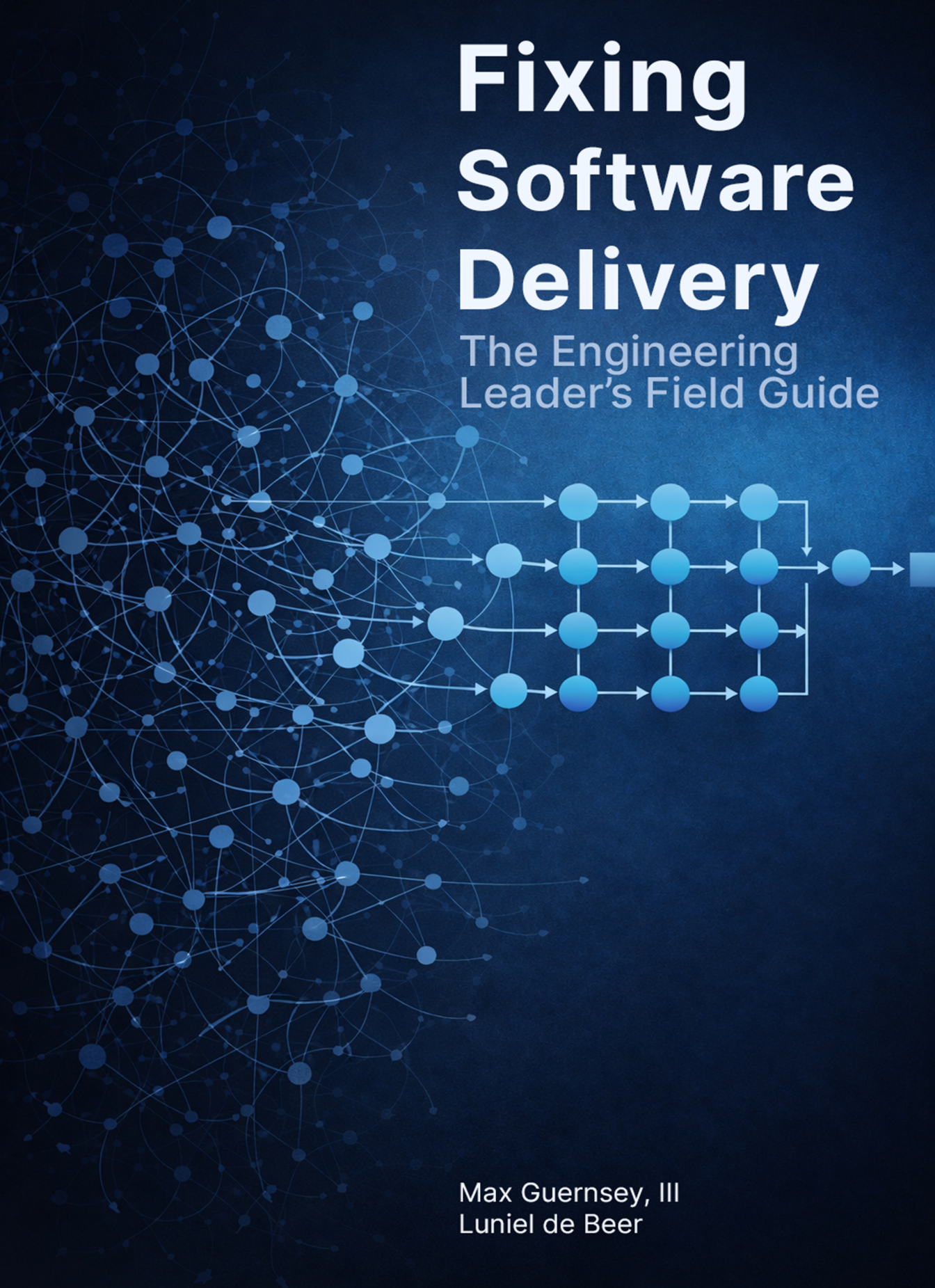
Fixing Software Delivery

The Engineering Leader's Field Guide



Max Guernsey, III
Luniel de Beer

Fixing Software Delivery

An Engineering Leader's Field Guide

Max Guernsey, III and Luniel de Beer

This book is available at https://leanpub.com/fixing-software-delivery

This version was published on 2026-03-23

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

How to Use This Book

This book is organized into short executive briefs (1-2 pages each), grouped into three parts:

1. Introduction–core ideas and mental model
2. Problems–frequent problems and solutions
3. Concepts–background and explanation of important concepts

Start with the five briefs in the Introduction. This prepares you to read any other section in the book.

The rest of the book is designed for on-demand use. Skip to the problem or concept you want clarity on. It can be read in isolation.

Use the book as a reference and return to it as new problems arise or when you need clarity on a topic.

Introduction

1-page summaries of key topics to use this guide.

Why Software Delivery Stalls

Software projects still struggle to meet business goals, even though the industry has been maturing for more than half a century. Many experts have analyzed this problem, but they view it through the lens of their own expertise, so their advice often fails to stick.

The following have been tried...

- New processes (like Scrum)
- Better adherence to processes
- Removing process altogether
- Increasing analysis
- Eliminating analysis entirely
- Technical skills training
- Adding people
- Replacing people with automation

But these efforts rarely address the most impactful causes:

- Poor information flow
- Code that is hard to understand or work with
- Insufficient clarity on requirements
- Technical habits that cause systems to decay over time
- Limited visibility into implementation work
- Low/no ownership of outcomes
- Weak connection between business goal and the work teams are doing

Most delivery-improvement initiatives fail because they address symptoms rather than causes. Leaders who focus on root causes usually achieve far better results than those who apply canned solutions.

What “Quality” Means

Quality and software delivered are linked, but the kind of quality that leads to sustainability is hard to measure.

All numerical measurements are proxies:

- # of bugs
- # of support incidents
- # of times the same code is changed
- # of times tickets reopened

All numerical measurements either:

1. Trail engineering decisions with a long delay (sometimes even years)
2. Track symptoms instead of quality itself

Because of this, quality cannot be directly measured with quantitative analysis. It can reveal outcomes, but not causes.

To understand quality problems, look for recurring sources of delay and risk, like:

- Parts of the code devs don’t want to touch
- Functionality or features that are known sources of pain/bugs
- Recurring incidents of tickets staying open much longer than expected

Focus on finding large or repeating impediments. These are the clearest indicators of underlying quality problems.

When Metrics Can Hurt

Executives are under strong pressure to define problems and track solutions in terms of metrics. Metrics are critical to business but, as with quality, the most meaningful ones trail improvements with a significant delay.

Organizations often introduce metrics hoping to show progress sooner, such as:

- # of bugs opened and closed within a Sprint
- Team velocity
- # of stories carried over between Sprints
- Cycle time for backlog items

These are all proxies for the desired behavior, but they are easy to game and often get treated as goals rather than signals. As a result, metrics meant to drive improvements are often counterproductive.

It can take years for a team's culture or code to express problems in terms of metrics, and just as long for improvements to take effect.

In the long run, there are metrics that track real impact:

- Time from ideation to realization of a solution
- Cost savings over a significant period of time
- Improved user retention/satisfaction

The challenge is that the meaningful metrics are too slow to be used in tuning. For short-term tuning, rely on technical expertise and reasoned judgments. Over time, the benefits will appear in business metrics.

What Holds Back Change

Every leader knows there is a risk that coaching or training will not stick, wasting time and money. This can happen for a variety of reasons:

1. Teams give up before value is delivered
2. Teams do not gain enough mastery during training
3. Nothing compels teams to continue practicing after training

It almost always boils down to the team not making it through the "Valley of Despair"—the phase of adoption where they've realized the change is hard but have not yet seen benefits.

The Valley of Despair is a problem because change is **work**. This work is often invisible to management, and pressure from more visible work pushes back against efforts to improve.

The solution is to treat change as a first-class initiative. Track it and budget for it the same way you would an improvement to the code. This also creates accountability around learning and adoption.

It is also valuable make expertise available to teams in support of the change. In addition to accelerating learning, expert guidance can help teams "keep the faith" while they pass through the Valley of Despair.

The New Angle

Software initiatives fail because most improvements focus on effects rather than causes. One of the most impactful causes is software quality.

Improving quality and removing other impediments to delivery is difficult to measure numerically. Meaningful quantitative metrics lag significantly, and others distort the signal.

At the same time, changing how a team works is itself difficult and requires sustained effort.

To thrive in this industry, a different angle is required:

1. Be root-cause focused
2. Set expectations that improvements have a long term impact
3. Use business metrics for quantitative analysis
4. Use qualitative signals to evaluate software quality
5. Favor reason and judgment over metrics for short-term tuning decisions
6. Treat improvements to software delivery as first-class initiatives alongside product work

Delivery Is Slow, Blocked or Unpredictable

Work does not complete when needed. It takes longer than expected, spills across iterations, or stalls entirely.

Chronic Iteration Carryover

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Deluge of Defects

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Disruptive Unplanned Work

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Constant Back-and-Forth on Requirements

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Delivery Slows as Organization Grows

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Marked Done When It's Not

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Time Wasted Repeatedly Fixing the Same Things

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Slows When Requirements Mean Different Things to Different People

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Low Morale from Burnout

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Late-Stage Rework

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Consistently Blocked Waiting for Clarification

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Consistently Blocked by Foreseeable Impediments

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Progress Stalls Due to Coordination Overhead

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Held Open Because “Done” Keeps Changing

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Fragmented/Rushed at the Ends of Cycles

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Fails to Produce Business Outcomes

Work items are completed reliably, but the results do not move the business closer to its goals.

Lots of Activity, Little Progress

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Late Disagreements about Scope

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Completed Work Frequently Reopened

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Gets Done but Isn't Accepted

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Top Priorities Do Not Get Done

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Teams Interfere with Each Others' Work

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Cannot be Completed Predictably

Work cannot be executed predictably. Timelines and expectations are consistently wrong, even with deliberate effort.

Wildly Wrong Estimates, Consistently

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Spins out of Control

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Inability to Predict Impact (Even w/Effort)

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Scope Cut Late in Project

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Trust in Commitments Breaks Down

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Deadlines Force Compromised Delivery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Deadlines Regularly Missed

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Cost of Change is too High

Small changes require disproportionate effort. The cost and risk of change increase over time.

Teams Avoid Certain Kinds of Changes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work in Certain Areas Sluggish

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Long Stabilization Phases

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

QA/Testing Bottlenecks

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Unsustainable Codebase

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

The Rewrite Cycle

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Skyrocketing Cost of Ownership

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Unstable Codebase

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Work Quality Degrades Near Deadlines

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Quality Collapses Right before Delivery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Defects Spike After Delivery

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.

Spikes in Technical Debt

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/fixing-software-delivery.