

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.

Copyright © 2015 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

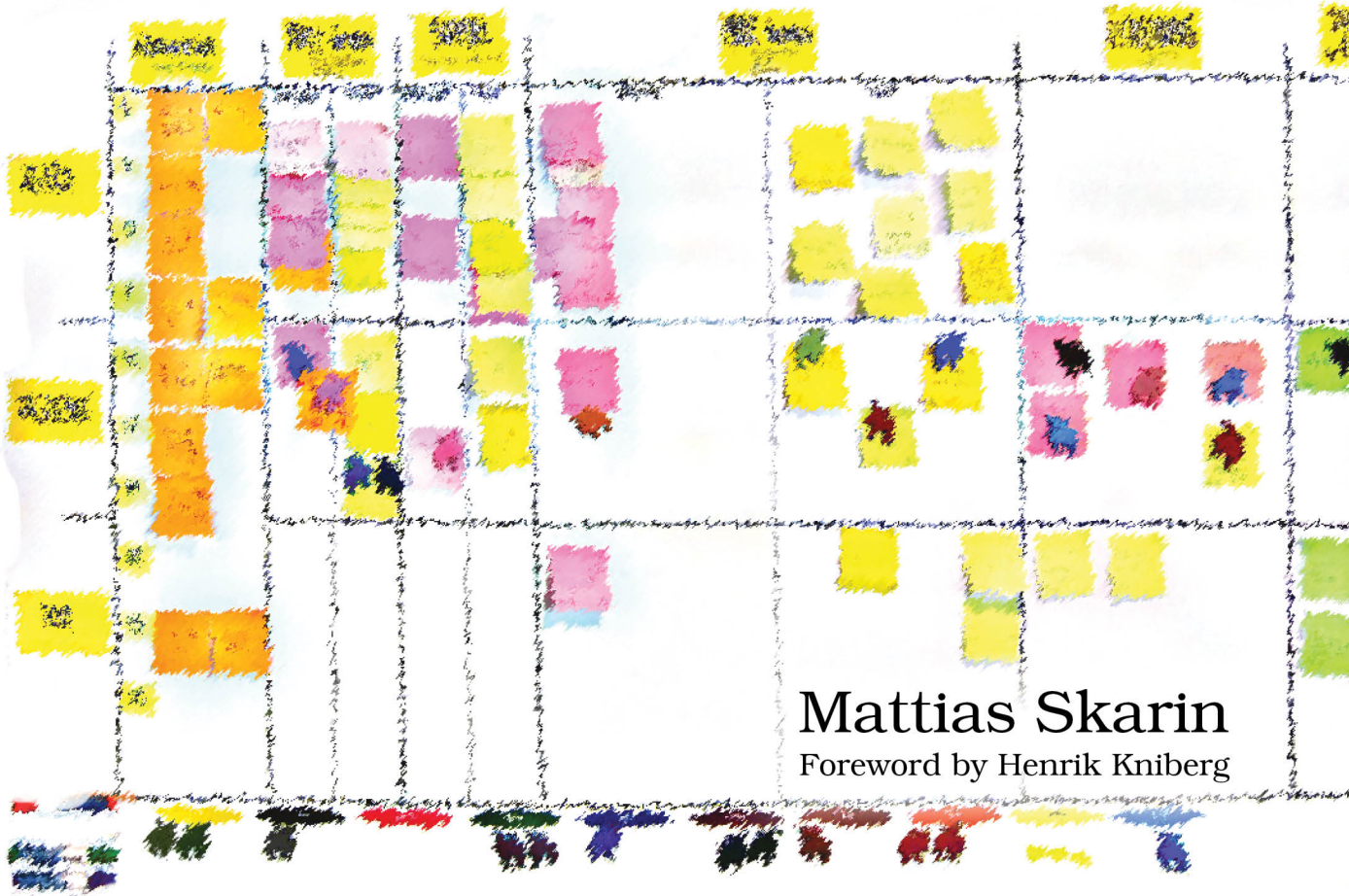
The Pragmatic Bookshelf

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



Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf, PragProg and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

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.

Copyright © 2015 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

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

Unfortunately, it's rather easy not to see problems even if they're right under our noses. In this case study, we'll look at how a company's change management team adopted Kanban. Using Kanban, managers were able to eliminate stress, build an effective team that took responsibility for the big-picture goals, and work with other teams to address behaviors that were causing recurring quality problems.

The change management team was responsible for tracking various requests from internal developers, system owners, and third-party vendors and grouping them into appropriate releases. The team was under pressure to release code more frequently and felt they didn't have time to thoroughly test beforehand. Let's see how the company improved its process and avoided burnout by introducing Kanban into change management.

The Challenge: Managing Dependencies Without Burning Out

We're still looking at Company H, the company we looked at in [*Enterprise Kanban: Improve the Full Value Chain*](#), but now the focus is on the team in charge of releases and production updates. The change management team accepts change requests from internal development teams, system owners, and external vendors (for upgrades, for example).

The team managed changes across 350+ systems, ranging from standard systems by outside vendors to custom in-house boxes. The technology stack ranged from old VMS systems to Java. Many of the production systems were parts of the country's critical infrastructure. Maintaining knowledge and dependencies across such a diverse technical stack was a huge challenge for a team of just eight people.

The team had been working under a lot of stress for a long time, and the work overload meant that they had little time to test code or perform quality assurance testing. At the same time, both internal development teams and outside vendors were requesting releases more frequently. The team was overwhelmed.

Before adopting Kanban, the team managed their work with a ticket system. Each team member got requests for a release, patch, or system change as a ticket and was responsible for making that change to test and/or production systems.

The team's manager, Birgitta, described how bad things were before Kanban. She was one of the key figures instrumental in pushing the changes through.

Why We Needed Kanban

by: Birgitta, head of change management

Q: Before you started, what did you think Kanban could help out with?

Most importantly, visualization, for everyone in the team to see what was going on. This was a big difference from before. When we were using the ticket system, we had little or no overview of what we were really working on.

Secondly, to be able to see what we were spending our resources on and to get a common forum going where we discussed that, like a standup meeting. Last but not least, we also wanted to improve flow through our team.

Q: At what point did you think, "This might work!"

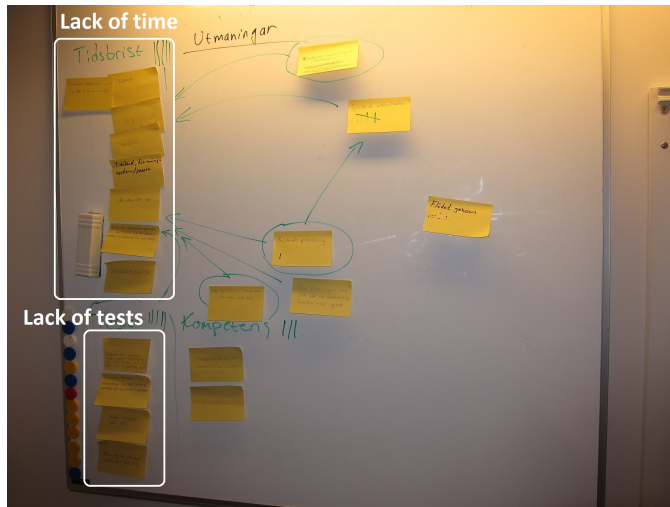
When we started, we had a very rough and chaotic board. We had tickets piled upon each other and quite a poor overview. At that point, we said, "We need to fix this," and we started to improve the structure and overview of the board. We also started asking questions about if work really was being done. It turned out that a lot of work that was piled up on the board was not actually being worked on. I remember on one occasion when we found a team member with 20 active items! We asked him, "Are you really working on 20 things?" And the reply was, "Hmm, actually, no." So we decided that the board should reflect active items only and told everyone that if something was not being worked on, it was no longer active, and we started putting the items back into the queue. That made a difference. The board got cleaned up. The team started to take responsibility for the tasks at hand, we got focus, and the stress decreased considerably.

How We Got Started

It began with two managers observing how Kanban was used in software development to visualize and track work across multiple teams. This inspired them to take a Kanban training class to see if Kanban could be helpful for them. On the train home back from the class, they sketched their first Kanban system and agreed that it would help them get an overview of the work they were handling and to work together as a team instead of being on their own.

I was invited to facilitate a kickoff workshop on Kanban for the change management team. Because we wanted to start with the challenges and get the team's perspective, we began with a small retrospective meeting. We asked each team member and manager to list the top three challenges on Post-its, which we then grouped on a whiteboard. We separated causes from effects. Finally, we asked the team members to vote on what they believed was their most pressing concern. This exercise allowed the team members to share their perspectives on the situation and to agree on what their key challenges were. We were now ready for a discussion about how to address the challenges.

This exercise allowed the team members to share their perspectives on the situation and to agree on what their key challenges were. We were now ready for a discussion on how to address them.













Two key challenges were identified at the workshop: lack of time and lack of testing during releases.

We discussed examples of how other teams use Kanban and the pros and cons of each approach. We asked the team to use simple thumb voting (see [How to thumb vote on page ?](#)) to tell us whether they would be willing to give Kanban a try. Once the team agreed, we could put up the board.

Let's look at how we set up the Kanban board for a change management team.

How Our Process Worked

Change management needed to see an overview of the current active release (in the Current Release lane) as well as upcoming change requests (in the Not Assigned lane) on the Kanban board. Each lane had its own priority: tasks higher on the board had higher priority. Notice the Follow Up section on the board—that is where services are verified in production after changes are released.

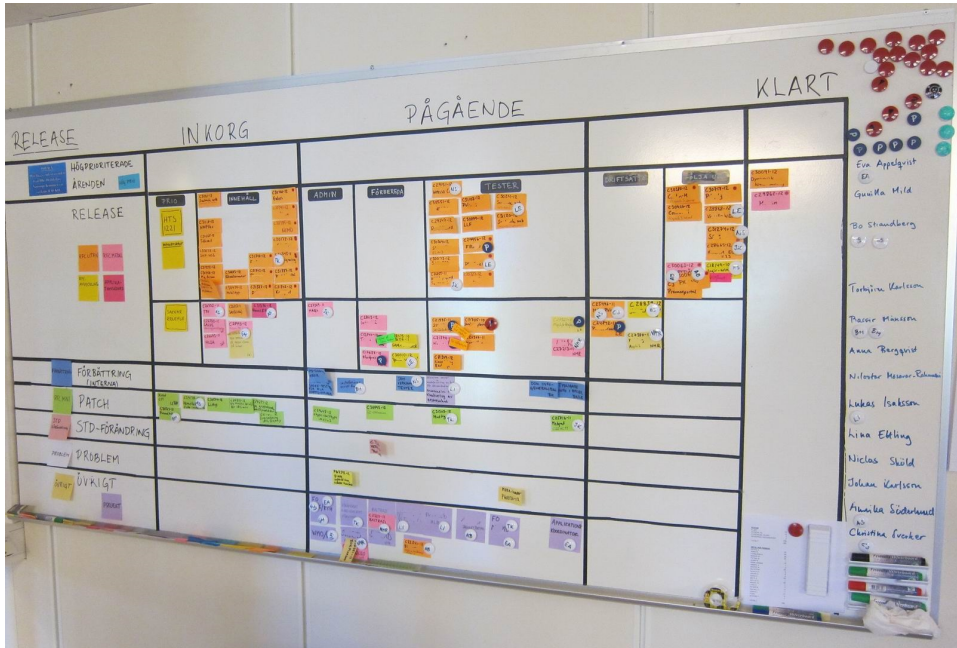
Release	In queue		Ongoing				Done	
 High prio								
Releases   Small Medium   Liquidate Coord.	Current release	Content	Admin	Prep.	In test	Put in prod	Follow up	
	Not assig.							
 Improvement								
 Patches								
 Std. change								
 Problem								
 Other								

Let's take a closer look at how work flowed through the board. Each lane represented a specific priority and demand type. Tickets in each lane had a specific color to make it easy to visualize what was happening. We could see progress on non-urgent demands, such as improvements.

Prio	Name on the Board	Description
1.	High prio	Critical production fixes
2.	Releases	Scheduled release updates to production. Runs on a four-week interval. Colored categories of release requests are: <ul style="list-style-type: none"> • Small - small changes • Medium - medium changes • Liquidate - remove from production • Coordinate - application change requiring explicit coordination
3.	Improvements	Improvements run and owned by the team
4.	Patches	Smaller changes to production, outside the release window
5.	Standard change	Standard change, impacting a single application
6.	Problem	Problems requiring investigations

7. Other Support to projects owned outside change management but which change management supports

Obviously, the board didn't start out looking this neat and pristine. The first version was so filled with tasks you couldn't actually see the board. The first version of the Kanban board also had a very different layout.



During the first week of using Kanban, we had trouble fitting all the work assigned to the team on the board. There simply wasn't enough space! To deal with this, the managers worked their way through the pile by asking "Is this active?" and then they would either remove or defer the item by putting it back in the prioritized queue or get it done. We quickly learned that the team was carrying a lot of passive work, which was stealing energy and focus.

Focus Your Board

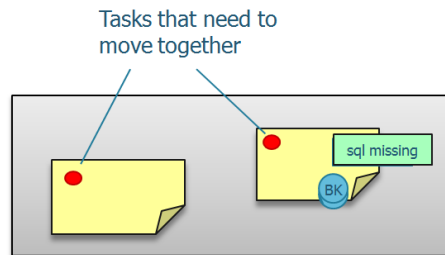


In the beginning, we envisioned that we would display up to four upcoming releases on the board, but we quickly found that was impossible. The overview was simply lost. We switched to displaying two releases (current and next), and this worked much better. Narrow your board's focus.

The board felt messy even after a month, so we cut the Post-it notes attached to the board in half. This reduced the size of the tasks and made it easier to see the overview. On each Post-it, we tracked the name of the change and the

ID number of the corresponding task in the ticket system. This let us identify and track tasks while keeping a neat board.

We also added colored magnets to show which tasks were blocked and to show *tags* explaining why the task was blocked. Dots showed which changes had to be coordinated with the upcoming release and which changes could be released individually or at a later time.



That made a huge difference. The work was largely distributed across different team members. The lack of an overview meant that no one could really make a snap decision about whether to move a change out to production without explicitly coordinating with the rest of the team. Now they could.

How You Know Your Kanban Board Works

When you look at a Kanban board, ask the following four questions to see whether the board is working:

- **Overview:** Can I get an overview at a glance? Can people working with the board see what to focus on, what is prioritized, and whether any items are blocked? This helps team members to see what to focus on and managers to see where to invest improvement efforts.
- **Transparency to progress:** Does the board help people track progress? By enabling people who depend on the team to see progress, you allow them to coordinate work proactively on their own, without having to bother the teams for status reports.
- **Meaning:** Can the team describe the meaning of the visual indicators on the board? The point is, if there is no behavior associated with the visual indicators, then they have no meaning.
- **Usefulness:** Are there conversations happening in front of the board? If the answer is yes, the board works well as a catalyst for insight and decisions.



Let's look at how we engaged with the board.

Working with the Board

Each team member was responsible for his or her day-to-day list of tasks, but team members still needed to work as a team on the combined workload. To balance this, change management had two standup meetings a week where they shifted from what they needed to do as individuals to what they needed to complete as a team.

The teams ran standup meetings in front of the board on Tuesdays and Thursdays. The meetings were short—a maximum of 15 minutes—and they were run by the team’s managers. The agenda was to walk the board, reviewing whether something needed attention. At this meeting, the team also solved problems and clarified who was responsible for following up on blocking issues. Finally, the team reviewed the Prioritized queue (which was kept to the left of the Kanban board) so that the team had an idea of what was coming up and what was important.

The management team prioritized and pulled requests from the ticket system and added them to the Prioritized queue. This helped the team keep track of work in progress even if there was an increase in the workload. Smaller requests coming directly to the team members were reviewed during the twice-weekly standups.

It’s worth noting that when prioritizing, the teams reserved the right not to take on a task. In which case, the stakeholder would be given the option either to put the task back into the queue for later or to solve the issue himself or herself. We tracked the number of *won’t do* requests over time.

Expect Your Kanban Board to Evolve



It is natural for a Kanban board to evolve many times during the first two months of operation. Items on the board such as WIP limits, lanes, and work states will inevitably change. This is natural. In the beginning, try to update the layout once a week. The goal is to maintain a clear overview for all to see.

Because this team acted as the company’s second-line support, answering application-specific questions and troubleshooting were part of the daily workload. Soon after we introduced Kanban, we experimented with having just one person field support questions during the week and freeing up the rest of the team to focus on other assignments. We called this role “Today’s Backo” (backoffice) and rotated the assignment among all the members.

We struggled with this policy at first because people tended to approach the person they were used to working with instead of the week’s designated Backo.

To solve this problem, the team suggested posting the name of the Backo on the Kanban board. This way, anyone looking at the board would know instantly who to talk to. We added a separate column on the far right of the board to list the contact person's information. This was tremendously helpful in getting people to go to the correct support person.

There was another challenge: the uneven distribution of skills. Some people did not have the experience necessary to be able to handle some of the more frequently occurring requests. The Backo person had to try to find a solution to the problem first before getting help from the rest of the team. It was important that the team member helping the Backo did not directly solve the problem, but instead guided the Backo toward finding the correct answer. This made the support issue a learning exercise.

Handling Common Issues



It's a good idea to maintain a FAQ on a wiki for common issues. After a couple of turns, the support person should be able to answer questions without pulling in help. You don't need cross-functional skills on all matters, just on frequently recurring support demands.

Keeping the ticket tracking system and the physical Kanban board in sync was a challenge. The physical board was tremendously valuable because it gave the team an overview of what everyone was working on. Team members working on their own also felt connected to the rest of the team. The board triggered conversations and helped members share responsibility for the workload. However, we needed a way to document changes in the production environment and to keep track of all the details.

The ticket system turned out to do a pretty good job, and the team was able to use it as a documentation tool rather than as a communication tool. We learned to keep both the system and the board in sync by having the team update the issue tracker before moving tickets on the board.

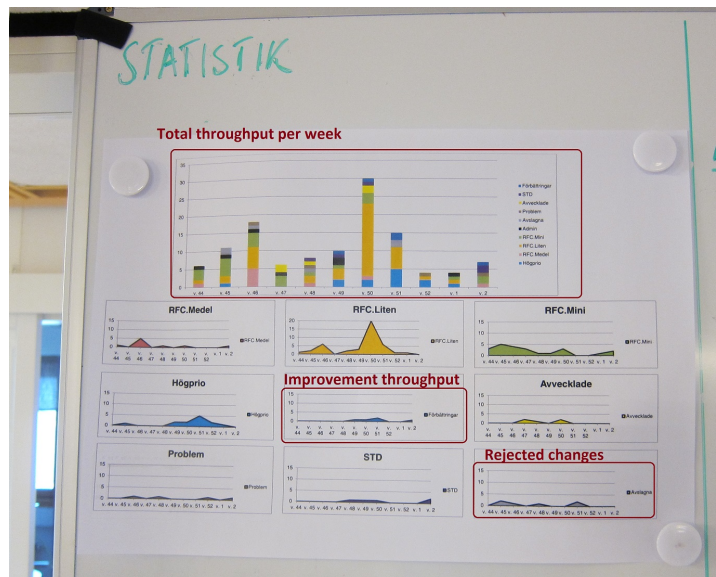
How We Continuously Improved

After each release, the team held a small retrospective meeting to review what worked well and what didn't. We also updated the Improvements section on the Kanban board with ideas that came out of the retrospective. At the same time, the managers gathered data from the board and updated measurements for everyone—the team, the managers, and me (as the coach)—to review.

We collected three types of flow data: total throughput per release, throughput per work type (lane), and lead time. Collecting this data over time allowed us to spot trends as well as variations in these values. We posted the statistics on the left of the Kanban board, above the In Queue section, all to see.

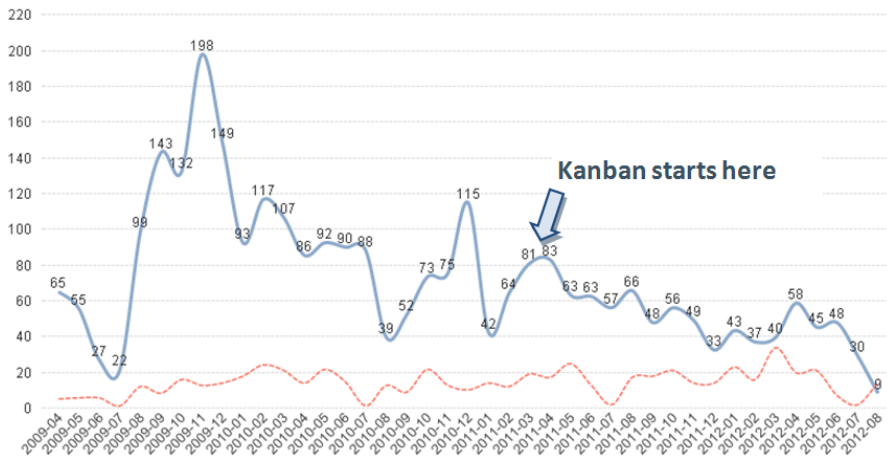


The bars in the top chart in the next image represent total throughput. Note the large variation over time, correlating with the end of season. The highlighted chart on the bottom right refers to rejected changes. Not all change requests were well prepared, so some were rejected from the workload.



The highlighted chart in the middle reflects the team’s improvement throughput (tickets designated as improvement initiatives). See how the team improvements moved from zero to a small but steady trickle? This showed us we had a healthy balance between short- and long-term activities.

Let’s take a closer look at the metrics we gathered. The flow statistics were very helpful in validating the effect of changes over time. The Kanban board should be treated as a snapshot in time, and the flow metrics as a way to see seasonal effects and trends. The graph shows lead time for the change management team.



The fixed blue line denotes lead time, and the red dotted line denotes the number of tickets the team received. The time to resolve a ticket dropped from 60 days down to roughly 30 days once the team was up and running with Kanban. Seasonal effects may have influenced the results, and we need more data before we can completely rule seasonal effects out. However, the drop in lead time does correlate with the observations from the Kanban board, along with the decrease in the team’s stress level.

Now we’ve seen the metrics, let’s take a look at the lessons the team learned.