

Joe's Unofficial Scrum Checklist

This list is based off Henrik Kniberg's Unofficial Scrum CheckList. See <http://www.crisp.se/scrum/checklist>

We recommend you use this list as basis for discussion, mostly likely discussion with the full team. Only a portion of the items have a yes/no response. The purpose of the good discussion should be action and improvement. Yes, it is good to do Agile in a more pure way, but only because (we are completely convinced) it will almost always lead to better results.

You may wish to compare Henrik's and my versions. Perhaps the comparison will make you think. Some of the differences might be explained by differences in the kinds of problems each of us (Henrik and I) have encountered most often.

This checklist should be used to review and consider. I think policing and blaming are less effective. Almost always we find teams can be doing better by doing Scrum better, but often enough the ideas suggested below may not work in your specific situation. And in any case, we are all always learning.

To me, attitude is quite key. We should always be asking ourselves: "In what way are we being stupidest today?" "Isn't there a better way to do it?" "Which is the next big impediment to remove?"

There is a lot more I wished to say, but I tried to keep it shorter. I trust that you will use common sense in applying these ideas. These days, as perhaps all days, common sense is not common enough.

The Bottom Line

Is the team having more fun? ¹

Are you delivering or releasing what the customer wants most?

Is it being delivered faster? (Less importantly: cheaper?)

Are you releasing more frequently?

¹ I am a businessman and I have an MBA. I bow to no one is wanting to see a team deliver more business, not least because each team member will have a more satisfying life. And I think it starts, for these creative teams, with having more fun. At least more fun than I typically had with waterfall teams. More 'serious fun', if you prefer that phrase.

Is the Team delivering working (eg, tested) product every 4 weeks or less? ²

Are you continuously improving?

Core Scrum

If you don't have all of these, you probably shouldn't call it Scrum. Maybe 'ScrumButt' (said with a smile).

The Team has a clear **Vision**?

This is in sync between the Business Stakeholders, the Product Owner (PO) and the rest of the Team?

The Team finds the Vision motivating?

The Vision and the Product Backlog seem to be in sync?

Do you have a clearly defined **Product Owner** (PO)?

Does the PO have the skill, respect, and power to prioritize (order) the Product Backlog (PB)?

Does the PO have the knowledge to prioritize?

Does the PO have direct contact with the Team?

Does the PO have direct contact with the Business Stakeholders?

Do you have the best possible Business Stakeholders? (eg, maybe, in some situations, real customers)

Do you have a clearly defined **ScrumMaster** (SM)?

Does the SM maintain a public Impediment List?

Is the Impediment List prioritized? How (what is the basis)?

Does the SM drive the removal (or mitigation) of impediments?

Which types of impediments is the SM best at identifying (or having the Team identify)? Which ones is the SM worst at identifying?

² This is very important. And probably on average the biggest thing that a Scrum team should improve. Working product does not necessarily mean the product (software) has been released for general use by the customers.

Which types of impediments is the SM best at remediating? Which ones is the SM worst at remediating?

Is the SM equally comfortable with the Business side and with the Technology side? And in contact with both?

Is the SM good at getting the right chickens to support impediment removal?

Do you have a small, stable, fully-dedicated **Team**?

Is the Team size 7 +/- 2? Plus a PO and a SM?

Does the Team have most of the skill sets to complete the Product Backlog Items (PBIs) in the PB?

Does the Team feel like a real Team (eg, 'all for one and one for all')? (Team includes PO and SM.)

What are the biggest 'people' challenges for the Team?

Does the Team have a single, good **Product Backlog**?

Are the PBIs in the Release ordered reasonably to achieve maximum Business Value for the release?

Is the effort estimated for each PBI in the release? By the Team?

Are the top PBIs small enough to fit in a Sprint? (eg, typically 8+ in a 2 week Sprint)

Are the top PBIs defined well-enough for the Team to execute them quickly?

Did the Team complete **Sprint Planning** in a reasonable timebox?

Does the PO bring an up-to-date PB?

Does the whole Team participate (implementors, PO, SM)?

Did the Sprint Planning result in a good Sprint Backlog?

Does the Sprint Backlog represent a reasonable commitment, given the Sprint length and other factors?

Is the commitment usually fulfilled? (Or do we regularly over-commit?)

Do team members (or Business Stakeholders) often feel 'we're not working on the right things?'

Are Business Stakeholders there to confirm the quality of the inputs to the Sprint? (eg, that we are doing the most important stuff)

Is the **Sprint Backlog** highly visible and updated daily?

Did the Team create the Sprint Backlog, and do they feel they own it?

Is the **Daily Scrum** done every working day, within 15 minutes?

Does the whole Team participate? Including the SM and the PO?

Are impediments (problems, etc) surfaced?

Does the Team feel like they know what their comrades are doing?

The Team maintains a **Sprint Burndown** chart?³

Is this maintained daily, with accurate, timely estimates of work remaining?

Is the Sprint Burndown chart visible to all?

Does the Team use the Sprint Burndown chart to see 'where we are now' and then to self-manage themselves to greater success in the Sprint?

Does the Team do a **Sprint Review** every Sprint?

Does it include a demo of the Working Product?

Does the Team have Working Product for every Demo? Are the PBIs done according to the Definition of Done?

Does the PO bring useful Business Stakeholders to the Sprint Review?

Does the Demo elicit useful feedback (positive and negative) early enough?

Does the Team have a clear **Definition of Done** (DOD)?

Is the DOD currently achievable within the Sprint? Are the PBIs (stories) small enough to make the DOD do-able?

³ Technically, it is not essential that the team track 'work remaining' in a burndown chart. What is essential is that the whole team knows 'where we are' each day (work remaining) and that they self-manage to success for the Sprint. Frankly, we have not seen a better way to do this than through the Sprint Burndown chart.

Does the Team actually comply with the DOD?

Does a **Retrospective** happen every Sprint?

Is there a reasonable time box for the Retrospective?

Does the meeting results in concrete improvement proposals?

Are some of these proposals successfully implemented?

Does the whole Team participate? (including PO and SM)

Is progress (eg, higher velocity) seen quickly (eg, by the end of the following Sprint)?

Are the **Sprints** of a consistent length (4 weeks or less)?

Does the Team respect the Sprint time-box (ie, no extensions of the Sprint and extremely few changes to sprint length)?

How much does the Team feel disrupted during the Sprint? By whom?

How often does the Team deliver at least the PBIs that they commit to?

The **Release/Product Plan is refactored** almost every Sprint to some degree? ⁴

The whole Team participates in refactoring the Release Plan, including re-estimating the effort on PBIs?

The PO, the Team, and the Business Stakeholders are becoming progressively more comfortable with the benefits of (and issues with) adaptive release planning?

Adaptive planning is seen more and more as a powerful tool rather than just a succumbing to the rapidity of change?

The Team knows their team **Velocity** (productivity rate)? By some useful measure.⁵

The Team commits to work based partly on their past velocity?

⁴ Release Planning is optional in Scrum, although I think it is almost always needed. See a footnote below that discusses this further.

⁵ “Velocity” here is not necessarily the sum of the Story Points completed. It is any reasonable measure of the Team’s productivity in the Sprint.

The PO can use the velocity to predict when the Release will be complete?

The Team challenges themselves to improve their velocity by removing impediments? (And not by working more hours.)

Recommended

Is the Team **co-located**? (eg, within 30 meters of each other, on the same floor) If not, how well have the impediments of distributed Scrum been addressed?

Do the Team members sit in a Team Room together? (eg, during 'core hours' from 10am to 2pm?)

Did the full team do the **initial Release Planning**, including Vision; Product Backlog creation; Business Value estimating; Effort estimating; discussion of Risks, Dependencies, and Other factors; Ordering the work; Deciding the Scope-Date trade-off, etc.? ⁶

How much is the Team dependent on **skill sets outside the Team** to deliver the PBIs they are working on?

How much are the team members **staying within specific sub-roles**? How much are they learning to work outside their specific roles? How much do they help each other?

Has a Sprint ever been **abnormally terminated**? If so, why? (Our view is that abnormal terminations should be very abnormal. Such as, when all the remaining work in the Sprint that could get done is useless. Fail levels of failure are not a good enough reason, and could become a good basis for learning by people inside or outside the Team.)

Is the Team using **User Stories**? ("As a role X, I want to do Y, so that Z") Are they good User Stories? (see, for example, the INVEST criteria)

Does the Team use **Agile Specifications**? (just enough information, delivered just in time)

The PB and the Vision are **highly visible**?

⁶ Scrum defines Release Planning as optional. And I agree it is not necessary in a very few cases, eg, on web apps where we release at the end of every sprint. Still, I think longer term 'product' planning is still very useful in virtually every real situation I can remember right now. Doing it as a full team accomplishes many wonderful things.

Everyone on the Team participates in **estimating** the effort? ⁷

Does the **SM sit with the Team**?

The **PO is available** when the Team is (re)estimating the effort?

The best experts on Business Value play **Priority Poker** to estimate the relative Business Value of each PBI?

The PO is able to start executing the **Pareto Rule** on the PB? (The 80-20 rule.) ⁸

Estimates of effort are in relative size (**story points**), not time (eg, hours, days)?

The whole Team knows the **top 5 impediments**?

Appropriate **impediments are escalated** to managers or a management “impediment removal team”? Usually leading to successful action?

PBIs are broken into **Tasks** in the Sprint Backlog? ⁹

The Sprint Tasks are **estimated**?

The Task estimates are **updated daily** (amount of work remaining to be done)?

The Team understands that any Task can be **re-estimated at any time**?

The Team uses Story Points (SP) to measure the team **Velocity**?

All PBIs have an **SP estimate before Sprint Planning**?

The PO uses the SPs and the Velocity to work out or **refactor the Release Plan**?

⁷ It is essential that the Team estimate the effort for themselves. It is not always essential that every single member of the Team participate in estimating, although we have not seen many good reasons to exclude a Team member.

⁸ We are suggesting that Priority Poker enables the PO to see and execute better on the Pareto Rule.

⁹ Some of the more advanced teams, who typically have very small user stories, do not break stories into tasks, and this can be quite successful. But I do not recommend that for teams less than 2 years old.

The Team 'scores' SPs for velocity only when the **PBIs are done** (according to the DOD)?

Is the **Daily Scrum** always at the same time and place?

PO typically participates?

When the PO is absent, are the reasons personal (day off) or 'customer' only (eg, he is talking to the customers or business stakeholders about this team's work)? Or is the PO insufficiently allocated to or engaged with the Team?

The Team discusses **Technical Debt**, and actively tries not to build any more technical debt. And maybe tries to reduce Technical Debt.

The Team considers that knowledge creation (Cf papers by Takeuchi and Nonaka) is the most important thing they do as a team?

Engineering Practices

The team is constantly improving its engineering practices?

The team is making some use of the idea of 'two heads are better than one'? Example: with some 'pair programming' (if in a software domain)? (See an Extreme Programming book for details on pair programming.)

The team is doing Test Driven Development at the Unit Test level? (with automated tests)

The Team has at least considered Test Driven Development at the Functional Test or Acceptance Test level? (meaning, mostly automated tests that could be included a functional regression test)

Where feasible at all, the team is building automated Functional or Acceptance tests for each PBI or user story?

The team's coding standards have recently been raised a notch? And are currently considered to be superior to other firms in your industry?

The Team uses a 10-minutes Build, that is tied with Continuous Integration (assuming this is feasible for your product)? And the 10-minute Build includes fast feedback to the implementor who has made a mistake? And the feedback includes any issues identified in a mini-regression test?

Scaling

If you have more than 2 Teams scaling together, each Team has a PO and the group has a **Chief Product Owner (CPO)**?

The CPO and the POs act as a good **product owner group**?

Inter-dependent teams do a **Scrum of Scrums**? Is it useful?

Inter-dependent teams do a **product integration** multiple times within the Sprint? How effectively?

Inter-dependent (software) teams use full **continuous integration**? How robust is the associated automated (mini) 'regression' test? ¹⁰

Positive Indicators

Team energy level is high.

Lots of laughter.

Team members feel like this is the best team they have ever been on.

"This is serious fun."

Overtime work is rare and, if it occasionally happens, happens voluntarily.

No one is ever working more than 10% overtime in any Sprint.

The Team is 'fighting' constructively about the ideas and implementation issues in various domains.

The Team is willing to try things and 'fail fast'. And learn from that.

Experimentation and learning applies to most domains (eg, the product, the process, etc.).

.....

Version 1.3

© Joseph H. Little See also: LeanAgileTraining.com

¹⁰ Not all organizations call the testing associated with Continuous Integration (CI) a regression test. Typically it is in effect a mini-regression test. It may be called a 'smoke test' and a build test, or other names. In any case, we care about the substance, not which words you use for it.