Software Engineering in der industriellen Praxis
TUM Wintersemester 2013/14
Dozenten: Stephan Frohnhoff et. al

Janosch Maier

16. Januar 2014

Inhaltsverzeichnis

1 Architekturprinzipien betrieblicher Informationssysteme

1.1 Motivation Software-Architektur

- Realisierung bestimmter Ziele
- Hilft bei Arbeitsteilung
  - Architekten: (Nichtfunktionale) Ziele, Grundlegende Prinzipien
  - Spezifikateure: Detailpläne (der funktionalen Anforderungen)
  - Entwickler
- Verschiedene Sichten
- Softwarearchitektur beschreibt innere Struktur eines Systems und die sichtbaren Eigenschaften

1.1.1 Eigenschaften betrieblicher Informationssysteme

- Groß & Komplex
- Lange Lebensdauer
- Ständiger Wandel
- Betreibbarkeit beim Kunden

$\Rightarrow$ Teilstrukturen, Wartbarkeit, Konsistente Lösungen, ...

1.2 Komponenten & Schittstellen

1.2.1 Komponenten

- Exportiert und Importiert Schittstellen
- Versteckt Implementierung
- Wiederverwendbare Einheit
- Können andere Komponenten enthalten

1.2.2 Schnittstellen

- Benutzerschnittstellen (UI)
- Programmierschnittstellen (Verbinden Komponenten)
- Operation:
  - Namen, Menge von Operationen, Protokoll beim Aufruf
  - Operationen beschreiben Verhalten von Komponenten (Signatur, Syntax, Semantik, Nichtfunktionale Eigenschaften)

1.3 Architekturprinzipien

- Abstraktion (Information auf das Wesentliche reduzieren)
- Modularisierung (Gliederung nach Zusammengehörigkeit)
- Kapselung (Abhängigkeiten reduzieren)
- Hierarchische Dekomposition (Gliederung in Ebenen)
- Separation-of-Concerns (Gliederung nach Eigenschaften)
- Einheitlichkeit (Strukturen, Muster, ... beibehalten)

⇒ Erfolgreiche Prinzipien zur Beherrschung von Komplexität (Verdichten von Information)

1.3.1 Prinzipien für Komponenten

- Hoher Zusammenhalt (Cohesion) innerhalb einer Komponente: Klar definierte Zuständigkeitsbereiche
- Geringe Kopplung (Coupling) zwischen Komponenten: Wenig Abhängigkeiten
- Kombinieren von Architekturprinzipien ergibt Architektur

1.4 Finden von guten Architekturen

- Guiedlines (Etablierte Methoden, nicht hart festgelegt)
- Architekturstile (Prinzipielle Lösungsstrukturen)
- Standard-Architekturen (Hohe Abstraktionsebene)

1.4.1 Beispiele für Architekturstile

- Schichtenarchitektur
  - Schichten übereinander angeordnet (nach Abstraktionsgrad)
  - Schicht bietet Services nach oben an
  - Schicht nutzt nur Services der direkt darunter liegenden Schicht
- Model - View - Controler
- Pipes & Filters
- Service Orientierte Architektur
- ...

1.5 Quasar-Softwarekategorien

- "Software, die sich unterschiedlich schnell ändert, wird in unterschiedliche Module aufgeteilt" [Parnas, 1972]
- Software-Kategorien sind Vorstufen zu Komponenten
- Wegweiser für gute Softwarearchitektur

1.5.1 Beispiel Gehaltssoftware

- Fachliche Änderungen ⇒ Komponenten müssen grundlegend verändert werden; Keine Wiederverwendbarkeit
- Technische Änderungen ⇒ Einheitliche Lösung; Wiederverwendbarkeit gegeben

⇒ Technik & Anwendung möglichst weit trennen.

1.5.2 Verschiedene Softwarekategorien

Wiederverwendbarkeit nimmt von 0 nach AT ab. A nutzt T, T nutzt 0-Software

- 0-Software: Unabhängig von Anwendung & Technik (Klassenbibliotheken)
- A-Software: Fachliche Anwendung, unabhängig von Technik (Buchungssystem)
- T-Software: Technische Komponente, unabhängig von Anwendung (Datenbank)
- R-Software: Repräsentation / Transformation (Anwendungsdaten in XML umsetzen)
- AT-Software: Mischung von Anwendung- / Technikteil

Regeln:

- Jede Schnittstelle, Klasse, Komponente gehört genau zu einer Software Kategorie
- Mischen von A und T führt zu schlecht wiederverwendbarer Software

1.5.3 Komponentenschnitt definiert Zuständigkeitsbereiche

- Trennung von Fach (A) und Technik (T). A ist T-unabhängig
- Eindeutige Zuständigkeiten: Vermeidung von Redundanz. Komponenten genau in einer Softwarekategorie
- Parallele Entwicklung (möglich machen)
- Abschottung Nachbarsysteme

1.6 Quasar-Architektursichten

- Technische Infrastruktur (TI): Technische Spezifikationen
- Technische Architektur (T): Virtuelle Umgebung für A-Architektur. Verbindet TI und A
- Anwendungsarchitektur (A): Strukturiert Software aus Sicht der Anwendung

1.6.1 Entwicklungsprozess mit Quasar

- Grobe A-Architektur
- Standard T-Architektur
- Auswahl der TI-Architektur
- Spezifikation der A-Architektur
- Schichten / Komponenten der T-Architektur
- Verfeinerung der A-Architektur
- Programmierung von T und A

1.7 Von Quasar zu Quasar Enterprise

- Geschäfft verstehen
- Ist-Modell bewerten
- Erheben und Bewerten
- Soll-Architektur erstellen

2 Testing in industriellen Software-Projekten

2.1 Qualitätssicherung

- Qualität ist Erfüllung von Anforderungen
- Produktqualität: Vorgabe & Überprüfung von Anforderungen (z.B. ISO 9126: Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Änderbarkeit, Übertragbarkeit; Müssen messbar sein)
- Prozessqualität (Nur mit guten Prozessen können gute Produkte entstehen)
- Maßnahmen zur Qualitätssicherung: Konstruktive Maßnahmen: Produktorientiert / Prozessorientiret (Verbesserung, Fehlervermeidung) / Analytische Maßnahmen: Messen eines Qualitätsniveaus, Analyse von Grund & Auswirkungen (Überprüfung)

2.2 Aufgaben beim Testen

- Analyse / Implementierung / Test → Komponententest, Integrationstest, Systemtest
- Testbasis: Grundlage der Qualitätsanforderungen
- Testobjekt: Gegenstand der Überprüfung
- Testfälle: Wie wird ein Testobjekt auf Basis der Testbasis getestet?
- Testspezifikation: Alle Festlegungen, um Testfall ausführen zu können (Testbasis, Testobjekt, Vorbedingungen, Eingabedaten, Soll-Ergebnisse, ...)
- Modellbasierte Testfallentwicklung: Prosa → Formale Modelle → Testfall
- Ad-Hoc-Test: Nur sinnvoll als zusätzliche Testart abgegrenzt zum systematischen Test; Meist Fehler in der Spezifikation
- Testfälle und Testdaten müssen nachvollziehbar sein. Regression (Zuordnung von Testfällen und Testdaten), Traceability (Zuordnung von Testfällen zu Anforderungen; Versionierung von Tests

2.3 Testarten

- Funktionale Test: Korrektheit und Vollständigkeit funktionalen Anforderungen
- Nicht-Funktionale Tests: Nicht-funktionale Anforedrungen, z.B. Sicherheit, Zuverlässigkeit
- Fehlertests: Verarbeitung von Fehlersituationen
- Datenkonsistenztests: Korrektheit von Datenbeständen
- Wiederinbetriebnahmetest: Nach Shutdown
- Interoperabilitätstests: Zusammenarbeit unabhängiger Komponenten
- Installationstests: Softwareinstallation in verschieden Umgebungen
- Oberflächentests: Gebrauchstauglichkeit der UIs
- Stresstests: Testen von Ausnahmesituationen; u.A. Crashtests
- Lasttest: Systemverhalten
- Performancetests: Systemverhalten bei bestimmten Anforderungen
- Sicherheitstest: Gegen Sicherheitslücken

2.4 Teststufen (V-Modell)

- Komponententest: Korrekte Funktionalität der Systemkomponenten. Isolierte Prüfung. Meist durch Entwickler durchgeführt.
- Integrationstest: Zusammenwirken der Systemkomponenten
- Systemtest: Test in realitätsnaher Umgebung (Funktionstest, Massentest, Usabilitytest)
- Abnahmetest: Test des Auftraggebers in Einsatzumgebeng als Grundlage für die Abnahme der Software unter Mitwirkung des Auftragnehmers

vgl. Folie 32

2.5 Erstellung von Testfällen

- Testfälle müssen zu Beginn der Tests verfügbar sein! ⇒ Entwicklung der Testfälle parallel zur Implementierung der Software

2.5.1 Risikobasiertes Testen

- Vollständige Abdeckung aller Ein-/Ausgabe-Möglichkeiten nicht möglich
- Beachtung der Wirtschaftlichkeit / des Schadens
- Welche Funktionalität wird wie häufig genutzt?
- Wichtigkeit / Qulitätsanforderungen / Risiko für Leib und Leben

2.5.2 Betrachtungsweise beim Test

- Whitebox-Test: Abdeckung aller Pfade
- Blackbox-Test

2.5.3 Äquivalenzklassen & Grenzwerte

- Gültige & Ungültige Werte werden in Äquivalenzklassen zusammengefasst
- Kombinationen aus den verschiedenen Klassen werden getestet
- Tests für Grenzwerte von gültigen / ungültigen Eingaben

2.6 Durchführung von Testfällen

- Erzeugen der Vorbedingungen
- Manuelles Ausführen der Testschritte
- Auswertung der Ergebnisse
- Abweichungen müssen Protokolliert werden. Fehler (im Testfall, Spezifikation, IT-System)

- Testergebnisse sind wichtiges Entwicklungsergebnis und müssen dokumentiert werden.
- Testen meist über Applikations- & Persistenzschicht. Sonst Nachbarsysteme nicht berücksichtigt
- Regressionstest: Erneuter test, eines bereits getesteten Programms, um nachzuweisen, dass Änderung keinen Fehler eingebaut hat.

2.7 Automatisierung von Testaufgaben

- Manueller Test aufwändig & fehleranfällig ⇒ Automatisiertes Testen
- Erstellung von Testfällen sehr schwierig automatisierbar
- Durchführung von Testfällen gut automatisierbar
- z.B. JUnit, Selenium, Silk Test

3 Change Management

3.1 Orientierung

- Commitment der Geschäftsleitung
- Projektziele / Struktur
- Projektvorlage und Konkretisierung
- Gemeinsame Vision

3.2 Warum Ändern?

- Veränderungsimpulse (Ziele?) → Anerkennung/Wertschätzung des aktuellen Status
- Enterprise 2.0 / Mobiles Arbeiten / Technologischer Forschritt / Agile Entwicklung / Arbeitgeber Attraktivität / Führung / Demografie / Workforcemanagement / Bürostrukturen
- Strategische Ausrichtung → Gemeinsame Vision

3.3 Aktivierung der Beteiligten

- Warum haben wir verändert?
- Was habe ich davon?
- Was bleibt gleich?
- Was kann schief gehen?

3.3.1 Überwindung emotionaler Barrieren

- Verunsicherung
- Angst
- Ärger über fehlende Wertschätzung des Vorhandenen
- Fehlendes Vertrauen
- Trauer (auf Gewohnheit zu verzichten)

3.4 Typische Phasen in Veränderungsprozessen

- Leugnen (Schockstarre, wenig Aktivität)
- Widerstand (Negative Energie, Viel Aktivität)
- Frustration (Tal der Tränen, wenig Aktivität)
- Anpassung (Positive Energie, Viel Aktivität)

3.5 Phasen des Veränderungsprozesses

- Unfreezing (Aufrütteln)
- Moving
  - Visionsentwicklung, Implementierung
  - Projekt am Anfang am stärksten zu beeinflussen
- Refreezing (In stabilem Zustand ankommen)

3.6 Wie verändern?

- Messbares Ziel der Änderung
- Disruptive Veränderung $\leftrightarrow$ Kontinuierliche Veränderung
- Überprüfen der Maßnahme

3.7 Ansatzpunkte für Veränderungen

- Equipment
- Process
  - Dokumentation vorhandener Prozesse
  - Analyse von Schwachstellen
  - Analyse von Prozesspotentialen
  - Definiton Sollprozesse/Verbesserungsmaßnahmen
- People
  - Wissen – Schulung
  - Verhalten / Einstellung – Coaching & Communities
  - Identität
- Materials
- Environment
- Management

4 Wirtschaftlichkeit von IT-Projekten

4.1 Grundlagen

4.1.1 Zahlen & Fakten

- IT Budget ~10% des Umsatztes
- Personalaufwand vs. Sachaufwand ~60% - 40%
- ~80% Kosten Wartung und Betrieb
- Ab 5 Jahren Produktion ~80% der Lebenszykluskosten für Produktion & Weiterentwicklung

4.1.2 IT-Wirtschaftlichkeit

- IT-Projekte
  - Große Investition
  - Häufig Intransparenter Nutzen
  - Begrenzte IT-Budgets ⇒ Priorisierung
- IT-Organisation
  - Wirtschaftlichkeit in Unternehmen. IT ist "Kostentreiber"
  - Häufig nicht Kerngeschäft.
- Ziele
  - Messbare Kriterien für Entscheidungen
  - Laufende Kontrolle

  ⇒ Wirtschaftlichkeit = Nutzen / Kosten

4.2 Wirtschaftlichkeit der IT-Organisation

4.2.1 Optimierung der IT-Wirtschaftlichkeit

- IT-Aufwand – Kosten, Leistung?
- IT-Nutzen – Nutzen, Risiken, Qualität
- Optimierung Nutzen / Aufwand
  - Effizienz – Gleicher Nutzen mit geringerem Aufwand (Kostenreduktion)
  - Effektivität – Höherer Nutzen mit gleichem Aufwand (Prozessverbesserung)

4.2.2 IT-Wirtschaftlichkeit auf verschiedenen Ebenen

- Mittelverwendende / -bereitstellende Einheit
  - Geschäftsfeld: Nutzen durch Geschäft getrieben
  - IT-Funktion: Nutzen durch IT getrieben
- Aktivitäten / Maßnahmen
  - Programme: Langfristiger Effekt (Strategien)
  - Projekte: Wirtschaftlichkeit einzelner Projekte
  - Portfolios: Gesamte Wirtschaftlichkeit aller Portfolios
  - Infrastruktur: Wirtschaftlichkeit IT-Basis

4.3 Wirtschaftlichkeit von IT-Projekten

4.3.1 Phasen eines IT-Projektes – Kostentreiber

- Planung
  - Machbarkeitsstudie / Business Case
  - Grobkonzept / Anforderungsanalyse
  - Make or Buy Entscheidung
  - Fachkonzept, Design
- Entwicklung
  - Implementierung / Customizing
  - Test / Integration
  - Hardware / Software
  - SLA-Erstellung (Service-Level-Agreement)
  - Inbetriebnahme / Roll-Out
  - *Organisationsänderung*
- Betrieb
  - Anpassung
  - Wartung & Pflege
  - Betriebsmittel
  - *Weiterentwicklung*
  - *Releasewechsel*
  - **Nur hier Nutzen! Sonst nur anfallende Kosten.**
- Ablösung
  - *Entsorgung*
  - *Datensicherung*
  - *Datenmigration*

4.3.2 Kostenanalysen

- Total Cost of Ownership (TCO) / Kostenanalysen schwierig. Keine einheitliche Definition, Kosten für Fachabteilungen, Differenzierungsprobleme, ...
- Üneternehmensübergreifende Benchmarks kaum möglich
- Business Case ist Zusammenfassung aller relevanten Informationen für Entscheidungsträger um Wirtschaftlichkeit & Strategie aufzuzeigen

4.4 Business-Case und IT-Investitionsvorhaben

4.4.1 Wirtschaftlichkeit im Projektverlauf

Business Case (BC) soll ganzes Projekt begleiten. Intiale Betrachtung der Wirtschaftlichkeit vor allem in der Anfangsphase. Controlling misst Zielerreichung und zeigt Korrekturmaßnahmen

- Idee
- Analyse / Vorstudie – Initialer Business Case
  - Zielkonkretisierung, Lösungsalternativen, Genehmigungsvorlage
  - Kosten-Nutzen-Kategorien, Wirtschaftlichkeitsrechnung, Lösungsszenarien vs. Referenzszenario
- Initialisierung – Anpassung BC / Ab hier: Controlling
  - Entscheidungskriterien, Lasten- & Pflichtenheft, Anbieterauswahl, Projektvertrag
  - Wirtschaftliche Bewertung, Konkretisierung
- Umsetzung – Anpassung BC
  - Umsetzung gemäß Vorgehensmodell, Projektmanagement (Termin vs. Inhalt/Qualität vs. Budget), Projekt-Steuerung: Vergleich Ist (Steuerung) & Soll (Anpassung)
  - Anpassung BC an Bedarf, Wirtschaftliche Betrachtung bei Änderung der Rahmenbedingungen, Aufsetzen Controlling
- Betrieb / Nutzung – BC für Änderungen
  - Betrieb und Wartung, Anpassung an neue Anforderungen
  - Anpassung BC an Bedarf, Wirtschaftliche Betrachtung bei Änderung der Rahmenbedingungen, Controlling gegenüber Kennzahlen, Controlling zur Optimierung: Einhaltung von Budget, Terminen & Qulität. Maßnahmen bei Abweichung von BC. Überprüfung von Kosten & Nutzen. Zusätzlich Verankerung im Betriebszielen

4.4.2 BC-Analyse

Voraussetzungen

- Eigenständiges Projekt / Teil einer Vorstudie
- Besitzt Ziele, Planung & Team
- Nachvollziehbare Ergebnisse
- Sponsor: "Wer steht hinter Hauptprojekt?"
- Stakeholder Analyse
- Unternehmensvorgaben

Ablauf

- Definiton Ziele & Inhalte sowie Planung
  - Rahmenbedingungen: Ziel (Übereinstimmung mit Unternehmenszielen), Verfahren, Abhängigkeiten
  - Team & Know-How vorhanden
- Szenarien Definiton
  - Vergleich Referenzszenario (Keine Änderung) & Lösungsszenario (Investition) mit Delta zum Referenzszenario
  - Mehrere Lösungsszenarien oder Varianten möglich. Achtung: Kosten-Nutzen-Analyse für jedes Szenario nötig!
- Analyse
  - Kostenanalyse & Nutzenanalyse: Identifizierung von Positionen, Klassifizierung (relevant?), Quantifizierung & monetäre Bewertung
    * Kostenkategorien: Investitionskosten (Entwicklung, ...), Betriebskosten (Lizenzen, ...), Prozesskosten (Personal, ...), Erhöhtes Umlaufvermögen (Lagerbestände, ...); Einmalige vs. Laufende Kosten
    * Nutzenkategorien: Umsatz/Einnahmensteigerung, Produktivitätssteigerung, Reduktion Umlaufvermögen, Kostenreduktion; Monetärer vs. Indirekt- (Kundenzufriedenheit ⇒ Höhere Einnahmen) vs. Nicht-bewertbarem Nutzen (Imagegewinn); Nutzen meist regelmäßig (jährlich), "Hebbar" (Kein Verpuffen – 5 Min Arbeitszeit pro Tag),
  - Risikoanalyse: Besondere Risiken im Hinblick auf Wirtschaftlichkeit
- Wirtschaftlichkeitsrechnung
  - Ermittlung von Kosten-Nutzen-Verhältnis. Wichtigstes Ergebnis des BC
  - Darstellung in Kennzahlen
  - Eigene Rechnung für jedes Szenario, nach Vorgaben des Controlling

    * Investitionsrechnung: Statisch (Vereinfachte Annahmen über Durchschnitte): Kostenvergleichsrechnung, **Dynamisch** (Einbeziehung kalkulatorischer Zinsen): Kapitalwertmethode, Interne Zinsfußmethode
    * Kapitalwertmethode: Übersteigt Investitionsvertrag die Kapitalverzinsung? (Bringt die Investition mehr, als das Geld anzulegen?)
    * Interne Zinsfußmethode: Bei welchem Zinsfuß wird Kapitalwert null? (Wie müsste der Zins sein, dass sich Anlegen gegenüber Investition lohnt?)

- Dokumentation
  - Dauerhafte Dokumentation der Ergebnisse. Quantifizierung
  - Hard-Facts (Wirtschaftlichkeitsrechnung) + Soft-Facts (nicht quantifizierbarer Nutzen)

5 Aufwandsschätzung und Projektkalkulation von Großprojekten

5.1 Grundlage

- Expertenschätzung auf Erfahrung von Experten (Ähnliches Projekt mindestens 3x durchgeführt.)
- Schätzdatenbanken mit Functional Size Measurement (FSM). Normierung bzgl. Größe, Komplexität, Umfeld
- Kontextrahmen in verschiedenen Ebenen
  - Projektinhalt (PI): Spezifikation, Konstruktion, Realisierung, Integration, Inbetriebnahme
  - Projektquerschnitt (PQ): Projektkoordination, Projekttechnik
  - Projektnebenaufwände (PN): Team, ..
- Aufwandsmodell: Verbindliche Struktur in Aufgabenkategorien (Kontenrahmen), Gemeinsame Sprache, QS, Vergleichbarkeit ⇒ Systematisches Lernen
  - Aufwandskategorien (+ Kontenrahmen)
  - Aufwandsblatt (Dokumentation)
  - Nachkalkulation (Erfassung des Ergebnisses)
  - Kennzahlen (Plausibilisierung von Schätzungen)
- Festpreisrisikozuschlag
- Gewährleistungsaufschlag
- Nettoaufwand (PI)
- Bruttoaufwand (PI + PQ + PN)

- Bottom-Up: Aufwände getrennt geschätzt und dann summiert
  - Expertenschätzungen: Einzelschätzung, Delphi-Methode, Schätzklausur – Analogiemethode wenn möglich, Erstmalige Schätzung durch Expertenwissen
- Top-Down: Schätzung mit Algorithmus auf Basis von funktionalen Anforderungen
  - Algorithmische Verfahren: COCOMO, Function Points, Use Case Points – empirisch, messbare Größen benötigt, aufwändig, meist gute Resultate
  - Vergleichsmethoden: Analogiemethode – Bezug zu alten Projekten, Keine messbaren Größen nötig dafür Nachkalkulation
  - Kennzahlenmethoden: Multiplikatormethoden, Prozentsatzmethode – Ähnlich Analogiemethode aber Messdaten alter Projekte nötig

5.2 Bottom-Up Schätzung (Expertenschätzung)

- Schätzposten Projektspezifisch
- Notwendig für inhomogene / kundenspezifische Projekte

5.2.1 Arten von Expertenschätzung

- Einzelschätzung: Pragmatisch, ungenau
- Delphi-Methode (unabhängige Schätzung mehrere Experten, Zusammenführung durch Projektleiter, evtl. Iterationen): Verlässlich, aber Zeitaufwändig
- Schätzklausur (Online Schätzen während gemeinsamen Workshops): Risikobewusstsein, Gleicher Informationsstand, Besser als Mittelwerte, Zeitaufwändig

5.2.2 Schritte der Schätzung

- Stückliste erstellen, Schätzung: Nettoaufwand (Schätzposten zählen und Bewerten)
  - Stückliste enthält alle Aufwandsposten (Arbeitsergebnisse, Sonstige Tätigkeiten). Zusammenfassung in Schätzposten
  - Muss nicht 1:1 mit Arbeitsergebnissen / Planungseinheiten übereinstimmen
  - 1 Bearbeitertag (BT) = 8 Stunden (BH), 1 BW = 5BT, 1BM = 20BT, 1BJ = 10 BM = 1600BH. Rüstzeiten (Mails, kleine Pausen) miteinberechnet
  - Gesamtaufwand = Schätzung (Anforderungen als Grundlage) + Aufwandsrisiko (X% der Schätzunsicherheit; Nicht bei jedem Schätzposten)
- Querschnittsaufgaben als %-Werte: Bruttoaufwand

  - Querschnittsaufgaben möglichst konkret abschätzen, Erfahrungswerte in % vom Nettoaufwand
  - Bsp: Projektleitung, Chef-Design, QS, Infrastruktur, Reisezeiten, Team-Sitzungen, Trainings
- Bewertung mit Stundensätzen, + FP-Risiko / Gewährleistung: Gesamtbudget
- Plausibilisierung durch Projektplan, Mitarbeitergebirge, Projektphasen, Vergleiche: Plausibilisiertes Budget
  - Teamgröße: Brooks Faustregel (Teamgröße = $\sqrt{\text{Aufwand in BM}}$). Produktiver Anteil (Arbeitsteilung) vs. Kommunikativer Anteil (Abstimmung)
  - Mitarbeitergebirge: Projektverlauf skizzieren (Teamgröße, Dauer) & Fläche berechnen. 1 Zeitmonat (ZtM) = 0,8 BM
- Soll- / Ist-Vergleich: Budgethochrechnung
  - Parameter als kalkulatorische Vorgaben Konkret / % vom Bruttoaufwand abschätzen
  - Bsp: Stundensatz, Bruttoaufwand × Stundensatz, Reisekosten, Festpreisrisikozuschlag, Gewährleistung, Sonstige Kosten

5.2.3 Grundsätze

- Konkret: Aufwandsposten konkret schätzen statt %-Schätzungen
- Schätzunsicherheit bei Aufwandsposten. Schätzposten haben nur erhöhte Aufwandszahl
- Aufwandsblatt zur Dokumentation
- Vollständigkeit und Plausibilisierung durch Aufwandsblatt
- Prämissen, wenn Vorgaben unscharf

5.3 Top-Down Schätzung (Use-Case Points)

- Messung Funktionaler Größen fachlicher Anforderungen (FSM)
- Annahme: Gleicher Aufwand, bei gleichen Anforderungen ⇒ Mathematischer Algorithmus
- Funktionale Grüße der Anforderungen (Punkte)
- Plausibilisierung von Expertenschätzungen (Vergleich mit UCP Schätzung, Basis ist Use-Case basierte Spezifikation) / Nachkalkulation (Vergleich Ist mit UCP, Basis ist Stückliste der Realisierung)
- Vorbedingungen: Grobspezifikation (Inception) vorliegend
- Schätzumfang: Feinspezifikation (Elaboration + Construction) bis Bereistellung + QS

5.3.1 Schritte der UCP-Schätzung

- Use-Cases klassifizieren: Systemverhalten nach Aktion eines Aktors; Use-Cases + Aktoren = Funktionaler Umfang des Projektes (A-Faktor)
- Aktoren klassifizieren
- T-Faktor ermitteln: Nicht-funktionale Anforderungen – Fragenkatalog
- M-Faktor ermitteln: Organisatorische Komplexität – Fragenkatalog
- Aufwand berechnen: Aufwand = A-Faktor × T-Faktor × M-Faktor × PF (Produktionsfaktor = z.B. Projektmanagement – Proportional zu Use-Case-Points)
- A + T = Systemanforderungen, M + PF = Projekteinfluss

5.3.2 Use Cases

- Komplexität
  - Einfach (0-3 Schritte / Dialoge /Szenarien): 5 Punkte
  - Mittel (4-7 Schritte / Dialoge /Szenarien): 10 Punkte
  - Schwer ($\geq$8 Schritte / Dialoge /Szenarien): 15 Punkte
- Schritt
  - Geschlossener, fachlicher Teil eines Use-Cases; Eindeutige Abtrennung von anderen Schritten (Aktorenwechsel, Zwischenergebnisse, Aufspalten von Szenarien)
  - Schritte in mehreren Szenarien werden nur einmal gezählt
  - Bsp: Dialogeingabe, Aufruf von Andwendungsfunktion, Server-Transaktion
- Dialoge
  - Reiter werden als einzelne Dialog gezählt; Druckstücke; Anzeigeseiten zählen nur, wenn Dateneingabe möglich
  - Pop-Ups, Bestätigungen keine eigenen Dialoge
  - Komplexitätsunterschiede von Dialogen nicht gut erfasst
- Szenarien
  - Äste in Verlaufsdiagramm (Erfolgsszenarien, Nicht-Triviale Fehlerszenarien)
  - Erfolgsszenarien: Hauptszenario, Fachliche Alternativszenarien
  - Fehlerszenarien nur, wenn fachliche Fehlerbehebung (Abbruch ist kein eigenes Szenario)
- Angemessene Granularität wichtig, Ausgewogene Verteilung kleiner/mittlerer/großer Szenarien, Schritte sollten etwa gleich groß sein
  - Max. 1 A4-Seite, max. 12 Schritte, max. 7 Szenarien
  - Min. 1 Dialog, min. 3 Schritte, min. 2 Szenarien

5.3.3 T- & M-Faktor

- Normieren technische & organisatorische Komplexität des Projektes
- Faktor 1: Keine Vereinfachung / Probleme erwartet

5.3.4 Eignung von UCP

- Individualentwicklung
- Neuentwicklung (fachlicher Geschäftsprozesse)
- Stammdaten-Systempflege
- NICHT: Produktanpassung, Wartung (kleine Anpassungen), Technikstufen
- Umfang muss durch Use-Cases erfassbar sein (A-Architektur)
- Bei Nutzung von UCP teilweise unschärfe nötig. Use-Case-Points müssen zum Projekt passen & Schätzbasis muss vorhanden sein (garbage in – garbage out)

6 Mobile Lösungen im Enterprise Einsatz

6.1 Consumerization of IT

- Neue Technologie wird zuerst im Consumer-Bereich, dann im Enterprise Bereich eingesetzt
- Neuen Technologien wie Smartphone, Tablets, ... für Unternehmensanwendungen. Know-How mehrseitig (Consumer / Enterprise) einsetztbar.
- Zunahme von Smartphones innerhalb von Unternehmen durch offizielle Anschaffung oder Mitarbeiter

6.2 Mobile Clients

- Mobile Geräte (Smartphones, Handscanner, ...)
- Software (LCD-Display, Betriebssysteme, ...)

6.2.1 Unterschiede

- Betriebssystem (z.B. iOS, Android, Windows Phone, BlackBerry, Bada, Symbian)
- Display
- Akkulaufzeit
- Schnittstelle zur Datenkommunikation
- Tastatur
- Sturzschutz
- Art der Datenerfassung
- Betriebstemperatur
- ...

6.3 Einsatz von mobilen Clients

- Datenerfassung
- Datendarstellung
- Steuerung

6.3.1 Mobile Datenerfassung

- Optimierung von Geschäftsprozessen: Weniger Redundanz, Keine Medienbrüche, weniger Übertragungsfehler, Aktuelle Datenbasis, Effizienzsteigerung, Einspareffekte
- Automatisierung nachgelagerter Prozesse (Beachtung von Schnittstellenthematik)
- Beschleunigung der Erfassung: Schnelle Verfügbarkeit von Daten, Prozessbeschleunigung durch feinere Planung
- Höhere Datenqualität: Plasibilitätsprüfung der Datenentstehung, Intrinsische Datenprüfung (Checksumme), Interaktive Führung, Keine nachgelagerten Prozesse nötig

6.3.2 Arten der Datenerfassung

- Manuelle Eingabe (GUI) – Günstig, Schwierige Bedienbarkeit
- Optische Erfassung (Barcodes)
- Elektromagnetische Erfassung (RFID) – Teuer, Einfache Bedienbarkeit

6.3.3 Einsatzgebiete

- Service-Außendienst
- Rettungsdienst
- Feuerwehr
- Gebäudemanagement
- ...

6.4 Architektur für mobile Lösungen im Enterprise Ansatz

6.4.1 Eigenschaften betrieblicher Informationssysteme

→ Architekturprinzipien betrieblicher Informationssysteme

6.4.2 Software-Kategorien trennen Zuständigkeiten

→ Architekturprinzipien betrieblicher Informationssysteme

6.4.3 Komponentenschnitt soll Abhängigkeiten minimieren

→ Architekturprinzipien betrieblicher Informationssysteme

6.4.4 Netzwerkarchitekturen

- GPRS, Gradle, WLAN, ...
- Verschiedene Komponenten

7 Von der Idee zum Produkt

7.1 Prototyping

7.1.1 Anforderungen

- Schnell, einfach, günstig
- Leicht änderbar
- Skizze / Kein fertiges System → Besseres Feedback, Mehr Ideen
- Änderungen ad hoc (im Meeting) einbauen
- Ausführbar, interaktiv (Workflow)

7.1.2 Prototypen im mobilen Umfeld

- Usability-Herausforderungen: Kleiner Bildschirm, Tastatur, Anderes Bedienkonzept, Style-Guides, Umegbung, Intuitiv, ...
- Möglichkeiten: Gesten, Bilschirmorientierung, Sensoren, Kompass, Ortung, Kamera, NFC

7.1.3 Prototyping Tools

- Nutzen von Prototyping Tools
  - Fortentwicklung im Team
  - Vorgefertigte Templates
  - Interaktive Bedienbarkeit
  - KEIN Fachkonzept
  - KEIN Source Code (Wegwerfen!)
  - Horizontal: Abfolge von Ereignissen
  - Vertikal: Schnitt durch Technologien
- Balsamiq Mockups
  - Flache Lernkurve
  - Intuitive Erstellung schneller Layouts, aber keine Bedienung
  - Scribble Optik
- Axure RP
  - Höhere Lernkurve
  - Funktionsumfang z.B. Interaktive Bedienbarkeit & Multiuser
  - Scribble Optik bis zu Glänzender Optik
  - HTML-Prototyp möglich
- Fuid UI

7.1.4 Papier oder Software

- Papier
  - Schnell & intuitiv
  - Keine Vorkenntnisse nötig
  - Keine Einschränkung
  - Kollaborativ
  - Post-Its
  - Schablonen
- Software
  - Baukasten
  - Wiederverwendbarkeit (Copy & Paste)
  - Digitalisiert
  - Verschicken
  - Desktopsharing
  - Klick-Dummy
- Programmieren
  - Live-Dome mit allen Möglichkeiten
  - Langsam & Teuer
  - Architekturfalle

7.2 Methodik für die Entwicklung mobiler Lösungen

7.2.1 Neue Bedien-Philosophien

- Gestensteuerung
- Ortung
- Keine Standard-Orientierung
- Ubiquitus Computing: Metaphern verwenden, Strukturieren, Auswählen statt Eingeben, Benutzerkontrolle
- Zielgruppenspezifische Modi

7.2.2 Definiton von Anwendungsfällen

- Brainstorming
- Priorisierung

7.2.3 Auswahl von Technologie & Architektur

- Web Technologie / Common Device API / Device Specific API
- Single Device OS / Multi Device OS / Multi Device with WebKit
- Nativ / Hybrid / HTML5 Web
- Von der Idee zum Produkt: Entwicklung, Architektur, Deployment
- Programmiersprache
  - Mainstream (Einfachheit)
  - Dokumentation
  - Lebenszyklus
  - Verfügbarkeit
  - API
  - Algorithmen
  - Netzweranbindung
  - Offlineverhalten
  - Lokale Datenhaltung
  - Datensicherheit
- Entwicklungsumgebung / Werkzeuge
  - Einrichtung
  - Nutzungskonzepte
  - Programmiersprachenunterstützung
  - Plattformunabhängigkeit
  - Dokumentation
  - UI Editor
  - Gerätesimulator
  - Lokaler Compiler
  - Anzahl von Werkzeugen
- User Interface
  - Trennung von Design und Code
  - Freiheit des optischen Designs
  - Performance
  - Look & Feel
- Mutlimedia-Unterstützung
- Frameworks & Bibliotheken
- Hardware
  - Sensoren

  - Bildschirmauflösung
  - Anzahl unterstützter Geräte
- Betriebssystem
  - Verschiedene Betriebssysteme (& Versionen)
  - Native Betriebssystemfunktionen
  - Native Anwendungen
  - Multi-Platform
- Zukunftssicherheit
  - Support
  - Community
  - Aktivitäten
  - Feature Support
  - Entwicklerverfügbarkeit
  - Reifegrad
- Lizenzen
- Preis

7.2.4 Design

- Umsetzung des Prototyps

7.2.5 Von der Idee zum Produkt

- Anwendung: Konzeption & Design
- Technologie: Architektur & Sicherheit
- Integration: Technische Möglichkeiten & Unternehmensintegration

7.3 Von der Geschäftsprozessbeschreibung zum Story Board

7.3.1 Storyboard-Inhalte passgenau erarbeiten

- Anforderungen erfragen
- Anforderungen verstehen (Benutzer beobachten)
- Benutzerverhalten hinterfragen
- Anwendungsfall durchspielen

7.3.2 Mobile Anwendungen in der Verwaltung

- Gewerbeaufsicht: Mobile Betriebsprüfung
- Ordnungsamt: Kontrollen, Bußgelder, Schadensmeldungen
- Elektronische Akte: Signaturen
- Kennzahlen-Cockpit: Management, Statistik
- Zoll & Finannzaufsicht: Mobile Einsatzgruppen
- Polizei: Mobile Abfragen

$\Rightarrow$ Anwendungsfälle mit Fachseite erarbeiten

7.3.3 Weitere Relevante Themen

- Business Case
- Cost Case
- Geschäftsprozessmodellierung
- Branchenspezifik
- Fachliche Methodik
- Make or Buy-Entscheidungen vorbereiten

7.3.4 Betriebswirtschaftliche Aspekte

- Wettbewerbspotentiale (Fachlogik, Produktkonfigurator, ...) $\Rightarrow$ Make
- Commodity (Software Update, Security, Datenbank , ...) $\Rightarrow$ Buy

8 Scrum Lego-City Hands-on Training

8.1 Motivation für agiles Vorgehen

- Beschleinigte Markteinführung
- Besserer Umgang mit Änderungen, Prioritäten, Umfang und Anforderungen
- Kundeneinbindung
- Frühe Risikominimierung
- Bessere Sichtbarkeit des Projekts
- Kunden haben agile Unternehmenskultur (Auch Voraussetzung)

8.2 Stabilität

- Klassisch – Stabilität
  - Projektleiter
  - Lange Entwicklungszyklen
  - Wasserfall-ähnliche Konkretisierung
  - Plangesteuert
  - Formare Change-Request Prozess & Nachvollziehbarkeit
  - Dokumentations- und Qualitätskontrollen von Zwischenergebnissen
  - Schwergewichtiger Prozess
- Iterativ
  - Projektleiter
  - Iterativ inkrementelles Vorgehen
  - Priorisierung
  - Plangesteuert
  - Häufiges Kunden-Feedback
  - Dokumentations- und Qualitätskontrollen von Zwischenergebnissen
  - Schwergewichtiger Prozess
- Agil – Agilität
  - Transparenz & Kommunikation (Verteil auf mehrere Personen: Team)
  - Iterativ – Inkrementell (Sprints)
  - Priorisierung
  - Anpassungsfähige Planung
  - Aktive Kunden
  - Selbständige / Eigenverantwortliche Teams
  - Leichtgewichtiger Prozess

8.3 Scrum

8.3.1 Charakteristika

- Leichtgewichtiger Management-Rahmen
- Einfach verständluch, schwer zu meistern
- Selbstorganisierende Teams
- Kurze Entwicklungszyklen
- Anforderungen sequenziell priorisiert
- Regeln, um agiles Umfeld für Auslieferung schaffen (Keine spezifische Entwicklungsmethode)

8.3.2 Gefahren

- Einführung ohne Erfahrung mit Scrum
- Beteiligte nicht reif für Scrum
- Passt nicht in Organisation
- Festpreisprojekte

8.3.3 Drei Säulen von Scrum

- Empirische Prozesssteuerung (Wissen aus Erfahrung gewonnen, Entscheidung auf Grundlage von Fakten)
- Iterativer / Inkrementeller Ansatz $\Rightarrow$ Vorhersagbarkeit / Risikokontrolle
- 3 Säulen:
  - Überprüfung
  - Anpassungsfähige
  - Transparenz

8.3.4 Agiles Wertesystem

- Agile Manifesto:
  - Team wichtiger als Individuum
  - Funktionierende Software wichtiger als umfangreiche Dokumentation
  - Kooperation wichtiger als Vertragsverhandlungen
  - Änderungen wichtiger als Pläne
- Agile Prinzipien
  - Heiße Änderungen an Anforderungen willkommen
  - Funktionierende Software wichtigstes Bewertungsmerkmal
  - Nachhaltige Entwicklung

8.4 Scrum auf einen Blick

- Product Backlog: Liste priorisierter Features (Product-Owner)
- Sprint Planning Meeting: Sprint Backlog & Detail-Planung (Team + Scrum Master)
- Sprint mit Daily Scrum Meating: Was seil letztem Meeting geschafft? Was hat mich behindert? Was bis nächstes Meeting?
- Sprint Review: Abnahme des Sprints
- Sprint Retrospektive: Feedback

8.5 Scrum-Rollen

- Entwicklungsteams
  - 5-9 Personen
  - Funktionsübergreifend (QS, UI, ...)
  - Vollzeitmitglieder
  - Änderung nur zwischen Sprints
- Product Owner
  - Bedürfnisse (Anforderungen)
  - Releasplan
  - Backlog
  - Akzeptiert Arbeitsergebnisse (Produkterfolg)
- Scrum Master
  - Verantwortlich für Scrum-Prozess
  - Coach. Hilft Hindernisse zu Beseitigen
  - Schutzt vor Störungen
  - Versucht Lerprozess anzustoßen
  - NICHT Inhaltliche Arbeit, Umsetzung, Weisungsbefugnis

8.6 Scrum Artefakte

- Product Backlog
  - Features (User Stories)
  - Priorisiert
- Definiton of Done (Leitlinien für Fertigstellung)
- Sprint Backlog
  - Userstories des aktuellen Sprints
  - In Tasks aufgeteilt (Aufwandsschätzung)

- Impediment List
  - Liste von Hindernissen
  - Teamverwaltet
  - Team-Interne Hindernisse selbt behoben
- Burndown-Charts (Visualisierung von Arbeit. Sprints / Release)

8.6.1 Product Backlog & User Stories

- Product Vision: Idee für Entwicklung. Detailiert durch User Stories
- User Story: Beschreibt Anforderung. Konkreter Mehrwert
- Product Backlog: Einzelne User Stories. Datum, Sequenz (Priorisierung), Nahe User Stories feinere Beschreibung, Storypoints (Komplexität)