Preface

Books such as Tom Demarco and Tim Lister's *Peopleware* and Fred Brooks's *Mythical Man-Month* are timeless. This is both because the advice provided is practical and because this advice is still rarely adopted. We all read these calls to action and imagine a better work environment, and then we lament that our managers aren't taking charge of change.

Tagging along for the ride is easy, but to be effective you need to be an active, proactive, and intentional team member. We must each take active responsibility for making things happen, for committing to and fostering a healthy environment in which great products can be built.

This is a book for everyone involved in software development, not just the managers or anointed change agents who are deemed as the point people to drive change. We all need to actively drive our work environment—it is not just a management concern. As individuals, we can all significantly improve our productivity. Practitioners are the ones best suited to propose and contribute to changes to our overall team performance.

"Leadership" here is cooperatively participating with others to most effectively achieve a common goal. A team of effective leaders can work miracles. We all need to take ownership and convert these miracles into reality on our teams.

Many of the subjects discussed in this book aren't necessarily about software development and could readily apply to almost any team environment. Indeed, many of the principles and practices discussed here are common across disciplines outside of software development. This book, however, was developed based on my experience in and with software teams, and a few factors seem to make software development experiences different:

- Because software is seen as abstract and intangible, there is generally little common understanding of the expected outcome or the scope of work.
- Because training for this field is primarily technical, there is rarely explicit focus on the management of individual attitudes or team dynamics.
- Because of these factors, teams often experience pain on projects, sometimes failing outright.
- Even so, team sustainability is rarely considered as a key success factor.

So, in one way, this book is all about software development. This is the domain in which dysfunction is most rampant, more so than in other fields I know anything about. Best practices are always discussed but rarely successfully applied. The typical approach for fixing problems is through a technical solution, by specifying more practices that end up further constraining the team. There has to be a better way.

It Is Time We Get Started

There really is not, and never will be, a "silver bullet," that magical solution to our woes. It's one thing to have Fred Brooks suggest that there were no silver bullets in 1986, and another thing altogether to have hopes dashed through trial and error of personal experience. We can't expect a hero to come along to save the day. It is up to all of us to contribute in a positive way to improving the experience of software development any way we can.

Having worked with a number of those who believe themselves to be part of the solution for this industry, I've come to the conclusion that we are all similarly challenged to some degree. Most consultants suffer from many of the same issues as the clients they are proposing to help. We all live in our own box, and we all find it difficult to see things from outside our box.

So, if there is no silver bullet on the horizon, no real heroes, and we are all in our own box, what are we to do? The solution to our problem is a *coordinated team* effort to make our life better. We all need to contribute to this solution, and these contributions need to work together for the common good.

We need to drive change ourselves. We can't blame external circumstances forever, and we can no longer rest on the relative youth of the software industry as an excuse. Although medicine has been called the youngest science, with a vast amount of information to absorb and a high rate of change, practitioners dare not use these challenges as excuses for failure. Here in the software industry, we also have much to learn and experience a high rate of change, but we have been much less proactive in managing our challenges.

We are the solution that we have been waiting for all this time. We are the ones who have the responsibility to make our software development experiences stronger. We need to start today.

We all have something to contribute, even if it is just a new set of eyes to look at a problem from a different angle. Smooth running teams build on each other's strengths to form a stronger whole.

Change is made one step at a time. Although it's nice to have an overall context or strategic goal to work toward, the problems in most software teams are so extreme, so rampant, so fundamental that even a few minor tweaks can do wonders. I've seen more than one organization turn things around with one or two well-selected adjustments.

Indeed, even with major dysfunction, massive changes will often bring more negative culture shock than positive benefits. There are companies that no longer exist today, at least partially because of the impact of a major improvement initiative.

You won't find a methodology in this book. There are plenty of them to go around, either too narrow to be widely applicable or too broad to be easily applied to a specific situation.

You won't find prescriptions or checklists here either. These are best when tuned to your specific culture, even if they are based on some of the excellent sources available in the software literature.

What you will find is an exploration of the problems that we need to consider, a reasonable order by which we should tackle those problems, and recommendations about how to deal with each stage. All points to consider, but no book will give you *the* answer.

This book is about making changes palatable to the people on our teams. Small, focused changes, with consideration to the existing culture.

The Lay of the Land

This book is divided into six major parts.

Part 1 focuses on the state of the industry today. For all the efforts spent on improvement, we are not significantly better off than we were 30 years ago. We explore the issues with both anecdotes and hard data, and I suggest a sequence of stages that can lead to a stronger overall solution.

We start through these stages in Part 2, and consider how everyone has a different view of any situation, and brings to the table unique viewpoints and emotions shaped by past preconceptions and experiences. All of these are important to consider, and should be dealt with in a conscious, objective fashion.

A range of interactions and forces will govern any group, in the workplace or elsewhere. These will strongly influence the overall nature of the relationships and the resultant quality of the work produced. Just as Part 2 explored individuals, Part 3 looks at the issues around group dynamics and relating with others.

Part 4 explores what happens as this group starts to organize into a team. The core issues here are how to effectively organize, align, guide, and coordinate the team toward a common goal.

After the team has dealt with the issues of coordination, they can look at the issues around stakeholders in Part 5. Here, the deepest challenges are in shared communication and management of change as work progresses toward successfully solving problems.

Finally, Part 6 deals with the issues of introducing change. Doing this in any form is difficult, and driving change within a comprehensive model while minimizing the negative impact can be even more so. This section identifies several guiding principles to use while fostering change in your organization.

Although the book proceeds logically, you could easily dive into each section independently. Each chapter is self-contained, but the topics are arranged in a progression that explores the range of issues from problem to solution. Icons in the margins highlight symptoms of trouble, success indicators, questions to ask, and tools to use, and are explained inside the front cover.

These reference points are thoroughly captured on the four inside covers of the book, providing a navigation guide for Parts 2 through 5.

Origins

The vast majority of the topics covered in this book have come directly from my 10 years of consulting experience, and from my direct software development experience over the 20 years before that. A problem would arise. We would work to distill the essence of the problem and identify an approach, and then resolve that problem. I've personally worked on embedded systems, air traffic control systems, and shrink-wrap products, and consulted to companies in a broad range of sectors. No one sector has yet solved the challenge of growing a sustainable team without pain, but there have been hints of success everywhere.

I published much of this material in weekly newsletters over the past five years. Initially, it seemed aggressive to launch a newsletter on a weekly basis. Over the years, however, it has been easy to consistently find new situations to write about. There is no shortage of challenges facing today's software teams. Many of those who believe it is a smooth ride are either deluding themselves or are being deluded by others.

The issues discussed herein occur in teams ranging from very early-stage start-ups (and even situations that could easily be described as pre-start-up, where they don't even realize they are trying to solve a problem with software) to very large, mature, safety-critical applications.

Many of the groups discussed in this book are only marginally aware that software development frameworks and maturity models exist, but others have been assessed at high maturity levels based on these models. Some have used established frameworks to varying degrees of success and even contributed to best practice content over the years.

The views here are not intended to replace existing approaches to resolving challenges in the software industry, but are seen more as a complementary but neglected space. We are all humans interacting to build novel systems, and to do so effectively, we need to be engaged, working in an environment that effectively channels our creativity. To some degree, what is discussed in this book is a precursor to allowing you to effectively use the more commonly discussed engineering solutions.

Anecdotes in this book reflect actual situations and events that have taken place, from a wide variety of situations. These have intentionally not been tweaked to appear as a single team throughout the book. Some readers might recognize themselves in these pages; I hope this recognition will be taken as positive.