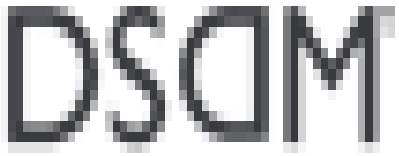
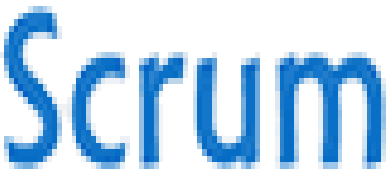
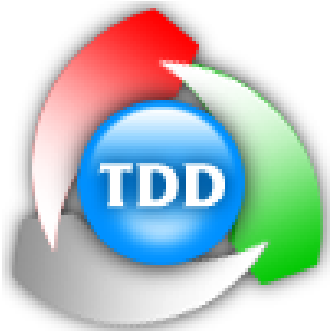
Agile Software-Entwicklung: Überblick







Inhalt

- Historie
- Agiles Manifest
- Agile Prinzipien
- Agile Methoden
- Agile SW-Entwicklungsprozesse

Historie

- Agile SW-Entwicklung als Gegenbewegung zu “schwergewichtigen” SW-Entwicklungsprozessen, z.B. Rational Unified Proces (RUP), V-Modell
- erste Ansätze Anfang der 1990er Jahre
- 1999: Kent Beck, “Extreme Programming”
- 2001: Agiles Manifest

Agile Werte: Das Agile Manifest

- Individuen und Interaktionen sind wichtiger als Prozesse und Tools.
- Funktionierende Programme sind wichtiger als ausführliche Dokumentation.
- Die stetige Zusammenarbeit mit dem Kunden ist wichtiger als die Verträge.
- Der Mut und die Offenheit für Änderungen sind wichtiger als das Befolgen eines festgelegten Plans.

www.agilemanifesto.org

Agile Prinzipien (1)

- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The best architectures, requirements, and designs emerge from self-organizing teams.

Agile Prinzipien (2)

- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Agile Prinzipien (3)

- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Working software is the primary measure of progress.
- Continuous attention to technical excellence and good design enhances agility.

Agile Prinzipien (4)

- Simplicity --the art of maximizing the amount of work not done-- is essential.
- Business people and developers must work together daily throughout the project.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Agile Methoden

- Pair Programming
- Test Driven Development (TDD) / Test First Development
- Ständige Refactorings
- Story Cards
- Quick Code Reviews

Agile Methoden: Pair Programming (1)

- 2 Entwickler an 1 Rechner
- häufiger Rollenwechsel
- häufiger Wechsel der Zusammensetzung der Paare

Agile Methoden: Pair Programming (2)

- Vorteile:
  - Steigerung der Qualität: höhere Disziplin, besserer Code
  - Collective Code Ownership
  - Mentoring, Team Building
- Nachteile:
  - Kosten (Kostenvorteile erst spät in Entwicklung)
  - Teamfindung
  - Autoritätsproblem (bei Streitfragen)
  - Vorurteile bezüglich Zeitaufwand und Produktivität

Agile Methoden: Test Driven Development (1)

- TDD / Test First Development
- “Extreme Testing”
- Test wird vor eigentlichem Code implementiert
- wichtig: automatisierte Ausführung von Tests

Agile Methoden: Test Driven Development (2)

- Vorteile:
  - testbarer Code
  - Grey-Box-Test
    - eigenschaften von White- und Black-Box-Test
  - Erfüllung der Anforderungen messbar
  - gefahrlose Refactorings möglich
- Nachteile:
  - Einstieg schwierig durch ungewohnte Denkweise
  - Achtung: keine falsche Sicherheit, da nicht alles testbar ist!

Agile Methoden: Ständige Refactorings (1)

- Strukturverbesserung unter Beibehaltung des Verhaltens
- kein klassisches Redesign, sondern Änderung “im Kleinen”
- manuell oder mit Tool-Unterstützung, z.B. Eclipse
- Unit-Tests als Regressionstests
- mögliche Refactorings:
  - Namensänderungen (Variablen, Methoden, Klassen, ...)
  - Verschiebungen (z.B. Methode in andere Klasse)
  - Aufteilung
  - Zusammenlegung

Agile Methoden: Ständige Refactorings (2)

- Vorteile:
  - bessere Lesbarkeit, Verständlichkeit, Wartbarkeit, Erweiterbarkeit
  - weniger Redundanz
  - bessere Testbarkeit
  - Aufwand für Fehlersuche und Erweiterungen sinkt
- Nachteile:
  - Risiko unerwünschter Änderungen und Fehler
  - Zeitaufwand, wenn Tests erst implementiert werden müssen

Agile Methoden: Story Cards

- Kunde erstellt Kurzbeschreibung (Prosa, User Story)
- 1 Story Card für 1 Funktion des Systems
- kein Technik-Jargon
- Medium frei wählbar: Karteikarte, Wiki, spezielle SW, ...

Agile Methoden: Quick Code Reviews (1)

- Durchsicht von Quelltexten
- manuell und/oder automatisiert
- Tools, z.B. CheckStyle, PMD, FindBugs
- Kriterien für Review festlegen

Agile Methoden: Quick Code Reviews (2)

- Ergebnisse:
  - verbesserter Code: Effizienz, Robustheit, Wartbarkeit
  - verbesserte Kommentare: Wartbarkeit
  - neue Testfälle: Robustheit
- XP-Verfechter: weniger Code-Reviews notwendig durch Pair Programming und Test Driven Development

Agile Prozesse (1)

- auf Agilität ausgerichteter SW-Entwicklungsprozess
- setzt überwiegend Agile Methoden ein
- Ziel: Prozess für alle Beteiligten effektiver und vorteilhafter gestalten

Agile Prozesse (2)

- Extreme Programming (XP) → XP-Vortrag

- Feature Driven Development (FDD) → FDD-Vortrag

- "The Eclipse Way" → Eclipse-Vortrag
- SCRUM

- Crystal

Agile Prozesse (3)

- Adaptive Software Development (ASD)
- Dynamic System Development Method (DSDM)
- Lean Development
- Pragmatic Programming
- Software Expedition
- Universal Application

Scrum: Rollen

- Product Owner: Auftraggeber, setzt Prioritäten
- Team: Entwicklungsteam
- Scrum Master:
  - Leitung des Teams
  - sorgt für Transparenz
  - Verbesserungen

Scrum: Artefakte

- Product Backlog:
  - enthält alle Features des Produkts (unvollständig)
  - bewertet und priorisiert
  - Aufwandsschätzung für hoch priorisierte Features (max. 16h)
  - auch technische und administrative Aufgaben
- Sprint Backlog:
  - alle Aufgaben eines Sprints (jeweils max. 1 Tag)
- Impediment List:
  - Hindernisse des Projekts

Scrum: Sprint

- Umsetzung einer Iteration
- max. 30 Tage
- vorher: Anforderungen in Product Backlog sammeln
- Sprint Backlog erstellen (mit Kunden)
- Team organisiert sich im Sprint selbst
- keine Vorschriften
- danach: Vorführung, Review, Akzeptanztests
- Ergebnisse des Reviews: Product Backlog

Scrum: Scrum-Meeting

- tägliches Meeting
- max. 15 Minuten
- Was wurde fertiggestellt?
- Was kommt heute?
- Welche Probleme gibt es?
- Impediment List pflegen

Scrum: Retrospektive

- wertfreier Rückblick auf Sprint, Ergebnisse notieren
- Was war gut?
- Was könnte verbessert werden?
- Verbesserungsvorschläge zur Organisation: Impediment List
- Verbesserungsvorschläge zum Team: Product Backlog
- Impediment List: Hindernisse werden von Scrum-Master und Team aus dem Weg geräumt

Crystal (1)

- Crystal Family: Familie von SW-Entwicklungsmethoden
- Mitglieder der Familie durch Farben bezeichnet
- Mitglied definiert Satz von Regeln
  - Rollen
  - Menge der Methoden
  - Umfang der Dokumentation

Crystal (2)

- Auswahl:
  - nach Anzahl der beteiligten Personen (Kommunikationsaufwand) und
  - Höhe des Risikos (Ausmaß des Schadens beim Scheitern des Projekts)
- Crystal Clear: einfachste Variante, bis 6 Projektbeteiligte
- projektorientiert
  - Gegensatz: XP ist an Art der Arbeit orientiert

Crystal: Prinzipien (Auswahl)

- passiver Wissenstransfer durch räumliche Nähe
- freie Meinungsäußerung
- laufende Kritik und Verbesserung
- fokussiertes Arbeiten (keine Ablenkung, keine anderen Projekte)
- häufige Releases
- Zugang zu kundigen Benutzern
- automatisiertes Testen