

SEPM Vorlesung – Block 5
Software Engineering & Projektmanagement

Ausgewählte Software Prozesse

Dietmar Winkler

Vienna University of Technology
Institute of Information Systems Engineering
Research Group for Information and Software Engineering

dietmar.winkler@tuwien.ac.at
<http://qse.ifs.tuwien.ac.at>

Motivation und Zielsetzung

- § Die Herstellung von qualitativ hochwertigen Softwareprodukten innerhalb von zeitlichen und budgetären Rahmenbedingungen erfordert ein systematisches Vorgehen.
- § Grundlegende Vorgehensweise
 - Systematische und strukturierte Vorgehensweise durch Softwareprozesse.
(wann soll welches Produkt in welchem Fertigstellungsgrad verfügbar sein)
 - Konstruktive Methoden zur Herstellung von Software Produkten,
z.B. für Spezifikationen, Testfälle, Source Code.
 - Analytische Methoden zur Überprüfung der Produktqualität,
z.B. Reviews, Inspektionen und Tests.
- § Vorgehensmodelle (Softwareprozesse) unterstützen den Projektleiter und das Entwicklungsteam durch die Bereitstellung eines Rahmenprozesses für den Projektablauf.
- § Vorgehensmodelle orientieren sich grundsätzlich am Software Life-Cycle Prozess, betrachten also alle wesentlichen Schritte eines Entwicklungsprozesses.
- § Unterschiedliche Projekte (Projektgröße, Anwendungsdomäne, Projekttyp) erfordern aber passende Vorgehensweisen!

à Auswahl eines passenden Software Prozesses.

Table of Contents

- § Software Life-Cycle (Wiederholung)
 - Phasen im Software Life-Cycle
 - Vom Software Life-Cycle zum Software Prozess

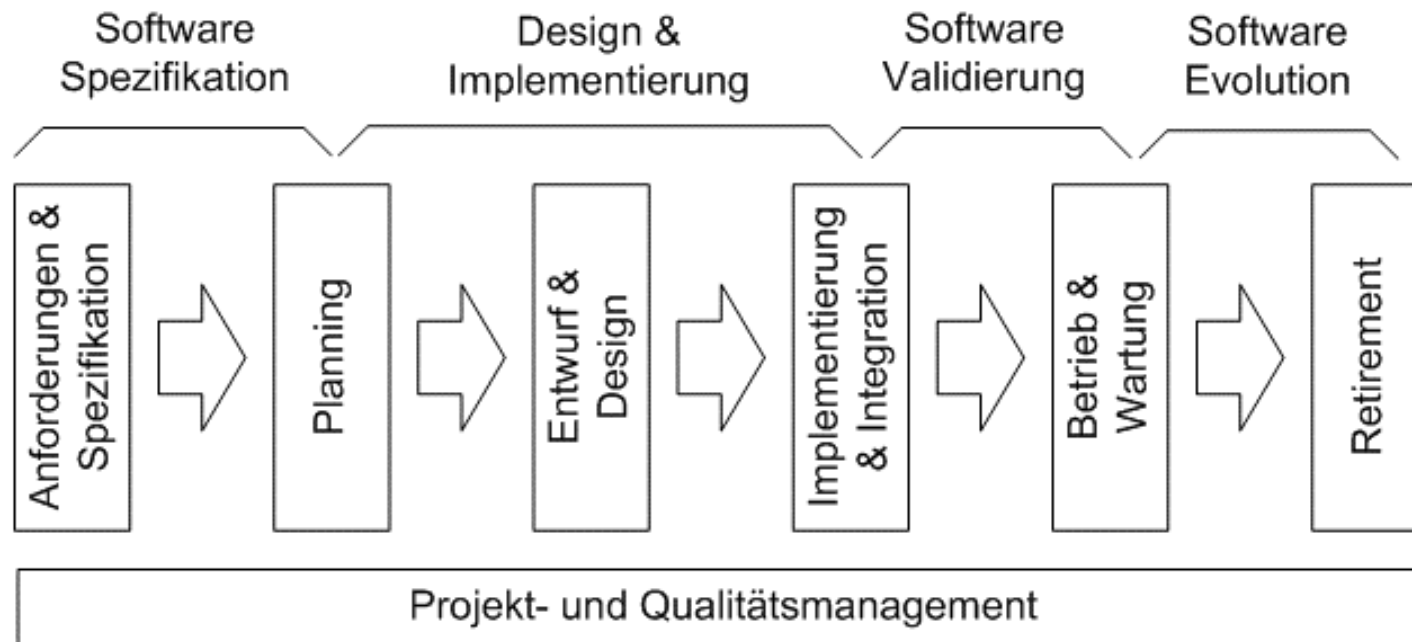
- § Traditionelle Ansätze
 - Wasserfall Modell
 - V-Modell Grundkonzept
 - V-Modell XT
 - Rational Unified Prozess

- § Agile Ansätze
 - Agiles Manifest
 - Scrum

- § Anpassung von Software Prozessen (Prozess Tailoring)

Software Life-Cycle

- § Ein Software-Prozess ist eine **Abfolge von Schritten** (Phasen) mit all seinen Aktivitäten, Beziehungen und Ressourcen.
- § Einsatz von qualitätsverbessernden Maßnahmen **in allen Phasen des Life-Cycles**, d.h. von der ersten Idee über die Entwicklung bis zum kontrollierten Auslauf des Produktes.
- § Der Software Life-Cycle beschreibt ein **Basiskonzept** für Software Engineering Prozesse und Vorgehensmodelle.



Vom Software Life-Cycle zum Vorgehensmodell

- § Die grundlegenden Phasen des **Software Life-Cycles** finden sich in allen Projekten.
- § Der Schwerpunkt der meisten Vorgehensmodelle liegt eher auf der **technischen Seite**, sie beginnen bei der Definition der Anforderungen und enden bei der Inbetriebnahme beim Kunden.
- § In der Praxis finden wir eine Vielzahl von unterschiedlichen Prozessmodellen
 - **Standardisierte** “common” Prozessmodelle (V-Modell, RUP, Agile Ansätze)
 - **Unternehmensspezifische** Vorgehensmodelle, die an die Bedürfnisse der jeweiligen Unternehmen bzw. Projekte angepasst werden.
- § Software Prozesse oder Vorgehensmodelle sind auf bestimmte **Kriterien** zugeschnitten und können – **je nach Projektkontext** – sinnvoll eingesetzt werden.

Ein Vorgehensmodell entspricht einer konkreten Strategie zur kontrollierten Durchführung eines spezifischen Projektes.

Table of Contents

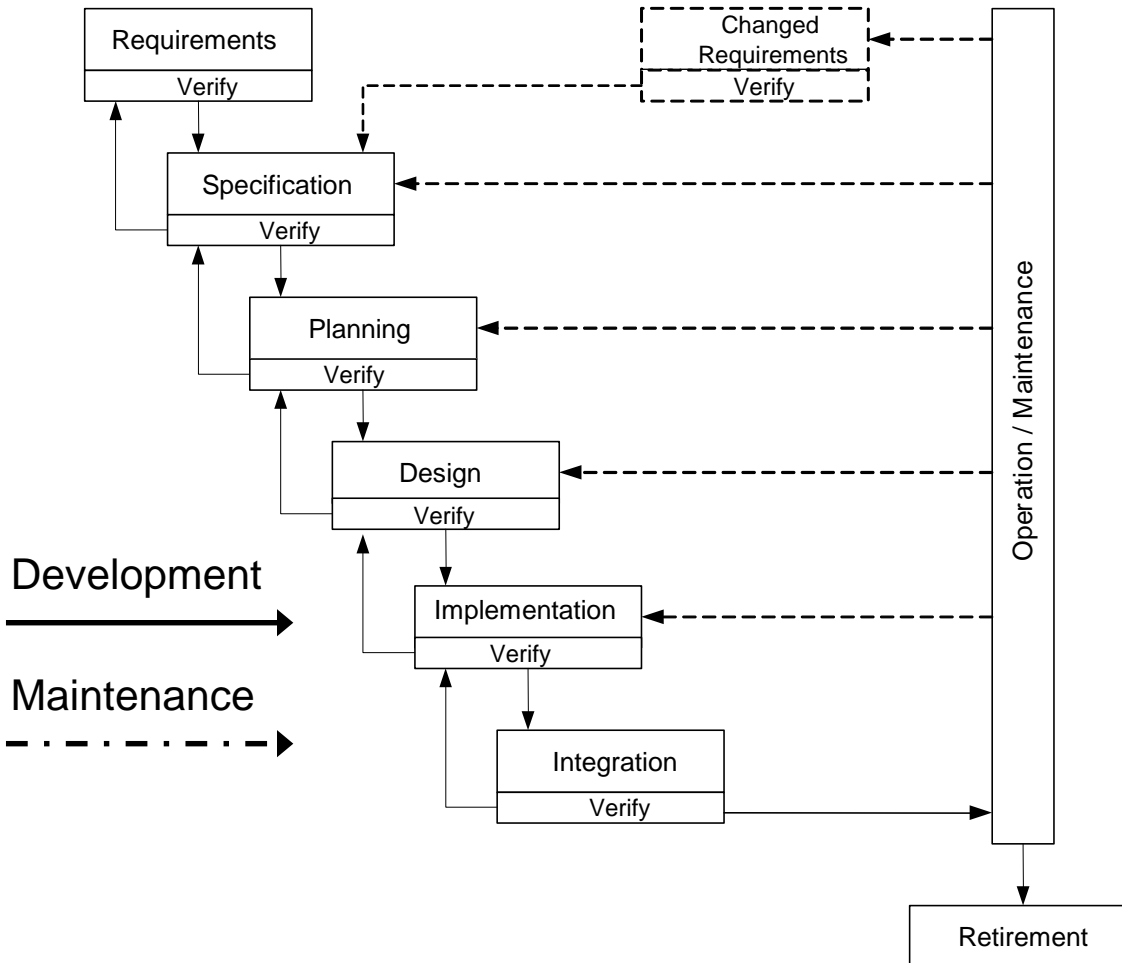
- § Software Life-Cycle (Wiederholung)
 - Phasen im Software Life-Cycle
 - Vom Software Life-Cycle zum Software Prozess

- § Traditionelle Ansätze
 - Wasserfall Modell
 - V-Modell Grundkonzept
 - V-Modell XT
 - Rational Unified Prozess

- § Agile Ansätze
 - Agiles Manifest
 - Scrum

- § Anpassung von Software Prozessen (Prozess Tailoring)

Wasserfall Modell (1)



§ Erste Veröffentlichung in den 80er Jahren (Royce).

§ Umsetzung des Life-Cycles.

§ (immer noch) stark verbreitet.

§ Einfache Anwendung.

§ Schwerpunkt auf Dokumentation.

Wasserfall Modell (2)

Vorteile

- § Backtracking zu früherem Entwicklungsphasen.
- § Risikominimierung durch “Abschluss” einer Phase.
- § Weite Verbreitung und hoher Bekanntheitsgrad.
- § Strikte Trennung der einzelnen Phasen.
- § Unterstützung von kleinen Entwicklungsteams.

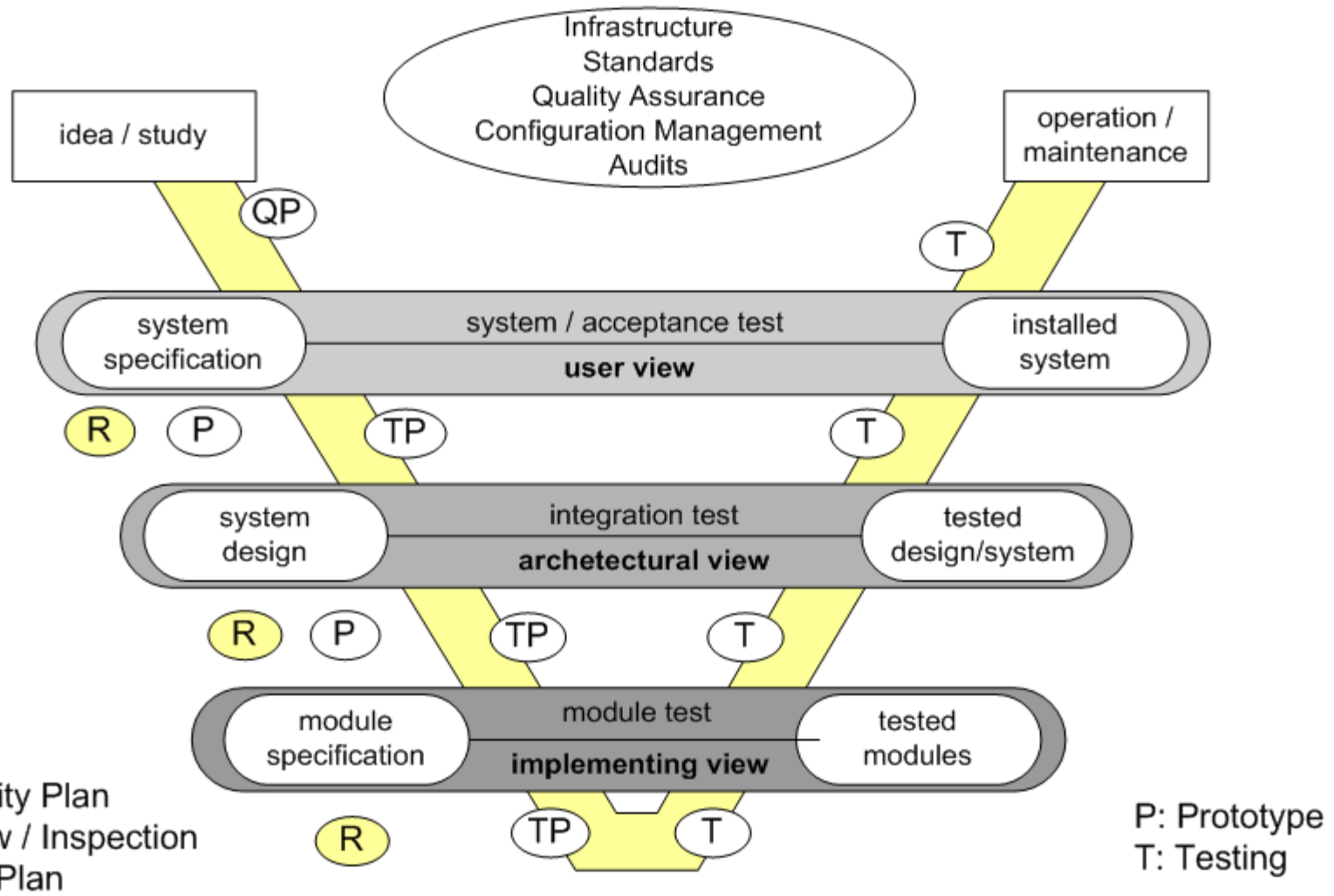
Nachteile

- § Alle Tasks einer Phase müssen abgeschlossen werden (keine parallele Entwicklung möglich).
- § Starke Auswirkung von Fehlern in frühen Phasen auf das Entwicklungsprojekt.

Anwendungsbereich

- § Gute Kenntnis der Anforderungsdomäne erforderlich (*No-Surprise Software*)
- § Klar definierte (und vollständige) Anforderungen erforderlich.

V-Modell Konzept mit QS-Methoden



V-Modell Konzept: Vor-/Nachteile

Vorteile

- § Spezifikationsphase vs. Realisierung und Testen.
- § Kontext von Produkten und Tests.
- § Verschiedene Abstraktionslevels (User, Architekten und Implementierungssicht).
- § Fehlerbehandlung in frühen Phasen des Softwareentwicklung (durch Einsatz von Reviews).
- § Basiskonzept für VM 97 und VM XT.

Nachteile

- § Klare Beschreibung der Systemanforderungen ist wichtig.
- § Hoher Dokumentationsaufwand.
- § Kritisch bei unklaren Anforderungen / sich ändernden Anforderungen.

Anwendungsbereich

- § Große Projekte im öffentlichen Bereich.
- § Klar definierten Anforderungen.

V-Modell XT

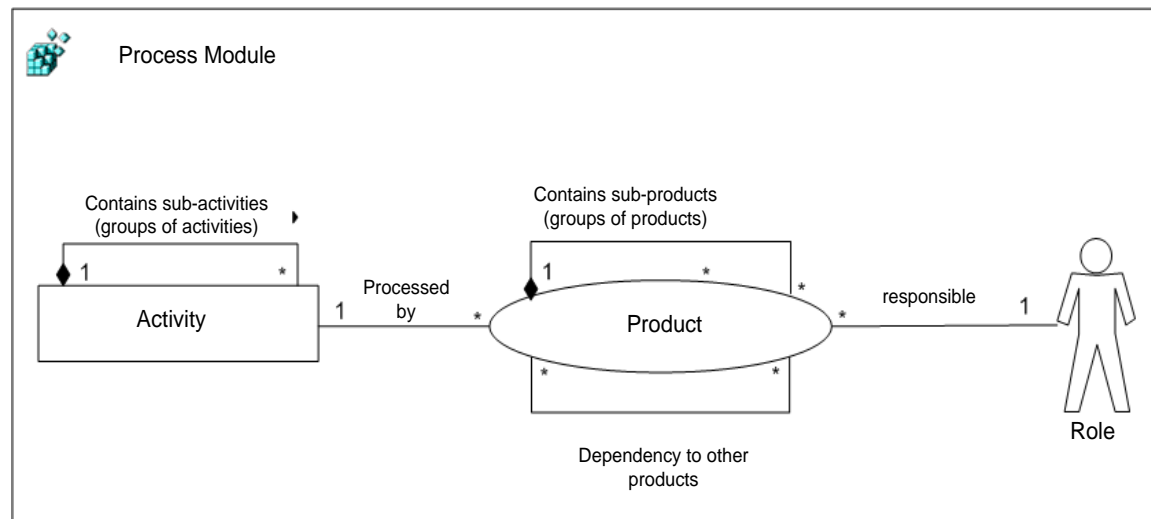
- § Das V-Modell XT ist eine Weiterentwicklung des V-Modell 97.
- § Veröffentlichung im Februar 2005.
- § Laufende Weiterentwicklung (derzeit Version 1.3)
- § **Verpflichtendes Vorgehensmodell für IT Projekte im öffentlichen Bereich in Deutschland.**



Zielsetzung der Entwicklung des V-Modell XT

- § **Verbesserung** der Unterstützung von Anpassbarkeit, Anwendbarkeit, Skalierbarkeit und Änder- und Erweiterbarkeit des V-Modells.
- § Berücksichtigung des neuesten Stand der Technik (**Best-Practice**).
- § **Kompatibilität zu formalen Richtlinien und Standards** (z.B. ISO 9000 Standard, CMMI).
- § **Erweiterung des Anwendungsbereiches** auf die Betrachtung des Systemlebenszyklus; Integration des **Auftraggebers** in das Projekt.
- § Integration eines Prozessmodells zur “Einführung und Pflege eines organisations-spezifischen Vorgehensmodells).

- § **Produkte** stehen im Mittelpunkt (=Projektergebnisse), für jedes Produkt gibt es **definierte Rollen** mit definierten Verantwortlichkeiten.
- § **Projektdurchführungsstrategien und Entscheidungspunkte** geben die Reihenfolge der Produktfertigstellung und somit den Projektverlauf vor.
- § Vorgehensbausteine sind die modularen Elemente des V-Modell XT.
 - kapselt **Rollen, Produkte und Aktivitäten**.
 - Kann als unabhängige Einheit **eingesetzt** werden.
 - Ist eine Einheit, die unabhängig **veränder- und aktualisierbar** ist.



Komponenten des V-Modell XT

- § Projekttypen vs. Projektgegenstand
- § Vorgehensbausteine kapseln Produkte, Aktivitäten und Rollen
 - § Verpflichtende Elemente (core elements)
 - § Optionale Elemente (um individuelle Projektanforderungen erfüllen zu können)
- § Unterstützung der Anpassbarkeit durch integrierte Tailoringmechanismen.
- § Integrierte Methoden- und Toolunterstützung zur
 - § Erstellung von Produkten durch
 - § Aktivitäten und
 - § Rollen (verantwortlich für ein Produkt).
- § Entscheidungspunkte (etwa Meilensteine) definieren einen Zeitpunkt, an dem eine Fortschrittsentscheidung getroffen wird.
- § Projektdurchführungsstrategien definieren die Reihenfolge der im Projekt zu erreichenden Projektfortschrittsstufen (Sequenz von Entscheidungspunkten)
- § Durch die Struktur des V-Modell XT ist eine Vergleichbarkeit zu herkömmlichen Prozessmodellen möglich
(z.B. Konventionsabbildungen zu Prozessmodellen und Standards)

Projekttypen

Projekttypen werden eingeteilt:

§ Nach **Projektgegenstand** (z.B. Hardwaresystem, Softwaresystem, Komplexes System)

§ **Projektrollen** (z.B. Auftraggeber / Auftragnehmerprojekte)

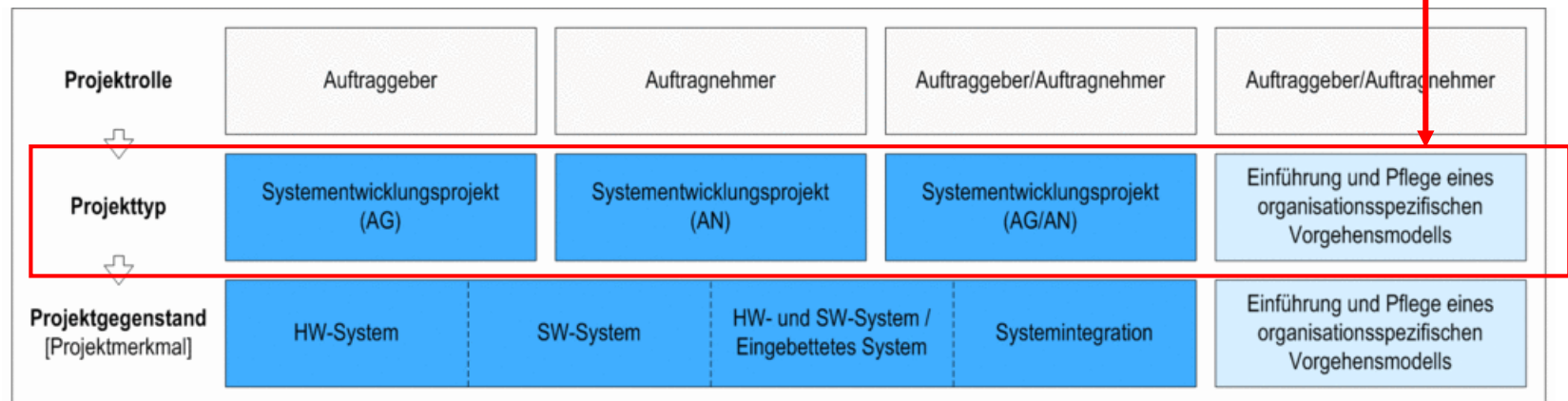
à daraus resultieren (derzeit) **4 Projekttypen**

§ Systementwicklungsprojekt des Auftraggebers.

§ Systementwicklungsprojekt des Auftragnehmers.

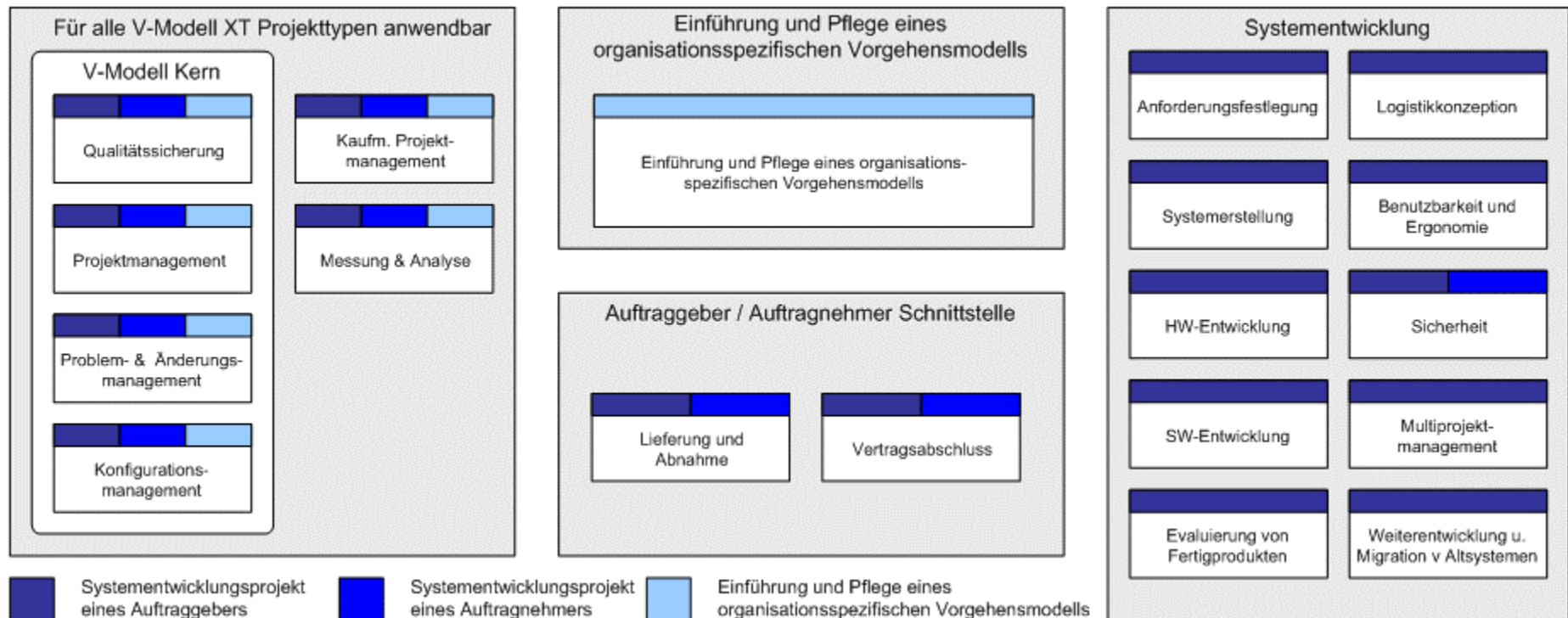
§ Einführung und Pflege eines organisationsspezifischen Vorgehensmodells

§ Systementwicklungsprojekt (Auftraggeber/Auftragnehmer);
z.B. in-house System Entwicklung (seit der Version 1.2 integriert).



Vorgehensbausteine im V-Modell XT

- § V-Modell Kern (verpflichtende Elemente für alle Projekttypen).
- § Einführung und Pflege eines organisationsspezifischen Vorgehensmodells.
- § Elemente für die Systementwicklung
- § Auftraggeber / Auftragnehmer – Schnittstelle.
- § Tool-Unterstützung durch den V-Modell Assistenten.

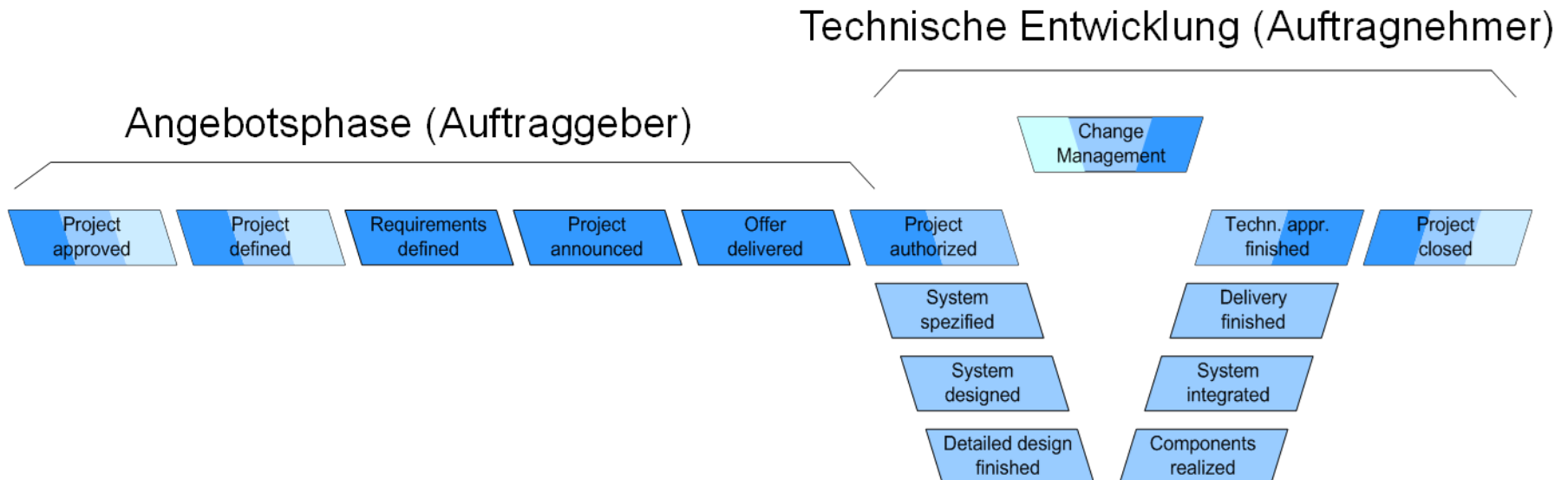


Projektdurchführung

§ Entscheidungspunkte und Projektdurchführungsstrategie.

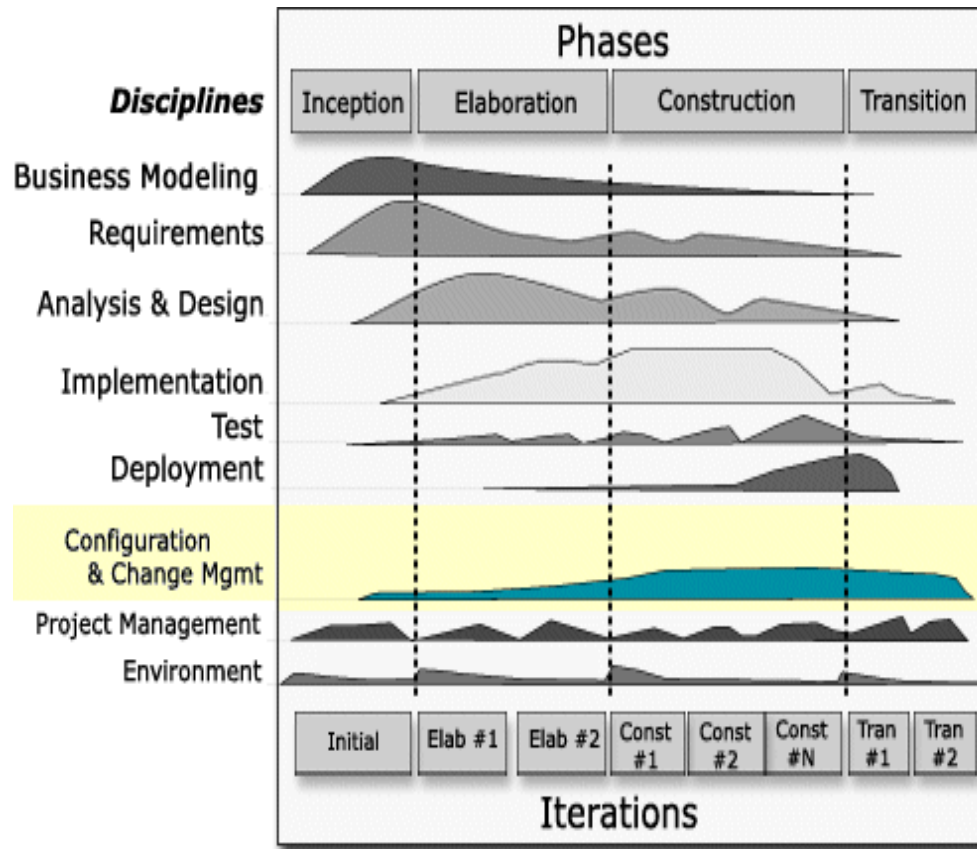
- Definitionen von **Entscheidungspunkten** (vergleichbar mit Meilensteinen)
- An Entscheidungspunkten müssen **definierte Produkte** vorliegen.
- Eine **Projektdurchführungsstrategie** ist eine definierte Abfolge von **Entscheidungspunkten** (z.B. inkrementelle oder agile Entwicklungsstrategie).

§ Beispiel



- § Flexible Anwendung des V-Modell XT durch Anpassung des Modells an unterschiedliche Projektgegebenheiten (Projekttyp, Projektmerkmale).
 - Auswahl von benötigten Vorgehensbausteinen.
 - Definition der passenden Projektdurchführungsstrategie.
- § Werkzeugunterstützung (Open Source)
 - V-Modell XT Projektassistent zur Anpassung des Modells an ein konkretes Projekt.
 - Ergebnis ist eine angepasste Vorgehensweise und angepasste Templates für die Projektdokumentation.
 - V-Modell XT Editor ermöglicht freie Konfigurationen des Vorgehensmodells, z.B. Anpassung auf ein Unternehmensmodell (Standard).
- § Verpflichtendes Vorgehensmodell für öffentliche IT Projekte in Deutschland.
- § Konventionsabbildungen ermöglichen die Kompatibilität zu Qualitätsmanagementstandards, wie CMMI und ISO 9000 sowie zu anderen Vorgehensmodellen, wie dem Rational Unified Process.

Rational Unified Process, RUP (1)



§ Inkrementelle und iterative Vorgehensweise.

4 grundlegende Phasen:

- § Inception (Beginn)
- § Elaboration (Concept & Design)
- § Construction
- § Transition (Auslieferung)

Definierte Workflows und Disziplinen:

- § 6 Engineering Workflows
- § 3 Supporting Workflows
- § Mehrere Iterationen innerhalb einer Phase.

Rational Unified Process, RUP (2)

- § Iterative und inkrementeller Workflow.
- § Integriertes Anforderungsmanagement.
- § Komponenten-orientierte Architektur.
- § Modellierung durch das UML Methodenframework.
- § Produkt-Verifikation an Meilensteinen.
- § Änderungsmanagement (*supporting discipline*).

Vorteile

- § Real-world Szenarien.
- § Werkzeugunterstützung (Rational XDE, IBM).
- § Vordefinierte Liste mit erforderlichen Artefakten.

Nachteile

- § Hohe Komplexität.
- § Hoher Dokumentationsaufwand.
- § Anbieterabhängigkeit?

Anwendungsbereich:

- § Grosse Projekte durch eine ganzheitliche Prozess-Sicht auf das gesamte Projekt (inkl. Deployment).

Table of Contents

- § Software Life-Cycle (Wiederholung)
 - Phasen im Software Life-Cycle
 - Vom Software Life-Cycle zum Software Prozess
- § Traditionelle Ansätze
 - Wasserfall Modell
 - V-Modell Grundkonzept
 - V-Modell XT
 - Rational Unified Prozess
- § Agile Ansätze
 - Agiles Manifest
 - Scrum
- § Anpassung von Software Prozessen (Prozess Tailoring)

- § Aus den Kritikpunkten der systematischen und „schwergewichtigen“ Ansätzen entwickelte sich das Agile Manifest.
- § Festgeschrieben durch 17 Softwareentwickler 2001.

„Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan“

- § Das Manifest beinhaltet **12 Prinzipien**, die die Basis für agile Software Entwicklung darstellen.
- § Agile bedeutet aber nicht „unkontrolliert“ – auch hier existieren **Prozesse und Regeln**, die eingehalten werden müssen.
- § Bekannte Vertreter: eXtreme Programming oder Scrum.

12 Agile Principles

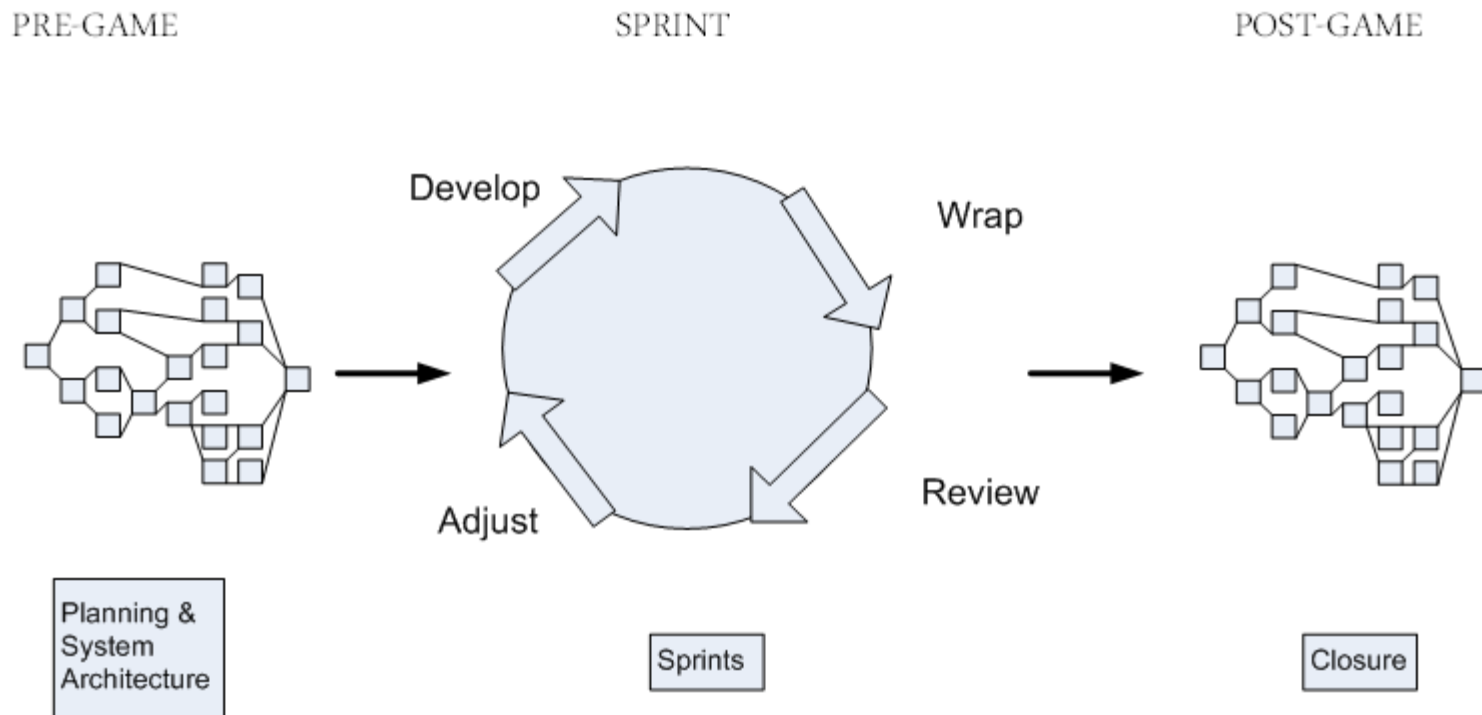
1. Our highest priority is to **satisfy the customer** through **early and continuous delivery of valuable software**.
2. **Welcome changing requirements**, even late in development. Agile processes harness change for the customer's competitive advantage.
3. **Deliver working software frequently**, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. **Business people and developers must work together daily** throughout the project.
5. **Build projects around motivated individuals**. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is **face-to-face conversation**.
7. **Working software is the primary measure of progress**.
8. **Agile processes promote sustainable development**. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to **technical excellence and good design** enhances agility.
10. **Simplicity**--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from **self-organizing teams**.
12. At regular intervals, **the team reflects on how to become more effective**, then tunes and adjusts its behavior accordingly.

- § SCRUM ist keine Abkürzung; der Begriff stammt aus der Rugby Start-Formation.
- § Analogie zu Software Engineering ????
- § Was leistet Scrum?
- Agiler Software Prozess aus Sicht des Projektmanagements (PM).
 - **Kleine** aber **hoch-effiziente Teams** (auch mehrere Teams möglich).
 - **Flexibles Prozessmodell**, um auf ändernde Anforderungen im Projektablauf reagieren zu können.
 - **(Teil-)Produkte** stehen dem Kunden frühzeitig zur Verfügung.
 - Das Projekt wird bestimmt durch **Zeit, Wettbewerb, Kosten, und Funktionalität**.
 - Deliverables werden beeinflusst von **Marktinformationen, Kundenkontakt und Skills der Entwickler**.
 - **Hoher Bekanntheitsgrad** in den letzten Jahren.
- § Hoher Erfüllungsgrad der agilen Prinzipien.

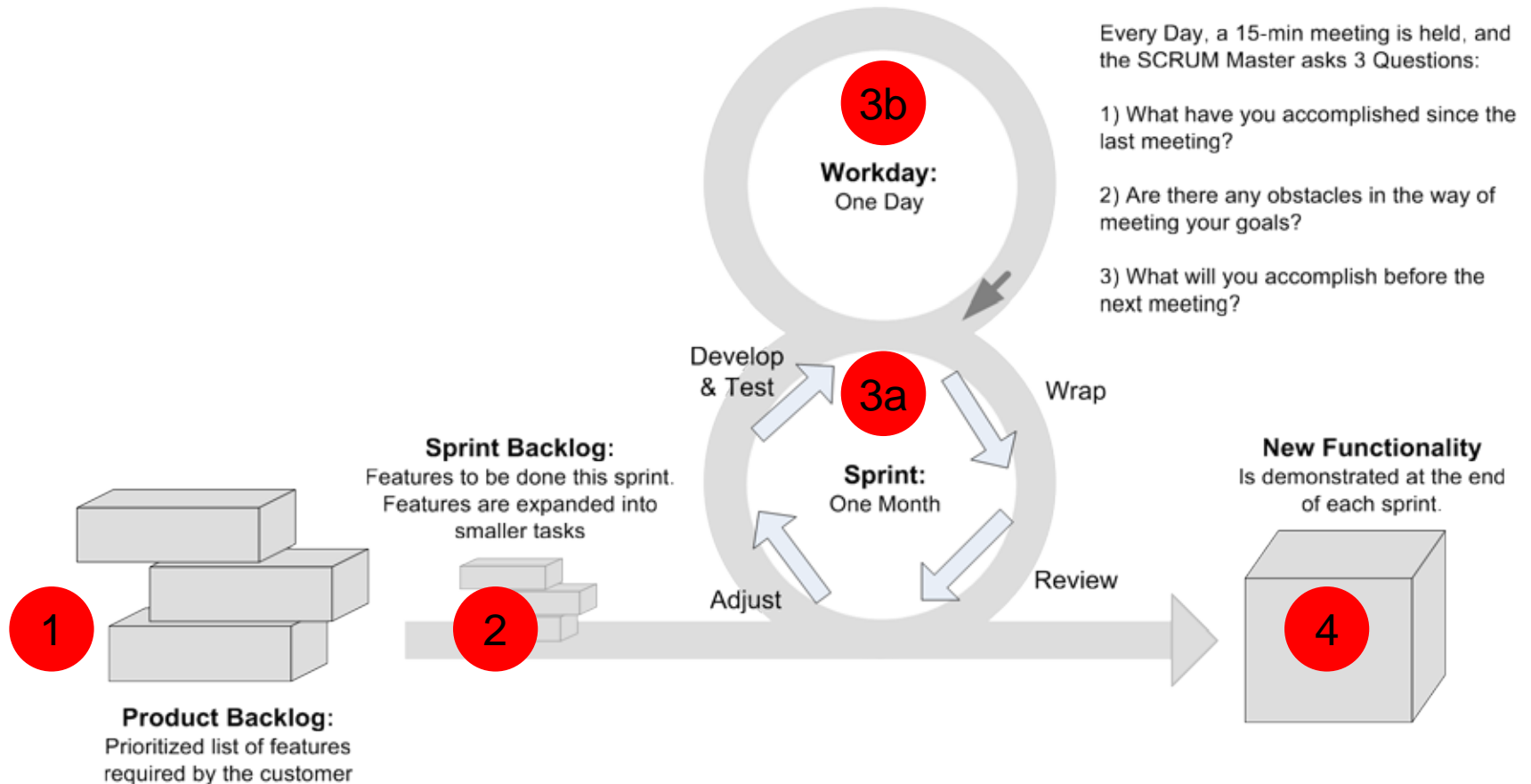


Der SCRUM Prozess

- § Scrum besteht aus einer Sammlung von **Prozeduren**, **Rollen** und **Methoden** für Projektmanagement.
- § **Selbst-Organisierende Teams**.
- § **Aufbau:**



Sprints



SCRUM Charakteristika und Begriffe

Charakteristika

- § Ein Team erstellt eine Einheit, ein Produkt oder Teilprodukt.
- § Klare Arbeitsaufteilung innerhalb des Teams.
- § Klar priorisierte Projektergebnisse (*Backlog Items*).
- § Ein gemeinsames Ziel (Erstellung des Produktes).
- § Der “Sprint” ist das zentrale Element.
- § Das Sprint-Team kann ungestört arbeiten; es sind keine Eingriffe „von aussen“ (z.B. durch den Kunden) zulässig.
- § Temporal structure = daily Scrum Meeting + Review + Retrospective.

Begriffe:

- § Backlog beinhaltet alle Arbeitspakete, die in “nächster” Zeit umgesetzt werden müssen (sowohl klar definierte als auch vage Anforderungen): Produkt vs. Sprint Backlog.
- § Ein Daily-Scrum ist eine tägliche Projektbesprechung um etwaige Fragen / Missverständnisse zu klären; Definition des täglichen Arbeitsauftrags.
- § Scrum Team: Funktionsübergreifendes Team, zuständig für den Sprint Backlog.
- § Burndown Chart: Visuelle Darstellung des Projektfortschritts.

Table of Contents

- § Software Life-Cycle (Wiederholung)
 - Phasen im Software Life-Cycle
 - Vom Software Life-Cycle zum Software Prozess

- § Traditionelle Ansätze
 - Wasserfall Modell
 - V-Modell Grundkonzept
 - V-Modell XT
 - Rational Unified Prozess

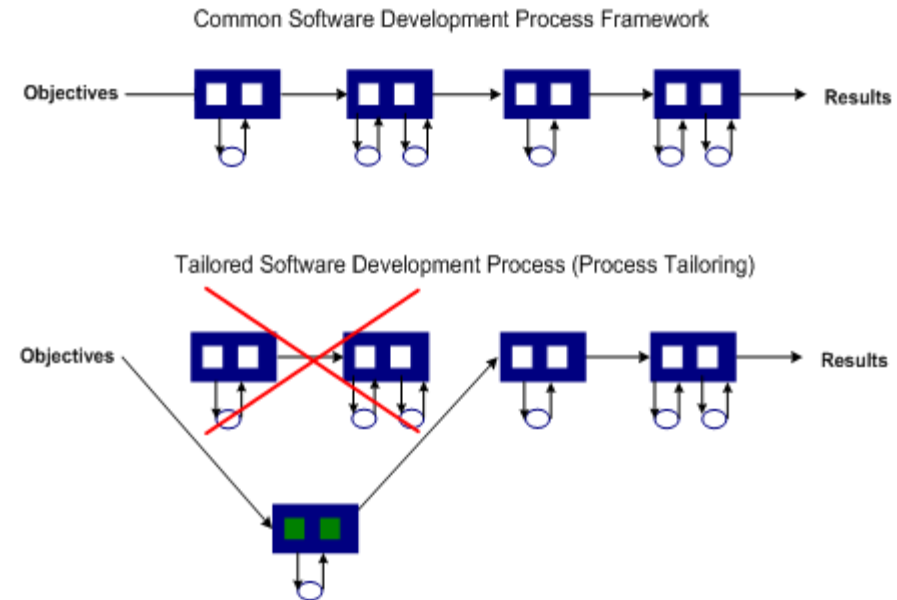
- § Agile Ansätze
 - Agiles Manifest
 - Scrum

- § Anpassung von Software Prozessen (Prozess Tailoring)

- § Anwendbarkeit von standardisierten Softwareprozessen im konkreten Projekt?
- § Standardisierte Software Prozesse sind in der Praxis **direkt eher schwer einsetzbar**, da eine Vielzahl an Projektattributen berücksichtigt werden müssen;
Beispiele: Projektgröße, Projekttyp und Anwendungsdomäne.
- § **Anpassungen an aktuelle Projektgegebenheiten** sind notwendig.
- § Diese Anpassungen von Vorgehensmodellen erfordern **erfahrene Projektleiter** und/oder effiziente Toolunterstützung (z.B. V-Modell XT Projektassistent).
- § **Prozess Tailoring** (bezogen auf der jeweilige Projekt)
 - Anpassung an individuelle Projektgegebenheiten.
- § **Prozess Customization (Standardisierung)**
 - Anpassungen an Unternehmensstandards (z.B. Siemens stdSEM)
 - Domänenabhängige Anpassungen, z.B. für Produktionsautomatisierung

Anpassung von Vorgehensmodellen an Projekte: Prozesstailoring

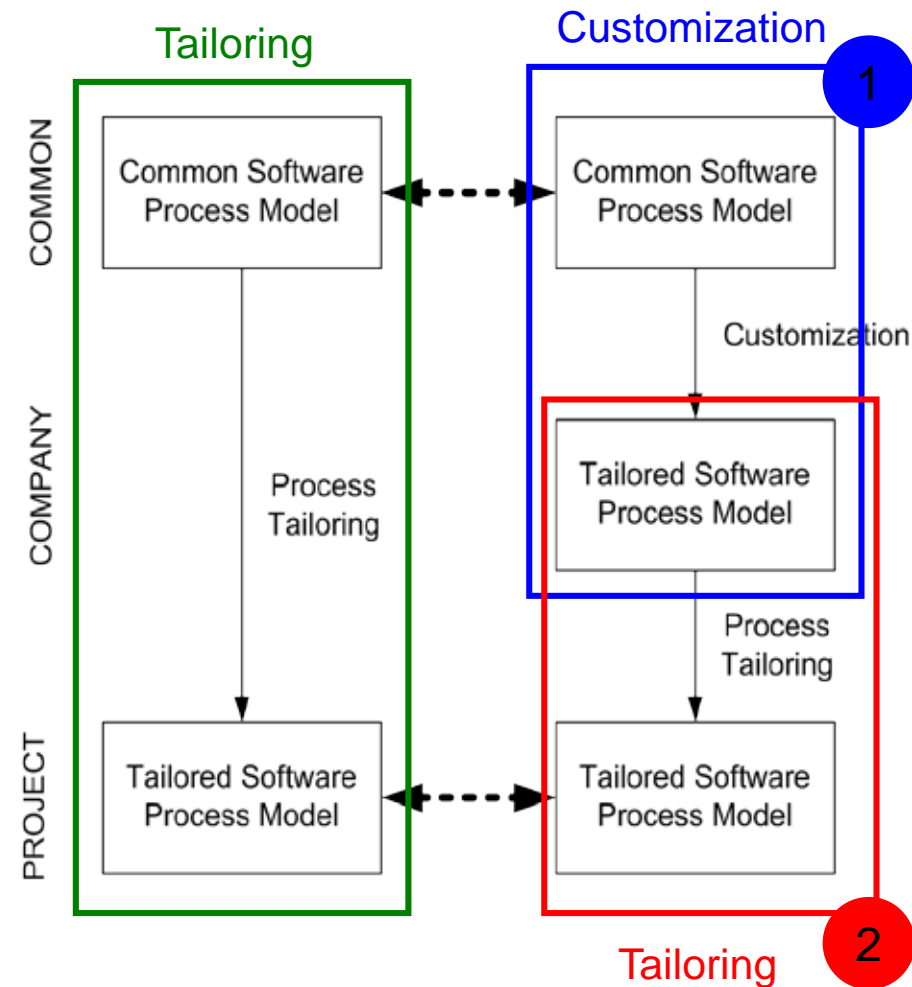
- § Anpassbarkeit eines generischen Entwicklungsprozesses an **spezifische Projektgegebenheiten** durch Prozesstailoring.
- § Ersetzen einzelner Prozess-Schritte (oder Vorgehensbausteine) durch passende alternative Lösungen.
- § Wiederverwendung von Best-Practices (Methoden / Tools).
- § Individuelle Anpassung des Projektplans.



- § Achtung:
 - Tailoring erfordert erfahrene Projektleiter, da ein fundiertes Verständnis des Modellaufbaus erforderlich ist.
 - Berücksichtigung von **definierten Tailoring-Kriterien und Produktabhängigkeiten**.

Prozess „Customization“

- § Verallgemeinerte Form des „Tailorings“.
- § Anpassung des Vorgehensmodells an **unternehmensspezifische** Gegebenheiten.
- § Effizienzsteigerung von Tailoring für ähnliche Aufgaben durch **Customization**:
 - **Unternehmensstandards**: Anpassung an das Unternehmen.
 - **Projektstandards**: Anpassung an ähnliche Projekte (z.B. Webapplikationen).
- § Diese **„angepassten Prozesse“** dienen als Grundlage für projektspezifisches Tailoring.
- § Achtung: Die **Kompatibilität** zum zugrunde liegenden Prozess muss sichergestellt werden!



- § Software Prozesse und Vorgehensmodelle ermöglichen vorhersagbare und nachvollziehbare Softwarelösungen.
- § Sie definieren, wie ein Softwareprojekt durchgeführt wird bzw. in welcher Reihenfolge die einzelnen Phasen ablaufen.
- § In der Praxis existiert eine Vielzahl an Prozessen für unterschiedliche Anwendungsbereiche (z.B. für spezifische Domänen, Projekttypen).
- § **Systematische Prozesse** sind durch ihre Struktur *plan-driven* und orientieren sich eher an Abläufen mit Schwerpunkt auf Produkten und Dokumentation.
Beispiele: Wasserfallmodell, V-Modell (XT), RUP, Inkrementelle Modelle.
- § **Agile Ansätze** rücken den Kunden und seine konkreten (sich ändernden) Anforderungen in den Vordergrund.
Beispiele: eXtreme Programming, SCRUM.
- § Durch **Tailoring** wird ein allgemeines Modell auf ein **individuelles Projekt** angepasst.
- § **Customization** ermöglicht die Erstellung **unternehmensspezifischer Vorgehensmodelle** für gleichartige Projekte / Produktgruppen.

- § Beck K.: „Extreme Programming. Das Manifest“, Addison Wesley, 2004.
- § Biffl Stefan, Winkler Dietmar, Frast Denis: „Qualitätssicherung, Qualitätsmanagement und Testen in der Softwareentwicklung“, Skriptum zur Lehrveranstaltung, 2004.
<http://qse.ifs.tuwien.ac.at/courses/skriptum/script.htm>
- § Höhn R., Höppner S.: „Das V-Modell XT. Grundlagen, Methodik und Anwendungen“, Springer, eXamen Press, 2008.
- § Kruchten P.: „The Rational Unified Process: An Introduction“, Addison-Wesley Longman, 2004.
- § Schatten A., Biffl S., Demolsky M., Gostischa-Franta E., Östreicher T., Winkler W.: „Best Practice Software Engineering. Eine praxiserprobte Zusammenstellung von komponentenorientierten Konzepten, Methoden und Werkzeugen“, Spektrum Akademischer Verlag, 2010, 978-3827424860.
- § Schwaber K., Irlbeck T.: „Agiles Projektmanagement mit Scrum“, Microsoft Press, 2007.
- § SCRUM, www.controlchaos.com, February 2006.
- § Software Engineering – Best practices:
<http://best-practice-software-engineering.blogspot.com/>
- § V-Modell XT: <http://www.v-model-xt.de>.