Peter Schmidt

Experience:

2 yrs experience w/ Scrum, 1 yr in 1 project

Sprint: 30d ays

Meeting: daily scrum (not time-boxed, 30-45 min), review (T, PO, SM, sometimes stakeholder).

Before, they had two hours each two weeks, and never finished. They realized they need a lot more communication 🡪 daily.

Initially, the invited lots of people in the review. But people were expecting big demo, and were disappointed, they had to argue, etc. They stopped inviting upper manager to have simplified review.

Retrospective: SM, T, PO.

Not sure whether pure Scrum fit the project, lots of things where Scrum is too tight

Team: 3 softw dev, product owner + assistant, external scrum master, stakeholders

In parallel, there is the team that build the machine, with a project manager, and very waterfall. The software part is just a part of the whole product. Hardware process would not be too happy with Scrum

Problem to match the waterfall of hardware, w/ scrum of the software.

Project leader has influence on the product backlog. PL talks to PO and priorizes the tasks.

Improvement over the yr:

* they stopped inviting stakeholder to all meeting (see above)
* removed the burndown chart. Effort to keep the burndown chart was too high. Nobody was really interested in it.

If PO is happy with the outcoming at the end of the Spring, the daily burndown is not necessary. Burndown was on the wall, manually. That’s an effort. If things were done automatically, would maybe be interesting. But people need to recaculate the hour they need for a task, etc.

Now that they introduced Scrum, the bottleneck shift to somewhere else. Other need to adapt their process. Why should I change my process if you change your process. Maybe that was a mistake to do a “submarine” initiative. They should have promoted Scrum at the company level. Because it will impact other teams.

The groundlagen team does “work package” three times a year. This team needs approx 3 month to add something. Won’t fit with agile process, so they need to do things on their own because it’s too long. There’s a gap, but not only in term of length of the iteration, but also in term of dev. Philosophy.

Mascha asks about tools:

* Template printed and sticked to the wall.
* User stories on the wall.
* Task on the wall.
* Bug are tasks, and on the wall.
* Non-functional user stories

Scrum says “you can’t take a team member away during the sprint”, but it happens if there’s some high priority things that happens on another project. 🡪 Scrum-but

- Spring backlog is an excel sheet

- Template for backlog items

- Rest is done with cards

- Word document for test documentation

-SCM only for software

- Other information are spread with shared folder, manual versioning with version number

- Cahier des charges of poor quality is initially. It’s impossible to do a requirement engineering phase correctly. Feature creeping. Better start doing something, crawl the code, and adapt.

 “That’s a problem to have all the document up to date for everybody”. \*\*\*\*\*\*

Not many tests are automated. Spring release + manual test. Bug are introduced in existing features, but there is no automated regression testing to discover them.

There are 3 dev in the team, and one of them is not keen on sticking card on the wall. He needs to be forced. But the wall is great to keep visibility about the project, notably when PO comes.

A tool might help, but if you need to sync the tool with the board, that would not work. If it’s fully automatic, why not. There’s so many tasks now, that they don’t have them in an electronic form.

They use a test tracker for bugs, then the PO priorizes the bug, and “convert” them to user stories.

There’s a special story “Fix all high priority bug” that represent the bug that are reported and need to be fixed, without waiting for the next iteration. This story is estimated with a defensive approach. \*\*\*\*\*\* 🡪 special stories

Mascha asks bout *knowledge sharing*.

People are reluctant to physically move, as would be required by the process.

Older people have seen so many process, they are not keen to change to adapt to the process.

Mascha ask about *product vs. project* knowledge

-Design document for all the part of the machine. Rapsody is used (UML code gen), tool introduced 2 yr ago. Not much documentation in the code any more. “We need to also keep track of why we **haven’t** done something”. Need to keep track of the **decision**. 🡪 \*\*\*\*\* information vs. decision

Now the company realizes that if one of these guy leave, they are in big trouble, because nobody can fix the bug. \*\*\*\*\*\*\*\*\*

We have machine now in the field since 10 yrs, with feature creeping in. That’s how the company makes money. The initial development is just a tiny part.

Masch ask about the *retrospective*

He would like to change a lot between the “edges” of the two processes. Reduce the friction between the processes \*\*\*\*\* Scrum is rarely the only process, most of the time, it participates into a bigger project with a bigger process, or different process (e.g. soft vs. hardware)

There’s something missing, like a loop where everybody is able to push requirement, which are handled with high priority. Like Toyota and Lean. Someone needs to be in charge of the product. For software, it’s the Scrum PO, but for larger project we need a product owner.

Linking info

Weak links, maybe from code to bug tracker.

Management fear vendor lock-in.

We have too many information, we can’t find anything there. You need to discuss with people. The name of the shared folder is poor. Network search. \*\*\*\*\*

You are better on it walking through the whole dev department, and ask people.

No vision paper. There’s one company owner, and he owns the vision. The dev. Just guess what’s the market need, and why they do things. Product management is poor.