Agiles Requirements Engineering professionalisieren

Ellen Leutbecher

Agile Day Karlsruhe, 10. Mai 2012

Themen dieses Vortrages

- Was ist klassisches Requirements Engineering?
  - Welcher Wertewandel ist nötig?
- Warum agiles Requirements Engineering?
  - Situation in der agilen Welt
- Was ist agiles Requirements Engineering?
  - Fünf ARE-Regeln
  - ARE mit Scrum
  - ARE Praktiken
- Zusammenfassung
- Diskussion

Inhalt

- Was ist klassisches Requirements Engineering?
  - Welcher Wertewandel ist nötig?
- Warum ARE? - Situation in der agilen Welt
- Was ist agiles Requirements Engineering?
  - 5 ARE-Regeln
  - ARE mit Scrum
  - Beschreibungstiefe
  - ARE – kombiniert Werkzeuge
  - ARE – eine andere Kommunikation
  - ARE – verwendet passende RE-Techniken
- Zusammenfassung
- Literatur

Zur Einhaltung des Urheberrechts wurden die meisten Fotos entfernt.

Ellen Leutbecher

Ellen Leutecher hat umfangreiche Erfahrung mit agilen Methoden und ist Certified Scrum Master und Certified Professional for Requirements Engineering FL.

Sie hat in einem großen Pharmakonzern 10 Jahre lang die Solution Delivery Prozesse des V-Modells mit agilen Praktiken angereichert und diese in einer weltweit verteilten Organisation eingeführt.

In ihrem letzten, mehrjährigen Projekt hat sie als Product Owner, Business Analyst und Solution Architect eine Geschäftsanwendung analysiert, entworfen sowie die Codierung und die Abnahmen geleitet.

E-Mail: leutbecher.e@arcor.de
LinkedIn: www.linkedin.com/in/eleutbecher
Xing: https://www.xing.com/profile/Ellen_Leutbecher

Was ist klassisches Requirements Engineering ?
Ein kurzer Blick zurück

Strukturierte Analyse
(1970 - 77)

Kontextdiagramm
Datenflussdiagramm
Glossary

Zielmodelle
Entity-Relationship-Diagram

Objektorientierte Analyse
(ca 1994)

Klassendiagramm
Use-Case-Diagramm
Aktivitätsdiagramm
Zustandsdiagramm

Was ist klassisches Requirements Engineering ?

Strukturie
(1970 - 77)

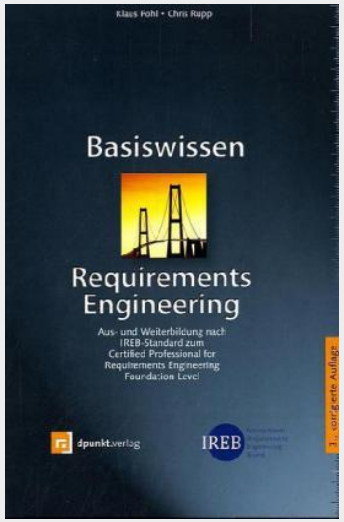
Objektorie
(ca 1994)

Klassisches Requirements Engineering

Kontextdiagramm
Datenflussdiagramm
Glossary

Zielmodelle
Entity-Relationship-Diagram

Klassendiagramm
Use-Case-Diagramm
Aktivitätsdiagramm
Zustandsdiagramm



Grundhaltung im klassischen RE

- Fehlerfreie und vollständige Anforderungen sind die Basis für eine erfolgreiche Systementwicklung

➔ Methoden und Arbeit des Requirements Engineers perfektionieren

Requirements Engineering perfektionieren

Kontext, Stakeholder exakt und vollständig identifizieren

Gleichmäßiger und sehr feiner Detailierungsgrad für alle Anforderungen

Exakte, eindeutige, korrekte, vollständige Anforderungen dokumentieren

Protokolle aller Gespräche mit Anforderungen ablegen

Anforderungen über den gesamten Lebenszyklus des Produktes verwalten

Grundhaltung im klassischen RE

- Fehlerfreie und vollständige Anforderungen sind die Basis für eine erfolgreiche Systementwicklung

➔ Methoden und Arbeit des Requirements Engineers perfektionieren

➔ Perfekte, sehr detaillierte Anforderungen werden dem Entwicklungsteam übergeben

Grundhaltung im klassischen RE – Vergleich mit Agilem Manifest

- Fehlerfreie und vollständige Anforderungen sind die Basis für eine erfolgreiche Systementwicklung

➔ Methoden und Arbeit des Requirements Engineers perfektionieren

Individuen und Interaktionen haben Vorrang vor Prozessen und Werkzeugen

➔ Perfekte, sehr detaillierte Anforderungen werden dem Entwicklungsteam übergeben

Funktionsfähige Produkte haben Vorrang vor ausgedehnter Dokumentation

Grundhaltung im klassischen RE – Vergleich mit Lean Principles

- Fehlerfreie und vollständige Anforderungen sind die Basis für eine erfolgreiche Systementwicklung

➔ Methoden und Arbeit des Requirements Engineers perfektionieren

➔ Perfekte, sehr detaillierte Anforderungen werden dem Entwicklungsteam übergeben

Optimierung einer Disziplin

Das ganze Problem sehen (Produktentwicklung)

Grundhaltung im klassischen RE – Vergleich mit Lean Principles

- Fehlerfreie und vollständige Anforderungen sind die Basis für eine erfolgreiche Systementwicklung

➔ Methoden und Arbeit des Requirements Engineers perfektionieren

➔ Perfekte, sehr detaillierte Anforderungen werden dem Entwicklungsteam übergeben

Übergaben verursachen Informationsverluste

Typischerweise nur 40% des Wissens des RE dokumentiert

Detaillierte Anforderungen sind teilweise gemachte Arbeit

Die Analyse , das Lernen, endet vor Beginn der Programmierung

Vermeiden von Abfall,
Entscheidungen so spät wie möglich treffen

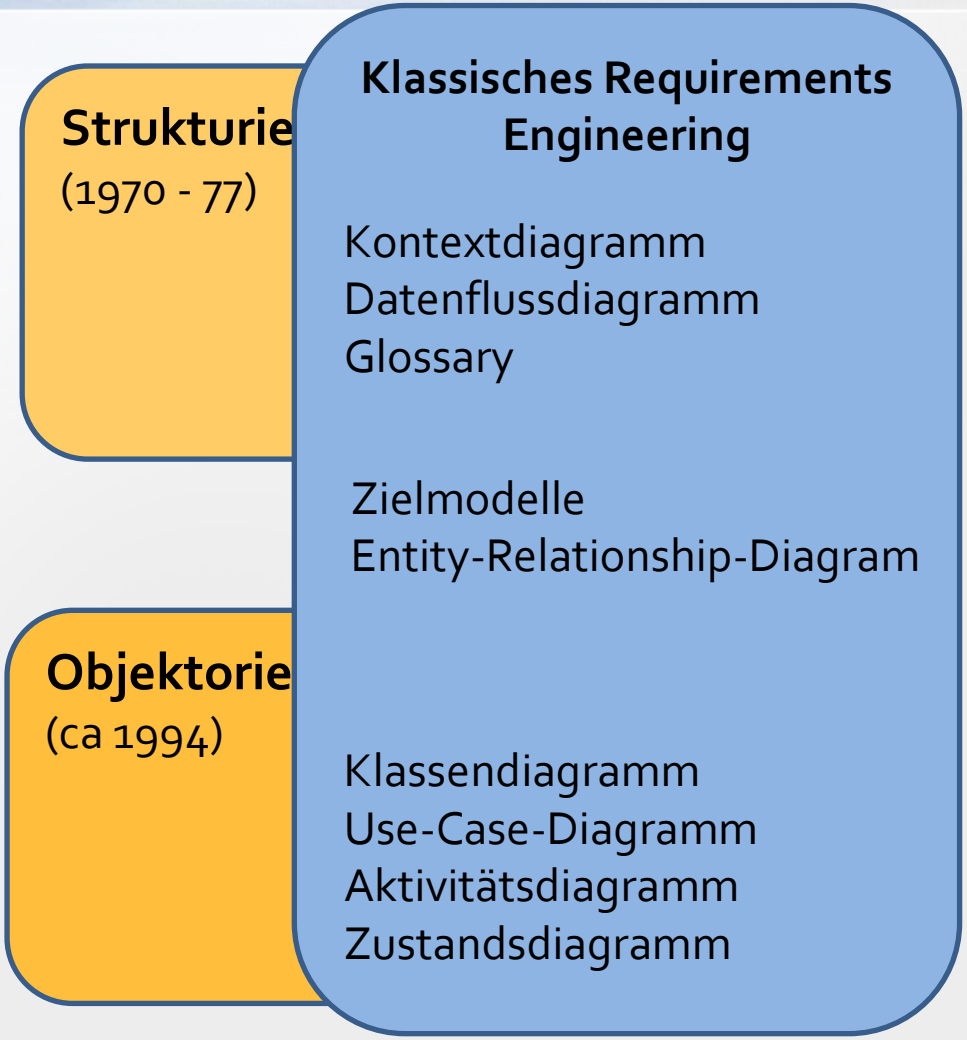
Welche Haltungen des klass. RE ablegen und welche neuen annehmen? (1/2)

- Perfektionieren einer Disziplin
  - ➔ die ganze Aufgabe sehen (Produktentwicklung)
  - ➔ in jedem Bereich gerade genug des Richtigen tun, um das bestmögliche Gesamtergebnis zu erreichen
- Kommunikation über Dokumentation
  - ➔ in fortlaufenden Gesprächen
- Arbeiten in Phasen
  - ➔ Iteratives Arbeiten und Lernen
- Verteilte Aufgaben, Übergaben
  - ➔ Teamaufgabe

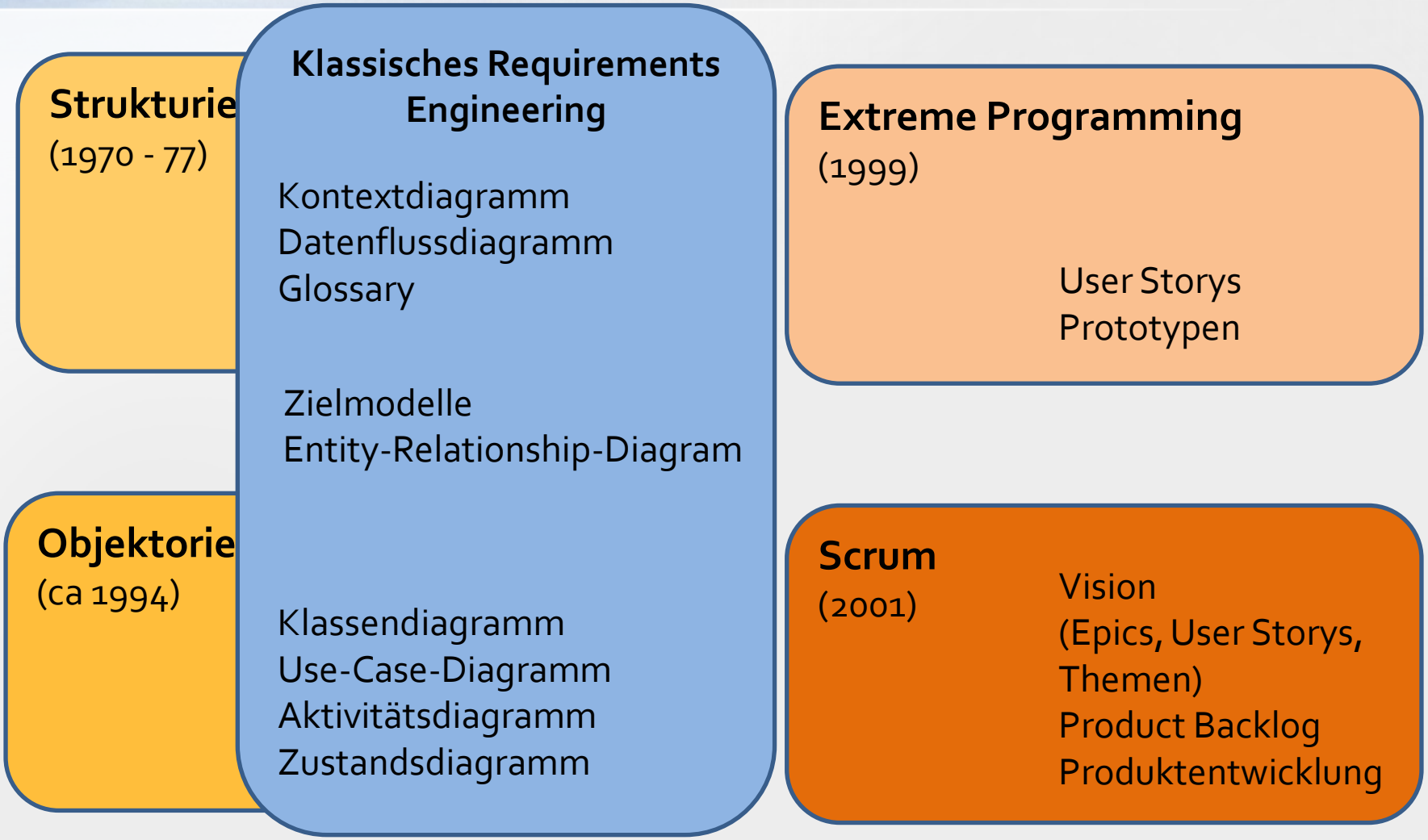
Welche Haltungen des klass RE ablegen und welche neuen annehmen? (2/2)

- Spezifikation durch eine Person im Voraus
  - ➔ gemeinsame just-in-time Spezifikation
- Beim ersten Mal alles richtig machen
  - ➔ Vorschlag erstellen, anschauen und anpassen
- Änderungen verhindern
  - ➔ Änderungen erwünscht
  - ➔ auf Änderungen der Umwelt flexibel reagieren

Was ist klassisches Requirements Engineering?



Nun kommen agile Methoden dazu



Situation in der agilen Welt

- Scrum ist in der Softwareentwicklung mittlerweile etabliert
- Mit Scrum, XP und anderen agilen Methoden wurden beachtliche Produktivitätssteigerungen erzielt

➔ Nun muss eine Verbesserung der Praktiken für

- Architektur und
- Requirements Engineering (RE)

stattfinden, damit weitere Vorteile realisiert werden können

Situation in der agilen Welt

- Gutes Requirements Engineering (RE) ist ein wesentlicher Erfolgsfaktor für agile Projekte

- Die Erfinder und Promotoren von Scrum bemühen sich jedoch, den Begriff RE zu vermeiden, da dieser häufig mit schwergewichtigem, dokumentengetriebenem Vorgehen assoziiert wird

Situation in der agilen Welt

- In Scrum findet eine andere Art RE statt
  - Produkt Vision
  - Product Backlog mit
    - Epen, Benutzergeschichten (User Stories) und Themen
  - Just-In-Time Detaillierung
  - Keine im Voraus erstellte, umfassende Spezifikationen
- Diese einfachsten RE-Mittel reichen zusammen mit regelmäßigen, kurzfristigen Rückkopplungs- und Lernmechanismen aus, um erfolgreich zu arbeiten
- **Wir können mehr tun, um schneller das Richtige zu tun und um erstklassig zu werden**

Was ist agiles Requirements Engineering ?

- Agiles Requirements Engineering ist eine Methode, die es ermöglicht, speziell in einem agilen Software-entwicklungsprozess und mit einem agilen Team
  - **das richtige Maß an Requirements Engineering** zu **praktizieren**
  - **agile Prozesse** zu **fördern**
  - **mit sich ändernden Anforde-rungen souverän umzugehen**
  - die **intensive Kommunikation** mit allen Beteiligten zu **fördern**

Ziel des agilen Requirements Engineerings

- Das agile Requirements Engineering (ARE) integriert die Disziplin des Requirements Engineering in den agilen Entwicklungsprozess

Fünf ARE-Regeln

- Priorisiere die Anforderungen nach ihrem Geschäftswert. Die hochprioren Anforderungen werden zuerst detailliert und umgesetzt
- Wähle für jede einzelne Anforderung die passende RE-Technik, um sie zu beschreiben
- Wähle für jede einzelne Anforderung die angemessene Beschreibungstiefe
- Im Umkehrschluss zu 1) und 2): Verwende kein einheitliches Beschreibungsverfahren für alle Anforderungen
- Erstelle nur Anforderungsartefakte, die unmittelbar verwendet werden

Anmerkung: Die Priorität hat keinen Einfluss auf die einzusetzende RE-Technik und Beschreibungstiefe

Effizienzsteigerungen

- Gerade genug tun
  - Für jede Anforderung die passenden Methoden anwenden, die nötige Verständnistiefe erreichen
  - Die gezielte Unvollkommenheit als notwendigen Optimierungsschritt akzeptieren

- Schnelles Feedback – schnelle Akzeptanz
  - die Zeit vom Detaillieren einer Anforderungen bis zur Akzeptanz ihrer Umsetzung möglichst kurz halten

  ➔ **Just-in-time Spezifikation**

Agiles RE mit Scrum

- Sehr enge Zusammenarbeit zwischen Kunde, RE und Entwicklungsteam
- Review **laufender** Software am Ende jeder Iteration durch Kunde und Benutzer

➔ **Fundiertes Feedback**

➔ **Besseres Verständnis der Anforderungen**

➔ **Kontinuierliche Verbesserung des Produkts**

Agiles RE mit Scrum

- Sprint Review und Sprint Retrospektive helfen die Frage „Wie viel Requirements Engineering brauche ich?“
  - erfahrungsbasiert
  - viel differenzierter und
  - für jede Anforderung

  zu beantworten

➔ **Vermeiden von Abfall**

- Denn RE an sich kostet nur und erfährt seine Rechtfertigung erst dadurch, dass es mehr Nutzen als Kosten im Entwicklungsprozess verursacht

Kriterien für Beschreibungstiefe

- Weniger Details
  - Kundenfeedback gut, Team eingearbeitet
  - Anforderung ist sehr ähnlich wie eine andere, bereits realisierte und abgenommene Anforderung
  - Anforderung mit Team im Detail diskutiert
- Mehr Details
  - Viele fachliche Varianten und Ausnahmen berücksichtigen
  - Viele verschiedene Anforderungsgeber oder -beeinflusser beteiligt
  - Hohes Risiko, dass vergessene Abhängigkeiten oder Details zu großen Mehrkosten und Verzögerungen führen
- **Wichtig: Heutige Bedürfnisse des Entwicklungsteams und anderer Stakeholder berücksichtigen**

Die richtige Granularität finden - Anregungen aus Lernpsychologie

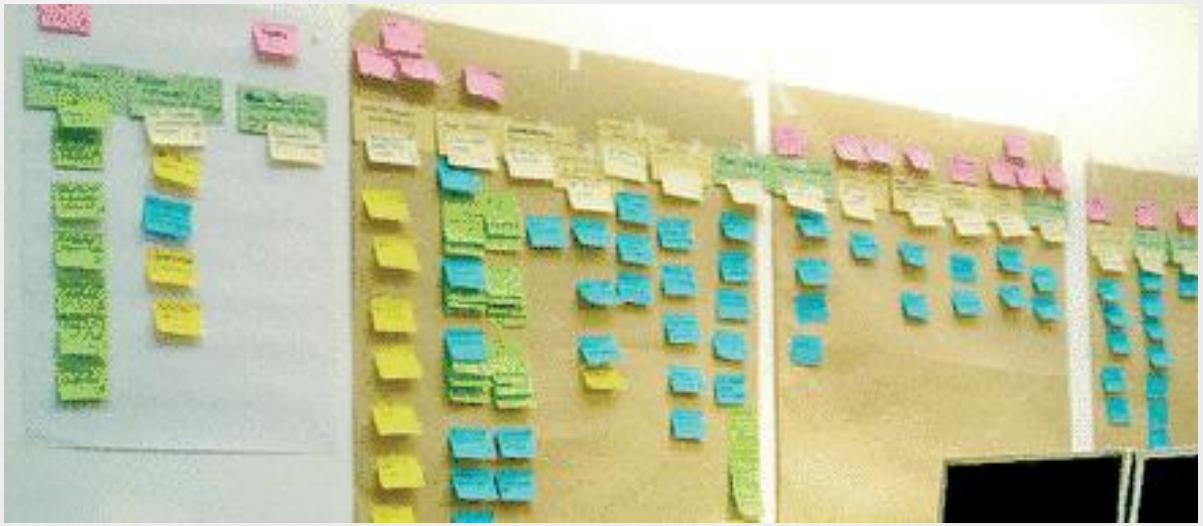
- Ich habe in der Lernpsychologie gelernt
  - Menschen können sich problemlos 3 Dinge gleichzeitig merken
  - Trainierte Menschen schaffen 7 – 10 Dinge
  - Sehr gute bis zu 12 Dinge
- Empfehlung: dies in Präsentationen, Dokumenten und Diagrammen beachten
  - Eher 3 bis 7 Punkte pro Thema
  - Maximal 12 Elemente pro Diagramm
  - Details auf der nächsten Detailebene darstellen

➔ **Schnelleres, nachhaltigeres Erfassen der Informationen**

➔ **Lerneffekt höher**

Die richtige Granularität finden - Anwendung auf ARE

- Beschreibe den Geschäftsprozess mit maximal 12 Teilprozessen (besser 7 – 10)
- Beschreibe die Anwendung, das System mit maximal 12 Epen
- Zerteile jedes Epos in maximal 12 User Stories
- Zerteile jede User Story entsprechend



It-agile: Lagerverwaltungssystem eines Versandhändlers

Beispiele aus meinem letzten Projekt

- Der zentrale Geschäftsprozess
  - 1 Trigger
  - 10 Teilprozesse
- Dauer des Prozesses
  - mehrere Monate bis Jahre
- Dauer der Teilprozesse
  - Monate, Tage, Stunden

Activty Diagram ohne Swim laines zeigt Geschäftsprozess

Die Diagramme dürfen leider nicht veröffentlicht werden

Beispiele aus meinem letzten Projekt

- Der zentrale Geschäftsprozess
  - Erläuterte 80% der Systemfunktionen
- Ein Prozess musste im Detail analysiert werden
  - Zusammenspiel versch. Abteilungen & Systeme

Activty Diagram ohne Swim laines zeigt Geschäftsprozess

Beispiele aus meinem letzten Projekt

- Der Detail-Prozess
  - 14 Teilprozesse
- Dauer des Prozesses
  - Monate
- Dauer der Teilprozesse
  - Monate, Tage, Stunden, Minuten

Activty Diagram
Mit Swim laines
zeigt
Detail-Prozess

Die Diagramme dürfen leider nicht veröffentlicht werden

Beispiele aus meinem letzten Projekt

- Wann werden welche Epen benutzt?
  - 8 Epen
  - 4 User Stories, die von 2 Epen verwendet werden
  - Restliche User Stories fehlen bewusst

Activty Diagram ohne Swim laines zeigt Geschäftsprozess, Epen und User Stories

Beispiele aus meinem letzten Projekt

- Welche Epen werden von welchen Benutzern verwendet?
  - 4 Benutzertypen
  - 8 Epen
  - 4 User Stories (Wiederverwendung)
  - Restliche User Stories fehlen bewusst

Diagramm zeigt Benutzertypen, Epen und User Stories

Die Diagramme dürfen leider nicht veröffentlicht werden

Die richtige Granularität finden - Zusammenfassung

- Empfehlung für Präsentationen, Dokumente und Diagramme
  - Eher 3 bis 7 Punkte pro Thema und Ebene
  - Maximal 12 Elemente pro Diagramm
  - Details auf der nächsten Detailebene darstellen
- Anwendung im ARE
  - Geschäftsprozesse mit maximal 12 Teilprozessen
  - Beschreibe das System mit maximal 12 Epen
  - Zerteile jedes Epos in maximal 12 User Stories
  - Zerteile jede User Story entsprechend

It-agile: Lagerverwaltungssystem eines Versandhändlers

Agiles Requirements Engineering – kombiniert Werkzeuge

- Ist vor allem ein anderer RE-Prozess,
  der besonderen Werten und Prinzipien folgt
  - ARE beschreibt **keine** neuen Werkzeuge zur Anforderungserhebung
  - ARE kombiniert die Anforderungswerkzeuge der agilen Welt mit den Anforderungswerkzeugen der klassischen Welt

1. Beispiel für Kombination

- Roman Pichler: Mit Product Backlog und User Stories zu arbeiten, bedeutet nicht, dass Sie auf andere hilfreiche Artefakte verzichten sollen
  - Übersicht der verwendeten Benutzerrollen
  - User-Story-Ablaufdiagramm
  - Diagramme für Geschäfts-prozesse oder –regeln
  - Skizzen und Prototypen der Benutzeroberfläche

2. Beispiel für Kombination

- Tim Weilkiens: Benutzergeschichten beschreiben in wenigen Sätzen eine Anforderung aus Sicht des Anwenders
  - Sie helfen Anforderungen zu benennen und
  - Nutzen die Akzeptanzkriterien für weitere Details

- Ergänzung durch klassische Werkzeuge für Anwendungsfälle (Use Cases)
  - Ablaufdiagramme oder Zustandsdiagramme
  - Normal Flow, Alternative Flows, Exceptional Flows

2. Beispiel für Kombination

- 1. Vorschlag
  - Benutzergeschichten verwenden
    - Details als Akzeptanzkriterien
    - Designs, Details, UML-Diagramme u.a. in Zusatzdokumente, die in den Akzeptanzkriterien gelistet werden

- 2. Vorschlag
  - Schema für die Beschreibung einer User Story erweitern mit Elementen aus der Use Case Beschreibung
    - Akteure, Auslöser, verwendete User Stories
    - Normal Flow, Alternative Flows, Exceptional Flows

3. Beispiel für Kombination

Empfehlung: Weniger ist mehr!

- Bernd Oestereich: Schema für Product Backlog Einträge
  - **ID**
  - **Name oder Titel**
  - **Kurze Beschreibung**
  - Priorität
  - Hinweis auf besondere Chancen und Risiken
  - Relativer Aufwand
  - Fachliches Thema
  - Subsystem, Komponente o.ä.
  - **Akzeptanzkriterien**
  - Urheber und Ansprechpartner
  - Evtl. Fragen, Lösungsideen und -empfehlungen

Agiles Requirements Engineering – eine andere Kommunikation

- Basis: das häufige, persönliche Gespräch
  - Kunde, PO, RE und Entwickler im gleichen Meeting
- inhaltlich intensiver, fokussierter
- über die gesamte Entwicklungszeit
- Der Kunde
  - kann und muss mehr Rückmeldungen geben
  - wird mehr in die Pflicht genommen
  - sieht viel konkretere Ergebnisse, so dass er einfacher korrigierend wirken kann

➔ **eine viel höhere Qualität im Anforderungsdiskurs**

Agiles Requirements Engineering – eine andere Kommunikation

- Scrum bewirkt eine Veränderung weg von der dokumentenbasierten Kommunikation hin zum Gespräch
- User Stories fordern und fördern einen Dialog über Anforderungen und Produktfunktionalität

➔ **Die effizienteste und effektivste Methode des Informationsaustauschs mit und in einem Entwicklungsteam ist das Gespräch von Angesicht zu Angesicht (Agiles Manifest)**

Agiles Requirements Engineering – verwendet passende RE-Techniken

- Für die Anforderungsermittlung
- Für die präzise Anforderungsdokumentation
- Für das Prüfen und Abstimmen von Anforderungen

Hierzu kann aus Zeitgründen nur auf ein Buch hingewiesen werden:

Chris Rupp, Requirements-Engineering und -Management, Hanser, 2009, 5. Auflage

Fazit - Agile Grundhaltung

- In jedem Bereich gerade genug des Richtigen tun, um das bestmögliche Gesamtergebnis zu erreichen
  - Kein Perfektionismus in Teilbereichen
  - Übergaben vermeiden
  - Persönliches Gespräch

➔ **Für welche Beteiligten (Stakeholder/Entwicklungsteam) brauche ich heute wie detaillierte Anforderungen?**

Regeln des Agilen Requirements Engineering

- Wähle für jede Anforderung und Dein Umfeld
  - die passende RE-Technik
  - die angemessene Beschreibungstiefe

  ➔ **Keine einheitliche Granularität in der Anforderungsbeschreibung**
- Erstelle nur Anforderungsartefakte, die unmittelbar verwendet werden
- Mische nach Bedarf RE-Techniken aus Scrum und XP mit klassischen RE-Techniken

Techniken des Agilen Requirements Engineering

- Methoden aus Scrum und XP
  - Benutzergeschichten (User Stories) mit Akzeptanzkriterien
  - UI Skizzen und Prototypen
- Klassische RE-Techniken
  - Use Cases
  - Einfache UML-Diagramme
- Kultur
  - Kommunikation: fokussiertes Gespräch mit allen Beteiligten im gleichen Meeting
  - Iteratives Lernen: discuss, do, inspect, and adapt


Danke
schön!

Literaturauswahl

- Agile Methoden
  - Roman Pichler, Agiles Produkt-management mit Scrum, Addison-Wesley, 2012
  - Tim Weilkiens, oose Hamburg, 30.09.2010 Klassisches Requirements Engineering und agile Entwicklungs-prozesse kombinieren in Elektronik Praxis, 9/2010 , http://www.elektronikpraxis.vogel.de/index.cfm?pid=5416&pk=284677&print=true&printtype=article aufgerufen 30.1.2012 17.00 h
  - Raphael auf der Maur, bbv Schweiz Agiles Requirements-Engineering: Ein Erfolgsfaktor für Produktentwicklungen, In ObjektSpektrum März/April 2012 Nr 2.

Literaturauswahl

- Klassisches Requirements Engineering
  - Klaus Pohl, Chris Rupp,
    Basiswissen RE
    dpunkt.verlag, IREB, 2011, 3. Auflage
  - Chris Rupp, Requirements-Engineering und -Management
    Hanser, 2009, 5. Auflage
- Lean Software Development
  - von Poppendieck, Mary; Poppendieck, Tom
    Leading Lean Software Development
    Addison-Wesley, 2009