The 2020 Scrum Guide - HIGHLIGHTS by Pumbaa

In order to obtain a *PSPO I Scrum Certification* I compiled some highlights from The 2020 Scrum Guide. You can find all my personal documentation and files on my dedicated github repository.

Scrum Definition

"Scrum is intended as a simple, yet sufficient framework for complex product delivery. Scrum is not a one-size-fits-all solution, a silver bullet or a complete methodology. Instead, Scrum provides the minimal boundaries within which teams can self-organize to solve a complex problem using an empirical approach."

Scrum exists only in its entirety and functions well as a container for other techniques, methodologies, and practices. Rather than provide people with detailed instructions, Scrum guide their relationships and interactions. Scrum employs an incremental approach to optimize predictability and to control risk. It consist of **Three Pillars** (*Transparency* > *Inspection* > *Adaptation*) and **Five Values** (*Courage*, *Focus*, *Commitment*, *Respect*, *Openness*). It should be used for complex projects. If you are able to predict your work more than 3 sprints ahead you should not be doing Scrum as your product is not complex enough.

Scrum is a lightweight framework

- The *Product Owner* orders the work for a problem into a *Backlog*.
- The *Scrum Team* turns a selection of the work into an *Increment* of value during a *Sprint*.
- The *Scrum Team* and its stakeholders inspect the results and adjust for the next *Sprint*.

Forecasting : Scrum does not forbid up-front planning.

Sprint

Short : a month or shorter.

Product Owner

The **Product Owner** ensures that **attendees** are prepared to discuss the most important *Product Backlog Items* and how they map to the *Product Goal*.

He is one persone, not a committee. He can be helped by other persons, but he needs to make the decisions.

One *Product Backlog* means we need one *Product Owner*. It creates the transparency required for proper empiricism. Having many *Product Owners* encourages micromanagement of teams, requires coordination and bring unclear responsibility.

How will the chosen work get done? For each *Product Backlog* item, the Developers plan the work. This is done by decomposing *Product Backlog* items into smaller work items of one day or less. How this is done is at the **sole discretion of the Developers**.

Product Goal and Sprint Goal

The **Product Goal** describes a future state of the product which can serve as a target for the *Scrum Team* to plan against. The *Product Goal* is in the *Product Backlog*. The rest of the *Product Backlog* emerges to define "what" will fulfill the *Product Goal*.

The **Sprint Goal** is the single objective for the Sprint. The Sprint Goal creates coherence and focus, encouraging the *Scrum Team* to work together.

It is created during the *Sprint Planning* event and then added to the *Sprint Backlog*. If the work turns out to be **different than they expected**, they collaborate with the Product Owner to **negotiate the scope** of the Sprint Backlog within the *Sprint* without affecting the *Sprint Goal*.

Product Backlog and Sprint Backlog

The **Product Backlog** is an emergent, ordered list of what is needed to improve the product. **Sprint Planning** initiates the Sprint. 8h or shorter.

The **Sprint Backlog** includes The Sprint Goal, the *Product Backlog* items selected for the Sprint, plus the plan for delivering them. It is flexible, as long as changes do not distract from the focus on the *Sprint Goal*. Could be done during the Daily Scrum.

Sprint Review and Sprint Retrospective

The **Sprint Review** inspect the outcome of the Sprint and determine future adaptations. The *Product Backlog* may also be adjusted to meet new opportunities. It's a *working session*, not a *presentation* ! 4h or shorter. They are an opportunitie to collect user and stakeholder feedbacks.

The **Sprint Retrospective** plan ways to increase quality and effectiveness, identifies what went well during the *Sprint* and what problems it encountered. You can add things to next *Sprint Backlog*. 3h or shorter.

Si on identifie des points d'améliorations, les noter dans le Sprint Backlog, car de toute façon le prochain event est le Sprint Planning.

Work cannot be considered part of an **Increment** unless it meets the **Definition of Done.**

Sprint Review vs Sprint Retrospective The Sprint Review focuses on the "inspect" and "adapt" of the increment (Potentially shippable), while the Sprint Retrospective give more focus on the "inspect" and "adapt" of the process of the sprint.

The Scrum Team

The **Scrum Team** consists of **one Scrum Master, one Product Owner, and Developers.** They are cross-functional and self-managing. It's small enough to remain nimble and large enough to complete significant work within a Sprint, typically 10 or fewer people. If Scrum Teams become too large, they should consider reorganizing into multiple cohesive *Scrum Teams*, each focused on the same product. Therefore, they should share the same *Product Goal*, *Product Backlog*, and *Product Owner*.

Daily Scrum

Daily Scrums improve communications, identify impediments, promote quick decision-making, and consequently eliminate the need for other meetings. 15 min.

Empiricism

Definition

Empiricism, the act of making decisions based on what is, not fictitious plans.

We do the best we can with what we have.

Three Pillars

Transparency > Inspection > Adaptation

- Transparency : We all know what is going on
- Inspection : Check you work as you do it
- Adaptation : Ok to change direction

Adaptation : What to change ?

- Too much : If it is a significant or challenging thing, then only take one action. Talk about this item in each Daily Scrum.
- Nothing : The current state can be compared with the goal, and then find the one change that will give the most benefit for the least effort. **Once you are moving the momentum can grow.**
- Team VS Orga : The organisation needs to move to the mindset that things will be different, every day, every week. That is at the heart of business agility.

Five Values

F.O.R.C.C : Focus, Openness, Respect, Courage, Commitment.

Successful use of Scrum depends on people becoming more proficient in living these five values. People personally commit to achieving the goals of the Scrum Team. The Scrum Team members have courage to do the right thing and work on tough problems. Everyone focuses on the work of the Sprint and the goals of the Scrum Team. The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work. Scrum Team members respect each other to be capable, independent people.

Examples of bad application of these Five Values

- Committing to something that you don’t understand because you are told to by your boss instead of committing yourself to the team and Sprint Goal.
- Focusing on keeping the customer happy instead of being focused on the Sprint and its goal.
- Telling everyone everything about all your work instead of highlighting when you have challenges and problems that are stopping you from success.
- Thinking you are helping the team by being a hero instead of helping people to learn the things that you are good at and not judging the things that others aren’t good at.
- Continuing to push back instead of being transparent, or that your opinion is not the direction that the team is going.

Sprint Goal

Talk to the Product Owner : what is it that we want to achieve at the end of this Sprint?

- Make it visible.
- Teach the team to talk about progress towards the *Sprint Goal* in the **Daily Scrum**. It should be the **only topic** of conversation.
- Make the *Sprint Goal* a team measure and keep it visible in the team space.
- Make it **business** or **user** focused when possible.
- Make it focused on testing business assumptions and getting **feedback**.

A *Sprint Goal* makes more sense than a bunch of *Product Backlog Items* packed together.

Invest time in Product Backlog refinement.

Backlog refinement

Myth: Refinement is a required meeting for the entire Scrum Team. The Scrum Guide is quite clear; refinement is not an event in Scrum. It is something that Development Teams do as a natural part of development. It is not something that necessarily happens at a *fixed moment* during the Sprint where *the entire Scrum Team* has to attend.

- **Clarifying** items on the *Product Backlog* that are too unclear to start work on.
- **Breaking down** items that are too big to pull into a Sprint.
- **Re-ordering** the *Product Backlog* to make the upcoming Sprints as valuable as possible.
- **Adding** or **removing** items from the *Product Backlog* as new insights emerge.
- **Estimating** the effort involved in implementing particular items.

Why Estimate At All?

Estimation is not so much about the estimates, but about the **discussions** that it triggers. It’s about having a **conversation** — ideally with the actual customer — and creating a shared understanding.

When those estimates are exposed beyond the team's circle of trust we may indeed run the risk of abuse, of story points being commoditized, of teams being compared or obliged to bid for work using points as a cryptocurrency, and other abominations. In Scrum this is a risk which lies squarely with a *Product Owner* to manage. As a member of the Scrum Team, the Product Owner is trusted to understand and respect the Development Team's estimates and to use any associated projections sensibly. The Product Owner will understand the limitations of using estimates to measure progress, and the importance of recalibrating a Product Burndown and any forecasts in light of the empirical evidence brought about by release. If there is doubt about the ability of other stakeholders to consume this data, then the Product Owner - as their representative, advocate and arbiter - must decide whether or not they ought to be exposed to estimated measures and forecasts at all. Perhaps they aren't.

A Good Sprint Goal

At end of a *Sprint*, is the entire team in agreement on whether or not the *Sprint Goal* has been achieved? If not, the *Sprint Goal* may be too vague.

- Ask "how will we know if we have achieved the *Sprint Goal*?"
- Make the *Sprint Goal* measurable

Bad examples

- "Get Jira tickets 17322, 17323, 171400 and 17888 to done"

- "Finish the authorization issues, create a procedure for capturing search queries and make it generic, create a new theme for mobile version"
- "Change the page to support HTML5 elements"
- "Enhance shopping cart functionality. Streamline purchasing process to enable an increase in conversion rates"
- "Improve performance. Increase page load time by X%"
- "On-board new market segment. Enable new market segment to purchase Service Y"

Good examples

- "Implement the functionality for user registration"
- "Launch a partnership program (traders) for acquiring new clients and increasing the company's earnings"
- "Test technology X and technology Y, make a final decision based on the results"
- "Partners need to get remuneration according to the selected earnings model"

What is a Definition of Done (DoD)

- **A short, measurable checklist** – try and have things on your DoD that can be measured.
- **No further work required** from the Developers to ship your product to production. Any additional work means that you were not Done. Ideally, you have a fully automated process for delivering software.

Why is "Done" so important?

Well, incomplete work has a nasty habit of mounting up. The tyranny of work which is nearly done, but not really done, can put a team in servitude to **technical debt**. Eventually the expense of fixing things, of repaying the debt, may exceed the value to be had from the next product increment.

Good examples

- Increment Passes SonarCube checks with no Critical errors.
- Increment’s Code Coverage stays the same or gets higher.
- Acceptance Tests for Increment are Automated.

Undone

Most, robust websites have the ability to rollback, but actually, it is much harder to do with complex transactional systems. If we ever want to break the "we can't release yet" cycle we have to start thinking about undone. How do we pull this back? Can we automate that?

CI/CD - Continuous Integration / Continuous Delivery

In the beginning, there was Continuous Integration (CI). Integration with others was normally a nightmare fraught with blame, fingers being pointed and problems. Then automated testing and then.... But there was a flaw – We delivered our code, even tested it, but then moving into production was a nightmare and there was a cost. Continuous Delivery was the response to this. It basically applied the ideas of CI on a much grander scale. This whole CI/CD/DCA would be documented in a simple artifact, the Definition of Done (DoD). The DoD guides the team as they plan, do and deliver work.

This is where we can put the non-fonctional development, like optimisation or refactoring. The infamous *nice-to-have*. The developpers are accounted to express their concerns about potential futur problems and the necessity to correct them soon enough.

Résumé du Workshop SCRUM - Product Owner par Samuel

- Les individus et interactions priment sur processus et outils : Instaurer un dialogue.

- Livrer un logiciel opérationel plutot que documentation exhaustive : On parle de documentation Utilisateur ou Développeur ? --> Documentation de projet. Il faudra le faire mais ce n'est pas le **but en soi**.

  Orienté Produit.

- Collaboration avec client plutot que négociation de contrat : Le client devient actif, il doit travailler avec la Scrum Team.

4 valeurs, 12 principes

- Satisfaction du client.

- Accueil **positif** des changements, car les changements sont naturels.

- Coopération quotidienne entre utilisateurs et dév.

- Livraisons fréquentes de versions **operationnelles**.

- Un logiciel opérationnel est la principale mesure d'avancement.

A chaque Sprint on doit avoir un produit livrable, mais pas forcément livré en prod (car dépendant d'un autre aspect, par stratégie marketing ou autre)

SCRUM vs Chef de Projet

Le rôle de chef de projet est éclaté entre plusieurs rôles Scrum, il n'existe plus en tant que tel car c'est un gros bottleneck et toute la responsabilité lui retombe dessus.

Il remédie aux imprévus, donc même si l'équipe connait bien Scrum et travail efficiacement il sera tjs nécessaire.

Le **Scrum master** devrait connaitre l'aspect technique/développeur.

Le **Product Owner** pas forcément. Idéalement il devrait venir de l'entreprise client, qui connait le mieux le métier.

Online Tests

- Product Owner Open

- Scrum Open