Software Entwicklung 1

Asha Schwegler

18. März 2022

1 Software Engineering

- Herstellung / Entwicklung von Software
- Organisation und Modellierung (Zugehörigen Datenstrukturen, Bettrieb von Softwaresystemen)
- Strukturiertes Projektplan f. Entwicklung
- Unterteilung Entwicklungsprozess
  - Schritte (überschaubar, zeitlich und inhaltlich begrenzt)
  - Phasen
  - Meilensteine
- Disziplinen während Entwicklungsprozess sind verzahnt.

Disziplinen

- **Kerndisziplinen**
  - Anforderungsanalyse
  - Softwarearchitekur und Design
  - Implementierung / Test
  - Softwareverteilung
  - Softwareeinführung
  - Wartung / Pflege
- **Unterstützungsdisziplinen**
  - Projektmanagement
  - Konfigurationsmanagemdnt
  - Risikomanagement

2 Prozess und Prozess-Modell

- Prozess
  - Beschreibung Aktivitäten, Rollen und Artefakte(Informationen)
  - Software-Entwicklung und Wartung
- Prozessmodell
  - Präskriptives Modell (Vorgehensmodell und Organisationsstrukturen)
  - Planung und Lenkung
  - Unified Process, V-Modell, Scrum,...

2.1 Vorgehensmodelle

- Code and Fix
- Wasserfallmodell
- Iterative und inkrementelle Modelle

Code and Fix

- Definition
  - Codierung / Korrektur im Wechsel mit Ad-hoc Tests
- Vorteile
  - Schnell vorankommen
  - Schnelle Ergebnisse
  - Einfache Tätigkeiten (Codieren, Testen, Fixen)
- Nachteile
  - Schlecht planbar und keine Unterstützung im Team
  - Aufwand hoch für Korrekturen
  - Sclecht wartbare Software

Wassefallmodell

- Definition
  - Folge von Aktivitäten/Phasen, gekoppelt durch Teilergebnisse (Dokumente). Reihenfolge ist fest definiert.
- Vorteile
  - hohe Planbarkeit
  - Klare Aufteilung der SWE (Analyse, Design, Test,...)
- Nachteile
  - Schlechtes Risikomanagement (Lösungskonzept nur auf Papier validiert)
  - Anforderungen zu Beginn nie alle bekannt

Iterativ-inkrementelle Modelle

- Definition
  - Geplante und kontrollierte Iterationen inkrementell entwickelt
- Vorteile
  - Flexibles Modell bei unklaren Anforderungen
  - Gutes Risikomanagement (Mitarbeiter und Technologie)
  - Frühe Einsetzbarkeit der Software und Feedback
- Nachteile
  - Upfront Planbarkeit hat Grenzen (Funktionalität, Zeit und Kosten)
  - Braucht Involvierung und Steuerung durch den Kunden über ganze Projektdauer

2.2 Agile SWE

- Basiert auf iterativ-inkrementellen Prozessmodell
- Fokussiert auf gut dokumentierten und getesteten Code statt auf ausführlicher Dokumentation
- Sammlung von Ideen SWE Prozess flexibler und schlanker zu machern
- Adressiert bekannten Probleme bei klassischen Software-Prozessmodellen

Strategie

- Definierte Prozesskontrolle
  - Planung am Anfang, Prozess gesteuert und überwacht
  - Geeignet für gut planbare Problemstellungen
  - Strategie: Steuerung
- Empirische Prozesskontrolle (Agil)
  - Nur Grobplanung am Anfang
  - Prozess fortlaufend überwacht
  - Rollende Planung
  - Geeignet für komplexe Problemstellungen
  - Strategie: Regelung, Deming-Cycle (Plan-Do-Check-Act)

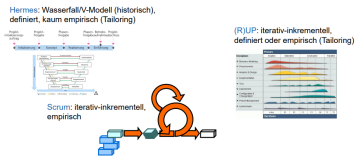

Abbildung 1: CharaktersierungProzessmodellen

3 Modellierung

Modell: Abbild oder Vorbild für ein zu schaffendes Gebilde.
Original: Abgebildete oder zu schaffende Gebilde

- **Modellierung**
  - Software selbst ein Modell
  - Anforderungen = Modelle der Lösung
  - Tesfälle = Modelle Korrektes Funktionieren des Codes

3.1 UML

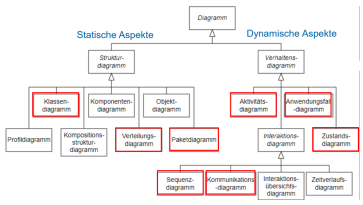

Abbildung 2: Guetereinteilung.

3.1.1 Gebrauch der UML

- UML as a sketch
  - Informell, unvollständig
  - Bevorzugt von agile Community
- UML as blueprint
  - Detallierte Analyse und Design-Diagramme für Code
  - Forward - und Reverse-Engineering
- UML as a Programming Language
  - Komplette, ausführbare Spezifikation eines Software-Systems in UML
  - MDA-Tools zur Modellierung und Generierung

4 Wesentliche Artefakte

- Anfoderungsanalyse
  - Funktionale Anforderungen mit Use Cases
  - Qualitätsanforderungen und Randbedingungen
- Design
  - Softwarearchitektur
  - Use Case Realisierung (Statische und dynamische Modelle)
- Implementation
  - Quellcode (inkl.Javadoc)
- Testing
  - Unit-Tests
  - Integrations- und Systemtests

4.1 Überblick Anforderungsanalyse

- **User Research**
  - Personas
  - Szenarien
  - Contextual Inquiry
- Sketching und Prototyping
- **Use Cases**
  - Ableiten und Modellieren
  - Detaillierung (UML-Use-Case-Diagramm, Use-Case-Spezifikation, UI-Sketching)
- **Qualitätsanforderungen, Randbedingungen** erheben
- **Domänenmodell**
  - Konzeptuelles UML-Klassendiagramm
- **objektorientierten Analyse(OOA)**
  - Objekte/Konzepte in dem Problembereich zu finden und zu beschreiben

4.2 Überblick Design

- **Softwarearfchitektur**
  - UML-Paketdiagramm
  - UML-Deploymentdiagramm
- **Use-Case-Realisierung und Klassendesign**
  - UML-Klassendiagramm
  - UML-Sequenzdiagramm
  - UML-Kommunikationsdiagramm
  - UML-Zustandsdiagramm
  - UML-Aktivitätsdiagramm
- Entwurf **Design Patterns**
- **Objektorientierten Design (OOD)**
  - Geeignete Softwareobjekte und ihr Zusammenwirken definieren

4.3 Überblick Implementation

- **Code**
  - Umsezung Design in entspr. OOP-Sprache
- **Refactoring**
  - Code smells aufdecken und verbessern
- **laufende Dokumentierung**
  - Quellcode

4.4 Überblick Testing

- **Unit-Tests**
  - Laufendes Design und Implementierung
- **Teststufen Integration und System**
  - Planung, Design und Durchführung
- **Dokumentation**
  - Testkonzept und Test

5 Usability und User Experience (UX)

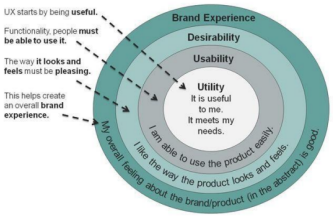

Abbildung 3: Usability.

5.1 Usability

Definition: Effektivität, Effizienz, Zufriedenheit -¿ Ziele erreichen im spezifischen Kontext

- **4 wichtige Aspekte**
  - Benutzer
  - Seine Ziele/Aufgaben
  - Sein Kontext
  - Softwaresystem (inkl.UI)

5.2 Usability Engineering

Ziel: Software entwickeln, die drei Anfroderungen erfüllt

- **Drei Anforderungen:**
  - Effektivität
    - Aufgaben vollständig erfüllen
    - Genauigkeit
  - Effizienz
    - Mit minimalem Aufwand (Mental, Physisch, Zeit)
  - Zufriedenheit
    - **Minimum:** nicht verärgert
    - **Normal:** Zufrieden
    - **Optimal:** Erfreut

5.3 Usability Anforderungen

- **7 Anforderungen:**
  - Aufgabengemessenheit
  - Lernförderlichkeit
  - Individualisierbarkeit
  - Erwartungskonformität
  - Selbstbeschreibungsfähigkeit
  - Steuerbarkeit
  - Fehlertoleranz

5.3.1 Aufgabenangemessenheit

- Minimale Anz. Schritte f. Aufgabe
- Nur wichtige Informationen
- Kontextabhängige Hilfe
- Minimale Anz. Benutzereingaben
  - Jede Eingabe nur 1x
  - Standardwerte
  - Liste vordefinierter Werte (z.B Länder)
  - Ableitbare Eingaben vorschlagen

5.3.2 Selbstbeschreibungsfähigkeit

- Benutzer ausreichend informieren
  - Wo er ist
  - Was er tun soll / kann
  - Wie er es tun soll (Formate, Werte)
- Begriffe des Benutzers verwenden (Labels, Fehlermeldungen)
- Affordanzen

5.3.3 Kontrolle

- Mit Interaktion Benutzer steuern
  - Initiative, Tempo
  - Dialogfluss
  - Darstellungsformate
  - Inputmodalität (Maus, Tastatur, Touch, Sprache)
- Benutzeraktionen rückgängig machen können
- Benutzeraktionen jeder Zeit abbrechen können

5.3.4 Erwartungskonformität

- Bezüglich
  - Design
  - Interaktion
  - Struktur
  - Komplexität
  - Funktionalität
- Konsistenz
  - Terminologie
  - Verhalten (Reihenfolge Aktionen, Änderungen)
  - Informationsdarstellung (Platzierung, Wortwahl)

5.3.5 Fehlertoleranz

- Benutzerfehler vermeiden
  - Klar kommunizieren (Erwarteter Input, Funktionen aktiv resp. sinnvoll)
- Benutzereingaben vor Aktion überprüfen
- Nicht unbedingt beim Tippen
- Benutzer helfen
  - Fehler zu erkennen
  - Ursache zu verstehen
  - Aus Fehlerzustand zu kommen
- Einfache Korrektur
- Kein Datenverlust

5.3.6 Individualisierbarkeit

- System anpassbar sein:
  - Know-How
  - Sprache
  - Kultur
  - Benutzer mit Einschränkungen

5.3.7 Lernförderlichkeit

- Informationen über unterliegende Konzepte und Regeln anbieten
  - Um mentales Modell anzugleichen
  - Nur auf Verlangen des Users
  - eifache Tasks ohne Vorkentnisse
  - komplexere Konzepte bei der Verwendung zu erlernen

6 User-Centered Design (UCD)

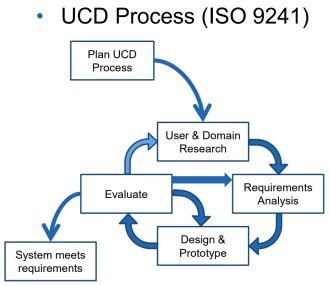

Abbildung 4: UCDProcess

6.1 User & Domain Research

- **Ziele bez. Benutzer:**
  - Wer ist Benutzer
  - Was ist die Arbeit (Aufgaben, Ziele)
  - Wie sieht Arbeitsumgebung aus
  - Was wird gebraucht um Ziele zu erreichen
- Welche Sprache, Begriffe
- Normen (organisatorisch, kulturell, sozial)
- Pain Points (Brüche, Workarounds)
- Für mobile Apps:
  - Nutzungskontext
    - Wo wird App benutzt (Umgebung)
    - Wann wird App benutzt (Tageszeit, involvierte Personen, Randbedingungen)
    - Warum wird App benutzt (Nutzen, Motivation, Trigger)
- **Ziele bez. Domäne:**
  - Buisiness der Firma verstehen
  - Domäne verstehen (Sprache, Wichtigste Konzepte, Prozesse)

6.1.1 GUI Design Process

Methoden User & Domain Research

- Contextual Inquiry
- Interviews
- Beobachtung
- Fokusgruppen
- Umfragen
- Nutzungsauswertung
- Desktop Research (Dokumentenstudium, Mitbewerber)

6.1.2 Wichtige Artekfakte

- Personas
- Usage-Szenarien
  - Kurze Geschichte
    - Usage Szenarien
      - aktuelle Situation
      - in User and domain research verwendet
    - Kontextszenarien
      - Zukünftige gewünschte Situation
      - in Anforderungsanalyse verwendet
- Mentales Modell
- Domänenmodell
- Stakeholder Map

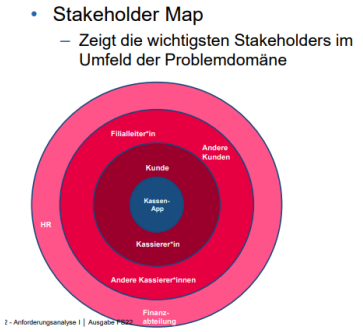

Abbildung 5: Stakeholdermap

- Service Blueprint/Geschäftsprozessmodell

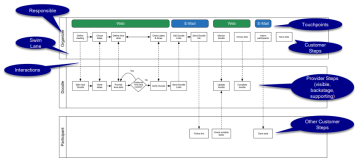

Abbildung 6: Blueprint

6.2 Anforderungsanalyse

Ziel:

- Ausgehend von den Resultaten des UCD -¿ User-Anforderungen ableiten:
  - Funktionale Abläufe, Interaktionen
    - **Kontextszenarien**
    - **Storyboards**
    - **UI-Skizzen**
    - **Use cases**
  - Konzepte, Beziehungen, Quantitäten
    - **Kontextszenarien**
    - **FURPS-Modell (Functionality, Usability, Reliability, Performance, Supportablility**

6.2.1 Use Cases

- Akteur
  - Primärakteur
  - Unterstützender Akteur
  - Offstage-Akteur
- Keine Kann-Formulierungen
- 3 Ausprägungen:
  - Kurz
    - Titel + 1 Absatz (Standardablauf)
  - Informell
    - Titel + Informelle Beschreibung (können mehrere Absätze sein, beschreibt auch Varianten)
  - Vollständig
    - Titel + alle Schritte und alle Varianten im Detail
    - UC-Name
    - Umfang
    - Ebene
    - Primärakteur
    - Stakeholders und Interessen
    - Vorbedingungen
    - Erfolgsgarantie/Nachbedingungen
    - Standardablauf
    - Erweiterungen
    - Spezielle Anforderungen
    - Liste der Technik und Datavariationen
    - Häfigkeit des Auftretens
    - Verschiedenes
  - Notation = Nomen + Verb

Use-Case-Diagramm

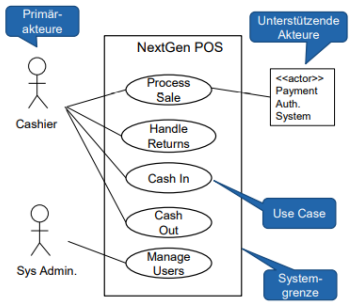

Abbildung 7: UseCaseDiagramm

Zusätzliche Beziehungen im Use Case Diagramm

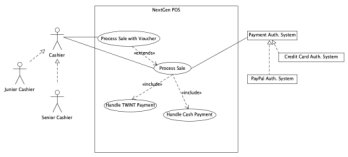

Abbildung 8: Zusätzliche Beziehungen UseCaseDiagramm

6.2.2 UML Sequenzdiagramm (SSD)

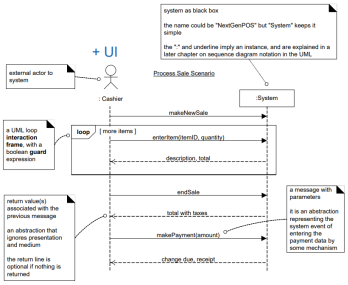

Abbildung 9: Zusätzliche Beziehungen Systemsequenzdiagramm

SSD zwischen zwei Systemen

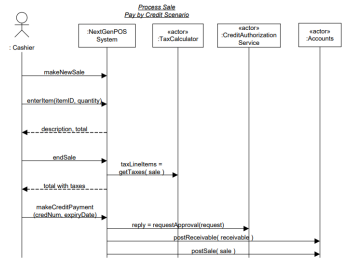

Abbildung 10: Zusätzliche Beziehungen Systemsequenzdiagramm zwischen zwei Systeme

System Operation

- Formal wie ein Methodenaufruf

- Treffender Name
- Evt. mit Parametern
- Durchzogener Pfeil für Methodenaufruf
- Rückgabewert (Kann fehlen falls unwichtig, Gestrichelte Linie weil Update des UI)
- Definieren API des Systems

6.2.3 Operation Contract

Definition: Spezifiziert (System)Operation

- Name plus Parameterliste
- Vorbedingung (Was muss zwingend erfüllt sein damit Systemoperation aufgerufen werden kann)
- Nachbedingung
  - Was hat sich alles geändert nach Ausführung (Erstellte / gelöschte Instanzen, Assoziationen, geänderte Attribute)
  - basierend auf Domänenmodell

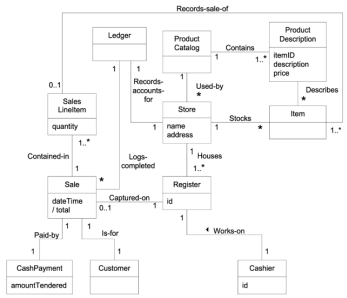

Abbildung 11: OperationContract

6.2.4 Zusätzliche Anforderungen

- Funktionale
- Nicht-Funktionale

Formulierung

- Anforderungstatements
  - Als Anforderung formuliert
  - messbar/verifizierbar
- So wenig wie nötig
  - Nur diejenige, die begründet gefordert werden
  - Keine ersten Lösungsideen als Forderungen

Checkliste FURPS+

- **F**unctionality
  - Features, Fähigkeiten, Sicherheit
- **U**sability
  - Usability Anforderungen (Kap.5.3)
  - Accessibility
- **R**eliability
  - Fehlerrate, Wiederanlauffähigkeit, Vorhersagbarkeit, Datensicherung
- **P**erformance
  - Reaktionszeiten, Durchsatz, Genauigkeit, Verfügbarkeit, Ressourceneinsatz
- **S**upportability
  - Anpassungsfähigkeit, Wartbarkeit, Internationalisierung, Konfigurierbarkeit
- +
  - Implementation (HW,BS,Sprachen, Tests...)
  - Interface
  - Operations
  - Packaging
  - Legal

Glossar

- Einfaches Glossar
  - Begriffe im Projekt und SW-Produkt
  - beliebige Elemente
- Data Dictionary
  - Zusätzliche Datenformate, Wertebereiche, Validierungsregeln