QUICKSANDS OF SOFTWARE DEVELOPMENT

D. PETRE BOGDAN

Quicksands of software development

D. Petre Bogdan

This book is for sale at
http://leanpub.com/quicksandsofsoftwaredevelopment

This version was published on 2022-11-12

References to trademarked or copyrighted works, individuals, companies, products, and services are included for criticism, commentary, and illustration purposes only, and should not be considered endorsements or any other type of affiliation with this book or its author.

The information in this book is distributed on an ”as is” basis, without any guarantees or warranties, express or implied. The book contains the opinions and ideas of its author. Much of the content is based upon personal experience and anecdotal evidence and is intended to provide helpful general information on the subjects that it addresses. The views expressed herein are subject to change without notice. While the author has used his best efforts in preparing this book, this does not guarantee the accuracy, adequacy, or completeness of any information found within it, and the author assumes no responsibility for errors, inaccuracies, omissions, or any inconsistency herein, regardless of cause. The author assumes no responsibility or liability whatsoever on behalf of any person or entity with respect to any consequence, loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work. Readers are ultimately responsible for their decisions and any potential damage resulting from the use of this information. It is your responsibility to review, analyze, verify, and evaluate any content or information provided in this book before taking any actions or relying on it. The author does not engage in rendering legal, career, or other professional services, and the book is not an individual and/or personalized recommendation to a particular person or group of persons of any kind. The explanations and strategies contained herein may not be suitable for your situation. Use the information contained in this work at your own risk. If expert assistance is required, you should seek the services of a competent professional.

Please note that this book contains ideas, opinions, and some occasional strong language that some readers may find offensive.

Any perceived slights of particular people, organizations, companies, or products are unintentional.

Because of the dynamic nature of the Internet, any web addresses or links contained in this book may have changed or disappeared between the time this was written and when it is read, and may no longer be valid. The fact that a resource or website is referred to in this work as a citation and/or a potential source of additional information does not mean that the author endorses the information the resource or website may provide or recommendations it may make. Further, the author does not have any control over and does not assume any responsibility for author or third-party websites or their content.

Version: 33e2fd96a162694ee3634bfa160f370cca89edee, f05c96fc4699e46f4385d5bb60bb5014eedd898a

Cover design, illustrations, and formatting by D. Petre Bogdan

Contact: quicksandssoftwaredevelopment@gmail.com

Website: quicksandsofsoftwaredevelopment.com

Contents

Intro **1**
- Heads up 5
- Nothing new 8

Staff developers **12**
- Neglected process 12
- A different approach 12
- Prepare 12
- The interview 13
- Loopholes 13
- Match check 13
- Performance reviews 13
- Frequent discussions 13
- Bad apples 14
- The working environment 14
- Exit interview 14
- It’s really important 14

Continuous learning **15**
- English 15
- Solid foundations 15
- Mistakes 15
- Mentors 16
- Technologies 16
- Specialist vs. Generalist 16
- Too much information 16

Problem-solving skills 16
Business domain 17
Your investment 17
Unlearning 17
Programmer vs developer 17
Team Lead 17
Team Lead first 18
Delegate 18
Motivation 18
The working environment 18
Trade-offs 18
Health 19
Sustained effort 19

Software projects 20
Estimations 20
Requirements 20
Negotiations 20
Forgotten bits 21
Risks and assumptions 21
The plan 21
Change 21
Measure 21
Define success 22
The deadline 22
Sacrificing quality 22
Other sacrifices 22
Lessons learned 22
It takes longer 23

The Business-IT divide 24
Us versus them 24
Cost 24
Shadow IT 24
The real cost 25

Bridging the gap 25
Trust 25
Build trust 25

Meetings and reports **26**
Interruptions 26
Bad meetings 26
Bad reports 26
Better reports 27
Better meetings 27
Necessary 27

Bad bosses **28**
Technical (in)competency 28
Cogs in the machine 28
A losing battle 28
Not bad 29
Not as bad 29

Bad managers **30**
In the middle 30
Communication 30
Failed communication 30
Tools of the trade 31
Project Manager first 31
Overdoing it 31
Just another boss 31
Management style 31
Lost cause 32

Bad developers **33**
Lack of skills 33
Poor attitude 33
Over-Engineering 33
Not code? Not interested! 34
Failed communication 34

Lost cause 34

Outsourcing **35**
Still your people 35
Hidden costs 35
Different cultures 35
Final responsibility 36

Scrum **37**
You do it! 37
The Scrum Master 37
The Product Owner 37
Values and principles 38
Empowerment 38
Technical considerations 38
It takes time 38

Picking up the pieces **39**
Professionalism 39
Just one layer 39

In ending **40**

References **41**

Intro

Here is a definition of *"quicksand"*, straight out of the dictionary[1]:

> Quicksand, (noun). Sand readily yielding to pressure; especially: a deep mass of loose sand mixed with water into which heavy objects readily sink.

Depending on where you live, you've probably also seen signs like this at some point in your life:

You need signs like that to warn you of the danger. Quicksand looks solid when undisturbed, but walking into it causes liquefaction. And that's when you start to sink.

I think many companies that develop software should have a sign like that somewhere in their offices. Why? Here is the second definition of *"quicksand"* from the same dictionary page[1]:

> Quicksand, (noun). Something that entraps or frustrates.

Again and again, software development projects fail for various reasons: unclear project goals or requirements, poor business decisions, lack of planning, poor communication, bad technological choices (like choosing the wrong programming language or the wrong architecture), complexity and size of the project, poor prioritization, project management mistakes, unrealistic deadlines, office politics, and many more. Even if the project succeeds, the result is often a combination of cost overruns, late product deliveries, poor quality, lots of frustrated people, and jokes like this one:

> *"Developer to Team Lead"*: We cannot do this project. WE CAN NOT! It implies a significant design change, and no one in our team understands this model. And, above all, no one in the company knows the programming language in which this application was written. So, even if someone wants to work on it, they can not. In my personal opinion, the company should never have gotten itself involved with this project.
>
> *"Team Lead to Project Manager"*: This project will involve a design change. Currently, we do not have anyone with experience in this type of project. Additionally, we don't know the programming language. We will have to organize a lot of training sessions if we are to take this project. In my opinion, we are not ready to engage in projects of this type.
>
> *"Project Manager to Head of Department"*: This project involves a design change, and we do not have much experience in this area. Also, not many people in the company are well prepared for this. In my opinion, we could do the project, but we will need more time than usual.
>
> *"Head of Department to Upper Management"*: This project involves redefining the design. We have some people who have worked in this area, and others who

> know the programming language. So they can train other people. In my opinion, we should accept this project but with care.
>
> *“Upper Management to CEO”*: This project will show everyone our ability to remodel the design of an entire system. We have all the necessary knowledge and people to execute this project successfully. Some employees have already provided training in this field to other employees. In our opinion, we should not let this project escape us.
>
> *“CEO to client”*: This is the type of project in which our company specializes. We have executed many such projects for many important clients. Trust me when I say that we are the best company in the field to do this kind of project. It is my opinion that we can execute this project successfully in the required time.

Many software projects have enough messed up things happening with them that they have the potential of generating a few jokes of their own. Another one of my favorite jokes is *“Project delivered on time!”*. The funniest jokes will be the ones you have lived through, because beneath the humor lies an undesirable situation you somehow got accustomed to. With time, just like with a frog being slowly boiled alive[2], we get so acclimatized to such situations that various problems just start to look normal, and we don’t do anything about them anymore. If jokes start surrounding your project, it might be people’s way of dealing with the working environment slowly slipping into a state of normalization of deviance[3], since humans use jokes as a coping strategy to make light of bad situations.

If you’re in your first software development job, you might have no idea what I’m talking about. Yet! Instead, you probably find software development challenging and lots of fun. You’ve most likely been busy and focused on things like *“What books should every programmer read”*, *“10 programming languages to learn next* *year"*, *"What every software developer should know about such and such"*, *"10 ways to improve your code"*, and so on, and so forth. And that was just to get your appetite going. You then discovered design patterns, modular architecture, source control, concurrency, scalability, the cloud, functional programming, IoC, unit testing, TDD, regex, refactoring, code reviews, and principles like SOLID, KISS, DRY, YAGNI, or SRP. You know… the tools of the trade, the things you need to learn, the things you need to practice, the things that earn you the big bucks, or reputation and recognition on developer communities like Stack Overflow[4]. You have a blast learning all of these things. You don't feel entrapped or frustrated. So, where is the problem? Why do we need the warning signs?

First of all, these things are all about you. *"You"* need to learn. *"You"* need to do. *"You"* need to master the various topics. But unless *"you"* are building the next successful business out of your parents' garage and working on becoming your own boss, it won't be just about *"you"*. Most likely, you will start working in a company on a software project that isn't going smoothly and might possibly be heading towards partial or total failure. I mentioned above some reasons why software projects fail, but those represent the *"macro"* issues. They are the results of smaller things that happen every day. Little things that combine in unfortunate ways to cause lots of harmful effects. And those things won't be just about *"you"*. They will also be about *"others"*.

Secondly, most of the things I mentioned above gravitate towards the technical. You might be tempted to say, *"Well, duh! Isn't this Information Technology dummy"*? It is, but it's also about businesses. And every business is a people business. You will not just write code all day long until you retire; you will do lots of stuff that won't be coding but interacting with others in the company (or outside of it). With time, you take on new roles and responsibilities - you might lead teams and manage projects, you might mentor novices, hire and fire people, communicate with clients and executives, and all sorts of stuff that isn't technical in nature.

So this book isn't about the technical stuff, the languages, the tools, or the patterns. These are important too. But there are plenty of good books on each of those topics. This book is about the situations when, despite using all of these goodies you have at your disposal, you can only scratch your head and say, *"WTF happened?"*. This book is about what happens in the trenches, things that aren't technology related. It's about some bad things common to software projects that people discover the hard way, by ending up in a situation that they didn't recognize. You will walk into such situations just like you walk into quicksand. As time passes and you pay more attention and get more experience, you will recognize some of these things as annoying and disruptive to your daily work, causing unnecessary suffering and frustration to people working on software projects. It's a bunch of tiny things, *"micro"* things that contribute to the *"macro"* issues.

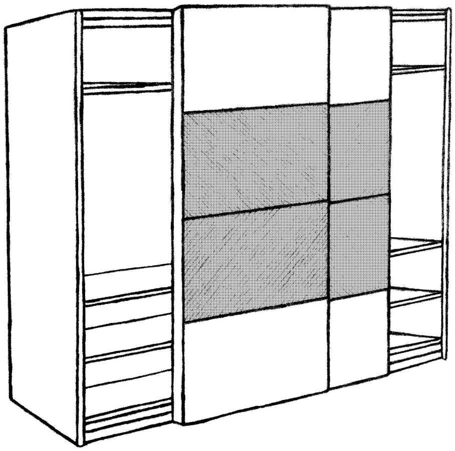
Heads up

This book is my attempt at a *"Heads up!"* for the new generations to know what else to expect when pursuing a career in software development. Young developers make the same mistakes as other developers did before them. People focus on technologies and how to use them best to write great software, while everything else just happens to them. Similar to learning how to drive a car, you start by concentrating more on the car and how to handle it, and only later on how to handle yourself in traffic. But we all know that being a driver isn't just about handling the car. It's about doing so on the road, in various weather conditions, together with other drivers, motorcyclists, pedestrians, or animals, paying attention to one's surroundings, and following and respecting the meaning of traffic signs. You might be a badass driver, drifting or doing donuts in parking lots, or breaking with your left foot, but if you don't know the meaning of signs like *"dangerous curve* *to the right, slippery road, railway level crossing, or give way"*, then you will be in for some unpleasant surprises. The signs are useful to let you know what you might encounter on the road ahead. Concentrating on the car and your driving skills is good, but it's even more important to pay attention to what's happening on the road. Similarly, good technical skills are a must for building software, but don't neglect what's around you: your environment and the things you encounter, challenges with the potential to transform into unpleasant situations that entrap or frustrate.

Just because you enjoy programming, don't think that a software development job will be just about programming in a company of like-minded people, working side by side with your developer buddies, eating pizza, sipping energy drinks, mastering the tools of the trade, and producing software that you can all be proud of. Some stuff will inevitably go into the weeds. So when the honeymoon is over, some of the things I mention in this book will be waiting for you around the bend. Be aware, be prepared, and pay attention to the warning sign.

I'm not trying to scare you. That's not my intention. There are a lot of positive things in software development. First of all, it's a lot of fun. Then there is always something new to learn or discover. You are not limited to working in just one domain; considering everyone needs software these days, you can combine programming with various other disciplines. You work with a lot of other smart people.

You are part of a community. You can work from home. You solve problems; the more complex, difficult, or challenging the problem, the more proud you feel when you make something work, you feel a sense of accomplishment. It's that satisfaction of creating something useful out of nothing (programming is an intellectual activity; you basically think a program into existence), something that other people use, sometimes that might even make a difference in their daily lives. You get a nice feeling when you see the result of your work. It's even more satisfying than assembling a sliding door wardrobe from scratch, all by yourself.



And the pay is pretty good too :).

But the reality is that, inevitably, many projects become *"stained"* by something. Developing software is fun and challenging, but often it's also hard and frustrating. Not just because the software itself is complex, but also because the process of making it is not without its challenges. Users never know what they want, the problems you are trying to solve might be very large and complex, you have technical, budget, business, legal, security, and time constraints, you depend on software you can't control (like operating systems, programming languages, libraries, frameworks, or platforms), it's almost impossible to test all potential inputs and execution paths, there is that elusive bug that keeps manifesting itself from time to time, and you just can't figure out its cause, and the list goes on and on and on.

Nothing new

What I write in this book is common knowledge. I haven't discovered anything new, haven't solved any mystery, haven't reached some higher level of understanding after scientific research, or aggregate the results of an IT survey with 50,000 respondents, from 2,500 companies, from 10 countries. I've just been around the block for a while. I'm trying to share some of the things I've learned at each stage of my career, highlight some of the things that can go wrong, and try to give some solutions I think might work to make things better. Anyone like me could have written a similar book. After a good amount of years working in this industry, in various contexts, with multiple people, you can write a book like this too. It's about experiences and lessons learned the hard way, things I wish I had known much earlier, or done differently, if only I had better understood them at the time.

I did a lot of dumb things and learned the hard way in the past 15 years or so. I've moved from being a young grasshopper[5] to senior developer, to Team Lead, to coach, and various mixed positions, both technical and managerial. Each stage brought new challenges and quicksands. The views, opinions, conclusion, and recommendations I write about in this book are thus my own, based on my own experiences. This is a personal story, and yours is likely to be different from mine. This book is about the quicksands you might encounter on the job, just like I did. If you are young, like writing code, and you are passionate about technology, it makes perfect sense to make a living from what you enjoy doing. But the way you program or build an application for yourself, or for school projects, won't be the same as when working in a company on projects with tight constraints and with lots of other people involved. There will also be frustrating aspects of being an employee programmer or developer.

I can't offer an exhaustive list of all of the problems you will encounter, and I can't talk about everything in great detail either. Nevertheless, I want to draw attention to some situations I have encountered more often and try to come up with some solutions based on my understanding and experience, so you too can recognize certain situations you are heading for and attempt to do something about them. Although I've done lots of things and had various responsibilities throughout the years, my perspective on things is, for the most part, that of a technical person. So you might view the same situations through different lenses. You might agree, disagree, and everything in between. Always use your judgment when trying to apply some of the proposed solutions. There are no guaranteed answers in this book. Some of the recommendations I offer may not apply to your situation, or things might not evolve exactly as described here. Your mileage may vary. Always look at the broader context to figure out what can actually work for you and within your environment.

There also isn't a certain progression in which these issues occur. You might not encounter them all, but you also can't steer clear of all of them. Many are not only specific to the software industry. Workplace issues can be found in any company. They come bundled together with having a job, and jobs sometimes may be unpleasant to some degree (that's why they pay you money to do it). Absurdity happens in the workplace because, as Scott Adams - creator of the now-classic book *"The Dilbert Principle"*[6] - mentions: *"everyone is an idiot"*.

Another thing to be aware of is that I don't always explicitly say throughout the book to what role I refer to when I talk about things, like for a programmer, a developer, Team Lead, manager, and so on. I can't say, *"As a developer expect this"*, *"As a Team*

Lead be aware of that" because many roles can have the same title in several companies, but not identical responsibilities. You might even be a developer with responsibilities of a Project Manager, or even vice versa. It's for you, from the context, to infer whose responsibility it is to avoid or fix some situation. And because responsibilities increase or change as your career progresses, you may have to come back to the book and reread it. Maybe now you're a programmer, and project management matters are not on your radar. But after a while, they may be. Perhaps you will even become a Project Manager. Who knows? Some things are valid for when you're a junior, others for when you're a senior, and some for when you're a Team Lead, for example. The quicksands are different in your new environment and in accordance with your new set of responsibilities.

When I started in this industry, all I knew was how to write code. It never even crossed my mind that I would write such a book. Many things interacted for me to get where I am today. Some were good, positive things. Others… not so much. The same thing happens with software projects. Many things interact to create either a good or a harmful situation, similar to how quicksand is formed: you need the right quantity of water and sand, certain friction between sand particles, water having no way to escape from the mixture, the force of the upward water opposing the force of gravity, or the right amount of pressure applied on the surface.

There are many other useful books that mention the topics covered here. Still, I couldn't find one that addresses a lot of them together. So I tried to cover a lot of ground. Maybe I insisted too much on some things, perhaps some of them I mentioned too briefly, maybe I repeat myself, and I'm sure I forgot some. But this is my attempt, and I hope it's successful, and you will find it useful. You be the judge of that. If you can learn from my mistakes and avoid some quicksand with what information you extract from this book, then I declare myself happy. If you encounter situations that I didn't mention in this book, want to send me comments, questions, or feedback, I would love to hear from you at quicksandssoftwaredevelopment@gmail.com. With this being said, let’s now put up some signs to warn about the quicksands and avoid falling in it.

Staff developers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Neglected process

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

A different approach

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Prepare

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The interview

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Loopholes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Match check

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Performance reviews

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Frequent discussions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Bad apples

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The working environment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Exit interview

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

It’s really important

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Continuous learning

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

English

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Solid foundations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Mistakes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Mentors

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Technologies

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Specialist vs. Generalist

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Too much information

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Problem-solving skills

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Business domain

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Your investment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Unlearning

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Programmer vs developer

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Team Lead

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Team Lead first

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Delegate

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Motivation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The working environment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Trade-offs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Health

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Sustained effort

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Software projects

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Estimations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Requirements

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Negotiations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Forgotten bits

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Risks and assumptions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The plan

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Change

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Measure

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Define success

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The deadline

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Sacrificing quality

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Other sacrifices

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Lessons learned

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

It takes longer

The Business-IT divide

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Us versus them

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Cost

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Shadow IT

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The real cost

Bridging the gap

Trust

Build trust

Meetings and reports

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Interruptions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Bad meetings

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Bad reports

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Better reports

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Better meetings

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Necessary

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Bad bosses

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Technical (in)competency

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Cogs in the machine

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

A losing battle

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Not bad

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Not as bad

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Bad managers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

In the middle

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Communication

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Failed communication

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Tools of the trade

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Project Manager first

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Overdoing it

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Just another boss

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Management style

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Lost cause

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Bad developers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Lack of skills

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Poor attitude

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Over-Engineering

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Not code? Not interested!

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Failed communication

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Lost cause

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Outsourcing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Still your people

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Hidden costs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Different cultures

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Final responsibility

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Scrum

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

You do it!

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The Scrum Master

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

The Product Owner

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Values and principles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Empowerment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Technical considerations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

It takes time

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Picking up the pieces

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Professionalism

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

Just one layer

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

In ending

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/quicksandsofsoftwaredevelopment.

References