Agilität:
Agiles Requirements Engineering
(User Stories, Schätzen, Verfolgen)

Eine Übersicht
zur Vertiefung / Ergänzung
Für Projektmitarbeiter und Requirements Ingenieure
(in agilen Projekten)
Stand: 05/2014

Sie finden diese und weitere Präsentationen unter (→ Klick):
http://www.peterjohann-consulting.de/praesentationen

Zusammengestellt von H. Peterjohann
zur Verteilung an Interessierte
Version 0.10 vom 12.05.2014
84 Seiten

Motivation und Einordnung (1/2)

Mit dem Einsatz von agilen Methoden sind viele der im klassischen Requirements Engineering eingesetzten Werkzeuge und Methoden zur Gewinnung und Verwaltung der Anforderungen nicht mehr passend, da umfangreiche Anforderungsdokumente in agilen Umfeldern kaum noch von Bedeutung sind. Ein Ziel des klassischen Requirements Engineerings ist die Gewinnung eines möglichst umfassenden Gesamtüberblicks – dieses wird in agilen Projekten ersetzt durch das Ziel, möglichst schnell zur Umsetzung von Teilbereichen zu gelangen, die dann wiederum direkt beim Kunden gewinnbringend zum Einsatz kommen können.

Daher wird in agilen Projekten das klassische Requirements Engineering (RE) durch Agiles Requirements Engineering (ARE) ersetzt.

Motivation und Einordnung (2/2)

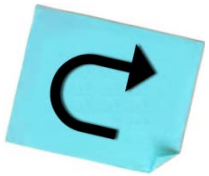
Agiles Requirements Engineering (ARE) ergänzt die agilen Methoden und Ansätze, wie sie beispielsweise in Scrum zum Einsatz kommen, um Methoden zur Ermittlung und Verwaltung der Anforderungen, die meistens über User Stories formuliert werden.
Grundkenntnisse von Scrum werden beim Agilen Requirements Engineering und damit auch in dieser Präsentation vorausgesetzt. Das klassische Requirements Engineering wird ebenfalls nur am Rande gestreift, so dass auch hier Basiskenntnisse des REs vorausgesetzt werden.

Diese Ausarbeitung beschreibt, wie Anforderungen in agilen Umfeldern ermittelt, beschrieben und verwaltet werden.

Bitte beachten Sie:
Diese Ausarbeitung ist zwar in sich geschlossen, basiert aber in Teilen auf der umfangreichen **Requirements-Engineering-Basispräsentation**, die unter http://www.peterjohann-consulting.de/requirements-engineering frei herunterladbar ist, sowie die **Präsentationen zu Scrum**, die unter http://www.peterjohann-consulting.de/scrum zu finden sind.

Übungen, Fragen, Tipps und Checklisten

Einige Folien sind besonders herausgehoben, da sie vom Leser / Teilnehmer besondere Aktivitäten erwarten; folgende vier Sonderfolien gibt es:

	Übung	Eine Übung beschreibt eine Aufgabe, die durch den Leser selbst oder besser in einer (Klein-)Gruppe gelöst werden sollte. Es wird immer eine Dauer mitangegeben, d.h. eine Zeitvorgabe, wie lange die Bearbeitung benötigen sollte. Musterlösungen existieren im Allgemeinen hierfür nicht.
	Fragen	Kontrollfragen, die zur Überprüfung des Lernziels / des Gelernten dienen und daher individuell beantwortbar sein sollten. Musterlösungen existieren, werden aber hier nicht veröffentlicht.
	Tipps	Empfehlungen und Ratschläge – zumeist unmittelbar aus der Praxis kommend.
	Check-liste	Checklisten dienen zur Überprüfung, ob „alles richtig gemacht wird“ (in konkreten Projekten). Es werden nur kurze Checklisten verwendet; Langfassungen werden hier nicht veröffentlicht.

Ziel dieser Ausarbeitung (für den Leser)

Folgende Inhalte werden in dieser Ausarbeitung behandelt und sollten Ihnen nach dem Durcharbeiten bekannt sein:

- Sie kennen die Grundelemente und Basisbegriffe des Agilen Requirements Engineerings (ARE)
- Sie wissen, welche Elemente des klassischen RE Sie im ARE verwenden können und welche nicht
- Sie kennen die Bedeutung der User Stories im ARE und können User Stories schreiben
- Das Erfassen, Beschreiben, Schätzen und Verwalten von Anforderungen im ARE ist Ihnen geläufig
- Sie können einige Methoden des ARE in Ihrem Projektalltag einsetzen

Zielgruppe: Projektmitarbeiter und Requirements Ingenieure (in agilen Projekten)
Voraussetzungen: Grundkenntnisse im Requirements Engineering und in Agilität (Scrum)
Schwierigkeitsgrad: Mittel

Aufbau dieser Präsentation

Diese Präsentation ist wie folgt gegliedert:

Kapitel 1 beschreibt die theoretischen Grundlagen: Es werden Beschreibungen für das Agile Requirements Engineering (ARE) geliefert und eine Abgrenzung zum „klassischen“ Requirements Engineering vorgenommen.

Das **Kapitel 2** beschäftigt sich ausführlich mit dem zentralen Element des ARE, den User Stories. Hierüber werden recht schnell und einfach Anforderungen erfasst und anschließend zur Umsetzung gebracht.

Das **Kapitel 3** hat das Schätzen (von Umsetzungsaufwänden der User Stories) zum Inhalt. Die Schätzungen haben dabei eine andere Bedeutung als beim klassischen Vorgehen, denn es werden nur relative Schätzwerte ermittelt.

Im abschließenden **Kapitel 4** wird beschrieben, wie beim ARE die Anforderungen verfolgt und verwaltet werden können. Typischerweise kommen hier Task Boards zum Einsatz.

Im **Anhang** sind die Literaturliste und eine Liste mit Weblinks zum Agilen Requirements Engineering zu finden.

Gliederung

Gliederung

Kapitel 1: Einleitung und Grundlagen

Kapitel 1

- Grundlegende Unterschiede des klassischen und Agilen Requirements Engineerings
- Das magische Dreieck beim klassischen und Agilen Requirements Engineering
- Unterschiede klassisches und Agiles Requirements Engineering
- Einige zentrale Begriffe im Agilen Requirements Engineering
- Das Product Backlog (Beschreibung, Aussehen)
- Was macht der Requirements Engineer im agilen Kontext?
- Tipps zum Kapitel
- Fragen zum Kapitel

Grundlegende Unterschiede des klassischen und Agilen Requirements Engineerings (1/2)

Beim klassischen RE werden zunächst „alle Anforderungen“ an ein neues Produkt erfasst und weiter ausformuliert. Der RE-Prozess ist dann (vorläufig) beendet, wenn die Anforderungen in ein Projekt einfließen. Hierzu wird der Aufwand zur Umsetzung der Anforderungen geschätzt und dann in einem zentralen System erfasst.

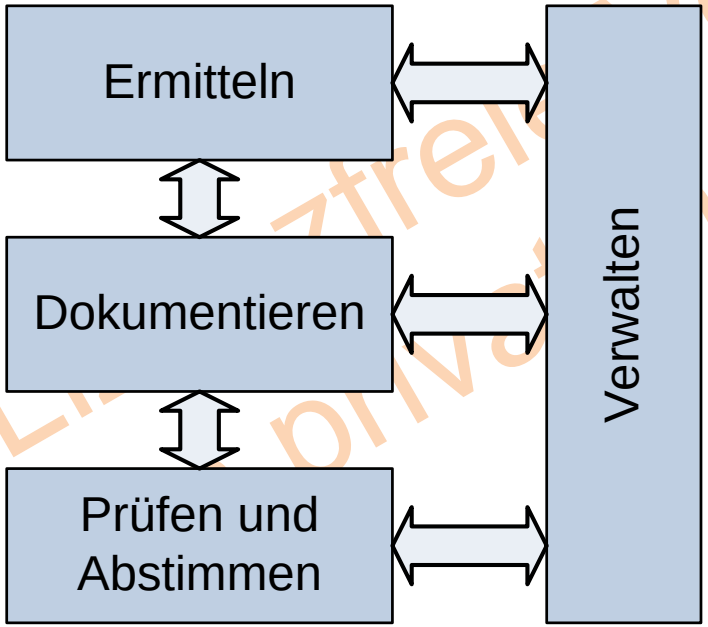
Beim „Agilen Projektmanagement“ wird im Vorfeld nicht das gesamte Projekt durchgeplant, sondern nur die Themen, die im nächsten Schritt benötigt werden und die dann auch unmittelbar zur Umsetzung gelangen. Hierdurch wird nur das im Detail (aufwendig) geplant, was tatsächlich für den nächsten Umsetzungsschritt benötigt wird.

Vorweg:

- Es werden hier bevorzugt die englischen Begriffe wie User Stories, Epics, Themes verwendet (und nicht die deutschen wie Benutzergeschichten, Epen, Themen)
- Wenn es nicht anders angegeben ist, so wird davon ausgegangen, dass Scrum als agile Methode eingesetzt wird

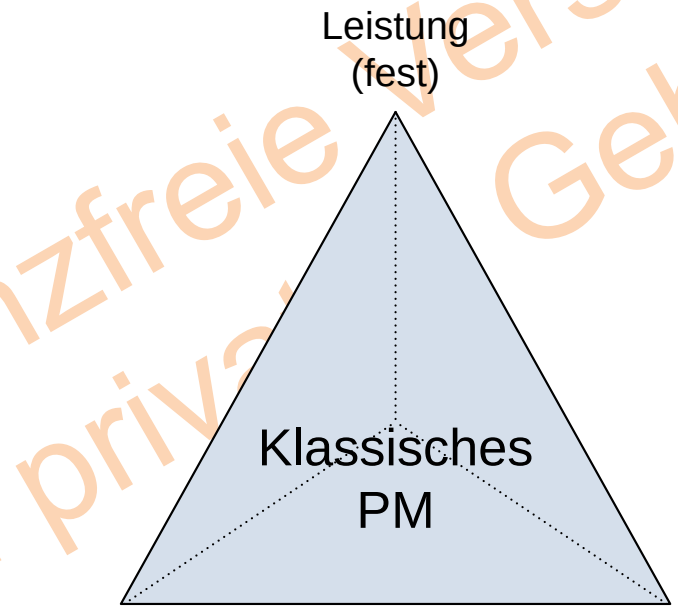
Grundlegende Unterschiede des klassischen und Agilen Requirements Engineerings (2/2)

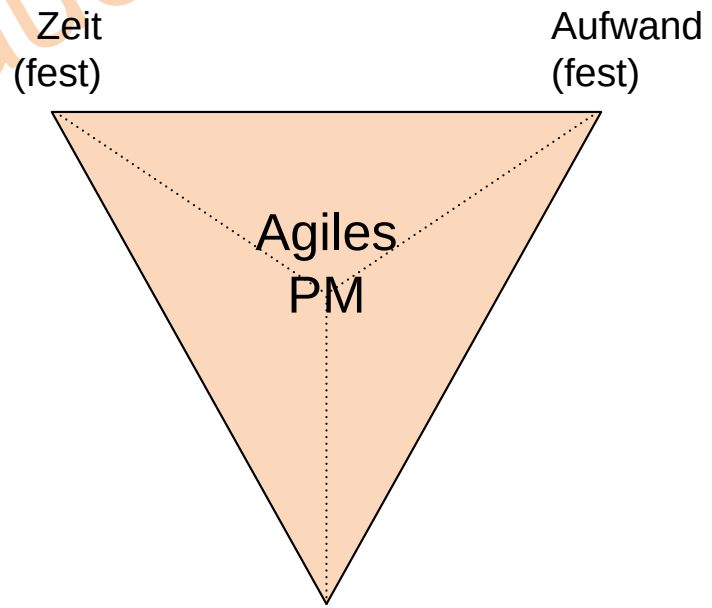
Somit muss ein Agiles Requirements Engineering (kurz: ARE) anders funktionieren als klassisches („traditionelles“) Requirements Engineering, obwohl die Themen (Ermitteln, Dokumentieren, Prüfen und Abstimmen, Verwalten) prinzipiell die gleichen sind.



Das magische Dreieck beim klassischen und Agilen Requirements Engineering (1/2)

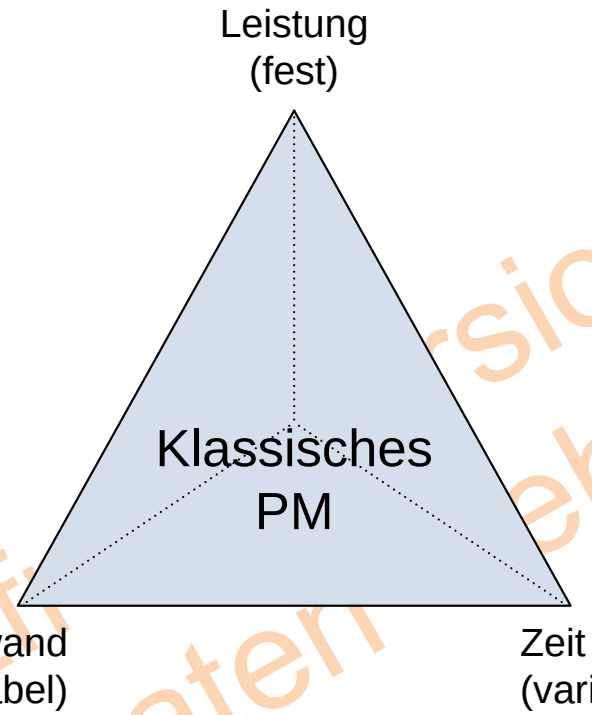
Der Zusammenhang von Leistung, Aufwand und Zeit wird im Projektmanagement häufig durch das magische Dreieck (engl. *„Iron Triangle“* oder *„Triple Constraint“*) beschrieben: Während beim klassischen PM der Leistungsumfang die einzig feste Größe ist, wird beim Agilen PM auf feste Zeit(abschnitte) und festen Aufwand Wert gelegt.

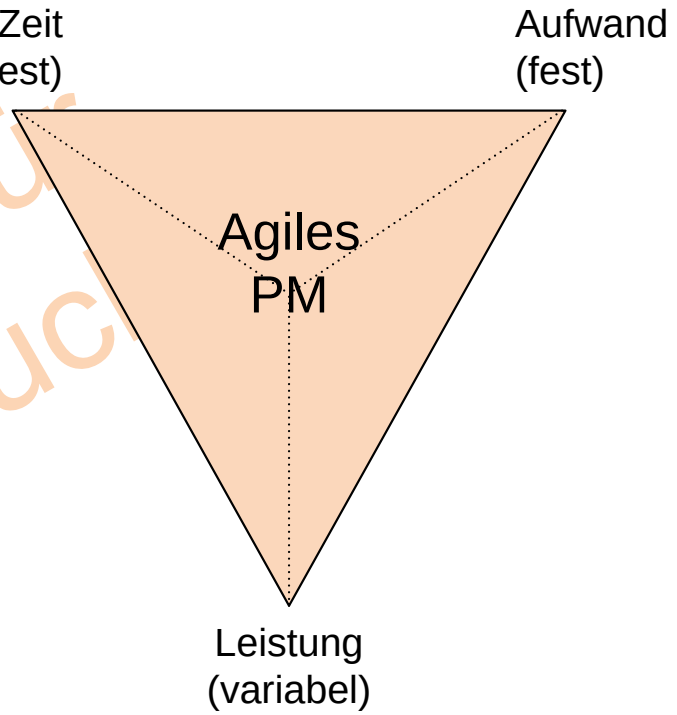




Das magische Dreieck beim klassischen und Agilen Requirements Engineering (2/2)

Aufwand = Kosten
und Mitarbeiter





- Alle Anforderungen (die die Leistung beschreiben) sind vollständig definiert und festgeschrieben
- Der Zeitraum ist geplant
- Der Aufwand kann erhöht werden, um die Fertigstellung zu beschleunigen

- Die Anforderungen (nicht unbedingt alle) sind grob beschrieben und teilweise priorisiert
- Im festgelegten Zeitraum werden die wichtigsten Anforderungen umgesetzt
- Der Aufwand ist festgelegt und wird gesteuert, um innerhalb des Aufwandrahmens zu bleiben

Unterschiede klassisches und Agiles Requirements Engineering (1/2)

Beim klassischen Requirements Engineering …

- werden Anforderungen als Ganzes möglichst vollständig vor Projektbeginn (oder vor Beginn einer Projektphase) ermittelt.
- orientiert sich der Entwickler an den Anforderungen, die er (technisch) verstehen muss.
- sind Änderungen vergleichsweise schwer und aufwendig einzubringen.

Beim Agilen Requirements Engineering …

- werden Anforderungen kurzfristig ermittelt.
- wird die Anforderungsermittlung kontinuierlich durchgeführt.
- werden Anforderungen dann vollständig bestimmt, wenn sie zur Realisierung gelangen sollen.
- werden selten schwergewichtige Werkzeuge, sondern eher „leichte“ Tools eingesetzt.
- sind Änderungen weniger schwierig umzusetzen.

Unterschiede klassisches und Agiles Requirements Engineering (2/2)

	Klassisches RE	Agiles RE
Zentrale Verwaltung	Ja – Anforderungskatalog	Ja – Product Backlog
Prozess	Ja, z.B. nach IREB (International Requirements Engineering Board)	Nein
Zentrales Element zur Erfassung	Anforderung – in UML-Notation (häufig Use Cases)	User Story
Zeitpunkt der Anforderungserfassung	Zu Beginn: Möglichst umfassend und detailliert	Fortlaufend
Wer erfasst Anforderungen?	Bei größeren Projekten: Eigene Anforderungsteams	Der Product Owner (und das Entwicklerteam)
Schätzungen	Ja – Einsatz verschiedener Schätzmethoden	Ja, meistens: Planning Poker oder andere
Weiterverwendung der Anforderungen	Ja – über Tools; ggf. Grundlage für Benutzerdokumentation	Nicht vorgegeben

Einige zentrale Begriffe im Agilen Requirements Engineering

Bei den Agilen Methoden gibt es einige zentrale Elemente, über die die Entwicklung des zu erstellenden Systems gesteuert wird. Dies sind (unter anderem):

- **Product Backlog**: Eine Liste mit allen Anforderungen, die nach und nach zur Umsetzung gelangen
- **User Stories, Epics, Themes**: Elemente, um die Beschreibung des zu entwickelnden Systems zu gliedern und so zu unterteilen, dass die Anforderungen (an das System) besonders einfach umgesetzt werden können
- **(Scrum) Task Board**: Wandtafel (Whiteboard), die sich häufig in agilen Umfeldern findet, auf der die einzelnen Prozessschritte über Spalten erfasst werden. Die User Stories werden dort (als Karten) angeheftet und entsprechend ihres Bearbeitungszustands umgehängt

Diese Elemente werden hier im Nachfolgenden erläutert.

Das Product Backlog (1/2): Beschreibung

Alle Anforderungen werden beim Agilen Requirements Engineering in dem **Product Backlog** festgehalten. Dieses enthält die Beschreibungen der (von den Anwendern gewünschten) Merkmale (**Features**) des zu erstellenden Produkts. Diese werden in einzelnen Punkten, den sogenannten **Product Backlog Items**, „tabellenartig“ festgehalten (siehe nächste Folie).

Die Backlog Items selbst sind bevorzugt **User Stories** (siehe hierzu Kapitel 2), es können aber auch Mindmaps o.Ä. verwendet werden. Typischerweise wird das Product Backlog über eine Wandtafel (Scrum Task Board), seltener über ein Software-Tool verwaltet.

Das Product Backlog ist das zentrale Tool zum Verwalten der Anforderungen im Agilen Requirements Engineering.

Das Product Backlog (2/2): Aussehen

Das Product Backlog kann als Tabelle aufgebaut sein und dann folgenden, minimalen Aufbau besitzen:

ID	Beschreibung	Priorität	Story Points	Notizen
1				
2				
3				
4				

Eintrag durch das Team

Eintrag durch Product Owner

ID	Fortlaufende ID, durch das System generiert
Beschreibung	Freitext - oder besser: Story-Text, wenn es User Stories sind
Priorität	Priorität; typische Skalen [A, B, C] oder [hoch, mittel, niedrig]
Story Points	Abschätzung für den Aufwand
Notizen	Freitext; kann auch leer bleiben

Ein Product Backlog Item entspricht genau einer Zeile in der Tabelle.

Was macht der Requirements Engineer im agilen Kontext?

Dem Requirements Engineer kommt beim Agilen Requirements Engineering – und hier speziell bei Scrum – die Rolle des Product Owners zu.

Der Product Owner hat folgende Aufgaben:

- Zusammenstellen aller Anforderungen im Product Backlog
- Koordinieren des Schätzens und des Priorisierens der umzusetzenden Anforderungen (Items) im Product Backlog
- Initiieren der Umsetzung (für den nächsten Sprint)
- Koordinieren des Einführungsprozesses
- Ansprechpartner für das Realisierungsteam und für den Kunden / das Management

Damit übernimmt der Product Owner mehr Aufgaben als der „klassische“ Requirements Engineer; einige Funktionen des klassischen Projektmanagers werden auf den Product Owner übertragen.

Tipps zum Kapitel

- Überlegen Sie vorab, welches Vorgehen beim Requirements Engineering zu Ihrem sonstigen Vorgehen passt
- Auch wenn Sie agil arbeiten, helfen Kenntnisse über die Vorgehensweisen und Methoden im klassischen Requirements Engineering

Fragen zum Kapitel

1. Was ist Agiles Requirements Engineering?
2. Was sind die Unterschiede von klassischem und Agilem Requirements Engineering?
3. Was ist der Unterschied zwischen einem Requirements Engineer und einem Product Owner?
4. Warum benötigt man in agilen Umfeldern ein eigenes, angepasstes Requirements Engineering?
5. Was ist das Product Backlog?

Kapitel 2: User Stories

Kapitel 2

- Was ist eine User Story? (Grundsätzliches, Definitionen, Charakterisierung, Die 3 Cs)
- Aufbau und Aussehen einer User Story (Schema / Template, Inhalte, Beispiele)
- Weiteres zu User Stories (INVEST, Probleme und Stolpersteine, Abgrenzung: User Story und Use Case, Begriffe rund um die User Story)
- Übung: Sind das (gute) User Stories? (Aufgabe, Lösung)
- Das Schneiden von User Stories (Story Decomposition, Das Story Splitting Cheat Sheet)
- Epics
- Themes
- Features
- Die ARE-Pyramide
- Definition of … (Definition of Ready, Definition of Done, Vergleich)
- Tipps zum Kapitel
- Fragen zum Kapitel

Seite 21-48

Was ist eine User Story? (1/4)
Grundsätzliches

Das zentrale Element zur Erfassung und Beschreibung von Anforderungen beim Agilen Requirements Engineering ist die User Story – eine einzelne User Story beschreibt dabei eine (einzelne) Funktionalität.
Eine User Story wird üblicherweise auf ein Karte („Story Card“, z.B. im DINA5- oder DINA6-Format) geschrieben und an ein Task Board gehängt, wo sie weiterbearbeitet wird.

Für die Formulierung einer User Story kann folgender (Satz-)Aufbau verwendet werden:

„Als <Rolle>
möchte ich <Ziel/Wunsch>,
um <Nutzen zu erzielen>“

Die User Story wird üblicherweise durch den Product Owner formuliert, kann aber auch durch das gesamte Team erstellt werden.

Was ist eine User Story? (2/4)
Definitionen

„Eine User Story („Anwendererzählung“) ist eine in Alltagssprache formulierte Software-Anforderung. Sie ist bewusst kurz gehalten und umfasst in der Regel nicht mehr als zwei Sätze.“ /#Wiki-User-Story/

„Eine User Story beschreibt eine Funktionalität, die entweder für einen User oder einen Käufer eines Systems oder einer Software von Wert ist.“ /Cohn10/

„User Stories sind keine vertraglichen Verpflichtungen.“ /Cohn10/

„A user story is nothing more than an agreement that the customer and developers will talk together about a feature.” (Beck & Fowler)

„User Stories sind in der Sprache des Anwenders formulierte Anforderungen an das zu entwickelnde Softwaresystem. Diese Anforderungen besitzen einen konkreten und sichtbaren Mehrwert für den Kunden.“ /Wirdemann11/

Was ist eine User Story? (3/4)
Charakterisierung

User Stories …

- geben die fachliche Sicht des Benutzers/Anwenders wieder.
- sind einfach zu handhaben.
- rücken den Business Value (Geschäftswert) in den Vordergrund.
- dienen in erster Linie der Kommunikation.

Was ist eine User Story? (4/4)
Die 3 Cs

Die User Stories werden durch die 3 Cs (von Ron Jeffries und anderen Anfang des Jahrtausends entwickelt) charakterisiert.

Die 3 Cs sind:

- Card: User Stories werden auf Indexkarten geschrieben und enthalten nicht (unbedingt) alle Details zu der Anforderung
- Conversation: Die Details zu einer User Story werden in einer Diskussion zwischen Auftraggeber und Auftragnehmer (Entwicklungsteam) geklärt
- Confirmation: Es gibt Akzeptanzkriterien (User Acceptance Criterias), die für eine User Story die „korrekte“ Umsetzung sichern

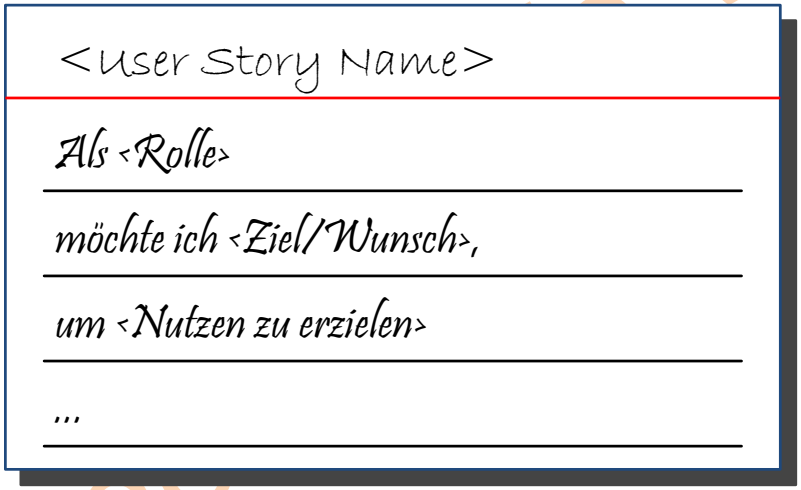
Aufbau und Aussehen einer User Story (1/3): Schema / Template

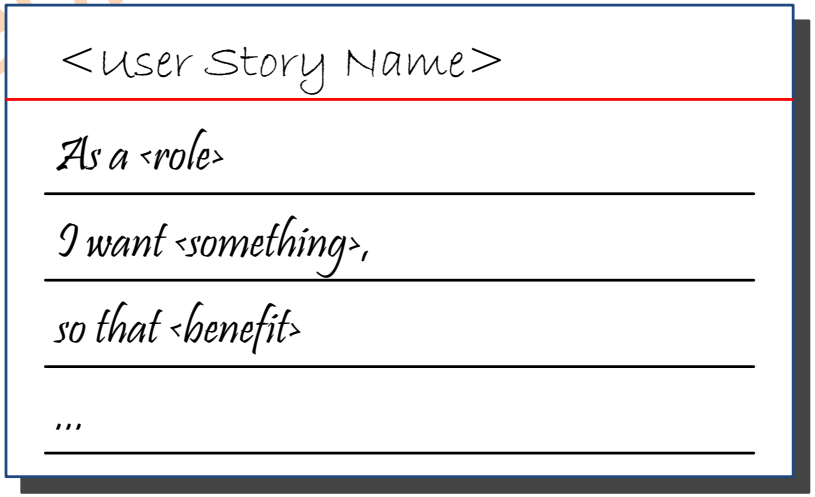
Der Kernsatz lautet:

„Als <Rolle> möchte ich <Ziel/Wunsch>, um <Nutzen zu erzielen>“

oder

“As a <role> I want <something>, so that <benefit>”





Aufbau und Aussehen einer User Story (2/3): Inhalte

Eine User Story wird üblicherweise auf der Story Card notiert und kann folgende Elemente enthalten:

- Eine Überschrift (den User Story Namen)
- Die eigentliche Story (mit Hilfe des Schemas aufgebaut)
- Akzeptanzkriterien, die bei der Abnahme herangezogen werden
- Eine Aufwandsabschätzung (meistens mit Story Points)
- Eventuell Testfälle
- Eventuell eine Priorität (Must have, Should have)
- Eventuell „Sonstiges“ (ergänzende Beschreibungen wie Mock-ups)

Nicht alle Elemente/Felder werden zu Beginn erfasst.

Aufbau und Aussehen einer User Story (3/3): Beispiele

Hier sind zwei Beispiele für User Stories (in der Kurzfassung) aufgeführt.

Präsentation

Als Leser

möchte ich die Präsentation mit einem

frei verfügbaren pdf-Reader lesen können

damit ich schnell darauf zugreifen kann

Backautomat

Als Kunde

möchte ich nicht länger als

3 Minuten auf mein fertiges Brot warten

da ich ansonsten abgepackte Ware kaufe

Weiteres zu User Stories (1/4): INVEST

(Gute) User Stories haben Eigenschaften, die den INVEST-Vorgaben folgen (/INVEST/, /Cohn10/). Dabei steht das Acronym INVEST für folgende Eigenschaften:

	Kurzform	Bedeutung
I	Independent	User Stories sollen unabhängig voneinander sein.
N	Negotiable	User Stories sollen verhandelbar sein.
V	Valuable	User Stories sollen einen Wert für den Kunden haben.
E	Estimatable	User Stories sollen schätzbar sein.
S	Small	User Stories sollen klein sein.
T	Testable	User Stories sollen testbar sein.

Weiteres zu User Stories (2/4): Probleme und Stolpersteine

In /#PMag-User-Stories-12/ werden folgende Probleme bei der Erstellung und Verwendung von User Stories genannt:

1. Formale „Stolpersteine“

- Fachliche Abhängigkeiten nicht bedacht
- User Stories zu groß gewählt

2. Strukturelle „Stolpersteine“

- Feste Ansprechpartner in den Fachbereichen fehlen
- Unterschiedliches Verständnis vom Inhalt einer Anforderung
- Anforderer gibt technische Lösung vor
- Ineffiziente Sprint Reviews
- Unvorhersehbare Veränderungen im Projektumfang

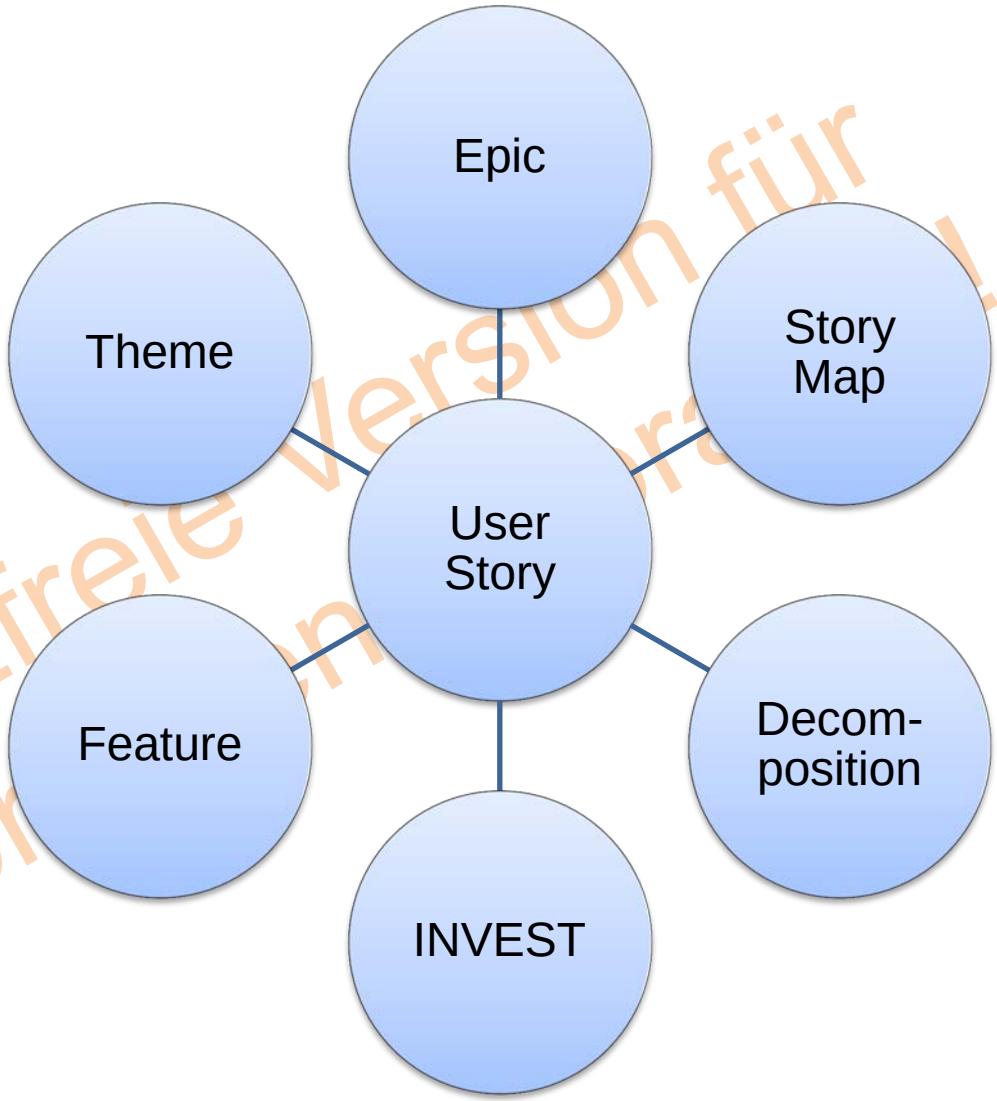
Weiteres zu User Stories (3/4): Abgrenzung User Story und Use Case

„Die Use Cases in der klassischen Softwareentwicklung ohne agile Methoden sind User Stories insofern ähnlich, als dass sie Anforderungen in der Sprache des Anwenders und im Kontext darstellen.

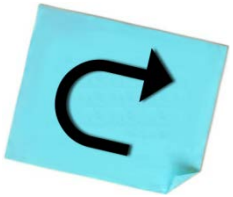
Bei einem Use Case werden jedoch alle Erfolgs- und Misserfolgsszenarien bei der möglichen Erreichung eines fachlich relevanten Ziels gebündelt dargestellt. Eine User Story stellt hingegen eine fachlich motivierte Anforderung dar, die von einem Anwender zwar als erfolgreich bzw. nicht erfolgreich umgesetzt beurteilt werden kann, allerdings wird der Anwender allein mit der User Story kein fachliches Ziel erreichen können.

Somit kann ein Use Case den Kontext für viele User Stories bilden: Eine User Story ist die Überschrift eines konkreten Szenarios, ein Use Case beinhaltet mehrere dieser Szenarien.“ /#Wiki-User-Story/

Weiteres zu User Stories (4/4): Begriffe rund um die User Story



Übung: Sind das (gute) User Stories? (1/2) – Aufgabe



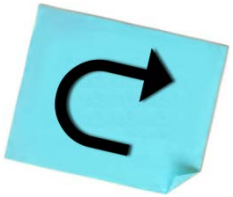
Überprüfen Sie, ob folgende Sätze eine User Story darstellen.

1. Als Online-Kunde möchte ich innerhalb von 10 Sekunden die Website angezeigt bekommen, um so direkt buchen zu können.
2. Die Anwendung soll mit Java realisiert werden, um so besser gepflegt werden zu können.
3. Als Arbeitsuchender will ich mehrteilige Lebensläufe anlegen, bearbeiten und löschen können, damit ich sie Arbeitgebern zeigen kann. /#AK-Req-ARE-Stam-11/

Dauer:
15 Min.

Lösung auf der nächsten Folie!

Übung: Sind das (gute) User Stories? (2/2) – Lösung



1. Das ist eine gute User Story.
2. Das ist (wahrscheinlich) keine gute User Story: Warum soll in Java realisiert werden? Der Mehrwert für den Kunden ist nicht erkennbar, es sein denn, die Pflege obliegt dem Kunden.
3. Ja, dies ist jedoch eine sehr große Story (= Epic) und sollte zerlegt werden.

Epics

Als Epics werden User Stories bezeichnet, die „sehr groß“ oder „grob-granular“ sind. Dies kann bedeuten:

Epics …

- sind zu groß, um in einem Sprint umgesetzt werden zu können.
- können (nochmals) in kleinere User Stories unterteilt werden.
- werden für eine späte Zukunft geplant.
- sind eine Zusammenfassung mehrerer, verwandter User Stories.
- können eventuell nicht abgeschätzt werden.

Um Epics in User Stories „runterzubrechen“ wird die Story Decomposition eingesetzt.

Das Schneiden von User Stories (1/2): Story Decomposition

„Story Decomposition stellt sicher, dass die User Stories in einem adäquaten Detaillierungsgrad beschrieben werden, und dass die User Stories sich aus den Anwenderzielen ableiten. Dabei wird von der Breite in die Tiefe vorgegangen, um die Anforderungen nach und nach zu verfeinern: Ausgangspunkt bilden die zu erreichenden Anwenderziele; daraus werden Epics abgeleitet, die in User Stories zerlegt werden; die User Stories werden um Akzeptanzkriterien ergänzt.“ /#Wiki-User-Story/

Das Schneiden von User Stories (2/2): Das Story Splitting Cheat Sheet

Das Story Splitting Cheat Sheet /#Story-Split-Cheat/ liefert neun Patterns zur Zerlegung von User Stories (die hier nicht erläutert werden):

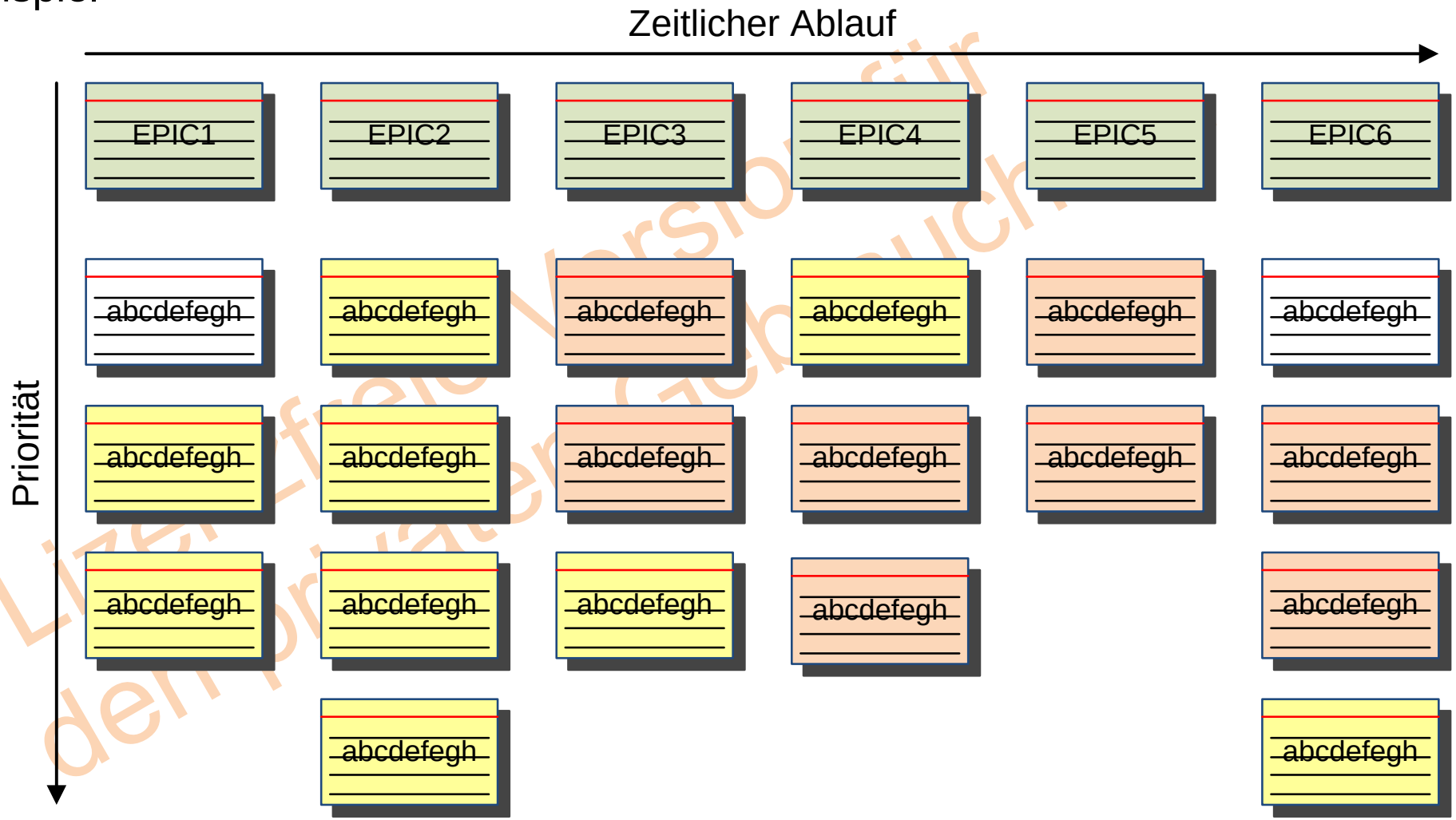
		Aufteilen, Zerlegen, Schneiden nach …
1	Workflow Steps	Workflows
2	Business Rule Variations	Business-Use-Case-Varianten
3	Major Effort	Hoher Aufwand
4	Simple/Complex	Komplexität
5	Variations in Data	Datenstruktur
6	Data Entry Methods	GUI-Design
7	Defer Performance	Performance-Varianten
8	Operations (e.g. CRUD)	Grundfunktionalitäten
9	Break out a Spike	Spikes (Recherchen)

Story Maps (1/2): Beschreibung

„Eine Story Map zeigt User Stories in einer grafischen Übersicht. Horizontal werden die aufeinanderfolgenden Aktivitäten des Anwenders jeweils mit einer User Story dargestellt. Vertikal wird von oben nach unten detailliert: z.B. angefangen bei den Kundenzielen über Epics bis hin zu den User Stories. Durch eine Story Map wird ein Überblick über alle User Stories hergestellt.“ /#Wiki-User-Story/

Story Maps (2/2): Darstellung

Beispiel



Themes

Themes fassen grob größere Bereiche einer (zu erstellenden) Anwendung zusammen, die thematisch zusammenhängen. Themes können somit mehrere Epics (und damit auch User Stories) umfassen.
Über Themes werden häufig Release-Planungen vorgenommen. Mit „Theme screening“ und „Theme scoring“ können die Themen grob bewertet werden, um den Planungsprozess zu unterstützen.

Features

Features bezeichnen die (von den Anwendern gewünschten) Merkmale des zu erstellenden Produkts.

Meistens werden sie in Form von Epics erfasst und dann entsprechend über User Stories realisiert.

Akzeptanzkriterien

Um feststellen zu können, ob nach der Umsetzung die Anforderungen an eine User Story auch wirklich eingebaut wurden, werden Akzeptanzkriterien (oder auch Abnahmekriterien) definiert. Diese beschreiben, was bei einer Abnahme überprüft und somit im Vorfeld getestet werden soll.

Typisches Format:

„Teste, ob …“

oder das (Given, When, Then, And, ...)-Schema

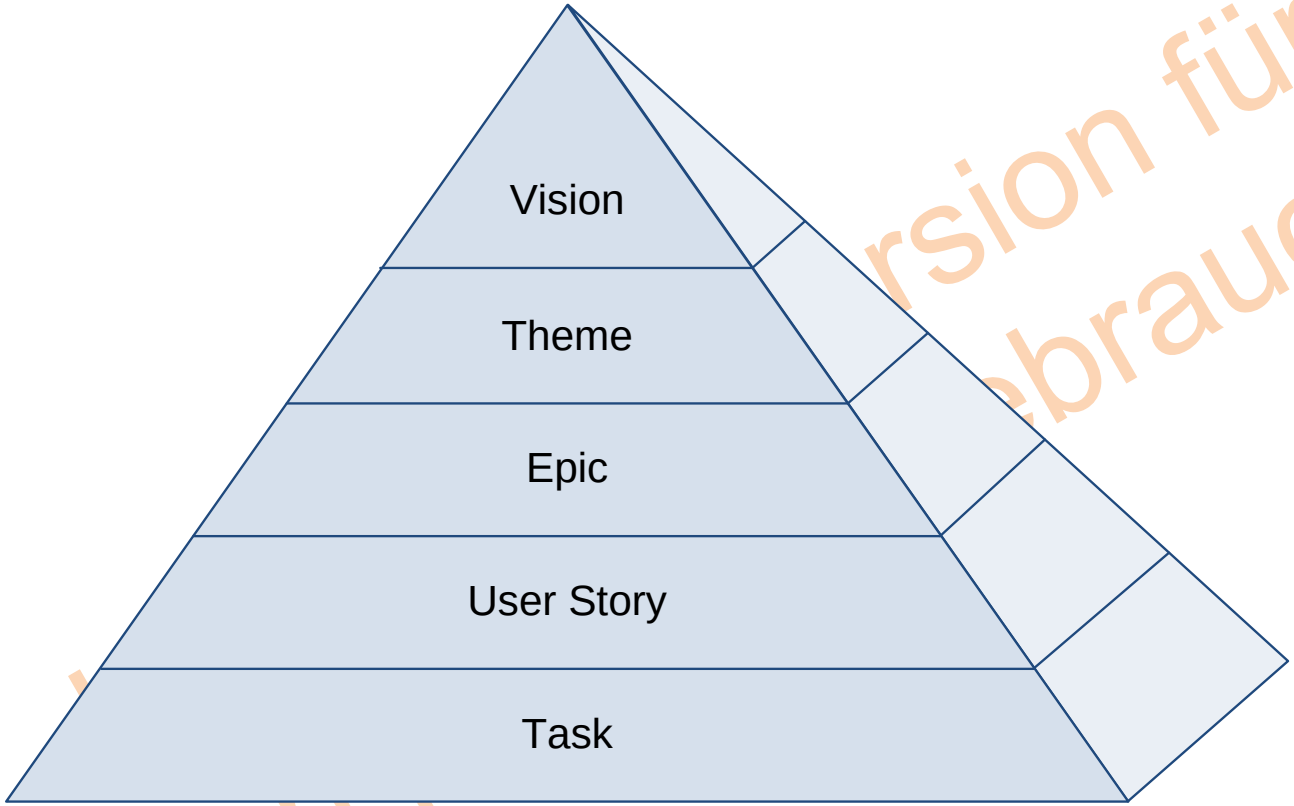
Gegeben ist, dass ein Kunde …,

wenn …,

dann …

und …

Die ARE-Pyramide



Die unterschiedlichen Betrachtungshöhen der einzelnen Gliederungsebenen beim ARE können über eine Pyramide dargestellt werden.

Definition of … (1/3): Definition of Ready

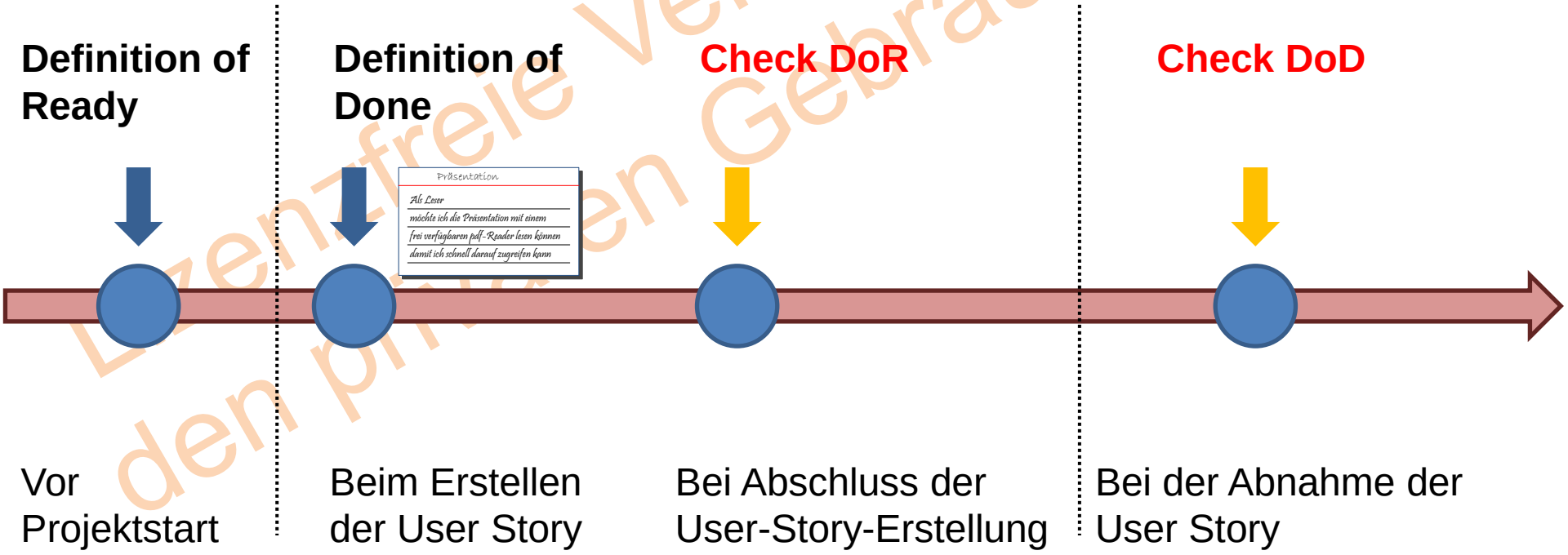
Ist eine User Story geschrieben, so soll sie auch in die Umsetzung gelangen. Um sicherzustellen, dass sie auch eine gewisse Qualität besitzt, die es dem Entwicklungsteam erlaubt ohne zu großen Aufwand die Umsetzung vorzunehmen, wird eine Definition of Ready (DoR) verwendet, die übergreifend für alle User Stories gilt. Hierüber wird geregelt, welchen Vorgaben eine User Story folgen muss.

Gerade bei Akzeptanz- und Testkriterien treten in der Praxis häufig Probleme auf: Hier kann die Definition of Ready helfen.

Die Definition of Ready ist kein allgemeiner Standard und unter Fachexperten umstritten, da schriftlich formulierte Regeln auch auf mangelhafte Kommunikation hindeuten könnten.

Definition of … (2/3): Definition of Done

Die Definition of Done (DoD) beschreibt (im Vorhinein), wann eine User Story als fertig gilt, das heißt, vom Anwender abgenommen werden kann. Die (universelle) Überprüfung der DoD-Kriterien findet nach der vollständigen Erstellung der User Story statt. Üblicherweise wird die DoD über eine (Check-)Liste realisiert.



Definition of … (3/3): Vergleich

	Ready	Done
Kurzcharakterisierung	Qualitätskriterien für Anforderungen	Abnahmekriterien für User Stories
Bezug	Bezieht sich auf die Anforderungen	Bezieht sich auf die entwickelte Software
Wesentlich	Vollständigkeit der User Story / Story Card inkl. Priorisierung (und Schätzung)	Qualitätskriterien: Die entwickelte Software muss getestet sein und funktionieren
Ziel	„Abnahme“ einer User Story vor dem Sprint	„Abnahme“ der Umsetzung einer User Story
Wird wann erstellt?	Vor dem Projekt	Vor dem Projekt
Wird wann überprüft?	Nach der Erstellung der User Story vor Sprintbeginn	Während und nach der Umsetzung (Sprint Review) der User Story im Sprint
Wer ist verantwortlich?	Product Owner	Entwicklungsteam

Tipps zum Kapitel

- User Stories sind das zentrale Element zur Erfassung von Anforderungen im Agilen Requirements Engineering. Daher sollte ein gemeinsames Verständnis aller Beteiligten im Umgang mit User Stories angestrebt werden
- User Stories dienen in erster Linie der Kommunikation und nicht der schriftlichen Fixierung von Sachverhalten
- Auch wenn die User Stories (über die Erfassung aus Story Cards) eher „klein“ erscheinen, so können die dazugehörigen Test- und Akzeptanzszenarien einen erheblichen Umfang annehmen

Fragen zum Kapitel

1. Was sind User Stories?
2. Nach welchem Schema werden die User Stories im Allgemeinen notiert?
3. Wer schreibt die User Stories?
4. Wie groß sollte eine User Story sein?
5. Wofür steht das Akronym INVEST?
6. Was sind Epics?
7. Was sind Themes?
8. Wozu dienen die Definition of Ready und die Definition of Done? Wie hängen sie zusammen?

Kapitel 3: Schätzen

Kapitel 3

- Warum Schätzen?
- Story Points
- Klassisches und agiles Schätzen (Beschreibung, Gegenüberstellung)
- Agile Schätzmethoden
- Planning Poker (Beschreibung, Bewertungsskala für Story Points, Vorgehensweise)
- Magic Estimation (Beschreibung, Vorgehensweise)
- Tipps zum Kapitel
- Fragen zum Kapitel

Seite 49-61

Warum Schätzen?

Im Normalfall wird in Projekten nachgefragt, wie groß der Aufwand zur Realisierung (bestimmter Teile) ist.

Die zentralen Fragen lauten:

- **Wie groß ist der Aufwand (und damit: wie hoch sind die Kosten)?**
- **Wann ist es fertig?**

Zum **Schätzen von Aufwänden (in Projekten)** gibt es eine eigenständige Präsentation des Autors, die ebenfalls auf der Website unter http://www.peterjohann-consulting.de/_pdf/peco-pm-schaetzen.pdf frei herunterladbar ist.

Story Points

Story Points sind ein Schätzmaß für die Größe einer User Story. Üblicherweise werden Story Points nicht in absoluten Größen angegeben, sondern in einem relativen Maß.

„Story Points sind eine Einheit, die die Größe einer User Story beschreiben."

User Stories sollten so groß sein, dass sie innerhalb eines Sprints umgesetzt werden können, d.h. innerhalb von ein bis vier Wochen.

Klassisches und agiles Schätzen (1/2): Beschreibung

Beim klassischen Requirements Engineering wird auf Basis der mehr oder weniger ausformulierten Anforderungen geschätzt, wie lange die Umsetzung jeweils dauert. Hierzu gibt es einige klassische Schätzverfahren. Je größer oder komplexer das Projekt und je länger die Projektdauer umso größer ist die Schätz**un**genauigkeit. Jedoch liegt bei Projektstart mit dem Projektplan auch eine Gesamtabschätzung vor.

Beim Agilen Requirements Engineering sind in erster Linie diejenigen User Stories aus dem Product Backlog wichtig, die unmittelbar zur Umsetzung gelangen: nur diese werden detailliert im Estimation Meeting (Schätzmeeting) geschätzt. Da beim Schätzen alle notwendigen Details besprochen werden, ist die Schätzgenauigkeit für den nächsten Umsetzungsschritt hoch. Für das Gesamtprojekt wird jedoch keine Aussage getroffen.

Klassisches und agiles Schätzen (2/2): Gegenüberstellung

	Klassisch	Agil / Scrum
Basis	Mehr oder weniger ausformulierte Anforderungen	User Stories
Was wird geschätzt?	Personentag – Aufwand, (Dauer, Kosten)	Story Points – Aufwand als Größenordnung
Wann wird geschätzt?	Zu verschiedenen Zeitpunkten, aber immer (auch) vor Projektstart	Vor einem Sprint
Wer schätzt?	Experten-Team	(Scrum) Team
Wie ist die Genauigkeit?	Je nach Zeitpunkt und Vorgaben sowie Vorlagen	Hoch
Wie ist die Verbindlichkeit?	Mittel	Hoch
Häufig eingesetzte Werkzeuge und Methoden	Schätzrunden mit Experten; Planauswertungen	Planning Poker; Magic Estimation
Weiterverwendung der Schätzergebnisse	Projektcontrolling	Burndown Chart

Agile Schätzmethoden

Folgende Methoden werden beim Schätzen in agilen Vorgehen angewandt:

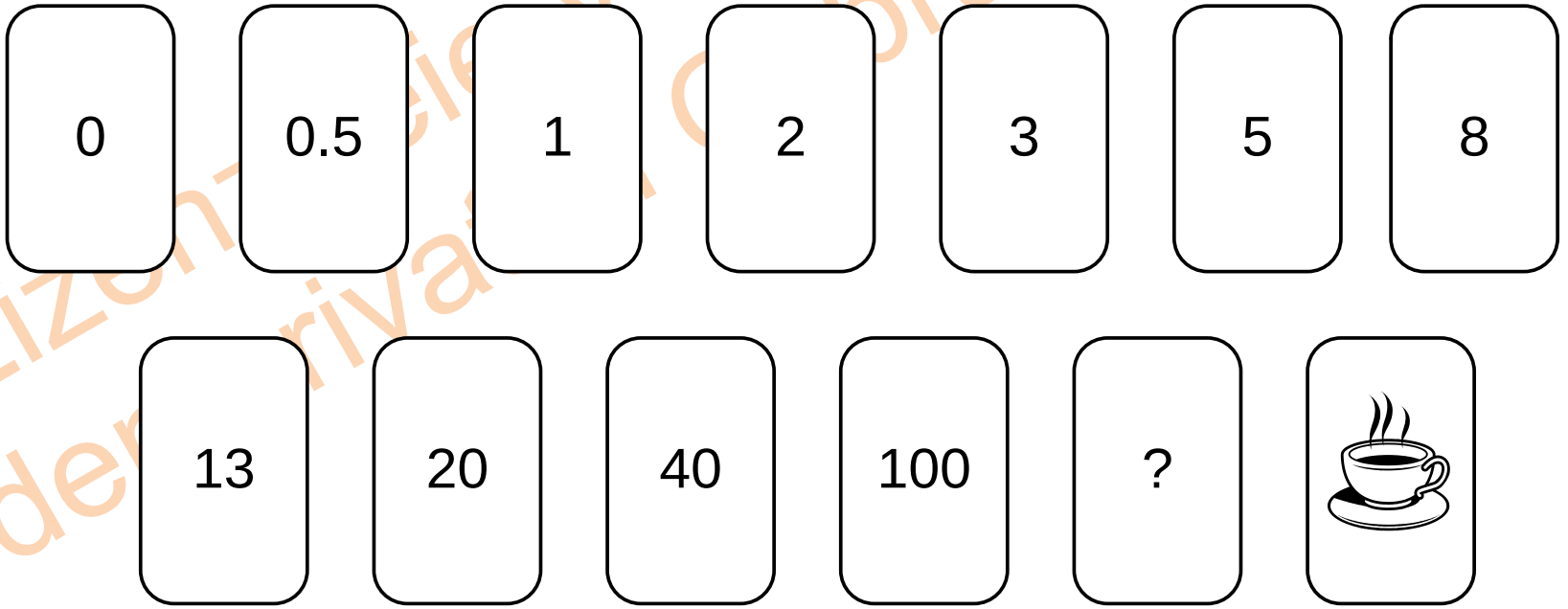
- Planning Poker
- Magic Estimation
- Affinity Estimation

Die ersten beiden Schätzmethoden werden hier kurz vorgestellt, die dritte wird nur der Vollständigkeit halber erwähnt.

Wesentlich ist, dass beim agilen Schätzen immer relative Größen („Komplexitätswerte“) und nicht absolute Werte, wie Aufwand in Personentagen, geschätzt werden.

Planning Poker (1/3): Beschreibung

Planning Poker (Schätzpoker) ist eine Schätzmethode zum Abschätzen einzelner User Stories, die das ganze Team einbezieht. Hierzu wird ein Schätzmeeting durchgeführt, bei dem jedes Teammitglied ein Kartenset mit 13 Karten erhält (mit aufgedruckten Werten von 0 bis 100, zudem noch das Fragezeichen und die Kaffeetasse) und pro User Story eine Bewertung abgibt. Die Bewertungsskala und die Vorgehensweise werden auf den nächsten Folien beschrieben.



Planning Poker (2/3): Bewertungsskala für Story Points

Die Story Points werden in relativen Maßen angegeben – die Skala könnte sein:

Story Points	Beschreibung
0	praktisch kein Aufwand
0.5	kaum wahrnehmbarer Aufwand
1	kleinster Aufwand
2	sehr kleiner Aufwand, etwa zwei kleinste Aufwände zusammen
3	kleiner Aufwand, etwa drei kleinste Aufwände zusammen
5	mittlerer Aufwand
8	großer Aufwand
13	sehr großer Aufwand
20	riesiger Aufwand
40	sehr riesiger Aufwand
100	viel zu großer, nicht mehr zu überblickender Aufwand
?	nicht einschätzbar
Kaffeetasse	zeigt an, dass eine Pause gewünscht wird

Planning Poker (3/3): Vorgehensweise

Es wird beim Schätzmeeting (Estimation Meeting) eine Tischrunde gebildet, an dem alle Teammitglieder sitzen. Jedes Teammitglied verfügt über ein Planning-Poker-Kartenset.

Nun wird die zu schätzende User Story in die Tischmitte gelegt und kurz (durch den Product Owner) erläutert. Dann gibt jedes Teammitglied eine Schätzung ab, indem eine verdeckte Karte mit dem geschätzten Wert auf den Tisch gelegt wird. Haben alle Teammitglieder ihre Schätzung abgegeben, werden alle Karten gleichzeitig aufgedeckt.

Nach dem Aufdecken der Karten wird geschaut, ob große Abweichungen in der Einschätzung vorhanden sind. Wenn nicht, so wird der häufigste Schätzwert übernommen und auf die Story Card eingetragen. Sind aber erhebliche Unterschiede zu erkennen, so muss über diese unterschiedlichen Bewertungen diskutiert werden. Bei großen Abweichungen kann nach der Diskussion eine erneute Schätzung (in einer weiteren Schätzrunde) vorgenommen werden, so dass ein abschließender Wert ermittelt werden kann.

Magic Estimation (1/2): Beschreibung

Magic Estimation ist eine weitere Schätzmethode in der agilen Welt, die sich neben dem Planning Poker etabliert hat. Es dient insbesondere dazu, größere Backlogs abzuschätzen, da es eine schnell durchführbare Methode ist und somit viele Backlog Items in kurzer Zeit abgeschätzt werden können.

Die Durchführung ist ähnlich wie beim Planning Poker: Es werden User Stories von den Teammitgliedern in einem Schätzmeeting abgeschätzt.

Magic Estimation (2/2): Vorgehensweise

In dem Schätzmeeting werden die abzuschätzenden User Stories durch den Product Owner vorgestellt und jede User Story an ein einzelnes Teammitglied weitergereicht. Wenn alle User Stories (gleichmäßig) verteilt sind, beginnt jedes Teammitglied mit der Abschätzung.

Hierzu werden die Story Cards (User Stories) in ein Raster (Story Points in einer Skala, vergleiche Planning Poker) eingeordnet, welches typischerweise auf dem Fußboden oder an Flipcharts untergebracht ist.

Nun wird das Team aktiv: Jedes Teammitglied überprüft, ob es den einzelnen Abschätzungen zustimmt. Stories mit abweichender Schätzung werden herausgenommen („herausgeschoben“), anschließend im gesamten Team diskutiert und erneut zugeordnet.

Tipps zum Kapitel

- Die agilen Schätzverfahren können vergleichsweise schnell und einfach eingesetzt werden – auch unabhängig von agilen Kontexten
- Versuchen Sie nicht, Story Points in Personentage umzurechnen

Fragen zum Kapitel

1. Warum funktionieren die klassischen Schätzverfahren in agilen Umfeldern nicht?
2. Wer schätzt die User Stories?
3. Was ist Planning Poker?
4. Was sind die Vorteile des Magic Estimation gegenüber dem Planning Poker? Was ist nachteilig?

Kapitel 4: Verfolgen

- Warum Verfolgen?
- Das Task Board (Grundsätzliches, Darstellung)
- Das Burndown Chart
- Velocity (Beschreibung, Grafik)
- Das Backlog Grooming / Refinement
- Wohin mit den fertigen Anforderungen / User Stories?
- Tipps zum Kapitel
- Fragen zum Kapitel

Kapitel 4

Seite 62-72

Warum Verfolgen?

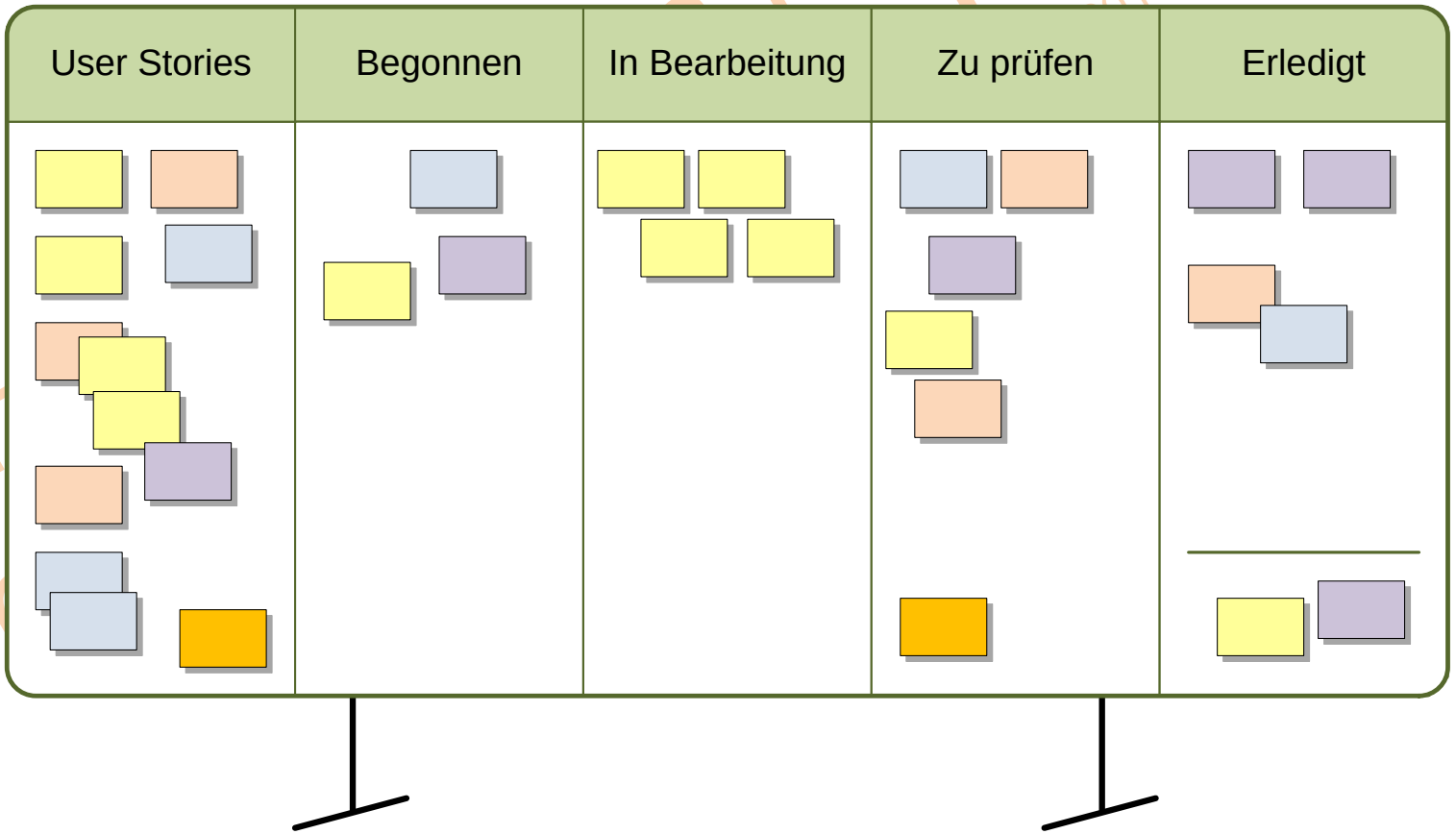
Bei der Umsetzung der einzelnen User Stories im Sprint ist es interessant zu erfahren, welche User Stories schon umgesetzt wurden und wie die generelle Umsetzungsgeschwindigkeit ist.

Hierzu gibt es bei den agilen Ansätzen einige Hilfsmittel, die hier zum Teil vorgestellt werden.

Was	Verfolgen	durch wen; Hilfsmittel
User Stories im Product Backlog vor der Umsetzung	Nein, nur indirekt (Releaseplanung)	Product Owner
User Stories im Sprint Backlog während der Umsetzung	Ja	Entwicklungsteam; Task Board und Burndown Chart
User Stories nach der Umsetzung	Offen	Offen

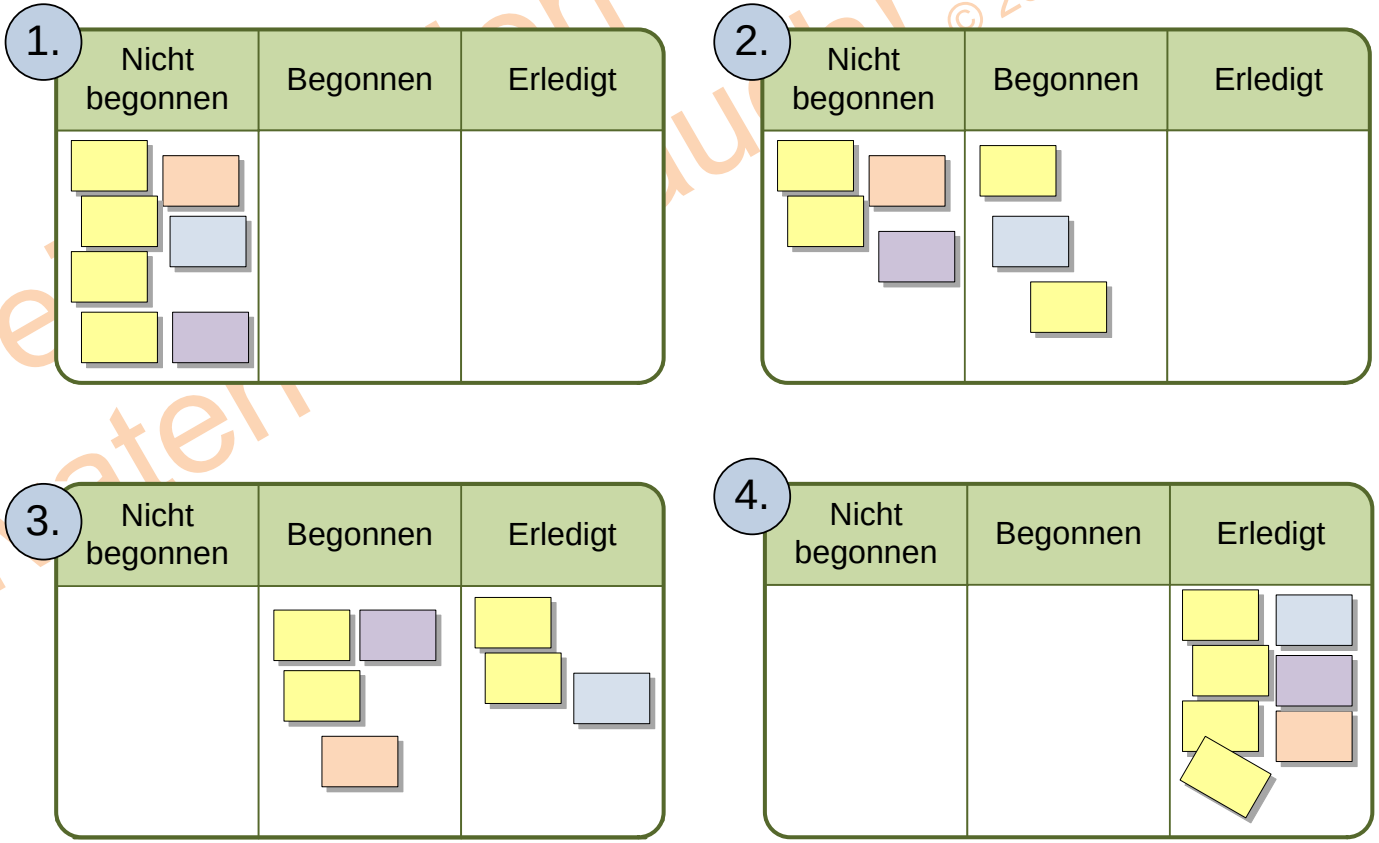
Das Task Board (1/2)
Grundsätzliches

In diesem Beispiel ist ein (typisches) Task Board dargestellt, bei dem in einem Sprint (der bereits begonnen hat) ein vierstufiger Bearbeitungsablauf gewählt wurde.



Das Task Board (2/2)
Darstellung

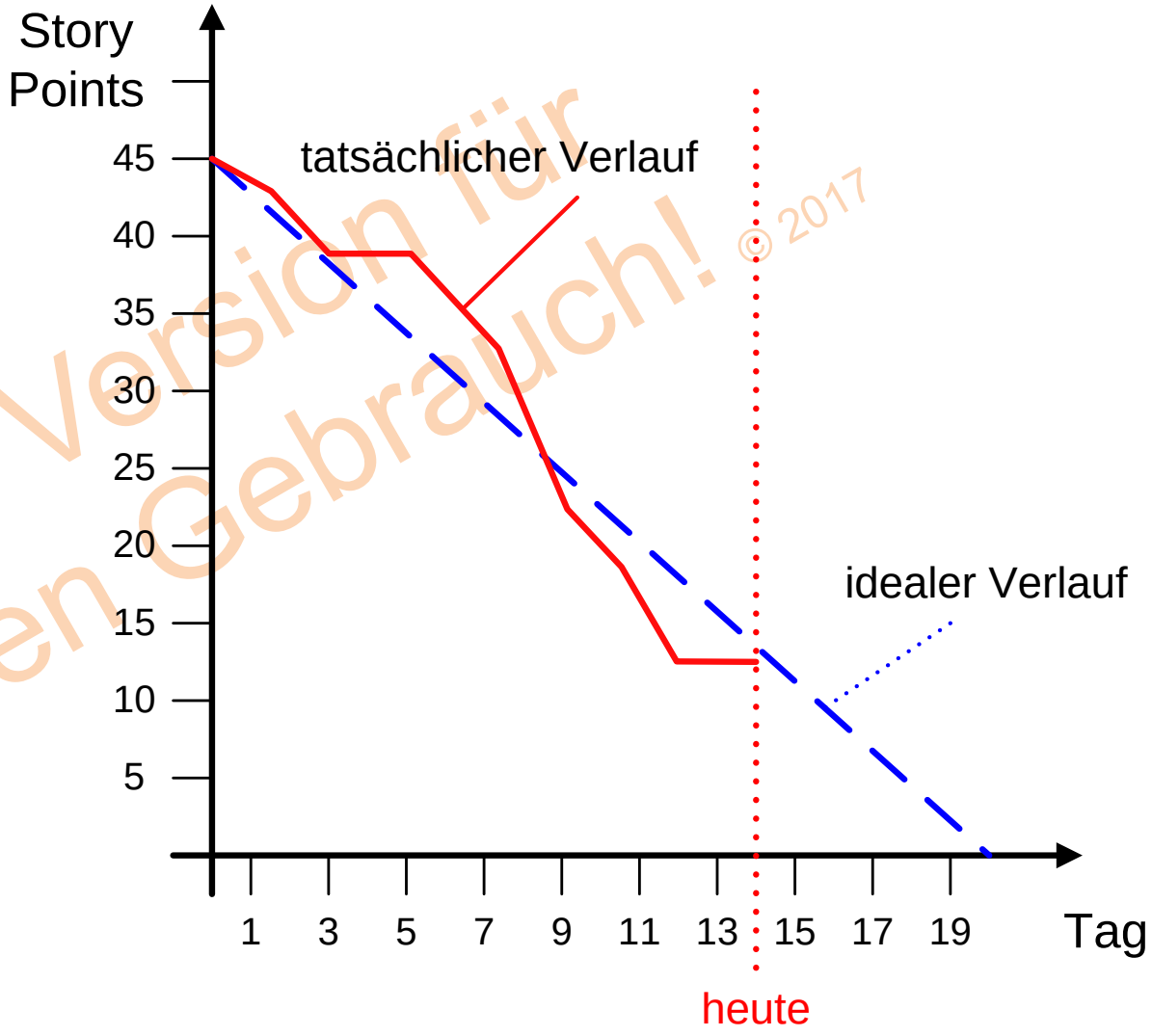
Das einfache Task Board wird insbesondere bei der Umsetzung der User Stories im Sprint herangezogen. Dabei „wandern“ die Story Cards von links („Nicht begonnen“) nach rechts („Erledigt“).



Das Burndown Chart

Das hier dargestellte Sprint Burndown Chart zeigt den Verlauf eines 20-Tage-Sprints bis zum 14ten Tag. Die Geschwindigkeit des Teams beträgt 45 Story Points pro Sprint.

Zum Stichtag („heute“) entsprechen die tatsächlich umgesetzten Story Points dem idealen Verlauf.



Velocity (1/2): Beschreibung

Die Velocity ist das (relative) Maß für die Umsetzungsgeschwindigkeit beim agilen Vorgehen. Die Velocity wird dabei definiert als

Velocity = Story Points pro Iteration

wobei Iteration in Scrum mit Sprint gleichzusetzen ist.

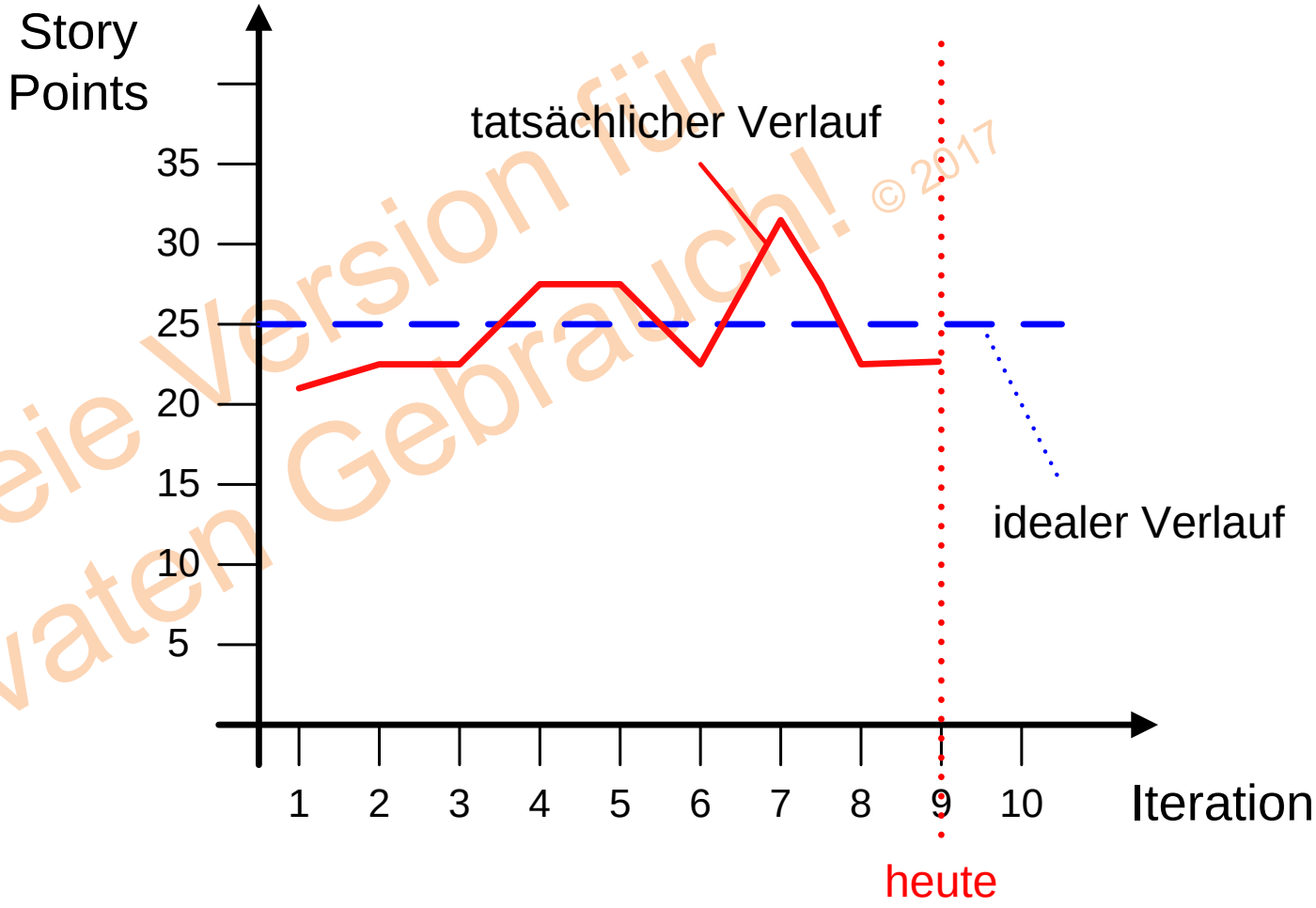
Zu Beginn eines Projekts liegt die erwartete Velocity noch nicht fest; diese wird über mehrere Iterationen dann eingeschätzt:

Anzahl benötigter Iterationen = Summe der Story Points / Velocity

Eine (typische) Entwicklung der Velocity ist auf der nachfolgenden Grafik dargestellt.

Velocity (2/2): Grafik

Das hier dargestellte Velocity Chart zeigt die Entwicklung der Velocity bis zur neunten Iteration.
Die Geschwindigkeit des Teams pegelt sich bei 25 Story Points pro Iteration (Sprint) ein.



Das Backlog Grooming / Refinement

Als Backlog Grooming (von to groom = pflegen/putzen; oder neuerdings auch: Backlog Refinement) wird das Überarbeiten / Überprüfen des Backlogs bezeichnet. Dieses sollte in regelmäßigen Abständen durch den Product Owner und das Entwicklungsteam durchgeführt werden.

Ziele:

- Überprüfung, ob die User Stories noch so stimmen („gültig sind“)
- Herausfinden / Streichen überflüssiger Backlog Items (User Stories)

Wohin mit den fertigen Anforderungen / User Stories?

Was geschieht mit den fertigen User Stories, d.h. den User Stories, die bereits umgesetzt worden sind? Im klassischen Requirements Engineering erhalten sie einen Status („umgesetzt“) und werden dann häufig in Abschluss-dokumentationen oder Benutzerhandbüchern weiterverwendet.

Im Agilen Requirements Engineering gibt es keine Aussage dazu, was mit den umgesetzten User Stories geschehen soll. Jedoch spricht nichts dagegen, hieraus ebenfalls Abschlussdokumente zu erstellen. Allerdings muss beachtet werden, dass die User Stories nicht komplett ausformuliert sind und der Inhalt mündlich kommuniziert wurde. Um aus User Stories weitere Dokumente zu erstellen, müssten die Entwickler dazu den Auftrag erhalten und entsprechende Ressourcen zugewiesen bekommen.

Tipps zum Kapitel

- Setzen Sie gleich zu Beginn Ihrer Scrum-Aktivitäten ein Task Board auf
- Versuchen Sie nicht, aus der Velocity den Aufwand in Personentagen zu ermitteln
- Backlog Grooming gehört nicht zu den Lieblingsaufgaben der Entwickler. Machen Sie daher hieraus ein „Event“, um die Motivation der Mitarbeiter zu steigern

Fragen zum Kapitel

1. Was versteht man unter Verfolgen der Anforderungen im agilen Kontext?
2. Was ist das Burndown Chart?
3. Wie misst man die Velocity? Was sagt sie aus?
4. Warum sollte man nicht aus der Velocity den Aufwand in Personentagen ermitteln?

Anhang:
Literatur, Weblinks, Sprüche und Kontakt

- Literatur
- Weblinks
- Sprüche
- Glossar – Top-Ten-Begriffe zum ARE
- Weitere öffentliche Präsentationen des Autors
- Meine Management-Wolke – Das kann ich für Sie tun
- Kontakt zum Autor

Anhang

Seite 73-84

Literatur (1/2)

/Adzic14/ Gojko Adzic, David Evans: Fifty Quick Ideas to Improve Your User Stories, Neuri Consulting, London 2014, ISBN 978-0-9930881-0-0

/Bergsmann14/ Johannes Bergsmann: Requirements Engineering für die agile Softwareentwicklung: Methoden, Techniken und Strategien zur erfolgreichen Umsetzung, dpunkt, Heidelberg 2014, ISBN 978-3-86490-149-2

/Cohn04/ Mike Cohn: User Stories Applied: For Agile Software Development, Addison-Wesley Longman, Amsterdam 2004, ISBN 978-0-321-20568-1

/Cohn05/ Mike Cohn: Agile Estimating and Planning, Prentice Hall International, Upper Saddle River, New Jersey 2005, ISBN 978-0-13-147941-8

/Cohn09/ Mike Cohn: Succeeding with Agile: Software Development Using Scrum, Addison-Wesley Longman, Amsterdam 2009, ISBN 978-0-321-57936-2

/Cohn10/ Mike Cohn: User Stories: für die agile Software-Entwicklung mit Scrum, XP u.a., mitp-Verlag, Bonn 2010, ISBN 978-3-8266-5898-3

/Ebert14/ Christof Ebert: Systematisches Requirements Engineering. Anforderungen ermitteln, dokumentieren, analysieren und verwalten, dpunkt, Heidelberg 5. Auflage 2014, ISBN 978-3-86490-139-3

Literatur (2/2)

/Gloger13/ Boris Gloger: Scrum: Produkte zuverlässig und schnell entwickeln, Hanser, München 4. Auflage 2013, ISBN 978-3-446-43338-0

/Gloger14/ Boris Gloger: Wie schätzt man in agilen Projekten – oder wieso Scrum-Projekte erfolgreicher sind, Hanser, München 2014, ISBN 978-3-446-43910-8

/Leffing10/ Dean Leffingwell: Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise, Addison-Wesley Professional, Boston, Massachusetts 2010, ISBN 978-0-321-63584-8

/Opelt14/ Andreas Opelt, Boris Gloger, Wolfgang Pfarl, Ralf Mittermayr: Der agile Festpreis: Leitfaden für wirklich erfolgreiche IT-Projekt-Verträge Hanser, München 2. Auflage 2014, ISBN 978-3-446-44136-1

/Patton14/ Jeff Patton: Story Mapping: Discover the Whole Story, Build the Right Product, O'Reilly Media, Cambridge, Massachusetts 2014, ISBN 978-1-4919-0490-9

/Rüping13/ Andreas Rüping: Dokumentation in agilen Projekten: Lösungsmuster für ein bedarfsgerechtes Vorgehen, dpunkt, Heidelberg 2013, ISBN 978-3-86490-040-2

/Rupp14/ Chris Rupp: Requirements-Engineering und -Management. Aus der Praxis von klassisch bis agil, Hanser, München 6. Auflage 2014, ISBN 978-3-446-43893-4

/Wirdemann11/ Ralf Wirdemann: Scrum mit User Stories, Hanser, München 2. Auflage 2011, ISBN 978-3-446-42660-3

Weblinks (1/3)

Auf den folgenden Seiten sind Weblinks aufgeführt, die sich mit Agilem Requirements Engineering beschäftigen. Auf die Weblinks wird zum Teil in dieser Präsentation verwiesen. Eine Bewertung der Weblinks und deren Inhalte wird hier nicht vorgenommen, kann aber vom Autor abgefragt werden. Soweit nicht anders angegeben, sind die Inhalte der Weblinks kostenfrei zugänglich.

Die Weblinks sind folgendermaßen eingeteilt:

- Generelle Quellen
- Buchergänzungen
- Einzelartikel

Legende für die nachfolgenden Folien – so werden die Weblinks klassifiziert:
/ / Verweis auf Website generell
/*/ Verweis auf eine Website, die als Buch-Ergänzung dient
/#/ Verweis auf einzelnes Thema auf einer Website
/#V/ Verweis auf ein Video (auf einer Website) mit Minutenangabe und Sprache

Weblinks (2/3)

/#AK-Req-ARE-Stam-11/ Agile Requirements Engineering: User Stories, Schätzen, Planen; Foliensatz von Sebastian Stamminger (deutsch) www.gi-muc-ak-req.de/joomla/downloads/doc/71/raw; eingesehen am 10.05.2014

/#CodeCent-ARE/ Übersichtsseite zum ARE der Firma codecentric: https://www.codecentric.de/kompetenzen/agile-softwareentwicklung/agile-requirements-engineering/; eingesehen am 10.05.2014

/INVEST/ Original-Artikel zu INVEST von Bill Wake (2003): http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/; eingesehen am 10.05.2014

/#OBJspek-ARE-Braun-09/ OBJEKTspektrum 05/2009, S. 64-71: Peter Braun, Sabine Canditt: „Wenn der Kunde nicht weiß, was er will: Tipps für den agilen Umgang mit Anforderungen“: http://www.sigs-datacom.de/fachzeitschriften/objektspektrum/archiv/artikelansicht.html?tx_mwjournals_pi1%5bpointer%5d=0&tx_mwjournals_pi1%5bmode%5d=1&tx_mwjournals_pi1%5bshowUid%5d=6395; eingesehen am 10.05.2014

/#OBJspek-ARE-Mödl-11/ OBJEKTspektrum 06/2011, S. 18-23: Thomas Mödl, Susanne Mühlbauer: „Eine sachliche Romanze: Scrum und evolutionäres Requirements-Engineering“: http://www.sigs-datacom.de/fachzeitschriften/objektspektrum/archiv/artikelansicht.html?tx_mwjournals_pi1%5Bpointer%5D=0&tx_mwjournals_pi1%5Bmode%5D=1&tx_mwjournals_pi1%5BshowUid%5D=7002; eingesehen am 10.05.2014

Weblinks (3/3)

/#PMag-User-Stories-12/ ProjektMagazin 17/2012 (05.09.2012): „Anforderungs-management in IT-Projekten. So vermeiden Sie Stolpersteine bei User Stories“, Autor: Steffen Thols: https://www.projektmagazin.de/artikel/so-vermeiden-sie-stolpersteine-bei-user-stories_1073296; eingesehen am 31.01.2014

/Scrum-Glossar/ Das deutsche Scrum-Glossar: http://www.scrum-glossar.de; eingesehen am 10.05.2014

/#Story-Split-Cheat/ Das Story Splitting Cheat: Patterns zur Zerlegung von User Stories aus dem Jahr 2009 (englisch, kostenfrei herunterladbar): http://www.richardlawrence.info/wp-content/uploads/2009/10/Story-Splitting-Cheat-Sheet.pdf; eingesehen am 10.05.2014

/Wiki-d/ Deutsche Wikipedia: https://de.wikipedia.org; eingesehen am 31.01.2014

/Wiki-e/ Englische Wikipedia: https://de.wikipedia.org; eingesehen am 31.01.2014

/#Wiki-User-Story/ User Story in der deutschen Wikipedia: https://de.wikipedia.org/wiki/User_Story; eingesehen am 31.01.2014

/XING-ARE/ Gruppe zum ARE bei XING: https://www.xing.com/net/pri092765x/are; eingesehen am 31.01.2014

Sprüche

„Das Management interessiert sich nicht für Story Points.“ (unbekannt)

„Der Product Owner hat beim ARE einen harten Job: Von der Vision bis zum Detail alles im Griff behalten und dabei keine Macht über Schätzungen.“ (unbekannt)

„Für das ARE gilt: Miteinander reden statt gegeneinander schreiben.“ (unbekannt)

Glossar – Top-Ten-Begriffe zum ARE (1/2)

Begriff	Beschreibung	Quelle
Backlog	Ein Backlog (deutsche Übersetzung: Arbeitsrückstand, Rückstand, Nachholbedarf) beschreibt die noch zu betrachtenden oder zu erledigenden Tickets (Aufgaben) in einem System; dies können beim Product Backlog Epics oder User Stories sein	selbst
Definition of Done	Die Definition of Done (DoD) beschreibt (im Vorhinein), wann eine User Story als fertig gilt, das heißt, vom Anwender abgenommen werden kann. Üblicherweise wird die DoD über eine (Check-)Liste realisiert	selbst
Epic	Beschreibung einer Anforderung auf grober Ebene, im Allgemeinen in Alltagssprache verfasst. Ein Epic kann in User Stories aufgeteilt werden: Hierzu wird die Story Decomposition angewandt	selbst
Feature	Fähigkeit oder Gruppe von Fähigkeiten eines Produkts (oder einer Komponente eines Produkts)	selbst
Minimal Marketable Feature (MMF)	Ein Minimal Marketable Feature (MMF) ist die kleinstmögliche Menge von Features, die in Betrieb genommen werden können und einen Mehrwert für den Kunden darstellen	selbst

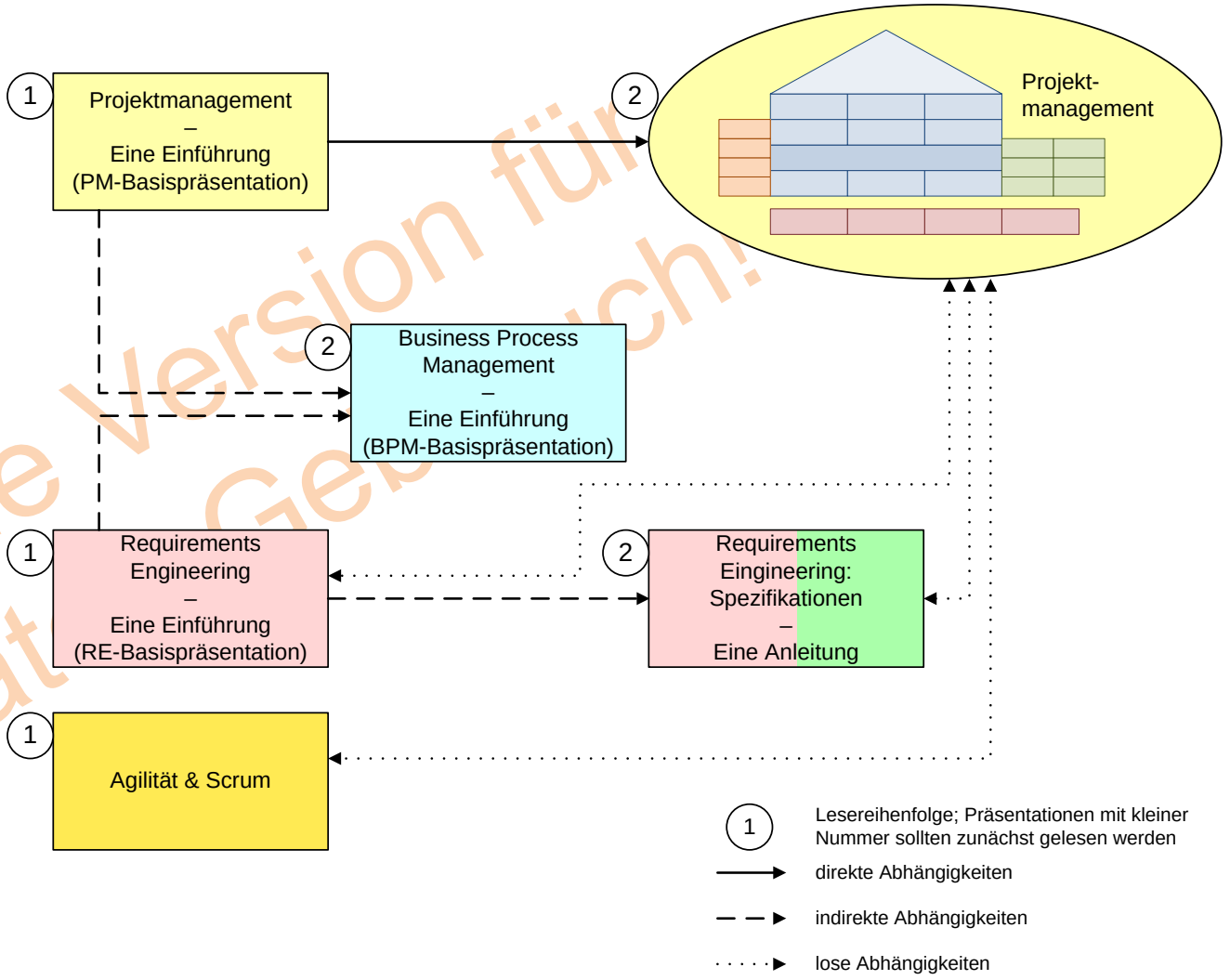
Glossar – Top-Ten-Begriffe zum ARE (2/2)

Begriff	Beschreibung	Quelle
Personas	Die Persona stellt einen Prototyp für eine Gruppe von Nutzern dar, mit konkret ausgeprägten Eigenschaften und einem konkreten Nutzungsverhalten	/Wiki/
Planning Poker	Methode zum Abschätzen von Aufwänden in einer Gruppe. Es werden hierzu Spielkarten genutzt, die zur Schätzung durch mehrere Einzelpersonen herangezogen werden	selbst
Story Map	Eine Story Map zeigt User Stories in einer grafischen Übersicht. Horizontal werden die aufeinanderfolgenden Aktivitäten des Anwenders jeweils mit einer User Story dargestellt. Vertikal wird von oben nach unten detailliert: z.B. angefangen bei den Kundenzielen über Epics bis hin zu den User Stories. Durch eine Story Map wird ein Überblick über alle User Stories hergestellt	/#Wiki-User-Story/
Theme	Ein Theme fasst Backlog Items zu einem Feature zusammen. Es werden auch die Begriffe Cluster oder Feature Group verwendet	selbst
User Story	Eine User Story ist eine Form der Beschreibung des Geschäftswerts von Arbeit für einen Auftraggeber. Üblicherweise wird zur Erstellung eine Schablone verwendet, die die Form "Als <Rolle> möchte ich <Ziel/Wunsch>, um <Nutzen> zu erzielen" aufweist	selbst

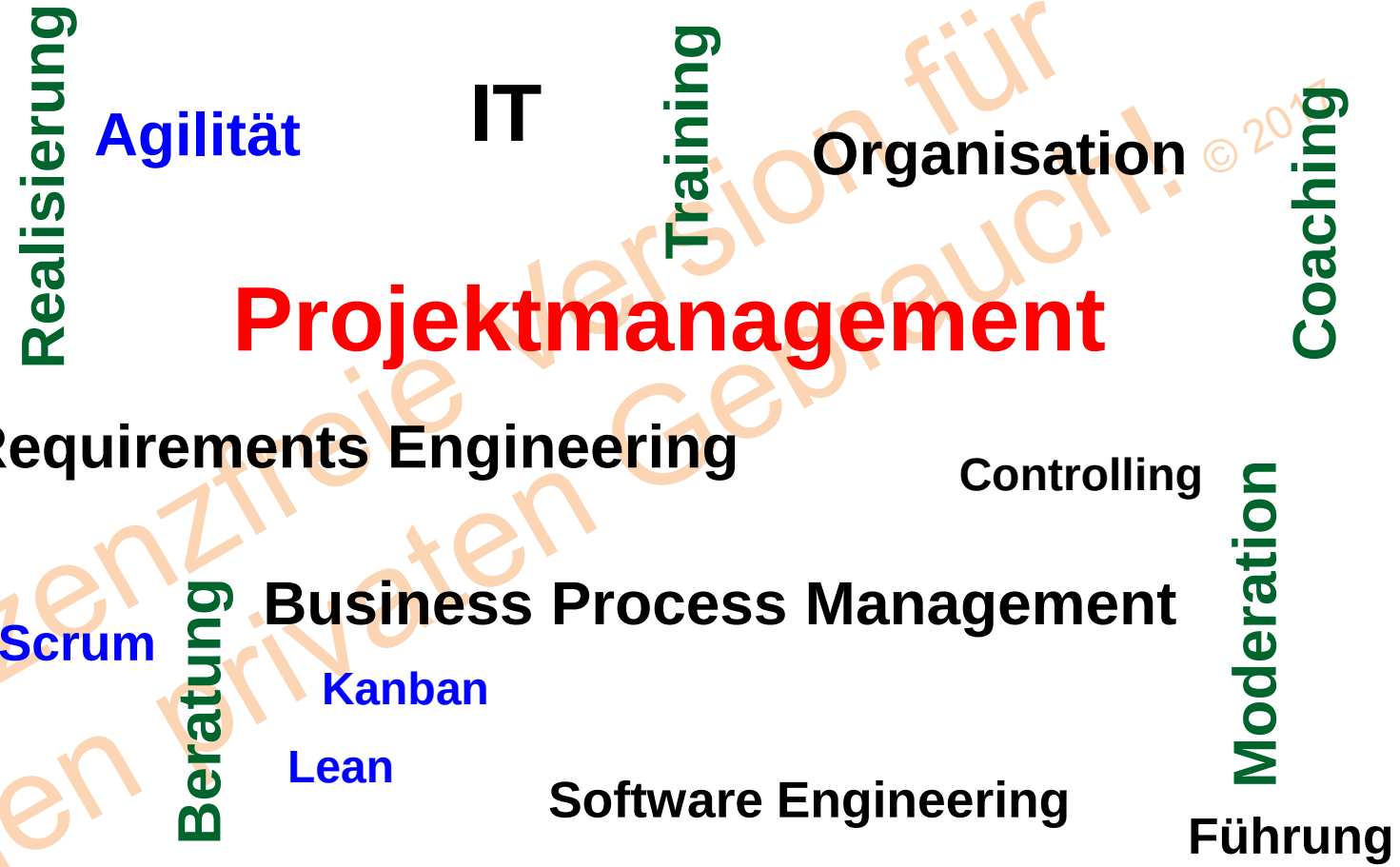
Weitere öffentliche Präsentationen des Autors

Zu meinen drei Kerndisziplinen Projektmanagement, Business Process Management und Requirements Engineering gibt es jeweils Einführungspräsentationen, die einen Einstieg in das Themengebiet ermöglichen. Diese sollten zunächst gelesen werden, bevor man weitere Präsentation anschaut.

Die Ausarbeitungen zum agilen Vorgehen („Agilität & Scrum“) sind unabhängig von den klassischen Präsentationen les- und einsetzbar.



Meine Management-Wolke – Das kann ich für Sie tun



Kontakt zum Autor

Sie benötigen noch weitere Informationen?
Kontaktieren Sie mich!

Peterjohann Consulting

Dipl.-Inform.

Horst Peterjohann

Kattenvenner Straße 24

49549 Ladbergen

Telefon: 0 54 85 / 830 17 29

Mobil: 0 162 / 977 47 65

E-Mail: kontakt@peterjohann-consulting.de

Website: http://www.peterjohann-consulting.de