Extracted from:

Real-World Kanban

Do Less, Accomplish More with Lean Thinking

This PDF file contains pages extracted from *Real-World Kanban*, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit http://www.pragprog.com.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printed versions; the content is otherwise identical.

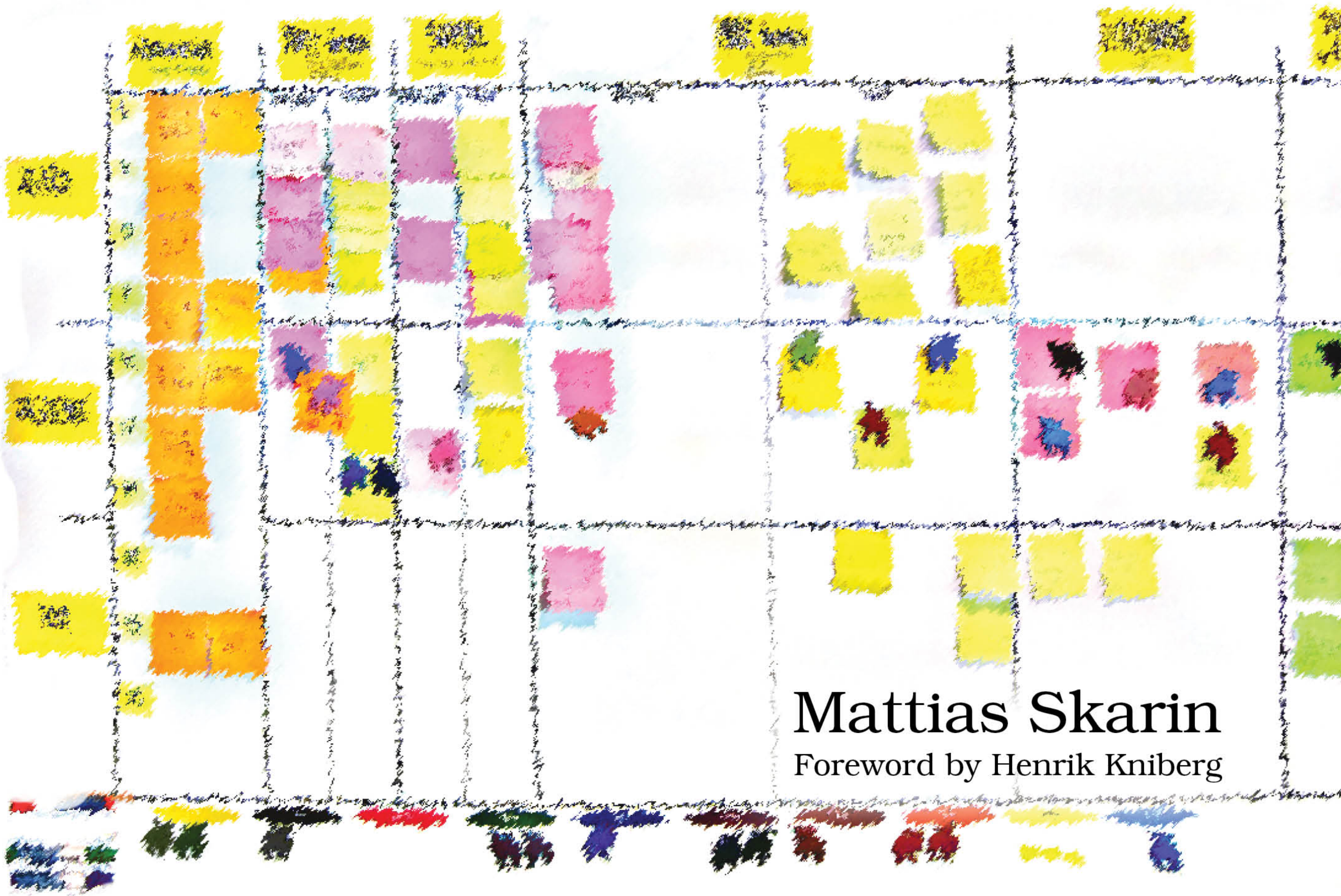
The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina

The Pragmatic Programmers

Real-World Kanban

Do Less, Accomplish More with Lean Thinking



Mattias Skarin

Foreword by Henrik Kniberg

edited by Fahmida Y. Rashid

Real-World Kanban

Do Less, Accomplish More with Lean Thinking

Mattias Skarin

The Pragmatic Bookshelf
Dallas, Texas • Raleigh, North Carolina

Pragmatic Bookshelf

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

Our Pragmatic courses, workshops, and other products can help you and your team create better software and have more fun. For more information, as well as the latest Pragmatic titles, please visit us at *https://pragprog.com*.

The team that produced this book includes:

Fahmida Y. Rashid (editor)
Potomac Indexing, LLC (indexer)
Cathleen Small (copyeditor)
Dave Thomas (typesetter)
Janet Furlow (producer)
Ellie Callahan (support)

For international rights, please contact *rights@pragprog.com*.

Printed in the United States of America.
ISBN-13: 978-1-68050-077-6
Encoded using the finest acid-free high-entropy binary digits.
Book version: P1.0—June 2015

Find Opportunities to Improve

It's all too easy to get emotionally attached to your current approach.

The first lesson we learned was to avoid scoring ourselves against a process model, an organizational structure, or a tool. Instead, you should build a culture of curiosity and experimental thinking. Train managers to hone their insights by observing outcome and to suggest improvements from there. This will help you learn from new information even if it is at odds with your current understanding, and it will help you avoid getting too emotionally attached to your current approach.

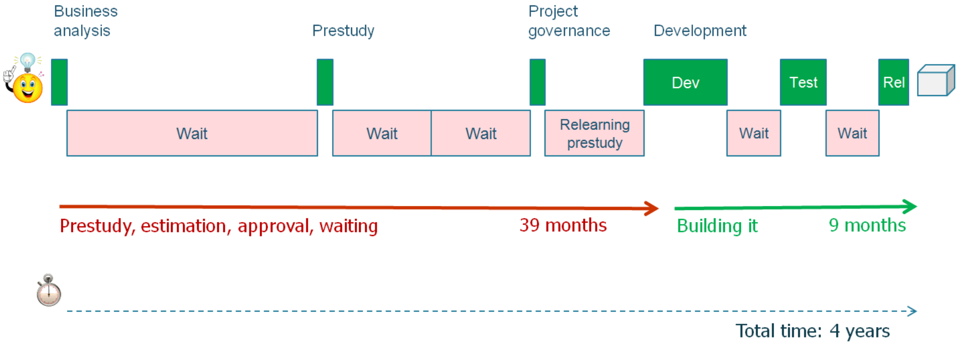
While this book talks about process, what you won't see if you go looking for process artifacts is the amount of effort we put in each case to develop thinking people who take initiatives, managers included. The key to sustained progress is not so much about implementing a process, but about developing a conducive culture for thinking people to help you evolve—people who care about *what* and *why*. Managers who fail to nurture thinking people as their key objective will just copy and paste others' solutions. These managers tend to be oblivious when their current approach fails to work.

Finally, we learned to improve end to end, not just our little part. Having great people is not enough, because they could be trapped in a dysfunctional system where they are unable to ship any high-quality products. To see these challenges, management needs to think about optimizing the whole, not just the parts. Shifting their attention lets them see how ideas get diluted through handovers and lets them grasp the delayed effect of pushing poor-quality products down the line.

A Change Should Never Come as a Surprise

A sanity check for long-term leadership is that a change should never come as a surprise. Follow this rule, and it will help you build up trust capital. This trust capital is hard currency when it comes to taking risk in periods of uncertainty.

Let's take a look at an example—a value-stream map over an idea's journey through a real company.



From this perspective, it's easy to spot improvement opportunities. In this case, the front-end part of the value stream (prestudy, estimation, approval, and waiting – 39 months) accounts for 81 percent of the total lead time. Even in Agile companies, the front-end parts of the value stream are generally quite intact and take up roughly 60 percent or more of the total lead time. The real trouble happens if this is spent planning rather than generating new information—for example, through market and user feedback.

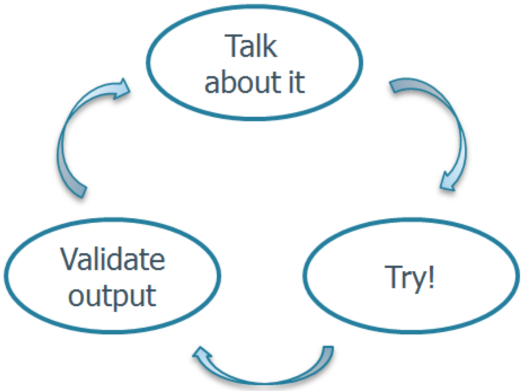
The problem is, we rarely see things from this perspective. The fact that the lead time is long would most likely go unnoticed among the people working on the project, because they have their nose so close to the grindstone that they see only their own part.

Another reason you don't see many people shifting perspectives has to do with our ingrained mental habits. When observing the world, our mind has to decide what information to act on. So if you see something that your mind doesn't think is important, you subconsciously ignore it. This is known as *change blindness*. So in a sense, we are predisposed to ignore and discard new information, even when it is staring us in the face.

The crucial first step in unlocking these improvement opportunities is to shift management's perspective from perfecting each part to improving the performance of the whole (for example, end-to-end lead time). The goal should not be to implement the perfect process, but simply to make it easy for great people to deliver value and to fix quality problems as soon as possible.

How We Got Buy-in to Change

Since it is our job to build thinking people who will be the ones to tweak processes and tools long after we are gone, we have to bear that in mind when we introduce changes. The way we introduce changes is actually really simple:



Talk about it. Talking to people before initiating a change is treating people you work with as thinking, intelligent beings. This is a necessary prerequisite to taking responsibility for the process and the quality produced. Talking with people gives them the chance to weigh in on what is going on and to provide better options.

Your leadership is only as strong as the conversation you are ready to have.

Try! People often have many opinions about how things should work. But there's only one way to find out how things really work, and that's by trying them out. It is often helpful to implement a change on an experimental basis for a limited time and then evaluate whether things get better.

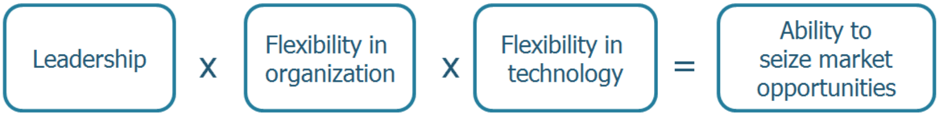
If you meet resistance, start with the pioneers in the company. You don't need to wait for everyone to get on board. Seed curiosity by handing out books or visiting others who have made the journey.

Evaluate output. It's easy to confuse actions (fulfillment of your implementation plan) with real improvement. You only really know whether your actions generated the desired effect by observing real outcome (product quality, lead time, predictability on delivery, and so on). The first role to connect with output is management. As a rule of thumb, management is responsible for the outcome of repetitive systems, because they have significant influence on system conditions such as who to hire, how the work is done, and what investments to make.

Improve the Organization with Long-Term Thinking

It is all too easy to fall into the trap of becoming reactive and activity-driven in the world of product development. But succeeding with product development is not a random happening. Using long-term thinking, you can build in *system enablers* in your culture, organization, and technology that can reduce friction for initiatives and can accelerate learning. This will enhance your ability to spot, seize, and explore market opportunities.

Let's introduce a simple model.


Leadership
x
Flexibility in organization
x
Flexibility in technology
=
Ability to seize market opportunities

The key to reading this model is to realize that the synergy of culture, leadership, and technology produce short-term agility. They work together. So if you invest in only one component and ignore the others, your payoff will be limited.

Leadership is your ability to seize opportunities and to turn information into action. This requires building up people and managers who are problem solvers and who take initiative when opportunities present themselves. Their efforts need to be supported by a system that is conducive to experimentation. Also, finding alignment across organizational borders must happen swiftly, without distorting the original problem. This is a challenge for tall hierarchies, which tend to have very slow decision cycles. A better approach would be to adopt a long-term leading strategy by building a culture focused on training.

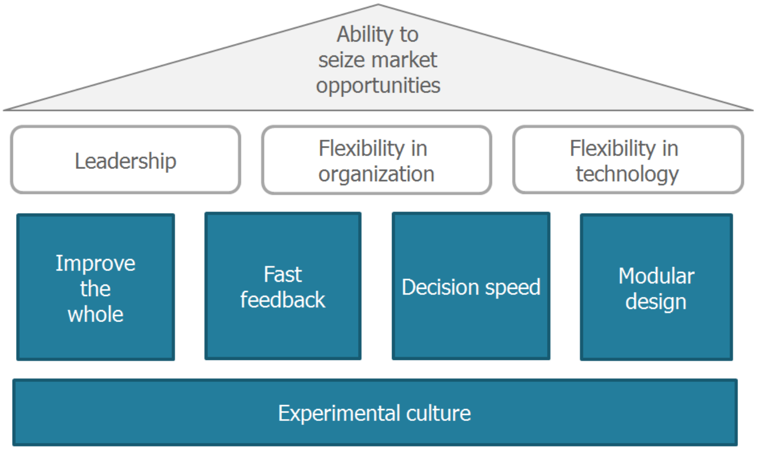
Flexibility in organization is your ability to adapt structures, rebalance teams, change processes, and build skills to seize market opportunities. This requires making the leap from a familiar comfort zone to something new and unproven. Making this leap requires *trust capital*, a certain amount of trust in your team's ability to deliver. Without trust capital, people in the organization will struggle during periods of uncertainty, look upon initiatives with suspicion instead of curiosity, and err on the side of safe bets. The good news is that management can choose to build up trust capital systematically. This is your buffer to handle risk and periods of uncertainty.

Clarifying your intentions, improving the whole, having regular conversations with people doing the work, and taking part in problem solving are examples of leadership behaviors that can help you build up trust capital.

Flexibility in technology is the ability to adapt your architecture and technology choices to late information.

If you find yourself in a vicious cycle of fighting fires and dealing with late surprises and late deliveries, what long-term leadership behaviors and system capabilities should you pay attention to? This is not always easy to see in the heat of the moment. To help you out, I've extracted five factors from the case studies in the book that you wouldn't see by simply looking at the Kanban boards. To notice them, you would have to observe events and behaviors for a longer period of time. What these five factors have in common is that they provide focus and direction for improvements and behaviors driven by the management team.

"All models are wrong, some are useful," statistician George E.P. Box wrote in *Empirical Model-Building and Response Surfaces*, and this is no exception. I've found models useful in helping to break out of vicious short-term cycles, so I share them with you in the following figure and table.


Ability to seize market opportunities
Leadership
Flexibility in organization
Flexibility in technology
Improve the whole
Fast feedback
Decision speed
Modular design
Experimental culture

System enabler	Examples
Improve the whole	• Improve lead time through the full value chain, not just through individual functions. • Test working user scenarios instead of individual functions. • Reduce total costs, not just IT costs.
Fast feedback	Adapt your products to make them fit for your purpose by learning quickly through continuous feedback.
Decision speed	Keep up with market pace by detecting weak signals and doing something about them.

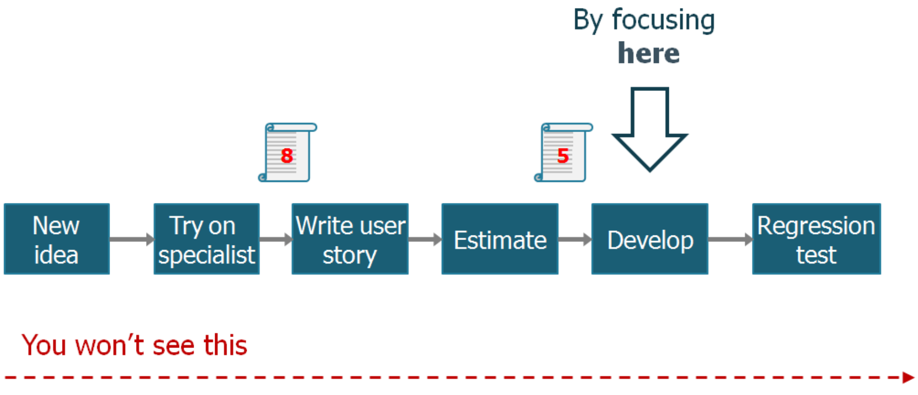
System enabler	Examples
Modular design	Employ product design using loosely coupled modules.
Experimental culture	Shift culture from "No, it's not the way we do it around here" to "Cool, how can we find out whether it works?" using small experiments. It starts with your management team.

I have met people who push the above buttons either intuitively or out of experience. The trouble with intuition is that it's not transferable or teachable. I'm making the intuition explicit here to give some guidance on system capabilities a management team should pay attention to in the long run to stay competitive. I invite you to challenge your own ideas of long-term improvements. And if the only thing this chapter does is to push you to be more concrete with your long-term improvement ideas, to write them down and share them, that will already be a great leap forward.

The economics of the above capabilities and behaviors are such that they are cheap to implement during the early phases of product lifecycle and growth, but very expensive to catch up with later. That is why a management team benefits from paying systematic attention to them, especially during periods of rampant growth.

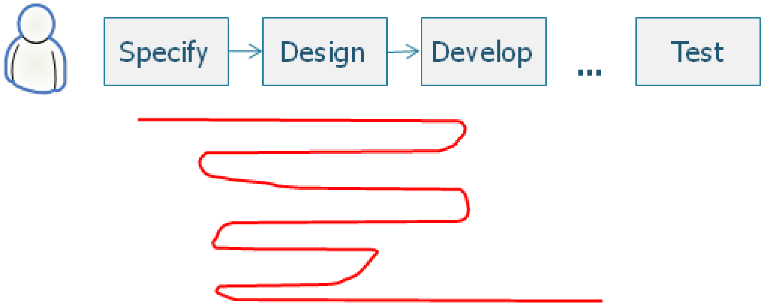
Focus on the Whole, Not Just the Parts

Improving the whole before the parts jars traditional thinking on how to manage, which is a legacy from the scale economy of the early twentieth century. As you can see in the following figure, the trouble with successful management of parts is that you lose focus on the whole, and this inadvertently becomes someone else's responsibility.



We touched on the importance of shifting behaviors in management from improving parts to improving the whole already.

A simple example of improvements that go unnoticed when perfecting parts is rework. Rework, often due to poor quality, hides between functions and is often invisible to the participants in a value stream.



In *Enterprise Kanban: Improve the Full Value Chain*, we chose the improvement perspective of end-to-end flow, and in *Using Kanban to Save a Derailing Project*, we used Kanban to visualize flow all the way to customer use, which enabled the parties involved to focus on solving one problem at a time.

Solicit Fast Feedback to Stay on the Right Track

No product is perfect the first time. No problem is perfectly solved by the first solution. Generally, some tweaking is needed once a solution is put to use for the first time. But the trick is not to wait for perfect information before starting. In any complex environment, waiting for the perfect solution means waiting until it's too late.

The trick is to leverage feedback throughout development to find out if you are on the right track.

For short-term cost-optimization purposes, it can be tempting to see test environments, test harnesses, simulations, and local prototyping abilities as cost-saving opportunities. But if the net effect is prolonged feedback cycles, they will produce devastating effects on your upcoming product development performance.

Feedback loops can help us understand if we are solving the right problem and if we are solving the problem the right way. Here are a few feedback loop examples. What is the time to feedback in your organization?

Feedback loop examples:

Product/Market fit	Who is my customer? What problems does he have? Are there different user roles involved?
Environment fit	What does the ecosystem look like where our product will be used?
Problem/Solution fit	Do my solution options fit the problem? What's the smallest thing that could possibly work? What product capabilities are important (performance, maintainability)?
Scenario fit	In what user scenario will our product work add value?
Product/Integration fit	How do we know if our product works as a whole? Are there any interference and compatibility issues?

Anything with "big" in it generally correlates negatively with learning from feedback. It's because there is a built-in incentive to stick with and to ratify earlier decisions rather than to act on new information, because change can be costly.

For both hardware and software, time to feedback is a leading indicator of innovation speed.

In *Enterprise Kanban: Improve the Full Value Chain*, early investments in automated tests enabled development teams and idea owners to shift focus and energy from validating what was expected to work to whether they were meeting customer expectations. In *Using Kanban to Save a Derailing Project*, time to feedback for both working solution and individual function was improved by faster test cycles.

Speed Up Time Spent on Decisions

An organization's ability to solve problems and make good decisions is key to keeping up pace. The fact that decision pace tends to slow down as an organization grows often goes unnoticed. In the worst case, denial prevails while voices of concern from real users or engineers are simply ignored instead of productively addressed, and the project eventually fails. Getting the right information at the right time is crucial to your ability to make shrewd decisions efficiently.

When your organization becomes Leaner or more Agile, it will put pressure on your decision cycles. One telltale sign that a decision cycle needs improvement is stress building up in management teams and "discussions at work." As flow improves, problem solving and tradeoff decisions need to happen daily instead of at the end of projects or on a monthly basis. Resource-allocation decisions need to move from yearly budgets to quarterly or monthly. One of the solutions that will shorten decision cycles is moving to a more decentralized organization.

But making decisions efficiently isn't the only thing that matters. Communicating the decision coherently so that people understand why it was made and what it means to them is a challenge, too. This step is often rushed through and taken far too lightly. If communicating what a decision really means is rushed and left up to different members in a management team who are bound to have their own takes, it can breed mistrust and a feeling that management is disconnected.

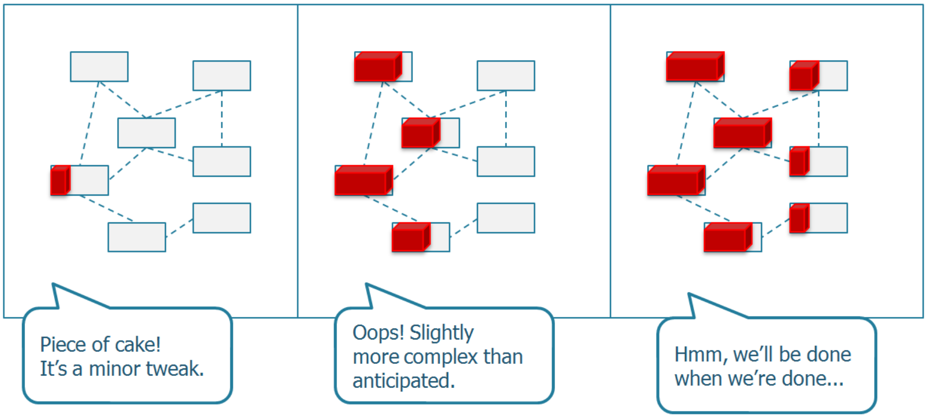
As a saving grace, however, there are a few things that can help out:

- Treating decisions in product development as tradeoffs rather than as absolute facts, and being wary of situations where you can't justify the tradeoff.
- Trusting first-hand information. The people closest to the problem often have the best view of the current situation.
- Using visual information. Having a shared view can greatly improve understanding of the current state.
- Fostering a "bad news first" environment. An environment where weak signals are taken seriously and problems get solved as a result, instead of one in which the messenger is ridiculed and shot.

Adopt Modular Design

We can do several things to build flexibility into our technology. One thing we can do to both give us a faster time to market and reduce the number of late surprises is to *modularize* our products with loosely coupled dependencies.

Modularizing products can help you simplify them by clarifying the exact purpose and function of each module. It allows you to control your dependencies instead of being controlled by them. It helps reduce the ripple effect. As you can see in the following figure, a small change in a product with unknown or hard dependencies can quickly ripple through the architecture.


Piece of cake!
It's a minor tweak.
Oops! Slightly more complex than anticipated.
Hmm, we'll be done when we're done...

Dependencies matter because they indicate risk, also known as changes that need coordination. By modularizing your product, you define a clear purpose for each module, and your dependencies become explicit.

You also have the choice of designing away the need for explicit coordination by coupling the dependency loosely. For example, if a change in Module A rarely requires a change in Module B, then the need for explicit coordination can be relaxed and downgraded to an as-needed basis. This will speed up flow.

To summarize, a modular design has the following merits:

- Knowing your dependencies up front, instead of being surprised by them
- Decentralized decision making within components
- Faster flow of customer value
- Clarity of purpose for each module
- Component reuse (preferably extracted from business needs)

This can in turn reduce the amount of coordination and planning overhead needed to manage your product development.

Cultivate an Experimental Culture

In product development, above all we need to continuously challenge the status quo. If we don't, we risk getting stuck in old processes, rigid structures, and obsolete tools simply out of familiarity and habit. We may never discover the fact that our outdated methods could be replaced by something simpler and smarter. As Pascal Van Cauwenberghe, XP expert and inventor of the

Bottleneck game, puts it, “Software companies get into trouble by consistently choosing the simple decision before the hard.”

The ability to learn is an essential competitive advantage. What does it matter if we know all about desktop products, when halfway through the project we realize that what we need is the mobile platform? If we have the right experimental culture, learning an insight late in the process should be seen as an opportunity, not a problem.

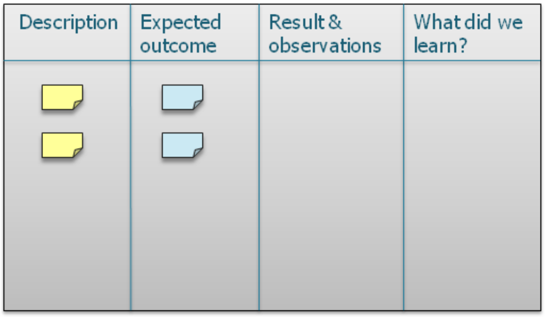
What behaviors drive a learning culture that promotes initiative? The easiest thing is to keep doing what we have always done, right? Have you measured initiatives in your organization lately?

Training management teams to improve on using experiments is one way to start unlocking system-level improvements across the full value chain. Since management decisions tend to have a significant impact on the structures in which people work (organization, processes, reporting, resource allocation, salary, and rewards), invoking management teams to challenge and improve at the system level is a natural step. So how do we train managers to learn?

The answer is by doing it! A simple way to train yourself to apply a learning behavior is by keeping a shared experiment board with your management team. Of course, the board is less important than the behaviors you seek to master:

1. Try ideas out, and try them quick (instead of promoting ignorance).
2. Talk about where you are and where you need to go (reflection).
3. Learn empirically using facts and observations.
4. Strengthen both critical thinking and curiosity.
5. Take responsibility for improvement ideas.

Let’s look at a simple example, a visual experiment board used by a management team.



An experiment board helps a team to:

- Keep a shared picture of the experiments running, kept, aborted, or sustained.
- Clarify intended outcome. When we know what we should expect up front, we are better able to understand the actions that produced unexpected results.
- Separate observations from learning.
- Discover when we have new insights—new information that was not expected. Based on the expected outcome, we can learn when something unexpected happens.

The number of experiments you can run is up to you, but make sure each experiment has an owner. No owner = no experiment.

Many of the changes coached in this book were run in a similar manner. For example, starting Kanban was itself an experiment. We tried it out for a period of time. We then asked both the team and the manager what was valuable, what to simplify, and what to retain.

You don't need to write a post-it note for everything you want to try. By training people around you to define the experiment first before jumping into action, you are setting them up to learn from the outcome, which is often more meaningful and valuable than checking off improvement actions.

Key Points to Remember

All the case studies in this book started with managers who refused to be victims of their circumstances. They wanted to make a difference. While Kanban helped them see and focus their improvement efforts, the initiatives of the people involved made these improvements happen. Freeing up space for improvement efforts couldn't have happened without changes in management behavior. Key changes included:

- *Evaluating output*. We achieved three things by shifting management's attention away from executing detailed plans and toward evaluating and improving output. First, managers focused on quality and flow efficiency instead of resource-usage efficiency. Second, and even more important, managers began to question and reflect on the effectiveness of the current way of doing things. Third, we now had transparency of the current state, which is the stepping stone to impacting the future.
- *Optimizing end-to-end*. By shifting focus from improving the parts to improving the whole, we can begin to recognize factors outside IT to improve on—for example, the front-end part of the value stream. By reducing time to market and percentage of features used, we can reduce total costs instead of the costs of individual functions.
- *Talking to people*. Have frequent conversations about where you're going and why. This helps nurture thinking and responsible people. It also builds up trust capital for your intentions, which is vital to have during times of uncertainty.
- *Leading using long-term thinking*. System enablers beyond the span of control of individual teams produce short-term agility. By making small, sustained investments over time in culture, technology, and leadership, you can spend more of your time in flow mode and grasping opportunities.

Finally, it's not the processes, the planning, or the execution that drives improvements and innovations. What it really requires is the initiative taken by the people involved. An experimental culture among the management team is crucial to help make the leap from *preserving what is*, to *looking at what is possible*.

Okay, that's it for now with theory. Let's walk through our case studies and see what really happened.