Agiles Schätzen

Quelle: Kap. 7 aus „Wie schätzt man in agilen Projekten – oder wieso Scrum-Projekte erfolgreicher sind“ [Boris Gloger 2014]

Schätzen der Größe

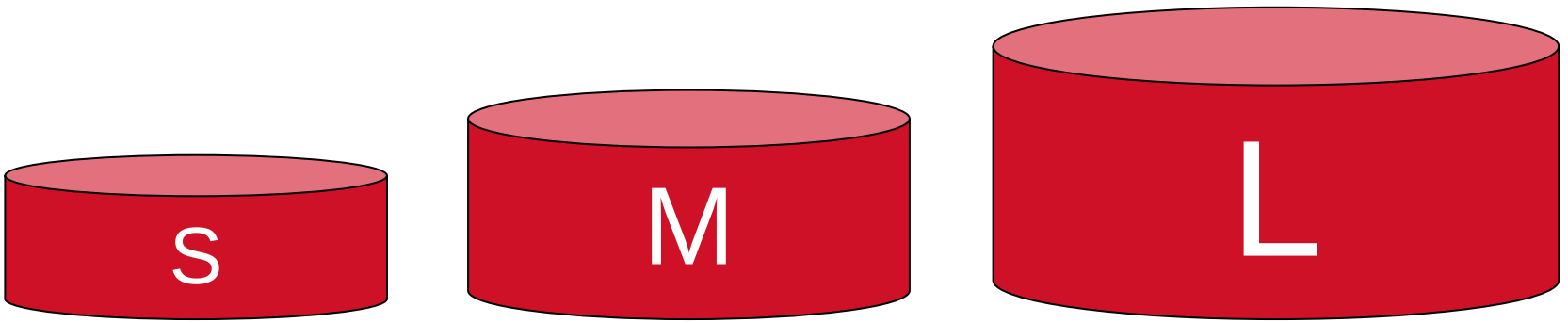
- **Wir bestimmen die Größe, nicht den Aufwand.**
- **Auf Basis der Größe lassen sich auch die Projektkosten ableiten.**
- **Auf Produktebene**
  - ersten Überblick gewinnen, ob es ein kleines oder großes Projekt ist
  - Einblick in die notwendigen Technologien erhalten
  - Eindruck von den relevanten Stakeholdern bekommen
  - Gefühl für die zeitliche Dimension des Projektes entwickeln
- **Auf der Entwicklungsteam- und Sprint-Ebene**
  - erkennen, ob die Skills des Teams reichen
  - beurteilen, ob eine User Story in einem Sprint Platz findet oder zu groß ist
  - lernen, worum es genau bei einer einzelnen Story geht
  - Abhägigkeiten zu anderen Entwicklungsteams erkennen
  - eine Vorstellung der nötigen Architektur bekommen

Fazit: Schätzungen generieren Antworten, die das Risiko gravierender Fehlentscheidungen minimieren.

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten

Nicht Aufwand, sondern Funktionalität

- **Wie viele Funktionen hat Ihr Auto?**
  Was kann mehr, Radio oder Außenspiegel?
- **Prinzip: Funktionalitäten in eine Reihenfolge bringen**
  Vorstellung von drei unterschiedlich großen Körben mit jeweils drei unterscheidlich großen Körben darin.



S	M	L
1	2	3

S	M	L
5	8	13

S	M	L
20	40	100

Das Verständnis für das Schätzen von Funktionalitätsgrößen anstelle von Aufwänden ist zu Beginn schwierig.

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten

Warum schätzen wir in Funktionalität?

- **Funktionalität ist konstant**
  - „Was das Ding tut“ ändert sich nicht über die Zeit.
  - Es kann höchstens sein, dass sie nicht detailliert genug beschrieben ist.
  - Die Angaben sind daher auch immer Schätzungen.
- **Betrachtung aus der Perspektive des Users**
  - Entwickler müssen mit der Sicht des Anwenders schätzen
  - führt zu einem besseren Verständnis der Zielsetzung
  - erfordert enge Kommunikation mit dem Product Owner
- **Schätzungen sind auch für Nicht-Techniker transparent**
  - es ist für alle nachvollziehbar, wie das Team zu den Schätzungen kommt
  - Fachabteilung und Entwicklung müssten zu den selben Schätzungen kommen
  - Ansonsten wird eine unterschiedliche Vorstellung der Funktionalität identifiziert

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten

Methode: Magic Estimation

- **Eigenschaften**
  - auch für viele Schätzer und viele Stories anwendbar
  - schnellste Methode (70 Stories von 10 Personen in 20 Minuten)
  - somit lässt sich jede Woche das gesamte Backlog neu bewerten
  - benötigt viel Platz (großer Raum, Flur)
- **Vorbereitung**
  - Jede Story auf ein DIN A4 Blatt (gut lesbar, inkl. Rang-Zahl)
  - Skala auf Boden auslegen (Fibonacci-Zahlen, Tiersymbole o.ä.)
  - jedes Teammitglied erhält eine etwa gleiche Anzahl an Story-Blättern
- **Ablauf**
  1. Wichtigste Regel: Keiner spricht, alle agieren schweigend!
  2. Man liest seine Stories und legt sie zu der Zahl, die seiner Meinung nach die Funktionsgröße der Story am besten repräsentiert (keine Zwischenwerte!).
  3. Sobald man fertig ist, sortiert man die Stories der anderen ggf. um.
  4. Der PO beobachtet genau und markiert Stories, die oft verschoben werden.

„Wir wollen aber Aufwand schätzen!“

- **Vermutung: man möchte Umfang der eigenen Aufgaben schätzen**
  - hieße, es ist schon zu Beginn klar, *wer* die Story bearbeitet
  - auch müsste klar sein, *was* konkret zu tun ist
  - oft werden Risikoaufschläge hinzugerechnet (also Arbeit + Störungen)
  - nicht zu beeinflussende Umgebungsfaktoren lassen sich nicht vorraussagen
- **Auf hohem Abstraktionsgrad (Projektebene) macht es Sinn**
  - Vergleich eines neuen Projektes mit einem Vorgängerprojekt
  - plus „Inflationskosten“ von 20%; Projekte werden über die Jahre teurer
  - man möchte auch wissen, was ein Haus kostet, bevor man es baut
- **Auf Detailebene aber nicht**
  - für eine Mini-Funktionalität braucht ein Team vielleicht zwei Tage (Dauer)
  - effektiv wird aber nur zwei Stunden (2h Aufwand) dran gearbeitet
  - oder aber acht Teammitglieder arbeiten je 14 Stunden (112h Aufwand)

Aufwandsschätzungen auf Detailebene sind zeitintensiv und sagen nichts über die Durchlaufzeit einer Story. Scrum ersetzt Planung durch Statistiken.

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten

Schätzen von Komplexität

- **Begriff der Komplexität ist eher ungeeignet**
  - Was ist Komplexität? Was heißt „schwierig“?
  - Inwiefern hängt Komplexität vom Kenntnisstand der Teammitglieder ab?
  - Etwas, das mir leicht fällt, muss meinem Kollegen nicht auch leicht fallen.
- **Versuch mit der „Things-That-Matter-Matrix“**

	Elektronik	Hardware	Software	GUI-Design	Field Tesiting
Story 1	S	S	L	M	S
Story 2	M	L	S	M	S
Story 3	M	M	S	L	L
Story 4	M	S	S	L	S
Story 5	L	M	M	M	XL

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten

Fazit

- **aus der Physik: *„Eine physikalische Größe ist eine quantitativ bestimmbare Eigenschaft eines physikalischen Objekts, Vorgangs oder Zustands.“***
- **Aufwands- und Komplexitätsschätzung beziehen immer den Faktor Mensch mit ein**

Das Schätzen von Funktionsumfängen kommt der Forderung nach Quantität am nächsten.

Methode: Planning Poker

- **Eigenschaften**
  - Team kommt zu einem gemeinsamen Verständnis jeder Story (Sicherheit)
  - stark unterschiedliche Auffassungen der Größe werden aufgedeckt
  - oft entstehen lange Diskussionen darüber, worum es in der Story geht
  - dauert lange (im Vgl. zu Magic Estimation); 10~15 Stories in einer Stunde
  - nach max. einer Stunde abbrechen (ist anstrengend)
- **Ablauf**
  1. PO beschreibt die Funktionalität der User Story
  2. Das Team kommt zu einem gemeinsamen Verständnis der Story
  3. Jeder legt eine Karte vor sich mit dem nach seiner Meinung gültigen Wert
  4. Alle decken gleichzeitig um (niemand wird beeinflusst)
  5. Haben alle den selben Wert, wird dieser für die Story notiert.
  6. Ansonsten diskutieren der mit dem höchsten und niedrigsten Wert. Danach decken alle neu auf.
- **Variante „Knobeln“**
  - Statt Karten einfach Null bis fünf Finger benutzen für 0, 1, 2, 3, 5, 8

Methode: Team Estimation Game

- **Eigenschaften**
  - nicht so schnell wie Magic Estimation, da sequentiell
  - Chance, die Einschätzung zu erklären → selbst-kalibrierend
  - kommt ohne Skala aus
- **Ablauf**
  1. Alle User Stories liegen auf einem Tisch.
  2. Ein Teammitglied nimmt eine Story, liest sie vor und pinnt sie an die Wand.
  3. Das nächste Teammitglied nimmt die nächste Story, liest sie vor
     - gleich viel Funktionalität → neben eine bereits hängende hängen
     - weniger Funktionalität → über eine bereits hängende hängen
     - mehr Funktionaltät → unter eine bereits hängende hängen
     - Umhängen einer vorhanden Karte (mit kurzer Erklärung, keine Diskussion!)
     - Aussetzen
  4. Product Owner markiert Stories, die oft umgehängt werden

Estimation Meeting

- **Rahmen**
  - mindestens einmal, besser zweimal pro Sprint
  - maximal 35 Minuten lang
  - gesamtes Entwicklungsteam nimmt teil
- **Ablauf**
  1. Magic Estimation durchführen
  2. Betrachtet werden dann zunächst die zehn mit dem höchsten Rang
  3. Unklare Stories (die markierten „Springer“) werden besprochen.
  4. Zu große Stories werden aufgeteilt.
  5. Termin für das nächste Estimation Meeting wird festgelegt.
- **Ergebnis**
  - Verständnis der Stories wird früh entwickelt
  - erste Lösungsansätze werden diskutiert
  - die Stories oben im Backlog sind klein genug für den nächsten Sprint

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten

Velocity – Schätzen überflüssig machen

- **Schätzung durch berechenbare Wahrscheinlichkeit ersetzen**
- **Paradigmenwechsel: Trend**
  - statt „wie lange braucht die Implementierung einer bestimmten Funktionalität?“
  - „Wie lange braucht das Entwickeln einer Funktionatlität in der Regel?“
- **Konzept bekannt aus der Fertigung; in Entwicklung aber**
  - Rate des Eintreffens neuer Funktionalitäten nicht bekannt
  - nicht bekannt, wie viel Zeit die Entwicklung einer *neuen* Funktionalität benötigt
  - → Entwicklung hat damit eine hohe Varianz
  - Ziel der Fertigung: Varianz reduzieren und Stabilität erzeugen
- **in der Entwicklung wollen wir aber Varianz zulassen und erreichen dadurch nie Stabilität**
- **→ Ausweg: Cycle-Time**

Quelle: Boris Gloger 2014 - Wie schätzt man in agilen Projekten