Scrum Your Jira!

Your Waterfall Organization Transformed into Multidisciplinary Teams

10 essays on using online tools to improve an offline technique

SCRUM YOUR JIRA!

Your Waterfall Organization Transformed into Multidisciplinary Teams

10 essays on using online tools to improve an offline technique

Published by Clemens Lode Verlag e.K., Düsseldorf

SCRUM YOUR JIRA!

Your Waterfall Organization Transformed into Multidisciplinary Teams

10 Essays on Using Online Tools to Improve an Offline Technique

CLEMENS LODE

2019, Second Edition

ISBN 978-3-945586-70-9

Edited by: *Conna Craig*
Cover design: *Jessica Keatting Graphic Design*
Image sources: *Shutterstock*, *iStockphoto*, *pexels.com*
Icons made by http://www.freepik.com from
http://www.flaticon.com is licensed by CC 3.0 BY
(http://creativecommons.org/licenses/by/3.0/)

Printed on acid-free, unbleached paper.
Subscribe to our newsletter. Simply write to newsletter@lode.de or visit our website.

MANAGEMENT
PSYCHOLOGY

Introduction

神の一手—*Kami no Itte*—Japanese, roughly meaning "move of God" or "godly move"—describes an entirely new insight concerning a move during the game called "Go." Such a move is a goal taught to students to be more attentive toward less obvious maneuvers, leading the students to focus on alternatives. Likewise in management, first put aside the conspicuous answers (like adding more people to an already late project) to make way for an objective mind and attention to alternatives. In management, the challenge is to discover the potential of the team and your organization and to build upon that.

This book shows some of the 神の一手—*Kami no Itte*—of management, focused on the Scrum method and the software tool Jira. *Scrum Your Jira* reminds the Scrum Master about the original idea of Agile principles. It teaches how she can implement those in practice, carry on the Agile idea, and become an example for others.

Contents

Publisher's Note

Thank you for keeping up the tradition of reading books. You and your fellow readers have created a market for this book. I hope that I can meet your expectations and I am looking forward to feedback from your side, no matter whether it is positive or negative. To send general feedback, mention the book title in the subject of your message and simply send it to feedback@lode.de. You can also contact us at https://www.lode.de/contact if you are having a problem with any aspect of the book, and we will do our best to address it. Also, we cordially invite you to join our network at https://www.lode.de.

Although I have taken every care to ensure the accuracy of the content, mistakes do happen. If you find an error in one of my books, I would be grateful if you would report this to me. By doing so, you can help me to improve subsequent versions of this book and maybe save future readers from frustration. If you find any errata, please report them by visiting https://www.lode.de/errata, selecting the book title, and entering the details. Once verified, the errata will be uploaded to our website. You will, of course, be credited if you wish.

Best regards,

Clemens Lode
CEO, Lode Publishing
Düsseldorf, Germany, September 1st, 2019

Preface

> “Nurture your mind with great thoughts; to believe in the heroic makes heroes.
>
> —Benjamin Disraeli

The decision to write this book began with an insight that Scrum is much more than a set of meetings and boards. But then again, that *is* the idea of what Scrum is: lean project management, without all the documents. It is also a buzzword, with companies wanting to copy the success of other organizations that have fully (and successfully) implemented Agile principles.

I guess I have to admit that to truly understand something, I need to "build" it one way or another. Despite the subject of my previous book, *Philosophy for Heroes: Knowledge*, inter-personal and management ideas come more naturally to me than abstract philosophical principles. So, instead of writing down everything I have learned about Scrum over the course of several years, I wrote these articles over the course of one week each—in a truly Agile way (followed by some editing to give it a coherent message). It is up to you to decide if this experiment was a success!

This book is about Scrum and Jira, the most popular management technique and the most popular ticketing system. It is written in a light-hearted tone, similar to how you might chat with a fellow consultant about your experiences implementing Scrum or Jira. The main message is that there is more to Scrum and Jira than meet the eye because Scrum is more than a series of meetings.

My hope is that you will take with you one or two interesting thoughts from this book, and develop them further. Personally, I would like to place this book in the hands of a younger version of myself, someone who finds himself at the beginning of his professional career, ready to start new projects. Even if I reach only a small handful of people who will take to heart a few of these core ideas and set out to lead projects better, I will be able to enjoy the rewards of this book.

Do you need help with the Agile process in your company? Subscribe to *our newsletter at https://www.lode.de/agile or contact us for coaching at coach@lode.de.*

Best regards,

Clemens Lode
Author, *Scrum Your Jira!*
Düsseldorf, Germany, August 1st, 2019

An Introduction... to Rugby

“ We scrum for possession, run for the try zone, bleed for the team, and live for the game.

The origin of Scrum is in sports, namely rugby. Unfortunately, connecting the name (“Scrum”) to the sport is usually where knowledge about the roots of Scrum ends. But I think it is worthwhile to take a look at how teams win games in sports. Other than the name, you will discover many similarities between the development process and Scrum. Also, as a Scrum Master, becoming aware of this background will provide you with a great analogy to explain the Scrum rituals to your team.

Basically, rugby is similar to European soccer and American football: two teams trying to get the ball to the other side. Figure 1 shows several differences that make each of the three games unique in their playing style:

	Rugby	**Football**	**Soccer**
Goal:	touchdown	touchdown	goal
Pass forward:	no	partial	mostly
Block:	ball owner	anyone	passive
Moves:	continuous	interruptions	continuous

Figure 1: Comparison of different sport games

While there a number of other differences (protection, penalties, etc.), the properties listed in the table lead to significantly different playing styles for each game.

- In **rugby**, with the objective of having to be near the goal (touchdown), there is little danger from scoring a lucky win from afar. Also, no forward passing is allowed; the team has to actually run together to the other side, creating the need for a cohesive team with quickly changing roles.
- **Soccer** mitigates that by having a separate goalkeeper role. In terms of field movement, soccer motivates the team to spread

out. Fast-moving forward passes, together with the lack of the ability to physically block the other teams' players, are the main causes for this.

- In **football**, forward passes are allowed, but so is blocking—making the field of operation much smaller. Football is slower than rugby or soccer, given regular interruptions between moves. This way, it becomes more strategic. Also, this gives room for a lot more communication between the players and the coach. The team might have to be less independent than in rugby or soccer.

This comparison gives you several hints about why rugby, and not soccer or football, was used as the blueprint for Scrum. It goes far beyond giving it a cool name. In Scrum, while you have your specialists, they are all on the field. There is less communication with management (coaches in football), and the product owner is part of the team. The rugby team does not have fixed specialized roles, like a goalkeeper or a quarterback. Depending on your software team's cultural background, teach them the differences between football in the US and soccer in Europe. Watch a few rugby matches together (team building!) and discuss how this applies to the situation in your company. It is an excellent way of opening people up. By seeing it "in action," your team can more easily understand the abstract and product development-related terms and concepts. In fact, I encourage you to watch a few minutes of rugby right now to get into the mood.

Back? OK! Now imagine the rugby players using Jira instead of coordinating everything on the field. An impossibility! That is where this book comes in, tackling the problems related to using Scrum with Jira, and opening your eyes about how to examine the use of Jira in your organization.

Now get ready for the kick-off!

Chapter 1

Implementing Agile Techniques

> Our Scrum Is Special!

> “ The Agile movement provides alternatives to traditional project management. Agile approaches help teams respond to unpredictability with incremental, iterative work cadences and empirical feedback. Agilists propose alternatives to Waterfall, or traditional sequential development.
>
> —*The Agile Movement* (edited)[1]

> “ Scrum is an Agile software development model based on multiple small teams working in an intensive and interdependent manner. The term is named for the scrum (or scrummage) formation in rugby, which is used to restart the game after an event that causes play to stop, such as an infringement.
>
> —*What is Scrum?*[2]

When clients ask me to help with implementation of Agile techniques by using Scrum, my first question is: “What do you mean by ‘Scrum’?” Usually, I then hear that the company has its own special version of Scrum (or other Agile technique) because, according to the people with whom I am meeting, their company is a special case.

First, yes, your company is a special case. Each company is unique and in a particular market niche. Second, if you followed the evolutionary approach of improvements in small increments, you have adapted your process to the environment of the company. No two companies are alike. Hence your need for an external consultant to examine the conditions in your company.

[1] AgileMethodology, 2008.

[2] TechTarget, 2007.

The reality I see all too often is that a company hires a Scrum Master who merely acts as a supporting firefighter, accompanying the former project manager (now "product owner") and running around the company putting out fires. This kind of extra resource is justified to upper management by pointing to "Agile" and its use in other companies...

Introducing (and running) Scrum means that you want to change your company according to proven methods. You cannot have your cake (Scrum) and eat it, too (changing Scrum to suit your company)—you cannot improve your company by adapting the Scrum process to your company. Obviously, you cannot change your whole company in one day—it might take years! But the *goal* should be clear: to take all the steps necessary to become truly Agile instead of finding rationalizations about why your company is a special case.

> **Waterfall** · *Waterfall* is a project management method where a product moves through several phases before a final version is finished for release. Compared to Agile, the problem with this method is that it requires additional communication channels between phases. Also, the time until a team or company gets feedback from a customer is generally much longer.

> **Scrum** · *Scrum* is a set of management tools that focuses a project back on the team level and uncovers internal and external impediments of the production process. By reducing communication paths through small, multidisciplinary teams, as well as frequent releases to the customer for review, the probability for project success can be improved even if the scope is not clear from the start. In addition, work is divided into units of fixed lengths (sprints), which helps to plan future sprints with your team working at a sustainable speed.

> **Sprint** · A *sprint* is a timespan of one to four weeks within which a selection of stories should be finished by the team. Given the fact that the whole team spends 10 percent of the time (depending on the sprint length) planning and reviewing each sprint, the

goal is to reach 100 percent completion of all stories while meeting the project's quality standards and without overtime. Like a marathon runner needs to carefully plan her energy, planning a sprint requires excellent estimation skills by the teams.

Scrum Master · The *Scrum Master* controls the Scrum process. Besides proactively identifying and removing impediments to the process, the Scrum Master also supports the team in meetings as a moderator and individually in personal talks. The Scrum Master also stands up against outside influence on the process, ideally by propagating the Agile idea throughout the organizations and by explaining why certain restrictions are necessary for the overall project success.

Product owner · The *product owner* is part of the Scrum team and represents the stakeholders. The main task is stakeholder management as well as having a deep understanding of what the project is about and being able to make decisions. A product owner fills and prioritizes the backlog, keeping the complexity estimations of the team in mind. The product owner should have full authority and the final say about the prioritization of the backlog. During the sprint, the product owner answers questions from the team about the scope of the project, as well as gives feedback about finished (but not necessarily done!) tasks, but otherwise does not interfere in how the team manages its work.

Looking at the actual causes of problems

One of the techniques used in project management is to find the cause of an issue. Digging deeper, my next set of questions to the client usually focuses on the greater picture or vision of the company. Instead of telling me about their mission, they typically respond that they want to "test" Agile and then implement it in other parts of the company. Besides noting the misunderstanding of Agile as the new (local!) management technique, questions arise: What would success look like? What would failure look like? What are the concrete, measurable business objectives of the project of introducing Agile?

I am convinced that introducing Agile itself should be managed with modern project management techniques, PMBOK being my favorite. Managing Agile goes far beyond the scope of this chapter, but you certainly must have an idea about where you are going with it and what you want to achieve.

> **PMBOK®** · *PMBOK* stands for *Project Management Body of Knowledge* and describes a generic system of workflows within a project. While it is mainly applied to Waterfall projects, many of its parts can also be used in an Agile project, like defining how the team communicates with the outside world, defining the vision and scope of the project, or defining why one would want to use Scrum at all. (PMI, 2013)

To illustrate this further, I recommend reading Ayn Rand's introduction to philosophy that looks at the example of an astronaut stranded on a planet:

> Suppose that you are an astronaut whose spaceship loses control and crashes on an unknown planet. When you regain consciousness and find that you are not badly hurt, the first three questions on your mind would be: Where am I? How can I find out? What should I do?
>
> You see unfamiliar vegetation outside, and there is air to breathe; the sunlight seems paler than you remember it and colder. You turn to look at the sky, but stop. You are struck by a sudden feeling: if you don't look, you won't have to know that you are, perhaps, too far from Earth and no return is possible. So long as you don't know it, you are free to believe what you wish—and you experience a foggy, pleas-

> ant, but somehow guilty, kind of hope.
>
> You turn to your instruments: they may be damaged, you don't know how seriously. But you stop, struck by a sudden fear: how can you trust these instruments? How can you be sure that they won't mislead you? How can you know whether they will work in a different world? You turn away from the instruments.
>
> Now you begin to wonder why you have no desire to do anything. It seems so much safer just to wait for something to turn up somehow; it is better, you tell yourself, not to rock the spaceship. Far in the distance, you see some sort of living creatures approaching; you don't know whether they are human, but they walk on two feet. They, you decide, will tell you what to do.
>
> You are never heard from again.
>
> This is fantasy, you say? You would not act like that and no astronaut ever would? Perhaps not. But this is the way most men live their lives, here, on Earth.

—Ayn Rand, *Address to the Graduating Class of the United States Military Academy at West Point New York* (adapted)[3]

In terms of a company, your immediate goal is of course to survive the next month. But then, you have to establish where you are on the map. You have to open your eyes, look at the sky, and check your instruments.

In terms of Agile, I recommend to my clients that they run it like a project. We know what works from hundreds of studies, and we can create a list of items that are implemented in Scrum. In that list, we simply mark the current state of the process. Often, even in non-Agile companies, some processes have already been implemented because the people who are managing projects notice which processes

[3] Rand, 1974.

work. A simple approach is to check again the *Principles Behind the Agile Manifesto*[4]:

- Welcome changing requirements.
- Trust and motivate individuals on your team and other teams in the company.
- Developers and non-developers (e.g., marketers, salespeople) must work together daily.
- Face-to-face is the most efficient and effective method of getting things done.
- Progress is measured in terms of working software.
- The entire team must promote sustainable development, they should be able to maintain a constant pace indefinitely.
- The entire team must work together to continuously improve technical excellence and to enhance agility.
- Keep in mind that simplicity is valuable; simplicity is the art of maximizing the amount of work not done.
- The best solutions emerge from self-organizing teams.
- Effective teams regularly reflect on how to become more effective.
- An Agile company satisfies customers through early, frequent, and continuous delivery.

Together with the client, for each point, I detail how this is implemented in the company—this is the first step of documenting the current process. If I am not able explain how it is implemented or if we find that it is not implemented at all, I focus on those points and try to find explanations for each: why the company is not able or not willing to fulfill this part of the Agile process. And I do not just ask, “Why?” I ask, “Why why why why why…?” until I find

[4] Beck, 2001.

out the actual reasons something has not been implemented.[5] *And at this point, the real work starts: addressing those issues that hinder the Agile process on a daily basis.*

[5] Ohno, 2006.

Chapter 2

Multidisciplinary Teams in Scrum

> “
>
> - The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
> - Business people and developers must work together daily throughout the project.
> - Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
> - The best architectures, requirements, and designs emerge from self-organizing teams.
>
> —*Principles Behind the Agile Manifesto*[1]

Working in multidisciplinary teams is one of the main success criteria in Scrum. It is crucial to have different elements of the project —including the user or customer—represented on one team. In reality, in my work with clients, I often see that their move to Agile consisted only of changing the management style of a team, and in most cases, this applied only to the software developer team. It is no surprise, then, that I often hear a description of the situation that goes something like this: “OK, we have this software development team here, and we want to test out Scrum. Now, we are looking for a Scrum Master. Oh, yes, and we have worked with Scrum for several years, but it is not working. Maybe you can help?”

My initial questions to a client revolve around what is not working and whether Scrum was set up as a project (see Chapter 1). I then focus on one of the points listed in the Agile Manifesto, namely multidisciplinary teams. What is a multidisciplinary team? Does creating such a team mean that you have to mix Java and Android program-

[1] Beck, 2001.

mers? Or that people on the team should have some PHP experience if a website is involved somewhere in the project?

Actually, multidisciplinary teams are one of the core principles of Agile, as opposed to Waterfall. To understand the benefits of multidisciplinary teams, let us first take a closer look at what the Waterfall method (which does not incorporate this approach) looks like.

The first essential step in setting up a company is assembling the right group of employees. In a company following the Waterfall method, ads for jobs are written for and specialists are hired to do specific tasks, like programming, marketing, sales, or editing. Each person works in his or her specialized field with maximum efficiency.

Initially, the specialists form a common team, a handful of people working together on one project. With the growth of the company, new people are hired, and each specialist will lead individual teams; the former specialists become managers, and each team gets its own room, its own planning, its own resource allocation, etc.

Depending on the environment (startup vs. large, existing company), the course of a company can vary widely, but essentially, in companies that have not adopted Agile, you will usually find specialized departments each performing a task with maximum efficiency. Sounds great, right? You have the best people working in specialized departments highly attuned to their tasks. What could go wrong?

What could go wrong is that this kind of departmentalization will most likely lead to the use of the Waterfall management method—at any given time, a department works on one phase of the product. First, it goes through design, then development, editing, quality assurance, marketing, and finally publishing and sales, with the managers busy with tons of meetings, cross-department communication, and resource balancing. With multiple teams, it is nearly impossible to have one team ready to start the next phase just as another team finishes. Ultimately, this leads to long release cycles —sometimes years-long.

Then, people hear about Scrum. And what do they do? They take individual teams, set up a product owner and Scrum Master for each team, set up a ticketing system, sprints, and more meetings—with the managers still busy communicating and negotiating resources between the teams. The result? Marginal effects, somewhat more disciplined development teams (mostly because of more investment into the build systems to allow more frequent releases), and a story to tell all the developers that now, with Scrum, everything will be better and will continue to improve over time.

I hate to break it to you, but this is Scrum in name only. Specialized teams cannot deliver anything but concepts or work packages. Suddenly, the users are no longer the customers—instead, the "customers" become the next team. Eventually, only one department (sales) ever faces an actual customer who will use the product.

Now, moving from the phase-based management of Waterfall to Agile, *how can companies benefit from having multidisciplinary teams?*

Break with the idea of departments and focus on products (or features). Put together all the people—including marketing, design, and sales—who will deliver the final project. Does that mean that the salespeople have to learn Java and the tech guys have to do sales? Well, yes, to some extent. At least they have to learn how to deliver a product together and help each other out. Ideally, this would have begun during the hiring process, recruiting people who are good in one area, and also know a little bit about all the other aspects of product development. If you have only specialists, have them work in pairs (two people, one computer) on features, allowing easy knowledge transfer between departments. Also, even in early development, there is a lot to do for everyone. The salesperson might not develop a new API, but he or she can help with testing, meeting with clients, preparing presentations, advocating the team's ideas within the company, creating beta- and sales channels, setting up newsletters and landing pages—all tasks marketing and sales could start doing early on, in addition to learning how the product works from the inside.

For an established company, this is a long road. But your competitor has already started doing it, and your only advantage is that you (still) have more resources than the startup next door. When will your organization improve and become genuinely Agile?

When it comes to multidisciplinary teams, how is your company doing? How are the teams in your company structured? Does that structure support information exchange between departments or specialization of individuals? How are HR decisions integrated into your Agile strategy?

Chapter 3

Scrum and Jira: A Love-Hate Relationship

Within a company, Scrum is usually initiated in software teams, because they are the teams who have to deal with the biggest insecurities in terms of planning. Each software project is an entirely new project, even if it is "just" a new version. The software market changes rapidly, hence Agile methods are the management method of choice.

There is a glitch in the system, though: hiring people based on their computer skills often comes at a price, namely interpersonal communication. The HR department of a company should take great care to not just hire the best individual coders, but instead, people who can communicate effectively and have high emotional intelligence.[1]

> **Jira** · The on-premise or cloud software *Jira* by Atlassian is one of the leading ticketing systems available. Beyond a mere ToDo list, it provides administration functionality for projects, Scrum and Kanban boards, custom workflows, custom screens, user rights management, plugins, and third-party integration. The name itself stems from Bugzilla, the software Atlassian used initially for bug tracking. They began calling it by the Japanese name for Godzilla, "Gojira." When they later developed their own bug tracker, they just dropped the Go—hence Jira! (see https://confluence.atlassian.com/pages/viewpage.action?pageId=223219957)

We are a culture that loves technology—sometimes to the exclusion of working with people. As a result, our communication tools tend to be complicated and directed at an audience of software engineers. This also applies to Jira, which is often used to manage the Agile process. But is that really a good decision?

The gold standard of Scrum is face-to-face communication. To what extent is your team practicing this? I recommend taking another look at the *Principles Behind the Agile Manifesto*[2], *What Google*

[1] Duhigg, 2016.

[2] Beck, 2001.

Learned from its Quest to Build the Perfect Team[3], and Chapter 1 on the subject. Then, as a first step, evaluate where you are in the process of becoming an Agile company. Do you think your Jira installation helps or hinders your progress?

In the previous chapter, I wrote about the importance of multidisciplinary teams. Tools highly optimized for use with software developers might cause problems when other departments are expected to use them, or when creating a team with a mix of people with different specializations. A company has to be careful not to focus on what is most effective for only part of the company but should instead look at the company (or a given product) as a whole. How well a company is doing in that regard is visible by a quick look into its Jira user directory: if you see only developers having logged in recently, the system is more an obstacle than a useful tool for communication between the teams.

I doubt you will ditch Jira just because of this book, though. I rely heavily on Jira. It is likely that you are reading this book because you are already using Jira, and you will probably not dismantle Jira any time soon! So, with the system already in place and the information already managed within Jira, how can we make it visible to the rest of the company?

In that regard, Jira is a step back in terms of Agile, as it disconnects people and makes things overcomplicated. What we need to do is recognize the drawbacks of Jira and examine ways around them.

For example, when creating new stories, Jira gives you the option to write a summary and a description. This immediately leads people to come up with a descriptive name as the "summary," and enter the actual user story "As a user, ..." in the description (if at all).

[3] Duhigg, 2016.

My proposal is that the description field should be used only for the *acceptance criteria* and the *definition of done* (see Chapter **??**). Descriptions should *not* be needed or even seen by anyone outside the Scrum team. Hence, **put the user story into the summary!** Your backlog board will look much more structured if all the stories follow a similar naming scheme. Nobody outside the team will be in a rush to look at your board if it is filled with technical jargon. And if we take a step back, that is actually what Scrum teaches: Create a board with stickers describing user stories! Why not do exactly that? Having less (e.g., just a sticker) sometimes is better than having 50 customizable fields. Hence, it makes sense to fill a product backlog with tasks to resemble a physical board with stickers.

Another example is epics. These are used differently from company to company; some call it a "bigger story," some call it a "feature," for others it is a "project." If you are using Jira, you first have to focus on defining an "epic," then on how it is displayed. How would you implement epics when using nothing but paper stickers?

In Jira, epics are essentially "super-stories." Why? Because Jira offers you a progress bar for each epic. (As this really reminds me of a pre-planned Waterfall project, I find this feature useless.) Much more interesting is the rather simple feature of color. Assigning epics to stories quickly gives you an idea what the story is about, as each epic shows up as a colored banner on the board. With epics, the Jira board comes alive: you can quickly and easily make visual sense of the entire project—where you are at, what is left to do, and who is going to do it. For example, in the past, I often used "Frontend," "Backend," and "IT" as three main epics when working on a pure server project (in a non-Agile business environment). On the board, you immediately see what type of stories there are. Of course once the Scrum process is fully adopted, you should utilize user-facing features (as opposed to system components) as epics.

Here, I will leave you with a final point. It is a small issue, yet one that annoys the perfectionist in me whenever I start looking at a Jira board of a new client: the "priority" field. In Scrum, there is no "priority" field. First, within a sprint, it is up to the team which tasks to work on and in what order. If you, as a product owner, want to direct the exact sequence of which tasks are built, that is eXtreme Programming (XP), a topic for later. Second, even if you take priority into account, who determines the priority? Certainly not the reporter, who might have no idea about the business value or the time needed to implement it, yet who is prompted, by Jira, to fill out the form (better to leave it empty than fill in a meaningless bit of information that will just cause more work for the one reading it). And the product owner already sorts the backlog according to priority, based on business value and estimation. The usual result is that in the Jira backlog, you end up with two types of priorities—critical and urgent—because every reporter thinks his task is kind of important. The priority even shows up as little red arrows—completely unnecessary and confusing.

> **BACKLOG** · The *backlog* of a project is a list of stories prioritized by the product owner according to the business value of each (estimated by the stakeholders and product owner) and complexity (estimated by the team). Keeping a clean backlog is key to success: it is not an idea graveyard! You can move all those nice-to-have stories to a separate list.

Unfortunately, Jira does not allow you to deactivate priorities directly. There is a little trick, though: In the project settings, in the priority scheme, you can create a new default priority and upload an empty transparent PNG file as the corresponding icon. This solves the problem and the board looks a little more like a real Scrum board!

How does your team use Jira in your Scrum process? What are your remedies to Jira's drawbacks? Do you ever simply leave Jira aside and use a pen and paper?