

# Scrum Master Checklist

---

## Die Arbeit meines Produkt Owners

- Ist der Product Backlog auf dem aktuellsten Stand, spiegelt also auch die letzten Ideen und Gedanken des PO's?
- Sind alle Anforderungen aller Stakeholder im Backlog hinterlegt?
- Hat der Produkt Backlog eine handhabbare Größe? Zur Pflege eines überschaubaren Backlogs sollten die nach oben (hoch priorisiert) Einträge immer feiner / granularer werden. Nach unten reichen allgemeinere Epics aus.
- Gibt es im oberen Bereich Einträge im Backlog, die besser strukturiert und formuliert sein sollten (Unabhängige, Verhandelbare, Wertstiftende, Schätzbare, Kleine und Testbare User Stories)?
- Weiß dein PO über die Auswirkungen von „Technical Debt“ bescheid und ist er sich darüber bewusst, wie man es vermeiden kann? Eine Möglichkeit unter vielen ist die Berücksichtigung von Refactoring und Testautomatisierung in der Definition of Done.
- Dient der Produkt Backlog als Wissensdatenbank, die für alle Stakeholder einsehbar ist?
- Wenn Ihr eine Software Lösung zur Pflege des Backlogs verwendet: Weiß jeder, wie dieses Werkzeug verwendet wird? Bei der Verwendung von EDV Tools, die nur wenige kennen und verwenden können, besteht die Gefahr, dass die wichtigen Informationen nicht ausreichend gestreut werden.
- Besteht die Möglichkeit übersichtliche Print-Outs aus dem Backlog zu erstellen?
- Lassen sich mit Hilfe des Backlog-Tools „BIG VISIBLE CHARTS“ erstellen? Große Diagramme, die an der sichtbar an die Wand gebracht sind, sind weit effektiver als Charts, die sich nur über Websites aufrufen lassen.
- Organisiert dein PO Backlog Einträge in größeren Strukturen (Release Plan, Front Burner, Back Burner)?
- Wissen alle Stakeholder ob der jeweils aktuelle Release Plan der Realität entspricht (z.B. basierend auf der aktuellen Velocity)?
- Passt der Produkt Owner den Release Plan nach jeder Iteration an die aktuellen Gegebenheiten an und sind diese Änderungen für alle Stakeholder einsehbar (z.B. durch Product Burndown Charts)

## Die Arbeit meines Teams

- Arbeiten die Team Mitglieder einen Teil ihrer Zeit im FLOW – Zustand?  
Charakteristika dieses Zustands nach Mihaly Csikszentmihalyi:
  - Klare erreichbare Ziele und Erwartungen, die an den Skills und Möglichkeiten der Menschen ausgerichtet sind.
  - Ein hoher Grad an Konzentration und Fokussierung auf einen kleineren Aufmerksamkeitsbereich
  - Verlust an Bewusstheit, da Handlung und Aufmerksamkeit ineinander übergehen

- Verzerrte Zeitwahrnehmung – das subjektive Gefühl für Zeit hat sich verändert.
- Balance zwischen Möglichkeit und Herausforderung – Die Aufgabe ist weder zu schwierig noch zu einfach.
- Direktes und unmittelbares Feedback
- Die Aktivität ist intrinsisch motiviert, wodurch der eigentliche Aufwand in der Freude am Tun verschwindet.
- Mögen sich die Mitglieder des Teams? Erfolg und Misserfolg wird immer gemeinsam getragen.
- Erwarten die Team Mitglieder einen hohen professionellen Standard voneinander? Fordern sie sich gegenseitig heraus, um aneinander und miteinander zu wachsen?
- Gibt es Probleme oder Chancen, die das Team aus bequemlichkeit nicht miteinander bespricht? Sorge du als Scrum Master dafür, dass auch schwierige Gespräche und Diskussionen gemeistert werden können.
- Habt ihr bereits verschiedene Formate und Locations für die Team Retrospektiven ausprobiert?
- Fokussiert sich das Team auf die Erfüllung der Akzeptanzkriterien? Führe mindestens einmal während des Sprints ein Review der Akzeptanzkriterien durch, um festzustellen ob das Team On-Track ist.
- Stimmt die aktuelle Aufgabenliste am Scrum Board mit dem überein, was das Team tatsächlich gerade tut?
- Ist das Task-Board auf dem aktuellsten Stand?
- Sind die Artefakte des Teams, die für das Selbstmanagement nötig sind, für alle Mitglieder sichtbar und einfach zu verwenden? (Task Board, Burndown, etc.)
- Sind die Artefakte des Teams vor „Mikro-Management“ geschützt?
- Erledigen die Team Mitglieder ihre Aufgaben von sich aus / freiwillig?
- Werden Aufgaben zur „Rückzahlung“ der „technischen Schulden“ (technical Dept) im Backlog gekennzeichnet (Sie führen zu einer geringeren Geschwindigkeit) und mit dem PO besprochen? Gibt es z.B. das Akzeptanzkriterium „Wir hinterlassen Code immer sauberer, als wir ihn vorgefunden haben“?
- Fühlt sich das Team gemeinschaftlich verantwortlich für Testaufgaben, Dokumentation etc.?
- Misst das Management das Team am gemeinsamen Erfolg?

## **Unsere Engineering Practices**

- Gibt es einen „Push-To-Test-Button“, der Gesamtsystem automatisch testet und die Ergebnisse jedem Entwickler und Stakeholder zur Verfügung stellt (z.B. durch generierte Berichte)? Damit weiß jeder, wenn es irgendwo ein Problem gibt.
- Habt ihr eine ausgewogene Balance zwischen automatischen End-To-End System Tests (aka Funktionale Tests) und Unit Tests?
- Schreibt das Teams sowohl System Tests als auch Unit Tests in der selben Sprache und Umgebung, in der das eigentliche System entwickelt wurde?
- Gibt es einen Server, der für die Continuous Integration zuständig ist und stündlich Alarm schlägt, wenn jemand Code veröffentlicht, der zum Fehlschlagen von Regressions Tests führt.
- Werden alle geschriebenen Tests in den Ergebnissen des Coninuous Integration Servers berücksichtigt?
- Haben alle Teammitglieder die Vorteile von Continous Design und Refactoring im Vergleich zum „Big Up-Front Design“ verstanden?
  - Refactoring (die Änderung der Struktur des Systems ohne das eigentliche

Verhalten zu ändern) sollte mehrmals in der Stunde stattfinden: Immer wenn es doppelten Code, komplexe Bedingungslogiken, schlechte Namen von Variablen und Methoden, Übermäßige Kopplung von Objekten etc.

- Sicheres Refactoring funktioniert nur mit einer ausgeprägten Abdeckung von automatischen Tests (Unit Tests)
- Lücken in Testabdeckung und Vermeidung von Refactoring führen zu Technical Dept.
- Beeinhaltet eure Definition Of Done für jedes Backlog Item vollständig automatisierte Testabdeckung sowie Refactoring?
- Entwickeln die Team Mitglieder die meiste Zeit in Paaren (Pair Programming)? Wichtig ist hier dass jedem (auch über das Team hinaus) Qualität wichtiger ist als Quantität.

## **Die Organisation**

- Gibt es ausreichende Kommunikation zwischen verschiedenen Teams und Abteilungen?
- Treffen sich die Scrum Master regelmäßig um gemeinsam an der Beseitigung von systemischen, organisatorischen Hürden zu arbeiten?
- Finden sich die organisatorischen Hürden an einer Wand der IT Leitung? Können die Kosten dieser Items in Euros, reduzierte TTM (Time To Market), reduzierte Qualität, reduzierte Kundenbindung etc. ausgedrückt werden?
- Sind Zielvereinbarungen, Karrierepfade etc. der Organisation auf die Kollektivziele des Scrum-Teams ausgerichtet? (Die Antwort lautet NEIN, wenn es Karriereanreize auf Programmier- oder Architekturaufgaben gibt, die auf Kosten von Tests, Dokumentation und Refactoring durchgeführt werden müssen.)
- Ist die Organisation als einer der besten Arbeitgeber und/oder Marktführer erwähnt worden (z.B. in Wirtschaftsmagazinen und anderen unabhängigen Quellen)
- Hilfst du dabei eine „Lernende Organisation“ aufzubauen?