Informationssammlung

# Agile Softwareentwicklung

## Das kleine Agile-Buch – Sander Hoogendoorn (2013)

1. **Wasserfall (S.15)**
   * Ursprung
   * Probleme
     + Wissensverlust nach jeder Phase
     + Wissen aus momentanen Phasen kann nicht mehr in die abgeschlossenen Dokumente einfließen
     + Anforderungen ändern sich
     + Vollständigkeit
     + Schätzung
2. **Agiles Manifest (S.21)**
   * 4 Prinzipien:
     + Individuen und Interaktion mehr als Prozesse und Werkzeuge
     + Funktionierende Software mehr als umfassende Dokumentation
     + Zusammenarbeit mit dem Kunden mehr als Vertragsverhandlungen
     + Reagieren auf Veränderung mehr als das Befolgen eines Plans
   * Linke Aussage wird stärker betont, jedoch wird die rechte Aussage nicht ausgeschlossen
   * Zusammenarbeit zwischen den Rollen
   * Rollen arbeiten miteinander/gleichzeitig 🡪 optimale Dokumentationsweise finden
   * Zusammenarbeit mit Kunden 🡪 Anforderungsanpassungen
   * Merkmale:
     + Kurze Iterationen
     + Zusammenarbeit der Teams
     + Kleine Arbeitseinheiten (Use Cases)
3. **Iterationen (S.33)**
   * Kurz 🡪 aber Projekt abhängig
   * Ablauf: Kickoff 🡪 Arbeit 🡪 Evaluation
   * Fester Zeitraum (Timebox)
   * Vorteile:
     + Risiken werden früher deutlich/entdeckt
     + Schnelle Veränderungen möglich
     + Rollen profitieren voneinander
     + Fortschritt ist gut messbar
4. **Kickoff**
   * Wahl der Arbeitseinheiten:
     + Wert
     + Zusammenhang mit anderen
     + Risiken
     + Balance: Rollen auslasten
   * 🡪 Kunde such aus NICHT Projektleiter (in Zusammenarbeit mit Team)
   * ! nicht zu viele und nicht zu wenige Use Cases einplanen
5. **Evaluation (S. 41)**
   * Verbesserungen können in den folgenden Iterationen sofort einfließen
   * Teilnehmer: Auftraggeber, Endanwender, Team
   * Umsetzung der Work Items präsentieren und Fortschritt klarstellen
   * Arbeitsweise
   * Tipp: Kickoff für nächste Iteration gleich im Anschluss
6. **Teams/Zusammenarbeit (S.50)**
   * Blickwinkel jeder Rolle 🡪 multidisziplinär
   * Kleine/kompakte Teams 🡪 Bildung mehrerer Teams bei zu großer Anzahl
   * Multidisziplinäre Mitglieder 🡪 Team deckt alle Aufgabenbereiche ab
   * Never change a winning Team 🡪 Teams auf Basis des Zusammenspiels bilden 🡪 Projekte den Teams zuordnen (nicht umgekehrt)
7. **Meetings/Techniken (S.56)**
   * Kickoff (zu Beginn jeder Iteration)
   * Standup (täglich)
     + Transparenz: Fortschritt, Hilfe, Probleme
     + Im Stehen 🡪 10-15 min
   * Evaluation (Ende)
   * Workshops
     + Multidisziplinär
     + Alle Ansichten treffen zusammen
     + Schnelle Entscheidungsfindung
     + Keine Übergaben
   * Pair Programming (an einem PC)
     + Driver + Navigator
     + Wissen teilen und voneinander lernen
     + Schneller zur Lösung
     + Code ist weniger komplex und weniger Code
     + Cleaner Code
   * Side-by-Side (jeder an einem PC)
     + Entwickler arbeiten intensive miteinander
     + Work Item in mehrere Teile zerlegen, die parallel entwickelt warden können
     + Hohe Produktivität
   * Pair Testing (mit Endanwender)
     + Tester in Kombination mit anderen Rollen einsetzen (spezielles
     + Tester als Navigator und Endanwender als Driver
8. **Rollen (S.72)**
   * Rollendefinition ist Projekt abhängig
   * Für jedes Projekt gilt:
     + Alles benötigte Fachwissen muss vertreten sein (Analyse, Entwicklung, Tests)
     + Generalisierende Spezialisten 🡪 bringen weitere Erfahrungen und Fachkenntnisse mit
     + Team und Rollenbesetzung nicht ändern
     + Automatische ausgewogene Beschäftigung
     + Externe Spezialisten für bestimmte Zeit