—leave me out of this, John. That train has left the station, and you can't put the toothpaste back in the tube. The only thing we can do is figure out how to keep this rocket from blowing up on the launch pad and killing us all. Pardon the mixed metaphor," he says with a loud laugh, looking around the room to see who's with him.

"Or was that a simile?" Wes asks suddenly, with a puzzled look on his face.

The woman behind John says in a deadpan voice. "It's a metaphor, Wes. When you say something 'is a pile of crap,' that's a metaphor. When you say that something 'is like a pile of crap,' that is a simile."

"Thanks, Shannon," he says with a big smile. "I've always gotten those confused."

John glares at Shannon, and then says angrily to Wes, "I will *not* leave you out of this, Wes. It is your moral responsibility to stop this release!" He turns to the whole room. "It is all of your moral responsibility to stop this release! You all know where I stand—as I said, this release will be deployed over my dead body."

Wes mumbles, "We can always hope."

Maxine hears some nervous giggles as John and his posse leave. Kirsten stands up, looking a bit uncomfortable. "Well, I should take a moment to say that we made a commitment to deploy Phoenix on Friday. But if any of you feel you have a, umm, moral obligation to not participate in this release, please let me know."

Wes chortles. "Kirsten, going down this path is almost certainly the stupidest thing I've seen in my entire career…but to support the team, I promise we'll all do what we can." With an air of weary exhaustion and resignation, he continues, "Let's just get it over with."

Maxine looks around, thinking about the sudden, surreal appearance of Sarah and then John. She's reminded of *Redshirts* by John Scalzi and Wil Wheaton, a funny book loosely based on a *Star Trek*-like universe. It's written from the perspective of one of the redshirts, the nameless low-ranking characters wandering in the background of the show, who learns that interacting with any of the bridge officers is bad news. Whoever is chosen to beam down to the planet with the officers is doomed to die in bizarre ways: an Alteran blood worm, mind virus, carnivorous plants, an errant Klingon disruptor blast. In the book, the redshirts plant sensors everywhere to detect when the equivalents of Captain Kirk or Commander Spock come below deck so that they can hide.

She is disheartened by how Parts Unlimited executives, the bridge officers, are so disconnected from the daily work of the "redshirts" in the technology organization. It was not helpful for Sarah to remind everyone of how "saving the universe" depends on Phoenix. And it was not helpful for John to appeal to their "moral sense of correctness."

We all know the threat the company faces is real, she thinks. The job of the bridge crew is to ensure the company strategy is viable, not to remind them of the strategy or to micromanage everyone to death. Their job should be to ensure everyone can get their work done.

How did this all come to be?

Maxine drags herself back to her desk with a sandwich, exhausted from the endless Phoenix release meetings, surrounded by everyone who has similarly been sucked into the launch vortex. Oddly, she also sees some people happily working at their desks, as if it were just a regular day.

Curious, she asks one of the them why he doesn't seem very worried. He replies with a puzzled look, "I'm a developer—I work on features. I give them to QA and Ops to test and deploy. Then I work on the features for the next release. That keeps me plenty busy."

Maxine leaves, boggling at what he said. She has never in her career abdicated all testing and deployment to someone else. *How can you create anything of value if you don't have feedback on how it's used?* she thinks.

When she gets to her desk, Kurt is there with a black three-ring binder. Seeing her, he flashes a big smile. "I have a present for you!"

It's an eighty-page document full of tabs. Just scanning the section headings makes her heart leap—they're the painstakingly assembled Phoenix build instructions, complete with links to documents, license keys, step-by-step tutorials, and even links to a bunch of videos. One is titled "Getting your uberjar to run in our (very) crazy, screwed up production web cluster (8 min)," and another is "How to monitor your apps despite our Ops groups (12 min)."

She sees twenty-character hexadecimal strings of activation codes and license keys. She sees user names and temporary passwords for network shares. Best of all, there's a link to a four-node virtual machine cluster *with* administrative access! That means Maxine will be able to do whatever she wants without having to fill out another service desk ticket!

She's speechless. She feels her eyes tearing up. *Over license keys?*

She wonders for a moment whether she has lost all sense of perspective. But after being stuck inside the Phoenix Project, having someone actually care about what she needs is…so unexpected and so utterly appreciated.

Maxine is reminded of when she and her family volunteered for a day to help new refugee families. She remembers how her then ten- and eight-year-old kids reacted when families cried when they were given food, soap, and laundry detergent.

There is nothing so rewarding as providing something to someone who really needs your help. She needed help and she received it.

Elated, Maxine flips through the document. She sees a long list of Windows registry keys that need to be set. "Don't worry, Maxine," Kurt says, politely ignoring her emotional reaction. "You have the electronic version of this in your inbox, so you can copy and paste everything."

With a twinkle in his eye, he adds, "And there's a link to a wiki page where you can incorporate any notes if we missed something. There's a bunch of people who really appreciate your work. We've been trying to crack the Phoenix build puzzle for months! But we've never been able to work on this full-time. Your notes helped us put all the pieces together. This saved us months of work!"

Maxine's brow furrows. She has no idea what Kurt is talking about, but she doesn't even care. "Thank you so much! I can't tell you how much this means to me. How can I ever repay you for this?"

"Anything to help another hard-working engineer trying to help other engineers be productive," Kurt says, laughing. But with a serious look on his face, he adds, "If you want to meet the people who made all this happen, despite considerable adversity and huge obstacles that typically prevent feats of greatness like this, come to the Dockside Bar tonight at five. We meet there on Thursdays."

"Wait, hang on a second," Maxine says, suddenly suspicious. "If this all works, how come everyone isn't using it?"

"That is a great question, with some very surprising answers," Kurt says. "The short version is the 'official build team' hasn't exactly authorized these. They seem to view our efforts as a nuisance, or worse, as competition. Which, on the eve of the biggest and potentially most risky application launch in the history of the company, sure does seem odd, doesn't it?

"But by all means, if you like what we've done, feel free to share it with anyone who needs it. I can explain more tonight. Please, try to join us at five—there's a bunch of people who are dying to meet you!" he says. "And good luck with the build!"

Maxine opens up a terminal window on her laptop and starts following the instructions Kurt gave her. Her excitement grows when she realizes that this might be an actual, working Dev environment.

She's exuberant when she's able to log in and type "make" on a command line, which starts streaming screenfuls of happy output onto her screen.

She's delighted as she sees files getting compiled, binaries getting linked, programs being copied, build tools being installed and run…the output keeps going, and going, and going…

Amazingly, things are still building for ten more minutes…fifteen minutes…thirty minutes…she's relieved as it keeps going without an error, but starts to become alarmed at the size of the Phoenix build. It's *huge*.

Forty-five minutes later, she can't hold off going to the bathroom any longer; she was too afraid she was going to miss something if she stepped away. She hurries there and back and is relieved to see that the build hasn't failed, still generating endless output in her terminal window.

She scrolls through the history to see if she missed anything interesting. She decides to skip the next release team meeting just so she can watch the continuing build, which seems a bit irresponsible, but she knows that having a great build process is key to having a good code deployment and release process. And maybe with the help of these mysterious benefactors, she's on the verge of finally conquering the Phoenix build.

The build output is hypnotic and educational, because she's seeing some components of Phoenix for the first time. There's Java JAR files, .NET binaries, Python and Ruby scripts, and lots and lots of bash scripts.

Wait, is that a remote shell and installer that popped up? Before she can figure out what it is, the window is gone. Maxine's awe and concern at the size and variety of Phoenix continues to grow.

She's about to scroll back further when she sees Eclipse being downloaded from somewhere. *What in the world?* she thinks. Twenty minutes later, she could have sworn that she saw an InstallShield installer, but she knows she's getting tired and might be imagining things.

Honestly, after another hour of watching build output, she's having trouble staying focused on the screen. But she can definitely see the

different personalities and tech stacks of the different teams working on Phoenix. She had no idea there were so many.

This is crazy, she thinks. *There can't be this many teams working on Phoenix, right?* And she wonders how any one person could possibly understand the system as a whole, especially when it's built from so many different technology stacks.

Maxine isn't usually a fan of rigid standardization, but she's not a fan of everyone getting to choose whatever they fancy in the moment. Each decision is a commitment to support it for years or even decades—these are decisions that go far beyond just one team.

Like most developers, she's very superstitious that if she stops watching the build, it will fail. Finally, nearly three hours after she started the build, she sees the scrolling output from her build window stop. Her heart falls when she reads:

```
builder: ERROR: missing file: credentials.yaml
```

Damn! She's guessing that she needs a login credential that she doesn't have.

She texts Kurt and he quickly replies:

> Ah, yes. For that you need to open a ticket to get your login tied to your ActiveDirectory account. Only Susan can issue those. Contact info coming.

Instead of emailing Susan, Maxine goes to Susan's desk and learns that this missing file contains a cryptographic certificate that comes from some distant security group. Susan searches through years of emails to find how to get a new one. When she finds it, Maxine takes a picture of the email address with her phone.

She is so close to getting a Phoenix build going!

CHAPTER 5

• *Thursday, September 11*

Still on a high after getting so far on the Phoenix build, Maxine hops in her car to make the five-minute drive to the Dockside parking lot, right on time for Kurt's mysterious meeting.

She suspects that the shiny new Lexus IS300 in the parking lot is Kurt's. She doubts it's the Datsun 300 she parked next to. It's surprising that the meeting is at Dockside. It's not one of the usual hangouts for technology people, but she knows it's a longtime favorite for many of the factory workers.

Maxine asked some people about Kurt that afternoon. Three people gave her enthusiastic endorsements, describing how competent and helpful he was. One development manager in her old group called him one of the smartest people in the entire technology organization. Curiously, one of her colleagues texted her:

> Kurt? He's not the sharpest knife in the drawer, which is why he's stuck in QA. He's also really nosy. Why do you ask?

This made Maxine even more curious. *What exactly is Kurt up to?* His gift of the binder probably saved her months of waiting. But what is his motivation? He clearly has an inside track on getting things people need. She's pretty sure he isn't pilfering corporate resources—and even if he was, why would he give them to her?

As she walks in the door, she's immediately hit by the smell of hops. She hasn't been here in years. She's relieved to see that it's much cleaner and brighter than she remembers. There's no longer sawdust on the floor, and it's more spacious than it seems from the outside.

The bar is half-full, but it's loud—maybe because of the cleanly swept cement floor.

Seeing her, Kurt smiles and waves her over to a group of tables on the far side of the room by some booths. "Hey, everyone, meet Maxine,

the newest member of the Rebellion if I can help it. She's the person that I've been telling you all about."

She immediately recognizes the cranky developer who backed her up in the Phoenix status meeting about environments and is startled to see the petite woman named Shannon who was with John earlier today. There's another man in his late thirties sitting next to someone who looks out of place—he's in his fifties and wearing a bowling shirt. Next to him is Brent, who she also saw in the Phoenix release meeting. He and Shannon are the youngest people at the table.

Everyone has an open laptop in front of them. Suddenly, she wishes she had hers with her—she'd gotten out of the habit of carrying it around lately because she's had so little to work on.

"You remember Dave?" says Kurt, gesturing at the cranky developer. "He's one of the Dev team leads. He complains a lot, but he's probably banging the drum loudest on the need to pay down technical debt and modernize our architecture, platforms, and practices."

Kurt laughs. "The reason Dave is so good is that he never asks for permission!"

Cranky Dave raises his glass at Maxine with the smallest of smiles, as if smiling causes him physical pain, then takes a sip of his beer. Up close, he looks older than her. "Breaking the rules is the only way anyone can get anything done around here," he grumbles. "Can't do anything without twenty meetings." Cranky Dave pauses. "You know, that's the best compliment Kurt has ever given me. You've probably noticed that he's running his own black market inside the company, right?"

Kurt laughs, clearly not bothered by the characterization. "I'm just trying to solve people's problems. If I'm guilty of anything, it's that I care too much about the success of Phoenix, and even the whole company, to allow bureaucracy to kill it! And if that's a crime, I plead guilty! It's a pity no one will ever give us a medal for the great work we do. The satisfaction of helping people must be reward enough, right?"

Everyone groans, and someone from across the table hollers out, "Good one, Kurt."

Ignoring the banter, Kurt points to the man in his late thirties who is wearing a funny vendor T-shirt. "This is Adam, one of my test engineers. But don't let his title fool you—he's a developer at heart, and he's also one of the best infrastructure people I've ever met.

"You can thank him for the all those virtual machines and pre-built containers you got—he built them all. And that's only a fraction of what Adam does. His day job is helping automate a big chunk of the legacy test suite that we inherited from an outsourcer."

Adam smiles sheepishly. "Actually, Brent over there did most of the work," he says. "He's an ace at infrastructure. We've been working together to try to automate environment creation for over a year. It's been a tough road, working evenings and weekends, because it's not officially sanctioned. Despite all the dead ends and cul-de-sacs, we're proud of what we've been able to achieve.

"Your build notes were awesome, Maxine. Brent here almost fell over and died when he was reading through them. He'd been trying to piece that together for months," Adam says.

Brent smiles at Maxine. "That was amazing detective work, Maxine. Documenting all those environment variables was so helpful!"

"Let us know how the environment works for you," Adam continues. "It's such a pain to get things from Operations through normal channels, so we scraped together enough hardware to build a cluster big enough to support a couple of teams. Now you can get an environment on demand, without needing to open up a ticket."

Maxine blurts out, "Wow, thank you so much. The environment worked! I got three hours into the Phoenix build with it until it failed because of a missing certificate."

"Wow! That's amazing," Brent says.

"So where did all that hardware come from if not from Operations?" Maxine asks.

Adam smirks. "Kurt has his ways—a little from here, a little from there, you know? Kurt keeps saying that it's best not to ask where it comes from…I've always suspected there are a bunch of people missing entire server clusters if they'd bother to check."

Kurt feigns a hurt expression. "Server-hoarding is a huge problem," he says. "Because it takes so long for Ops to get anything to us, people always ask for way more than they need. And that makes Ops' job harder and lengthens the lead times for everyone else, making the shortages even worse! It's like being in the old Soviet Union, where you have to wait in line for everything. You could say we're creating a secondary market to ensure that some of those unused environments go where they're needed most.

You know, to ameliorate the mismatch between supply and demand," he says.

Cranky Dave mutters, "Don't get him started," rolling his eyes as Kurt lectures like a professor.

Adam adds, "But Dave is right—Kurt *is* running a black market."

"Pay no attention to them, Maxine," Kurt continues. "Next down the table, we have Shannon, a security engineer who works on building automated security tools. She spent nearly five years in the data warehouse team before that. She's currently working with Brent, experimenting with some machine-learning and data visualization toolkits and standing up some big data infrastructure, trying to get ahead of some of the marketing initiatives that we know are coming. You probably remember her from the full-scale red-team exercises that she ran last year."

Maxine smiles. That's why she looked so familiar. She definitely remembers—it was the first time she had been the target of a no-holds-barred penetration test. They had tried to plant malware by getting physical access to the manufacturing facilities, sending emails with malicious links, pretending to be company executives, and, in one case, one of their most critical vendors.

She had been very impressed. *It takes a lot of balls to run those types of exercises*, she thinks. Maxine remembers one person being fired for trying to do one because he made a bunch of people look bad.

Shannon looks up from her laptop and says, "Nice meeting you, Maxine. I remember your group. You were one of the best-prepared in the whole company. I was very impressed that everyone in your division knew not to click on links in emails, no matter how official they looked. Someone did a great job training everyone."

Maxine nods with respect, saying, "Nice meeting you, Shannon. We spent weeks fixing the problems you all found. Nice work."

Shannon looks back down at her laptop and types something. Suddenly, she looks up at everyone and says, "Oh, by the way, sorry about that episode with John. He's such a tool. But he's my boss."

Everyone laughs, and several people imitate Shannon's expression from earlier today.

"Up next is the aforementioned Brent, who has his hands in everything infrastructure related," Kurt continues. "If it connects to AC power, Brent has probably mastered it. Networks, storage, compute, databases.

But he's not just good with a screwdriver, he's always on the frontier of automation. Unfortunately, he's so good at what he does, everyone seems to have him on speed-dial. And he's on pager duty way too often, which we're trying to fix."

Brent merely shrugs his shoulders. Suddenly, the camera flash on his phone flickers and notifications flood his screen. He picks up his phone and mutters, "Dammit, another outage call. I probably need to jump on this." He drains the rest of his beer and starts dialing his phone.

"Yeah, that's a real problem," says Kurt, watching Brent walk away. "We've got to bring some sanity to his work life. He's brilliant, but because of the way people dump things on him, he hasn't been able to go on vacation without a pager for years . . . "

He pauses. "In the meantime, last but not least is Dwayne," says Kurt, gesturing to the oldest person at the table. He's not only dressed differently than everyone, his laptop is different too—it's a beast with a massive screen. "He's a senior database and storage engineer from Ops and was the person who brought Brent into this group. They conspire all the time to find better ways to manage infrastructure."

Maxine smiles. To most people on the Phoenix Project, centralized Ops are merely the people on the other side of a ticket. They're the people everyone is always complaining about. But clearly Kurt and this motley crew have a different way of working, bypassing the normal organizational lines of communication, however informal.

Dwayne reaches across the table, extending his hand. "Great to meet you, Maxine."

Maxine realizes that Dwayne is wearing an actual bowling shirt, complete with his initials sewn on it, "DM," and a faded mustard stain right next to them.

"Dwayne has been trying to get automation initiatives going for years, but he and Brent always get shot down," Kurt continues. "So, they've been helping Adam build up our own infrastructure instead. He knows almost everyone in Operations, and he can usually get them to do anything. Like earlier this week when we needed a firewall port opened between two internal networks. Dwayne made that happen."

"All in a day's work," Dwayne says with a friendly smile. "But to be fair, Kurt is really the master of getting the impossible done . . . I'm just learning from him!"

Maxine is certain Dwayne is exaggerating. Dwayne looks like he's in his mid-fifties. Just how much could he learn from a young guy like Kurt?

Kurt leans back in his chair, arms spread out. "Maxine, your work cracking the code of the Phoenix builds has impressed us all. We are in awe of the technical and social skills you displayed to successfully hunt down almost all the pieces of the environment, which required incredible perseverance, focus, and never taking 'no' for an answer!"

Confused, Maxine looks around, but she sees everyone nodding at her, genuinely impressed at her work. Kurt continues, "We invite you to be a part of the inner-circle of the 'Rebellion.' We're recruiting the best and brightest engineers in the organization, training and preparing in secret for the right time to overthrow the Empire, the ancient, powerful, and unjust order that definitely needs to be toppled."

Everyone chuckles, and Cranky Dave raises his glass, shouting with a laugh, "To the overthrow of the Empire!"

Confused, Maxine looks around the table. These are people from Dev, QA, Security, and Ops—a very unlikely group of people to be socializing, let alone working together. And she notices that everyone has a small sticker of the Rebel Alliance from Star Wars on their laptop, just like the X-Wing pilots wore on their helmets. She grins at their subtle but subversive badges of solidarity.

Seeing Maxine toast with an empty hand, Kurt jumps up. "What do you want to drink?"

"A pinot noir, please."

Kurt nods and heads toward the bar, but before he can take three steps, a tall and somewhat overweight man with graying hair walks up to him and gives him a big hug. In a loud and boisterous voice, he says, "Kurt! Good seeing you again, my young friend. What do you need?"

Noting the attention that Kurt's group gets from the bar staff, Maxine guesses they must come here often. She smiles. For the first time since her move to the Phoenix Project, she feels like she's in the company of kindred spirits.

"Who are you people? Why are you all are here? What are you possibly trying to achieve?" she asks quickly, while Kurt is at the bar.

Everyone laughs. Dwayne says, "As you know, we're a huge Kumquat database shop, which is what I cut my teeth on. I want to migrate us to MySQL and open-source databases wherever we can, because I'm tired of

sending millions of dollars each year to an abusive vendor. We're figuring out how to engineer our way there."

Looking around, he says to everyone, "Other companies have done this already. I think that anyone who is still paying Kumquat database maintenance fees is simply too dumb to migrate off it."

Maxine nods in approval. "Smart thinking! We've saved millions of dollars in my old group doing this, which we can now spend on innovation and other things the business needs. And it's been fun. But why this crusade for open-source software?"

"I'll tell you why," Adam says. "For almost five years, back when I was in Operations, I had a team that kept getting pager alerts at two in the morning for some middleware we used. In almost every case, it was because of their database driver. I was the guy who had to generate a binary driver patch! After all that work, the problems started happening again six months later, because when the vendor released their patches, they didn't integrate my fixes into their code. Next thing you know, we're all up at two a.m. doing the same thing over again."

Maxine is impressed. *Adam has great kung fu too. And so does everyone else here.*

Cranky Dave frowns. "I've been at Parts Unlimited for almost five years, and I can't believe how the bureaucracy and silos have taken over. You can't do anything without first convincing a bunch of steering committees and architects or having to fill out a bunch of forms or work with three or four different teams who each have their own priorities. Everything is by committee. No one can make decisions, and implementing even the smallest thing seems to require consensus from everyone. Almost everything I need to do, I have to go up two levels, over two levels, and down two levels just to talk with a fellow engineer!"

"The Square!" cries out Adam, and everyone laughs.

Dwayne chimes in. "In Ops, we often have to do the return path—up, over, down, and then back up, over, and down before two engineers can finally work together to get something done."

"I want to bring back the days when a developer could actually create value for someone who cares, easily and quickly," Cranky Dave says. "I want to build and maintain something for the long haul, instead of shipping the 'feature of the day' and dragging all this technical debt around."

Cranky Dave is on a roll. "This company is run by a bunch of executives with no clue about technology, and project managers who want us to follow a bunch of arcane processes. I'll scream at the next one who wants me to write a Product Requirements Document."

"The PRD!" everyone shouts, laughing. Maxine raises her eyebrows. Those made sense decades ago, when you wanted written justification before you wasted a bunch of developers' time. But now you can prototype most features in the time it takes to even write one page of a PRD. One team can now build things that used to require hundreds of people.

Kurt sits next to Maxine, putting a glass of red wine in front of her. "We're like the redshirts in *Star Trek* who actually get the real work done."

"I was literally thinking that earlier," Maxine says, smiling.

"Right? You've seen firsthand the reality bubble the bridge crew is in," Kurt says. "They know the Phoenix Project is important, and yet they couldn't have come up with a worse way to organize everyone to achieve it. They outsourced IT, brought it back in, outsourced one piece, maybe two pieces, shuffled them around…In many areas, we're organized as if we're still outsourced, and nothing can get done without permission from three or four levels of management."

"Kurt's right," says Cranky Dave. "We're just another cost center, little cogs in a big machine that can be easily outsourced to some random corner of the globe. We're viewed as replaceable and fungible."

"That's why I'm here, Maxine," Shannon says. "We could build a world-class technology organization and create an engineering culture. That's how we survive and innovate for our customers. And my dream is that *everyone* is the custodian of company data. It's not just the job of one department.

"In Steve's Town Hall, he talked about how we're being disrupted and how we need to compete with the e-commerce giants," she says. "Well, we can only win by innovating and understanding our customers, which we can only do by mastering data. I think the capabilities we're building are the future of the company."

Everyone cheers and hoists their glasses.

After everyone is done toasting each other, Dwayne turns to Kurt and asks, "So, how did the meeting go with your boss? You said you pitched William on funding an automated testing pilot."

Everyone leans in.

"You know, I really thought he was going to go for it. I had testimonials from two of the Dev managers and a product owner about how great it would be. One of them had this great line: 'Without automated testing, the more code we write, the more money it takes for us to test.' Ha! I really thought that would scare the pants off of William!" Maxine can feel the mood deflate around the table.

"Don't keep us in suspense, Kurt. What did he say?" prompts Dwayne.

"'Son, let me explain something to you,'" Kurt says, in a shockingly good impersonation of William. "'You're young. You clearly don't understand how this game works. We're QA. We protect the organization from developers. It sounds to me like you've been hanging around too many of them. Do not trust them. Do not get chummy with them. You give developers an inch, and they'll take a mile.'"

Maxine laughs at Kurt's uncanny impression.

"'Son, you're a pretty good QA manager with a half-million-dollar budget.'" Kurt's on a roll. "'If you do your job well, you can be like me with a three-million-dollar budget. And if I do my job well, then I'll get promoted and have a $20 million budget. You go around automating your QA, your budget shrinks instead of grows. I'm not saying you're stupid, son, but you sure don't seem to understand how this game works.'"

Maxine laughs with everyone else. She is sure Kurt is exaggerating.

"William is like a union leader, not a business leader," Shannon says. "He only cares about growing his union membership dues, not about what's right for the business. You see the same thing inside Ops and even Infosec."

A frown crosses Dwayne's genial face. "Trust me, it's way, way worse in Ops. At least Development is seen as a profit center. In Ops, we're a cost center. The only way to fund infrastructure is through new projects. If you don't find new funding sources, you're screwed. And if you don't spend your whole budget, they'll take the money away from you next year."

"Ah, the project funding model… Another big problem here at Parts Unlimited…" Kurt says, as everyone groans in agreement.

"So, what's your plan now, Kurt?" Dwayne asks.

"Don't worry, Dwayne. I've got another plan," Kurt says, confidently. "We're going to lie low and keep doing what we're doing, looking for new potential customers and recruits. We keep our eyes and ears open for opportunities to get in the game."

"Oh, that's a great plan, Kurt," Dwayne says, rolling his eyes. "We hang out at a bar, complain, and drink beer. Brilliant."

Dwayne leans over to Maxine, explaining, "It's actually not that crazy. It's like in that movie *Brazil*, where the number-one fugitive is the rogue air conditioner repairman who fixes people's air conditioners because Central Services never gets around to it. That's us. We're always on the lookout for places we can help. It's a great way to make friends and find potential new recruits for the Rebellion."

"What?" she says in disbelief. "That can't work, can it?"

"Well, it's how we got you here, isn't it?" Dwayne says with a big smile.

"I'm working all the angles," Kurt continues. "I'm even thinking about asking William if I can have a meeting with him and his boss, Chris. I'd tell William that it's really important to me that Chris hears my proposal and that I want him there."

Wow, Maxine thinks. *That's pretty gutsy, maybe savvy, and probably fatal.*

"I'll keep you posted," Kurt says. "Okay, who has new information or intelligence to share?"

Shannon updates everyone on a nascent data analytics group in Marketing she's been working with and how she's setting up a meeting between them and Kurt. "They're working on a bunch of projects to increase customer promotion conversion rates, and boy, they really need help. They're not even using version control! They're struggling with basic data engineering problems, and they're still trying to get what they need from the data warehouse people," she says, visibly bothered by their suffering. Kurt quickly pulls out an org chart on his laptop.

He asks her, "Another data analytics project? Who's funding it? How much budget do they have? Who's leading it?" As she talks, he takes notes.

When it's his turn, Dwayne says, "I've got bad news. The Phoenix release caught everyone in Ops flatfooted—no one had it on their radar until last week. No budget was assigned to support it. Everyone's scrambling to find enough compute and storage infrastructure. This is the biggest launch we've done in almost twenty years, and everything we need, we don't have enough of. It's bad."

"Holy shit," says Adam.

"Yep," Dwayne says. "I've been trying to tell everyone for months, but no one cared. Well, now they do, and everyone's dropping everything to

support the Phoenix launch. Today, I heard someone trying to work with procurement so they can break the rules and order outside of the annual ordering process."

Despite the crisis, bean counters are still bean counters, Maxine thinks.

"Everyone is still scrambling to get environments ready for the release tomorrow," Dwayne says. "No one has any build specifications that Dev and Ops both agree on. I gave them the ones we wrote, and they pounced on them and started using them right away. But still, this release is going to go real bad, real fast."

"I think you're right," Maxine says. "I'm really, really good at this stuff, and I spent nearly a week trying to get a Phoenix build going. If it weren't for the environment that Kurt gave me, I'd still be at square one. With the release team only starting today and the launch tomorrow, they are in big trouble."

Kurt leans forward, a serious look on his face. "Tell me more."

Suddenly, Maxine realizes why she was invited and that Kurt is no dummy after all.

Over the next twenty minutes, Maxine describes her experiences, reading from her work diary, which she can access from her phone. She mentally kicks herself again for not bringing her laptop. Everyone takes notes, especially Brent when he returns. He and Adam pepper her with questions as if she were a captured secret agent being debriefed by the CIA. Everyone's interested in how she was able to piece together the Phoenix build puzzle faster than anyone else had done. They ask lots of questions about who she talked to, what teams they were on, where she got stuck, and so forth.

"That's really impressive, Maxine," says Cranky Dave. "Years ago I put together a build server that my team could use on a daily basis. But that was when Phoenix only had two teams; now we have over twenty. The build team is completely out of their league, with people who, I'm sorry to say, are the people who didn't have enough experience to be application developers."

Adam says, "We're really close now. I think we're down to just one missing signed certificate for the payment processing service."

"He's right," Brent says. "Maxine, can you show me the build logs? I bet we can create that certificates ourselves—it wouldn't actually be valid, but it would be good enough for a Dev or Test environment."

Maxine curses, mentally picturing her laptop still on her desk. "I can show you first thing tomorrow," she sighs.

"This is great, people. Here's what we still need: we need an automated way to create environments and perform code builds," Kurt says, counting off on his fingers. "We need some way to automate those tests and some automated way to get those builds deployed into production. We need builds so that developers can actually do their work.

"So, who's willing to volunteer some of their time to help Maxine get those Phoenix builds going?" Kurt asks. To Maxine's surprise, all hands shoot up.

"Maxine, would you be able to lead this effort, with the help of any or all of these willing and talented volunteers?" Kurt asks.

Maxine is overwhelmed by the sudden support of all these people. Last week, she was unable to get help from anyone and was thinking about interviewing at other places. Suddenly, she's not so sure.

She takes a moment to collect herself and says, "Yes, I'd love to. Thank you, everyone. I look forward to working with you all."

Maxine is excited. She's genuinely amazed at what this group has been doing and that she's been chosen to help. *I've finally found my tribe,* she thinks. *And* this *is what an effective network is all about—when you can assemble a group of motivated people to solve a big problem, even though the team looks nothing like the official org chart*.

I'm pretty sure I'll learn and achieve more with this group then I would by having lunch with Sarah, she thinks. She wonders if she's being small-minded and petty. She still wonders if she should take the meeting or just wait for Sarah to forget about her.

"Excellent! Let me know if you need anything from me," Kurt says to the table. To Maxine, he says, "We try to meet every week. We typically have only two agenda items. First, we share intelligence on who needs help and other people to potentially recruit. After that, we usually share about something we've learned lately or new technologies that we think could change the game here at Parts Unlimited. I propose we add a third agenda item, which is discussing the progress of Phoenix builds, yes?"

Everyone nods.

Kurt looks at his watch. "Folks, one more thing before we adjourn. I'm starting a betting pool on when the release team will have the Phoenix application successfully running in production."

The most optimistic bet comes from Cranky Dave, who guesses Saturday at two a.m., fully eight hours after the deployment starts. Most bets are scattered between three and nine a.m., with Maxine betting six a.m.

"After all," she says, "the in-store point of sales systems need to be up by eight on Saturday morning."

To everyone's surprise, Dwayne bets Sunday evening, "You people have no idea how unprepared we really are for this release—this one will go down in the record books."

From: Alan Perez (Operating Partner, Wayne-Yokohama Equity Partners)
To: Dick Landry (CFO, Parts Unlimited), Sarah Moulton (SVP of Retail Operations)
Cc: Steve Masters (CEO, Parts Unlimited), Bob Strauss (Board Chair, Parts Unlimited)
Date: 3:15 p.m., September 11
Subject: Maximizing Shareholder Value **CONFIDENTIAL**

Sarah and Dick,

Thanks for the call today, and for walking me through the strategy and the Phoenix Project. I agree that an omni-channel strategy is required for any retailer to survive these days, especially given the e-commerce threat. And selling products manufactured in-house with low cost of sales is intriguing.

However, I'm concerned at how much cash you've diverted from Manufacturing ($20MM) to invest in Retail over the last three years, with no obvious return. The question becomes what return you could have gotten if this were invested elsewhere in the business or just returned to shareholders. As of right now, investing in lottery tickets would have made more economic sense.

Stories about innovation and omni-channel are nice, but the board needs more than stories and PowerPoint slides.

Good luck with the Phoenix release tomorrow. I know a lot rides on it.

—Alan

CHAPTER 6

• *Friday, September 12*

Friday goes by in a blur for Maxine as the emergency release preparations continue. She sees endless mayhem as Dev, QA, and Ops try to line up hundreds of moving pieces for the deployment. *Dwayne was right*, she thinks. And it's too late to change her bet to Sunday in the betting pool.

At five p.m. the release starts on schedule. There are rumors of last-ditch attempts to call it off, because William, Chris, and Bill are nowhere to be seen. These hopes are crushed when an email comes out from Sarah and Steve, making it very clear that the release is to proceed as scheduled.

Maxine is still at the office at ten that evening. By now, there's a sense of genuine panic that things are going very, very wrong. So spectacularly wrong that even Dwayne, who was the most pessimistic in the Phoenix release betting pool, mutters to Maxine, "This is going worse than I thought it would."

That's when Maxine becomes genuinely frightened.

By midnight, it's clear that a database migration is going to take five hours to complete instead of five minutes, with no way to stop it or restart it. Maxine tries to be helpful, but she isn't familiar enough with the Phoenix systems to know where she would be the most useful.

In contrast, Brent is being pulled every which way, needed for just about every problem, from the huge database meltdown in progress to helping people fix their configuration files. Seeing this, Maxine organizes a team to play goalie, protecting Brent from interruptions and fielding problems that don't require him.

Maxine notices something else. There must be two hundred people responsible for some portion of the release, but for most of them, it's only about five minutes of work. So, they have to wait around for hours to perform their little part in this excruciatingly long, complex, and dangerous operation. The rest of their time is spent watching and . . . waiting.

Even in the middle of this crisis, people are just sitting around, waiting.

By two a.m. everyone realizes there is a very real risk that they are going to break every point-of-sale register in every one of the nearly

thousand stores, knocking Parts Unlimited back into the Stone Age. And with all the promotion that Marketing has been doing, the stores will be filled with angry customers unable to buy what they were promised.

Brent asks her to join a SWAT team to figure out how to speed up the database queries, still nearly a thousand times slower than they need to be in order to handle the expected load when stores open up later that morning.

For hours, she works with a bunch of Phoenix developers and Ops DBAs with her IDE and browser open. They are stunned when they discover that clicking the product category drop-down box floods the database with 8,000 SQL queries.

They are still working on fixing this when Wes pokes his head in the room, "Brent, we've got a problem."

"I'm busy, Wes," Brent replies, not even looking up from his laptop. "No, this is serious," Wes says. "The prices have disappeared from at least half of our products on the e-commerce site and mobile apps. Where the price should be displayed, either nothing shows up or it says 'null.' Screenshots are in the #launch channel."

Maxine blanches, pulling up the screenshot. *This is much more serious than slow database queries*, she thinks.

"Dammit, I bet it's another bad upload from the pricing team," Brent says after staring at his screen for several moments. Maxine leans over as Brent pulls up various administrative screens and product tables—some are inside of Phoenix and others on systems she doesn't recognize.

Maxine takes notes as Brent pulls up log files, runs SQL queries against a production database, pulls up more tables in various applications… Only when he opens up a terminal window and logs into a server does Maxine ask, "What are you doing now?"

"I need to inspect the CSV file that they uploaded into the app," he says. "I think I can find one in the temporary directory on one of the application servers." Maxine nods.

When Brent squints at his screen, Maxine does as well. It's a comma-separated text file with column names in the first row—product SKU, wholesale price, list price, sale promotion price, promotion start date… "It looks fine," Brent mutters.

Maxine agrees. She says, "Can you copy that file into the chat room? I'd like to take a look at it."

"Good idea," he says. She imports it into Excel and several other of her favorite tools. It looks fine.

While Wes tries to get one of the development managers on the phone, Brent tries to figure out what is going wrong. It's almost thirty minutes later when he curses. "I can't believe it. It's a BOM!"

Seeing Maxine's confused expression, he says, "A byte-order mark!"

"No way," mutters Maxine, pulling up the file again, this time in a binary file editor. She stares at her screen, stunned that she missed it. A BOM is an invisible first character that some programs put in a CSV file to indicate whether it's big-endian or little-endian. She's been bitten by this before.

Years ago, a colleague gave her a file exported from the SPSS statistical analysis application, and she spent half a day trying to figure out why her application couldn't load it as expected. She finally discovered that the file had a BOM, which got interpreted as part of the first column name, which caused all her programs to fail. *Which is almost certainly what is happening here*, she thinks.

Any intellectual satisfaction she feels at understanding this particular puzzle quickly disappears. She asks Brent, "This has happened before?"

"You have no idea," Brent says, rolling his eyes. "Different problem every time, depending on who generated the file. The most common problems lately are zero-length files, or files with no rows in them. And it's not just the pricing team—we have data problems like this all over the place."

Maxine is appalled. The first thing she would have done right away is write some automated tests to ensure that all input files are correctly formed before they allow them to corrupt their production database, and that the correct number of rows are actually in the file.

"Let me guess. You're the only one who knows how to correct these bad uploads?" Maxine asks.

"Yep," she hears Wes say from behind her. "All roads lead to Brent." Maxine jots down more notes, determined to investigate this and do something about it later.

It's almost two hours later before the pricing tables are corrected. Because of what Brent said, Maxine double-checks the file and is certain that it's

missing a significant number of product entries. And because the pricing team wasn't part of the release, no one knows how to get a hold of them in the middle of the night (or early morning as is seems to be). Maxine adds some more things to her list of things that she'll insist on building so that this won't happen again.

At seven a.m., Maxine rejoins the database team. They're still working on speeding up queries—but it's too late. An announcement is made that stores are beginning to open on the East Coast.

The Phoenix release is still nowhere near complete. "We're fourteen hours into the launch, and the missile is still stuck in the tube," Dwayne says glumly.

Maxine doesn't know whether to laugh, smirk, or throw up—when missiles are stuck in the launch tube, it's a very dangerous scenario, because at that point the missile is already armed and too dangerous to approach.

At eight a.m., they are still hours away from having a working point-of-sale system. Sarah and her team are forced to train every store manager on how to use carbon paper imprints, and some stores are forced to only accept cash or personal checks.

For Maxine, the rest of Saturday goes by in a blur. She's unable to go home. The Phoenix rollout was more than just a spectacular outage…it was the most amazing example of production data loss Maxine has ever seen.

Somehow, they managed to corrupt incoming customer orders. Tens of thousands of customer orders were lost, and an equal number of customer orders were somehow duplicated—sometimes three or four times. Hundreds of order administrators and accountants were mobilized, reconciling database entries against paper order slips being emailed or faxed from stores.

Shannon texts everyone in the Rebellion, horrified that boxes of customer credit card numbers are being transmitted in the clear—but in the grand scheme of things, it's just another blip in the Phoenix disaster.

At three p.m., Kurt texts everyone:

> Not to put light on this big pile of suck, but Dwayne wins the betting pool. Congratulations, Dwayne.

Dwayne replies:

Not worth it! FUUUUUUUUUUUU . . .

He posts an image of a burning tire fire.

By Saturday night, Maxine finally manages to go home and sleep for six hours before coming back to the office. *Dwayne was right, this will go down in the record books*, she thinks glumly.

On Monday morning, Maxine is shocked to see her reflection in a mirror. She looks like crap, just like everyone around her—bags under her eyes, hair stringy. Long gone are her carefully pressed blazers. Now it's jeans and a wrinkled jacket to cover up a stain on her equally wrinkled blouse. Today she doesn't look classy. Like everyone else, she looks like she's recovering from a hangover, having slept in her outfit from the night before.

Since Saturday morning, their e-commerce site has been continually crashing under the unprecedented levels of customer traffic. In a status update meeting, Sarah crowed about what a great job Marketing did promoting Phoenix, then demanded that IT pull their weight.

"She's unbelievable," Shannon mutters. "She created this whole disaster! Is anyone ever going to call her on this?" Maxine just shrugs.

The carnage is unbelievable. Most of the in-store systems are still down—not just the point-of-sale registers, but nearly all of the back-office applications that support the in-store staff.

For reasons that continue to mystify everyone, even the corporate website and email servers are having problems, further hampering their ability to get critical information to people who need it—not everyone has access to the developer chat rooms.

In situations like this, technology failures cascade through the organization, like water flooding through a sinking submarine.

Trying to stay alert, Maxine goes to get more coffee from the kitchen. Dwayne's there doing the same thing. They nod at each other, and he says, "Did you hear we have hundreds of people who can't even get into their buildings because their keycards won't work?"

"What?!" Maxine exclaims, exhausted but laughing. She says, "I was just talking to someone who's trying to figure out why a bunch of batch

jobs aren't running. He's even saying payroll might be delayed again—umm, I'll leave that to other people to fix," she concludes with a small laugh.

"Huh," he muses. "I wonder if we managed to knock out an interface to an HR application. That might explain these strange errors. We managed to screw up everything else."

All day during the recovery efforts, she hears questions like: Why are all those transactions failing? Where are they failing? How did it get into that state? Of the three ideas that might fix the problem, which one should we try? Will it make it worse? We think we fixed it, but is it really fixed?

Once again, Maxine's sensibilities are offended by how entangled all these systems are with each other. It's so difficult to understand any part of the system in isolation.

At times, it was difficult not to feel panicked. Earlier in the day, it looked like the Parts Unlimited e-commerce site was being attacked by an external party actively stealing credit card numbers. It took over an hour for Shannon and the security team to send out an email concluding that it was an application error—if someone refreshed the shopping cart at the wrong time, the full credit card number and three-digit CVV code of a random customer was displayed.

The good news was that it wasn't an external hack. The bad news was that it was a genuine cardholder data exposure event and likely another reason to be front-page news. All the attention and ridicule exploding on social media only added to everyone's stress.

Taking a break, Maxine walks back to her desk. She sees the developer who was so unconcerned with the release last week. He's wearing fresh clothes and appears to be well-rested.

"Rough weekend, I'm guessing?" he says to Maxine.

Maxine stares at him, speechless. He's still working on features for the next release. The only big change for him is that all his meetings have been canceled because most people have been sucked into the Phoenix crisis.

He turns back to his screen to work on his piece of the puzzle, not caring that none of the pieces actually fit together. Or that the entire puzzle has caught on fire over the weekend, along with the house and the entire neighborhood.

From: Alan Perez (Operating Partner, Wayne-Yokohama Equity Partners)
To: Dick Landry (CFO), Sarah Moulton (SVP of Retail Operations)
Cc: Steve Masters (CEO), Bob Strauss (Board Chair)
Date: 8:15 a.m., September 15
Subject: Phoenix Release **CONFIDENTIAL**

Sarah and Dick,

I've been reading the news headlines about the Phoenix release. Not a great start. Again, I question whether software is a competency Parts Unlimited can create. Maybe we explore outsourcing IT?

Sarah, you mentioned the large number of developers you've contracted to help. How long until they are fully contributing? When you grow a sales team, it takes time for new salespeople to carry full quota capacity. Can new developers really be onboarded fast enough to make a difference? Or are we just throwing good money after bad?

Sincerely, Alan

From: Sarah Moulton (SVP, Retail Operations)
To: All IT Employees
Cc: All Company Executives
Date: 10:15 a.m., September 15
Subject: New production change policy

Thank you for all your hard work helping deliver Phoenix to our customers. This is a badly needed step for us to regain parity in the marketplace.

However, due to the harm that we did to our customers because of unanticipated problems caused by poor judgment exercised by certain

members of the IT organization, all production changes must be approved by me, as well as Chris Allers and Bill Palmer.

Changes made without approval will result in disciplinary action.

Thank you, Sarah Moulton

Maxine reads the email from Sarah. There's a new, maybe even sinister, dynamic creeping into the Phoenix Project. In each of the outage calls and crisis management meetings, senior leaders seem to be going out of their way to posture about how they did their job but other people didn't do *their* jobs, sometimes subtly, sometimes very blatantly.

While the redshirts battle to contain the raging engine fire that is threatening the entire ship, the bridge officers continue to cover their asses, Maxine observes. Some are even using the disaster to their political advantage, often to punish individual engineers or entire departments for supposed dereliction of duty.

Apparently, no one in IT leadership is safe—Maxine hears whispers that both Chris and Bill, as the heads of Dev and Ops, are in jeopardy of being fired, and there are rumors of all of IT being outsourced again. However, most believe William, as head of QA, is most likely to be axed.

Which is bullshit, thinks Maxine. *William was assigned to head up the release team less than twenty-four hours before the release! No one can get fired for trying to avert a disaster, right?*

"It's like the TV show *Survivor,*" says Shannon. "All the technology executives are just trying to last one more episode. Everyone is freaking out. Steve has been demoted, and Sarah is trying to convince everyone that she can save the company."

Later that afternoon, Brent invites Maxine to join a meeting. "We've got nearly sixty thousand erroneous and/or duplicate orders in the database, and we've got to fix them so that the finance people can get accurate revenue reports."

Maxine helps the group wrangle the problem for an hour. At the end, once they find a solution, one of the Marketing managers says, "This is above my paygrade. Sarah is super-sensitive about changes right now. I've got to get her approval."

Ah, the Square in action, just like Cranky Dave described. But now, decisions that might have needed only to go "up and over one" now have to go "up and over two." Now, all product managers need to run everything by Sarah. Someone mutters, "Don't hold your breath—she never responds right away."

Great, Maxine thinks. *Sarah has effectively paralyzed everyone in this room even further*.

Throughout the day, all decisions and escalations quickly grind to a standstill, even for emergencies, which Maxine didn't expect. She discovers why: every manager insists on being a part of the communication plan. Why? They want to hear any bad news first, so they don't appear out of touch and can massage any messages up the chain.

Maxine is sharing this observation with Kurt when his phone buzzes. Seeing his sour expression, she asks, "What's up?"

"It's Sarah," he says. "She says she's getting conflicting information from Wes and me about the corrupted order data. I need to spend thirty minutes explaining it to her when I've got two actual emergencies going on."

Kurt storms off before she can even wish him good luck. Maxine shakes her head. The lack of trust and too much information flowing around is causing things to go slower and slower.

On Tuesday, Maxine joins a meeting led by Wes about more mysterious, intermittent outages for both the e-commerce site and the point-of-sale systems.

Sarah has been sending out emails, sometimes in all caps, reminding people how important this is. But everyone already knows how important this is—processing orders is one of the most important functions for any retailer.

The room is almost empty, even though this is a Sev 1 outage.

Apparently, everyone has had to go home sick. The Phoenix release forced people to work long hours together in close proximity all day and night, and with little sleep. Now everyone is dropping like flies. Of the people needed on this call, no one is healthy enough to be in the office. In fact, only two people are healthy enough to even be on the conference line.

Maxine looks up when she hears Sarah shouting, "What can you do about this? Who can fix this? Our store managers need our help! Don't people realize how important this is?"

Maxine stares at Sarah in disbelief, noting that she looks tired, not her usual immaculate self. Even Sarah isn't escaping the Phoenix carnage completely unscathed, despite her Teflon-like ability to avoid getting blamed for nearly anything in her three-year tenure at Parts Unlimited.

Wes throws up his hands. "What can we do about it? Nothing. The entire application support team is out sick. Brent just went home sick. The DBAs are out sick. Even though we've got the supremely competent Maxine here, she's like me—we don't know enough about the service to do anything except reboot the systems, which is what the support teams are already doing."

Maxine sees that Wes is sick too—he's congested and looks terrible. Bags underneath his red eyes, hoarse voice . . . she suddenly wonders if she looks as bad as he does.

"This is not acceptable, Wes," says Sarah. "The business depends on us. The store managers depend on us. We need to do something!"

"Well, these were the risks we warned you about when you proposed proceeding with the Phoenix launch—but you emailed saying that we 'need to break some eggs to make omelets,' right? We're doing everything we can, but unless you want to help reboot some servers, I'm telling you there's nothing we can do."

Wes continues, "But here's something that we should talk about: How do we keep our people healthy enough so they can actually do their jobs? And how do we keep them happy enough so they don't quit? Chris says two of his key engineers quit in the last week. I've lost two people on the Ops side too, and there's a good chance I may lose three more. Who knows how many more are actively looking?

"And when that happens, we will truly be up shit creek, because then we'll have empty meetings like this all the time," Wes says with a half-hearted laugh that turns into coughs.

He grabs his laptop and starts walking out the door. Before he leaves, he says, "Sarah, I know you think it's strange that we have no one left on the bench to solve this important problem, but that's the way it is. If you want to help, learn to be a doctor or learn some middleware. In the meantime, just stay out the way because we're doing our best."

Maxine likes the way Wes rolls—he's fearless and he always says what he thinks.

She makes a mental note to ask the Rebellion about recruiting Wes.

Thinking about the Rebellion, she realizes how important that group is. To her, it's a beacon of hope. Maxine knows she may be manic and loopy from lack of sleep, but the Rebellion has assembled some of the best engineers in the company. And they could liberate everybody from . . . from . . . all of this.

We need to keep the Rebellion together and keep this important work going, she thinks.

She texts Kurt right away:

> No matter what, we cannot cancel our Dockside meeting on Thursday.

His reply shows up right away.

> Great minds think alike. In fact, I have a surprise for everyone. See you in two days!

By Thursday, things have stabilized substantially. The most glaring defects and performance problems in Phoenix have been fixed. And it helps that customer traffic is way, way down. Who wants to go to a store or website that can't take orders? The result is that it's no longer necessary for everyone to work all night. Maxine slept in until ten this morning. As she was driving into work, she realized how much she was looking forward to the Dockside meeting that evening.

As promised, Kurt texted everyone in the Rebellion:

> I'll be a little late. Dwayne and Maxine, please run the standard agenda, including the Phoenix environment build. I will be bringing a very special guest.

Maxine is pretty sure everyone will be there tonight.

But despite getting some sleep, she doesn't feel well. She desperately hopes she is not getting whatever illness decimated her fellow co-workers. Despite that, she is very glad to be working on the Phoenix builds again.

That evening, when she arrives at the Dockside, Maxine's excited to see everyone. She wants to find out how to get a Rebellion sticker for her laptop and to trade war stories. She's surprised to see that everyone looks angry and dejected.

Throwing her jacket over the back of a chair, she says cheerily, "Hi, everyone! What's got everyone so grouchy?"

Dwayne looks at her. "Read the email that was just sent out. They fired William."

From: Chris Allers (VP Development)
To: All IT Employees
Date: 4:58 p.m., September 18
Subject: Personnel changes

Effective immediately, Peter Kilpatrick (Front-End Dev Manager) will be leaving the company, and William Mason (QA Director) will be on a leave of absence. We especially appreciate all their contributions.

Please direct all front-end Dev emails to Randy and all QA-related emails to me.

Thank you, Chris

Maxine slumps as she reads the message. The witch hunt has begun. Adam shakes his head angrily. "I wasn't a huge fan of William," he says, "but to blame him for everything is wrong."

In Chris' email there's no mention about his own culpability in the Phoenix disaster. And even though Maxine doesn't believe in punishment or scapegoating, it's doubly unfair that all the blame is being put on the technology organization, and no one from the business or product side is being held accountable.

Cranky Dave looks up from his phone, disgusted. "Ditto for Peter—he was just doing what the business managers demanded. What a complete shit show."

"This is so wrong," Shannon mutters. "I don't suppose it would help to write a petition or anything, right? You know, lodge our protest about their firing?"

Adam says, "No one who matters is being held responsible! We should…"

He suddenly stops talking, staring slack-jawed at something behind Maxine. "Holy shit…" he finally says. Everyone next to Adam is also looking shocked at whatever is behind her.

Maxine turns around and sees Kurt walking through the entrance.

Next to him is Kirsten, the director of project management.

"My God," Maxine hears Adam say. He looks frightened, closing his laptop and standing up, as if he's going to flee the scene.

"Oh, for Chrissakes, sit back down, Adam," says Maxine. "This isn't like the secret police showing up. Not one of us has done anything wrong—have some dignity."

Cranky Dave laughs nervously, but like everyone else, he's already closed his laptop, as if he has something to hide.

Kirsten is wearing a fancy blazer, two steps up from Maxine's usual casual business garb and a full four steps up from the hoodies, T-shirts, and bowling shirts worn by the other engineers around the table. People in the bar are staring, clearly wondering who invited the management suit here.

Maxine knows that she looks slightly out of place at the Dockside, but wow, Kirsten looks *way* out of place, like she was on the way to an event for senior law partners but had a flat tire while driving by with a dead cell phone and had to come in to find help.

Looking around, Kurt smiles and says, "For those of you who don't know Kirsten, she leads Project Management, which is undoubtedly the most trusted organization at Parts Unlimited, despite their association with us technology people." Kurt laughs. "All of the most important company initiatives go through Kirsten and her project management clerics, and she routinely briefs Dick Landry, our CFO, on how they're going."

This is true, Maxine thinks. Kirsten is truly the high priestess of order and discipline. She assigns the score of red, yellow, or green to each major initiative of the organization, which can have career-catapulting or career-ending consequences for the people involved. Besides Sarah and the VP of sales, Kirsten is the person most mentioned by the CFO in his Town Halls.

Sitting, Kirsten pours herself a beer from the pitcher on the table and then pours a glass for Kurt. Kurt introduces everyone to Kirsten and then gestures at Maxine, "Maxine is the latest addition to our elite group of rebels. She was exiled to the Phoenix Project as punishment for the payroll outage, and of course, her vast talents have been completely wasted ever since. That is, until we recruited her to help overthrow the uncaring, ancient, powerful existing order…oh, um…" Kurt suddenly looks embarrassed, realizing Kirsten is part of that order. "Present company excepted, of course," he finishes.

Kirsten merely raises her glass in response.

Kurt continues, "It turns out that Maxine, in her boredom and search for meaning, began working on creating repeatable Phoenix builds, something that has eluded the Phoenix teams for well over a year. We believe in many great and virtuous things, but one thing we all agree on is that getting builds going again is one of the most urgent and important engineering practices we need right now. Once we get continuous builds going, we enable automated testing. We get automated testing, we can make changes quicker, with more confidence, and without having to depend on hundreds of hours of manual testing. And that, I believe, is the critical first step for how we can deliver better value, safer, faster, and happier.

"Without continuous builds, we are like a car manufacturer without an assembly line, where anyone can do whatever they want, independent of the plant goals," he continues. "We need to discover problems only when we are in the build or testing process, not during deployment or production.

"I've wanted to own this for a year, but my boss, uh, rather, my recently departed ex-boss, didn't think it mattered. So, I've been taking people off my team to work on it in secret and seeking out the best engineers in the company who are willing and able to help. And Maxine has been a tremendous help in an amazingly short amount of time," he adds.

Kurt pauses. "Uh, let's all raise a glass to William—he and I had our differences, but he certainly didn't deserve to take the blame for the entire Phoenix fiasco."

Maxine raises her glass, as everyone else does the same. She takes the time to clink glasses with everyone around the table.

Looking at Kirsten, she says, "It sounds crazy, Kirsten, but I really think this group can make a big difference. I've seen developers wait for

months to get a Dev environment. The lack of environments and centralized builds slow us down in countless ways. In fact, most Dev teams eventually stop waiting for environments or builds and just write code in isolation, without caring whether it actually works with the system as a whole."

Maxine continues, "Look at what happened last week with the Phoenix release. Better engineering practices would have prevented so much of that. What a waste…"

"We all agree with Maxine," Cranky Dave says. "But, Kirsten, uh, what in the world are you doing here?"

Kirsten laughs. "I've long harbored a suspicion that how we manage technology at this company is not working. And it's not just the Phoenix release catastrophe. Look at all the things we need from Phoenix that are still years away on the project plan.

"Kurt has been telling me for months about the work the Rebellion is doing. But my aha moment was when Kurt pointed out that we've somehow created a system where hundreds of engineers are unable to get simple things done without an incredible amount of communication and coordination," she explains. "Sure, it's our job to safeguard the most important projects in the company. But ideally, everyone should be able to get what they need done without any help from us. Somehow, I think Project Management has turned into an army of paper pushers, being dragged into every single task because of all the dependencies.

"We track the work of nearly three hundred people working on the various parts of Phoenix. But, the real effort is even larger," she continues. "You'd think we have thirty teams of ten people, with each team able to get things done independently. But at times, it's like we have only one team of three hundred people…Or maybe three hundred teams of one. In either case, something is very wrong…"

She turns to Kurt. "What was that term you used? Watermelon projects? Green on the outside, but red on the inside? That's what every one of our IT projects is these days," Kirsten observes wryly.

She continues, "I've been here for fifteen years, and we've been playing this game of outsourcing and insourcing IT the whole time. The last time around, the CIO proclaimed that Parts Unlimited was 'no longer in the people business,' if you can believe that, and outsourced everything.

We eventually brought most of it back in-house, but everything we got back was in worse shape than ever. And we've lost the capability to do some of even the most basic things ourselves. Last year, we had to make a simple schema change for our data warehouse. We put out the request to our normal list of outsourcing partners. It took them about three weeks to get an estimate back to us. They said the work would take about ten thousand hours to complete," she says. "Before we outsourced IT, this was something we could have done in a couple of hours."

Maxine does the math in her head. From her consulting days, she knows one fully loaded engineer works about two thousand hours per year—that's forty hours per week, fifty-two weeks per year, if they don't take any vacation. She bursts out laughing. "That's five engineers working full-time for a year, just to make a database column change?! That's something I could do in fifteen minutes!"

"Yep," Kurt says, with a sad smile. "The data warehouse change requires work from two or three different outsourcers. You'd need to pull together meetings from the account managers from each of those teams. Each account manager would require a change fee and a feasibility study. It takes weeks to get all the technical people to agree upon a change plan, and even then, the tickets bounce back and forth for weeks. It takes a super-heroic effort to actually get the change made."

Dwayne laughs loudly. "You think that's bad? That's nothing! We used to have three networking switches in all of our manufacturing plants. One for internal plant operations, one for employees and guest WiFi, and one for all our equipment vendors that need to phone home to their mothership.

"A couple of years ago, probably during budgeting season, some bean counter looked at those three networking vendors and decided to consolidate them down to one switch. Sort of makes sense, right?" he continues.

"So without asking anyone, they went ahead and did it. And not just in one plant, but in a bunch of the plants. They replaced the three switches with one bigger, beefier switch, and then moved all plant traffic onto it," Dwayne says. "But what they didn't know was that they had three separate outsourcers managing the three different networks. So now all three outsourcers who used to work on their own separate switches had to work on one switch and were suddenly stepping on each other's toes all the time.

"Within a week, one of the manufacturing plants had their entire network knocked offline—absolutely nothing from inside the plant could talk with the outside world. No one could get plant scheduling information, no one could send out replenishment orders, equipment couldn't get maintenance updates…All interfaces were dead!" Dwayne continues, still clearly in awe of the scale of the outage.

"The only thing that worked was the fax machine. Everyone from every department had to wait in line to send out things like weekly production reports to management, orders for raw materials…" Dwayne says.

Maxine bursts out laughing. "I remember that—it was incredible. We had to buy some USB printers from the local office supply store for a couple of systems that couldn't connect to the network printers. It was like going back to the 1970s for almost a week."

Adam mutters from across the table, "Yeah, just like we did to the in-store systems this weekend."

Dwayne takes another drink of beer and leans back, enjoying having everyone's attention. "You're probably wondering why it took a week to restore service. Well, that entire time, no one took responsibility for what happened. All three outsourcers denied that it was them, even when we presented them the log files that clearly showed that one of the them had disabled everyone else's accounts. Apparently someone got tired of having their changes trampled on by the other two, so they just locked them out."

Everyone roars in laughter, but Maxine's jaw drops.

Dwayne continues, "That entire week all three outsourcers kept blaming each other, and the network stayed down for days. It escalated all the way to Steve. Yep. The CEO. Even after he got all the CEOs of all three outsourcers on the phone together, it still took almost twenty-four hours for the network to be restored."

As everyone jeers, Maxine says slowly, "That's so interesting. Consolidating network switches isn't inherently a bad idea. Before, three teams were able to work independently on their own networks. And when they were all put on one network switch, suddenly they were coupled together, unable to work independently, having to communicate and coordinate in order to not interfere with each other, right?"

With awe in her voice, she continues, "You know, after they got put onto one switch, I bet those teams needed to create a master schedule with

all of their work on it. And I'm even betting that they needed to bring in project managers where they probably didn't need them before.

"Holy cow," Maxine continues, on a roll. "They did it to reduce costs, but surely, in the end, it was more expensive for everyone all around. And I bet it took everyone longer to do their work, with everyone having to communicate, coordinate, get approvals, with project managers shuffling and deconflicting all the work.

"Oh, my God. It's just like the Phoenix Project!" she exclaims.

Silence falls upon the table as everyone stares at Maxine in a mix of horror and dawning realization.

"You mean everything that's wrong with the Phoenix Project we did to ourselves?" Shannon asks.

Kirsten looks rattled, brow furrowed, but says nothing. "Yes," says Maxine. "I think we did it to ourselves."

"You are correct, Maxine. You are truly on the cusp of understanding the magnitude and scale of the challenges that await you," a voice says from behind Maxine.

CHAPTER 7

• *Thursday, September 18*

The owner of the familiar voice is, to Maxine's surprise, the bartender the last time she was at the Dockside.

He sets a tray of drinks down next to Maxine and gives Kurt a friendly pat on the back. Then he turns to Kirsten, saying, "Oh, ho—if it isn't Ms. Fingle! Long time no see! Welcome to the Dockside, headquarters of the budding Rebellion."

"Holy cow," Kirsten says, staring.

"Uh, you know each other?" Kurt asks, his usual confident tone missing.

Kirsten laughs. "This is Dr. Erik Reid. You may not know this, but Steve and Dick have been trying to recruit him to serve on the board of Parts Unlimited for months. He's worked with the company for decades. In fact, Erik was part of the initial MRP rollout in the '80s, and then he helped the manufacturing plants adopt Lean principles and practices. We were one of the first companies to have an automated MRP system, and he's a genuine hero among the manufacturing ranks."

"Him?" Kurt says in disbelief, pointing his thumb toward the bartender.

Maxine is surprised too. After all, she took over continuing development and operations of the amazing homegrown MRP system years ago. She's always been impressed at how it codified not only a wonderful way of working that led to fantastic flow but also enabled continual learning, for both line workers and plant managers.

"Don't believe everything you hear," Erik says, snorting.

Maxine quickly sizes him up. He appears to be in his mid- to late-fifties, about the right age to be the progenitor of the MRP system. He has the build of someone large who used to be in great shape. He has shoulder-length, graying hair, reminding her of The Dude from *The Big Lebowski*. But instead of being mellow and cool, Erik is clearly sharp and attentive.

He turns to Maxine with a sly smile. "On behalf of everyone in manufacturing operations, thanks for taking such good care of the MRP system. You've helped create and sustain software that is a masterpiece of simplicity and locality. You're not only magnificently meeting the business objectives, you've also created a system where small teams of engineers are able to work productively and independently of each other, with components painstakingly and splendidly isolated from each other, instead of being complected into a giant, ugly, knotty mess.

"A truly magnificent feat of engineering and architecture!" he says, beaming. "The developer productivity you've enabled is a beautiful testament to elegant simplicity. And even more impressive is your ruthless eradication of technical debt as a part of your daily work. I'm pleased to finally meet you!"

Maxine stares at Erik. *It's not every day that a bartender compliments you on the code you've painstakingly written and shepherded for years*, she thinks.

"Thank you—I'll make sure to pass that on to the team," she says, perplexed, but unable to hide her pride.

"Uh, what does 'complected' mean?" Kurt asks.

Erik answers, "It's an archaic word, resurrected by Sensei Rich Hickey. 'Complect' means to turn something simple into something complex.

"In tightly coupled and complected systems, it's nearly impossible to change anything, because you can't just change one area of the code, you must change one hundred, or even a thousand, areas of the code. And even the smallest changes can cause wildly unpredictable effects in distant parts of the system, maybe in something you've never even heard of.

"Sensei Hickey would say, 'Think of four strands of yarn that hang independently—that's a simple system. Now take those same four strands of yarn and braid them together. Now you've complected them.' Both configurations of yarn could fulfill the same engineering goal, but one is dramatically easier to change than the other. In the simple system, you can change one string independently without having to touch the others. Which is very good."

Erik laughs, "However, in the complected system, when you want to make a change to one strand of yarn, you are forced to change the other three strands too. In fact, for many things you may want to do, you simply cannot, because everything is so knotted together.

"And when that happens," he continues, "you've trapped yourself in a system of work where you can no longer solve real business problems easily anymore—instead, you're forced to merely solve puzzles all day, trying to figure out how to make your small change, obstructed by your complected system every step of the way. You must schedule meetings with other teams, try to convince them to change something for you, escalate it to their managers, maybe all the way up the chain.

"Everything you do becomes increasingly distant from the real business problem you're trying to solve," he says. "And that, Dwayne, is what everyone discovered when they switched out the routers in those manufacturing plants. Before, you had three independent strands, with teams able to work independently but at the cost of having to maintain three networking switches.

"When you put them all on one switch, you complected their value streams, all now having dependencies on each other that didn't exist before. They must constantly communicate, coordinate, schedule, marshal, sequence, and deconflict their work. They now have an extremely high cost of coordination, which has lengthened lead times, decreased quality, and, in your story, led to a week-long catastrophe that significantly impaired the business, going all the way up to Steve," Erik says with glee.

"The importance of lead times in software delivery is tantamount, as Senseis Dr. Nicole Forsgren and Jez Humble have discovered in their research," Erik says. "Code deployment lead time, code deployment frequency, and time to resolve problems are predictive of software delivery, operational performance, and organizational performance, and they correlate with burnout, employee engagement, and so much more.

"Simplicity is important because it enables locality. Locality in our code is what keeps systems loosely coupled, enabling us to deliver features faster. Teams can quickly and independently develop, test, and deploy value to customers. Locality in our organizations allows teams to make decisions without having to communicate and coordinate with people outside the team, potentially having to get approvals from distant authorities or committees so far removed from the work that they have no relevant basis to make good decisions," he says, clearly disgusted.

"You should be able to create value by changing one file, one module, one service, one component, one API call, one container, one app, or whatever! Which is why putting cross-cutting concerns in one place

is so great, like logging, security, or retry policies. You change it there, and you've changed it everywhere," he says. "Isn't it absurd that when you build a feature, changes sometimes have to be made by the UI team, the front-end team, the back-end team, and the database team?"

"Interesting," Maxine says. "Locality in our code and organization is so desirable, as opposed to what we have now, which is code scattered everywhere!"

"Yes, exactly. Scattered!" Erik says. "And achieving this greatness is never free. It requires focus and elevation of *improvement* of daily work, even over daily work itself. Without this ruthless focus, every simple system degrades over time, increasingly buried under a tundra of technical debt. Just look at the disaster that is the Phoenix build system."

Maxine furrows her brow. "You're saying that Phoenix used to be simple, but now it has become complected beyond recognition. That Phoenix used to have a great build process, but over the years it has become neglected, taking a backseat to features, and eventually bumped out of the car entirely."

"Precisely," says Erik. "Build responsibility moved from Dev to QA to interns. Tech giants like Facebook, Amazon, Netflix, Google, and Microsoft give Dev productivity responsibilities to only the most senior and experienced engineers. But here at Parts Unlimited, it's the exact opposite."

Dwayne laughs, "At least our builds aren't outsourced anymore. Not too long ago, it cost $85 each time a build was performed." Everyone, including Maxine, guffaws in disbelief.

Kirsten says, "I hear engineers complain all the time about technical debt? But what exactly is it, besides being something bad?"

Erik laughs. "There are many definitions, but my favorite is how it was originally defined by Ward Cunningham in 2003. He said, 'technical debt is what you feel the next time you want to make a change.' There are many things that people call technical debt, but it usually refers to things we need to clean up, or where we need to create or restore simplicity, so that that we can quickly, confidently, and safely make changes to the system.

"Sometimes it's a build and test system that doesn't give fast feedback to developers, or when it stops working entirely," he continues. "Sometimes it's when simple components become complected, and you

can no longer reason about it or change it without immense effort or risk of catastrophe. Sometimes it's when decision-making processes or the organizational structure loses locality, forcing even small decisions to be escalated—your infamous 'Square.'

"I've started calling all of these things 'complexity debt,' because they're not just technical issues—they're business issues. And it's always a choice," he says. "You can choose to build new features or you can choose to pay down complexity debt. When a fool spends all their time on features, the inevitable outcome is that even easy tasks become difficult and take longer to execute. And no matter how hard you try or how many people you have, it eventually collapses under its own weight , forcing you to start over from scratch."

He looks at Maxine and says, "Which is why what you've done with the MRP system is so remarkable. Your teams are able to add features at a rate that the entire Phoenix team should envy. And that is only possible because you pay down technical debt as a part of daily work. It's a magnificent example of the First Ideal of Locality and Simplicity in our code and organizations. Well done, Maxine."

Erik stands up. "I'm a little short-staffed tonight. I'll catch you later, and great to see you, Kirsten!

"Oh, one more thing," he says, turning around. "Think about the engagement scores of the technology employees versus the rest of the business and ponder the differences, especially on the Phoenix Project."

As Maxine watches Erik head back to the bar, she hears everyone burst into conversation.

Maxine says, "I have no idea what just happened." Looking at both Kirsten and Kurt, she asks, "What was that all about? And what did he mean by the First Ideal?"

"I have no idea," Kurt says, shaking his head. "I've known Erik for over a year. I had absolutely no idea he had some connection to the company…"

Dwayne says to Kurt, "I never bothered to tell you because, you know, it didn't seem that important. But one evening he asked me whether I knew anything about configuring Kubernetes clusters. That was pretty strange."

"That's odd," Shannon says. "Now that I think about it, I once had a debate with him about how completely you should or shouldn't isolate the cardholder data environment in order to comply with the PCI Data

Security Standard. He even sent me links to the specific subsections in the standard. He seemed very knowledgeable. An expert, even. I thought it was just because this bar took credit card payments…"

"I've heard he's been having many conversations with Bill Palmer, the new VP of IT Operations," Kirsten adds. "Bill told me about how Erik is teaching him something called the Three Ways and the Four Types of Work."

"I've never heard of those," Maxine says. "He only mentioned the First Ideal…I wonder how many other Ideals there are?"

"And what did he mean by engagement scores?" asks Kurt.

"I don't know," Kirsten says. "But I do know that we have some of the highest employee satisfaction scores in our industry…except for the IT department…which I think is negative twenty-seven."

"Is that bad?" Dwayne asks.

Kirsten looks embarrassed. "Very bad."

Maxine is not surprised. And yet, something bothers her. In the Town Hall, Steve talked about how much he cares about employee engagement. What does he think when he sees that the department responsible for the most strategic program in the company is miserable? Shouldn't that worry him?

When Erik walks by with a full beer glass, Maxine gets up and rushes to catch up with him. "Thanks again for the kind words, Erik. You mentioned the First Ideal—How many of them are there and what are they?"

"Ha! That's not the way it works," Erik says, laughing. "In fact, I've got Bill Palmer running hither and yon, trying to find all the Four Types of Work, watch. But…perhaps I can give you all a head start."

Erik and Maxine walk back to the table. "There are Five Ideals," Erik begins. The whole table turns their attention to him. "I've already told you about the First Ideal of Locality and Simplicity. We need to design things so that we have locality in our systems and the organizations that build them. And we need simplicity in everything we do. The last place we want complexity is internally, whether it's in our code, in our organization, or in our processes. The external world is complex enough, so it would be intolerable if we allow it in things we can actually control! We must make it easy to do our work."

Maxine sits back down, opens her laptop (pleased she remembered it this time), and starts taking notes.

"The Second Ideal is Focus, Flow, and Joy. It's all about how our daily work feels. Is our work marked by boredom and waiting for other people to get things done on our behalf? Do we blindly work on small pieces of the whole, only seeing the outcomes of our work during a deployment when everything blows up, leading to firefighting, punishment, and burnout? Or do we work in small batches, ideally single-piece flow, getting fast and continual feedback on our work? These are the conditions that allow for focus and flow, challenge, learning, discovery, mastering our domain, and even joy."

He looks around the table with a smug expression on his face. "And that's all you get for now. I'll share with you the other three Ideals when you're ready."

"You're kidding me," Maxine says. "You're pulling some sort of Yoda or Mr. Miyagi routine on us? Come on, at least tell us the *names* of the other Ideals!"

"Lucky for you, Young Grasshopper, I don't have time to argue, as there's a line at the bar I need to take care of," he says. "In its briefest form: The Third Ideal is Improvement of Daily Work. Reflect upon what the Toyota Andon cord teaches us about how we must elevate improvement of daily work over daily work itself. The Fourth Ideal is Psychological Safety, where we make it safe to talk about problems, because solving problems requires prevention, which requires honesty, and honesty requires the absence of fear. In manufacturing, psychological safety is just as important as physical safety. And finally, the Fifth Ideal is Customer Focus, where we ruthlessly question whether something actually matters to our customers, as in, are they willing to pay us for it or is it only of value to our functional silo?"

Erik finishes his beer and says with a smile, "Good luck to you all. See you next week."

"Wait, wait, that's it?" Maxine says, but Erik is already gone. Maxine looks down at her quickly typed notes:

> The First Ideal—Locality and Simplicity
> The Second Ideal—Focus, Flow, and Joy
> The Third Ideal—Improvement of Daily Work

The Fourth Ideal—Psychological Safety
The Fifth Ideal—Customer Focus

Maxine stares at the list—all of the Ideals sound nice, but how in the world are they supposed to use them to change the trajectory of the Phoenix Project?

"That was so strange," Kurt says, saying what everyone is thinking.

Cranky Dave adds, "That bit about the Fourth Ideal hit home. A culture of fear where everyone is afraid to share bad news? That's us.

"Erik is right," Adam says. "No one talks about the real problem. Most people aren't brave enough to say what they think or to do the right thing. They just say 'yes,' whether they agree or not. But maybe this creates an opportunity. There are some big, gaping holes in the org chart now," he says to Kurt. "You should throw your name into the hat for one of them. Maybe even for William's position?"

Silence descends upon the table as everyone turns to look at Adam and Kurt.

"That's a pretty good idea, Kurt. You could make a huge difference in the QA organization. You know all of us would be pretty happy about that," says Shannon, with everyone around the table murmuring assent.

"Maybe," says Kurt, nodding slowly. "But you know, if we really want to make a difference, there's another move. I'm thinking about telling Chris that I want Peter's position."

Maxine hears some gasps around the table, followed by Cranky Dave's loud laugh. "You're right, Kurt. You would definitely make a much bigger difference by taking over a Dev team. We all know we need to change how QA does testing, but the best place to start is by changing how *Dev* does testing. And that requires being a Dev manager…but that brings up a teeny, tiny, little problem…they'll never give you that position, Kurt," he says. "You know, because you're 'just a QA manager.'"

Maxine winces. Cranky Dave is voicing an all-too-popular prejudice that developers have about QA people, which embarrasses her. QA is often viewed as an underclass, but at least they're above Ops. *All of which is crap*, Maxine thinks. After all, she started her Ops career in high school, rotating backup tapes, and later, before graduate school, QA—if

it weren't for that background, she wouldn't have become the person she is today. Technology is still too often a caste system.

Adam says to Kurt, "You know I'm a big fan and I love working with you—you're a fantastic leader—but I agree with Dave. There's no way that a bunch of Dev managers are going to let a QA manager take that spot. Maybe you should just settle for William's old role. After all, someone has to lead QA out of the Stone Age and bring automated testing to the rest of the Phoenix Project."

"I have to agree with your friends, Kurt," Kirsten says. "You and I both know that William was never a big fan of yours. He never spoke very highly of you in meetings. They're probably going to bring in someone from the outside."

Kurt grins, seemingly not bothered by Kirsten's observation. In his great William impersonation, Kurt says, "'Yes, Kirsten, you are right. Although Kurt shows some potential, it's clear to me that he doesn't understand the testing game. Maybe in a couple of years he'll have the maturity to run the QA organization.'"

Everyone laughs. Kurt continues in his normal voice, "Folks, here's an opportunity for us to make a difference. But I don't think we can do it from anywhere in the QA organization—QA as we know it is changing. We can't keep being the people who test after the fact. We need to get into the game, and that means finagling our way into the development teams that are actually responsible for shipping features and the quality of their outcomes. Anything else is a waste of our time."

He continues, "In fact, if we can take over Peter's team, my goal will be to show that we can out-perform every other Dev team in the Phoenix Project. Gathered around this table is some of the best technical talent in the company, and we've already created the infrastructure that can bring some great technical practices to the game."

Kurt leans forward. "If I can get Chris to give me that chance, would you all be willing to join the team and show that we can change the trajectory of the Phoenix Project?"

"Hell yes, Kurt. Count me in!" says Cranky Dave. Maxine is surprised that he's the first to volunteer.

Maxine follows. "And me. This is what I want to work on. And I know we can run circles around all the other teams. I've seen the competition up close," she says with a smile.

Everyone around the table chimes in, excited at the potential opportunity. Cranky Dave says, "Okay, we're all in, Kurt. But frankly, I'm not holding my breath. Adam is right—you getting a Dev team is a long shot."

Kirsten says, "Kurt, I agree with your instincts. If you want, I'll write a letter of recommendation to Chris."

"That would be fantastic, Kirsten," says Kurt, beaming and obviously genuinely surprised and grateful for Kirsten's offer. At that moment, Maxine realizes that Kurt has been operating this entire time without any real leadership air cover. *He could get fired for going rogue*, she realizes.

"Happy to help," says Kirsten. "But let me be clear. I'm willing to write a letter to support Kurt's ideas, but I really can't be seen publicly with you all. At least, not yet. People need to see me as impartial."

"Oh, you're willing to give us a chance to take a risk and get fired, but you want to stay safely on the sidelines?" says Cranky Dave, halfway joking. Kirsten merely raises her glass to Dave.

PART TWO

September 23–November 9

CHAPTER 8

• *Tuesday, September 23*

On Tuesday of the following week, Maxine arrives at work to see Kurt beaming. "I got the job," he says exuberantly.

"Really? The Dev job?" Maxine asks.

"Yes, the Dev job!" he says, as if he can't quite believe it himself. "It couldn't have happened without Kirsten's support. I'm joining the Data Hub team, and you're coming with me."

"That's awesome!" Maxine says, jubilant. "How'd you get Randy to approve my reassignment?"

"Well, he wasn't happy about losing you. He kept going on about how you're the best thing to happen to this place since sliced bread, but . . . well, I have my ways," Kurt says with a sly smile.

Maxine gives him a high five.

He looks around and whispers, "All the managers are talking about something very strange happening. Apparently, the technology executives had an off-site with Steve earlier this week, and one of the things they agreed upon was a one-month feature freeze. Apparently, they're actually hitting the brakes on feature delivery to pay down all the technical debt we've built up over the years!"

"Really?!" Maxine is shocked.

"They realize they need to fix all the crap that's been built," he says. "Ops is halting all work not related to the Phoenix Project so they can pay down technical debt and automate things. And Dev and QA will halt all feature work to pay down their technical debt too.

"This is our moment to shine. Here's our chance to show people what engineering greatness looks like," Kurt exclaims.

Later that day, an email goes out announcing Kurt's new role. Maxine doesn't want to hurt Kurt's feelings, but she's pretty sure that the real reason he got the job was that absolutely nobody else in Development

wanted it. Data Hub is being widely touted as the "root cause" of the catastrophic crashes during and after the Phoenix release. Chris even called them out by name during one of the meetings Maxine was in, which she thought was quite unfair.

Blaming the Data Hub team for the smoking crater in the ground that was the Phoenix deployment was like blaming an airline crash on the passenger in the back of the plane who didn't fasten his seatbelt tight enough.

She knows why blaming Data Hub is so easy. It's one of the least glamorous technology areas in the company. Data Hub is part of a big, boring message bus system, which Maxine already loves because it's how most of the major applications and systems of record talk to each other: the product database, pricing database, inventory management systems, order fulfillment systems, sales commissioning systems, company financials, and almost a hundred other major systems, many of them decades old.

Maxine has never liked that there are actually three inventory management systems—two for the physical stores (one was inherited by an acquisition and never retired) and another for the e-commerce channel. And there's at least six order entry systems—three supporting the physical stores, one for e-commerce, another for OEM customers, and another for service station channel sales.

Maxine loves byzantine processes like pediatricians love sick children, but even Maxine is taken aback by just how many systems Data Hub has to talk to.

The more Maxine studies up on what Data Hub does, the more perplexed she becomes. Data Hub just didn't seem like it should be part of Phoenix at all. After all, the majority of Data Hub was written over twenty years ago, which was long before Phoenix was even a concept.

Apparently, Data Hub used to be a collection of smaller applications that were scattered across the company. Some resided in finance with the ERP systems, some inside the manufacturing business units, and others within the Development group under Chris.

As the Phoenix juggernaut started rolling, an incredible number of new demands were put on those teams, and those teams just weren't staffed to deal with it. Tons of new Phoenix functionality was blocked because of competing business priorities in Data Hub, and soon Phoenix features were being delayed, month after month.

Finally, as part of a re-org, all those components were rolled up into a new group called Data Hub and put under the Phoenix Project, making sure Phoenix priorities always came first. And now, everyone is blaming Data Hub for what went wrong.

On Wednesday morning, Maxine and Cranky Dave join Kurt for the first meeting with the Data Hub engineers. Maxine's surprised to see that Cranky Dave was able to join Data Hub so quickly too. She asked him how he managed to swing that.

Cranky Dave merely smiled, saying, "One of the many benefits of my winning personality—no manager passes on the opportunity to give me to a different team. It allows me to go wherever I want."

She stands next to Cranky Dave as the other five Data Hub engineers assemble in the central meeting area.

They're all either her age or fresh out of college, no one in between. She suspects the senior developers have been on the team since the beginning, and the younger engineers will quickly leave for more interesting work and be replaced with other new college grads.

Chris clears his throat and addresses the room. "Good morning, everyone. Please welcome Kurt Reznick, who will be taking over for Peter."

Kurt seems surprised by the short introduction but says cheerily, "Hello, everyone. As you may know, this is the first Dev team I've managed. I believe my job my job is very simple: listen, do whatever you need me to do to help make you successful, and remove any obstacles in your way." It's clear from everyone's unimpressed looks that they're well aware of Kurt's lack of experience.

Kurt continues, "I've talked with our numerous internal customers, and they told me how important Data Hub is. But they also told me about how we're often the bottleneck for changes needed across the enterprise, as well as for the Phoenix Project. And as we all know, when our service goes down, so does Phoenix. I've scheduled a session later this week for us to brainstorm how we can make our service more reliable and resilient."

"Blaming Data Hub and Peter for Phoenix going down is bullshit," says one of the senior developers.

"I totally agree with you, Tom," says Kurt. "And rest assured that I'll be working to correct that perception."

Kurt continues, "I really appreciate that Peter was willing to meet with me before I started. He told me that he's been asking for additional headcount for senior developers for years because the business needs have kept growing, especially around the Phoenix integration. He recommended that I keep trying."

Kurt gestures at Chris. "And I promise you that I'll keep lobbying Chris for more headcount."

"And I'll keep lobbying Steve," Chris replies with a tight-lipped smile. Kurt laughs. "So, in the meantime, I've brought with me two senior developers who have volunteered to join the team. Maxine is the most senior developer from the MRP team, and Dave is a senior developer from the Phoenix back-end server team. They're two of the developers I trust the most."

The Data Hub developers look at them, surprised but genuinely happy that she and Cranky Dave are here.

"There will be a directive coming out soon from Chris about a feature freeze, so we can work on fixing defects that impact our customers and fix problematic areas of our code," Kurt says. "But don't wait for the announcement. The top priority is to fix things you think should be fixed and, for that matter, anything that you think will help you be more productive or make Data Hub more stable. I'll handle any complaints that come our way."

Maxine smiles at the expressions of grudging approval from the Data Hub engineers.

As the new engineers, Cranky Dave and Maxine integrate themselves into the daily rituals of the Data Hub team. They attend the stand-ups and are quick to volunteer to help with things.

Maxine pairs up with Tom, the older developer who had commented on the unfairness of being scapegoated for the Phoenix failure. Tom is in his late forties and wears glasses, jeans, and a T-shirt. She sits at his desk with her laptop open as he explains what he's currently working on.

As he talks, Maxine sees that Data Hub is a mishmash of technologies built up over the decades, including a big chunk that runs on Java servlets, some Python scripts, and something that she thinks is Delphi. There's even a PHP web server.

She doesn't judge or dismiss any of the technology stacks—after all, it's been successfully serving the enterprise for decades. It may not be the most elegant piece of software she's seen, but things that have been in production for twenty years rarely are. Software is like a city, constantly undergoing change, needing renovations and repair. She will, however, acknowledge that Data Hub is not the hippest neighborhood. It's undoubtedly difficult to recruit new college grads who want to learn and use the hottest, most in-demand languages and frameworks.

But at least Data Hub is in much better shape than the Phoenix build systems, which were like uninhabitable, radioactive Superfund sites, or the shelled-out remains of a war zone.

Maxine is sitting at Tom's desk as he explains what he's working on, "I'm working on an urgent defect. Data Hub is occasionally generating incorrect message transactions and is crashing under high loads. It sometimes happens when in-store employees mark customer repair work as complete in the service station application," he says. Looking embarrassed, he continues, "I've spent days working on this. I've finally created a semi-reproducible test case—it happens about one out of ten times. I'm pretty sure it's because of a race condition."

Talk about being thrown in the deep end, Maxine thinks. But she relishes the challenge and is sure that when they solve the problem, it will make a very positive impression on the entire team. After all, race conditions are one of the toughest categories of problems in all of distributed systems and software engineering. If working with the middle-school girls was a yellow belt challenge in karate, what Tom is describing can drive even the most experienced tenth-level black belts to despair and madness.

Maxine is impressed that Tom can even reproduce the problem at all. Someone once called these problems "heisenbugs," referring to the quantum physics phenomena where the act of observation changes the nature of reality itself.

This type of work is very different than how coding is portrayed in the movies: a young, male programmer is typing away furiously, wearing a hoodie, of course, but curiously, also wearing sunglasses (which she has never actually seen a developer do in real life). He has many different

windows open, text quickly scrolling by in all of them. Behind him, a crowd of people is watching over his shoulder, waiting anxiously. After a couple of seconds, the coder cries out, "I got it!" and everyone cheers. The solution is created, the feature is delivered, or the world is saved. And the scene ends.

But in reality, when developers work, they're usually staring at the screen, deep in concentration, trying to understand what the code does so they can safely and surgically change it without breaking something else as an unintended side-effect, especially if they're working on something mission-critical.

Tom walks her through the problem. "When there are multiple repair transactions being processed concurrently, sometimes one of the transactions gets the wrong customer ID, and sometimes Data Hub completely crashes," he says. "I've tried putting a lock around the customer object, but it slowed down the entire application so much, it's just not an option. We have enough performance problems as it is."

Maxine nods, because Tom is confirming her long-held belief that multi-threading errors are at the very limits of what humans can reason about, especially since most mainstream programming languages like Java, C#, and JavaScript encourage mutation of shared state.

It is almost impossible to predict how a program will behave if any other part of the program can change data that you're depending on at any time, Maxine thinks. But she's pretty sure she knows how to fix this problem.

"Can we walk through the code path again?" Maxine asks. As they do, Maxine goes through a mental checklist to confirm her hypothesis. There's a thread pool that handles incoming messages. Check. Service records can be handled by multiple concurrent threads. Check. The threads pass around objects, which are being mutated when its methods are called. Check.

Hypothesis confirmed. The problem is almost definitely state mutation going wrong, she thinks. *Just like at the middle school.*

"You're right, it's definitely a race condition," Maxine says. "And I'm pretty sure we can solve this problem without putting a lock around the entire customer object. Can I show you what I'm thinking?"

When he nods, just as Maxine did with the middle-school girls, she proposes rewriting the code path using functional programming principles. Tom's test case has a lot of mocks and stubs to simulate the

production environment: a configuration server, a database, a message bus, a customer object factory....

She jettisons all of them, because those are not areas of the system she wants to test. Instead, she pushes all that input/output and side-effects to the edges and creates unit tests around how an incoming repair order message is processed, how customer data is transformed, and what outgoing messages are sent.

She has each thread make its own copy of the customer object. They rewrite each object method into a series of pure functions—a function whose output is completely dependent upon its inputs, with no side-effects, mutations, or accesses to global state.

When Maxine shows Tom a unit test that reproduces the problem 100% of the time, as well as the completely thread-safe fix that now works 100% of the time, Tom stares at her, eyes wide with wonder. "That's... that's... incredible."

She knows why he's impressed. Her code is so simple that it's easy to understand and test for correctness. Eventually, marveling at the screen, he says, "I just can't believe how much you simplified it. How can that achieve the same thing as the complex mess that we had before?" For the rest of the afternoon, he asks questions, obviously trying to prove to himself that Maxine's test case captured the problem and that the rewrite is correct. At last he says, "I can't believe it, but I think you're right. This will definitely work!"

Maxine grins at Tom's reaction. Another testament that functional programming principles are better tools to think with. And they've already made the code way better than when they found it—it's definitely safer, easier to test, and way easier to understand. *This is so much fun,* she thinks. *And a great example of the First Ideal of Locality and Simplicity*.

"Okay, let's get this fix merged in!" he says, opening up a terminal window, typing in some commands. He turns to Maxine. "Congratulations! You just fixed your first defect and checked in your first change!"

Maxine gives him a huge high-five, a big grin on her face. Vanquishing a race condition error on her first day is freaking awesome. "That's great! So, let's get this thing tested and pushed into production." Maxine is excited at the thought of a grateful store manager thanking them.

"Errrr... Uhhhh..." Tom says, pausing. "Testing doesn't start until Monday."

Maxine feels her heart drop. "We can't test it ourselves?"

"We used to be able to, before we got re-orged into Phoenix," he says, wistfully. "The QA group took over the testing. And when they had some problems with different teams using the test environment at the same time, they took everyone's access away. Now they're the only ones who can log in, let alone run the tests."

"Wait," she says. "We write the tests but can't run them?"

He laughs. "No, no. They write the tests. They don't even let us see the test plans anymore."

Maxine deflates even more, knowing where this is going. "And we can't push it into production?"

Tom laughs again. "Nope, not any more. We used to be able to do that too. But now someone else deploys it for us. 'Stay in your lane,' they told us." He shrugs his shoulders. Maxine is pretty sure she knows who said, "Stay in your lane." That'd be Chris.

The joy that Maxine felt all day while working on the problem disappears. After all, fixing the code especially for features is just a fraction of the entire job. It's not done until that customer can use what they've written. And even then, it's probably still a work in progress, because we can always learn more about how to help that customer best achieve their goals.

"Crap," she mutters. *I'm back in the same place I was before, a long way away from the First Ideal. I still can't actually do anything myself*, Maxine thinks. Once again, she is dependent on others to create customer value.

Oblivious, Tom laughs and opens up a new window. "It's not so bad. We just need to go into the ticketing system and mark this issue as 'done.' That lets the QA team know to test it, so it can be promoted into production."

Tom looks at his watch and turns back to her, "That was great. We got a lot done today. Want to pick another defect to work on?" Maxine forces a smile and nods. *This sucks,* Maxine thinks. She likes finishing things, not just starting things.

Maxine continues to work with Tom all day, picking the next most urgent defect to be fixed. Tom once again compliments Maxine on how she thinks about problems. He's impressed at how she writes unit tests that

can be run without the need for a complex, integration test environment.

But there are limits—Data Hub's job is to connect systems with each other. There's only so much you can simulate on a single laptop. *It would be nice to rearchitect Data Hub so you could,* Maxine thinks wistfully.

Although she enjoys learning about Data Hub and the parts of the business it connects, there's something about all this work that is deeply unsatisfying to her.

She thinks of Erik's Second Ideal of Focus, Flow, and Joy. All the joy she felt vaporized when Tom told her that they had completed only a small portion of the work needed to create value. That's just not good enough for her. In her MRP team, any developer could test their own code and even push code into production themselves. They didn't have to wait weeks for other people to do that work for them. Being able to test and push code to production is more productive, makes for happier customers, creates accountability of code quality to the people who write it, and also makes the work more joyful and rewarding.

Maxine starts thinking about how to introduce some of the tools being built by the Rebellion. *At the very minimum, we need to make standardized Dev environments available, so I can do builds on my laptop*, she thinks. More things to talk about at the next Dockside meeting.

She continues to grind away, helping Tom with the work that he's been assigned. Together they fix two defects and then tackle a crash-priority feature, this time to create some business rules around extended warranty plans, critical enough to be exempted from the feature freeze.

"Why is this so high priority?" Maxine asks Tom as she reads the ticket.

"This is hugely revenue-generating," Tom explains. "One of the highest margin products are these new extended-warrantee plans. Customers loved the pilot warrantee program, especially for things like tires. Now in-store staff need a way to pull up this information, so they can do the repair work and file the claim with the third-party insurer."

Tom continues, "Great for the customer, great for us, and a third-party insurer is taking all the financial risk."

"Cool," Maxine says, perking up. It's features like this that support everything that Steve said in the Town Hall. It's been a long time since Maxine's done work on the revenue-generating side of the business.

Recommitting herself to feel relentlessly optimistic, Maxine and Tom start studying the feature, trying to figure out what is required to enable

this important business capability. She tries not to think about how, even if they get it done today, it'll just sit, waiting for the QA team to test it.

The next morning, Tom and Maxine are at a whiteboard, inventorying all the systems that they'll need to change in order to enable extended warrantees. Two more engineers have joined them as the scope keeps increasing. And then they realize that they'll need to talk with engineers from two other teams, as well. Maxine guesses that they'll have to bring in six other teams because of how many business systems this affects.

Maxine is dismayed as the number of teams that need to be involved keeps growing. This is again the opposite of the First Ideal of Locality and Simplicity. Here, the changes that need to be made are not localized. Instead, their scattered across many, many different teams. This is not the famous Amazon ideal of the "two-pizza team," where features can be created by individual teams that can be fed with two pizzas.

We'll need a whole truckload of pizzas to ship this feature, Maxine thinks, watching as Tom draws another set of boxes on the whiteboard.

Kurt pokes his head into the conference room. "Hey, sorry for the interruption. Someone from Ops and the manager of the channel training management application are on a conference bridge. All their customer logins are failing. They say the connector has stopped working?"

"Not again," says Tom. "Authentication has been flaky ever since the Phoenix deployment. We're on it…"

"Roger that," Kurt says, tapping something on his phone. "I just created a chat channel for all of us, okay?"

Maxine follows Tom back to his desk. As Tom opens up another browser window and types something, a login error appears on his screen.

"Okay, something's definitely not working right. Let's see if we can isolate why…" Tom mutters. "I doubt it's actually a Data Hub connector. More likely it's the enterprise customer authentication service or a problem in the network."

Maxine nods, taking notes as more of the Data Hub universe comes into view. Skeptical, she offers, "Can't we rule out network and authentication right away? If either of those were down, we wouldn't even be able to get to the website, and authentication being down would take out every service…"

"Good point…" Tom says. "Could definitely still be networking, though…we've had a bunch of issues lately. Last week, the networking people accidentally blocked some internal IP addresses that caused us problems."

"Networking. It's always the networking people, right?" she says, smiling. "But if it's always the networking people, why are they calling us?" Maxine asks.

"Yeah, well, all the users know is that they can't connect to Data Hub," he says. "We always explain that it's not us; it's something we need to connect to. But they don't care."

When Maxine sees Tom pull up the Ops ticketing system and create a new ticket, she asks, "What's this for?"

"We need the production logs for Data Hub and its connectors to see if they're handling traffic or if they've crashed," he responds, filling out the numerous fields.

"We can't directly access production logs?" Maxine asks, afraid of the answer.

"Nope. Ops people won't let us," he says, typing into the form.

"So, someone has to respond to the ticket and copy the logs off the server for us?" she asks in disbelief.

"Yes," he says, continuing to type, obviously very practiced at filling it out. He tabs between fields, types, mouses over to hit the drop-down box, hits the submit button, only to find that there's still another required field that needs to be filled in.

Maxine groans. The Data Hub application that they're working on might as well be running in outer space or at the bottom of a deep well. They can't directly access it, they can't see what it's doing, and the only way they can understand what's actually happening is to talk to someone in Operations through the ticketing system.

She wonders whether the ticket will get routed to her friend Derek at the helpdesk.

Tom finally succeeds in submitting the ticket. Satisfied, he says, "Now we wait."

"How long does it usually take?" Maxine asks.

"For a Sev 2 incident? Not too bad—we'll probably get it within a half hour. If it's not related to an outage, it could take days," Tom says. He looks at the clock. "What should we do while we wait?"

Even in the Data Hub team, she can't escape the Waiting Place.

Four hours later, after reviewing the production logs, they confirm that the problem isn't Data Hub. Two hours after that, everyone finally agrees. As Tom had suspected, it was an internal networking change that caused the problem.

Another round of intense finger-pointing ensues between Business Operations, Marketing, and within the technology organization. Eventually, Sarah gets involved and demands that there be severe consequences.

"Uh, oh," says Tom, watching with Maxine from the far end of the table. "This can't be good."

From: Wes Davis (Director, Distributed Operations)
To: All IT Employees
Date: 7:50 p.m., September 25
Subject: Personnel changes

Effective immediately, Chad Stone in network engineering is no longer with the company. Please direct all emails to his manager, Irene Cooper, or me.

For the love of all that is holy, please stop making mistakes so that I don't have to write these stupid emails. (And if they fire me, direct your emails to Bill Palmer, VP, IT Operations.)

Thank you,
Wes

Finally, the day is over, which means another meeting at the Dockside Bar. They've invited the entire Data Hub team to join them. Maxine approves of being over-inclusive rather than accidentally leaving some worthy people out. Tom and three other engineers show up. Maxine is glad they're here. After the last couple days, she's eager to brainstorm ways to dramatically improve developer productivity on the Data Hub team.

Seeing everyone having fun, Maxine observes that this is a group of people who love hanging out with each other. Kurt stands up and addresses the group.

"Hello, new Rebellion teammates! Let me introduce everyone," Kurt says. He introduces all the Rebellion members, as he did for Maxine and Kirsten. "And if you don't mind, now that you've heard about some of the subversive things we're working on to bring joy back to Parts Unlimited engineers, how about you tell us something that could make your lives a little easier?"

Tom's two colleagues go first, introducing themselves and sharing their backgrounds. One has been on the Data Hub team, like Tom, for nearly a decade, but he doesn't come up with anything to complain about, saying, "Life is okay, and I appreciate the invitation for drinks."

When he clearly doesn't have anything more to say, Tom starts. "Like my colleague, I've been on the Data Hub team for a long time. Back when it used to be called Octopus. We called it that because of how it connected to eight applications. Now it connects to over a hundred.

"I've been having a blast pair-programming with Maxine, and I still can't believe we fixed a race condition bug! I'm delighted at her idea to get Data Hub Dev environments that we can all use," he continues. "I'm not proud of this, but there have been times when we've hired new developers and six months later, they still can't do a full build on their machines," he says, shaking his head. "It wasn't always like this. When I started, it was simpler. But over the years, we've hard-coded some things that we shouldn't have, updated some things here, updated other things there, never quite documenting all of it . . . and now? It's a mess."

Looking up, he smirks at his teammates around the table, saying, "You know the developer joke of 'it worked on my laptop'? Well, in Data Hub, we can't even get it running on most people's laptops."

Everyone laughs. At one point or another, every developer on the planet has had this problem. It usually happens at the worst possible time, like when something crashes in production but mysteriously works perfectly on the developer's laptop. Maxine remembers countless times when she's had to painstakingly figure out what exactly was different between the developer's laptop and the production environment.

"My pain points?" Tom muses. "It's our environments. We used to have a good handle on this, but then we got moved into the Phoenix

Project and they made us use environments from their centralized environments team.

"It's crazy. We're puny compared to the rest of Phoenix. To run Data Hub now, we have to install gigabytes of completely irrelevant dependencies," he continues. "It takes forever to figure out how to get everything to run, and it's so easy to break something by accident. No joke: I back up my work laptop every day because I'm so afraid that my builds will stop working and I'll have to spend weeks figuring out how to fix them."

Tom laughs, "Ten years ago, I lost my *emacs* configuration file and couldn't find a recent backup. I just didn't have it in me to recreate it. I finally gave up and switched editors."

Everyone laughs, adding their own stories of loss, anguish, and grief of having to give up their most treasured tools.

Tom turns to Maxine. "I'd love to spend a couple of days exploring how we can make a Dev environment that all of us could use in our daily work. If we had a virtual machine image or a Docker image, any new team member could do a build on any machine, any time. That would be incredible."

"You and I are definitely going to get along," Maxine says, smiling. "We need developers to be able to focus their best energies on building features, not trying to get builds to work. I have a ton of passion for this too and would love your help."

"That's terrific," Kurt says. "We all know how important environments are. For now, feel free to spend half your time on this—I'll hide it in the timecarding system."

Later in the evening, Kirsten shows up and pours herself a glass of beer from the pitcher on the table. Smiling, she says, "What did I miss?"

"Just plotting the inevitable toppling of the existing order, of course," Kurt says. The new Data Hub team members openly stare at Kirsten as she takes a seat.

Kurt asks, "Kirsten, how's Project Inversion going? The feature freeze? I heard that Bill Palmer convinced Steve to put all feature work on hold so everyone can pay down technical debt."

"Confirmed," she says. "Sarah Moulton is going ballistic, complaining how 'all the idle developers' are jeopardizing the promises the company

has already made to customers and Wall Street. I still can't believe she doesn't get how this helps her. But Project Inversion is definitely happening: for thirty days, Ops is not doing anything except things to support Phoenix."

"They're not kidding around," Brent says. "Bill has been awesome. He's told me in no uncertain terms that I'm to work only on Phoenix-related things. He's taken me off of pager rotation for basically everything. He's even taken me off of every mailing list, had me turn off notifications from every chat room, and told me not to answer the phone for anyone. And best of all, he said to absolutely not show up for any outage calls. If I do, he'll fire me."

Hearing this, Maxine is shocked. Bill would fire Brent? Thinking of all the people who've been fired lately, Maxine can't figure out why Brent is smiling.

"It's so fantastic," Brent says, even appearing to be . . . tearing up? "Bill told me that he can't fire the business unit executives or tell them what to do. He said that the only thing he *can* do is ensure that I'm not wasting time on those things. He said to tell anyone trying to reach me that I'll be fired if I call them back."

Brent laughs, obviously elated, finishing his beer and pouring himself another. "He's assigned Wes to screen all my emails and phone calls and to yell at anyone trying to get ahold of me. Life is fantastic! Seriously, never better."

Maxine smiles. She has seen how engineers can become the constraint many times in her career. It can be fun to be at the center of everything, but it's certainly not sustainable. Down that road, only chronic wakeup calls, exhaustion, cynicism, and burnout await.

Kirsten smiles. "It's working. Brent's name shows up on more critical action items than anyone, and Bill has told everyone that their goal must be to protect his time.

"On the Development side, Chris promises that for thirty days, for all teams working on anything related to Project Phoenix, no new features," Kirsten says, reading from her phone. "'All teams need to be fixing high-priority defects, stabilizing the codebase, and doing whatever rearchitecting is needed to prevent another release disaster.'"

Maxine hears lots of excited murmurs from around the table. Maxine knows something like this is needed—and that this could be a fantastic opportunity for the Rebellion.

"There's still a lot of disagreement among Chris's direct reports on how to roll this out," Kirsten continues. "They've spent so much time legislating what should and shouldn't be worked on that we've already lost a week—lots of teams are still working on their features, business as usual. We're going to need a lot more clarity from leadership on this—at this rate, the entire month will be gone, and we'll have the same amount of technical debt as before, if not more."

"I'm surprised no one is talking about all the problems they're having with environments or automated testing or the lack of production telemetry," Kurt says. "We've built some amazing capabilities that other people can use too. But we can't be the people with a solution, peddling them to people who don't know they have a problem."

Kurt looks stumped. And frustrated.

"I totally want to help with this," Shannon says, raising her hand. "I've worked with a bunch of the Phoenix teams. I could swing by each one tomorrow to start asking them what their constraints are and any ideas they have on how to fix them."

"Good, good," Kurt says, writing down some notes in his notebook.

"I'd love to help too, Shannon," Maxine says. "But Tom and I will be a little tied up on Monday, because Monday is Testing Day. I'm going to finally get my changes tested with the QA folks. Outside of that, I'm yours!" A full tray of beer pitchers and two more glasses of wine appear.

They are soon in deep conversation about technical debt and ideas on how to take advantage of Project Inversion. Maxine turns to see Erik grabbing the seat next to her.

He joins the conversation as if he's been there all along. "With Project Inversion, you are all on the beginning of a great journey. Every tech giant has nearly been killed by technical debt. You name it: Facebook, Amazon, Netflix, Google, Microsoft, eBay, LinkedIn, Twitter, and so many more. Like the Phoenix Project, they became so encumbered by technical debt they could no longer deliver what their customers demanded," Erik says. "The consequences would have been fatal—and for every survivor, there are companies like Nokia who fell from the loftiest heights, killed by technical debt.

"Technical debt is a fact of life, like deadlines. Business people understand deadlines, but often are completely oblivious that technical debt even exists. Technical debt is inherently neither good nor bad—it happens

because in our daily work, we are always making trade-off decisions," he says. "To make the date, sometimes we take shortcuts, or skip writing our automated tests, or hard-code something for a very specific case, knowing that it won't work in the long-term. Sometimes we tolerate daily workarounds, like manually creating an environment or manually performing a deployment. We make a grave mistake when we don't realize how much this impacts our future productivity."

Erik looks around the table, pleased that everyone is listening intently to his every word.

"All the tech giants, at some point in their history, have used the feature freeze to massively rearchitect their systems. Consider Microsoft in the early 2000s—that was when computer worms were routinely taking down the internet, most famously *CodeRed, Nimda*, and of course *SQL Slammer*, which infected and crashed nearly 100,000 servers around the world in less than ten minutes. CEO Bill Gates was so concerned that he wrote a famous internal memo to every employee, stating that if a developer has to choose between implementing a feature or improving security, they must choose security, because nothing less than the survival of the company was at stake. And thus began the famous security stand-down that affected every product at Microsoft. Interestingly, Satya Nadella, CEO of Microsoft, still has a culture that if a developer ever has a choice between working on a feature or developer productivity, they should always choose developer productivity.

"Back to 2002—that same year, Amazon CEO Jeff Bezos wrote his famous memo to all technologists, stating that they must rearchitect their systems so that all data and functionality are provided through services. Their initial focus was their OBIDOS system, originally written in 1996, which held almost all the business logic, display logic, and functionality that made Amazon.com so famous.

"But over time, it became too complected for teams to be able to work independently. Amazon likely spent over $1 billion over six years rearchitecting all their internal services to be decoupled from each other. The result was astonishing. By 2013 they were performing nearly 136,000 deployments per day. Interesting that these CEOs I mention all have a software background, isn't it?

"Contrast that with the tragic story of Nokia. When their market was disrupted by Apple and Android, they spent hundreds of millions of

dollars hiring developers and investing in rolling out Agile. But they did so without realizing their real problem: technical debt in the form of an architecture where developers could not be productive. They lacked the conviction to rebuild the foundations of their software systems. Just like at Amazon in 2002, every software team at Nokia was unable to build what they needed to because they were hamstrung by the Symbian platform.

"In 2010, Risto Siilasmaa was a board director at Nokia. When he learned that generating a Symbian build took *a whole forty-eight hours*, he said that it felt like someone hit him in the head with a sledgehammer," Erik says. "He knew that if it took two days for anyone to determine whether a change worked or would have to be redone, there was a fundamental and fatal flaw in their architecture that doomed their near-term profitability and long-term viability. They could have had twenty times more developers, and it wouldn't have made them go any faster.

Erik pauses. "It's incredible. Sensei Siilasmaa knew that all the hopes and promises made by the engineering organization was a mirage. Even though there were numerous internal efforts to migrate off of Symbian, it was always shot down by the top executives until it was too late.

"Business people can see features or apps, so getting funding for those is easy," he continues. "But they don't see the vast architectures underneath that support them, connecting systems, teams, and data to each other. And underneath that is something extraordinarily important: the systems that developers use in their daily work to be productive.

"It's funny: the tech giants assign their very best engineers to that bottom layer, so that every developer can benefit. But at Parts Unlimited, the very best engineers work on features at that top layer, with no one besides interns on the bottom working on Dev productivity.

Erik continues, "So your mission is clear. Everyone has been told to pay down technical debt, which will help you realize the First Ideal of Locality and Simplicity and the Second Ideal of Focus, Flow, and Joy. But almost certainly, you will have to master the Third Ideal of Improvement of Daily Work." Then he gets up and leaves as quickly as he joined them.

Everyone watches him leave. Then Kirsten says, "Is he coming back?"

Cranky Dave throws his hands in the air. "What happened at Nokia is happening here. Two years ago, we could implement a significant feature in two to four weeks. And we delivered a ton of great stuff. I remember those days! If you had a great idea, we could get it done.

"But now? That same class of feature takes twenty to forty weeks. Ten times longer! No wonder everyone's so pissed off at us," Cranky Dave yells. "We've hired more engineers, but it feels like we're getting less and less done. And not only are we slower, those changes are incredibly dangerous to make."

"This makes sense," Kirsten says. "By almost any measure, productivity is flat or down. Feature due date performance is way down. I did some research since our last meeting—I asked my project managers to sample a couple of features and find out how many teams were required to implement them. The average number of teams required was 4.2, which is shocking. Then they told me that many had to interact with *over eight teams*," she says. "We've never formally tracked this, but most of my people say that these numbers are definitely higher than they were two years ago."

Maxine's jaw drops. *Absolutely no one can get anything done if they have to work with eight other teams all the time*, she realizes. Just like the extended warrantee feature she started working on with Tom.

"Well, Project Inversion is our shot to fix some of these things and to engineer our way out of this," Kurt says. "Shannon will find out what the Phoenix teams need help on. How about us? If someone gave us the authority, and we were given infinite resources for one month, what would we do?"

Maxine smiles as she hears the suggestions fly fast and furious. They start making a list: Every developer uses a common build environment. Every developer is supported by a continuous build and integration system. Everyone can run their code in production-like environments. Automated test suites are built to replace manual testing, liberating QA people to do higher value work. Architecture is decoupled to liberate feature teams, so developers can deliver value independently. All the data that teams need is put in easily consumed APIs…

Shannon looks over the list they've generated, smiling. "I'll post the updated list when I'm done interviewing the teams tomorrow. This is exciting," she says. "This is what the developers want, even if they can't articulate it. And that's something I can help them with!"

It's a great list, Maxine thinks. Everyone's enthusiasm is evident.

"That is indeed a great list, Shannon, which could dramatically change the dynamics of how engineers work," Erik says, sitting down next to Kirsten once again. Maxine looks around, wondering where he came from.

Gesturing at Kirsten, he continues, "But consider the forces arrayed against you. The entire Project Management Office aims to keep projects on-time and on-budget, following the rules and enforcing the promises written long ago. Look at how Chris' direct reports act—despite Project Inversion, they keep working on the features because they're afraid of slipping their dates.

"Why? A century ago, when mass production revolutionized industry, the role of the leader was to design and decompose the work and to verify that it was performed correctly by armies of interchangeable workers, who were paid to use their hands, not their heads. Work was atomized, standardized, and optimized. And workers had little ability to improve the system they worked within.

"Which is strange, isn't it?" Erik muses. "Innovation and learning occur at the edges, not the core. Problems must be solved on the front-lines, where daily work is performed by the world's foremost experts who confront those problems most often.

"And that's why the Third Ideal is Improvement of Daily Work. It is the dynamic that allows us to change and improve how we work, informed by learning. As Sensei Dr. Steven Spear said, 'It is ignorance that is the mother of all problems, and the only thing that can overcome it is learning.'

"The most studied example of a learning organization is Toyota," he continues. "The famous Andon cord is just one of their many tools that enable learning. When anyone encounters a problem, everyone is expected to ask for help at any time, even if it means stopping the entire assembly line. And they are thanked for doing so, because it is an opportunity to improve daily work.

"And thus problems are quickly seen, swarmed, and solved, and then those learnings are spread far and wide, so all may benefit," he says. "This is what enables innovation, excellence, and outlearning the competition.

"The opposite of the Third Ideal is someone who values process compliance and TWWADI," he says with a big smile. "You know, 'The Way We've Always Done It.' It's the huge library of rules and regulations, processes and procedures, approvals and stage gates, with new rules being added all the time to prevent the latest disaster from happening again.

"You may recognize them as rigid project plans, inflexible procurement processes, powerful architecture review boards, infrequent release schedules, lengthy approval processes, strict separation of duties…

"Each adds to the coordination cost for everything we do, and drives up our cost of delay. And because the distance from where decisions are made and where work is performed keeps growing, the quality of our outcomes diminish. As Sensei W. Edwards Deming once observed, 'a bad system will beat a good person every time.'

"You may have to change old rules that no longer apply, change how you organize your people and architect your systems," he continues. "For the leader, it no longer means directing and controlling, but guiding, enabling, and removing obstacles. General Stanley McChrystal massively decentralized decision-making authority in the Joint Special Operations Task Force to finally defeat Al Qaeda in Iraq, their much smaller but nimbler adversary. There the cost of delay was not measured in money, but in human lives and the safety of the citizens they were tasked to protect.

"That's not servant leadership, it's *transformational* leadership," Erik says. "It requires understanding the vision of the organization, the intellectual stimulation to question the basic assumptions of how work is performed, inspirational communication, personal recognition, and supportive leadership.

"Some think it's about leaders being nice," Erik guffaws. "Nonsense. It's about excellence, the ruthless pursuit of perfection, the urgency to achieve the mission, a constant dissatisfaction with the status quo, and a zeal for helping those the organization serves.

"Which brings us to the Fourth Ideal of Psychological Safety. No one will take risks, experiment, or innovate in a culture of fear, where people are afraid to tell the boss bad news," Erik says, laughing. "In those organizations, novelty is discouraged, and when problems occur, they ask 'Who caused the problem?' They name, blame, and shame that person. They create new rules, more approvals, more training, and, if necessary, rid themselves of the 'bad apple,' fooling themselves that they've solved the problem," he says.

"The Fourth Ideal asserts that we need psychological safety, where it is safe for anyone to talk about problems. Researchers at Google spent years on Project Oxygen and found that psychological safety was one of the most important factors of great teams: where there was confidence that the team would not embarrass, reject, or punish someone for speaking up.

"When something goes wrong, we ask 'what caused the problem,' not 'who.' We commit to doing what it takes to make tomorrow better than

today. As Sensei John Allspaw says, every incident is a learning opportunity, an unplanned investment that was made without our consent.

"Picture this scenario: You are in an organization where everyone is making decisions, solving important problems every day, and teaching others what they've learned," Erik says. "Your adversary is an organization where only the top leaders make decisions. Who will win? Your victory is inevitable.

"It's so easy for leaders to talk about the platitudes of creating psychological safety, empowering and giving a voice to the front-line worker," he says. "But repeating platitudes isn't enough. The leader must constantly model and coach and positively reinforce these desired behaviors every day. Psychological safety slips away so easily, like when the leader micromanages, can't say 'I don't know,' or acts like a know-it-all, pompous jackass. And it's not just leaders, it's also how one's peers behave."

A bartender walks up to Erik and whispers something in his ear. Erik mutters, "Again?" He looks up and says, "I'll be right back. Something requires my attention," and walks away with the bartender.

They stare at Erik walking away. Dwayne eventually says, "He's so right about the Third and Fourth Ideal. What can we do about the culture of fear that's all around us? Look at what happened to Chad. He tried to do the right thing and got fired. I probably have more reasons to dislike Chad than any of you—those rolling network outages during the day drove me crazy. But firing Chad doesn't do a damned thing to make those outages less likely in the future.

"I did some asking around to find out what actually happened," Dwayne continues. "Apparently, Chad had worked four nights in a row, in addition to working his normal daytime hours, to support the store modernization initiative. When I asked why, he told me he didn't want the store teams to get dinged on their status reports because of him."

Kirsten raises an eyebrow. Dwayne continues, "His manager kept badgering him to go home, he finally went home on time on Wednesday. But he was back online at midnight because he didn't want to let the store launch team down. He was so worried about all the work piling up, in tickets and in the chat rooms, he wasn't sleeping through the night anymore.

"So he comes into work early on Thursday morning, still tired from all those late nights, and he takes on an urgent internal networking change that needed to be made," he says. "He opens his laptop, and there's like

thirty terminal windows open from all the things he's working on. He types a command into the terminal window and hits enter. And it turns out, he typed it into the wrong window.

"Blam! Most of the Tier 2 business systems become inaccessible, including Data Hub," he says. "The next day, he's fired. Does that seem right to you? Does that seem fair and just?"

"Oh, my God," Maxine blurts out, horrified. She knows exactly how this feel. She's done it several times in her career. You type something, hit enter, and immediately realize you've made a huge mistake, but it's too late. She's accidentally deleted a customer database table thinking it was the test database. She's accidentally rebooted the wrong production server, taking down an order entry system for an afternoon. She's deleted wrong directories, shut down wrong server clusters, and disabled the wrong login credentials.

Each time, it felt like her blood turned to ice, followed by panic. Once, earlier in her career, when she accidentally deleted the production source control repository, she literally wanted to crawl under her desk. Because of the OS it ran on, she *knew* no one would ever know it was her. But despite being afraid to tell anyone about it, she told her manager. It was one of the scariest things she had done as a young engineer.

"That really, really sucks, Dwayne," says Brent. "That could have been me . . . Seriously, every week I'm in situations where I could have made that same mistake."

She says, "It could have been any of us. Our systems are so tightly coupled around here, even small changes can have a catastrophic impact. And worse, Chad couldn't ask for help when he obviously needed it. No one can sustain those insanely long working hours. Who wouldn't start making mistakes if you can't even sleep anymore?"

"Yes!" Dwayne exclaims. "How did we get into this position where someone is so overworked that they're working four nights in a row? What sort of expectations are being set when someone can't take a day off when they need to? And what sort of message are we sending when the reward for caring so much is that we fire you?"

"An excellent point, Dwayne," Maxine hears Erik say, once again rejoining them at the table. "You'd be surprised how deeply this sense of injustice would resonate with Steve. You'd know that if you've spent any time on the manufacturing floor."

"How so?" Maxine asks. She's spent plenty of time working with the plant floor personnel.

"Did you know that when Steve signed on as the COO and VP of manufacturing, he made it contingent upon the company publicly targeting zero on-the-job workplace injuries? He was almost laughed out of the room, not just by the board of directors but also by the plant personnel and even the union leadership," Erik said, smiling. "People thought he was naive, and maybe even a bit addled in the head. Probably because a 'real business leader' would want to be measured on profitability or due-date performance. Or perhaps quality. But safety?

"Rumor was that Steve told Bob Strauss, who was CEO at the time, 'If you can't depend on the manufacturing workforce to not get hurt on the job, why should you believe anything we say about our quality goals? Or our ability to make you money? Safety is a precondition of work.'"

Erik pauses. "Even these supposedly enlightened days, leaders rarely talk like that. Steve had closely studied the work of Sensei Paul O'Neill, the legendary CEO of Alcoa in the 1980s and 1990s, who prioritized workplace safety above all else. His board of directors initially thought he was crazy, but in the fifteen years of his tenure as CEO, net income increased from $200 million to $1.5 billion, and Alcoa's market cap went from $3 billion to $27 billion.

"Despite that impressive economic performance," Erik continues, "what Sensei O'Neill talks about most is his legacy of safety. For decades, Alcoa has remained the undisputed leader of workplace safety. When he joined, Alcoa was rightly proud of having an above-average safety record. But with two percent of their workforce of 90,000 employees being injured every year, if you worked your entire career at Alcoa, you had a forty percent chance of being hurt on the job.

"Alcoa has far more hazardous working conditions than in your manufacturing plants," he says. "In the aluminum business, you have to deal with high heat, high pressure, corrosive chemicals, end-products weighing tons that need to be safely transported…

"Sensei O'Neill famously said, 'Everyone must be responsible for their own safety and the safety of their teammates. If you see something that could hurt someone, you must fix it as quickly as possible.' He told everyone that fixing safety issues should never be budgeted—just fix it, and they'd figure out how to pay for it later," Erik continues. "He gave out his home

phone number to all plant workers, telling them to call him if they ever saw plant managers not acting quickly enough or not taking safety seriously.

"Sensei O'Neill tells a story about his first workplace fatality," Erik continues. "In Arizona, an eighteen-year-old boy died. He jumped into an extrusion machine trying to clear a piece of scrap material. But when he did, a boom released, swinging around and killing him instantly.

"This boy had a wife who was six months pregnant," Erik says. "There were two supervisors there. Sensei O'Neill said, they watched him do it, and probably trained the boy to do exactly what he did.

"In the end, Sensei O'Neill stood up in front of the entire plant and told everyone, 'We killed him. We all killed him. I killed him. Because I clearly didn't do a good enough job communicating how people must not get hurt on the job. Somehow it was possible that people thought it was okay for people to get hurt. We must all be accountable for keeping ourselves and everyone safe.'

"As he later said, 'Alcoans were extremely caring people. Every time people were injured, they mourned and there was always lots of regret—but they didn't understand that they were responsible. It had become a learned condition to tolerate injuries.'"

Erik pauses to wipe a tear from his eye. "One of Steve's first actions was to incorporate Sensei O'Neill's True North of *zero workplace injuries* into every aspect of manufacturing plant operations here at Parts Unlimited. One of his first acts on the job was to institute a policy that every workplace injury must be reported to him directly within twenty-four hours, along with remediation plans. What a magnificent example of the Third Ideal of Improvement of Daily Work and the Fourth Ideal of Psychological Safety."

As Erik stares at the wall for several moments, Maxine suddenly realizes why Steve talks about workplace injuries at every Town Hall. He knows he can't directly influence everyone's daily work. However, Steve can reinforce and model his desired values and norms, which he does so effectively, Maxine realizes.

Maxine stares back at Erik. She's never even talked to Steve. How could she possibly do what Erik suggests?

From: Chris Allers (VP, R&D)
To: All Dev; Bill Palmer (VP, IT Operations)
Date: 11:10 p.m., September 25
Subject: Project Inversion: feature freeze

Effective immediately, as part of Project Inversion, there will be a feature freeze for the Phoenix Project. We will make a maximum effort for thirty days to increase the stability and reliability of Phoenix, as well as all supporting systems.

We will suspend all feature work so that we can fix defects and problematic areas of code and pay down technical debt. By doing this, we will enable higher development productivity and faster feature throughput.

During this period, we will also suspend all Phoenix deployments, except for emergency changes, and our Ops teammates will be working on making deployments faster and safer and increasing the resilience of our production services.

We are confident that doing this will help the company achieve its most important strategic goals. If you have any questions or concerns, please email me.

Thank you,
Chris

From: Alan Perez (Operating Partner, Wayne-Yokohama Equity Partners)
To: Sarah Moulton (SVP of Retail Operations)
Date: 3:15 p.m., September 27
Subject: Strategic Options **CONFIDENTIAL**

Sarah—in confidence…

Good meeting yesterday. I'm glad that I had the opportunity to share with you my philosophy of creating shareholder value—in general, we favor "value" and operational discipline over "growth." Our firm has created outsized returns by investing in companies like Parts Unlimited. My plan would create fantastic and consistent cash flow, at a rate higher than most people even think possible. At other companies, we've created considerable wealth for investors (and company executives).

As promised, I'm introducing you to several CEOs in our portfolio of companies whom you may be interested in talking to. Please ask them about how we helped increase shareholder value.

Sincerely, Alan

PS: Did I understand correctly that there's now a "feature freeze" for Phoenix? Doesn't that put you even further behind? And now what do you do with all those new developers you talked about last time? And what will they work on?

CHAPTER 9

• *Monday, September 29*

On Monday, there's a spring in Maxine's step as she walks into the building. And it's not because of the Dockside meeting. It's because it's Testing Day! Her code will finally be tested and put into production.

She carries five boxes of Vandal Doughnuts she bought on her way in. She even got some of their special "cronuts," a crazy hybrid of a croissant and donut, her favorite.

She's feeling so good that she wonders whether the aroma of sixty freshly made donuts might be elevating her blood sugar levels. *What a great way to break the ice with the people who will be testing her code*, she thinks. It's always easier to make new friends when you bring tasty treats.

Everywhere she walks, people ask her, "Are they for me?"

She happily yells back, "Nope, it's for Testing Day!"

Putting down all the donuts on a table near her desk, she slings her bag by her chair. Tom is already there, editor open, typing away.

"Hurray, it's Testing Day!" Maxine announces happily. "At long last."

"You are very strange," Tom says, not even looking up from his monitor. He sniffs the air. "Wow, are those Vandal Doughnuts?"

"Yes, to celebrate Testing Day!" she responds with a big smile. "I think it's super exciting to finally see whether all our changes actually work or not," Maxine says. "So, when do they start? Can we go watch?"

Tom turns to face her, looking at his watch. "I suppose they probably start today. But it's not just our changes. They're testing changes to all the other big chunks of Phoenix—ours are just a fraction of what they need to do. They might not even get to ours today."

"What!?" Maxine interrupts, shocked. She had been waiting all weekend for this! "Can we see where we are in line? Can we help? In fact, where do the QA people sit? I bought all these donuts for them!"

Tom looks surprised. "Well, I've met a bunch of them—some of them are offshore, some are on-site, but I haven't talked with them directly in

a long time. We usually meet the QA manager at the end of next week, when they present the testing results."

"Next week? Next week?!" Maxine's jaw drops. "What are we supposed to do in the meantime? Hey, can we follow along as they work? We'll get notifications on our feature tickets, right?"

"Uh, not exactly," Tom says, frowning. "The QA team uses a different ticketing system. It does their scheduling and reporting and manages all their test cases. We don't have access to it—at least, non-mangers like us don't. After two weeks, they'll send us a spreadsheet with a list of all the defects they've found, labeled with our feature ticket numbers. We'll look through them, copy that information into our ticketing system, and then we'll fix anything that needs fixing."

"…and then?" Maxine asks, fearing the worst.

"QA rolls up everyone's fixes and tests again," replies Tom.

"So, let's suppose that all our changes work perfectly—when would be the soonest that our customers actually get to use what we wrote?" she asks.

Tom starts counting on his fingers. "Two weeks for another testing cycle. Then they open up a ticket with Operations, requesting that they deploy the changes into production. Sometimes it takes a bit of time for them to work it into their schedule…that could take another three weeks." He looks at his fingers. "So that'd be seven weeks from now."

Maxine crumples forward, groaning as she buries her head in her hands, her forehead on the table.

I am so naive, she thinks. Head still on the table, she asks, "And during that whole time, we're just supposed to work on more defects?"

"Yep," she hears Tom say. "You okay, Maxine?"

"Yeah, I'm fine," she says, trying to not feel depressed. *This is the opposite of the Second Ideal*, she thinks. *We're just a stupid feature factory, pushing out widgets that customers may or may not care about. Work is not fun and full of joy, like I know it should be. There is no flow of features, there is no feedback, and there certainly isn't any learning*, she thinks.

She hears Tom ask, "Umm, can I have one of those donuts?"

"No," Maxine responds. Then she has an idea. She lifts her head and looks up at Tom, smiling. "But you can help me deliver them to the QA people."

Finding the QA people was much more difficult than she thought it would be. Tom hadn't been in the same room with one in over a year. His main interactions with them were through formal rituals—he turned over the code and waited for the list of fixes in a spreadsheet, rinse and repeat until the team got the formal acceptance letter that the release was ready for production.

Of course, it was never that easy. There would be all sorts of escalations up the Dev and QA management chains because of disagreements and problems. Is this defect Priority 1 or Priority 2? When developers couldn't reproduce the problem, they'd close the defect, only for it to be reopened by QA later. Or if QA couldn't reproduce the fix, it would bounce back to Dev.

Maxine and Tom stop at Kurt's desk to tell him about their quest. "Those are a lot of donuts. What a great way to make friends," Kurt says. "You tagging along too?" he asks Tom.

"Absolutely," he responds. "I've always wondered where our work goes after we're done with it. It's always felt like flushing the toilet—you put your code in the toilet bowl, press the lever, and it disappears from sight…"

Kurt snorts. "Given the quality of the code we've seen in Phoenix, your metaphor seems pretty appropriate. Roy is the QA Manager assigned to Data Hub. And he'll be tied up for at least ninety minutes," he says, picking up his phone, tapping out a message to someone. "Go over to Building 7 to deliver those donuts while he's preoccupied. I'll connect you with Charlotte, who is, or was, William's assistant. She's like the mother hen for all the QA people."

Kurt finishes typing. "She's expecting you. I think three boxes will be enough for the Data Hub team. Ask Charlotte how to most strategically deploy the remaining two boxes," he adds with a smile.

"She'll get a conference room for you and bring the Data Hub QA team by," Kurt says. "You'll get a chance to meet all of them. And maybe you'll find some people who are looking for help."

Maxine smiles. This is exactly the support she was looking for. "Thanks, Kurt. We'll go make friends. In fact, how about we get pizza delivered for lunch to give us an excuse to hang out even longer?"

"Perfect," Kurt says. "Tell Charlotte to charge it to my old QA department code. With William gone, I'm sure it'll take them a while to shut

that down. Let's take advantage of it," he adds with a grin. "But before you go . . . can I have a donut?" he asks.

"No. Sorry, they're for our new QA friends," Maxine says.

Maxine and Tom walk across the courtyard to Building 7 with the boxes of donuts. They greet the security guard. When Maxine puts her badge on the electronic card reader on the side of the closed door, the light stays red.

Maxine swipes her card again, but again, red light. Maxine sighs. She hadn't expected to not be able to get into the building.

"Interesting that developers can't get into the QA building," Tom says. "Does that mean QA people aren't allowed into the Dev building?"

Maxine is about to call Kurt when she hears the door open. A cheerful, elfin woman bustling with energy greets them. Right away, Maxine finds her irresistibly likeable.

"You must be Maxine? And Tom? Kurt's told me so much about you both! Come on in . . . I was pretty sure that your badge wouldn't work in this building. It's only a matter of time before Kurt's stops working too. We're all so happy for him—well, most of us are, that is. Many of us always knew that he was destined for bigger and better things than managing a QA team."

Charlotte's comment about "destined for bigger and better things" makes QA sound like an underclass. *Like Kurt had escaped some sort of ghetto*, Maxine thinks.

"What a wonderful idea to throw a party for QA! I'm not sure anyone's ever done that before. Everyone will love it. I reserved the biggest conference room for the whole day—people will swing by whenever they're not in meetings and such. And I also ordered pizza for everyone in the lunchroom." Maxine is impressed that Charlotte has taken care of every detail so quickly. In the conference room, Maxine sees that Charlotte has already written on the whiteboard, "We Appreciate QA!!!" with hearts on either side of the large lettering.

After looking for a moment, Maxine asks if she can make some changes.

"Sure thing," Charlotte is enthusiastic.

Maxine makes her change: "We Appreciate Our QA Team Members!!!"

She then adds the names of Tom, herself, Kurt, and the other five members of the Data Hub development team at the bottom.

"Good idea," Maxine hears Tom say behind her. "I suppose we should invite all of the Data Hub developers to lunch too? Want me to send an email to them?"

Maxine quickly agrees, adding, "We're going to need more pizza…"

"No problem, I'll take care of it," says Charlotte with a big smile.

Over the next couple of minutes, members of the QA team start trickling into the conference room. Maxine introduces herself to each one. She notices that the QA people are demographically a little different than the developers. No one is in their twenties. Maxine wonders if that's because college grads are applying for developer roles instead?

"So, what is this celebration about?" a woman with an Indian accent asks.

"It's Testing Day!" Maxine smiles, delighted to be asked. "I'm so excited that the features we've been working on for weeks are going to be tested. I thought it would be fun to throw a party, so we could meet the people who are doing this important work and to let you know that we'd love to help in any way."

"Gosh, that's really nice," the woman says, returning Maxine's smile. "I'm not sure if that's ever happened before."

Charlotte hollers from the other side of the room, "I've been here for seven years, and I've never seen it happen. This is such a nice idea, Maxine. Let me introduce you to everyone. Purna is one of the QA leads, and these are her team members…"

And then there's silence. Maxine wonders if everyone expects her to give a speech. As the host of the party, maybe she should.

"So, uh, again, thank you so much. We're having pizza delivered for lunch, and the Data Hub developers will be joining us then," Maxine says. "What are you all working on these days?" That's always a good ice breaker.

They tell her about the projects they're working on, which provides a shared context. Then she asks what they find most frustrating about the testing process.

The flood gates open. Their pain points and stories sound all too familiar to her: waiting for environments, environments not completely cleaned up, the problems that cascade when something goes wrong, the inability to determine whether problems were caused by errors in the code or something wrong with the environment.

Suddenly, she and Tom have lots of common ground and things to talk to them about. After all, everyone loves complaining about work. Maxine starts taking notes. And thus, the party gets fully underway.

After ninety minutes, it's clear to Maxine that it isn't really about Dev versus QA—instead, it's about how Phoenix business requirements change so often, which almost always requires urgent code changes. This reduces the time available for testing, resulting in poorer quality, as evidenced by the latest Phoenix disaster.

Everyone understands that change is a part of life, but the Phoenix Project seems ill-suited to this rapid pace of change. And everyone, absolutely everyone, expresses real concerns about the decreasing quality of the Phoenix Project and the potential consequences to Parts Unlimited. Someone says, "In his Town Halls, Steve talks about what's needed from us. We're just not delivering what's needed—and when we find something wrong, there's not enough time to fix it."

There's lots of enthusiasm about the feature freeze, despite the ambiguity of what exactly will be frozen. People are excited that this indicates a real change of values from the top, and it's definitely for the better. However, many managers are convinced they're somehow exempt from the freeze.

Eventually, the party moves to the lunchroom and fifteen large pizzas of every imaginable variety are on the tables. The smell makes Maxine hungry—she's feeling a bit jittery from eating all those donuts; her heart is racing, and she's even sweating a bit. As a borderline hypoglycemic, she needs to eat some protein soon or she'll have a headache and a serious blood sugar crash.

By now, scores of QA people have arrived. Maxine doesn't really know who is supporting Data Hub and who isn't, but she doesn't care. The goal is to make friends today. A bouncer at the door would have squashed that.

Maxine finishes her second slice of pepperoni pizza and throws away the paper plate in the compost bin. After carefully washing her hands, she follows Purna to her desk. Purna has happily agreed to show Maxine how she performs her daily work. Maxine sees rows of desks packed more closely together than in the developer area, but not as dense as the help-desk area where she met Derek.

On Purna's desk are two large monitors, pictures of her with her kids, and a bottle of eight-year-old single malt scotch. Maxine gestures at it, "Your favorite?"

Purna laughs, saying, "Not even close, but good enough for celebrations here. You need it working on the Phoenix Project." She moves windows around her screen and shows Maxine the release project that she has created in the QA ticketing tool.

At last, Maxine thinks. She's been dying to see the QA team workflow. When Maxine sees the tool, she is momentarily taken aback.

"Is that IE6?" Maxine asks, hesitantly. The last time she saw that version of Internet Explorer was in Windows XP.

Purna smiles, as if she's used to having to explain this to people. "Yes. We've been using this tool for over a decade, and now we have to run the client inside of an old Windows VM. It contains all our test projects and runs some of our automated functional tests. There's thousands of test plans that we've built over ten years in here."

"But IE6?" Maxine asks.

"The vendor has an upgrade that supports a modern browser, but it requires upgrading the server it runs on," Purna says. "We finally got budget for it, but we're still waiting for Operations to provision it."

This isn't the first time Maxine has seen people having to use old versions of Internet Explorer. They had some plant support systems back at her old position where the vendor went out of business long ago. They've managed to migrate off of those systems, with one exception. They had to create a completely air-gapped network called "6.6.6.6" for a mission-critical server. It was running on a known vulnerable version of SunOS, which was completely unpatchable.

Good times, she thinks.

As Purna gives her a tour, Maxine sees that despite its age, the QA workflow application is very well-organized and functional.

Purna pulls up a network share with over two hundred Word documents containing test plans. When Maxine asks, she opens up a couple at random. Some describe the test procedure to test a given user scenario: go to this URL, fill out this form with these values, click this button, verify correct values at this other URL…

Other documents describe the test plan for input validation, to ensure that every field in each form rejects any non-conforming input.

Reading these brings back memories from decades ago for Maxine. After all, her first job was doing software QA. Great QA requires a perverse and sometimes sadistic intuition for what will cause software to blow up, crash, or endlessly hang.

Maxine once heard a joke: "A QA engineer walks into a bar. Orders a beer. Orders zero beers. Orders 999,999,999 beers. Orders a lizard. Orders negative one beer. Orders a 'sfdeljknesv.'"

Great QA people are notoriously good at breaking other people's code. They'll fill in forms with thousands of characters, unprintable Unicode characters and emojis, put negative numbers into date fields, and other wildly unexpected things. As a result, programs crash or wildly malfunction, usually causing developers to slap their foreheads, marveling at the diabolical test case.

Some of these injection errors can be used by hackers to gain complete access and potentially grab all the data from the entire system. This is what led to some of the worst personally identifiably information (or PII) thefts in history.

Finding these errors and vulnerabilities is very important work. Maxine feels awful that Purna and her team must manually perform them. During the next two weeks, how many times will they clean up from the previous test, bring up a fresh Phoenix application state, go to the right URL, type the same information into the fields . . . ?

Purna shows other tests to check whether the feature actually worked as designed. Often, this means connecting it with other business systems in an integrated test environment carefully engineered to resemble what is currently running in production.

Maxine keeps thinking about how many of these great tests could be automated. It would liberate the QA teams from work that is tedious, time-consuming, and error-prone, and free up their genius to find more ways to break the code.

Moreover, these automated tests would be run every time a developer checks in code, giving fast and immediate feedback that Maxine and other developers love. They could find mistakes right away and not make the same mistake day after day, week after week.

Maxine doesn't say any of this aloud. The last thing a QA person wants to hear from a developer they just met is their ideas on how to automate their job away.

Nearly an hour later, Maxine is still eagerly taking notes. Purna is being so nice, but Maxine feels impatient. She's here to see her code run and help the QA team make sure it's correct.

Purna turns to her and says, "Well, that's about all we can do. The QA1 environment still hasn't been reset. We're waiting on a customer test data set from the Data Warehouse team, and the Phoenix Dev teams still haven't started their merge… until we get those, there's really nothing we can do."

"The developers haven't started merging?" Maxine says, her heart sinking. "How long does that take?"

"We usually get something within two or three days… I know they're always trying their best…" says Purna.

Maxine groans. In her short tour of duty with the Phoenix Project, she's experienced almost every side of a ticket transaction. To get a Phoenix build going, she opened up tickets to what felt like half of the QA and Operations organization and waited helplessly while they worked to get the things she needed.

She enjoyed working on some Data Hub development tickets, because they represented things that their customers needed. They marked them for QA as "ready to test," and now that she's hanging out in QA, she discovers that QA is waiting on work still being done back in Dev.

And they're also waiting for other people to vacate the test environments they need to use. They're waiting for Ops to provision a server so they can upgrade their test management system. And they're waiting for refreshed test data from the Data Warehouse team. Where does all this madness end?!

"What exactly do you need from the Data Warehouse team?" she asks, reminded of Brent's data problems during the Phoenix release and Shannon describing her five frustrating years on that team.

"Oh, everyone waits for them," she says. "They're responsible for getting data from almost everywhere in the company, and cleaning and transforming it so that it can be used by other parts of the business. We've been waiting almost a year for anonymized customer data, and we still don't have test data that includes recent products, prices, and active promotions. We always get pushed down the priority list, so our test data is years old."

Interesting, Maxine thinks. Data Hub is actually how the Data Warehouse team receives most of its data.

More and more dependencies, as far as the eye can see, she thinks. *There is no place in this whole screwed-up system where you can get anything done.* It doesn't matter whether you're creating the ticket, processing the ticket, waiting on the ticket, or working the ticket. It doesn't matter. You're trapped in a web of dependencies, completely unable to get anything done, no matter where you are.

"This sucks," Maxine says finally, sighing loudly. "I just hate all this waiting…"

"Actually, it's much better than it used to be," Purna says. This makes Maxine feel even worse.

Purna stares at Maxine. Maxine feels like she needs to explain why she's upset: "I'm pissed off that Dev doesn't have their crap together. We've got to do better than this," Maxine finally blurts out.

"I know we sometimes contribute to the problem too," Purna says.

Oh, great, Maxine thinks. *On top of everything else, we're also suffering from Stockholm syndrome.*

Just then, she hears a loud commotion in the lunchroom behind her. A tall man in his early fifties is angrily yelling at Charlotte and pointing at the pizzas and then at Tom and the other Data Hub developers.

Uh oh. That must be Roy, she thinks. She quickly texts a note to Kurt:

> Roy is here. You'd better come!

"Excuse me," she says to Purna, walking quickly to the kitchen.

"…we can't have people here disrupting and interfering with our work. Of course, I certainly appreciate the gesture, but this should have gone through me. Next time, get my approval first, Charlotte!"

"Oh, but it just seemed like such a nice gesture," Charlotte responds. "I mean, donuts and pizza! No one has ever done that for QA before. What a very nice thing for Kurt to do."

"Kurt! Kurt's always up to something. This is just part of some scheme he's hatching," Roy fumes, waving his clipboard at everyone. Watching the scene are about fifteen people standing motionless, eyes wide. Some look frightened; some look amused.

"He's probably put all this food on my department code!" Roy says, turning back to Charlotte. "If so, there's going to be hell to pay."

Maxine strides confidently into the lunchroom, extending her hand. "Hi, Roy. I'm Maxine, one of the developers on the Data Hub team. I'm sorry. This is all my fault. It was my idea to bring in donuts this morning. I just wanted to celebrate Testing Day with you and offer our help."

Roy shakes Maxine's extended hand but looks blankly at her. He finally asks, "To celebrate what?"

"Testing Day," Maxine says simply, unable to stop smiling. Roy's expression is nearly identical to Tom's when she brought up Testing Day to him earlier that morning. "I've had so much fun working on the features for Data Hub that I thought it would be just as fun to offer our help to test the code too."

Maxine points behind her to the conference room whiteboard, which everyone can see from the lunchroom, especially the big, pink hearts that Charlotte drew.

Roy looks at her, speechless. Finally, he releases her hand and says loudly, "Oh, no you don't. I don't know what you all are up to," pointing at Maxine and Tom and five other Data Hub developers who stand out in their T-shirts and hoodies. "I'm pretty sure Kurt is up to no good, as usual. This is probably some empire-building scheme he's running, now that he's trumped up a position for himself in Development. I'll get to the bottom of this, you can count on that."

As Roy turns to go, Maxine wonders how she can reiterate her message of "we come in peace."

As she tries to decide what to do, she sees Kurt stride into the room. "Oh, hello, Roy! I'm so glad you're still here. Sorry for not coordinating with you. We just thought it would be fun to throw a surprise party. QA is the next critical part of the chain, and we want to do whatever we can to help."

At the sound of Kurt's voice, Roy turns around, his face red. "Oh, ho, here he is! I'd like to have a word with you. Right now, please."

Kurt is about to respond when Kirsten walks into the room behind him, saying, "Hi, Kurt. Hi, Roy. Mind if I join you all? Oh, I love pizza."

Maxine is surprised to see Kirsten. Other people are wandering into the lunchroom, watching the drama unfold.

"I'm so glad you could make it, Kirsten." Kurt turns to everyone, "As we were walking here, we were talking about how important the QA

effort is and that QA's concerns deserve a bigger voice. Kirsten, would you mind sharing what you told me? I think everyone here would love to hear it."

"Of course, Kurt," Kirsten says, holding a paper plate with a slice of sausage and pineapple pizza. "As everyone knows, the Phoenix Project is the most important initiative in the history of the company. The disaster two weeks ago was a big eye-opener for everyone, especially at the most senior levels of this company. We have a lot riding on the next release. We have three years of promises that we've made to the marketplace that we're finally starting to fulfill.

"We just announced Project Inversion, the first time we've had a feature freeze to shore up quality," she continues. "This is a demonstration of the commitment, at the highest levels of the company, not just to do the right things, but to do the right things *right*. And getting you a code release on time is part of that. I know Development is often late merging their changes.

"In our meeting with all the Dev and QA leadership, they gave their commitment to deliver you something to test by five today," she says. "We know how important it is that you have something stable to test, and that needs great development processes. Improving those processes will be a part of Project Inversion, as well."

People cheer, the QA staff clapping especially loudly.

"We're in a relay race, and we need to get the baton handed to you," she continues, gesturing expansively with her free hand. "Your work is important, and my job is to help you get whatever you need to succeed. Thank you in advance for all your hard work, and please let me know how I can help." The room erupts in applause again, and Maxine joins in. She's reminded of a fancy party she was at in Chicago, where she saw the mayor of the city address the room. She was amazed at how gifted of a communicator he was, making everyone not only feel comfortable, but appreciated and part of something special.

Kirsten has that gift too, Maxine thinks. She's never seen this side of Kirsten before and is impressed.

The crowd starts to dissipate, and several people make their way to Kirsten. Others approach Kurt, shaking his hand, congratulating him on his new role.

Roy is at the back of the lunchroom, glaring at Kirsten and Kurt.

At that moment, Charlotte appears beside her. "Life is always interesting around Kurt, isn't it? I'm going to introduce myself to Kirsten. I've always wanted to meet her. She's so cool. We've had so many interesting people visit us here in QA today!"

Maxine sees Roy approach Kurt. She inches over so she's just close enough to hear him say, "…This isn't over. You somehow managed to find a patron, but she won't be able to protect you forever. You think you're better than us? You think you can come in here, put on airs, and automate everyone's job away? Not on my watch. I'll make sure we bring you down."

Roy strides out of the room. Maxine looks at Kurt, who has an unconcerned smile on his face. He says to Maxine and Tom, who just joined her, "Well, that was fun. Don't worry about a thing. I saw that coming a mile away."

"Worried?" Tom responds, laughing. "*I'm* not worried about anything. This is more exciting than your average day coding. What's going to happen next?"

"Apparently, developers merging their code in a hurry to make a five o'clock deadline," Maxine says, deadpan.

Tom's smile quickly disappears. "Let's go watch." Kurt smiles.

CHAPTER 10

• *Monday, September 29*

Over the decades, Maxine has tried to explain to non-technical people how frightening code merges are. Her best description is having fifty screenwriters simultaneously working on a Hollywood script when they haven't decided who the main characters are, or what the ending will be, or whether it's a gritty, detective story or a bumbling sleuth with a sidekick.

They break up the writing responsibilities between all the writers, and each writer works on their part of the script in isolation, typing away in Word for weeks at a time. Then, right before the script needs to be finalized, all fifty writers get together in a room to merge all their work back together into a single story.

Of course, any attempt to merge their scripts together is a disaster. There's still no agreement on who the main characters are, there's hundreds of extraneous characters, completely disconnected scenes, and gaping holes in the plot…just to name a few of the problems..

And most of the writers didn't read the memo from the executive producers that stated they were making the story a horror movie with giant undersea monsters due to changing market tastes.

Merging code is equally difficult. Editing code isn't like editing in Google Docs, where all the developers can see each other's changes. Instead, like the scriptwriters, they create private working branches of the source code, their own private copy. Like those scriptwriters, developers may work in isolation for weeks, sometimes even months.

All modern source control systems have tools to automate the merge process, but when there are many changes, their limitations become all too evident. Someone may discover that someone else has overwritten their changes, or that they changed or deleted something that everyone else depended upon, or that multiple people made conflicting changes to the same parts of the code…just to name a few things that could go wrong.

Maxine loves it when everyone merges their changes frequently to the 'master branch,' such as once per day. That way, the size of the changes being merged are never allowed to get too large. Small batch sizes, like in manufacturing, create a smooth flow of work, with no jarring disruptions or catastrophes.

On the other hand, you have what Phoenix developers do—a hundred developers work for weeks at a time without merging, and from what Purna says, it usually takes at least three days to merge. Maxine thinks, *Who would ever want to work that way?*

Maxine walks with Kurt and Purna back to Building 5 to the "merging war room," which she thinks is a very appropriate name. The moment she walks into the room, she's hit with a wall of humid air caused by too many people being packed into a hot, crowded room. Looking around, she says to Kurt with certainty, "I don't care what Kirsten says. There is no way we're going to get a release branch today."

Purna walks to the front of the room and takes out her laptop. On the walk over, Maxine learned that Purna is the integration manager responsible for making sure that all the promised features and defect fixes make it into the QA release branch. Everyone affectionately calls her the "merge boss."

Maxine looks at the printed spreadsheet that Purna gave her. There are 392 Dev tickets to be merged. Each row has a ticket number from the Dev ticketing tool, a description of the issue, a checkbox showing whether it's been merged, a link to the QA test plan, the QA ticket number, and on and on…

Purna is responsible for getting all these changes merged so QA can finally test them as a whole, find and report any problems, and make sure that any reported defects are fixed. It's a big and often thankless job.

Maxine grabs a seat at the back of the room with Kurt. Gathered around the table are about twenty-five developers and managers representing each team with changes to merge. They're bunched up in groups of two or three, with at least one laptop in front of them. Typically, one person is typing at their laptop, with the others looking over their shoulder.

There is a low buzz of frustrated conversation. "Sounds like developers merging," Tom says, grabbing the seat next to her.

"You know the joke—what's the plural of 'developer'?" says Maxine. "A 'merge conflict.'"

Tom laughs, opening his laptop. "I might as well merge all of our changes into the release branch now. I usually don't do it right away, because what's the rush? I mean, look around…It usually takes days for everyone to get their changes in."

He opens up the source code management application, drags and drops a couple of things, clicks here and there, and types something. He says, "Done!" and closes his laptop.

"I usually don't stick around," he says. "We barely have any shared code with the rest of the Phoenix teams. I can't remember ever having a merge issue…"

Maxine nods, thinking again about how strange it is that Data Hub is part of Phoenix.

"Do *you* think these people will be done merging today?" Maxine asks.

Tom laughs, pointing at the large TV in front of the room connected to Purna's laptop. "Four changes have been merged. Five, when you include the one that we just did. That's 387 more to go. At this rate, I think it'd be a miracle if they're done tomorrow. Three days, I'm guessing. At least."

Over the next hour, as people have problems with their merges, more developers enter the room to help. When people no longer have room to stand, they create a second war room across the hallway. One of the managers gripes, "I don't know why we don't just pre-reserve two or three conference rooms. This happens every time."

Maxine sees a Dev lead type into a terminal window on his laptop "git pull." He immediately gets a long error message showing forty-three merge conflicts. Maxine actually recoils from his screen in shock. She wonders how long it will take them to untangle that mess.

Later, when she hears another team talk about bringing printouts of the source code for everyone so they can manually reconcile each change, she almost spits out her coffee.

There's a group of ten people crowded around the TV at the front of the room, studying a code diff from four different sets of changes to the same part of a file.

Seeing the expression on her face, Kurt asks, "What's wrong?"

Speechless, she gestures at all the chaos and disruption around her. "Developers should be solving business problems…Not…this…This is madness."

Kurt just laughs. "For sure. All the Dev managers are complaining about how much of a hassle this is. Some are lobbying to do these merges less frequently—instead of once per month, maybe once per quarter."

Maxine blanches. "You're kidding, right?"

"Nope," Kurt says, genuinely amused by Maxine's reaction. "If it hurts, do it less often. That's their reasoning."

"No, no, no," Maxine says, dismayed. "They've got it all wrong. It hurts because the merge sizes are so large. To make it hurt less, they need to do it more frequently, so that the merges are small and create fewer conflicts."

Kurt laughs again. "Yeah, well," he says, shrugging his shoulders, gesturing around the room.

Maxine doesn't laugh, because she doesn't see anything funny. She looks at her watch. It's nearly four thirty. She looks at Purna's laptop screen. Only thirty-five changes are merged, with 359 more to go. They were only ten percent complete.

At this rate, Maxine thinks, *it'll take them another forty hours of work—a full week away.*

The next day, Maxine is slumped in a chair in the lunchroom, surrounded by pizza. It's almost the end of the second day of the code merge. She stares at the big posted signs everywhere, admonishing, "For Merge Teams ONLY."

Maxine doesn't know why they bother. Over the past day and a half, she's guessing that every developer has been in one of the merging war rooms.

"Maxine, there you are," Kurt says, interrupting her reverie. "Holy cow. Uhh, you look like hell, if you don't mind me saying."

Maxine just gives Kurt a tight-lipped smile. She just doesn't have it in her to explain what she's seen and how much it bothers her.

Maxine knows that code merges are never anyone's idea of a fun time, but she wasn't prepared for what she saw over the last two days.

She's seen managers copy source files from computer to computer on USB drives because their teams didn't want to use the same version control system as everyone else.

She's seen people trying to resolve merge conflicts that were over one thousand lines long, splattered across scores of files.

She's seen people forget to merge in their changes, caught only when Purna reconciled her spreadsheet.

She even saw two teams grapple with a genuine semantic merge conflict—a rare occurrence, usually only found in stories that developers tell to scare each other. It's the result of an automated merge that compiles correctly but wildly changes how the program functions. The worst part was that it was a near-miss. They discovered it almost by accident. Frankly, she's amazed that they caught it when someone said, "That doesn't look quite right." Otherwise it would have escaped into production, where it undoubtedly would have wreaked havoc.

She keeps wondering how many similar errors weren't caught that are now in production, sitting there like ticking bombs ready to explode when that code path is finally executed.

Looking back at Kurt, she says, "I've seen things. Unspeakable things, Kurt. Such waste and needless suffering…No developer should have to go through…this…this…madness!" Again, she's at a loss for words.

"Ah," Kurt says, suddenly looking concerned. "Throw that pizza away and come join the rest of the Rebellion. Shannon just reported out the results of her interactions with the Phoenix Dev teams, and she has a great idea."

Maxine looks down at her hands and sees that she's holding a cold, half-eaten slice of pizza, the cheese completely hardened into a white, greasy slab. She didn't remember eating it.

She throws it away and follows Kurt without saying a word.

Kurt brings Maxine to another conference room, far away from the ongoing merge madness. She sees Tom, Brent, Shannon, and Dwayne gathered around the table. They all smile and wave at her. Shannon stares at her for a moment, but unlike Kurt, politely says nothing about her haggard appearance.

"Maxine, I think you're going to love this," Shannon says. "We've all been thinking about how all the Data Hub changes have been merged.

But in order for them to be tested, we have to wait for everyone else to be done merging too.

"We're thinking about what it would it take to get Data Hub decoupled from Phoenix, so that we can test independently," Shannon continues. "If we can, we have QA people ready to work on our changes right now."

It takes several moments for Maxine to understand what Shannon suggested, her mind still shell-shocked and ravaged from the merge. Then it hits her.

"Yes!" Maxine exclaims. "Yes, that's a great idea. We can't do much for the rest of the Phoenix teams right now, but that doesn't mean that we have to suffer along with them."

Kurt says, "I've been talking with Purna and Kirsten. They've assigned two people to help us get Data Hub tested and certified. As long as the Phoenix merge is still going on, they're ours. In fact, I bet we could get *all our changes tested* before then."

Frowning, Maxine says, "But we still need a test environment to run Data Hub in." She thinks for a moment. "I wonder if we could create a Data Hub test environment to run in our cluster. It would be so much smaller and simpler than the Phoenix environments. That way, the QA group could use them anytime instead of the scarce environments everyone is always fighting for."

"The environments team won't be happy about that," Kurt says, with a smile. "What do you need?"

She looks around. "If I had Brent and Adam's help for two or three days, I think we could get at least a simplified environment running by Monday. I know Brent is on a Phoenix lockdown, but, hey, technically Data Hub is still part of Phoenix, right?"

Suddenly, Maxine is excited again. The idea of liberating Data Hub from the Phoenix Project morass is thrilling.

"The First Ideal," says Brent with a smile.

The next day, Maxine, Brent, and Adam are furiously working around the clock to create a slimmed down environment that they can use to run and test Data Hub. It's a race against the Phoenix merge.

Purna gave a thumbs-up on the plan. Kirsten did as well, saying, "We created all these rules, so we can break them too. Especially if this

permanently eliminates these damned dependencies. Any project manager would jump for joy." That was good enough for Kurt, telling them to go for it, without bothering to get any approvals from further up the chain.

"I'll ask for forgiveness later if we need to," Kurt had said with a smile.

At the moment, Brent is trying to reproduce the environment build that currently only works on Tom's laptop. Meanwhile, Maxine is working with Adam, trying to get the last Data Hub release to work in their trimmed-down Phoenix environment.

She's delighted that they're cracking yet another piece of the build puzzle that has vexed her ever since she was exiled. They're both watching a scrolling terminal window as Data Hub boots, hoping they resolved the last error. They're still watching the log messages scroll by when she hears commotion from the merging war room.

One of the Dev managers yells out, "I need everyone's attention! We've been having intermittent production issues for the last two hours on the e-commerce site. Something in Phoenix is causing incorrect or incomplete promotion pricing to be displayed when users are in the shopping cart. Does anyone know what could be going wrong?"

Good timing for an outage, Maxine thinks as she walks back to the war room. Virtually all the Dev managers are already in there, so it should be pretty easy to figure out which part of the code is causing the problem. It's like having a heart attack at a cardiologist convention—lots of qualified doctors are around.

As she watches, Maxine approves of their disciplined problem-solving. They are efficient, logical, and there's no hint of any blame as they try to replicate the problem on their laptops and systematically think through what could be going wrong.

Ten minutes later, the middleware Dev manager takes the lead. She makes a convincing case that the problem has to be in her area of the code. In takes only another fifteen minutes for her team to generate a fix. "It's a one-line change. We can just push the change to the current release branch," she announces. "Oh, dammit, no we can't…Only the SCM manager can push to this old release branch. We need Jared. Anyone know where he is?"

"I'll go find him," yells someone, who runs out of the room.

"Who's Jared?" Maxine asks Kurt. Kurt rubs his eyes, trying not to laugh.

The middleware Dev manager next to them says in a tired voice, "Jared is the source code manager. Developers aren't allowed access to production. The only time developers can push changes to the release branch is for P1 issues. This is only a P3 issue," she explains. "So, either we need to ask Ops to change it to a P1 issue, which is never going to happen, or Jared needs to grant me temporary access so I can check in our fix."

"And if Jared were here, what would he do?" Maxine asks, knowing what's coming.

The middleware manager says, "He'd take the commit ID of our fix, manually copy it into the release branch, and promote it into production."

"That's it?" Maxine asks.

"Yep," she replies.

Maxine curses under her breath. To her surprise, she's actually angry. Like, actually mad.

A couple of minutes ago, she thought the timing of the outage was fortunate. *That lucky patient*, she had thought. All the best experts in heart trauma who could correctly diagnose the problem and administer emergency treatment just happened to be in the room.

But here at Parts Unlimited, doctors aren't allowed to touch the patient. Well, except if there's a P1 ticket open. But if the patient *isn't* on the verge of death, like right now, apparently only *Jared* can touch the patient. And then Jared just does whatever the doctors tell him to, because, you know, doctors can't touch the patient. Jared isn't a doctor. He's probably just an administrator, just adding and removing users and making sure things are backed up.

"No one can find Jared. I think he might still be at lunch," says the guy who searched for him.

"Oh, for Chrissakes," Maxine mutters under her breath. *It's happening again*, she thinks, remembering how wrecked she felt in the lunchroom yesterday.

Everyone tries to come up with a backup plan because no one can find Jared. Twenty minutes later, Randy shows up declaring nothing can be done, but he's still working on finding Jared.

Everyone nods, going back to merging their changes.

"How is this okay?!" Maxine says loudly, addressing the entire room, no longer able to just watch. "Why aren't developers able to push their own changes into production? Why do we need *Jared* to push the changes? I mean, I'm sure he's really good at what he does, but why can't we just do it ourselves?"

The entire room falls silent. Everyone stares at her, looking shocked. Like she had just belched loudly at a wedding or a funeral. Finally, someone says, "Compliance." And another person chimes in, "And Information Security."

Around the room, she hears people utter other reasons.

"ITIL."

"Change management."

"SOX."

"PCI."

"Regulators."

She looks around. All of these people are capable and responsible. And yet… "Come on, everybody. Those reasons don't make sense at all. I think I know the real reason we aren't allowed to push our changes… they don't trust us. Doesn't that *bother* you?! How can Jared know more about making changes than the developers who wrote them?"

Scanning the room, Maxine sees that only about ten people are remotely troubled by her epiphany.

"Do they think we'll deliberately sabotage the changes? That someone copying and pasting our changes can do a better job than we can?" Maxine knows she's pretty far out on a limb here, but she can't stop herself. "We're almost all developers in this room. Doesn't it bother anyone that we're not trusted enough to push our own changes into production?"

A couple of people just shrug their shoulders. Several others stare back at her as if she's crazy or hopelessly naive.

Maxine knows she hasn't delivered a stirring *Henry V* Saint Crispin Day rallying speech, but she's dumbfounded that people aren't more bothered by this situation. She was hoping someone would yell out, "Hell yes, that bothers me, and we're not going to take it anymore!"

But instead, there's just silence.

We don't even need guards anymore. We love being prisoners so much, we just think the bars are there to keep us safe.

She is about ready to leave when a young man with a ponytail and a laptop underneath his arm enters the conference room followed by two people.

"Oh, no," Maxine accidentally says aloud. *This is Jared?*

He's even younger than the intern who helped her on her first day here. She has nothing against young engineers. On the contrary, her fondest hopes and aspirations lay in the hands of the next generation, and she does everything she can to help them achieve their goals. But it's so difficult for Maxine to think that Jared is more qualified to resolve this outage than everyone else in the room. *If Jared can deploy changes, we should be able to too*, she thinks.

Maxine watches as he sets up his laptop to perform the code push. It takes him ten minutes to get successfully logged in, get the link to the code that needs to be pushed, confirm with everyone that it's actually the right code… Just like in all the movie scenes about coding that Maxine poo-pooed, a crowd gathers behind Jared, waiting in breathless anticipation as he performs his work. When he finally says, "It's in," people clap him on the back.

Maxine rolls her eyes in frustration. She's glad Jared performed the work, but come on, all he did was a bunch of copying and pasting and clicking a button.

When Maxine asks the middleware manager if the problem has been resolved, the manager replies, "Not yet. Now that Jared put it into the last release branch, he needs to work with the Ops people to get it deployed into production."

The patient still hasn't been saved. He needs to be transported to the next department for that. She decides to follow Jared, more out of a sense of morbid curiosity then any sense of adventure.

Four hours after following Jared out of the room, Maxine is dazed and disoriented. Any sense of well-being and excitement that Maxine had working on the Data Hub environments has vanished. And Maxine is missing something else—she is no longer certain about what is good, what is bad, and the processes that govern her world.

She also feels distinctly unwell. *Am I running a fever?*

It all started when she followed Jared two floors down to the ground floor, where Operations resides. In an Ops conference room,

she recognized Wes and Patty, but almost no one else, although they all looked uncannily similar to each other.

The room looked almost the same as the merging war room upstairs. Same furniture, same speaker phone on the table, same projector on the ceiling. But sitting around the table are a completely different group of people discussing the exact same topic as upstairs: how to get this urgent change deployed. Only they're discussing slightly different obstacles: No changes outside of the maintenance windows. ITIL. Security. Change Management. Compliance. Different ticketing system. Same number of fields that need to be filled in, and the same errors when you miss a field. Same escalation processes but different people.

They've pushed an emergency change request to Bill Palmer, the VP of Operations, and Maggie Lee, the senior director of product marketing. And just like upstairs, everyone is standing around waiting for approval.

At five o'clock, someone orders pizza. Maxine follows everyone to the lunchroom, identical to the one upstairs. When she sees the pizza, she almost trips and falls—it's from the same place as the pizza brought in for the merging lunch yesterday.

The same pizza, being eaten by different people, complaining about the same problems. It is then that Maxine starts to really feel sick, the room spinning slightly. *Maybe I'm just hungry?* But the look of the pizza instantly turns her stomach.

Maxine feels like she's replaying the same scene of a movie that she just lived through six hours earlier. *Like some horrible version of* Groundhog Day, she thinks. Like Bill Murray's character, she is doomed to replay the same day over and over. But for Maxine, they keep changing all the actors. First, they're Dev, then they're QA, and they're Ops. But it's all the same.

Up until now, deep down Maxine suspected developers were being imprisoned by a heartless, evil, uncaring bureaucracy. Maybe it was run by Operations or a secret ITIL change management cabal. But after following Jared into the heart of Operations, she sees that Ops is imprisoned by the same wardens as the developers upstairs.

Who profits from all of this? Who benefits by oppressing everyone in the technology organization? Maxine doubts that Chris or Bill are the wardens of this endless sea of prisons. If anything, they're prisoners too.

Maxine throws away her slice of pizza before even taking a bite. Back in the conference room, Wes announces the urgent change was just approved. Maggie (the person who needed to approve the change) had missed their first calls because she was at her own birthday party, but she's stepped out now to join the conference call.

It takes forty minutes to push out the change. Maxine watches as the teams rummage in network shares, wiki pages, source code repositories … Patty then confirms that the problem has been resolved.

Wes thanks everyone for staying so late, and people start to disperse. Soon, Maxine is alone in the conference room. The lights start to turn off as motion sensors no longer detect movement. In the dark, Maxine wonders how the oppressive bureaucracy gained so much control.

It's just like Erik said. This is the opposite of the Third Ideal, where instead of improving the processes we work within, we blindly follow them, she thinks. *And now the process has fully imprisoned us, sucking out all the joy from our daily work, pushing us ever further away from the Second Ideal.*

In the darkness, Maxine picks up her phone and texts Kurt and the rest of the Rebellion:

> Anyone else still here? I really need some help. And a drink. Can anyone meet me at the Dockside?

CHAPTER 11

• *Wednesday, October 1*

Kurt is already at the brightly lit Dockside, a couple pitchers and a bottle of wine on the table, when Maxine shows up. She's glad to see him, and those pitchers, because it means other Rebellion members are coming. She's grateful for their company.

Maxine rarely drinks to self-medicate, but as soon as she sits down, she does exactly that. She works through two pinot noirs in short order, despite the fact that she knows she'll suffer tomorrow morning.

But tonight it doesn't matter, because the wine is definitely making her feel better. The combination of sugar and alcohol is helping her combat the jarring and jangling emotions she's been reeling from ever since she followed Jared to the Ops Bizarro World.

As people arrive and sit down, the mood around the table is upbeat. Tom and Brent are working at the table with their laptops open, having made terrific progress on getting Data Hub running in a slimmed-down environment. They can't stay long. They're meeting with the QA team tomorrow morning to get them up and running, with the hopes that they can start their testing soon. Apparently, Purna and her team might swing by later.

Shannon has written up her notes from interviewing the Phoenix teams, identifying nearly ten developers who want to use what the Rebellion has created to address problems they face on a daily basis. And with Project Inversion fully underway, they have the time to do it.

Maxine smiles blearily as she listens to everyone sharing stories. Eventually, Kurt pours another round for everyone and turns to Maxine. "So, what's up, Maxine?"

"Kurt, we are so screwed up." She runs her hands through her hair in frustration.

Maxine tries to explain. She's usually extremely articulate and precise, but as she listens to herself talk, she is acutely aware that she sounds raving mad.

Starting over, she tries hard to convey how much this afternoon disturbed her. "Ever since I was exiled to the Phoenix Project, I've opened up hundreds of tickets, trying to get things done. I've followed those tickets around, seeing where they went. Many of them went to Ops, some went to QA. Then, when I joined the Data Hub team, I opened more tickets. But more importantly, I got to work the other side of those tickets, doing work that people needed. But to get that work done, I had to open more tickets. It's just a giant circle of tickets, Kurt, being created and passed around, over and over again, without end.

"Who did this to us?" she finally asks.

Adam smiles sadly. "We did it to ourselves. Long ago, QA used to be a part of Dev, but when I joined, QA had been made independent. We made a bunch of rules about how we needed to be separate from Dev concerns, you know, to protect the business from all those crazy, reckless developers. Each year, we used anything that went wrong as an excuse to create more and more rules to 'make developers more accountable,' which just slowed us down even more. What makes me so excited about the Rebellion is that we're trying to undo all of that."

Dwayne nods. "Adam's right. In Operations, we did it to ourselves too. It started for all the right reasons—we brought in ITIL processes that created some sense of order, which was infinitely better than the chaos we had before. In Ops, it's worse, because we have so many areas of specialization. Complex work like deployments hit every one of those areas. We have servers, databases, networks, firewalls . . . heck, in the last decade, we've created even more silos, like storage, VLANs, automation teams, virtualization, hyperconverged infrastructure, and who knows how many more.

"And with the modern technology stacks, we need people with deep expertise in containers, logging, secrets management, data pipelines, NoSQL databases. No one can be an expert in all of those!" Dwayne says. "So we need a ticketing system to manage those complex flows of work. But it's so easy for people to lose sight of what the purpose of all this work is. It's why the Rebellion is so important. Look how many people are working late to help with the Data Hub effort."

Everyone around Maxine raises their glasses, yelling out, "Hear, hear! To overthrowing the ancient, powerful order!" Maxine raises her glass as well, but says nothing.

She's often heard that IT is the nerve center of the entire organization, because over the last thirty years almost every business process has been automated through IT systems. But for whatever reason, businesses have allowed their nervous system to become degraded, like multiple sclerosis disrupting the flow of information within the brain and between the brain and the body.

Maxine pours one more drink, but she only takes a sip. Suddenly, she doesn't feel well. It has nothing to do with what she drank. She is definitely coming down with something. She quickly bids farewell to everyone, thanking them for joining her tonight.

When she gets home, she hugs her husband, says goodnight to her kids, and is relieved when she crawls into bed after taking a shower.

Later that night, Maxine starts to sweat uncontrollably, then is overcome by chills and chattering teeth, then back to a fever. She has succumbed to the illness that decimated the ranks of her teammates after the Phoenix release.

That night, she has unending dreams about being trapped in a bureaucracy, handed off from one desk to another, put on hold, asked to fill out more forms, shuffled from one department to another, and put back into another line with more forms to fill. The forms go into vast data warehouses where they are pulverized, turned into a steaming, greasy miasma of comma-separated text files, spiked with random byte-order marks.

She sees the heartless machinery of bureaucracy turning, with helpless people trapped inside the countless gears. She hears their helpless screams, until they fall silent, all energy sucked out of them only to be periodically revived to fill out their timecards.

She pushes mountains of paper tickets up a set of stairs, across a section of cubicles, and down more stairs, doomed to traverse this Square of Sisyphus forever as punishment for the payroll outage.

When she wakes up, the sun is rising. Her pillow is drenched with sweat. Her sinuses and lungs are congested. Her chest hurts from coughing so hard. She can barely move.

She forces herself to get out of bed and shower. The hot steam feels good, but when she gets out, another cycle of uncontrollable sweating and chills begins. She shambles downstairs to eat a piece of toast and drink some water only to realize how much her throat hurts.

Her husband tells her to stay in bed, that he'll make sure the kids get off to school on schedule. Grateful, she mumbles thanks. She makes it halfway up the stairs before having to take a break, eventually crawling back into bed.

Barely able to read the phone screen, she texts everyone to let them know she's unable to come into work. She falls asleep and bolts awake, realizing that she left the office without filling out her timecard. But she's too weak to do anything about it. She finally falls back to sleep, groaning from the aches everywhere.

The next day Maxine can barely get out of bed. She has become one of the walking wounded, joining the ranks of the people unable to do their jobs, whether due to illness, bureaucracy, or being stuck in the Waiting Place.

Desperate for more cold medication, she ventures out and walks around the store's aisles huddled in five layers of clothing, looking for relief. To keep her family from getting sick, she buys a surgical mask as if she were a Japanese office worker on a subway. When her husband sees her wearing it, he just laughs.

By midday Friday Maxine starts to feel mildly better, finally able to stay awake for more than an hour during the day. She hasn't touched her phone in nearly two days and, in fact, has hardly spoken at all, except to her husband in forced monosyllables. Tired of reading novels in bed, she heads downstairs and texts Kurt and Purna:

> Are the developers done merging yet?

Within seconds, Maxine receives a reply from Kurt:

> Hahahahaha! Sorry, no. Maybe Monday. But Data Hub and its environment are almost ready to be tested. QA likely starting this evening! If you want to hear more, call me! Hope you're feeling better.

Maxine dials his number. He doesn't even say hi. "Brent and Tom have been working non-stop. They're close to getting Data Hub running in

the new, smaller environment. All the Data Hub developers are working with the QA team to write automated tests together. Adam and a bunch of developers are leading coding classes, and some of the QA folks are writing tests without any help. You might have seen those tests being checked into the source code repo."

"And Shannon is paving the road for Security," Kurt says. "Environment images will be automatically patched daily, and maybe soon, application dependencies too!"

Maxine tries to smile. She's impressed by how much they've achieved while she was sick. She looks in the chat channel and sees exciting messages about the progress they're making. She loves seeing all the code commits from both the Dev and QA teams.

Without doubt, Maxine knows that the developers will eventually be responsible for testing their own code, with QA taking a more strategic role, coaching and consulting. It means all the automated tests they're writing will soon run with every check-in once they get their centralized build and continuous integration (CI) server going. They're so close!

"This is great," she croaks, which makes her teeth hurt so much that she tells Kurt that she'll see him next week and hangs up.

Maxine crawls back into bed and closes her eyes, thinking about what's next. If they could get Ops to agree, they might even be able to automate deployments to the production Data Hub service. And, although this seems like a long shot, maybe they could even run the production Data Hub services from their cluster.

It would make things so much easier for everyone, even Operations. For starters, they'd be able to test and deploy their changes immediately after working on them, instead of waiting two weeks for the next test cycle.

The real question, Maxine realizes, is which features they should be working on. She wonders what features in Data Hub would be most important for the business. And which business unit they should focus on. Data Hub is unique in that it touches so many areas of Parts Unlimited, each with their parochial needs and priorities.

She tries to go back to sleep, but she keeps thinking about what the highest business value activity is for Data Hub. Curious, she sits up and opens her laptop, bringing up the ticketing system. But instead of opening a new ticket or working on a ticket from someone else, she just looks around. It's the first time she's done that since her exile.

With a couple of clicks, she figures out how to view all the open Data Hub tickets. There are hundreds of them, nicely color-coded based on what business systems they touch. She winces when she sees how many of these tickets are over a year old. No wonder everyone seems so frustrated.

She wonders which features in this backlog are most important for the company. That last part is easy. Steve tells everyone what the company's top priorities are in the Town Halls. Steve and Sarah consistently talk about the importance of helping customers keep their cars running and providing a way for customers to easily buy what they need. Doing this well should increase revenue per customer, average customer order size, and overall revenue and profitability.

With this in mind, she scrolls through pages and pages of features. It's difficult to know what the features really are from the ticket name or reading their contents. They're long on what and how, but not on why.

Maxine eventually notices a term that comes up over and over: "Item Promotion."

She sees a bunch of tickets related to a summer promotion, offering discounted product bundles of battery replacements, air conditioning, and cooling system maintenance items. They were never started. Maxine sighs. Given that it's already fall, the opportunities associated with that campaign have come and gone.

She wonders what the process is for deleting features that are no longer relevant. There's a very real cognitive and spiritual burden of having to carry so many unfulfilled promises forever into the future, where anyone can ask at any time "Where is my feature?"

Curious, she searches for "Winter Promotion" and sees a string of tickets. She starts clicking into them. One ticket marked as complete was to create a SKU for a bundle of wiper blades and ice scrapers. A ticket still in work is to create a discounted price for the bundle.

She sees another pattern just like this, but for winter tires and chains, chains and windshield de-icing fluid, and many more. There's another string of tickets for "Thanksgiving Promotion." Each of these campaigns requires two Data Hub deployments—one to create the new product in the products database, and another to create special promotional pricing in the pricing database.

That means each discounted bundle always takes two months to create. Feeling like she's onto something, she scans the other requested

features in the promotions category. One immediately catches her attention. The ticket was created seven months ago, and the title reads, "Create in one step: new product bundle SKU with associated discount."

Opening the ticket, Maxine reads about how Marketing wants the ability to create and price new SKUs entirely self-service, without having to go through the Data Hub team.

Yes! Exactly as I had thought. The feature description points out that the current process requires almost ninety days for a newly created discount to be available to customers.

The author of the ticket is Maggie Lee, the senior director of products. Suddenly Maxine suspects that Data Hub is sitting on an organizational constraint! She emails Kurt and Maggie. In about five minutes she gets a call from Kurt.

She croaks, "You saw my email?"

"Yep," Kurt says. "I checked out the links you sent me. That's definitely interesting. While you've been gone, I've been trying to figure out who our most important customers are. And I'm also on the lookout for who the heavy hitters are who can give us some air cover as we move Data Hub out of Phoenix. Maggie's name has come up over and over again.

"She works for Sarah, and all the product owners for in-store and e-commerce report to her," Kurt continues. "I'll send you the org chart I dug up. I've already met her admin, and we have a meeting scheduled with her soon."

"Fantastic, Kurt!" Maxine says, but when she smiles, she groans in pain. She's excited to get back to work… when she's healthy again.

Groaning, she hangs up, crawls back into bed, and goes to sleep.

On Monday, Maxine is back at work, discussing Data Hub with Dwayne, Tom, and Kurt in a conference room. Tom is displaying his laptop on the screen in the front of the room. "We worked on this all weekend, making sure it's stable enough to show. Holy cow, I'm excited. We now have the Data Hub environment running entirely in a Docker image, so absolutely anyone can use it. Brent and I based it all off the work Maxine did before she was out sick. Thank you, Maxine!

"Now, instead of waiting weeks to get access to one of the scarce QA environments, you can just run this Docker image on your laptop. It

takes a couple minutes to download, but only a couple of seconds to start up. It's incredible…" Tom says, typing into a terminal window. "With Brent's help, I got these environments wired into our CI server so it can run Data Hub tests. We are finally in the build and test business! We're using it with QA to test the four features we got completed since the last release."

Looking at Maxine, he says, "We have enough capacity on our CI server for anyone who wants to use it. We couldn't have done it without your great work, Maxine."

Tom smiles, shaking his head. "We've been wanting to do something like this for years, but we never had the time. I'm so excited because it will completely change how Data Hub developers spend their time. Everyone can be more productive—they'll be able to develop and test so much faster. And maybe, if a miracle occurs, we can even get these features into production faster too."

Kurt cheers, raising his fists in the air. "Now *this* is an amazing success story! We can finally start showing people the value we can deliver."

Maxine is impressed. This is an awesome accomplishment, and she's proud that Brent and Tom, who she met only weeks ago, were able to get so much done without her.

Kurt frowns. "Actually, I take what I said back. This is a Dev and QA success story. We still have angry business stakeholders who don't have their features. How do you get these features into production?"

"Now that is a totally different enchilada," Dwayne says, shaking his head and drumming his fingers on the table. "Maxine is right. There's a long history of not letting developers push things directly into production. There's entire institutions whose only purpose is to prevent that from happening."

"Who's the most powerful opposition?" Kurt asks.

"Security, most definitely," Dwayne says. "They're going to want to do a security review of the code before it goes into production—that's enterprise policy. And Operations won't be so keen on this, either. And for that matter, there are so many people in the business that have been screwed over by bad changes that most of *them* won't be jumping for joy when you propose this…So, yeah, basically what I'm saying is that everyone is against developers deploying directly into production," he says with a humorless smile.

Maxine nods. "Security is already very familiar with Data Hub. It's not like we're popping an entirely new application on them. We just need them to schedule their review of Data Hub separately from Phoenix."

"Let's meet with them. The worst they can say is no, and it's not like we haven't heard that before, right?" Kurt says. "So besides Security, what is the official process to get approval with Operations?"

Dwayne sighs, not responding for several moments. Finally, he says, "We probably need to go through TEP-LARB."

"Oh," Maxine says. Kurt flinches as if something just stung him. Tom looks around the table, confused. "Is that bad?"

"Well, there are certainly easier things to get through than the TEP-LARB . . . " Kurt says, staring at an empty spot on the table in front of him.

Dwayne says, "Actually, that's a bit disingenuous, Kurt. The truth is there is *nothing* more difficult than getting through TEP-LARB. *Nothing* gets through TEP and LARB. And I should know, I'm *on* the LARB."

"He's right, Kurt," Maxine says. "In all my years here, I've never been able to get anything through. It's a ton of work to even fill out their forms, and I've never seen them actually approve anything. They're the Grand Pointless Council of No."

She looks at Dwayne and says, "No offense."

He quickly replies with a smile, "None taken."

"What is TEP-LARB? And why do they always say no?" Tom asks.

"'LARB' stands for Lead Architecture Review Board," Dwayne explains. "It was a committee created decades ago after a whole bunch of bad things happened in technology, long before I joined the company. Someone decided to create a bunch of rules to make sure anything new was 'properly reviewed,'" Dwayne says, air quoting with his hands.

"It's a committee of committees. There are seven Ops architects, seven Dev architects, two Security architects, and two Enterprise architects. It's like they're frozen in time, still acting like it's the 1990s," he says. "Any major technology initiative needs their sign-off.

"And to pitch anything to them, you first have to fill out the Technology Evaluation Process form, or the TEP," he explains. "Maxine is right. It's a lot of effort. It's about fifty pages these days."

Maxine's eyes widen. Assembling all the information to fill out the TEP was an incredible ordeal the last time she attempted it. And it was

only about half that size then. She asks, "If you're on the LARB, why haven't you made the process easier?"

Dwayne says, "It's a committee. They all think their job is to say no. I'm the lone radical voice in the whole group, and without more kindred spirits, it's impossible for me to drive a yes vote or bring in younger committee members. Trust me, I've been trying."

Kurt drums his fingers for a moment. "Dwayne is right. Any major technology initiative needs to go through TEP-LARB. If we don't, they'll kill our effort before we even get started."

He takes a deep breath. "It pains me to say this, but I think we need to prepare a TEP and pitch the LARB. Just like we're going to have to ask Security, even though we already know they're going to say no too."

Maxine replies, "You know, we could just run Data Hub ourselves. Like, run it completely without any help from Operations, similar to how we ran our own MRP system in my old group. Besides, whenever something goes wrong with Data Hub, it's not like we're not being escalated to eventually."

Everyone looks at Maxine, shocked. In particular, Dwayne and Shannon look scandalized, as if Maxine has just proposed doing something illegal or maybe even immoral. Brent says, "But how? What about Information Security? And compliance? And the TEP-LARB?"

Maxine snorts, recalling that these were exactly the same reasons she heard for why only Jared could deploy code.

She watches Dwayne switch between nodding and shaking his head, as if two wildly opposing views were violently battling inside of his head. "Oh, wow, that would be great. But they'd *never* let us run this type of enterprise-class service ourselves. It's not like we don't have the skills on the team…we'd just have to be responsible for all the data, make sure it's backed up and all. It would be amazing, because we could run it the way we want…"

His voice trails off. Maxine acknowledges his concerns, "That's right. We run our own MRP system, which all our manufacturing plants rely on. That's about as mission-critical as you can get. We do all the backups, preventive maintenance, patching…It's not easy or simple, but it's not exactly rocket science, either. But we've got some of the best Ops people in the company in this room. We can do it."

Brent says, "Hell, yes. I fear nothing in production."

Dwayne slowly nods. "Okay, I'm in. We need this badly, and of course, I *know* we can run everything ourselves just fine."

Kurt smiles wide. "Okay, we have a Plan B. If all else fails, we operate Data Hub ourselves. This will require getting Chris on board, of course."

Maxine chokes on her coffee, but nods with everyone in agreement.

Tom is clearly excited at the idea that everything that he's helped build could soon be running in production. Suddenly, he frowns. "Wait, wait, wait. Does that mean we're all going to have to wear a pager?"

"Yes," says Brent, adamantly. "You build it, you run it."

Tom's excitement visibly fades.

Maxine laughs.

Even Maxine is stunned by how quickly the entire Data Hub Dev team starts using the new environments. Everyone is using one in some fashion or another. They've spread like wildfire. Some are just using the Docker images on their laptops, some are using environments in *vagrant, Git,* or *terraform* configurations, simulating both Dev and test environments.

More importantly, Purna and the QA teams are using the Data Hub environments as well; once features are flagged as "Ready to Test," they're tested within hours. And because tests are being checked in with the code themselves, it's easy for the developer to quickly reproduce and fix the problem.

This new way of working means that many defects and even a couple of features are completely implemented and tested in one day. Because of some reporting requirements that Maxine doesn't quite understand yet, they're still having to use two separate ticketing tools. But the Dev and QA teams are coordinating more closely than ever. In fact, many of the QA teams are sitting side by side with developers each day. Some in Building 5 and some in Building 7.

Watching how the teams work reminds Maxine of her startup days, where everyone was working together toward a common goal. She's amazed at how quickly attitudes changed in Data Hub.

Over the next three days, they close out more fixes as "shippable" than they used to in most months, and everyone's energy is high and enthusiastic. More importantly, Maxine knows that everyone is having fun.

Maxine and Tom finish another issue, marking it as done in the ticketing system. Within a minute, two engineers in the chat room say that they'll review and test it within the hour.

Maxine stands up. "Unless you need me, I'm heading over to visit Cranky Dave and Purna to see how they're doing."

"Hey, I'm coming with you," says Tom, grabbing his laptop. "I'm going to help them test our fixes."

They find Purna with Cranky Dave and another Data Hub developer, all looking closely at something on her monitor. "Whatcha doing?" Maxine asks.

"We're finally testing the surplus inventory functionality," Purna says.

Cranky Dave adds, "It's to support one of the biggest Phoenix initiatives. It's the critical glue that enables Promotions to scan the in-store inventory systems for products that have been sitting on the shelves gathering dust and ship them to one of the regional warehouses, making them eligible to be promoted on the e-commerce site."

"This is the first time we've been able to get it running," Purna says. "This feature was completed over six months ago, but in the last two releases, we couldn't get it to work. The first time it wouldn't connect with the inventory and customer profile systems. The next time, it couldn't connect with the purchasing history systems. Both times there was some environment or configuration problem, but there just wasn't time to figure it out.

"We had to yank this feature out of the release, otherwise it would have made all the other features late too," Purna says.

This is one of the great things about using Docker containers, Maxine thinks. Containers are immutable, unable to be changed after they're created, so if it works in Dev, it will almost certainly work in Test.

Immutability is another concept from functional programming that is making the world a more predictable and safer place, Maxine thinks, smiling.

"We're on step twenty of the eighty-step test," Cranky Dave says, with no trace of crankiness at all, Maxine observes. "I have a good feeling about this. We found one problem earlier today, but I fixed it in less than five minutes, and we kept marching down the list. This is great!"

Even Cranky Dave can't be that cranky when his features are working, Maxine thinks.

He continues, "Every developer knows that in the next interval, they need to write automated tests as they write the feature, not afterward. Which reminds me, we should really have some of the QA team permanently co-located with us. It seems so silly that we have to walk to a different building to pair on small problems."

"Great idea," Maxine says. "Let's give that one to Kurt—dealing with all the politics of office space and facilities is definitely in his bailiwick. But I think it would be terrific."

"By the way, you should check out what Adam and Shannon are doing over in the conference room. I think it'll make you smile," Cranky Dave says, obviously trying very hard not to spill the beans.

Maxine sees Adam and Shannon at a large table with six other Dev and QA engineers around them, laptops open. Adam is projecting his laptop screen on the TV.

"Holy cow, is that what I think it is?" Maxine asks, stopping midstride and staring at the screen.

"If you mean, does this look like a continuous integration server that is doing code builds and automated tests on Data Hub for every check-in, running in the environments that you helped build? If so, then you would be absolutely right," says Adam, a huge smile on his face.

Maxine recognizes the CI tool immediately. Everyone thinks that Data Hub is so archaic and backward, and yet it's now running under continuous integration. They now have better technical practices than most of Phoenix.

"This is incredibly beautiful," Maxine says, feeling misty-eyed. "Everyone from Data Hub has access? When will other teams be able to use this?"

Shannon looks up from her laptop and says, "Everyone from Data Hub is in. And as you know, getting their code into CI was one of the top requests from the Phoenix teams. Adam and I are onboarding the first teams and getting them trained. We're going to do whatever it takes to make sure they're successful. When they're up and running, we've got a line a mile long to be the next onboard," she says.

Maxine savors the moment. This is what she had been hoping since her first day on the Phoenix Project. Every developer deserves to have

this infrastructure to make them productive and a team of experts to help them get up and running.

She looks at the screen and sees that in the last four hours, five Data Hub developers have checked in code changes, and in two cases, the tests failed but were corrected within ten minutes.

Erik would be proud, she thinks. This fast and frequent feedback is such a big part of achieving the Second Ideal of Focus, Flow, and Joy. And all of this was enabled by properly elevating the improvement of daily work over daily work itself, as dictated by the Third Ideal.

"I *love* the idea of co-locating QA and Dev," Kurt says, addressing everyone assembled at Dockside. "Although when I brought this up with some of the other Dev managers, they found the idea quite scandalous," he adds with a smile.

"Right before I came over, I showed some proposed floor plans to the director of Facilities," he continues. "When he saw them, he almost hit the ceiling. I actually think he wanted to confiscate them." Kurt laughs. "He started telling me about all the rules they have about conforming to the space guidelines that HR came up with. Apparently, there are rules about how large the spaces can be based on job titles…"

"Sounds like the USDA rules about the sizes of cow pens," says Cranky Dave. Everyone looks at him. "What? I came from a family of farmers. I had to deal with the occasional USDA audit."

"Great," Shannon says. "He's calling engineers livestock now."

"What's the timeline, Kurt?" asks Adam.

"Nine months," Kurt says.

Maxine hears several people repeat back, "Nine months?!" Some just guffaw.

"Yeah, well…" Kurt says, looking at his notes. "Anything that Facilities does will take forever. We'd have to order the officially supported chairs and desks through Purchasing and schedule the furniture installation with the Facilities staff…"

"Can we just do it ourselves over the weekend?" Dwayne says. "It wouldn't affect anyone outside the team. We could just go to an office supply store or furniture store, buy the bare minimum, and move it into the building. We can use my truck."

"But what happens when Facilities shows up with their badges and says that we don't have the right permits or that we're out of code?" Cranky Dave asks.

Kurt bursts out laughing, "Facilities isn't going to haul it away themselves because no one will give them the budget." He thinks for a moment. "Let's do it. But let's make sure to bring in some furniture that can't be carried away easily…like a couple of bookshelves, and we'll fill them up with books. Maybe a goldfish tank. What do you think?"

Cranky Dave and Shannon laugh. Adam nods thoughtfully, "Possession is nine-tenths of the law. But shouldn't you get Chris's go-ahead first?"

Kurt snorts. "No way. He'd never go for it. Let's just do it."

"Since we have limited space, how about we move some of the QA people to the Dev area, and move some of the developers to the QA area?" Shannon suggests.

"Great idea, Shannon," Maxine says. She's delighted that the team is organizing itself, just as Erik predicted.

CHAPTER 12

• *Monday, October 13*

Over the past week, it's clear to Maxine that Data Hub has figured out how to deliver better value, sooner, safer, and even happier. But it's also clear that a new constraint has emerged. The constraint used to be getting environments—no one could ever get one, and when they did, it was never quite right. Then the constraint became testing, which started only when Development was finished with all their features; finding and fixing defects would take weeks instead of the hours or days it takes now.

Now it is obvious that the constraint is deployment—they are now able to quickly get features production-ready, but they still have to wait weeks for Ops to deploy their code into production.

Figuring out how to get Data Hub into production more quickly is no longer an academic concern. Tom is standing in the front of the conference room with the rest of the Data Hub team. He says, "Maxine, the suspicions that you had while you were out sick were right on target. According to Maggie and all her product owners, creating effective promotional bundles is one of the most critical and urgent Phoenix priorities.

"Kurt, the meeting we have with Maggie is scheduled for tomorrow, and you asked me to study up on this beforehand," Tom continues. "Here's what I've learned: Marketing is constantly experimenting with promotion campaigns to accelerate sales, and this is incredibly important as we approach the holidays, our peak sales season. For example, now that it is snowing in many areas, they want to create a winter promotion bundle: tire chains, ice-melting salts, and window scrapers. They also need to create a discounted price for that bundle, say twenty percent off. They also do promotions to customer segments—if you buy lots of windshield wipers, you may get offered a bundle of wiper fluid and glass defoggers, knowing that you may only need the smallest nudge to buy.

"Conceptually, it sounds easy. But here's all the insane steps they need to go through to get this done: First, every new product bundle needs a new SKU, just like every other item we sell. These SKUs are used by almost

every application in the business: inventory tracking, in our supply chain, our in-store registers, our e-commerce sites, even the mobile apps…

"We only create new SKUs in large batches every six weeks. After the SKUs are created, we also need to push all the application and business logic changes for these new SKUs. These are pushed to every application that needs to know about them. That's often scores of back-end and front-end applications across the enterprise. You might have seen these go out at eight on Friday nights. And when that's done, sometimes we even need to manually refresh certain production databases.

"Here's problem number one: we only create new SKUs every six weeks, which is way too slow. Thanksgiving is a month away, and we're already in danger of not getting those product bundle SKUs created in time.

"And the real truth is that it often takes us much longer than six weeks. We need to change so many applications during those pushes that if anything goes wrong during testing, the entire release is canceled… You can't have new SKUs out there when some of the applications don't know how to deal with them. There's just not enough time to fix these things during the test period, so it's all-or-nothing.

"And on top of that, Promotions also needs to rapidly experiment and iterate to discover which bundles customers are actually interested in and what specific factors lead to an actual purchase. Right now, iterating only once every six weeks is not fast enough to learn and adapt—our e-commerce competitors are doing multiple experiments per day," Tom concludes.

"Wow, that's really incredible," Maxine says, looking at all the boxes he's drawn on the whiteboard. "This is so much like the Phoenix architecture, which makes it so difficult for any team to independently develop, test, and deploy value to our customers. The architecture that supports the Promotions value stream that you've just drawn on the whiteboard shows how it's almost impossible to move any work quickly to where it needs to go."

She gestures at his diagram. "At every step, it's entangled with so many other value streams. We have to synchronize with everyone else's release schedules. If any of *them* can't be released, then we can't be released… It's just crazy."

"It really is. It's frustrating that Data Hub is so tightly coupled to Phoenix and the BWOS," Tom says.

"What's a BWOS?" Maxine asks.

"Oh, that's what we call the . . . you know, the big wad of . . . umm, crap. You know, the hundred-plus applications we connect to," Tom says.

Maxine laughs. "I really think if we could deploy Data Hub changes into production on demand and fully decouple them from the Phoenix release schedule, we'd be so much better off . . . That way, if we have to cancel a release, we could at least try again the next day. With some practice, I bet we could get SKU creation down to one or two days."

"I definitely agree," Tom says. Maxine smiles, satisfied that they're on the right track. *And the value of doing this will be huge*, she thinks.

"This may not be related, but I think it's worth mentioning," Tom says. "We have other huge problems being connected so closely to Phoenix. It sometimes sends us tons of messages that hammer the back-end systems that we connect to. We routinely see massive waves of transactions that cause huge reliability and throughput problems, and sometimes even data integrity problems. Sometimes it's Data Hub crashing, but most of the time, it's the systems that we're calling that are the ones crashing."

Cranky Dave piles on. "Dealing with those systems of record are a huge a pain in the ass. We don't have any real API strategy around here. No one knows what APIs are available, and even if you do, no one knows how you get access to them or deal with their crazy authentication or pagination schemes. Everyone's documentation is crap, and some of these teams don't even care if their APIs don't work as advertised.

"And once you *do* get someone's API working, they'll break it however and whenever they want, especially since they probably don't version their APIs. So transactions start failing for our customers, and they blame *us*," he continues. "They never give you all the data you need, so when you actually have an API change you want, you have to go through all these committees to get them approved!"

"It's enough to drive anyone crazy," Cranky Dave says, exhausted.

"We can definitely stop this madness," Maxine says with certainty.

As promised, the next day, Kurt, Maxine, and the Data Hub team meet with Maggie. As usual, Kurt introduces all the Data Hub team members to Maggie and then asks Maggie to introduce herself.

"Many of you already know me," she says with a smile. "My name is Maggie Lee. I'm senior director of retail program management. What that really means is that I have the P&L responsibilities of all the products and programs behind our stores, which includes physical stores, e-commerce, and mobile. My group of product managers own strategy, understanding the customer and market, customer segmentation, identifying which customer problems we want to solve, pricing and packaging, and managing the profitability of everything in our portfolio."

She continues, "We bridge the business goals and everything that's required to actually achieve them. That includes business operations, business analysts, and product managers, who work with Chris' technical teams. I also have all the operational pieces required to deal with Sales, Finance, and Operations on my team.

"When Kurt said that you had some ideas on speeding up how we create promotion product bundles, you certainly got my attention," she says. "Sorry I couldn't meet even earlier, but as you can imagine, we're all buried with a million things right now. It's definitely a make-or-break quarter for us. For *all* of us."

Maxine is already impressed. Maggie is in her mid-forties and has an unmistakable intensity about her. She is the same height as Maxine and obviously competent. She's a no-nonsense type and always has a serious expression on her face. Maxine suspects that she's Sarah's forebrain, handling the million things required to keep a billion-dollar retail operation going.

Kurt explains what they've been working on.

Maggie looks at Kurt. "So you're telling me that you could enable Marketing to create promotions entirely self-service, like our e-commerce competitors can? And that other changes could potentially be pushed into production on the same day?" Maggie says. "Holy shit, folks. If you can actually do what you say, this might be the miracle we've been hoping for. I'm not prone to overstatements, but I'm not kidding when I say that this could potentially save the quarter. And maybe even the company."

Maxine smiles. "From everything we've studied, it's clear that it's way too difficult and takes too long to get those promotional bundles created. We'd love to fully empower your teams to do what it takes to create new promotions anytime you want and have them pushed out to all your sales channels within hours. There's a lot we don't understand, but conceptually,

we should be able to do it. We just wanted to explore whether this would be valuable to you."

Maggie nods. "Hugely valuable. Look, Steve has promised all the analysts that this holiday season we're finally going to see an uptick in revenue. This is after years of over-promising and under-delivering. Everything hinges on Promotions being able to move the needle on sales. If you really think you can make this happen, we'll do whatever it takes. What exactly is in the way?" she asks.

"Who isn't in the way?" Kurt laughs. "We're meeting with Information Security tomorrow, who could kill this effort on a whim. But the real threat is the TEP-LARB. We've put together a team to create our proposal, but people usually wait six to nine months to get in front of them," Kurt says. "Unless, of course, there's an urgent business need with a powerful sponsor."

Maggie finally smiles, in not an entirely kind way. "For this, I think we need to bring in the big guns."

"Who's that?" asks Maxine, curious who could possibly be a more powerful sponsor than Maggie.

Maggie grins. "Sarah. Take it from me, there is *no one* more effective at busting down inconvenient barriers than she is. She's like a chainsaw, great at cutting down trees."

"…and sawing off hands," Kurt mutters under his breath.

The next day, Kurt and Maxine meet with Ron, the security manager that Shannon introduced them to. They walk into the conference room and see that Shannon has arrived early.

"There's no way I'd miss this," she says, smiling. "I should have brought popcorn."

Ron, who is in his mid-thirties, comes in and sits down. After introductions, Kurt walks him through their idea to decouple Data Hub from the rest of Phoenix.

Ron says, "Interesting idea. I remember when Data Hub was still called Octopus. Why the need for such a big change? It seems to be working well enough now."

Kurt walks through all the reasons, and to Maxine's surprise, Ron nods agreeably. *This is going better than I thought it would*, Maxine thinks.

"That's exciting," he says, agreeably. But then he takes off his glasses and puts them on the table. "Look, I really want to help, but I can't. I'm responsible for making sure applications in my portfolio meet all applicable laws and regulations and that all those applications are secure. Given how radical of a change you're making, I'm afraid we need to perform a complete due-diligence effort. And you simply can't jump the entire queue. You have twenty people ahead of you who would scream bloody murder," he says.

"But the Promotions capability is one of the most important features inside Phoenix, which is the most important initiative for the company," responds Shannon. "Surely you see that ours should have higher priority, right?"

"Yes, but…" Ron says, shaking his head. "I don't set the priority or order of the applications. That comes from the business. You know, our customer."

"But we are 'the business!' And those 'customers' you're talking about aren't our customers—they're our colleagues! Our customers are the ones who actually pay us money!" Shannon says, bright red with exasperation. "Everyone knows what the top goals are. The top priorities are what Steve always talks about in all the Town Halls. What else is more important than getting Data Hub successfully decoupled from Phoenix, so that the Promotions team can meet the holiday sales goals?"

Ron shrugs his shoulders. "If you want to change the order, you'll have to talk to our boss, John."

Kurt closes his laptop, clearly concluding that there's nothing to be gained in this meeting.

"Fine, fine, fine," Shannon says, resigned as well. Then she turns up the charm, saying, "Hey, could you at least give us all of the testing procedures that you'll use to certify Data Hub, along with a list of tools you use to scan it? We'll do our best to replicate it in our automated test suite. Maybe we can generate security audit reports for you on-demand."

"That's a great idea, Shannon," he responds. "Come to my desk and I'll show where all the documentation for the previous audits are."

Maxine loves how Shannon takes every opportunity to get people on their side.

Watching them leave, Kurt looks at Maxine, shrugging his shoulders. "Could have gone worse, I suppose. Maybe we'll fare better with the LARB."

Maxine sighs. She wonders what is required to generate a true sense of urgency. When her dad had a stroke two years ago, she had remarked on all the bewildering processes in the hospital to one of her doctor friends. Her friend responded, "You were lucky. The processes in a stroke ward tend to be superb, because everyone knows that every minute counts and waiting could be the difference between life and death.

"The worst systems tend to be in mental health and elderly care, where there is less urgency and often no patient advocate," she had said. "You can get lost in the system for years. Sometimes even decades."

Maxine remembers what it felt like to be the patient advocate for her dad, doing whatever it took to get him through the healthcare system. Now, she recommits herself to doing whatever it takes to get her teams through the company bureaucracies—the Data Hub team's sense of mission and urgency deserve no less.

Relentless optimism, she reminds herself.

As Maggie promised, they are on the LARB agenda on Thursday. Maxine is amazed and wonders what strings Sarah must have pulled to get them in so quickly. Then she wonders what Maggie had to do to convince Sarah.

Although Maxine recognizes the political necessity of pitching the LARB, she still resents all the time the team spent filling out the TEP—engineers should be writing code, not filling out forms.

It had many valid questions about architecture and security, but some questions seemed dated, reminding her of TOGAF architecture diagrams from decades past, clearly written for a different era: software development and testing phase gates, datacenter specifications, HVAC specifications, Check Point firewall rules (if applicable, of course)…

The Data Hub gang responsible for putting the proposal together is all here, sitting in the back of the room: Tom, Brent, Shannon, Dwayne, Adam, Purna, and Maxine. At one table sit all the senior Dev and Enterprise architects, and at the other table sit all the Ops and Security architects. They are all close to Maxine's age, but mostly white males with a couple of Indian and Asian males in the mix. Maxine notices there's not a single woman at either table.

Data Hub is second on the agenda. First up is a group pitching to re-platform all of their applications from a commercial product onto

Apache Tomcat, a battle-tested and fully open-source Java application server. A younger woman confidently presents their case, which she delivers in a very thoughtful and competent manner. But when Maxine hears that all they're looking for is permission to *use* Tomcat, she's aghast.

Having to ask permission to use Tomcat in production is like asking permission to use electricity—maybe it was once considered dangerous, but now it's commonplace. Worse, it's apparently their second time pitching the LARB. Maxine's heart sinks. If Tomcat is considered risky and controversial, their Data Hub proposal is going to get laughed out of the room.

After twenty minutes of skeptical questions from the LARB, the young engineer throws up her hands in exasperation. "Why are we so frightened of running software we wrote? We're a manufacturing company. We wrote our own MRP and we run it *ourselves*. And for Tomcat, we don't need to rely on a commercial vendor anymore. Some of the largest companies in the world use it. We'd not only save the company hundreds of thousands of dollars a year, we could finally do things that our current vendor won't allow us to do. There's so many capabilities we need to better serve our customers."

Maxine gets goosebumps—not because the presenter mentioned her old MRP system, but because the presenter is clearly a brilliant engineer, fearlessly doing what she thinks needs to be done and not afraid to run things in production.

While the young engineer fields more questions from the LARB, Maxine texts the Rebellion in the chat channel:

> Who is this engineer presenting? She's awesome! She's obviously Rebellion material. We should recruit her.

Adam texts back:

> That's Ellen. She's one of the best Ops people around

Everyone nods at Maxine, agreeing with Adam's assessment.
Brent adds in the chat channel:

> Agreed. I had no idea she was working on this. This is great!

Maxine looks up when she hears Dwayne talking. "You have got to be kidding me. We created TEP-LARB to help evaluate new technologies. Apache Tomcat was created decades ago, and it's either the second or third most widely used application server out there. If we aren't brave enough to run Tomcat, we should get out of the technology game once and for all. I vote yes. And if you don't, I think we all need to hear why."

Someone from the Ops delegation says, "I don't have anything against Tomcat. I'm just not comfortable with our ability to support this given our current staffing levels. We're stretched thin as it is, and while I appreciate that this technology isn't bleeding edge, we still need people to operate and maintain it…"

Dwayne interrupts, "But you *just heard* Ellen say that her team is willing to support it!"

Not even acknowledging Dwayne's comment, the Security architect joins in, "And there's the security risks. I'd like to get a historical report of Tomcat vulnerabilities, how quickly patches were made available, and any reported problems in patching. Maybe then we can come to a decision."

Dwayne mutters, "For crying out loud. *Ellen* is the person who would write the security and the patching guidelines."

"Thank you for your proposal. We look forward to this team presenting the requested information at our next meeting," the Ops architect says, not looking up from the note he's writing.

At the front of the room, Maxine sees Ellen and her teammates slump in exasperation. Ellen closes her laptop, nods respectfully to all the assembled architects, and takes a seat at the back of the room.

Maxine gives Ellen and her teammates the most enthusiastic thumbs up she can manage.

"Next up is Maxine and Adam on the proposal to move Data Hub into a new environment, running on containers, with automated code builds, tests, and deployments?" the Ops architect prompts.

Adam stands up, but after seeing the last presenter, Maxine already knows they're sunk. No matter how well prepared they are, they'll never be able to convince the LARB.

"…and to summarize, the urgent needs of the Promotions team requires us to get Data Hub functionality more quickly to our internal customers.

We need a radically different way to store and retrieve data that allows us to be decoupled from the rest of the Phoenix teams," Maxine concludes. "We've found a set of technologies that can help us achieve that, which have been battle-tested and used in production for over a decade at some of the largest internet properties on the planet: Google, Netflix, Spotify, Walmart, Target, Capital One, and many more. Based on our trials over the past several weeks, we are confident in our ability to support it, and we're willing to support it ourselves if necessary."

Brent, who joined them at the front of the room, adds, "The team supporting Data Hub production would be some of the most experienced people we have in Ops. Personally, I can't even describe how excited I am by this effort. I think these technologies have applicability far beyond Data Hub and could really improve things for almost every application we support. We are willing to be available and responsible to resolve anything that goes wrong. Utilizing these techniques will help every Dev and Ops person at Parts Unlimited."

Maxine sees Kurt smile at the team from the back of the room. Maxine is proud of everyone. It was a solid presentation. She sees Ellen grinning wildly, obviously impressed. But Maxine knows that it's all for naught. The LARB was designed to be an organizational immune system to prevent dangerous changes—they are just too powerful and conservative.

Dwayne tries to rally support. "The LARB should foster innovative efforts like this, picking technologies that can help us win in the marketplace. We used to set the industry direction, making bold choices that left our competitors in the dust. People laughed when we created our own MRP system, saying we were idiots, but history has shown that that was the right thing to do. We were the first company in our industry to use thin clients in our factories, and because of that and hundreds of brilliant technology decisions, we became one of the most efficient and effective manufacturers in the country."

Maxine looks around the room and sees some stirrings of excitement and renewed curiosity among the Dev architects. However, she sees all the Ops and Security architects shaking their heads. One of them says, "Dwayne, I appreciate what you're saying, but we've never done anything even remotely similar to this. It's embarrassing that we can't even support Tomcat—but that shows you exactly why we can't possibly support this.

Unless there's a group willing to volunteer to support this initiative as a side project, I think we need to table it."

Dwayne speaks up, "Hell yes, I volunteer. And I'll grab some people I know who would love to help the Data Hub team with the support responsibilities."

"I'd love to help," says Ellen from the back of the room. "I've been using Docker and the other tools you've mentioned for years. These are competencies we need at this company."

"You're in," Maxine says to Ellen, smiling.

The Ops Chair looks surprised but says, "I appreciate your enthusiasm, but I'm afraid that we cannot support your initiative at this time. Let's pick this up in six months and see if conditions have changed by then."

Hearing enough, Kurt stands up and addresses the room. "Didn't you hear the business context? Both Maggie Lee and Sarah Moulton have clearly stated that the company's survival depends on this. This is so important that if you can't support it we're going to have to support it ourselves in Development."

"We hear business people say things like that all the time," the Ops Chair says. "We invite you back in six months to discuss it again. And now to other matters…"

Defeated, the team leaves the meeting, reassembling in a nearby conference room that Kurt booked in advance. Maxine invites Ellen and the three other engineers who presented the Tomcat proposal.

"Wow, that was so great. Are you really going to go rogue and run all this yourselves?" Ellen says, smiling ear to ear, not affected by the glum faces all around her. "If so, count me in. I'm Ellen, by the way," she says, extending her hand to Maxine and then introducing her team.

"Good seeing you again, Ellen," Adam says with a big smile. "Welcome to our merry band of rebels. If I'm reading the tea leaves right, I think we're going to need your help soon."

Ellen smiles. "The fact that you have Brent onboard is enough for me. What you presented was amazing. I had no idea anyone was working on these types of things here at Parts Unlimited."

Brent smiles modestly, "But we still got our asses kicked, right?"

Kurt says, "We did indeed. But if all goes according to plan, by the end of the day there will be a memo going out from Chris and Sarah announcing a small re-org that will allow Data Hub to operate outside the conventional Ops and QA processes. That will be the official go-ahead to do whatever we need to do."

Everyone on the Data Hub team cheers, surprised at the good news. Maxine hears Ellen mutter, "Wow. That's some pretty powerful mojo you have on your side."

Brent mutters back, "You have no idea. I'll tell you later." Adam laughs in agreement.

While almost everyone is celebrating, Dwayne is glum. When Maxine asks why, Dwayne says, "I just can't believe the LARB didn't support these efforts. We let you all down. What was *supposed* to happen was that they would see the grave danger on the horizon. They were *supposed* to support our cause. They were *supposed* to help…Like Gandalf getting the support of the White Council in *Lord of the Rings*…"

Maxine is surprised when Dwayne puts both of his hands on his temples, groaning. After a minute, he finally says, "But it didn't work out that way at all."

Brent laughs. "You've got it wrong, Dwayne. The Fellowship of the Ring wasn't ever officially sanctioned by the White Council. Gandalf warned everyone that the One Ring was at large, but Saruman refused to help because he was already working for the evil Sauron. So, Gandalf went rogue. He went it alone. Just like we're going to do."

"Damn right," says Kurt. Turning to Ellen and her team, he says, "You all doing anything after work? There's a bar that we go to…"

"What the hell have you gotten me into?!" Chris says, fuming at Kurt. "Maggie and Sarah tell me that you've proposed to create your own Ops organization inside of Dev?! And that you've gotten some sort of exception waiver to start running some new Tier 2 services in the cloud?! I don't suppose you ever thought to ask me first?"

Maxine is in Chris' office with Kurt, Dwayne, and Maggie. Chris is clearly not happy, but Maggie goes to extraordinary lengths to describe the business outcomes that need to be achieved and the grave consequences of not doing so.

Chris stares out his window for several moments and then turns to Maxine. "Do you think we really have the chops to keep all this from blowing up in our faces?"

"Absolutely, with the help of Dwayne and Brent from Ops," she says with certainty. "I'll do everything in my power to make sure things go smoothly. I really think we've got this, Chris. And I promise to take the blame for anything that happens."

At the mention of Dwayne and Brent, a pained expression appears on Chris' face. He looks at her, obviously thinking, *What about 'don't rock the boat' and 'stay in your lane' do you not understand?*

Maxine shrugs. She knows that Chris supported mission-critical services early in his engineering career, over twenty years ago. But ever since then, he's only been responsible for the code, no longer running the actual services that it enables. Maxine could almost see him tabulating all the inconveniences this could create, all the things that could go wrong, balancing it against what could happen if he refuses.

"Fine, fine, fine. I'll do it," he says reluctantly. "You people are going to give me a heart attack," and then shoos them out of his office.

As promised, Chris sends out a memo to everyone announcing a re-org—the Data Hub team is now reporting directly to him, and as an experiment, they'll be exempt from the normal rules and regulations around changes, able to test their own code, deploy it, and operate it in production themselves.

"The email just went out," Kurt says, grinning wildly. "We're in the deployment and operations business!"

"Wow, that's incredible," Maxine says, still staring at the email on her phone. "You know, despite everything we did, I was pretty sure it wasn't really going to happen."

Kurt laughs. "I don't think Chris had that much choice in the matter. Both Maggie and Sarah took this all the way up to Steve."

With the Data Hub re-org, the team is now committed. They are working furiously to automate the production deployments and to figure out how to do production operations without centralized Ops. To what extent they needed to really divorce themselves from Ops for things like backup was still unclear and being negotiated.

The enormity of the challenge is exhilarating. The goal is clear: enable fast and safe deployments into production, and for the first time in years, do it using the same environments across Dev, Test, and Production. And everyone wants to prove that they can get everything up and running before the rest of the Phoenix Project even finishes their testing cycle.

Once again, they are in an imaginary race against the lumbering Phoenix Project.

Maxine is working with Dwayne, Adam, Shannon, and Brent, making slow but sure progress on getting the Data Hub production services to run on something besides the fastest bare-metal servers that money can buy…a decade ago. Many things in Data Hub blew up when installed on a current OS version…from *this* decade. They found several binary executables that no one could find the source code for. Data Hub had become this fragile and irreproducible artifact. *That's great if you're an art collector*, Maxine thinks, *but utterly unacceptable when you're running a mission-critical service*.

They work methodically to create a Test and Production service that behaves like the old one, but can be spun up instantly in a container. For days, she's mired again in the messy world of infrastructure, dealing with Makefiles, YAML, and XML configuration files; Dockerfiles; purging secrets from their source code repositories; and using all her experience to speed up build and test times. This, unfortunately, required lots of Bash scripts.

Maxine remembers a quote from Jeffrey Snover, the inventor of PowerShell. He once said, "Bash is the disease you die with, but don't die of." Maxine shares this sentiment. Infrastructure is messy work, almost the opposite of the pure functional programming she loves—in infrastructure, almost everything you do has a side-effect that mutates the state of *something* in the environment, making it difficult to isolate and test changes, as well as diagnose problems when things go wrong.

But she knows how important this work is, and every bit of knowledge and expertise that she can put into these environments and CI/CD platforms will elevate the productivity of every engineer at Parts Unlimited.

Looking around, she realizes that now some of the best engineers in the company are working on making everyone else more productive. *That's the way it should be*, she thinks.

By the next Thursday, Maxine is thrilled at how much they've been able to accomplish with all the restrictions lifted. But something strikes

Maxine as odd. She notices that all the Data Hub engineers are pitching in. She certainly appreciates their help and she knows that Project Inversion was supposed to disallow feature work, but still, there's almost always some urgent feature that needs working on.

Suspicious, she asks Tom what's going on. He says, "This sounds strange, but technically there isn't any feature work even ready. Believe it or not, every feature is waiting on something from Product Management," he says. "It's everything from a customer requirement that needs clarification, a question about a wireframe, a choice that needs to be made between different options or priorities . . . Sometimes it's something small, like where a button should go. And sometimes it's something big, like them not showing up to the demo to validate what we've built." Tom laughs. "They think we're the bottleneck, but we're always waiting for them."

"Can you show me?" says Maxine. None of the things Tom described sounded good, but the part about the product manager not showing up for the demo pisses her off. What a disrespectful thing to do to engineers who built what you asked them to.

She watches as Tom pulls up a tool she hasn't worked with before, this one used by the product managers to capture ideas from customers: the ideal customer journey, value hypotheses, manage experiments, and so forth.

"What are all those blue cards?" she asks.

"Good eye. That's exactly the problem," he says. "Those are all the features that we're working on, but we're blocked because of something we need from Product Management. Like all those reasons I mentioned before. Oh, and here's some yellow cards which are the features we've completed but that haven't been accepted by the business stakeholders yet. This one has been waiting for forty days."

Maxine feels her face turn red, indignant that as much as Product Management complains about the need to get features to market quickly, all these blue and yellow cards represent where *they* are in the way, not Development. *How can we keep Product Management accountable?* Maxine thinks. *Time to bring in Kurt.*

Ten minutes later, Kurt is with them, staring at the sea of blue cards. "I get it. This is not good, but I have an idea," he says. "By the way, did you know that Sarah put a huge design agency on retainer, and now they're flooding some of the other teams with wireframe diagrams that

will probably never get worked on? And no matter how much the Dev managers ask them to stop sending wireframes, they still keep coming."

"Why?" Maxine asks.

"I think it's because Sarah needs to show off the apps she wants to build," he says. "But what's funny is that when the designers came here, the last thing they wanted to do was wireframes. They wanted to learn about our customers, and they did a bunch of exercises to better clarify goals around the personas we used. There was even one session where *we* all drew wireframes," Kurt says, laughing.

Working with designers fascinates Maxine. Early in her career, the ratio of UX and designers to developers was 1:70. These days, great teams doing consumer-oriented products have ratios of 1:6 because it's that important to create products that people love. Every consumer these days knows what a professional app feels like. Apps that don't have great designers are often ridiculed as "enterprisey."

She's seen teams still waiting to be assigned designers, eventually making their own wireframes, HTML and CSS styling, and icons just to keep feature flow moving. *These are the projects that teams are actually embarrassed to show other people*, she thinks.

The good news is that Sarah got a bunch of great designers. The bad news is that she put them all where they weren't needed and were actually slowing important development work down by flooding their backlogs with things that didn't matter.

That evening after dinner, while her family played with Waffles, she opened up her laptop. Something about the sea of blue cards that Tom showed her earlier had been bothering her, and she's determined to get to the bottom of it.

That sea of blue cards is a part of the tool that the product managers use to manage the funnel of ideas to achieve business outcomes. This is a process that starts long before a feature is created in the Dev ticketing system. She logs into that tool using the credentials that Tom gave her. Browsing around, she can see when ideas were first conceived and brainstormed and all the various phases until it becomes an approved feature.

She searches for the first feature that she worked on with Tom about extended warrantee programs. When she finds it her jaw drops. That

feature was first discussed almost two years ago. It started off as a small feature but was rolled up into a larger warrantee initiative, which then had to be pitched to a steering committee. When it was approved, they wrote up detailed specifications, which were pitched six months later. Only then were they approved (a second time) and finally funded.

This idea bounced around in the marketing and project management organization for almost two years, and then turned into a super-crash priority feature that had to ship by the end of the year.

For something this important, we wasted almost two years, she thinks. In the ideal, they should have just assigned a team that included developers to explore the idea and build a solution together. *Instead of one product manager working on this the entire time, we could have had five people working on it. And we could have been learning the whole time,* Maxine thinks.

She wonders how much of this specification document that was written two years ago is now out of date.

She pulls up the Dev and QA ticketing system and copies some dates into the spreadsheet. She spends nearly ten minutes Googling around, trying to remember how to do date conversions and date arithmetic correctly.

She stares at the screen, shocked. She does the formula a couple different ways, but she still gets the same number.

She texts Kurt:

> We've got to meet tomorrow. I have something to show you.

Maxine is with Kurt, Tom, and Kirsten in a conference room projecting her laptop on the screen. Everyone is staring at it in disbelief, which she totally understands. She's been thinking about this number all night. "Can that actually be right?" Kurt finally asks.

"I'm afraid so," says Maxine. Kurt looks over at Kirsten, who is still staring at the numbers.

"Only 2.5 percent of the time required to go from *concept* to *customers actually using the feature* is spent in Development?" she finally asks, the disbelief evident in her voice. She stands up and walks to the large TV screen to look more closely at the spreadsheet. "Where is all the time going?"

Maxine says, "Long before the feature ever gets to Development, it goes through the funding approval process, which often takes over a year.

And then once the feature is created, most of the time isn't spent in-work, it's waiting for a product manager to respond to a question. It's the Square again. Teams are spending too much time waiting for product managers to get them what they need…

"And then once they're done with the feature, they're waiting for QA and deployment," Maxine says. "This is terrible. We've spent all this time hiring more developers, but they often don't have things ready to be worked on. And when they do finish a feature, it takes forever to actually get things into production so that our customers can use it. And often the only feedback we get are the annual focus groups.

"We don't have a fast value stream," Maxine says. "What we've got is more like a stagnant value pond, full of scum, breeding malaria."

"Time to call Maggie," Kurt says.

That afternoon, Maggie comes up with an elegant solution. She decides to move the Data Hub product manager from the Marketing building to a desk right by Maxine starting Monday.

In the conference room, Maggie tells him, "You're the bottleneck. Your top priority now is to make sure any questions that the technology teams have are quickly answered. Nothing else takes priority over that."

He balks and then proceeds to describe all the other demands on his time. Talking with customers, helping sales with negotiations and trying to break them of bad habits, briefing internal executives, working with business operations, arguing with business stakeholders to agree on a product roadmap, escalating things up the chain to get approvals for urgent issues… And way down the list was answering questions from developers.

Maxine listens with interest, realizing that no one can get anything done when you're pulled in that many directions. Maggie also listens patiently, nodding and occasionally asking questions.

When he's done, she says, "If you're too busy to work with the technology teams, I'll move you into a pure product marketing role, and you don't have to move your desk. Right now, I need product managers who are working side by side with the teams who are building what will achieve our most important business objectives. If you still want to be a product manager, I'll figure out how to clear your plate and get those other responsibilities assigned to someone else.

"Don't give me an answer right now," Maggie says. "Think about it and let me know first thing Monday morning."

Maxine is impressed. *Maggie does not mess around*, she thinks.

By mid-day Monday, that product owner moved his desk right next to Maxine. The dynamic immediately shifts. To get answers, things no longer wait on tickets. Engineers are able to just swivel their chairs around and ask him. Things that normally took days are being resolved in minutes. And better yet, engineers start gaining a much better understanding of the business domain.

Maxine smiles. The team of teams keeps growing, and it feels good.

From: Alan Perez (Operating Partner, Wayne-Yokohama Equity Partners)
To: Dick Landry (CFO), Sarah Moulton (SVP of Retail Operations),
Cc: Steve Masters (CEO), Bob Strauss (Board Chair)
Date: 7:45 p.m., November 5
Subject: Strategic Options **CONFIDENTIAL**

Dick and Sarah,

For our next meeting, I've asked an investment banker we've used in the past to brief us on the market outlook for the retail and manufacturing sides of the Parts Unlimited business. Could you present a high-level briefing on the Phoenix initiative so we can get their thoughts?

Given the criticality of the upcoming holiday sales performance, I thought it might be useful to introduce ourselves to them sooner rather than later. Hopefully any valuation estimates will be anchored before any disasters. (You never want to talk to bankers when you really need them. They can always smell fear.)

Sincerely, Alan

CHAPTER 13

• *Thursday, November 6th*

It's six thirty on Thursday evening, and Maxine is again in a conference room, along with the entire extended Data Hub team. Everyone is tense and on edge, looking at a large screen that has all the production telemetry and dashboards showing the health of the test and production Data Hub services. Maxine is pretty sure that everyone is holding their breath, just like she is.

The team had been deploying into a Test environment for weeks before having the confidence to start deploying into Production, which required days of negotiations with seemingly every area of the business. An agreement was reached that production pushes would occur after business hours, after internal business users had gone home but before thousands of internal batch jobs run at midnight.

For the past two days, at the same time each day, as a test, they've been pushing "whitespace changes" into production—adding a couple of blank lines to the end of HTML or configuration files, which in theory should not change functionality in any way.

Of course, reality is much, much messier. It was the "world as imagined" colliding violently with the "world as it actually is." They discovered that they accidentally forgot some critical files in their container images, which knocked Data Hub offline for nearly a half hour. Three hours later, after a painstaking investigation, they were able to execute the whitespace deployment without crashing anything.

The next day, they performed a second whitespace deployment, but absolutely nothing happened. It took them another hour to discover a configuration error they had made earlier in the day had broke all their pipelines. It was messy and imperfect, but the fact that they were quickly solving these problems gave Maxine confidence that they were on the right track.

Today, Tom and Brent are about to start the first push of Data Hub application code into production.

"Okay, here we go," says Tom. "Starting code deployment." He clicks a button and some new boxes appear on the CI/ CD pipeline page, showing that a new deployment has been initiated. They all watch with bated breath as the log files start scrolling by.

Over the next ten minutes, Maxine sees notifications of the tests being run, the tests passing, files being copied onto the production system, Data Hub being restarted, more log messages as it starts up, and then the log messages stop.

On the screen in front, the big circle representing the health of Data Hub goes from green to red, and stays red.

"Uh oh," Tom says. "Data Hub just crashed on startup…" He types quickly into a terminal window.

Maxine hears people swearing all around her, and Maxine joins Tom at his laptop as he tries to figure out what went wrong. She sees him scrolling through seemingly endless Java stack traces, looking for any clue on why Data Hub crashed. He yells out, "It's some type of uncaught exception, but I can't find a useful error message…"

Shannon calls out from the other side of the table, "Folks, I'm not seeing any active connections to the database."

Brent looks up with an expression of horror on his face. "Shit, did I forget to change the database connection string?"

When he just stares off into space, Maxine gently asks, "Good hypothesis, Brent. What are you thinking? How can we test your idea?"

As if jolted out of a trance, Brent looks back at Maxine. "I can't remember where the database connection string is stored! Is it an environment variable? Or is it in a configuration file? Does anyone know?"

"It's an environment variable. I'm pasting where it's set in the chat room," Purna says. Maxine watches as the team jumps into action.

Twenty long minutes later, the necessary fixes are made, and Data Hub is back up and running. Everyone breathes a sigh of relief. The transactions that had been blocked have all been processed and everything is green again. "Okay, we found two other places where we missed some configuration settings in environment variables. Those are now all in version control. It should work this time. Is everyone ready to try again?" Tom asks, and everyone gives a thumbs up, if not as confident as earlier.

Again, they watch as the Data Hub deployment starts… tests are run in the test environment, files are pushed into the production server, Data

Hub is stopped, the new files copied onto the server, Data Hub is restarted, and the startup messages start to scroll by.

This time, there is only a half-second pause where they got stuck before, and then screenfuls of logging messages scroll by faster than anyone can read. Tom whoops in delight, but he still watches his laptop, knowing that lots of things still need to go right before Data Hub is properly handling requests again.

Moments later, the red indicator next to the Data Hub health turns green. Some people clap, but most people realize that any celebration is premature as eyes quickly turn toward the production telemetry. Maxine sees the logging messages come to a crawl and then stop, and the production graphs start climbing again.

The entire room erupts in cheers. Almost the entire room. Maxine notices that Brent looks upset, as if he's angry at himself for the first database connection error.

Tom confirms, "Data Hub is processing transactions again. We're in the deployment business!" He looks around with a big smile. "Who wants to go to the Dockside to celebrate?"

"Now that everyone is here, I can properly hoist a glass to all of you for your amazing work!" says Kurt with a big smile. "Rest assured that you've earned the attention of some *very important people* who have decided to join us today!"

Kirsten raises her glass. "My congratulations to you, everyone. And you did it all without even one project manager from my team, which makes it even better!"

Everyone laughs and applauds. Even Brent is smiling now.

"Ah, what great timing," Kurt continues, raising his glass to someone walking toward them

Maxine turns around to look. *Holy cow*, she thinks.

It's Maggie Lee. The crowd at the Dockside keeps getting classier and classier. Kurt smiles and says, "Meet our newest visiting VIP."

"Hello, everyone," Maggie says, sitting next to Maxine. "I'm delighted to be here to celebrate the successful Data Hub code push."

Kurt introduces all the Data Hub team members to her, and Maggie stands up and introduces herself to everyone. "What you're doing with

Data Hub is amazing, and trust me, all my product managers are incredibly excited about how what you're doing could help us quickly create new product bundles," Maggie says. "We know so much about our customers, and we want to use that information to help them solve their problems. If we do this right, this will naturally lead to achieving our revenue goals. That's the bet we're making. I don't need to tell you all how important the upcoming Thanksgiving and Christmas season is.

"I just want to thank you for being willing to help us, and I'm really looking forward to working with you all," Maggie says. "The work you're doing is important, and I think it's critical to the success of the company."

She raises her glass to everyone's loud applause.

Over pitchers of beer and glasses of wine, Maggie tells the group more about their struggles, some of which surprises and worries Maxine. They have only completed integrations with two systems of record. They are still waiting for nearly twenty API integrations, including product, pricing, promotions, purchases…

They've hired a bunch of data scientists to help create more effective offers, but they're still waiting on the Data Warehouse team: purchasing history from all of the disparate systems, car service histories, their customer loyalty programs, and their branded credit cards. When it's not an executive asking for data for a board presentation, even the simplest data requests take six months, as their requests lumber through the Data Warehouse Dev and QA processes. And like Brent found, the data they get is often malformed, unreadable, incomplete, or, worse, inaccurate.

When Maggie and team complain, the Data Warehouse manager emails everyone a graph showing that they're keeping up with incoming data requests, but that's only because people have given up and stopped asking them for anything.

After the challenges in front of them become clear, Kurt turns to Maxine. "Maggie already has a bunch of development teams assigned to support the Promotions effort, but they clearly need some help. Based on what you've heard, who would you want to take with you to make the biggest difference?" He gestures at everyone around the table. "You have the pick of the entire litter. You can have anyone from the Data Hub Dev and QA teams. Heck, anyone from the Rebellion."

"Pick of the litter. Nice, Kurt," Maxine snorts, trying not to picture their best engineers as if they were a basket of puppies at the Humane Society.

She ticks through the mental list she's been accumulating. "We'll need someone with experience with architecture and decoupling components that are deeply entangled with each other. We'll need someone really good at databases, because we'll probably need to reduce our reliance on the big, centralized Phoenix databases and all those systems of record. We'll need some serious infrastructure skills to support a new deployment and operations model. And because we'll likely be running things in production ourselves again, we'll need people with superb skills in Security and Ops."

She thinks for a minute. "I'd take Cranky Dave, Adam, and Purna on Dev and architecture. Dwayne and Brent on databases, infrastructure, and Ops. Shannon on security and data."

As she calls off people's names, they smile, sitting up straighter. She points at Dwayne and Brent. "I think we'll probably need two or three more people on infrastructure and databases, since we'll probably be spinning up a bunch of new things, probably in the cloud. Can you think of anyone you'd want on the team?"

Dwayne and Brent look at each other. Dwayne says, smiling, "I think we can come up with a short list of awesome engineers."

To Kurt, she says, "I haven't met any of the Promotion developers, so I don't know their skill levels. Ultimately, if we need to make a difference in time for Thanksgiving, we need to get heads-down in the code soon and ensure that all those Promotions teams are productive—either we onboard them onto the platforms we've already built or we build or buy what they need."

She points at Tom. "I'd want to take three or four developers to go native in the Promotions teams. Tom, do you know who you'd pick?"

When he nods, she says to Kurt, "That's twelve people. I have no idea how you're going to convince everyone to let us strip the benches. None of those managers will want to lose their best people."

Kurt looks at Maggie. "We'll have to convince the higher-ups to make a massive investment in speed to achieve your goals. Do you think you can swing that?"

"Hang on a second. You'd all do that for me?" Maggie said, suddenly looking a little suspicious. "What's in it for you?"

Kurt smiles. "Ma'am, you're looking at a renegade group of engineers who want to solve big problems that actually matter to the business. Our attempts to go through the normal channels haven't worked, so here's our chance to work directly with the business instead of through technology middle managers. If we succeed, we get credibility. We'd love your endorsement supporting these new ways of working."

Kurt shrugs, continuing, "If it doesn't work, we all pretend like it never happened and promise not to bug you again."

"You've got a deal," Maggie says after thinking a moment. "And here's the good news. I don't need approval from anyone—it's my call. Sarah is already onboard. It's my belief that the survival of the company depends on this."

Almost on cue, Maggie looks down at her phone, saying, "Hang on a second, it's Sarah," as she taps out a reply. "Uh, she's getting heat from some people because Data Hub was down earlier today, and she wants to know what's going on, who caused the problem, and if we need to make an example of them."

Oh great, Maxine thinks. By working with Maggie, they're falling even deeper into Sarah's orbit.

The next morning, Kurt, Maxine, Kirsten, and Maggie are once again in front of Chris. When Kurt proposes temporarily swarming the Promotions efforts, not surprisingly Chris seems exasperated.

"You want my job, Kurt? Cause you're sure acting like it," he grumbles. But Maggie implores him on the importance of the need for accelerating the work to support the Black Friday holiday promotions and how it could generate very visible and quick wins, and Kirsten reassures him that the other efforts can absorb the temporary reassignments.

Chris furrows his brow. Just like last time, he turns to Maxine. "What do you think, Maxine? Do we really need to do this?"

Maxine studies him, realizing how uncomfortable he is with the constantly changing plans, very different from the static plans that characterized the Phoenix Project.

"Without a doubt, Chris. This is clearly where the company needs developers most. We can't be hamstrung by our org chart or, for that matter, the annual plan we made last year," Maxine says, reassuringly.

He looks at her for another moment, grunts his approval, and again briskly shoos them out of his office.

Maxine and Kurt give each other discrete thumbs-up as they walk out.

Despite Sarah's loud demands to find someone to blame on the temporary Data Hub outage yesterday, Kurt refuses to do anything along those lines. Instead, he gathers everyone in a conference room.

Kurt starts the meeting, saying, "Every time we have an outage, we'll be conducting a blameless post-mortem like this one. The spirit and intent of these sessions are to learn from them, chronicling what happened before memories fade. Prevention requires honesty, and honesty requires the absence of fear. Just like Norm Kerth says in the Agile Prime Directive, 'Regardless of what we discover, we understand and truly believe that everyone did the best job they could, given what they knew at the time, their skills and abilities, the resources available, and the situation at hand.'

"Let's first start by assembling a timeline and gathering details on what happened. To help with the process, Maxine pulled together our production telemetry and logs, as well as our chat rooms, just to provide a framework for discussion. The goal is to enable the people closest to the problem to share what they saw, so we can make our systems safer. The only rule is that you can't say 'I should have done X' or 'If I had known about that, I would have done Y.' Hindsight is always perfect. In crises, we never actually know what's reallyl going on, and we need to prepare for a future where we have an equally imperfect understanding of the world."

He looks over at Maxine, indicating for her to proceed. Maxine is impressed and wonders briefly if Kurt has been coached by Erik before this meeting. If so, she's glad. But despite Kurt's strenuous declaration that people shouldn't be afraid to talk, everyone seems reticent to react . . . even the members of the Rebellion. Given the ever-growing culture of fear and blame, Maxine has been prepared to model the behaviors you'd see where there's real psychological safety—Erik's Fourth Ideal.

But before Maxine can start, Brent blurts out, "I'm so sorry, everyone. It was all my fault. I can't believe I missed the database connection string. I never make that type of mistake, but I was in such a hurry . . . "

Brent looks so distraught, as if he'd been wanting to make this confession for days. Kurt puts his hand on Brent's shoulder and says, "Brent, let's go back to the Agile Prime Directive. No one is at fault. Everyone did the best they could, given what they knew. Let's just stick with assembling the timeline. Maxine, please lead the way."

"My pleasure," Maxine says, winking at Brent, projecting her laptop on the TV. "I'm choosing to start our timeline at 6:37 p.m., which is after Tom initiated the deployment, after all the tests passed but the app failed to start. The health indicators went red, and Tom was the first to notice. Tom, what exactly did you see?"

"I was watching the logs scroll by in the deployment tool, and I saw the startup messages, as you'd expect, and then I saw a bunch of error messages and a stack trace," he says, his face darkening, reliving the crisis.

"Got it," she says, adding to her notes that everyone can see on the TV in the front of the room. "What happened next? I remember feeling almost borderline panic, because despite all our preparation, we were clearly in uncharted waters." With a wry smile, she adds, "Umm, that's code for 'I was so scared I was crapping in my pants.'"

People around the table laugh, and Tom says, "Yeah, me too. I've spent decades looking at stack traces, but I've never seen them in our deployment tool. I couldn't stop the window from scrolling on me, and I couldn't see anything long enough to read it."

Maxine had no idea, because Tom had seemed so calm and was so effective at making sense of the logs. She is typing when Tom says, "You know, I should have rehearsed looking at logs in this new tool."

"I totally get that, and I've been there…and it totally sucks to feel that way," Kurt responds. "But remember, we're doing this so that we can be better prepared for the next crisis, when we will be equally ignorant of entirely new things that are just as important and will be just as obvious in hindsight…This is great stuff, Tom. Keep going. What happened next?"

Over the next hour, Maxine and the group assemble an amazingly detailed and vivid timeline of what actually happened. Once again, she marvels that anything can run in production at all given all the imperfections and sharp edges present in their daily work. Log files scrolling by too fast to

read, configuration settings scattered across scores of locations, potential failure points hiding in almost every nook and cranny, surprises lurking around every corner… *Given all this, it's amazing that Data Hub has worked mostly without incident for over a decade*, she thinks.

Maxine is certain that everyone has learned something about how Data Hub actually works, in stark contrast with their mental models of how they *thought* it works. She records a list of five things people will change right away that will likely prevent future outages and will certainly make fixing certain problems faster in the future.

As they adjourn, Maxine smiles and says to Kurt, "Nice job running the meeting." She means it. It was a pitch-perfect example of improving daily work and fostering a culture of psychological safety, as Erik described in the Third and Fourth Ideals.

Reflecting on the meeting, Maxine now appreciates how tenuous and fleeting the conditions that enable psychological safety can be. It depends on the behavior of leaders, one's peers, their moods, their sense of self-worth, wounds from their pasts… *Given all this, it's amazing that psychological safety can be created at all*, she thinks.

Later that day, Kurt, Maxine, and the rest of the newly selected team are gathered in a conference room to meet with Maggie and the rest of the Promotions team leads.

During introductions, Maxine notes that most of the twenty people in the existing Promotions team are front-end developers—they own the mobile application, the product landing pages for the e-commerce site, in-store applications, and all the applications that Marketing staff use to manage the product promotions lifecycle.

Maggie is presenting. "Thank you to Kurt, Maxine, and the rest of the engineers from the Data Hub team who have volunteered to help us achieve some badly needed near-term wins. I put together some slides to frame some of the higher-level business outcomes that this team was created to make happen.

"Our market share is declining, primarily because we have little presence in the e-commerce market, the fastest growing part of the broader market," she says. "This is where our competitors and the e-commerce giants are taking share from us. The good news is that we have fiercely

loyal customers…the bad news is that the average age of our customer base continues to increase. Our competitors are clearly winning younger customers, a real market segment, despite declining car ownership because of the rise of ride-sharing, such as Uber and Lyft. But the number of car-miles driven per year keep growing, although who's doing the driving is definitely changing. But without doubt, demand for car maintenance should grow, not shrink."

Maggie continues, "For our loyal customers, we know what they buy and how frequently they buy it. We're focused on enabling personalization and knowledge of current inventory to drive promotion. Until recently, we've never been able to use this information to create compelling offers for them.

"We know from our customer research that our core market uses mobile phone apps extensively—in fact," she points to the projected slide, "here is a picture of Tomas, a customer we interviewed during our market research. He's a fifty-two-year-old public school teacher. For decades, he's done all his own car maintenance. It's something he did with his dad and now it's something he does with his two teenage daughters and son. He wants his kids to focus on STEM, but he insists that they understand mechanical basics and learn self-reliance.

"He also maintains his wife's car, and when he has time, his parents' cars too," Maggie says. "Tomas doesn't consider himself to be very technical, but he has six computers at home that he supports for his entire family.

"Right now, he uses a spiral notebook and these file folders to keep records for each car he maintains. He uses his mobile phone all the time, primarily for messaging but also Amazon. He would love to have more of the maintenance routine codified. He loves using Parts Unlimited, but he says he would far rather look in the app for parts rather than having to call the store. He says he likes the in-store employees and knows many of them by name, but complains about our terrible automated phone system and hates having to listen for which button to push to get to a real person."

Maxine laughs. No one likes those things.

"For the Thanksgiving and Christmas season, we want to find inventory that we have too much of, combine it with our personalization data, create compelling promotions, and deliver them through our e-commerce site, email, and our mobile apps. We want to drive real revenue through

these promotions and increase the average monthly usage of the app to prove that we're actually building something that they value.

"The Phoenix teams have already identified all the needed interfaces to all the various systems where this data is stored: the customer and orders database, the POS transactions, fulfillment systems, the e-commerce website, and the ad campaign data from the Marketing team.

"One of the most critical sources of data is the in-store inventory systems. We want to promote overstocked items, but we've got to be very careful that we don't promote items that we don't have on hand in that region.

"We finally rolled out a customer relationship management or CRM system a couple of years ago. But as I described last night, connecting data about the customers, such as which automobiles they own and some demographic information, with the vast wealth of other data we have is a real struggle.

"You can see what we're trying to achieve, right? If only we had a single view of the customer: top of funnel, bottom of funnel, as well as their complete history with the company. Not only what they purchased, but also what they did on our site, what they browsed, searched for, their credit card transactions, repair history… There's so much potential!

"If we could combine all this information, we would know so much about what they need, and we'd be so much better able to help them," she says, almost wistfully.

Maxine nods, impressed. She says, "After analyzing even the small amount of data that we've been able to combine, we've built some customer profiles based on their behavior. The archetypes we've created so far are: Racing Enthusiast, Frugal Maintainer, Meticulous Maintainer, Catastrophic Late Maintainer, and Happy Hobbyist."

"For now, we are focusing on the Meticulous Maintainers and the Catastrophic Late Maintainers, because we think these groups have the highest probability of buying for the types of campaigns we're thinking of," Maggie says. "We know the Meticulous Maintainers purchase things like oil change products every month without fail. On the other hand, the purchase history of Catastrophic Late Maintainers suggests that they are constantly accumulating more expensive tools and engine parts, just needing a nudge to complete their work.

"On the screen, you can see a bunch of hypotheses we have. These are offers we think will be a big hit with these customer segments. And

this report shows the attributes of customers that we're at risk of losing entirely," Maggie says. "The problem is that executing on any of these ideas requires months. Anytime we want to do something, we've got to make a million changes across all of Phoenix. Phoenix has been rolling for three years, and we haven't made one targeted promotion yet. And if we can't experiment, we can't learn!"

"You haven't been able to execute even one of these promotion ideas?" Maxine asks, surprised. "How is that possible?!"

Maxine hears grumbles from around the room from the Promotions team. They start describing why.

"We're still waiting for access to all those back-end systems. The only one we have access to is the inventory management system," someone complains. "We already have all the data from the TOFU—top of the funnel. We need information about the customer lifetime value from BOFU—bottom of the funnel."

"The integration teams take six to nine months to create any new integration for us," says someone else.

"When we do query the inventory management systems, they often shut us down because of the CPU load we generate or the amount of data we copy," says a third person.

"The APIs from many of the back-end systems don't deliver the data we need. We've been waiting for months for those teams to implement the necessary API changes."

"We're still waiting for correct data from the Data Warehouse team, because their reports are always wrong. Last time we found people's last names in the zip code field."

"We're still waiting on getting new database instances created for us." And on and on.

There were twenty developers on the Promotions team, with a ton of good ideas that could deliver on so many of the Phoenix promises, but they were all bottlenecked on the back-end systems.

Suddenly, Maxine is very sure that they can help. But another part of her is aghast at how helpless these developers have been, unable to complete their work.

Maxine and the rest of the engineers from the Data Hub team smile at each other. Seeing this, Kurt folds his hands in front of himself, smiling. "I think we can help."

After nearly ninety minutes of excited discussion and brainstorming, everyone adjourns. Maggie takes Kurt and Maxine aside. "That was amazing. We've been crying out for help for so long, but this was the first time that we've been able to engage like this with anyone."

"Well, we haven't done anything yet," Kurt says. "But hopefully by the end of next week, we'll have something that we can point to as progress."

Maxine nods emphatically. She looks at Maggie for several moments and then asks the question she's been wanting to ask since last night. "Just what does it take to build great products? And how can we as developers help?"

"Where do I start?" Maggie says. "It usually begins with understanding who our customers are, both current and desired. Then we typically segment the customer base, so we know what set of problems each faces. Once we know that, we can understand which of those problems we want to solve, based on market size, ease of reaching them, and so forth. Once we know that, we think about pricing and packaging, offer development, and more strategic issues, such as the overall profitability of the product portfolio and how it affects the achievement of strategic goals.

"I need each of my product managers to be able to live in this domain," Maggie continues. "Almost all great product organizations create customer personas so that everyone can better understand and relate to the people who you're building products for. That's the reason behind so much of the UX and ethnographic research we do. For these personas, we articulate their goals and aspirations, figure out what causes problems for them during a typical day, and describe how they do their daily work. If we do things well, we end up building a set of user stories, framed inside of the business outcomes we want. We should be testing and validating all these assumptions in the marketplace and learning all the time."

Maxine says, "I love this relentless focus on understanding the customer—it reminds me of the Fifth Ideal."

Maggie looks at her quizzically.

"I'll explain later," Maxine says.

"You know, if you're that interested in the customer, you could do the same in-store training that all employees and managers do. Two weeks ago, all the new sales managers flew here to headquarters to spend a week with our local store. You missed that, but if you want, there's a new employee training happening on Saturday. Want to join them?" Maggie asks.

Maxine's jaw drops. She's always been envious of people who have been able to do this program, and Maggie just offered her the chance to take part in it. "Holy cow, I'd love to. Frankly, I'm a little bummed that I've been here for almost seven years and have never been offered this."

"I require all my product owners and everyone at the manager level and above to go through it," Maggie says. "I'd be happy to get it arranged."

"Sign me up!"

It's Saturday morning. Maxine is in front of the bathroom mirror making sure her "Hello, My Name Is Maxine: How Can I Help?" name tag is straight.

She's very excited to finally take part in this program. Famously at Parts Unlimited, every leader at the director-level or higher must work in the stores as a front-line employee twice a year. Not as a high-falutin' store manager, but as a regular employee behind the register or out on the floor. This is something that Parts Unlimited has been doing since they opened in 1914.

Maxine quickly says goodbye to her family, who are lounging in the living room on various forms of technology, and dashes to her car. She doesn't want to be late for her first day of in-store training. Maxine is a stickler for punctuality and expects that the store manager she'll be reporting to is likely the same way.

The previous evening, she spent three hours watching YouTube videos on home car maintenance with her kids. She's relieved that changing the oil on a 1984 Toyota Tercel is exactly the same as it was twenty years ago. An oil change is an oil change since the invention of the internal combustion engine. Even now, Maxine still refills the windshield wiper fluid in her family's cars, refusing to pay someone to do it for her. However, it's been decades since she's changed her own oil or transmission fluid.

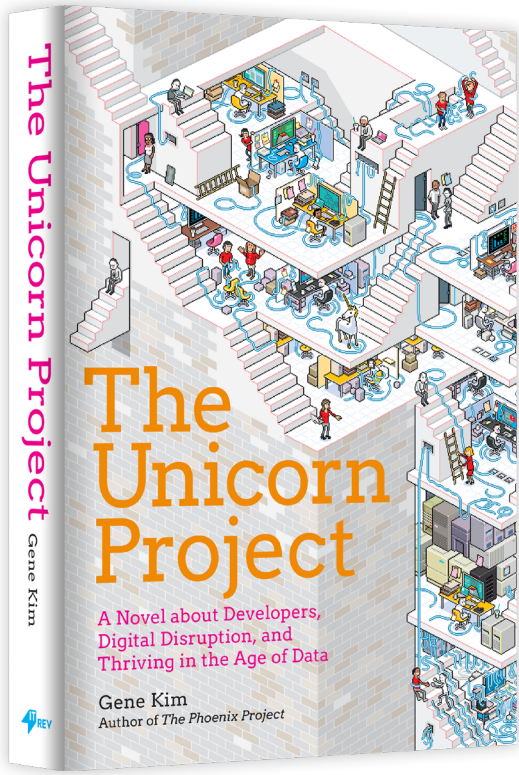
When Maxine walks into the store, she immediately feels out of place. She sees four young men and one woman, all in their twenties, and an older man in his forties.

Mildly irritated that she's not the first person here, she joins the semi-circle facing the store manager, Matt. Maxine recognizes him from having been in the store before. He's in his early thirties and looks almost like a drill sergeant. He glances at his watch and gives Maxine a small nod and smile of recognition.

Thank you for reading excerpt 5 of

The Unicorn Project!

Stay tuned for the next excerpt!



In the meantime, feel free to:

1. Share what you thought of the book (please use #UnicornProject and mention @realgenekim), or share a picture of the cover! (You can find images and so forth **here**.)
2. Of course, you can find more information about the book and **pre-order it here** or at your favorite local retailer.

The Unicorn Project officially releases on November 26, 2019.

Thank you!

Gene Kim