Sprinting In Place: Why Your Agile Team Isn't Agile and What You Can Do About It

INTRODUCTION

The dictionary says that "agile" means "moving quickly and easily" and gives synonyms like "nimble" and "acrobatic". With this in mind, what should we expect from an "agile software development" team? Surely they would whip out changes fast, respond quickly to business needs, and innovate relentlessly. As a result, productivity, team morale, and confidence would be consistently high, and internal and external customers would be delighted with their work.

If the previous paragraph describes the development team you work with, this book is *not for you*. We wrote it for everyone else - those of you involved with teams who follow agile practises like daily standups, weekly sprints, and regular retrospectives, but nevertheless blow deadlines, build the wrong features, and disappoint internal and external customers. You may be…

- ...an executive or company founder at her wits' end trying to explain yet another project delay to the board, or
- ...a CTO or technical lead stuck between impossible customer demands and a mountain of technical debt, or
- ...a member of a development team who wishes she could build something meaningful but instead is trapped on a treadmill of context-free features and bugs with unrealistic deadlines.

In our careers, we've helped loads of teams overcome these frustrations and get real agile results, and the techniques in this book are designed to help you do the same.

If you flipped through the book before reading this introduction, you might have noticed that we don't have much to say about the mechanics of agile development (sprints vs iterations vs kanban) or its various methodological flavours (Scrum vs XP vs AUP). Instead, our approach, based on Chris Argyris's Action Science methods, is all about humans and their relationships: how they can exchange valid information, jointly design solutions, and make meaningful commitments. We find these are common needs no matter what type of agile team you have, and that relationship-based techniques work whether you use Scrum with month-long sprints or Kanban with complex swimlanes.

To help you master the methods we describe here, we've organised the book around the values described in the Agile Manifesto (http://agilemanifesto.org), a document from 2001 that unified a wide variety of practitioners under the then-new term "agile software development". The values map very nicely to our practises, and as a bonus, the values are not in any particular order, so you can turn to the chapter that appeals to you most right now without worrying that you'll miss something explained earlier.

Before you do that though, you might want to have a look at the next chapter, which explains that everything is your fault. Seriously, it will really help.

- Jeffrey and Squirrel