Inhalt der Vorlesung

- Einführung
- Kommunikation
- Software Qualität
- Vorgehensmodelle
- Requirements Engineering
- Software Architektur und – Design
- Konfiguration Management

Vorgehensmodelle

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

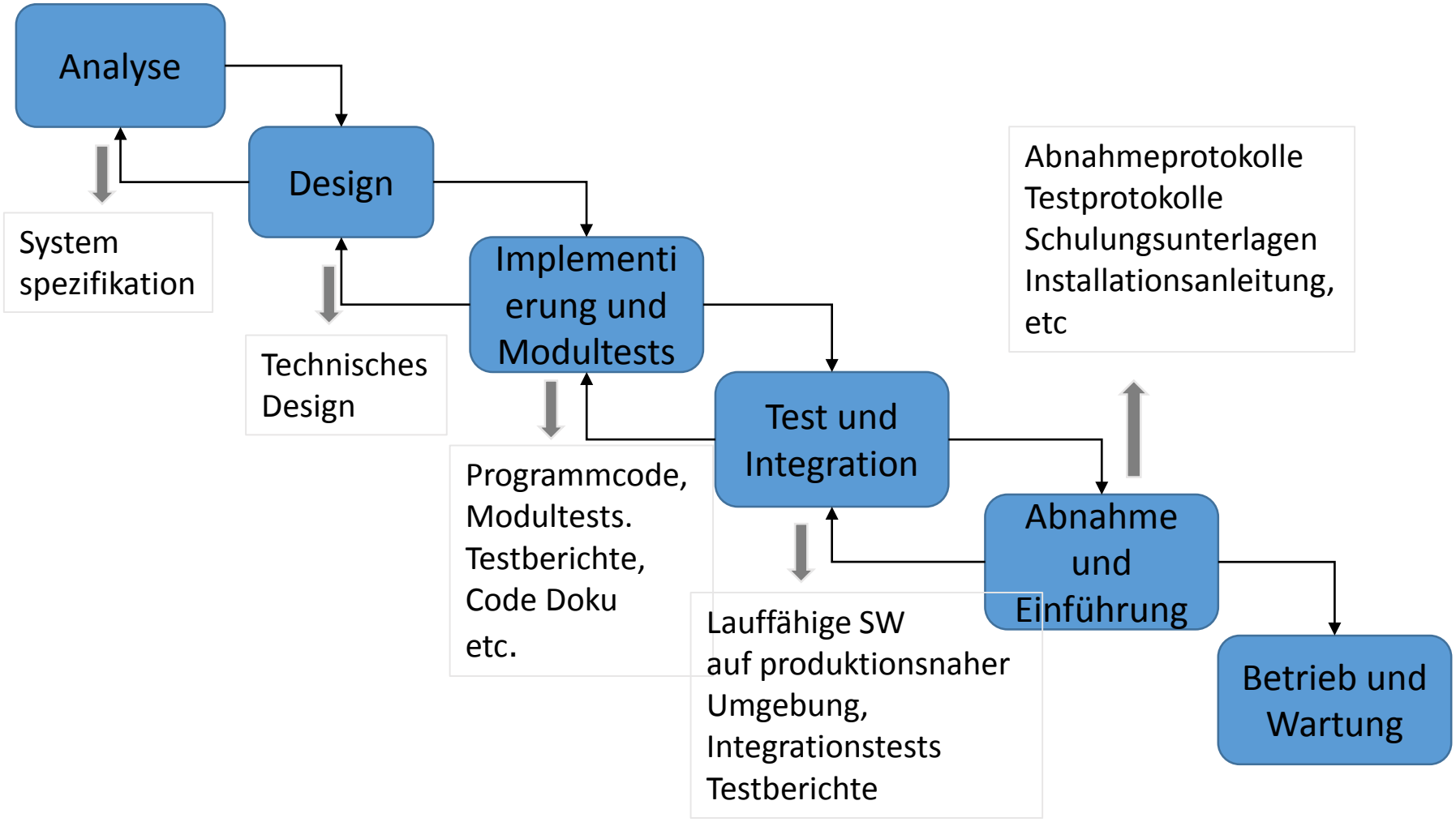
Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Tätigkeiten und Artefakte in einem SW Projekt



Vorgehensmodelle

Ein Vorgehensmodell beschreibt:

- Menge von Aktivitäten
- Produkte
- Rollen
- Vor- und Nachbedingungen

Es gibt kein ideales und für alle Projekte passendes Vorgehensmodell.

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Wichtige Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Wichtige Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Code and Fix

Wie war das früher?

Code and Fix:

Code and Fix bezeichnet ein Vorgehen, bei der Codierung oder Korrektur im Wechsel mit Ad-hoc-Tests die einzigen bewusst ausgeführten Tätigkeiten der Software-Entwicklung sind.

Quelle: Ludewig, Lichter: Software Engineering, 3. Auflage, dpunkt.verlag

Code and Fix

Pros

- Entspricht dem Bedürfnis, sofort loszulegen
- Im Idealfall schnell lauffähige Software
- Einfach

Cons

- Schlecht planbar
- Arbeiten schlecht zu verteilen
- Anforderungen sind nicht explizit und daher schwer zu erfüllen.
- Keine Soll-Vorgaben für Tests
- Ergebnis oft schwer wartbare Programme
- Keine oder schlechte Dokumentation

Code and Fix

Fazit für Code and Fix:

- Teuer
- Liefert schlechte Qualität
- Wird trotzdem immer noch angewendet

➔ Bessere Möglichkeiten?

Wichtige Vorgehensmodelle

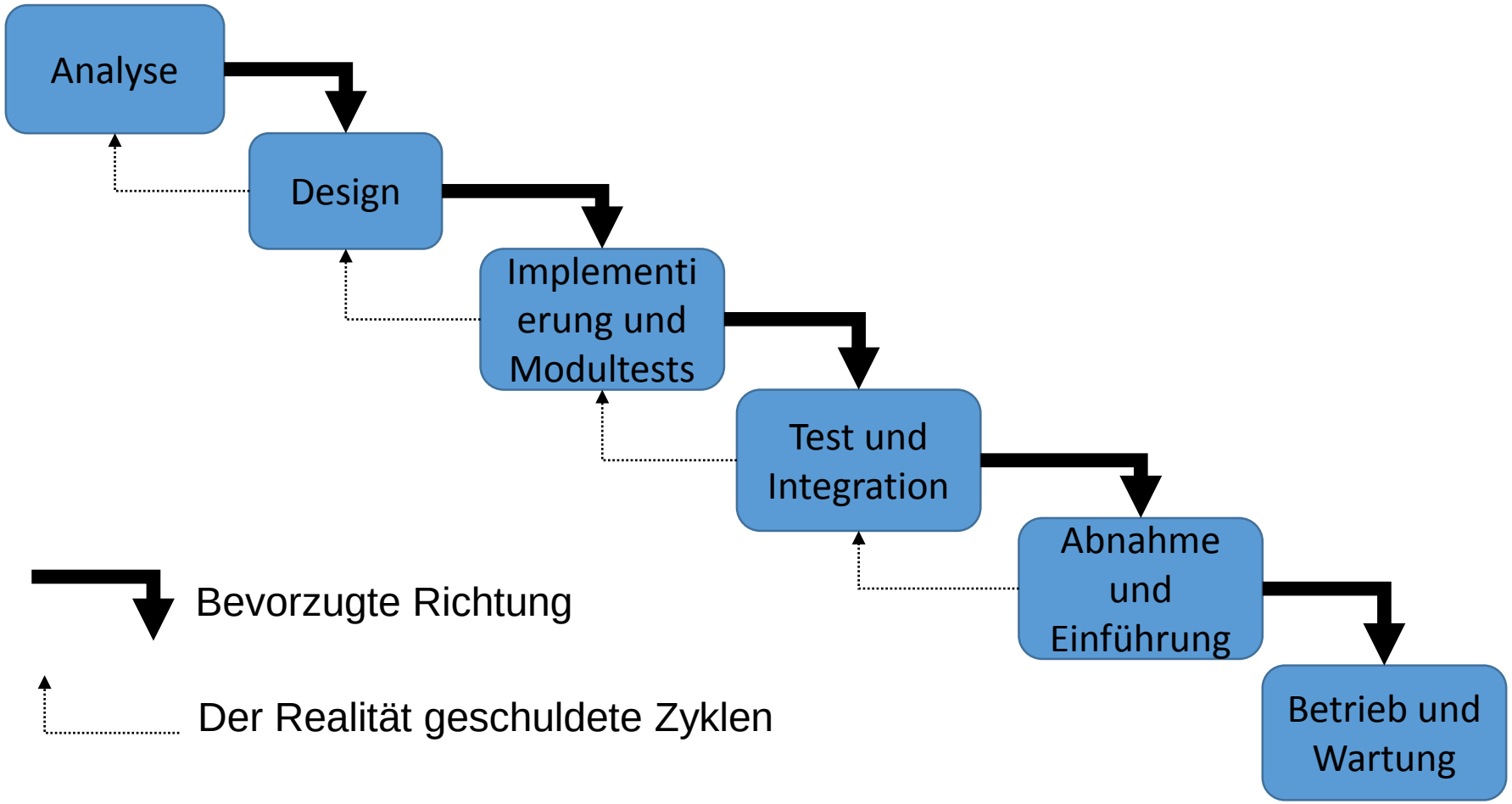
- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Wasserfallmodell

waterfall model — A model of the software development process in which the constituent activities, typically a concept phase, requirements phase, design phase, implementation phase, test phase, and installation and checkout phase, are performed in that order, possibly with overlap but with little or no iteration.

IEEE Std 610.12 (1990)

Wasserfallmodell



Wasserfallmodell

- Wegen der Kaskade von einer Phase zur anderen so genannt.
- Plangesteuerter Prozess – alle Aktivitäten werden inhaltlich und zeitlich geplant.
- Aus jeder Phase gehen Artefakte hervor, die abgenommen werden und für die nächste Phase als Input verwendet werden.
- Bsp: ITPM (BMW Vorgehensmodell)

Wasserfallmodell

Pro

- Klare Abgrenzung der Phasen
- Einfach zu planen und zu kontrollieren
- Effizient bei stabilen Anforderungen

Con

- Phasenabgrenzung oft unrealistisch
- Sequenz oft unrealistisch (Rückschritte nötig)
- Unflexibel
- Anforderungen normalerweise nicht hinreichend stabil
- System erst sehr spät verfügbar

Wichtige Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Prototyping

Wenn zunächst ein System entsteht, das dem geplanten in irgendeiner Weise ähnlich ist, aber wesentliche Anforderungen **nicht** erfüllt, spricht man von Prototyping.

Sinn des Prototyping:

- Überprüfung und Detaillierung der Anforderungen (speziell der GUI)
- Überprüfung des Lösungsansatzes

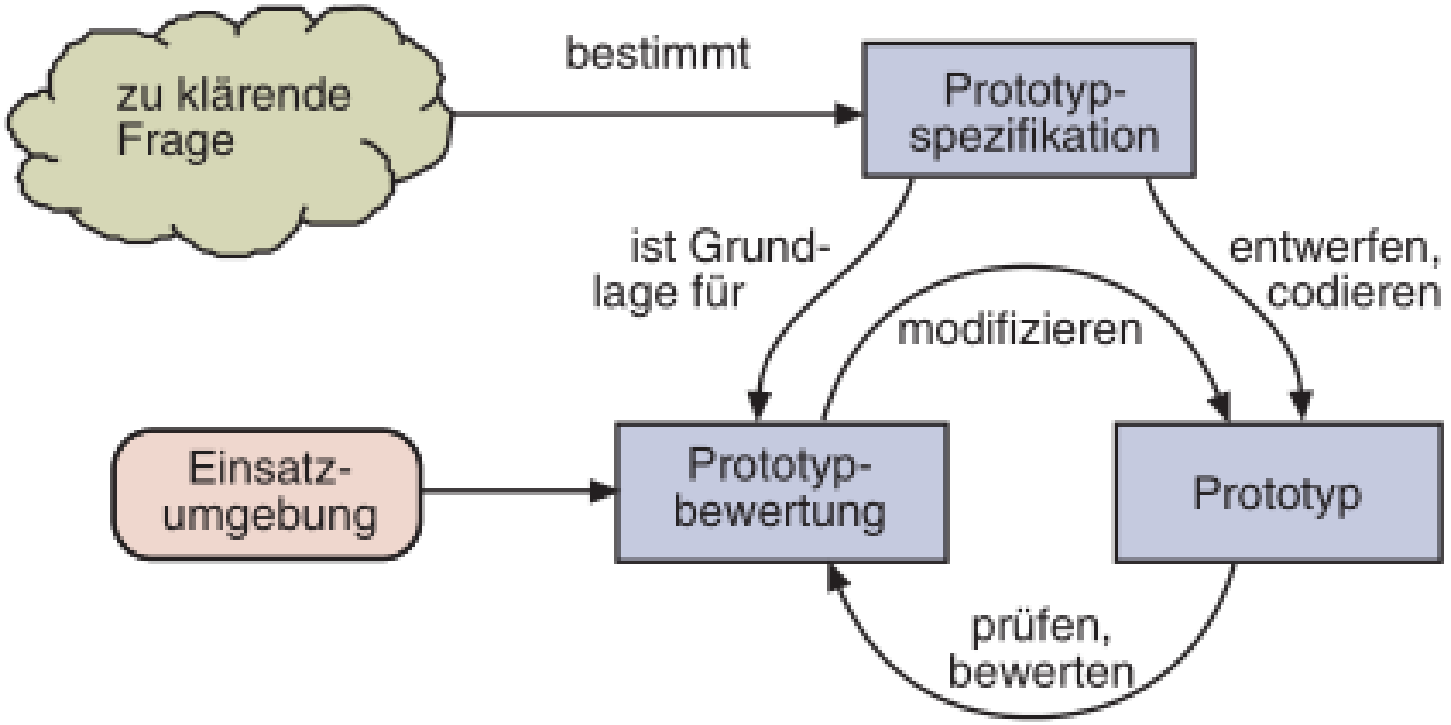
Prototyping

Um Prototypen zielgerichtet entwickeln zu können, sind zunächst folgende Fragen zu klären:

- Welchen Zweck hat der Prototyp, also welche offenen Fragen sollen beantwortet werden?
- Wer entwickelt den Prototyp und wer beurteilt ihn?
- Wie viel Aufwand soll/darf in den Prototyp fließen?

Prototyping

Allgemeine Vorgehensweise:



Quelle des Bilds: Ludewig, Lichter: Software Engineering, 3. Auflage, dpunkt.verlag

Prototyping

Spezielle Prototypen:

- **Demonstrationsprototyp**
  zeigt prinzipielle Einsatzmöglichkeiten. Nützlich für Akquise oder in der Startphase eines Projekts.
- **Funktionaler Prototyp**
  Ausschnitte der GUI und der Funktionalität helfen bei der Anforderungsanalyse.
- **Labormuster**
  technische Machbarkeitsstudien
- **Pilotsystem**
  abgeschlossener Teil des Systems für begrenzte Nutzergruppe

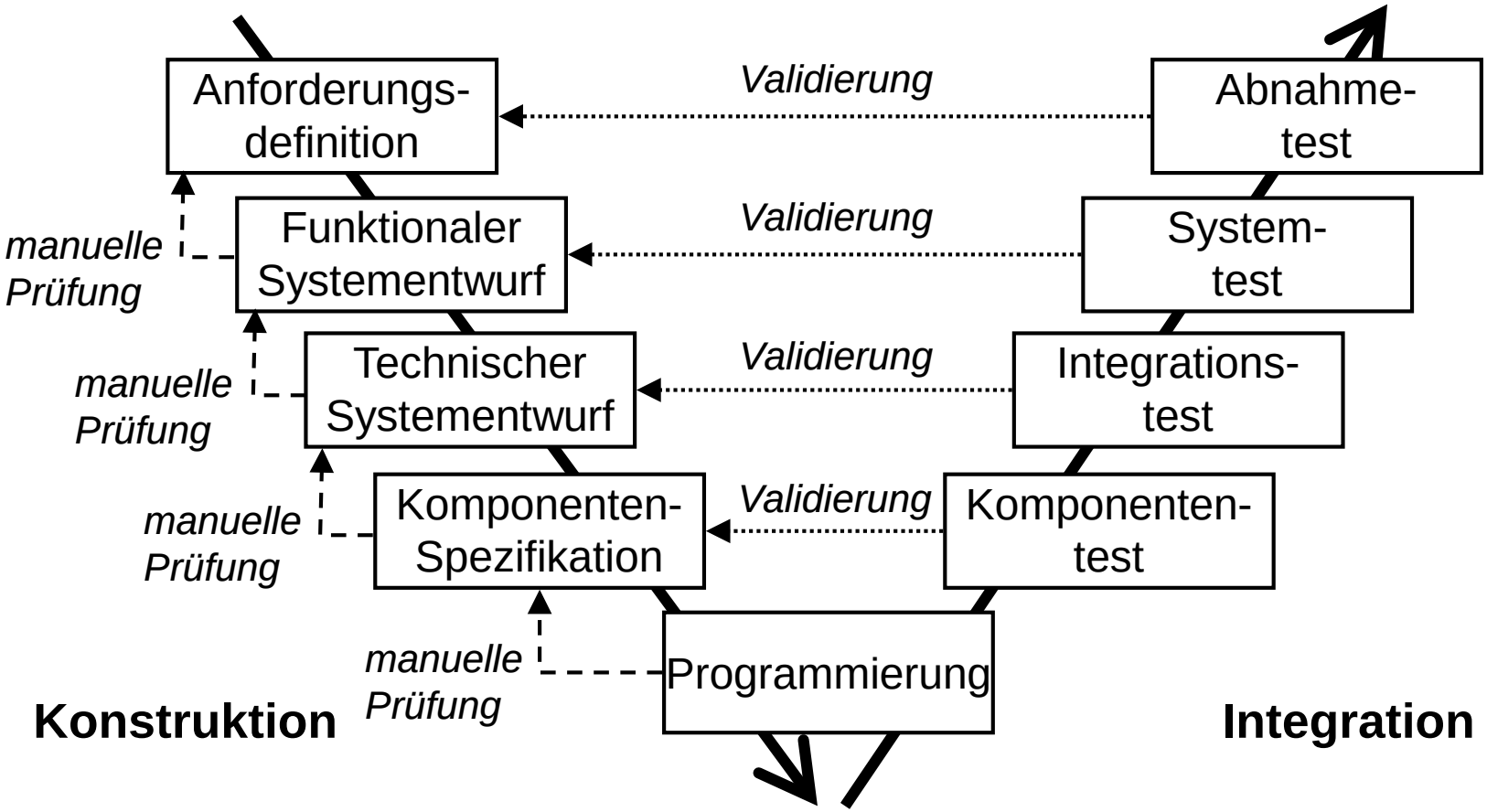
Wichtige Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

V-Modell

- Der Software Entwicklungsprozess ist in Phasen eingeteilt, die mit zunehmendem Projektfortschritt immer mehr ins Detail gehen.
- Die Qualitatssicherungsschritte sind ebenfalls in Phasen eingeteilt, die mit zunehmendem Projektfortschritt vom Detail zum Abstrakten gehen.
- Jedem Entwicklungsschritt entspricht ein Qualitätssicherungsschritt.

Allgemeines V-Modell



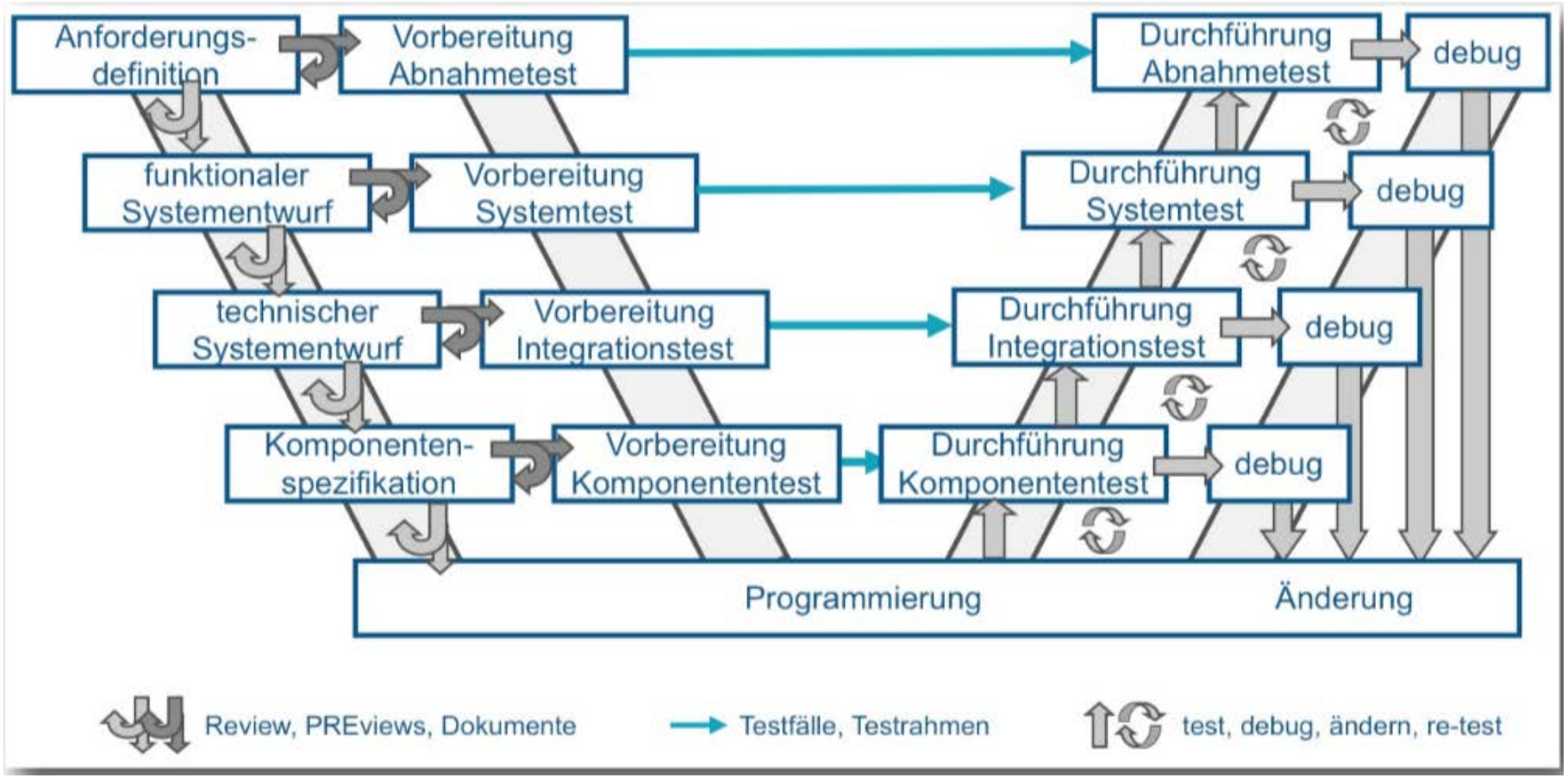
Quelle: Kleuker: Grundkurs Software-Engineering mit UML, Springer Vieweg

V-Modell

- In verschiedenen Darstellungen werden die Anzahl der Phasen unterschiedlich dargestellt.
- Aber immer: 1:1 Relation zwischen Entwurf und Test Stufe.

- Das allgemeine V-Modell ist die Grundlage für bestimmte Entwicklungsstandards, wie z.B. das V-Modell der Bundesrepublik Deutschland. (http://www.cio.bund.de/Web/DE/Architekturen-und-Standards/V-Modell-XT/vmodell_xt_node.html)

W-Modell als Erweiterung des V-Modells



Wichtige Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Rational Unified Process

- Der Rational Unified Process ist ein iteratives, Use-Case-getriebenes und architekturzentriertes Prozessmodell.
- RUP baut auf die Verwendung von UML auf.

- Wird von IBM vermarktet.
- Anwendung erfordert eine Lizenz.

- Hier nicht mehr näher besprochen.

Wichtige Vorgehensmodelle

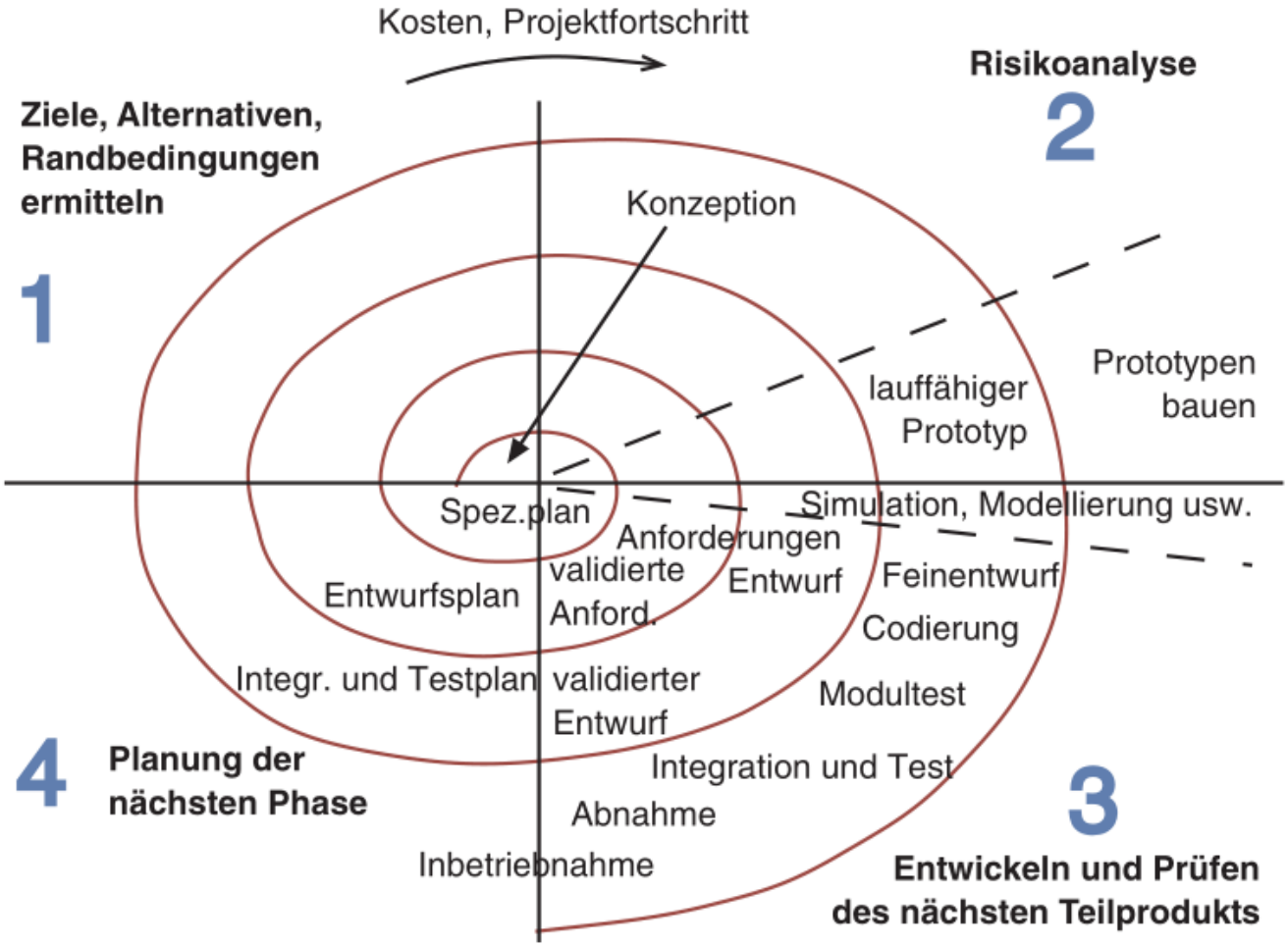
- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Spiralmodell (Böhm 1988)

Idee:
Vorgehen in Zyklen:

1. Festlegung von Zielen, Identifikation von Alternativen und Beschreibung von Rahmenbedingungen
2. Evaluierung der Alternativen und das Erkennen, Abschätzen und Reduzieren von Risiken, z. B. durch Analysen, Simulationen oder Prototyping.
3. Realisierung und Überprüfung des Zwischenprodukts
4. Planung des nächsten Zyklus der Projektfortsetzung.

Spiralmodell



Spiralmodell

- Fokus des Spiralmodells liegt auf dem Risikomanagement. Es wird immer versucht, das größte Risiko zu identifizieren und zu beseitigen. Das Projekt gilt als gescheitert, wenn das nicht möglich ist.
- Das Modell ist als inkrementelles Vorgehen zu verstehen.
- Das Modell ist sehr generisch, kann und muss also für den jeweiligen Einzelfall betrachtet werden

Wichtige Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Prototyping
- V- Modell
- Rational Unified Process (RUP)
- Spiralmodell
- Agile Methoden → nächstes Kapitel

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Kritik an klassischen Vorgehensmodellen

- Es müssen viele Dokumente erzeugt und gepflegt werden.
- Eigene Wissenschaft Modelle wie V-Modelle und RUP zu verstehen und zurecht zu schneidern.
- Prozessbeschreibungen hemmen Kreativität.
- Anpassung an neue Randbedingungen, z. B. neue Technologien (Web-Services) in Prozessen und benutzten Werkzeugen ist extrem aufwändig.

Alternativer Ansatz: „Menschen machen Projekte erfolgreich, traue den Menschen“

=> agile Prozesse

Agiles Manifest (Februar 2001)

http://www.agilemanifesto.org/iso/de/

Wir erschließen bessere Wege, Software zu entwickeln,
indem wir es selbst tun und anderen dabei helfen.
Durch diese Tätigkeit haben wir diese Werte zu schätzen gelernt:

- **Individuen und Interaktionen mehr als Prozesse und Werkzeuge**
- **Funktionierende Software mehr als umfassende Dokumentation**
- **Zusammenarbeit mit dem Kunden mehr als Vertragsverhandlung**
- **Reagieren auf Veränderung mehr als das Befolgen eines Plans**

Das heißt, obwohl wir die Werte auf der rechten Seite wichtig finden,
schätzen wir die Werte auf der linken Seite höher ein.

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas
www.agileAlliance.org

Prinzipien agiler Software Entwicklung 1/2

Prinzipien hinter dem Agilen Manifest

1. Unsere höchste Priorität ist es, den Kunden durch frühe und kontinuierliche Auslieferung wertvoller Software zufrieden zu stellen.
2. Heisse Anforderungsänderungen selbst spät in der Entwicklung willkommen. Agile Prozesse nutzen Veränderungen zum Wettbewerbsvorteil des Kunden.
3. Liefere funktionierende Software regelmäßig innerhalb weniger Wochen oder Monate und bevorzuge dabei die kürzere Zeitspanne.
4. Fachexperten und Entwickler müssen während des Projektes täglich zusammenarbeiten.
5. Errichte Projekte rund um motivierte Individuen. Gib ihnen das Umfeld und die Unterstützung, die sie benötigen und vertraue darauf, dass sie die Aufgabe erledigen.

Prinzipien agiler Software Entwicklung 2/2

6. Die effizienteste und effektivste Methode, Informationen an und innerhalb eines Entwicklungsteams zu übermitteln, ist im Gespräch von Angesicht zu Angesicht.
7. Funktionierende Software ist das wichtigste Fortschrittsmaß.
8. Agile Prozesse fördern nachhaltige Entwicklung. Die Auftraggeber, Entwickler und Benutzer sollten ein gleichmäßiges Tempo auf unbegrenzte Zeit halten können.
9. Ständiges Augenmerk auf technische Exzellenz und gutes Design fördert Agilität.
10. Einfachheit -- die Kunst, die Menge nicht getaner Arbeit zu maximieren -- ist essenziell.
11. Die besten Architekturen, Anforderungen und Entwürfe entstehen durch selbstorganisierte Teams.
12. In regelmäßigen Abständen reflektiert das Team, wie es effektiver werden kann und passt sein Verhalten entsprechend an.

Einschub

Zitat Andy Hunt:

> *"Agile methods ask practitioners to think, and frankly, that's a hard sell."*

Aus: http://blog.toolshed.com/2015/05/the-failure-of-agile.html

(Beachten Sie den Titel!)

Einschub

Was macht agile Methoden schwierig?

- Bedarf an Regeln
- Angst, sich zu blamieren
- „Save your ass“ Mentalität
- …

Agile Prozesse

Das Agile Manifest findet Niederschlag in den agilen Prozessen:

- Extreme Programming
- Inkrementell iterative Prozesse
- Scrum
- Kanban

Agile Methoden

Einige Methoden sind mehreren agilen Prozessen gemeinsam:

Bsp. für agile Methoden

- Paarprogrammierung
- Testgetriebene Entwicklung
- ständige Refaktorierungen
- Story-Cards
- schnelle Codereviews

XP als zeitweise populäres agiles Modell

Die 12 XP Praktiken im Zusammenspiel (Quelle: Kent Beck: Extreme Programming, Addison Wesley, 2000)

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Scrum als populäres Beispiel.

Offizieller **Scrum Guide** von Ken Schwaber und Jeff Sutherland unter
http://www.scrumguides.org/download.html

oder auf der e-learning Plattform

Im deutschsprachigen Raum einflussreiche Literatur:

B. Gloger; *Scrum*, Hanser, 2011

Gliederung

Motivation

Forrester: 60% aller IT Projekte sind nicht im Plan

Falsche Annahmen in IT Projekten:

- Die Anforderungen sind klar.
- Die Anforderungen sind stabil.
- Der Entwicklungsprozess ist vorhersehbar.

Scrum

Ansatz von Scrum: empirisch, inkrementell, iterativ

Prinzipien:

- (Zerlegung)
- Transparenz
- Überprüfung
- Anpassung

Gliederung

Idee: In kurzen Zyklen releasefähige Software auszuliefern

Rollen: Product Owner, Scrum Master, Entwicklungs Team, *(nach Gloger zusätzliche Rollen:* Customer, Manager, User)

Meetings: Sprint Planning (1&2), Daily Scrum, Sprint Review, Sprint Retrospective, (nach Gloger zusätzlich: Estimation Meeting)

Artefakte: Product Backlog, Sprint Backlog, Produkt Inkrement (nach Gloger zusätzlich: Vision, Sprint Goal, Tasks, Releaseplan, Impediment Backlog)

Definition of Done

Einführung in Scrum – Scrum Teams

- Scrum Teams sind **selbstorganisierend** und **interdisziplinär**.
- Selbstorganisierende Teams entscheiden selbst, wie sie ihre Arbeit am besten erledigen, anstatt dieses durch andere Personen außerhalb des Teams vorgegeben zu bekommen.
- Interdisziplinäre Teams verfügen über alle Kompetenzen, die erforderlich sind, um die Arbeit zu erledigen, ohne dabei von Personen außerhalb des Entwicklungsteams abhängig zu sein.
- Das Team-Modell in Scrum wurde konzipiert, um **Flexibilität**, **Kreativität** und **Produktivität** zu optimieren.

Vier Arbeits Prinzipien bei Scrum

- Selbstorganisation

- Pull Prinzip

- Timebox

- Nutzbare Funktionalität

Scrum – die Rollen

Scrum – der Product Owner

Product Owner:
Der Product Owner ist für die Wertmaximierung des Produkts sowie die Arbeit des Entwicklungsteams verantwortlich. Wie dies geschieht, kann je nach Organisation, Scrum Team und Einzelpersonen stark variieren. Der Product Owner ist die einzige Person, die für das Management des Product Backlogs verantwortlich ist.

Scrum – das Entwicklungsteam

Das **Entwicklungsteam** besteht aus Profis, die am Ende eines jeden Sprints ein fertiges Inkrement übergeben, welches potentiell auslieferbar ist. Nur Mitglieder der Entwicklungsteams erstellen das Produkt-Inkrement.

Scrum – der Scrummaster

Der **Scrum Master** ist für das Verständnis und die Durchführung von Scrum verantwortlich. Er tut

dies, indem er dafür sorgt, dass das Scrum Team die Theorie, Praktiken und Regeln von Scrum

einhält.

https://www.youtube.com/watch?v=oheekef7oJk

Scrum – die weiteren Rollen (nach Gloger)

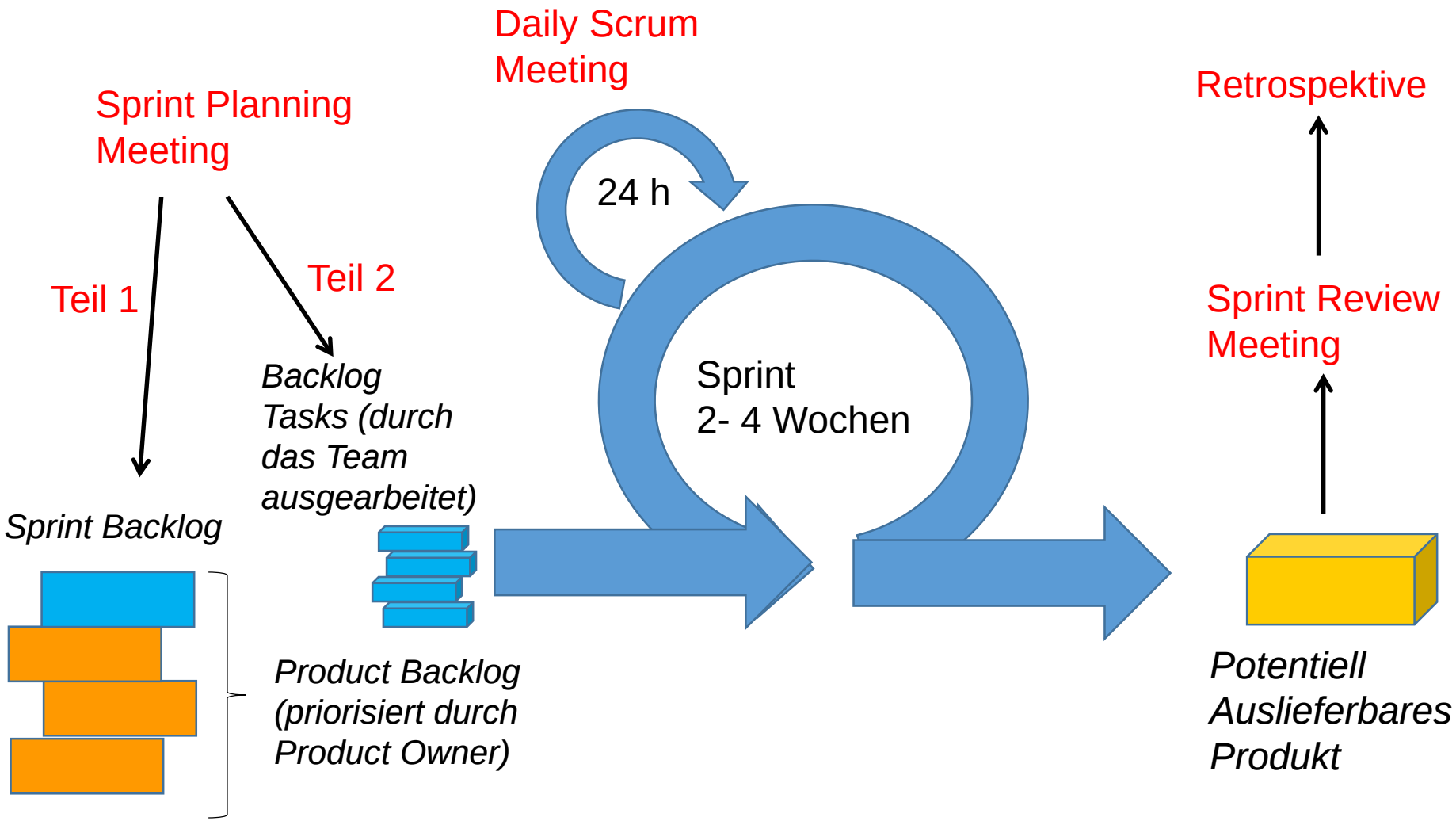
- **Customer**
  Auftraggeber. Finanziert das Projekt.

- **Manager**
  stellt Ressourcen und Richtlinien innerhalb der Organisation bereit.

- **User**
  wesentliche Informationsquelle für das Scrum Team.

Scrum – der Prozess

Scrum Prozess



Scrum Ereignisse

Der Sprint als zentraler Container für die Arbeit:
Ein Sprint beinhaltet und umfasst das Sprint Planning, die Daily Scrums, die Entwicklungsarbeit, das Sprint Review und die Sprint Retrospektive.

Während des Sprints:

- werden keine Änderungen vorgenommen, die das Sprint-Ziel gefährden,
- wird der Qualitätsanspruch nicht geschmälert, und
- der Anforderungsumfang kann zwischen Product Owner und Entwicklungsteam geklärt und neu ausgehandelt werden, wenn sich neue Erkenntnisse ergeben haben.

Die Meetings - Sprint Planning

Im **Sprint Planning** (manchmal geteilt in zwei Meetings) wird die Arbeit für den kommenden Sprint geplant. Dieser Plan entsteht durch die gemeinschaftliche Arbeit des gesamten Scrum Teams.
Das Sprint Planning beantwortet die folgenden Fragen:

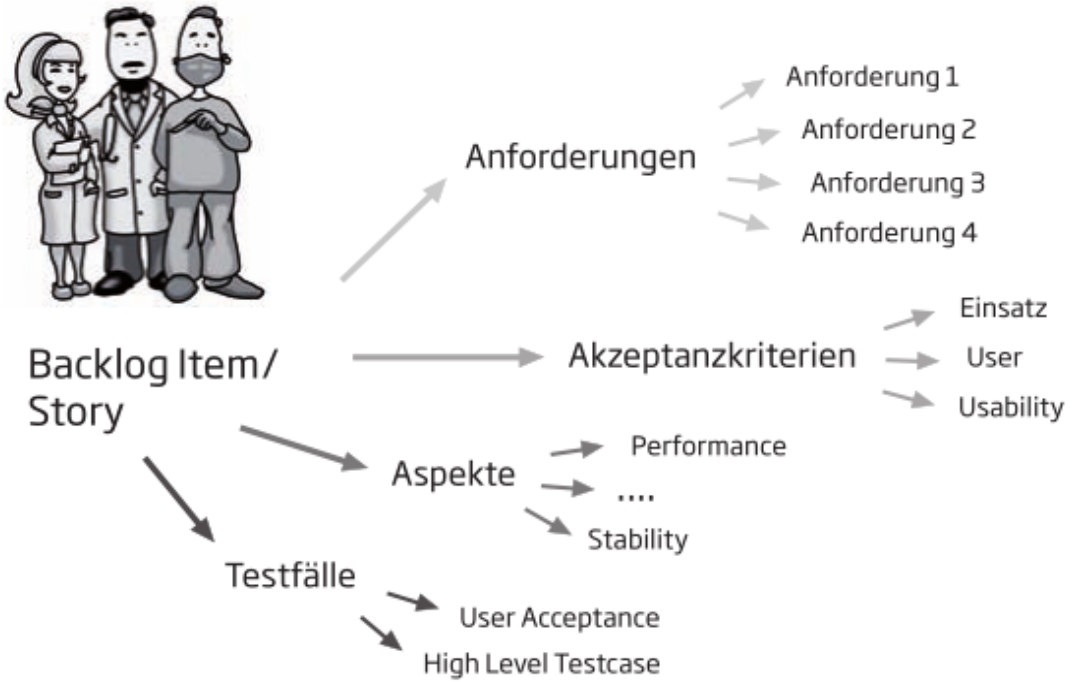
- Was ist in dem Produkt-Inkrement des kommenden Sprints enthalten?
- Wie wird die für die Lieferung des Produkt-Inkrements erforderliche Arbeit erreicht?

Zweigeteiltes Sprint Planning

Sprint Planning 1:

entspricht einem **Anforderungs Workshop**,

Ergebnis: Selected Product Backlog

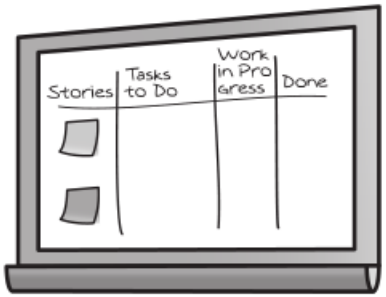


Zweigeteiltes Sprint Planning

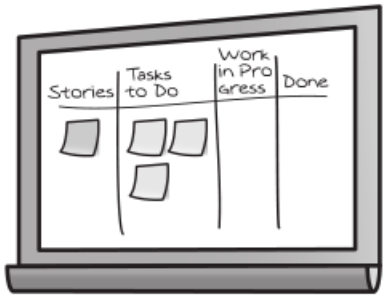
Sprint Planning 2:

Entspricht **Design Workshop**,

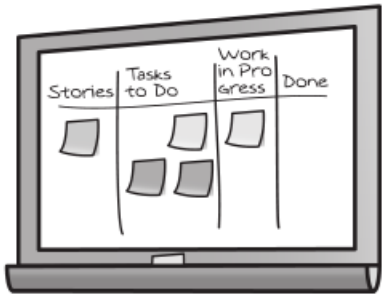
Ergebnis: Sprint Backlog: Liste der Tasks für den Sprint am Taskboard



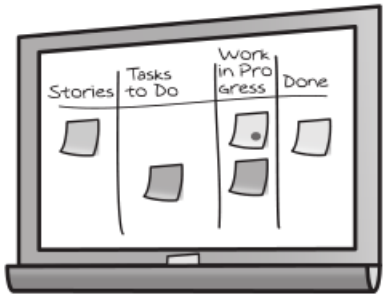
1. Selected Product Backlog am Taskboard unter „Stories"



2. Stories aufgeteilt bearbeitbare 1-Tages „Tasks to Do"



3. „Work in Progress" zeigt Tasks, an denen jetzt gearbeitet wird, evtl. verändern sich Tasks, sind zerlegt



4. Tasks sind „done" oder sind mit Punkten (Impediments) markiert, wenn sie nicht fertig wurden

Meetings - Daily Scrum

Jedes Team Mitglied beantwortet drei Fragen:

1. Was habe ich gestern gemacht?
2. Was werde ich heute tun?
3. Was hindert mich bei meiner Arbeit?

Scrum Board

Estimation Meeting

- PO und Team schätzt mindestens einmal pro Sprint das Backlog, priorisiert neu und aktualisiert ggfs den Releaseplan.

- Zum Vorgehen beim Schätzen siehe unten.

Meetings – Sprint Review

Am Ende eines Sprints wird ein **Sprint Review** abgehalten, um das [Produkt-]Inkrement zu überprüfen und das Product Backlog bei Bedarf anzupassen. Während des Sprint Reviews beschäftigen sich das Scrum Team und die Stakeholder gemeinsam mit den Ergebnissen des Sprints.

Meetings - Retrospektive

Die Sprint Retrospektive bietet dem Scrum Team die Gelegenheit, sich selbst zu überprüfen und einen Verbesserungsplan für den kommenden Sprint zu erstellen.

Detail: Meetings – Estimation Meeting

Der Product Owner hat die Aufgabe, einen Release Plan zu erstellen. Dafür muss er Größe der Backlog Items und Durchsatz des Teams schätzen.

Im Estimation Meeting werden die Einträge des Product Backlogs geschätzt. Dafür existieren verschiedene Verfahren und verschiedene Kriterien, nach denen geschätzt werden kann.

Schätzen in Scrum

Aussage aus dem offiziellen Scrum Guide:

> Ein Product Backlog-Eintrag enthält als Attribute eine Beschreibung, die Reihenfolge, die Schätzung und den Wert.

Keine Aussagen über die Art der Schätzung

Schätzen in Scrum nach Gloger

Boris Gloger:

„Mein Vorschlag ist revolutionär.“

Gloger: „Die Größe bezeichnet den Grad des Verständnisses, welches das Team von dem Backlog Item, von der Funktionalität hat.“

Nach Gloger noch zu ermitteln:

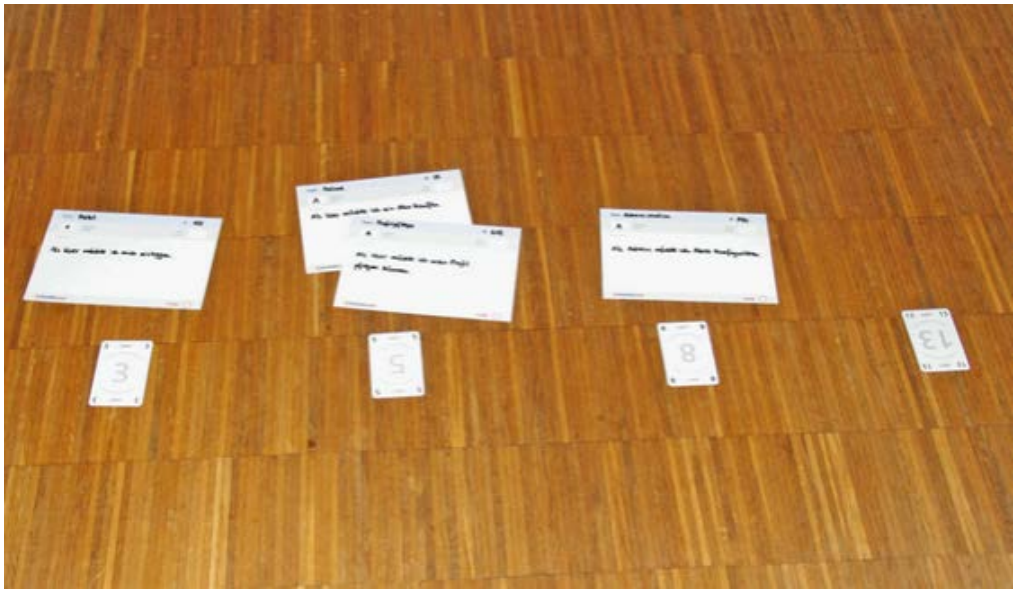
1. Referenz
2. Maßeinheit= Storypoints
3. Skala = Fibonacci Zahlen

Planning Poker

- Einigung auf Referenz Backlog Item
- Vorstellung eines Backlog Items
- Klärung offener Fragen
- Teammitglieder vergeben Storypoints
- Diskussion
- Wiederholung bis ein Ergebnis vorliegt

Magic Estimation

- Jedes Teammitglied erhält ausgedruckte BL Items.
- Die Items werden auf einer Skala angeordnet und dadurch geschätzt.
- Jedes Teammitglied kann die Items verschieben, bis die Lage stabil ist.



Einschätzung (Bulenda) der Glogerschen Schätzmethodiken

- Werden nicht so durchgeführt, wie von Gloger propagiert.
- Es wird implizit immer Aufwand geschätzt.
- Nicht widerspruchsfrei.
- Werden wieder zunehmend durch klassische Aufwandsschätzungen abgelöst.

Auftraggeber haben mit der Abgabe der Kontrolle zunehmend Probleme.

Siehe auch: http://www.mountaingoatsoftware.com/blog/story-points-are-still-about-effort

Scrum – die Artefakte

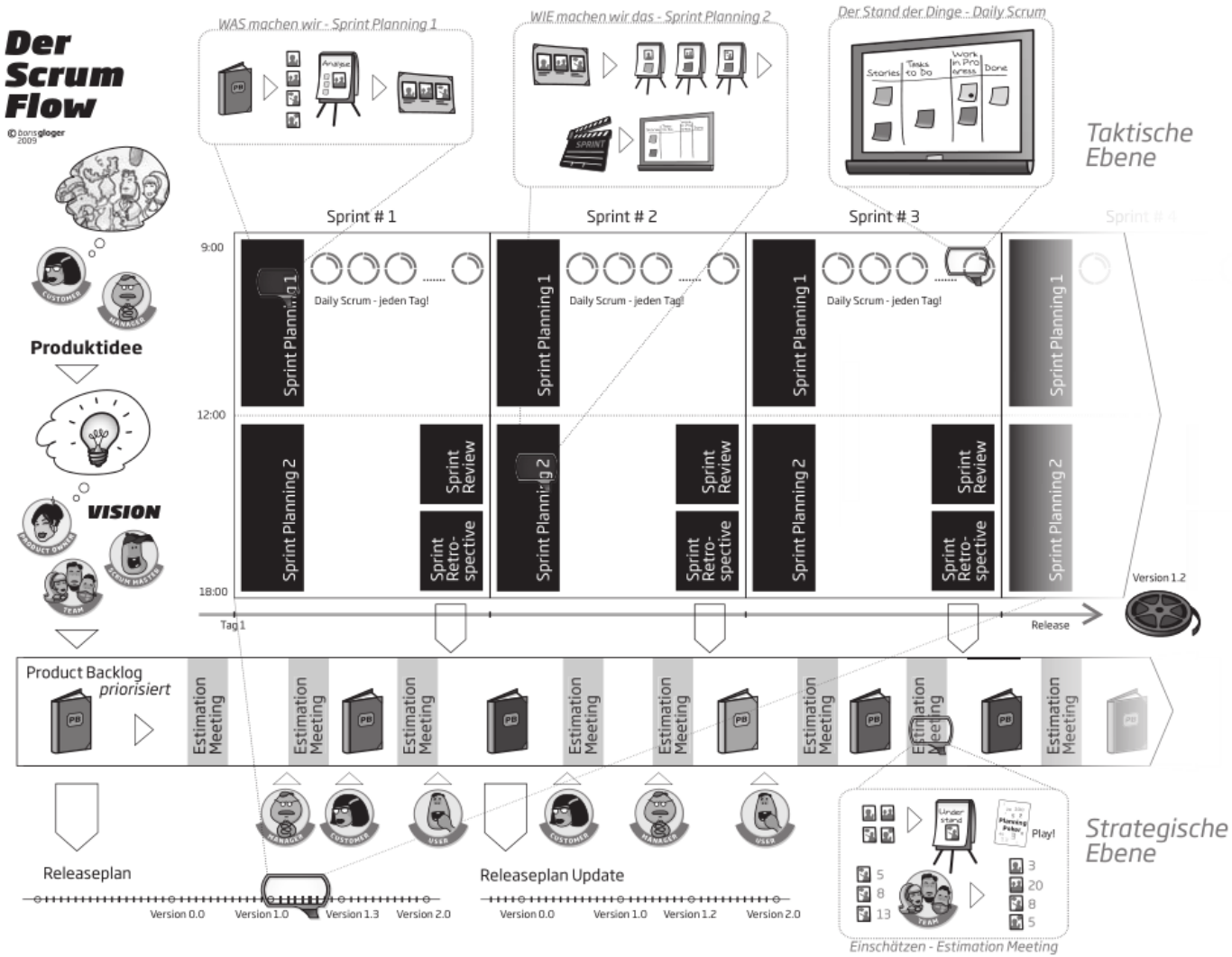
Scrum – die Artefakte

- **Product Backlog**
  Das Product Backlog ist eine geordnete Liste von allem, was in dem Produkt enthalten sein kann.
- **Sprint Backlog**
  Das Sprint Backlog ist die Menge der für den Sprint ausgewählten Product Backlog-Einträge.
- **Sprint Ziel:** Die ausgewählten Product Backlog-Einträge bilden eine zusammenhängende Funktionalität, die als Sprint-Ziel angesehen werden kann
- **Produkt Inkrement:** Das Inkrement ist das Ergebnis aus allen in einem Sprint fertiggestellten Product Backlog-Einträgen

Weitere Artefakte – Nach Gloger:

- Vision, Tasks, Releaseplan, Impediment Backlog

Scrum Prozess (Quelle: Gloger: Scrum)



Definition of Done

Es müssen alle verstehen, was „Done“ bedeutet, sobald ein Product Backlog-Eintrag oder ein Produkt-Inkrement als „Done“ bezeichnet wird. Alle Teammitglieder müssen ein gemeinsames Verständnis davon haben wann Arbeit fertig ist, um Transparenz zu gewährleisten.

Dies erfolgt durch die **Definition of Done** des Scrum Teams.

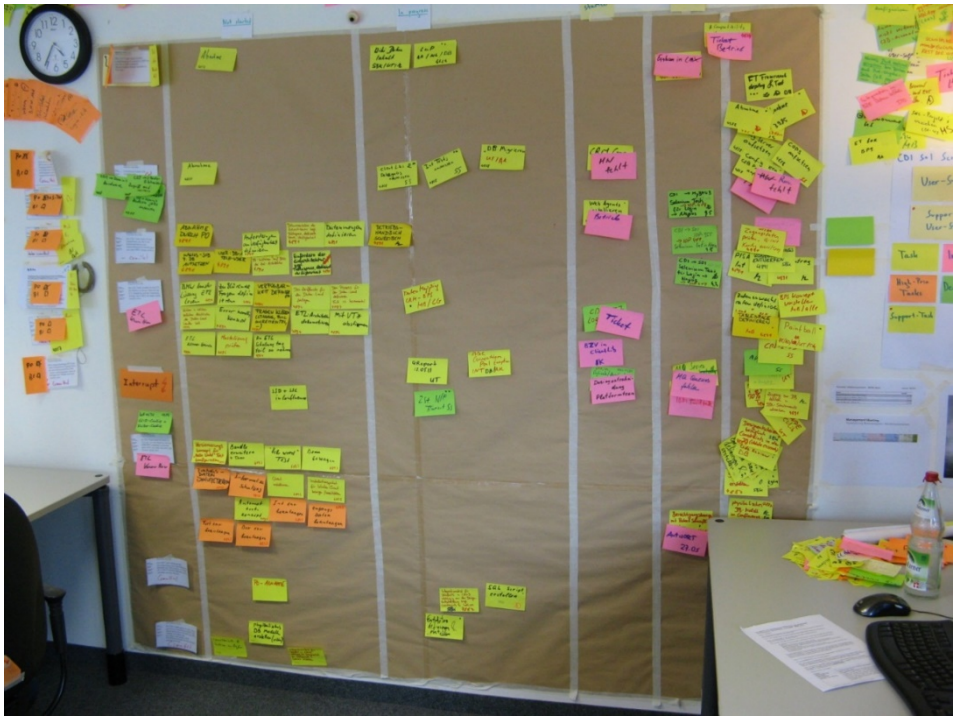
Scrum - Reporting

Der Projektfortschritt und die Prognosen beruhen auf verschiedenen Berichten:

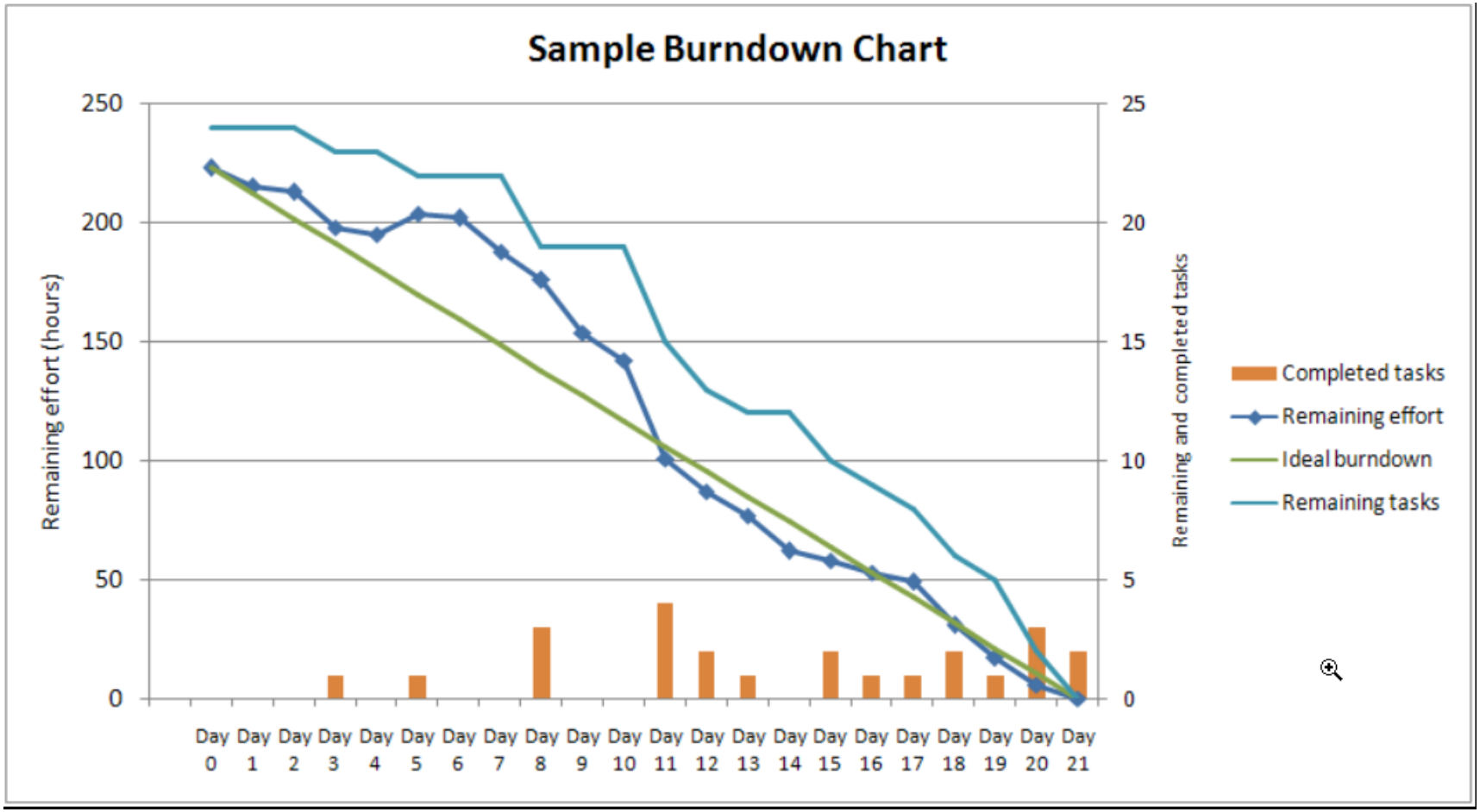
1. Taskboard (= Scrum Board)
2. Verschiedene Formen von Charts
3. Berechnung der Velocity

Task Board

- Täglicher Fortschritt
- Impediments sichtbar



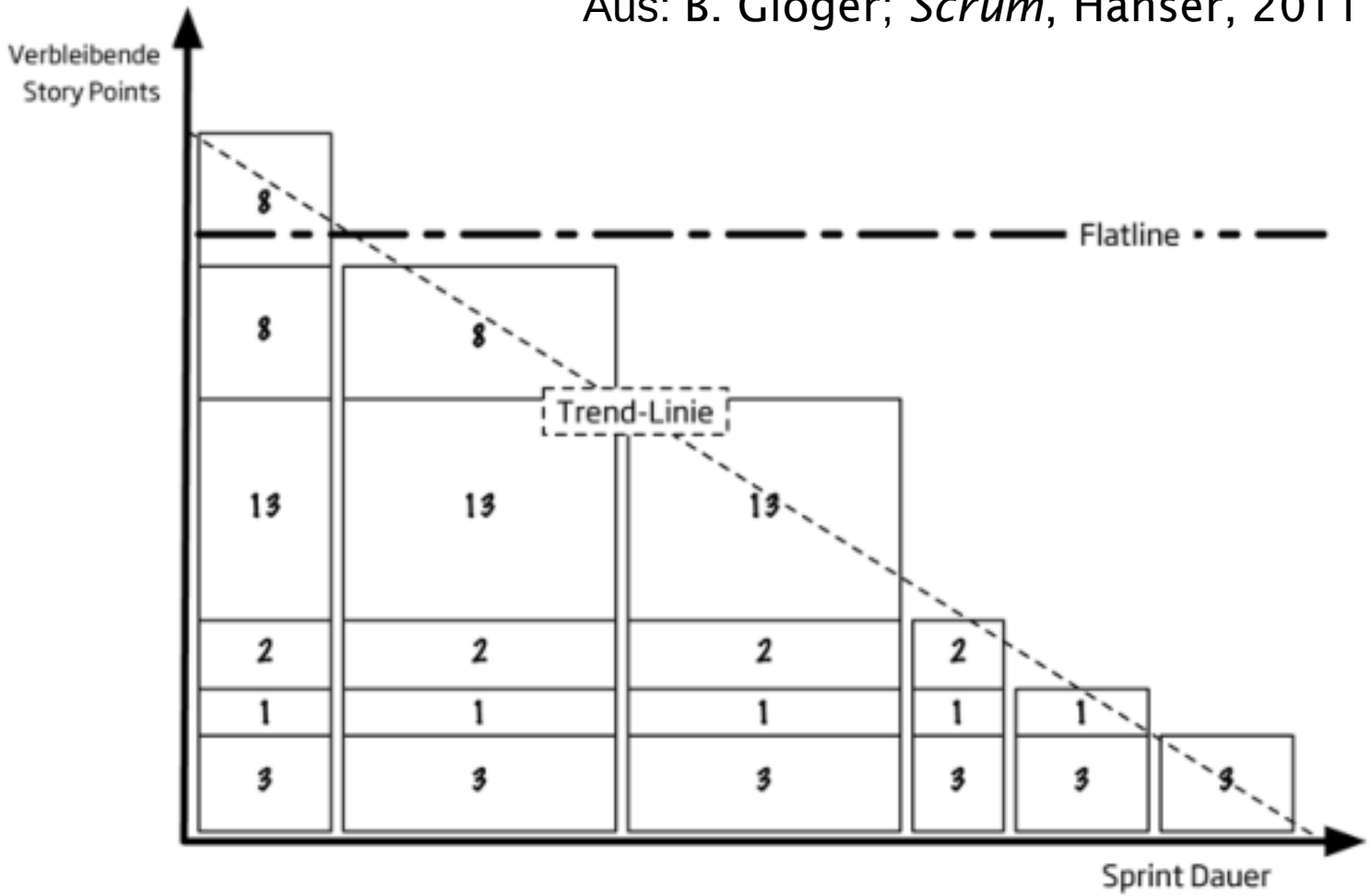
Sprint Burndown Chart



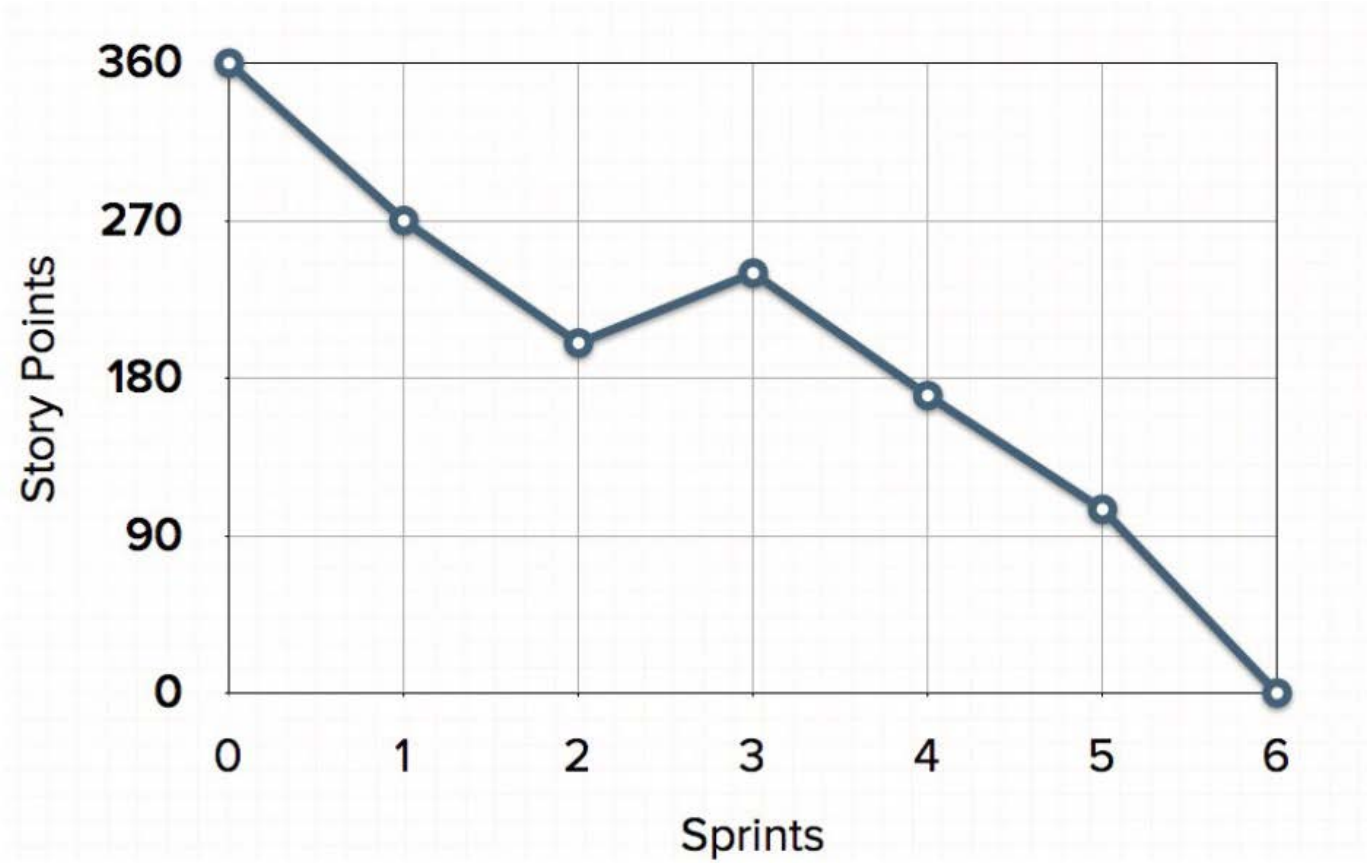
Quelle: https://de.wikipedia.org/wiki/Burn-Down-Chart#/media/File:SampleBurndownChart.png

Sprint Product Burndown Chart

Aus: B. Gloger; *Scrum*, Hanser, 2011



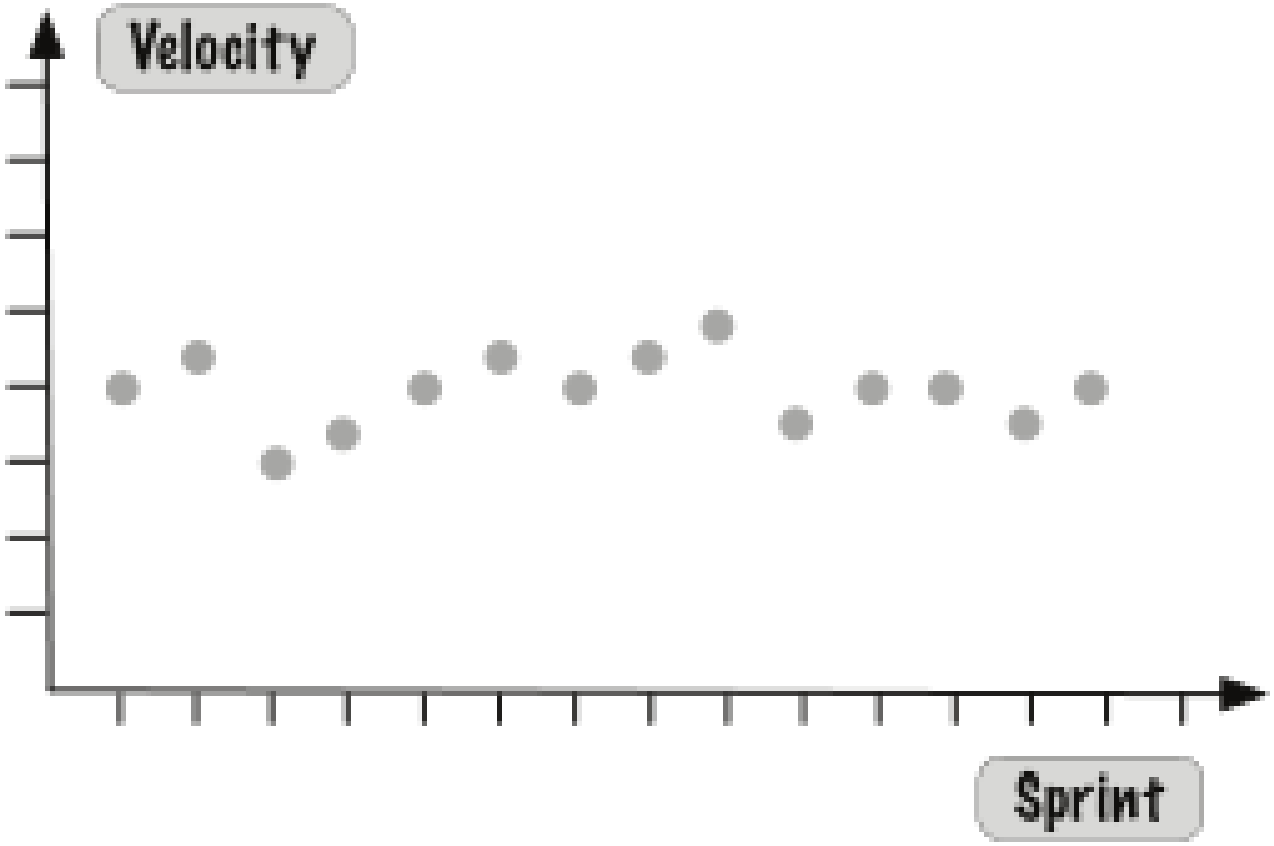
Release Burndown Chart



Aus http://www.mountaingoatsoftware.com/agile/scrum/release-burndown

Velocity Chart

Aus: B. Gloger; *Scrum*, Hanser, 2011



Weiteres bei Scrum

- „Commitment“
- Methoden:
  - Continuous Integration
  - Hohe Testautomatisierung

Gliederung

Einschätzung

- Agiles Vorgehen bedeutet nicht auf Planung zu verzichten.
- Scrum ist gut geeignet um Transparenz zu schaffen.
- Kein Allheilmittel.

Gliederung

Referenzen

- The Home of scrum: https://www.scrum.org/
- Scrum Guide by Ken Schwaber and Jeff Sutherland: https://www.scrum.org/Scrum-Guide
- http://de.wikipedia.org/wiki/Scrum
- B. Gloger; *Scrum*, Hanser, 2011
- Forrester: *Integrated Thinking: The Answer To IT's Perpetual Struggle, Oct. 2013: http://www.effectiveui.com/downloads/publications/EffectiveUI_Study_Integrated_Thinking.pdf*
- *Larman, Vodde: Practices for Scaling Lean&Agile Development, Addison Wesley, 2010*

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Literatur:

- B. Gloger: Scrum, 3. Auflage, Hanser
- Larman: Scaling Lean& Agile Development: Thinking and Organizational Tools für Large Scale Scrum, Addison-Wesley Professional
- Larman: Practices for Scaling in Lean & Agile Development, Addison-Wesley Professional
- Material unter http://www.craiglarman.com/wiki/index.php?title=Main_Page

Große Projekte

Was ist mit großen Projekten und Scrum?

➔Mehr als ein Scrum Projekt benötigt.

Vor dem kommenden die Warnung :

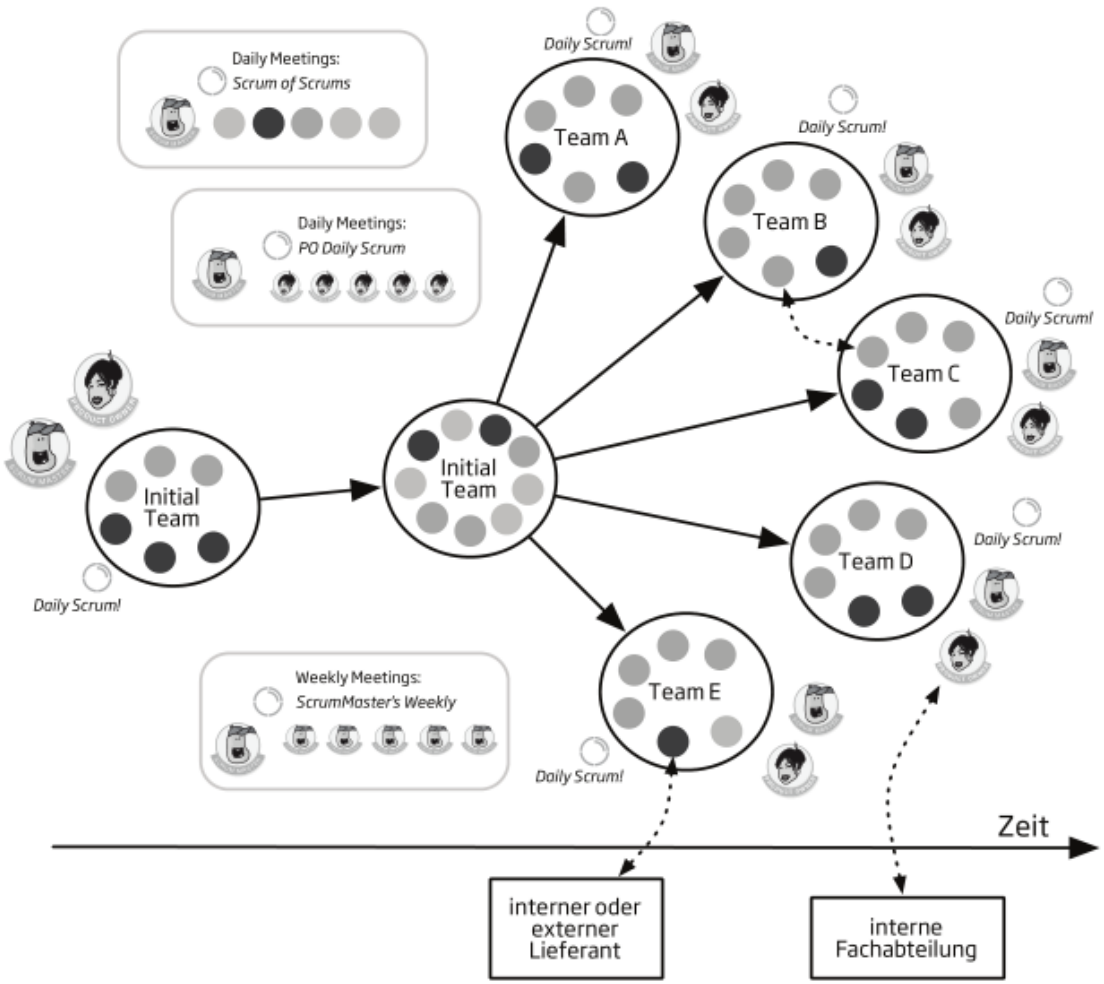
Wenn es sich vermeiden läßt, dann lassen Sie das bleiben!

Aufbau von mehreren Scrum Teams

Nach Gloger prinzipiell zwei Arten:

- Organisches Wachstum
  - Einarbeitung neuer Mitglieder, Team entscheidet, wann es sich teilt
- Sprunghaftes Skalieren
  - Die Mitglieder des initialen Teams übernehmen die Rolle des (Sub) Produkt Owners in den neuen Teams

Sprunghaftes Skalieren



Quelle: B. Gloger: Scrum, 3. Auflage, Hanser

Wie werden die Teams geschnitten?

Nach welchen Kriterien werden die Teams geschnitten?

Prinzipiell zwei Möglichkeiten:

- Component Teams
  - Verantwortung für technische Komponenten
- Feature Teams
  - Verantwortung für fachliche Funktionen

Synchronisation in großen Teams

Synchronisation durch:

- Scrum of Scrums
- Product Owner Team
- Scrum Master Group
- Virtuelle Teams für spezifische querschnittliche Aspekte (Architektur, Dokumentation, Tests, …)
- Gemeinsame Planning Meetings
- Evtl weitere Meetings

Organisation von mehreren Scrum Teams

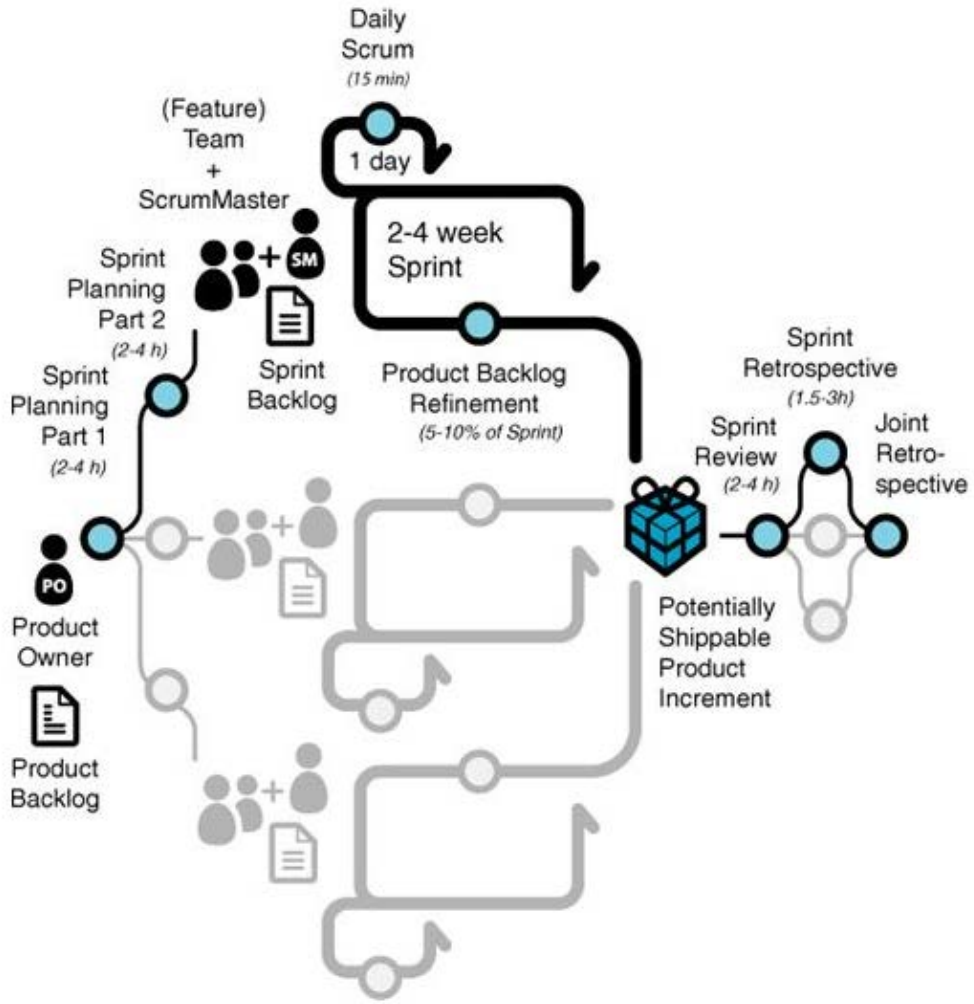
Wenn mehrere Scrum Teams existieren, wie ist die Organisation, welche Rollen existieren für die Teams?

Nach Larman: Practices for Scaling in Lean & Agile Development, Pearson

Versuche:

- FW 1 für bis zu 10 Teams ➔ siehe folgende Folie
- FW2 für größere Teams ➔ siehe übernächste Folie

FW1 für bis zu 10 Teams



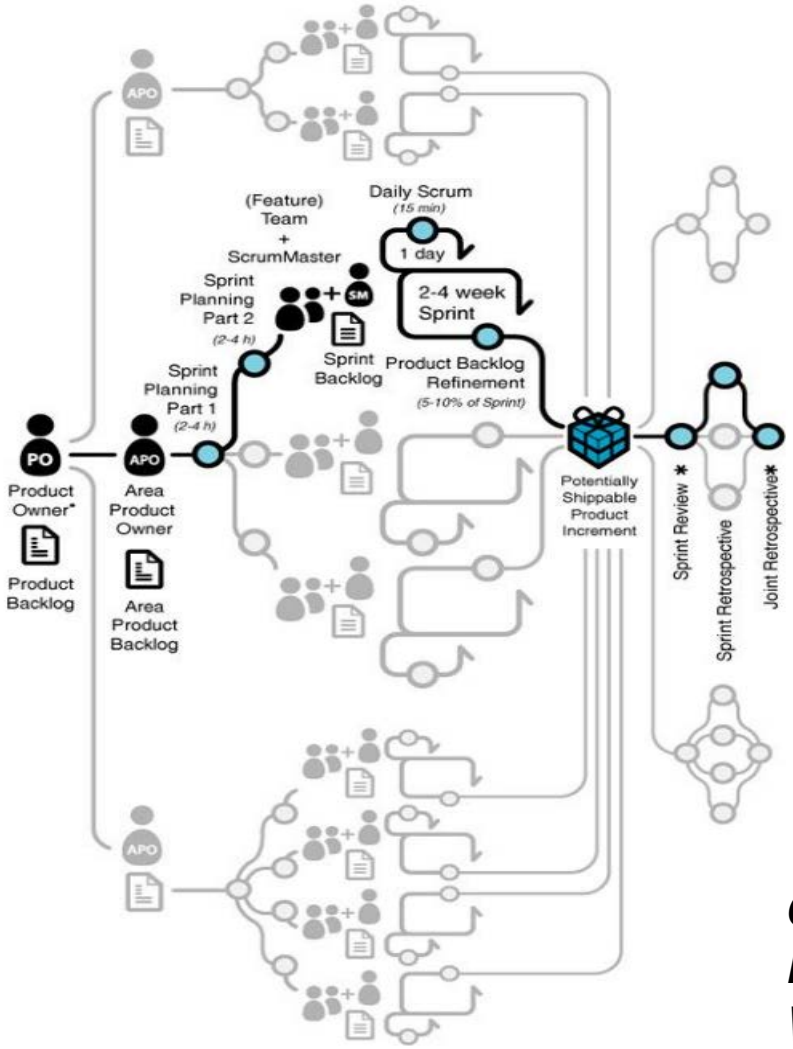
- Ein gemeinsamer PO
- Scrum Masters für jedes Team

- Product Backlog
- Sprint Backlogs
- Product Increment

- Sprint PLanning
- Daily Scrum
- Product Backlog Refinement
- Sprint Review
- Sprint Retrospectives
- Joint Retrospective

Quelle: Larman: Practices for Scaling in Lean & Agile Development, Addison-Wesley Professional

FW 2 für vieleTeams



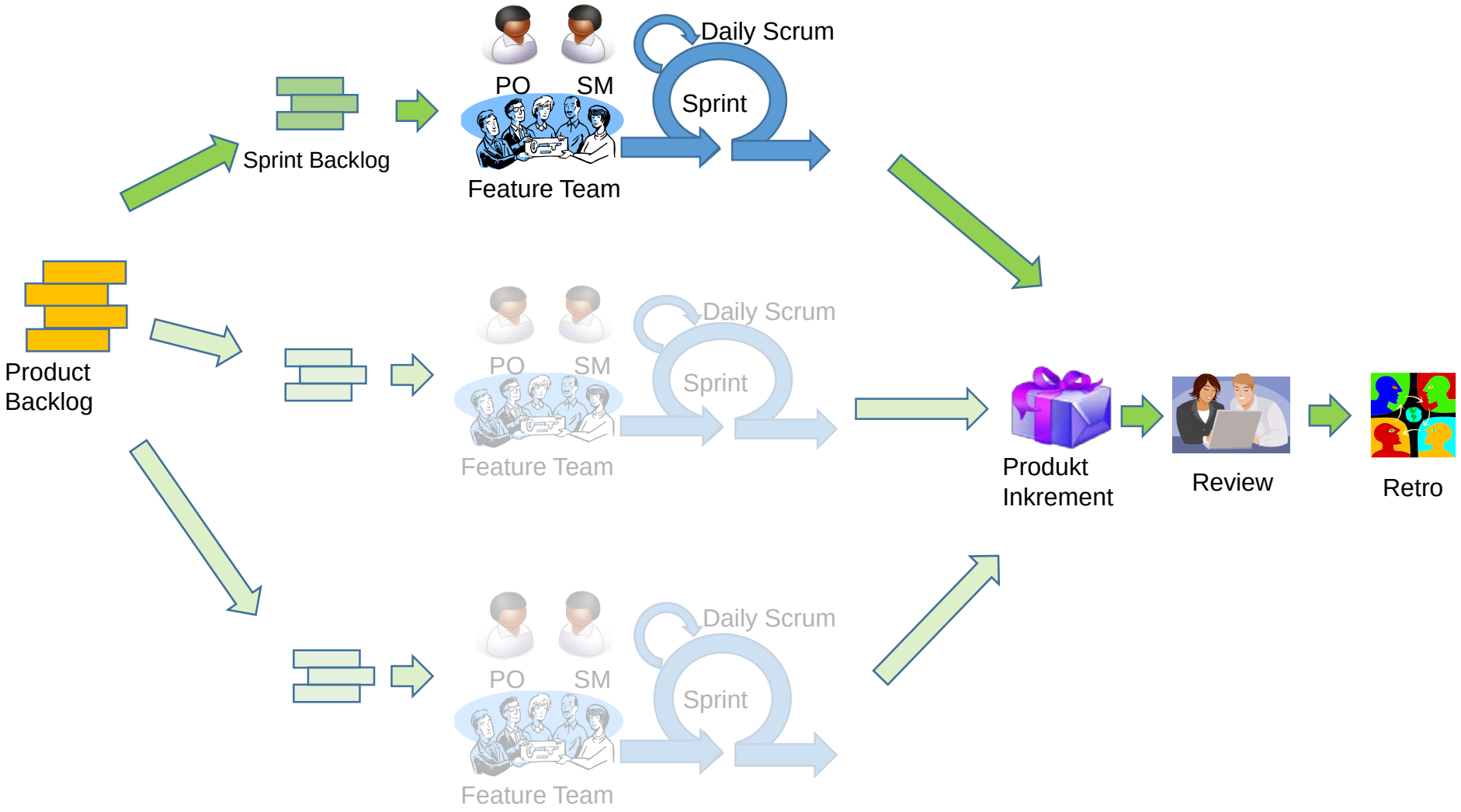
- Aufteilung der Teams in requirement areas
- Area Product owners
- Gruppen von FW 1 Organisationen

Quelle: Larman: Practices for Scaling in Lean & Agile Development, Addison-Wesley Professional

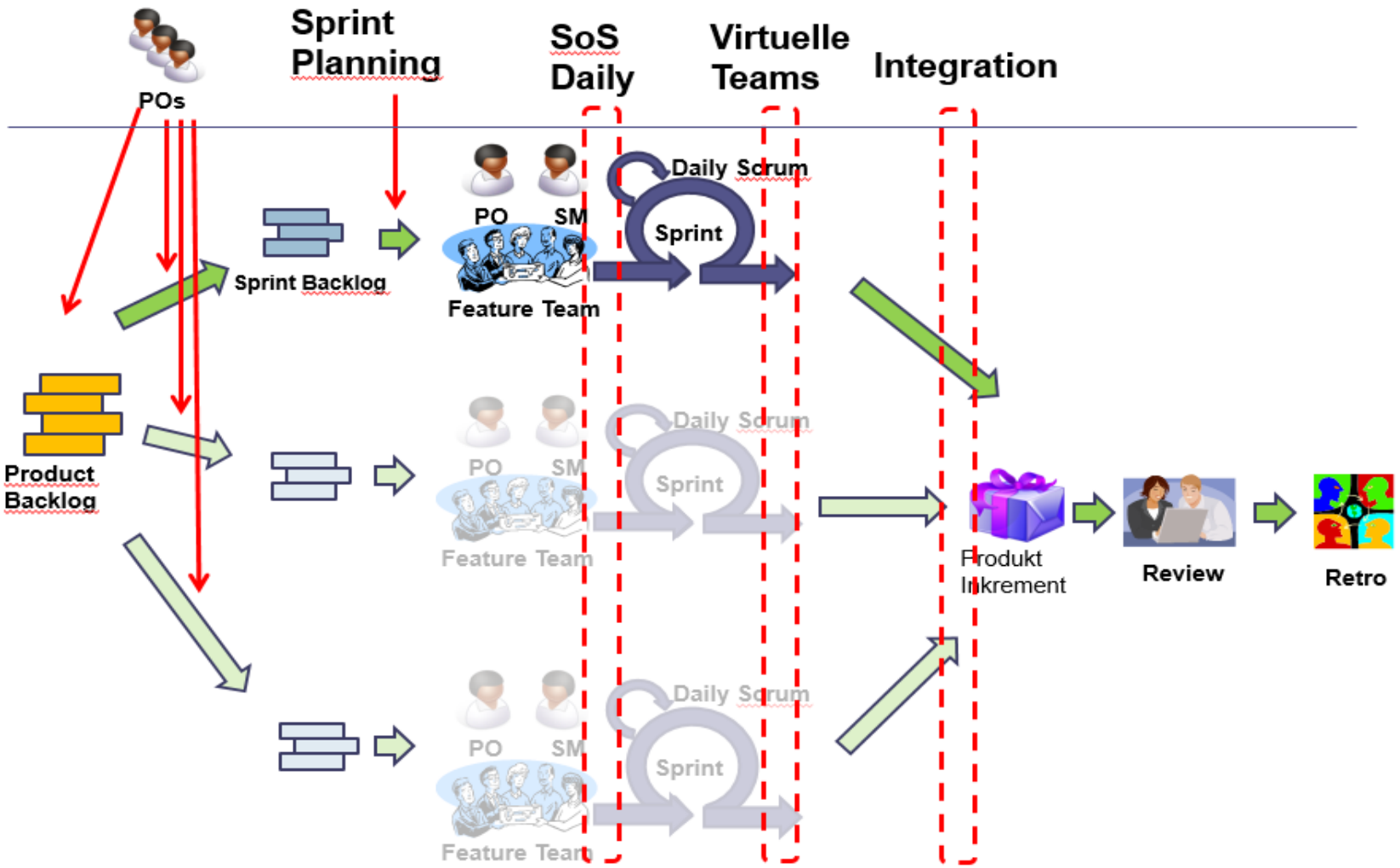
Beispielimplementierung von Scrum in großen Projekten

- Projekt mit bis zu 17 Scrum Teams
- Insgesamt bis zu 200 Projektbeteiligte
- Projekt unternehmenskritisch
- Mehrere Fremdfirmen beauftragt.

Scrum in großen Projekten - Bsp

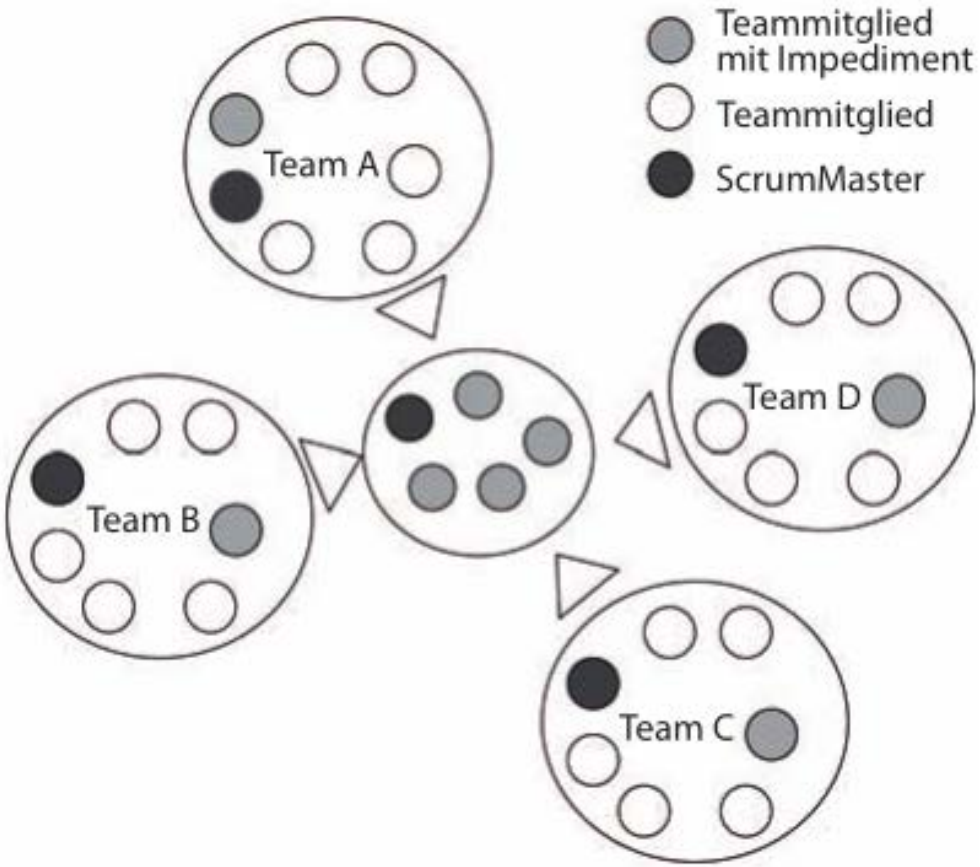


Scrum in großen Projekten



Tägliche Synchronsiation durch SoS

Scrum of Scrums



Quelle: Gloger: *Scrum*

Scrum of Scrums

Beispiel – Spezielle Themen

- **Projektleitung**
  - Trotz Scrum Projekt übergeordnete Projektleitung
- **Tests**
  - Im Bsp Projekt: Eigenes Testteam
- **Integration**
  - Im Bsp Projekt: Eigenes Team für Integration zu Sprint Ende
- **Entstehung der Teams:**
  - Durch die Projektleitung bestimmt

Vorgehensmodelle

- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Vertragsmodelle

Die Wahl eines Preismodells hängt von verschiedenen Faktoren ab:

- Projektart (Neuentwicklung, Anpassung, Weiterentwicklung usw.)
- Sicherheitsbedürfnis und Planbarkeit (stabiler Preis, Haushaltsplanung)
- Konkurrenzfähigkeit (möglichst niedriger Preis)
- Gewinnchance (für den Auftragnehmer)
- Nutzenerwartung (Nutzen soll höher sein als Kosten)
- AG-AN- Vertrauensverhältnis

Vertragsmodelle

Prinzipielle Unterscheidung:

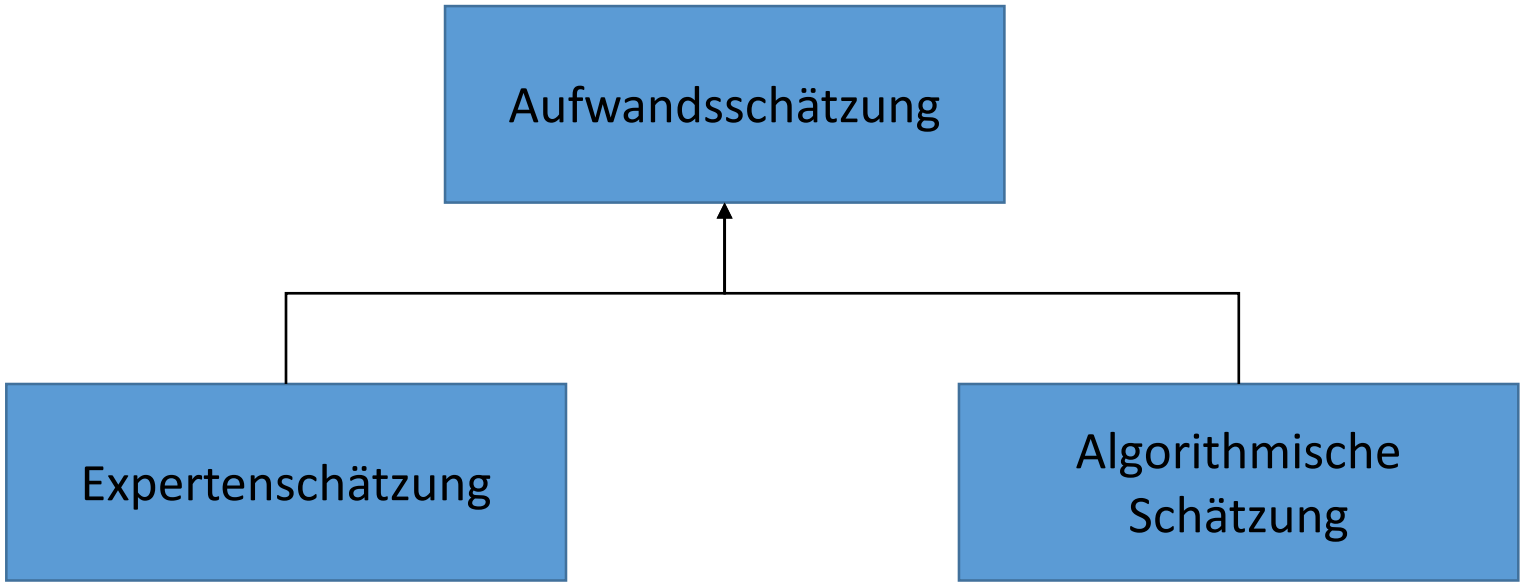
- Festpreis
- Aufwandspreis (Time&Material)

Varianten

- Aufwandspreis mit Obergrenze
- Phasenfestpreis
- Agiler Festpreis
- Festpreis mit inhaltlichem Spielraum

Vorgehensmodelle

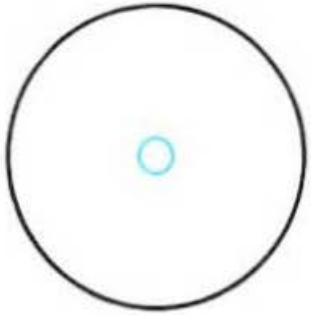
Ansätze zur Aufwandsschätzung



Fachleute nutzen ihre Erfahrung, um den Aufwand zu schätzen.

Kosten werden aus Größen berechnet, die frühzeitig bekannt sind oder leichter und genauer als der Aufwand geschätzt werden können.

Expertenschätzung

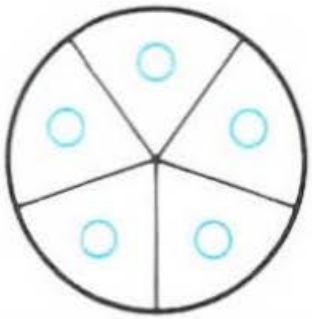


Einzelschätzung

- Einzelne Person schätzt Aufwand

Mehrfachbefragung

- Mehrere Personen schätzen unabhängig voneinander
- Durchschnittstbildung, ggfs mit Wichtung (z.B. Drei Punkte Schätzung)



Delphi-Methode

- Experten schätzen unabhängig voneinander
- Ergebnis wird aggregiert und den Experten anonym zurückgegeben. Experten verfeinern Schätzung
- Iteration, bis Ergebnis stabil ist.

Schätz Klausur

- Mehrere Experten schätzen im Kollektiv und nutzen gruppendynamische Aspekte

Variation der Delphi Methode

Oft praktiziertes Vorgehen:

1. Experten erarbeiten gemeinsam eine Work Breakdown Structure.
2. Jeder Experte schätzt unabhängig den Aufwand für jedes Element der Structure (ggfs mit Ober/Untergrenze und Erwartungswert).
3. Die Schätzungen werden verglichen und grobe Abweichungen voneinander diskutiert, bis eine gemeinsame Schätzung vorliegt.

Algorithmische Verfahren - COCOMO

C**OCOMO** (Constructive Cost Model)

Ausgangspunkt: geschätzte Programmgröße S, angegeben in DSI oder KDSI (Thousands of Delivered Source Instructions)

Prinzip:

- Aufwand und Entwicklungsdauer werden über Formeln berechnet, die empirisch aus archivierten Projektdaten gebildet wurden.
- Besondere Umstände werden durch Einflussfaktoren berücksichtigt, die man Tabellen entnimmt.

COCOMO

- Erste Version COCOMO 81
- Aktuell: COCOMO II

Umfangreiches Modell, das viele Eingabe Parameter enthält.

Es gibt keine Studie, die die Zuverlässigkeit der Methode zeigt.

Details siehe z.B. Ludewig, Lichter: Software Engineering, 3. Auflage 2013, dpunkt.verlag

Algorithmische Verfahren Function Point Verfahren

- **Basis**: Umfang des Programms in zu implementierenden Funktionen
- **Vorgehen**:

1. Alle für das Problem relevanten Daten, d. h. die logischen Datenbestände (Dateien) und alle Ein- und Ausgaben der zu realisierenden Vorgänge, werden erfasst und den folgenden Kategorien zugeordnet:
   - Externe Eingabe
   - Externe Ausgabe
   - Externe Abfrage
   - Interne Anwenderdaten
   - Externe Referenzdaten

Algorithmische Verfahren Function Point Verfahren

2. Anschließend wird der Schwierigkeitsgrad jedes Datums als niedrig, mittel oder hoch bewertet.
3. Mapping (Typ, Schwierigkeitsgrad)➔Punkte
4. Addition der Punkte ➔*Unadjusted Function Points*
5. Berücksichtigung von 14 Einflussfaktoren ➔bestimme Korrekturfaktor
6. ➔*Adjusted Function Points*
7. Mapping *Adjusted Function Points* ➔ PM
8. Mapping aus 7. wird ständig kalibriert.

Vorgehensmodelle

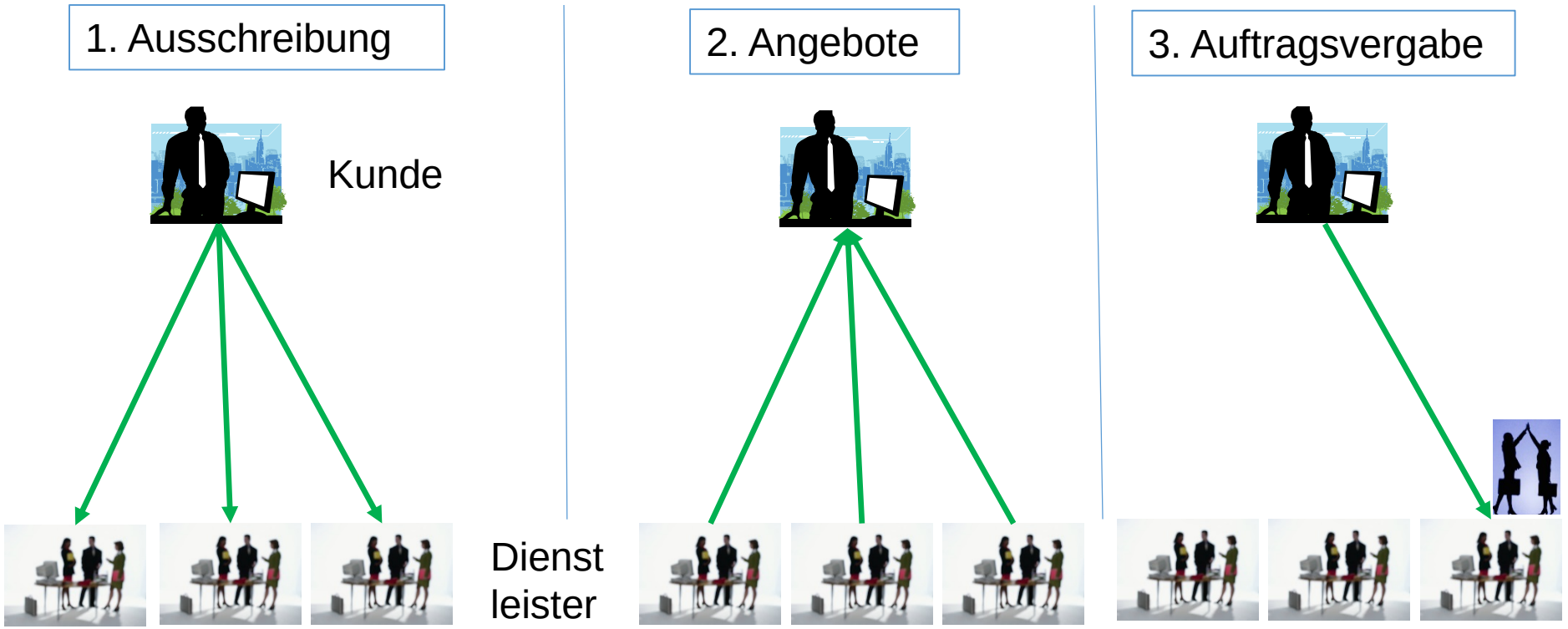
- Überblick (Wiederholung)
  - Einleitung
  - Wichtige Vorgehensmodelle – nicht agil
- Agile Methoden
  - Agile Methoden allgemein
  - Scrum
  - Scrum in großen Projekten
- Vertragsmodelle
- Aufwandsschätzung und Projektplanung
- Agiler Festpreis

Der Agile Festpreis

Literatur:

- Opelt et al.: *Der Agile Festpreis*, Hanser, 2012

Extern vergebene Leistungen



Auftragnehmer vs. Auftraggeber

Auftraggeber:

- Will abschätzen, was er für sein Geld bekommt.
- Will wissen, was ein System kosten wird.
- Will den günstigsten Anbieter auswählen können.
- Muss dafür Budget freischalten.
- Will sein Risiko minimieren.

Auftragnehmer

- Will abschätzen, wieviel er verdienen kann.
- Will die Mitbewerber ausstechen.
- Will sein Risiko minimieren.

Auftragnehmer vs. Auftraggeber

Auftraggeberinteressen:

- Tendenz zu Festpreis
  - ➔ Risiko beim Auftragnehmer

Auftragnehmerinteressen:

- Tendenz zu Aufwandsvertrag
  - ➔ Risiko beim Auftraggeber

➔**Lösungsversuch**: Agiler Festpreis als neues Vertragsmodell.

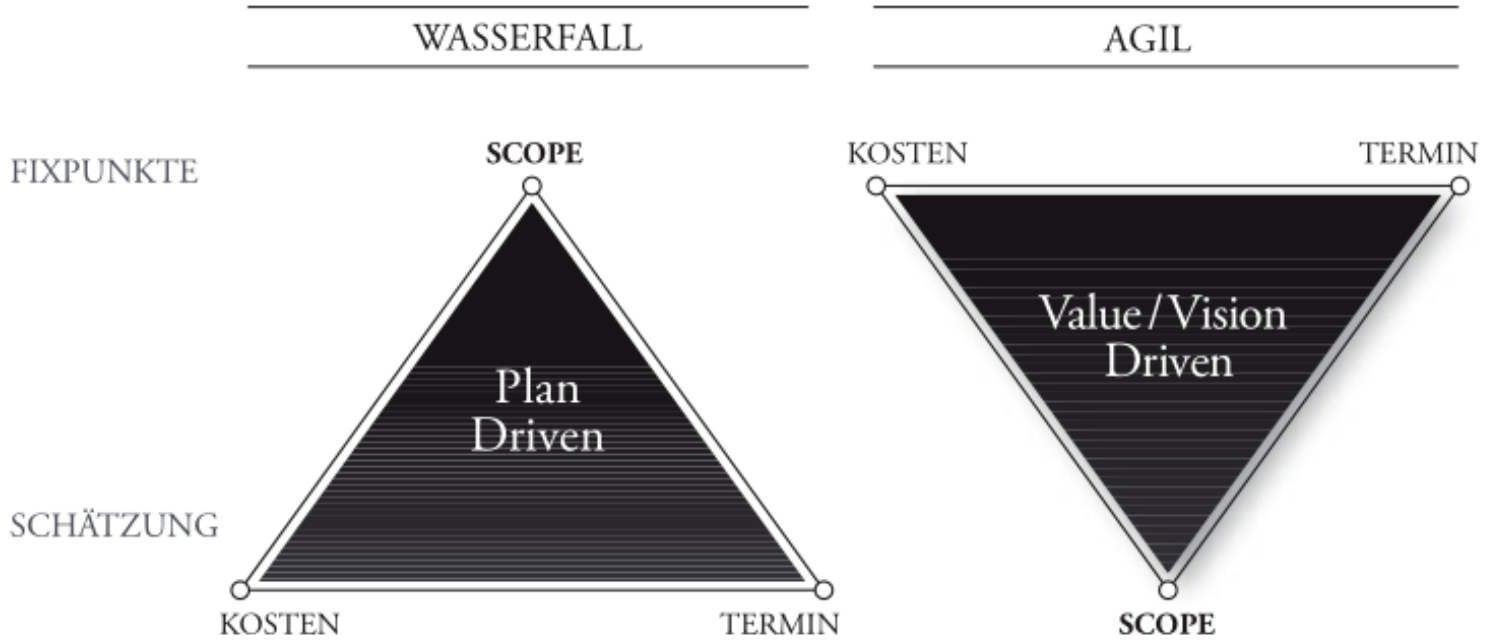
Der agile Festpreis – Unterscheidung nach Bernd Oesterreich

Anforderungseinheitspreis: Es wird ein verbindlicher Gesamtpreis vereinbart und ein für den Auftraggeber transparentes Verfahren definiert, wie der Preis einer realisierten Anforderung geschätzt und bemessen werden kann. Der Auftraggeber bekommt insgesamt Anforderungen im Wert des Festpreises und kann noch nicht realisierte Anforderungen während des Projektes ändern.

Inhaltsvarianter Festpreis:
Für eine gegebene Menge von Anforderungen wird ein verbindlicher Gesamtpreis vereinbart. Zusätzlich wird (wie beim Anforderungseinheitspreis) ein für den Auftraggeber transparentes Verfahren definiert, wie der Preis einer realisierten Anforderung geschätzt und bemessen werden kann. Der Auftraggeber bekommt insgesamt Anforderungen im Wert des Festpreises und kann noch nicht realisierte Anforderungen jederzeit durch andere gleichteure Anforderungen ersetzen.

Agiler Festpreis

Aus: Der Agile Festpreis



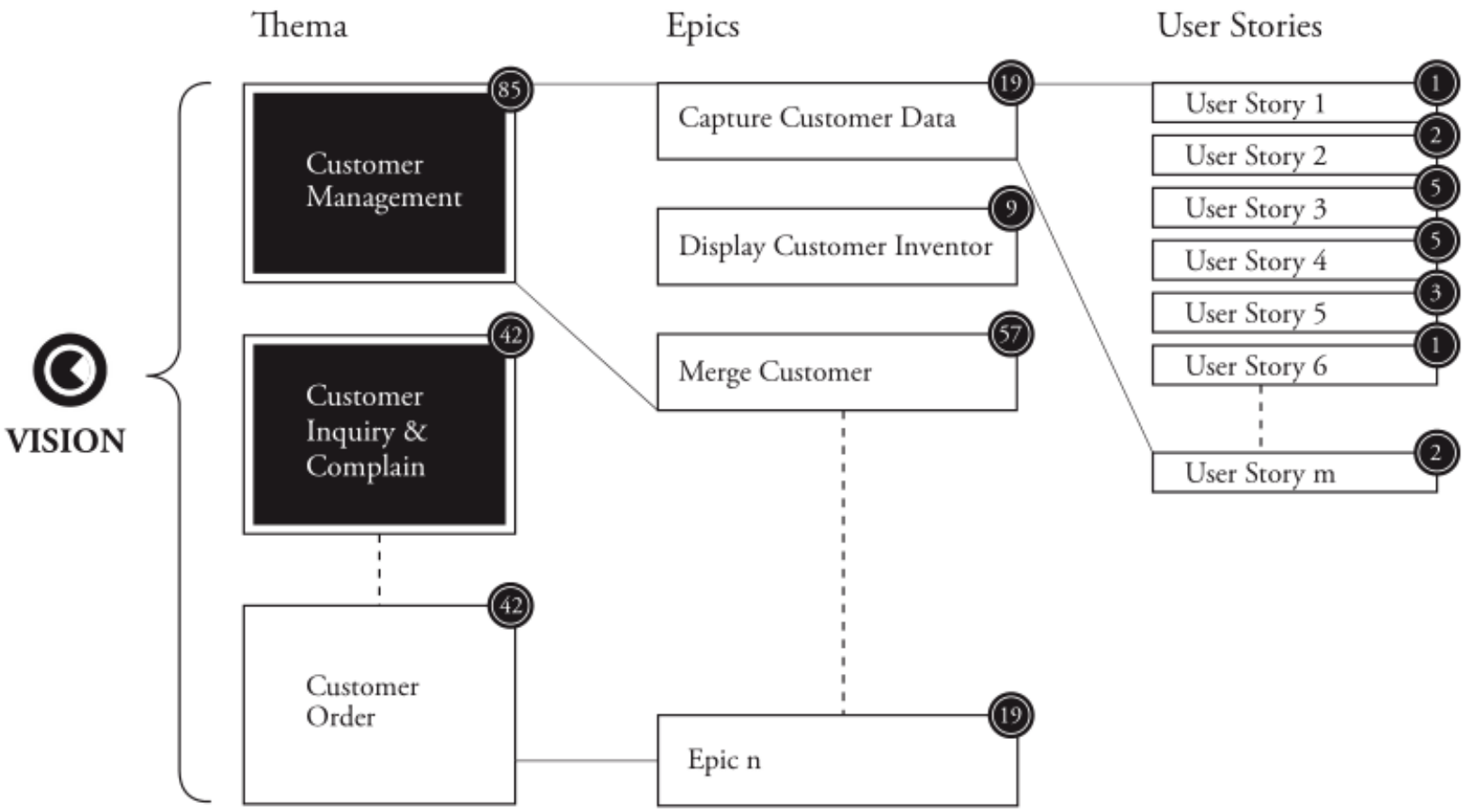
Wie kommt man zu einem agilen Festpreisvertrag?

Nach „Der Agile Festpreis“ kommt man mit folgenden Schritten zu einem Agilen Festpreisvertrag

1. Definition des Vertragsgegenstands auf grobgranularer Ebene. (Aus Sicht des Anwenders, vollständig, aber nicht detailliert)
2. Detailspezifikation einer exemplarischen Menge an Referenz User Stories.
3. Gemeinsamer Workshop zum Gesamtscope ➔ indikativer Festpreisrahmen
4. Definition von Riskshare, Checkpoint Phase und Ausstiegspunkten
5. Vereinbarung zur Scope Governance
6. Definition von Motivationsmodell und Kooperationsmodell

Quelle: Der agile Festpreis, Hanser 2012

Detaillierung der Vision



Quelle: Opelt et al: Der agile Festpreis

Wie kommt man zu einem agilen Festpreisvertrag?

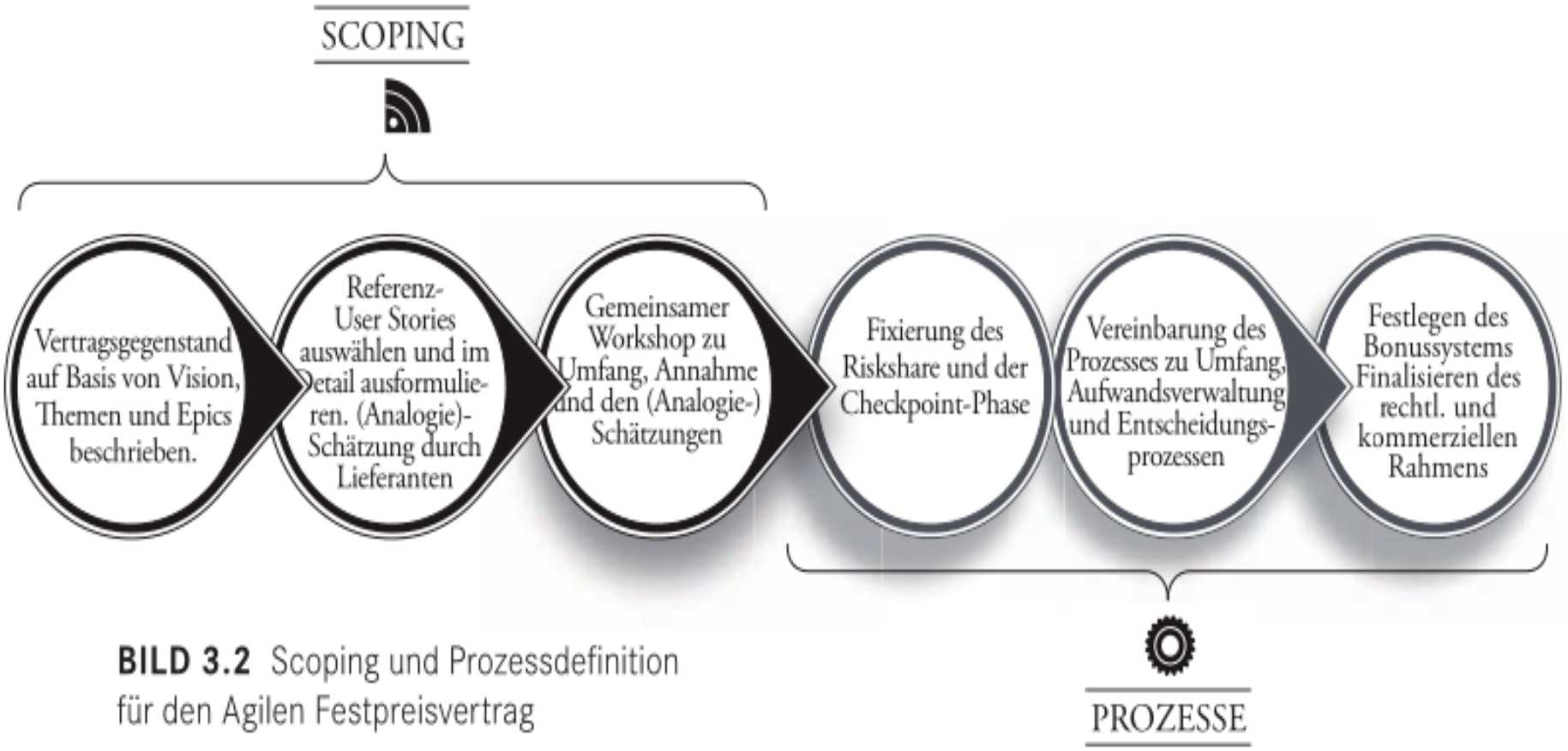


BILD 3.2 Scoping und Prozessdefinition für den Agilen Festpreisvertrag

Aus Der Agile Festpreis, Seite 46