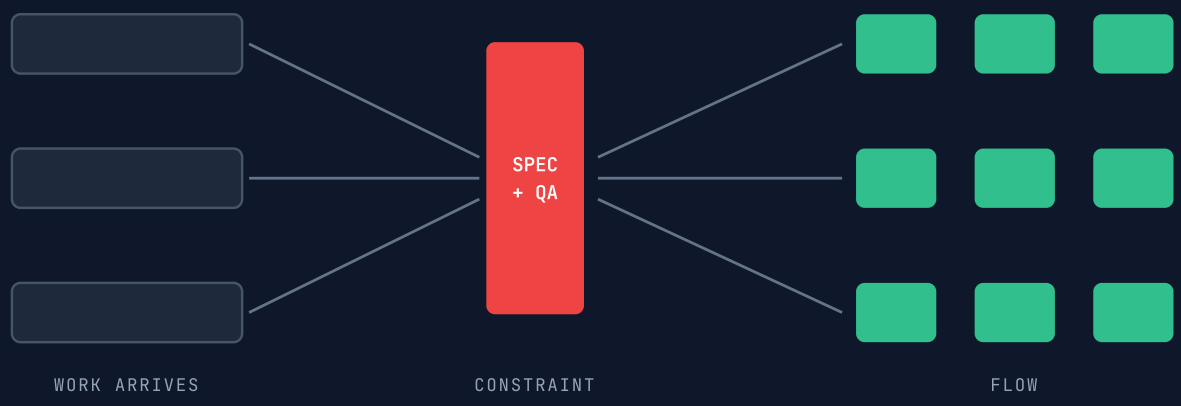
ENGINEERING LEADERSHIP PLAYBOOK

POST-AGILE

A Framework for Software Product Management and Engineering in the Age of AI



By Olivier Bregeras • 2026 Edition • 26-Minute Read

v1.0

CONTENTS

Introduction

01. The Three Pillars of Post-Agile

02. The New Team Topology

03. The Transition Playbook

Conclusion

Sources

INTRODUCTION

The Bottleneck Has Shifted

A feature that used to take a sprint now takes an afternoon. The clarification of what that feature should do still takes a week.

That inversion is the subject of this book. For as long as anyone now practicing has been employed, the expensive part of software was translation: turning a clear intent into working syntax. We built an entire industry of process around that cost: the planning, the estimating, the handoffs, the ceremonies that existed to keep a slow pipeline from jamming.

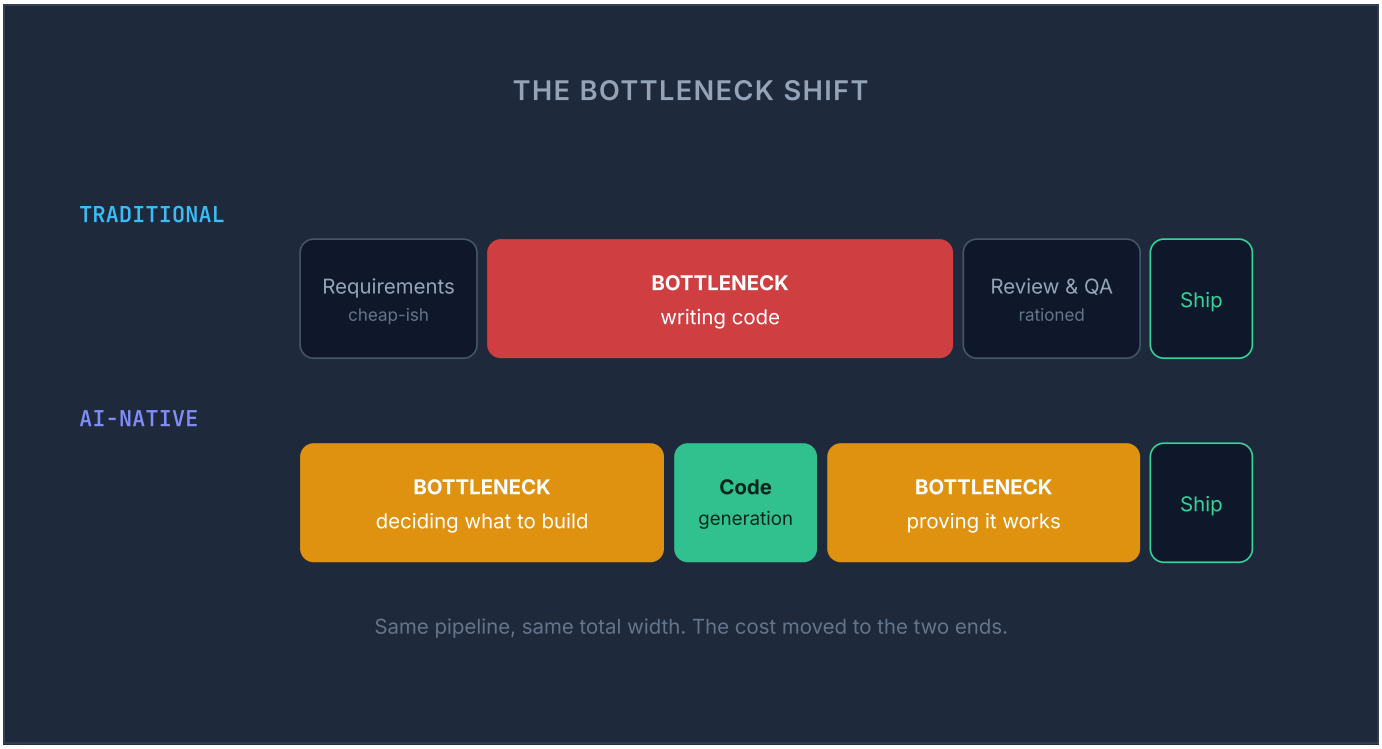
The cost of writing code collapsed. The cost of deciding what to build did not. Everything downstream of that asymmetry is now mispriced.

What actually changed

Generation is cheap. Review is not. An agent can produce a plausible implementation of a vague request before lunch, and the bill arrives in the afternoon, when someone has to decide whether the thing it built is the thing you meant.

Quality assurance has the same shape. When code was expensive, review was the last checkpoint before release, and a reviewer's attention was rationed against the cost of writing. When code is cheap, review becomes the narrowest part of the system. A human reading every diff is still a bottleneck, even when they call it diligence.

The bottleneck moved to the two ends of the pipeline: the precision of the requirement, and the proof that the result is correct. Those were always the hard parts. They were hidden behind the cost of typing, which was high enough that nobody had to look.



Now the typing is free and the hiding is over. **The ambiguity you used to resolve later, in conversation, now gets resolved immediately, by a machine, without asking you.** That is a failure mode the old process never had.

Agile was built for human latency

Sprints, standups, and planning sessions were coordination machinery for a specific set of constraints: people forget, people context-switch, people need to agree before work starts, and deployment was slow enough that waiting was normal.

Those constraints were real. They are no longer the binding ones.

A two-week cadence around work that completes in minutes does not slow the work down. It slows the feedback. You still get the code quickly; you just don't find out whether it was right until the ceremony comes around.

If you have ever sat in a standup where people reported progress on work a machine had already finished, you have watched a ceremony outlive its reason. **The ceremony is not stupid. It is historical.**

None of this makes the people who wrote Agile wrong. They solved the problem in front of them, and they solved it well. The problem moved.

What this book delivers

Three mechanisms replace three ceremonies:

- **Shape Up** replaces estimation with an appetite and a pitch.
- **Specification-Driven Development** replaces the user story with an executable `spec.md`.
- **Continuous Flow** replaces the sprint boundary with work-in-progress limits and shipping on green.

Three roles replace the org chart's assumptions: **Shapers** who decide what is worth doing, **Builders** who execute against a spec, and **Orchestrators** who hold the definition of done.

A three-phase migration gets you from here to there, in an order that does not collapse under its own ambition.

Each chapter does one job:

- **Chapter 1** defines the three mechanisms.
- **Chapter 2** assigns them to people.
- **Chapter 3** sequences the change.

This introduction names them and stops, because a manifesto that explains itself in the foreword has nothing left to say.

The bottleneck moved. Everything that follows is about what to build around its new position.