Teil V: Projektmanagement
50 Projektplanung

Prof. Dr. rer. nat. Uwe Aßmann
Institut für Software- und Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
TU Dresden
Version 13-1.0, 13.07.13

50.1 Projektmanagement

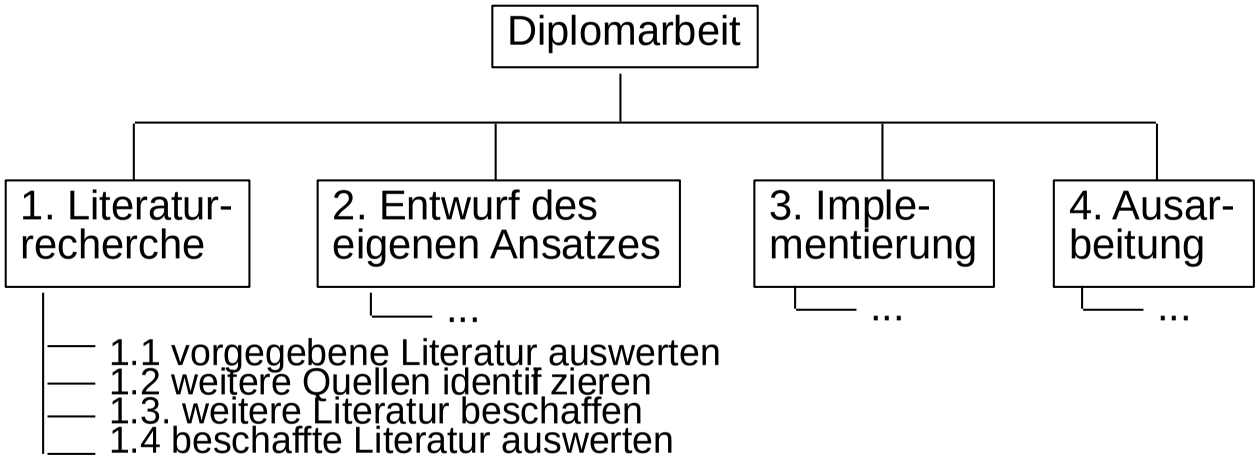
Das Glück des Lebens besteht nicht darin, wenig oder keine Schwierigkeiten zu haben, sondern sie alle siegreich und glorreich zu überwinden.

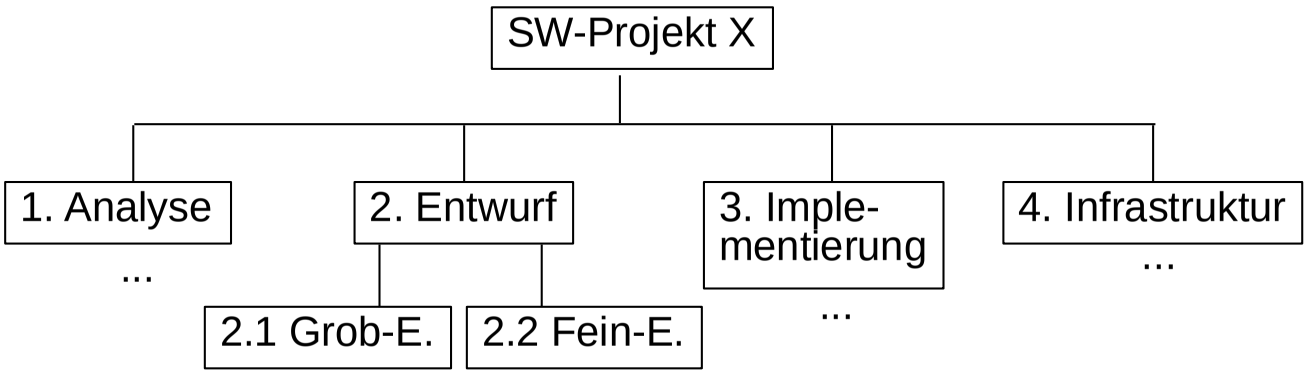
Carl Hilty, 28.02.1833 - 12.10.1909

Schweizer Richter und Staatsrechtler, Buchautor und christl. Staatsrechts-Philosoph

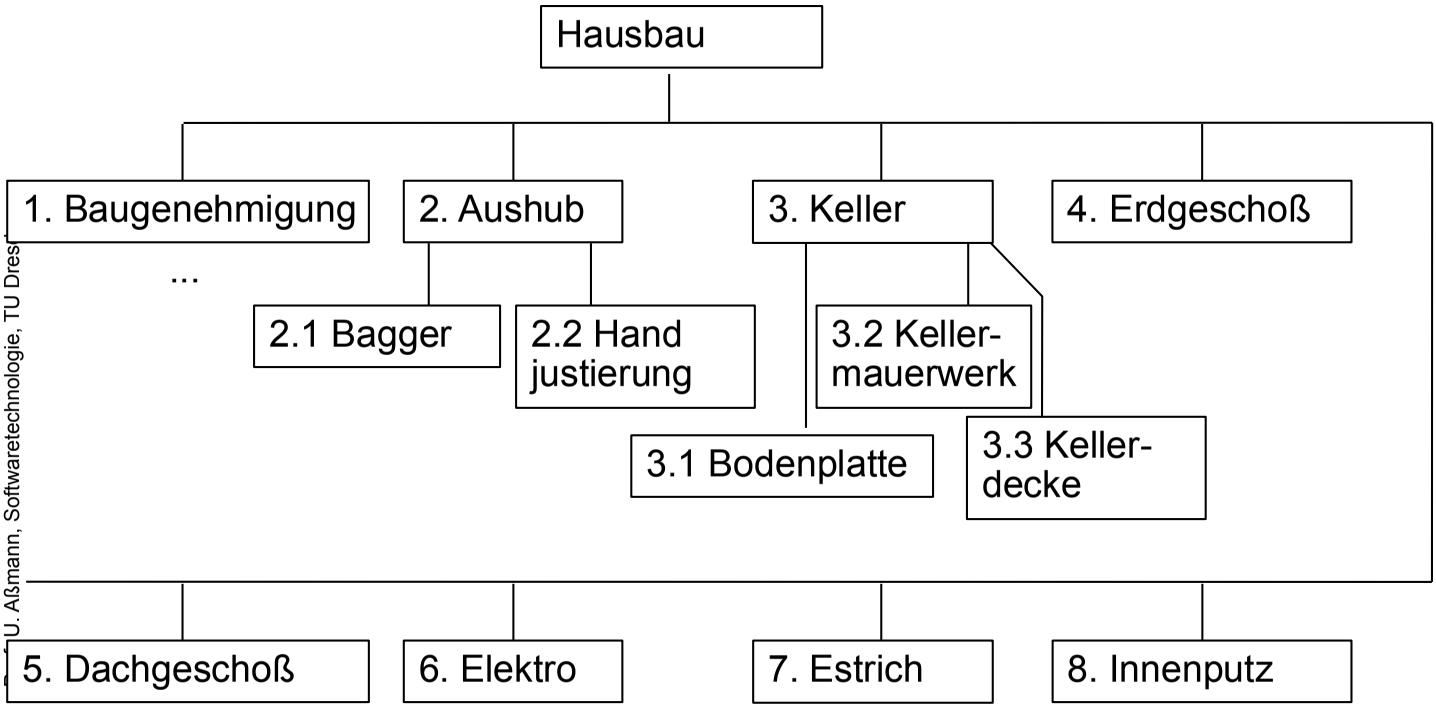
Seine Bücher beeinflussten auch K. Adenauer

Projektstruktur: Beispiele





Hausbau

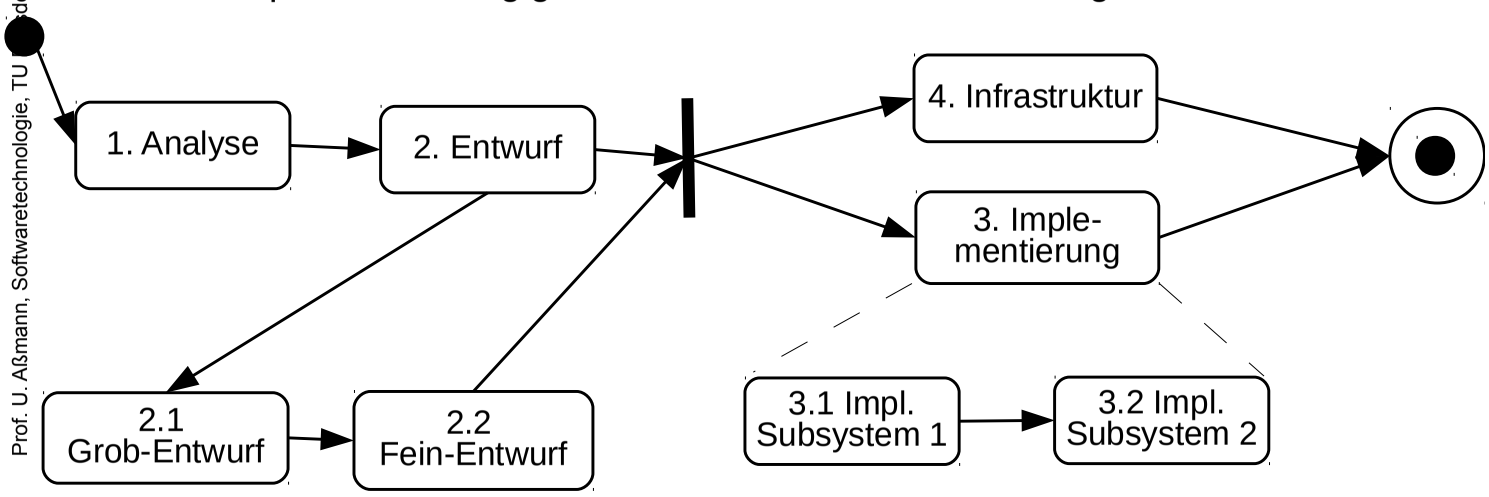


Aufwandsschätzung

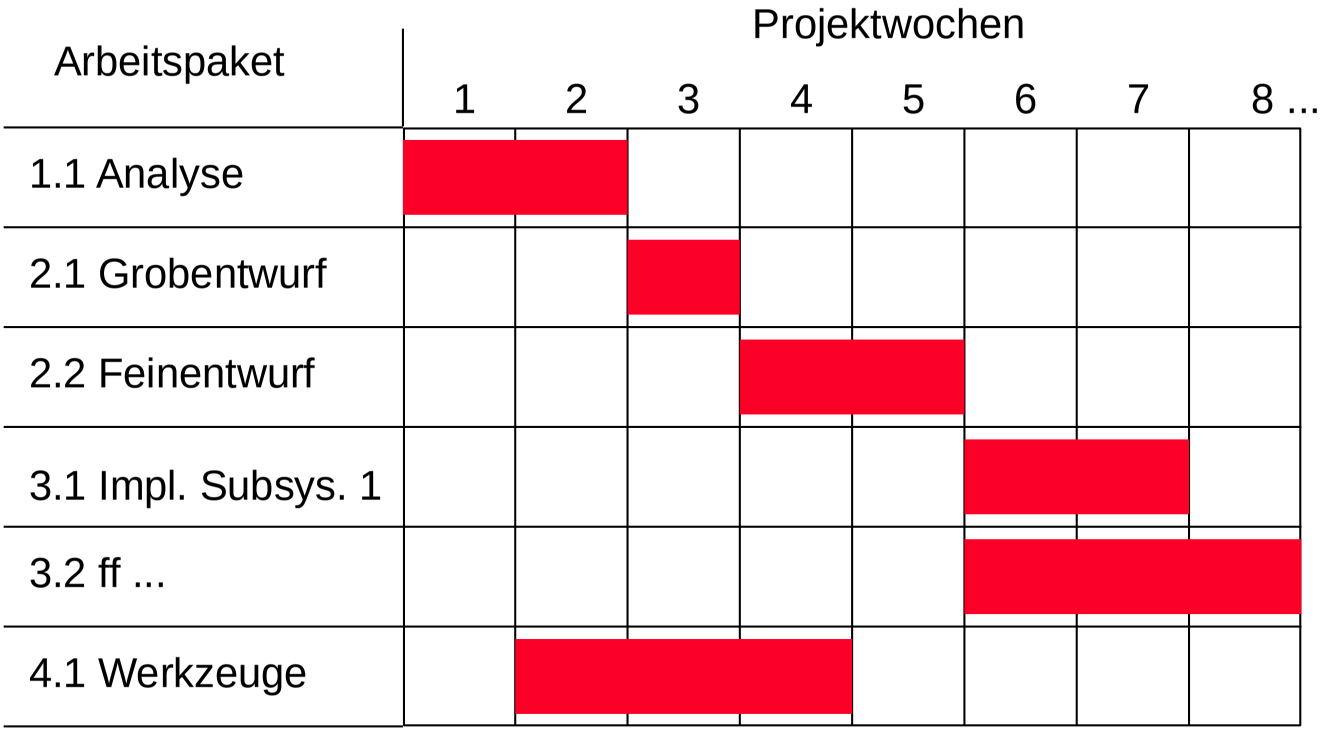
- Schätzungen für:
  - relativen Aufwand der Teilaufgaben
  - absoluten Aufwand für Subsysteme
- Faustregeln, Erfahrungswerte
- Techniken der Aufwandsschätzung:
  - Befragung von Entwicklern
  - Klassifikation z.B. durch "Function Point"-Methode
    - Wie viele Teilfunktionen?
    - Wie schwierig ist jede Teilfunktion?
  - Metriken für Spezifikationen
  - "Kalibrierung" durch eigene Erfahrungswerte
- Mehr in Vorlesung „Softwaremanagement“, SS

Abhängigkeiten

- Welche Aktivitäten hängen von Ergebnissen anderer Aktivitäten ab? (Abhängigkeitsgraph)
- Aufwandsschätzung + feste Termine + Abhängigkeiten:
  - Netzplantechniken (z.B. PERT)
  - GANTT-Diagramm
- Beispiel für Abhängigkeiten, erfaßbar in Aktivitätendiagramm:

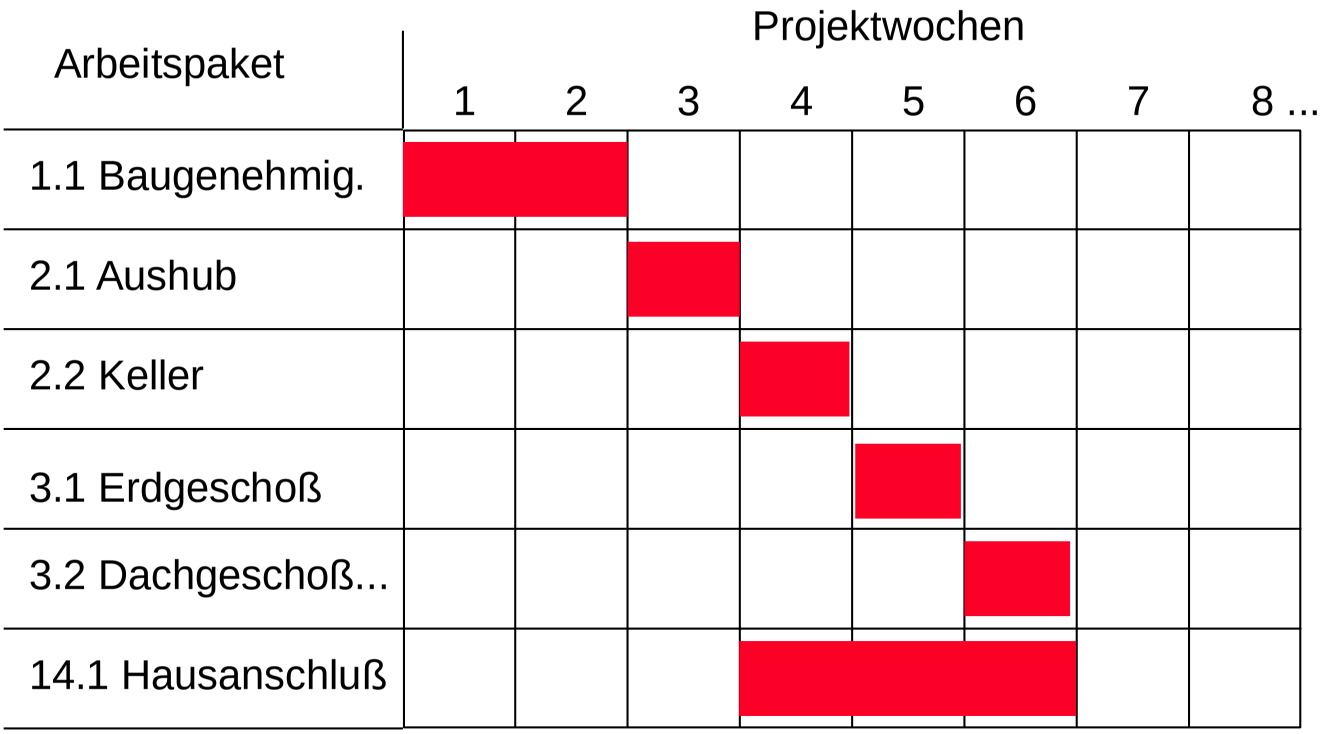


Zeitplanung: Gantt-Diagramm, eine Aktivitätentabelle



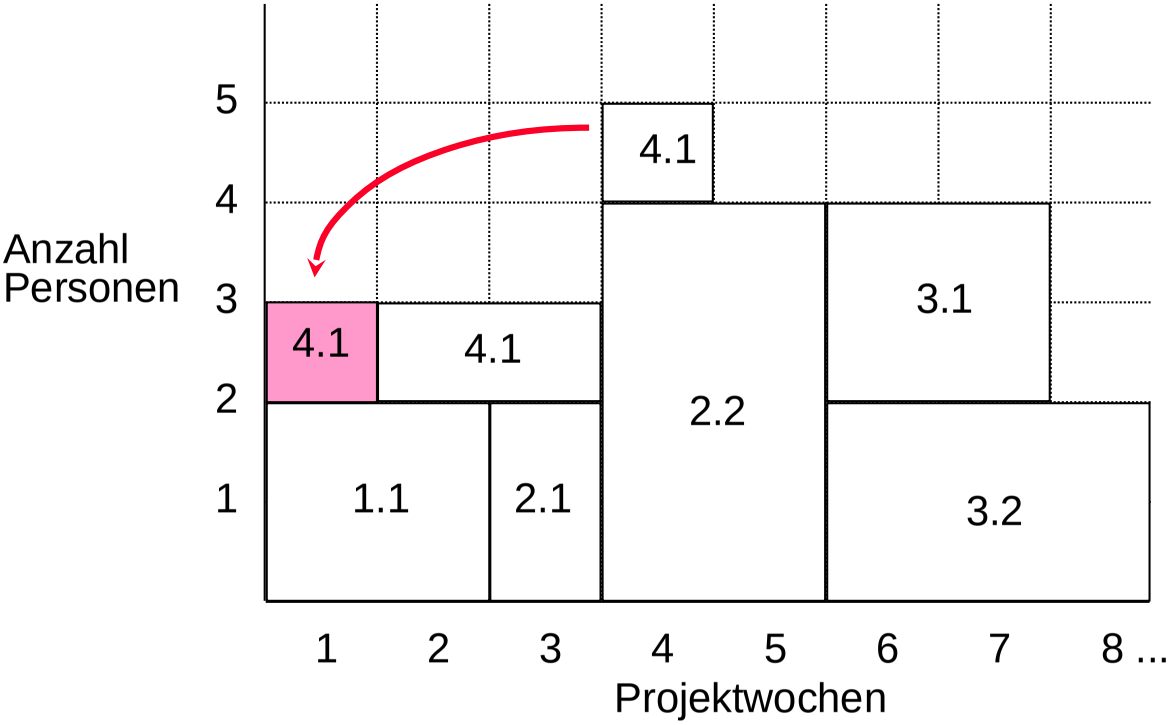
Identifikation *kritischer* und *unkritischer* (4.1, 3.1) Arbeitspakete
(kritisch = Verlängerung verlängert Gesamtprojektdauer)

Zeitplanung Hausbau: Gantt-Diagramm



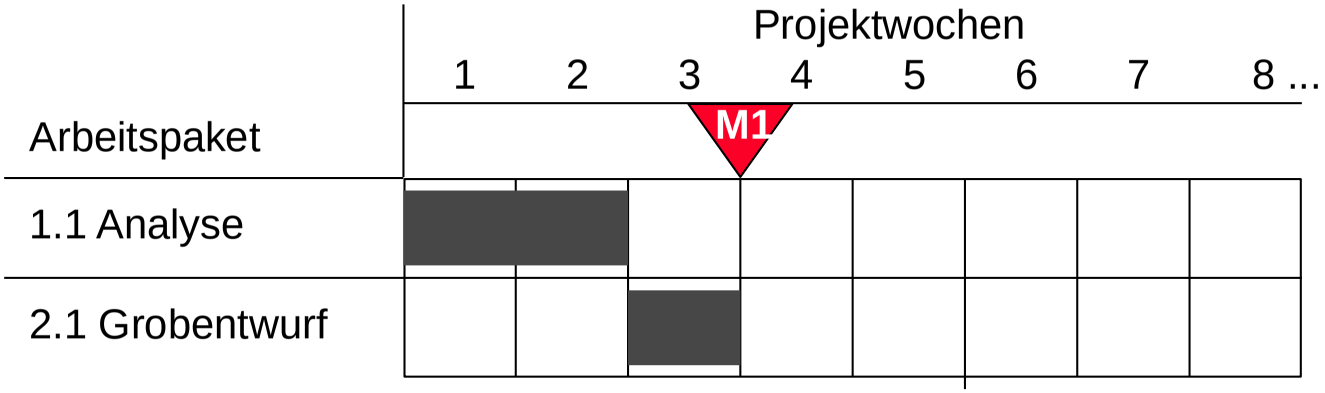
Ressourcenplanung

- Umplanung mit dem Ziel: Anpassung an vorhandene Ressourcen
- *Packen* in Flächen über Anz. Personen und Projektwochen



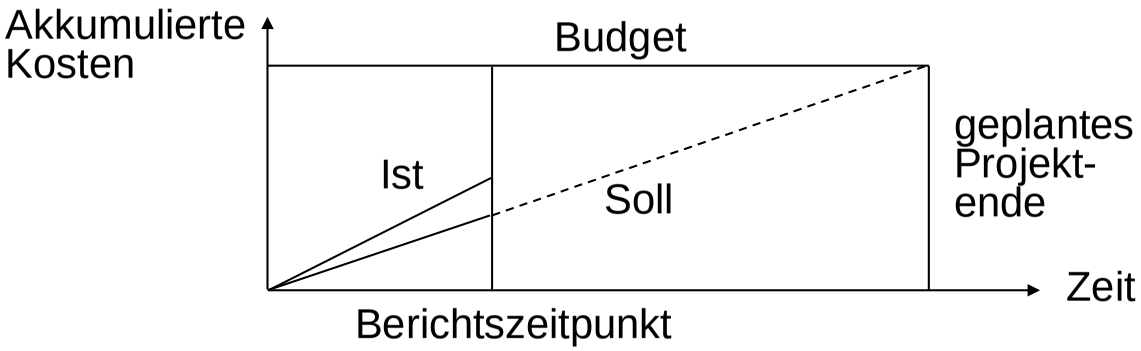
Meilensteine

- Ein *Meilenstein* ist ein klar definiertes Zwischenresultat, an Hand dessen der Projektfortschritt beurteilt werden kann.
- Beispiele:
  - "Anforderungsspezifikation zusammen mit Auftraggeber verabschiedet"
  - "Erster Prototyp lauffähig"
  - Schlechtes Beispiel: "Code zu 50% fertig"
- Meilensteine im Gantt-Diagramm:



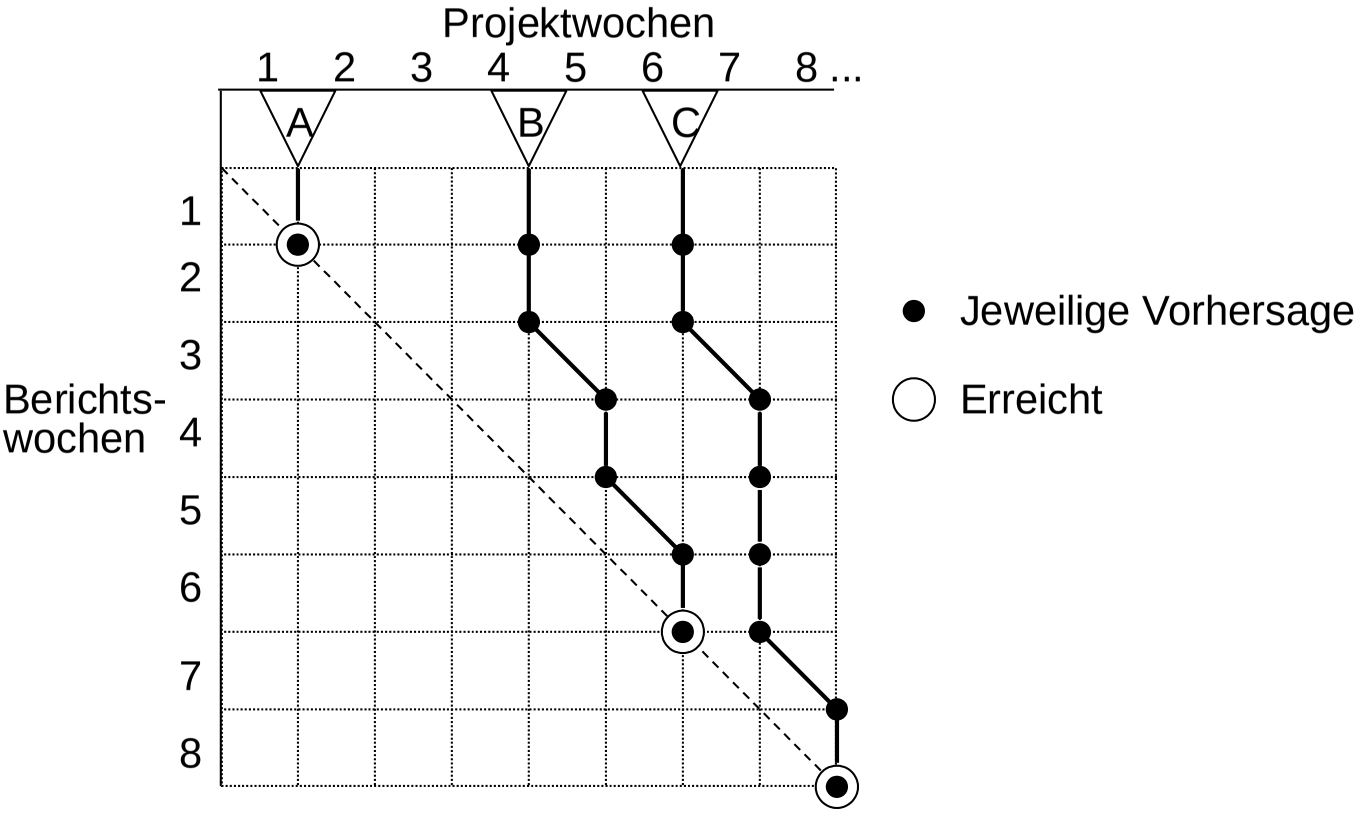
Projektverfolgung

- Das Projektmanagement muß ein "Frühwarnsystem" für eventuelle Probleme betreiben (Projektverfolgung).
- Informationsquellen:
  - Laufende (z.B. wöchentliche) Management-Berichte
  - Arbeitszeit-Kontierung
  - Resultate *(deliverables)*
- Rückkopplung zum Projektteam
  - Regelmäßige Projektbesprechungen
  - Beispiel: Akkumulierter Ressourcenverbrauch



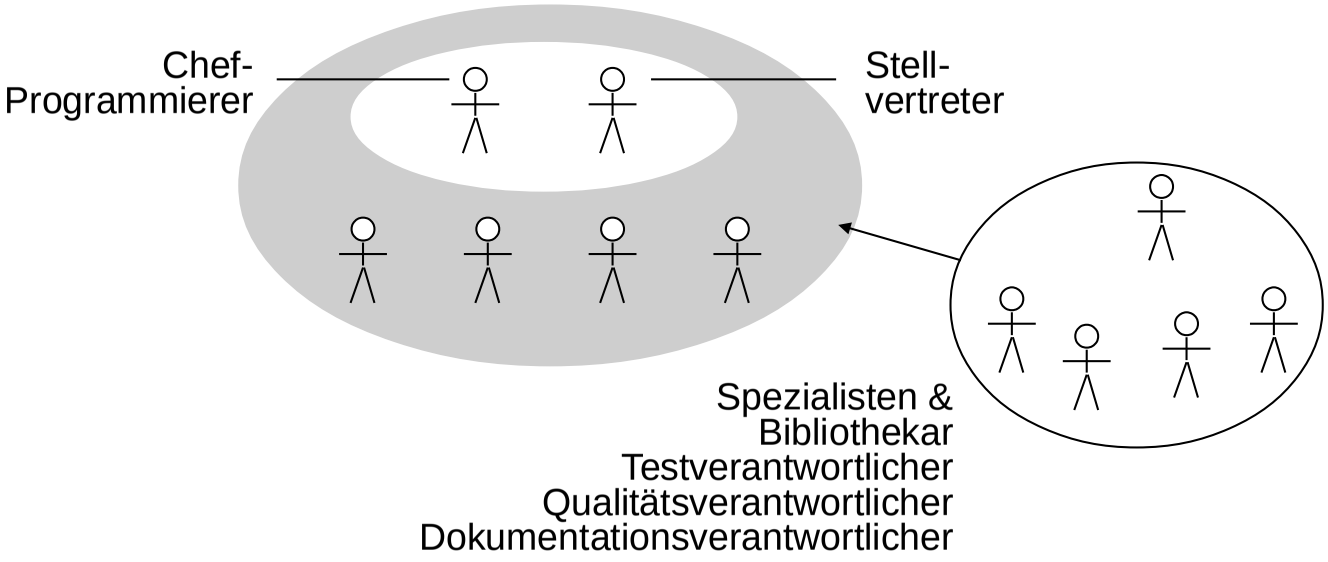
Meilenstein-Trendanalyse

- Anhand jedes Managementberichts sagt das Management die Meilensteine neu voraus



Teamzusammenstellung (Staffing)

- Regeln für Teamproduktivität:
  - Optimale Teamgröße: ca. 5-7 Personen
  - Gemischte Qualifikationen
  - Team von externer Kommunikation entlastet
  - Große Projekte aus vielen Teams zusammengesetzt
- Harlan Mills / Baker 1972: *Chefprogrammierer-Struktur*



Organisation von Sitzungen

- Vor Sitzungen sollte man immer folgendes (schriftlich) fixieren:
- Ziele
  - Zweck des Treffens (was wollen wir erreichen?)
  - Erfolgskriterien des Treffens (wie können wir kontrollieren, dass wir das Ziel erreicht haben?)
- Agenda
  - Welche Teilnehmer? Haben diese versteckte Zielkonflikte?
  - Zeitplanung: Wie lange welcher Punkt?
- Verantwortlicher für ein Ergebnisprotokoll

Typische Gliederung eines Ergebnisprotokolls

- Name der Sitzung
- Teilnehmer, Moderator, Ort, Zeit
- Tagesordnung
  - Standard-Tagesordnungspunkte:
    - Protokollkontrolle
    - Bericht über den erreichten Stand
    - Einzelaufgaben
    - Nächster Termin
- Ergebnisse
  - gegliedert nach Tagesordnungspunkten (TOPs)

Einzelaufgaben *(Action Items)*

- Einzelaufgabe *(action item, action point)* besteht aus:
  - Lfd. Nr., Verantwortliche Person
  - Kurztitel
  - Beschreibung
  - Ursprung (Sitzung, auf der Aufgabe definiert wurde)
  - Termin
  - Status (offen, verlängert, erledigt)
- Liste der Einzelaufgaben wird bei **jedem** Treffen durchgegangen und aktualisiert:
  - Welche Aufgaben sind fällig?
  - Was ist das Ergebnis?
  - Was ist weiter zu tun?
    - Termin verlängern
    - Neue Aufgaben definieren
- Können in einem *issue tracker* verwaltet werden (z.B. Mantis)

50.2 Vorgehensmodelle (Phasenmodelle)

17

Thus it will be seen that engineering is a distinctive and important profession. To some even it is the topmost of all professions. However true that may or may not be to-day, certain it is that some day it will be true, for the reason that engineers serve humanity at every practical turn. *Engineers make life easier to live--easier in the living*; their work is strictly constructive, sharply exact; the results positive. Not a profession outside of the engineering profession but that has its moments of wabbling and indecision--of faltering on the part of practitioners between the true and the untrue. Engineering knows no such weakness. *Two and two make four. Engineers know that*. Knowing it, and knowing also the unnumbered possible manifoldings of this fundamental truism, engineers can, and do, approach a problem with a certainty of conviction and a confidence in the powers of their working-tools nowhere permitted men outside the profession.

Charles M. Horton. Opportunities in Engineering. 1920, by Harper & Brothers

http://www.gutenberg.org/ebooks/24681

Obligatorisches Lesen

18

- Zuser Kap. 1-3 *oder*
- Ghezzi Chapter 1 *oder*
- Pfleeger Chapter 1; Chap 8.1

Vorgehensmodell *(engl. process model)*

- Strukturiertes Modell zum Erstellen von Software

Phasenmodell

- Vorgehensmodell, das den Herstellungsprozesses in def nierte und abgegrenzte Phasen einteilt
- Vorgabe einer Reihenfolge in der Bearbeitung der Phasen

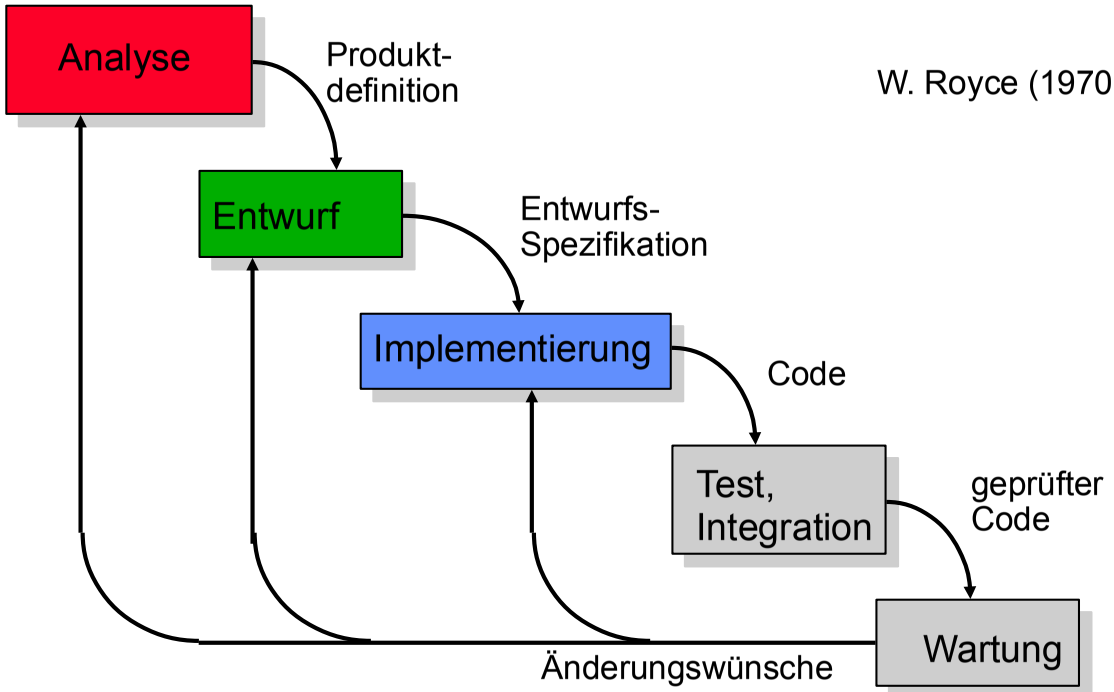
Wie gehe ich vor, um Software zu entwickeln?

- Ad hoc
- Es lief schon oft schief...
  - Denver International Airport, Krise 1993-95
  - Hamburger Güterbahnhof 1995
  - Toll Collect 2004
  - Fiscus 2004
- Gibt es nicht irgendwelche Hilfen, strukturiert vorzugehen?

Vorgehen nach einem “Phasenmodell”

- ***Phasenmodell*** (*process model, software development life cycle*)
  - Einteilung des Herstellungsprozesses für ein (Software-) Produkt in definierte und abgegrenzte Abschnitte, abgegrenzt durch ***Meilensteine***
    - Grobgliederung: Phasen (*phases*)
    - Feingliederung: Schritte (*stages*, *steps*)
  - Vorgabe einer Reihenfolge in der Bearbeitung der Phasen
  - Richtlinie für die Definition von Zwischenergebnissen
    - Detailliertes Phasenmodell + Zwischenergebnisdefinition = „Vorgehensmodell“
- Grundaktivitäten:
  - Analyse
  - Entwurf
  - Implementierung
  - Validation (v.a. Test, Integration)
  - Evolution (v.a. Wartung)

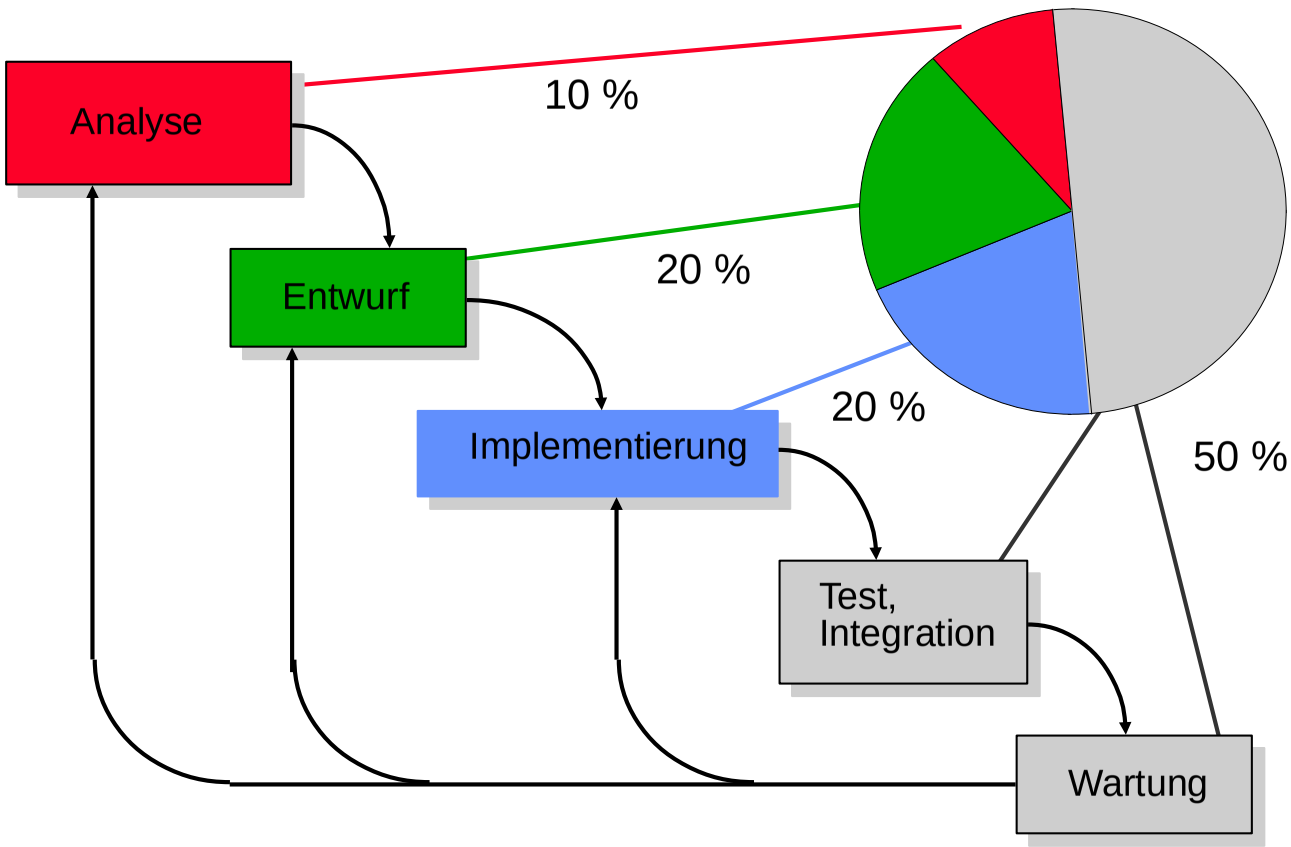
Wasserfall-Modell (pur)



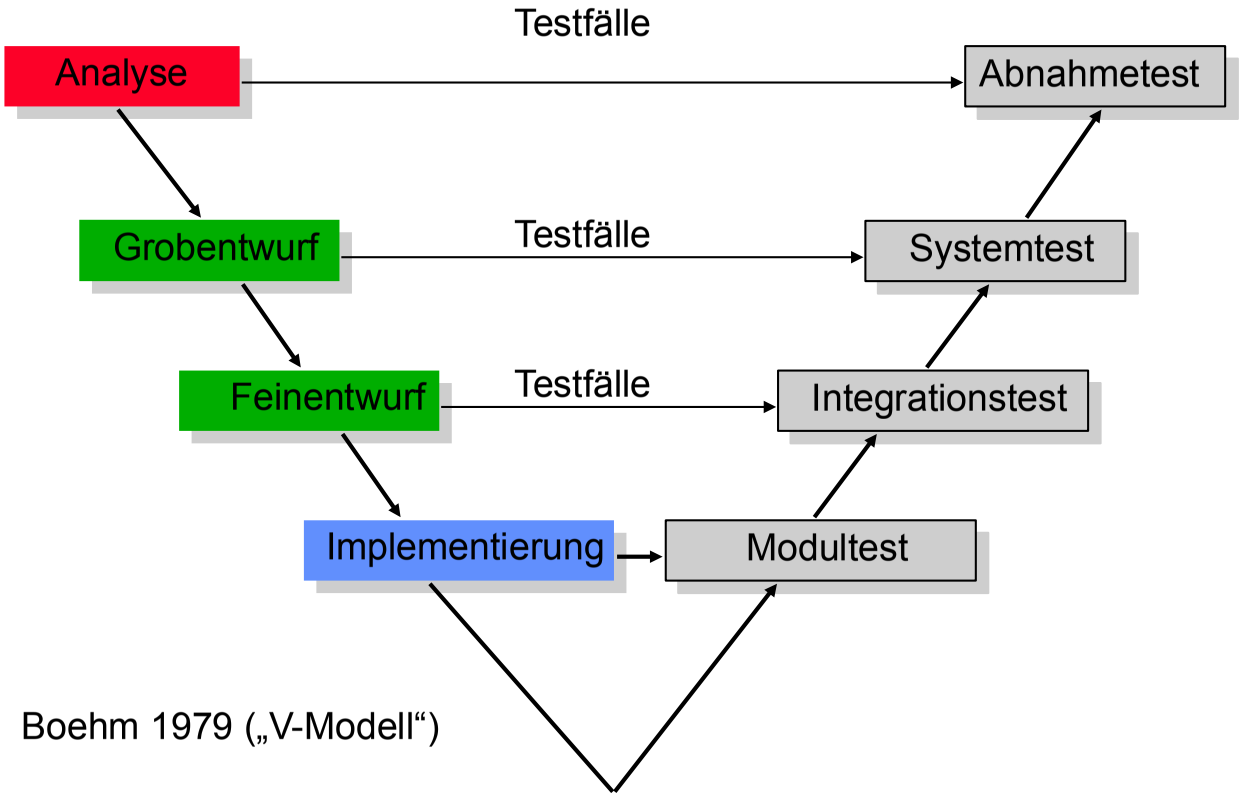
W. Royce (1970)

- Das Wasserfallmodell ist nicht realistisch. Für ein Produkt müssen, schon um des Geschäftsmodells willen, Verbesserungen (Lebenszyklen) eingeplant werden
- Ein Lebenszyklus dauert i.D. 2 Jahre
- Dennoch muss ein Softwareingenieur den “Wasserfall” beherrschen, denn viele andere Vorgehensmodelle setzen darauf auf

Ungefähre Verteilung des Arbeitsaufwandes

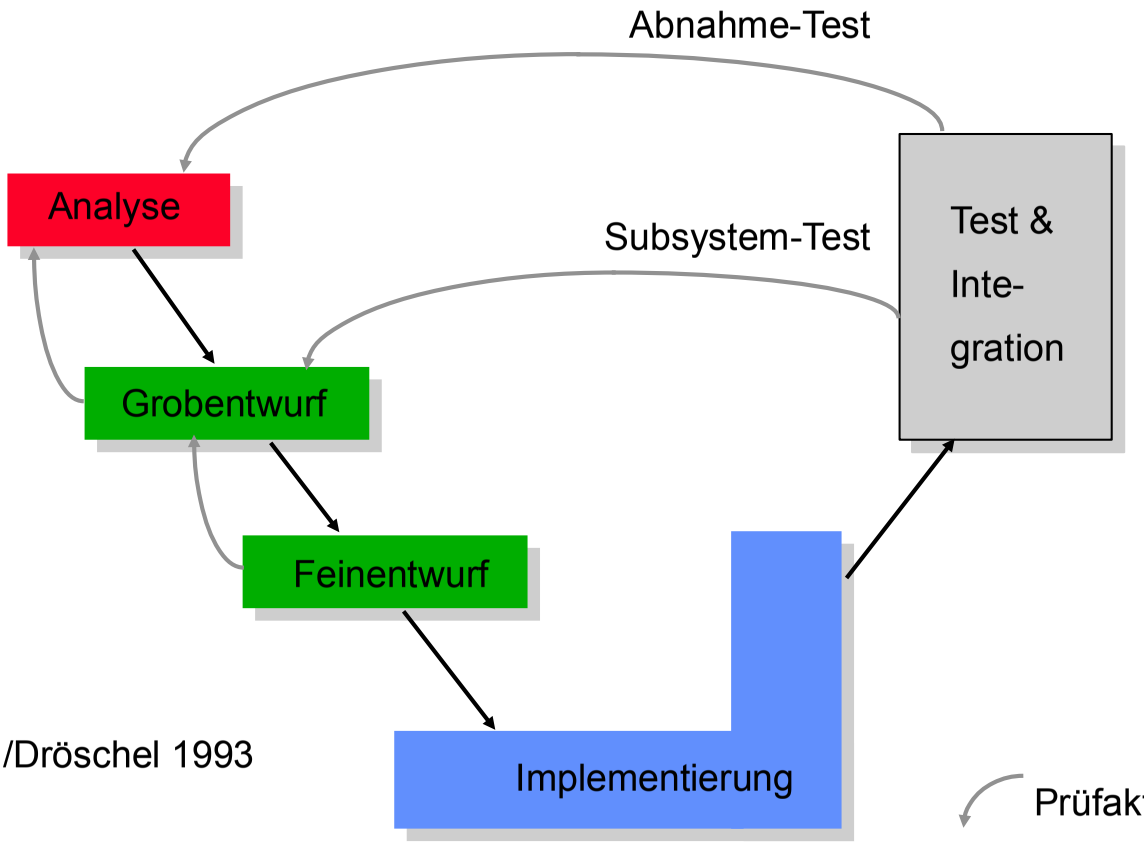


Qualitätssicherung im V-Modell



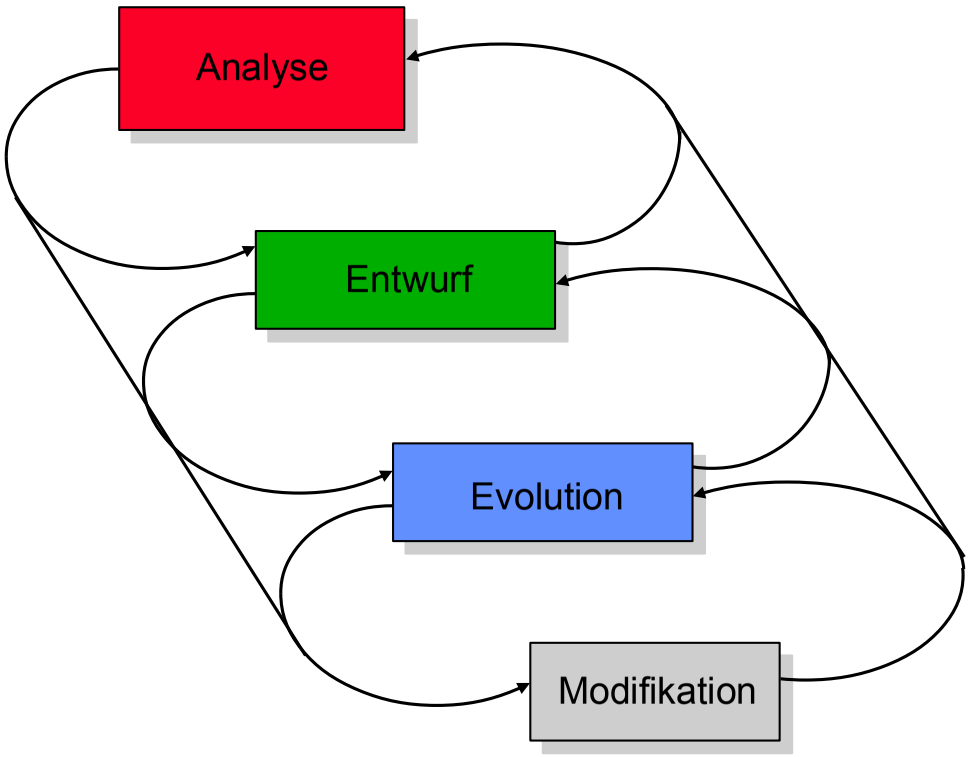
Boehm 1979 („V-Modell“)

V-Modell des BMI (vereinfacht)



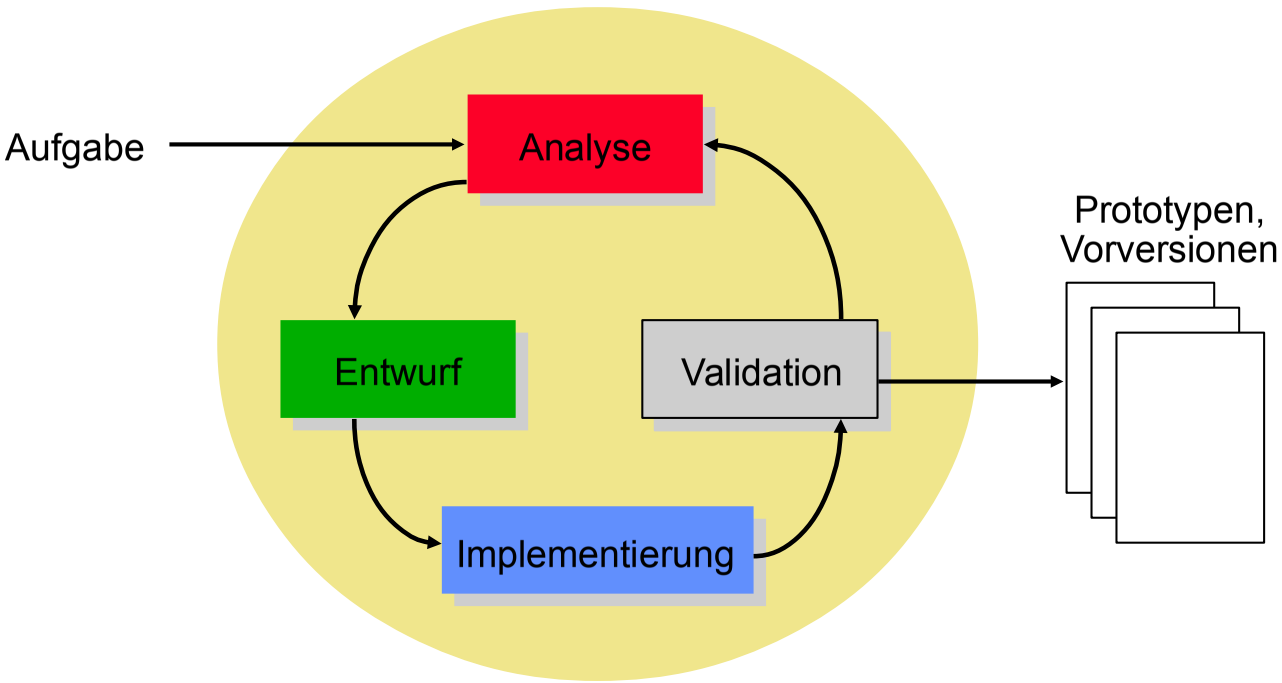
Bröhl/Dröschel 1993

Inkrementelle (evolutionäre) Entwicklung



Evolutionäre Entwicklung

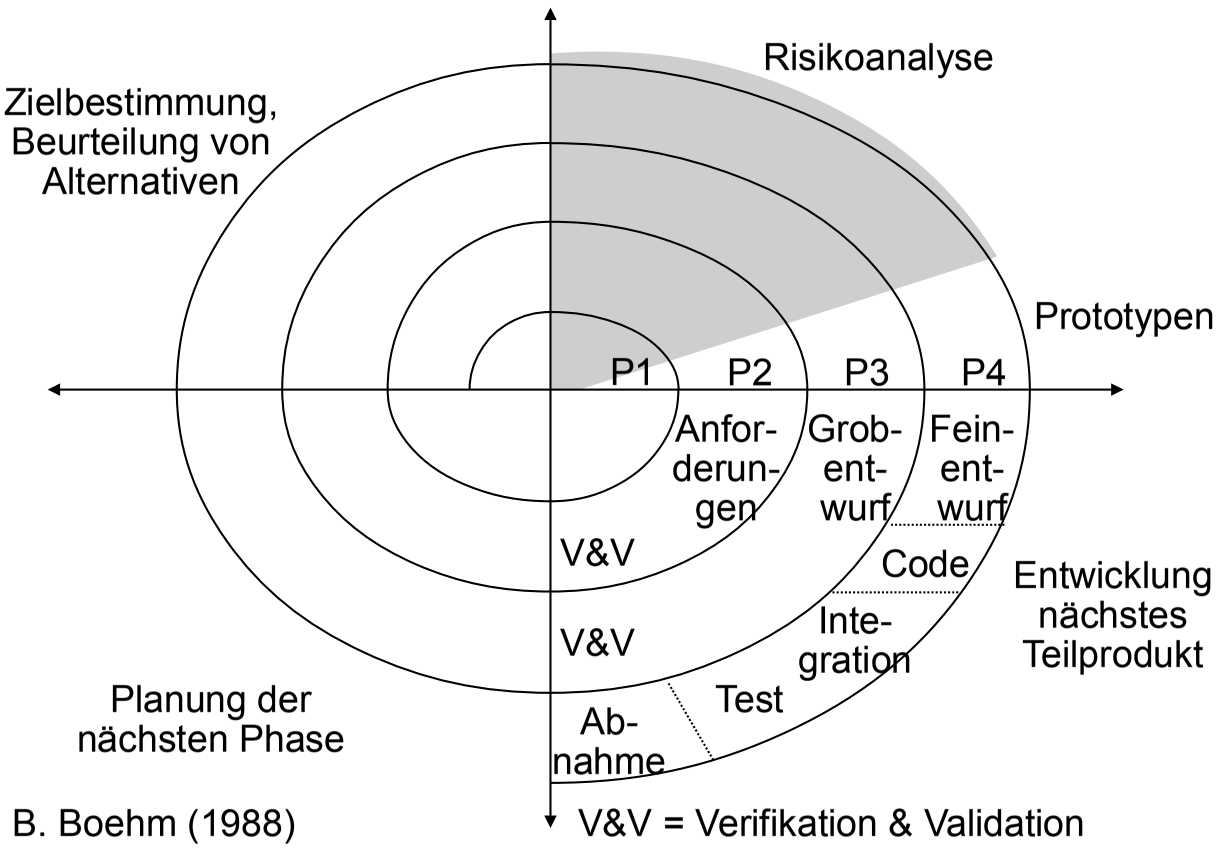
- Typisch für kleinere Projekte oder experimentelle Systeme
- Bei Objektorientierung auch für größere Projekte anwendbar ?



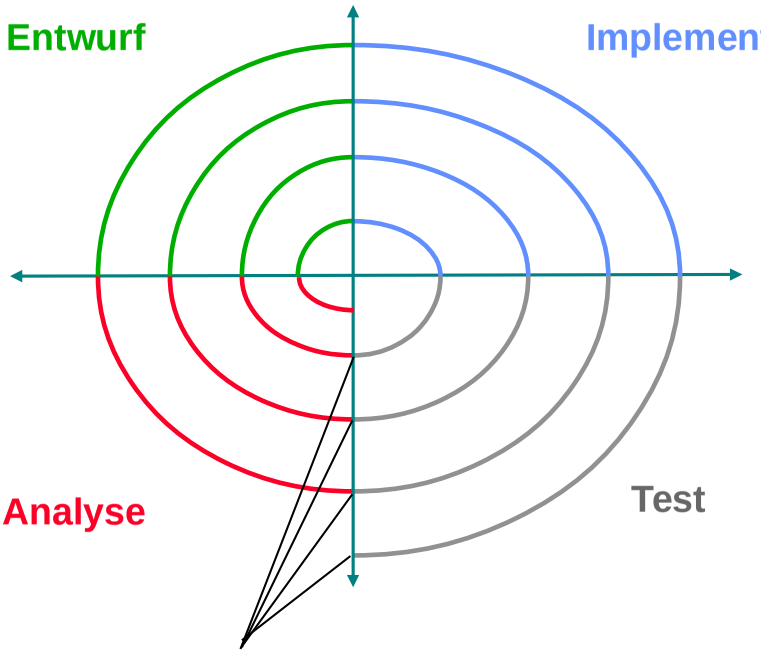
eXtreme Programming (XP)

- Aktuelle, kontrovers diskutierte Entwicklungsmethodik (Kent Beck)
  - Konsequente evolutionäre Entwicklung
  - Der Programmcode ist das Analyseergebnis, das Entwurfsdokument und die Dokumentation. Code wird permanent (Tagesrhythmus) lauffähig gehalten
  - Diszipliniertes und automatisiertes Testen als Qualitätssicherung
  - Diverse weitere innovative Techniken (z.B. Paar-Programmierung)
  - liefert schnell Ergebnisse, aber u.U. auf Kosten der Langlebigkeit
  - kann prinzipiell mit traditionelleren Analyse- und Entwurfstechniken kombiniert werden
- Nachteile
  - wird manchmal als Gegenbewegung zu sauberem Softwareentwurf **miß**verstanden
  - ist nur geeignet für relativ überschaubare, isolierte Anwendungen
- "Agile" Softwareentwicklung (www.agilemanifesto.org):
  - weitere Ansätze, z.B. Crystal, Scrum

Spiralmodell



Objektorientiertes Spiralmodell



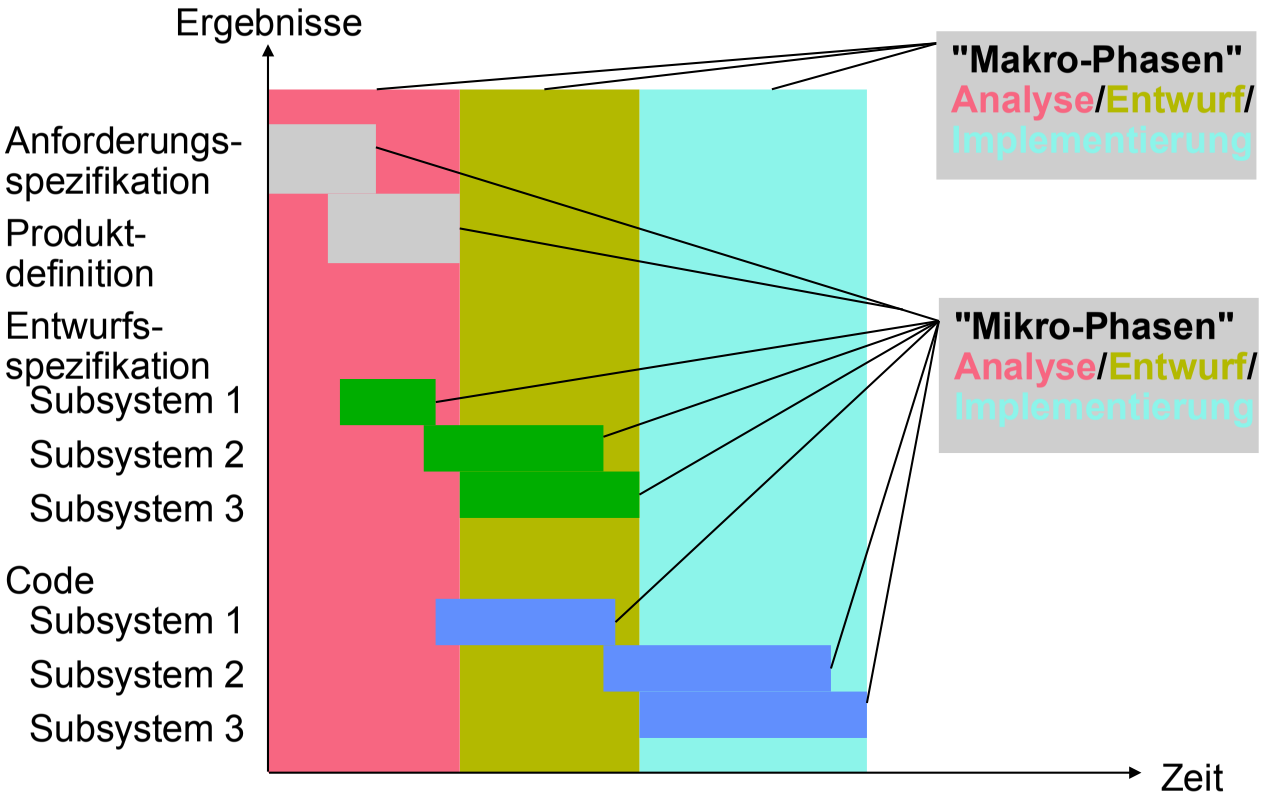
Produkte (Releases)
einschl. *Prototypen*

Langfristige Vorplanung
der Zyklen-Durchläufe

Spiralmodell vs. evolutionäre Entwicklung

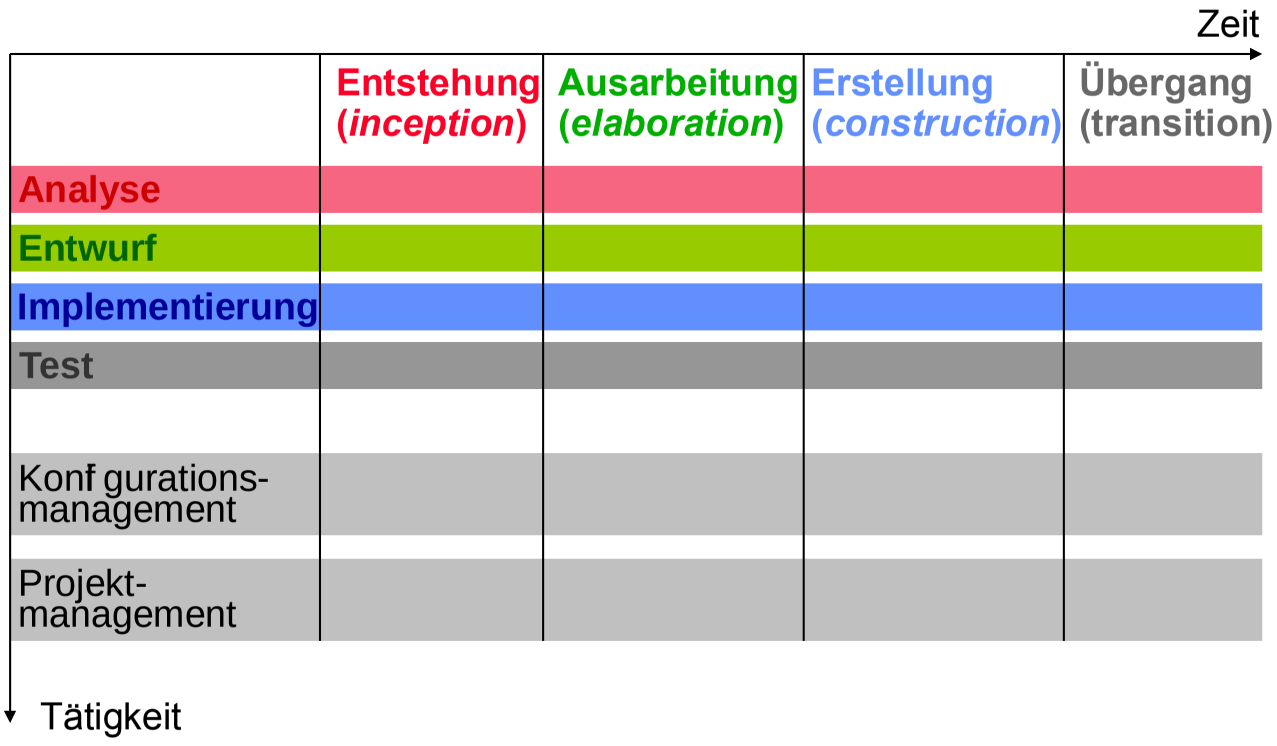
- Grundidee identisch:
  - Zyklisches Durchlaufen von Entwicklungsaktivitätem
  - Aufeinanderfolgende Prototypen
- Evolutionäre und agile Entwicklung:
  - Reaktion auf Änderungen ist wichtiger als Verfolgung eines Plans
  - Planung nur für sehr kurze Zeiträume (Tage, Wochen) im voraus
  - Viele, häufige Durchläufe (z.B. Tagesrhythmus)
- Spiralmodell:
  - Einsetzbar in verschiedener "Strenge"
  - Vorausplanung von Durchläufen
    - Anzahl Durchläufe manchmal schon bei Projektbeginn festgelegt
  - Wenige Durchläufe (z.B. Quartalsrhythmus)
  - Kompromiß zwischen Planbarkeit und Agilität

Parallelität im Entwicklungsprozeß



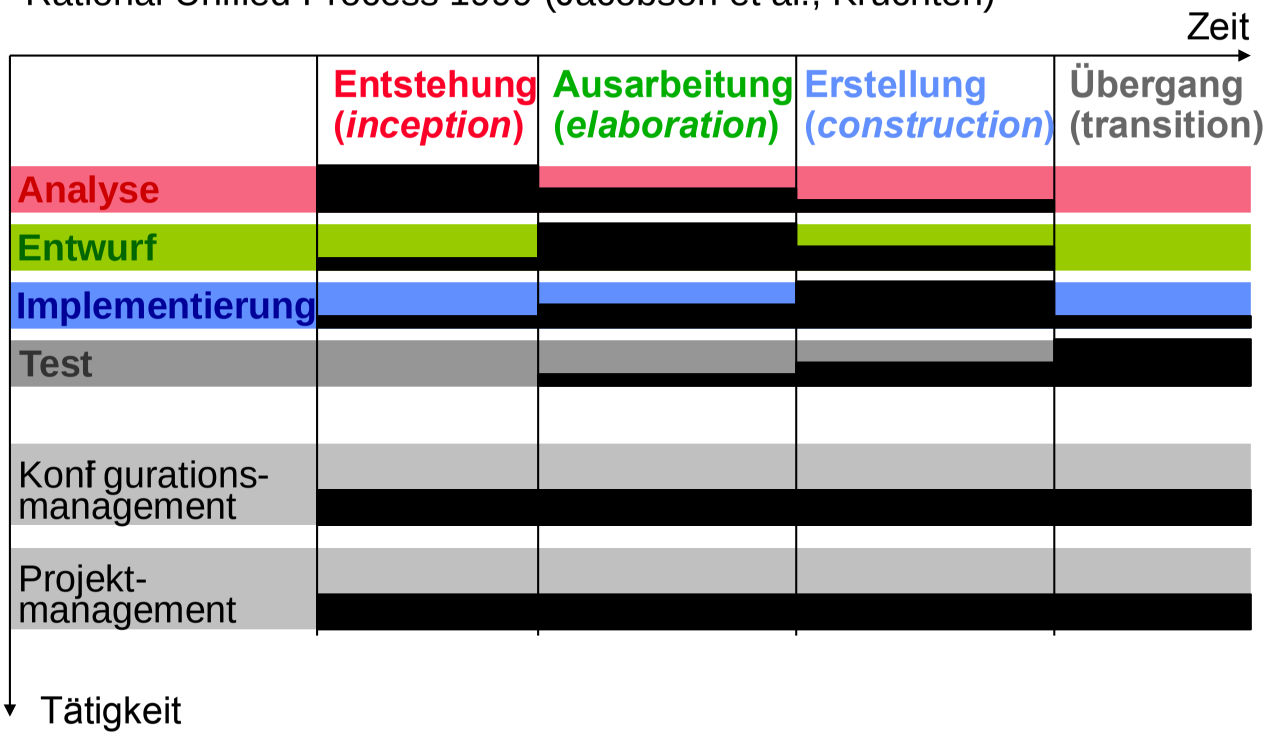
Zweidimensionales Modell

- Rational Unified Process 1999 (Jacobson et al., Kruchten) mit Mikro- und Makrophasen



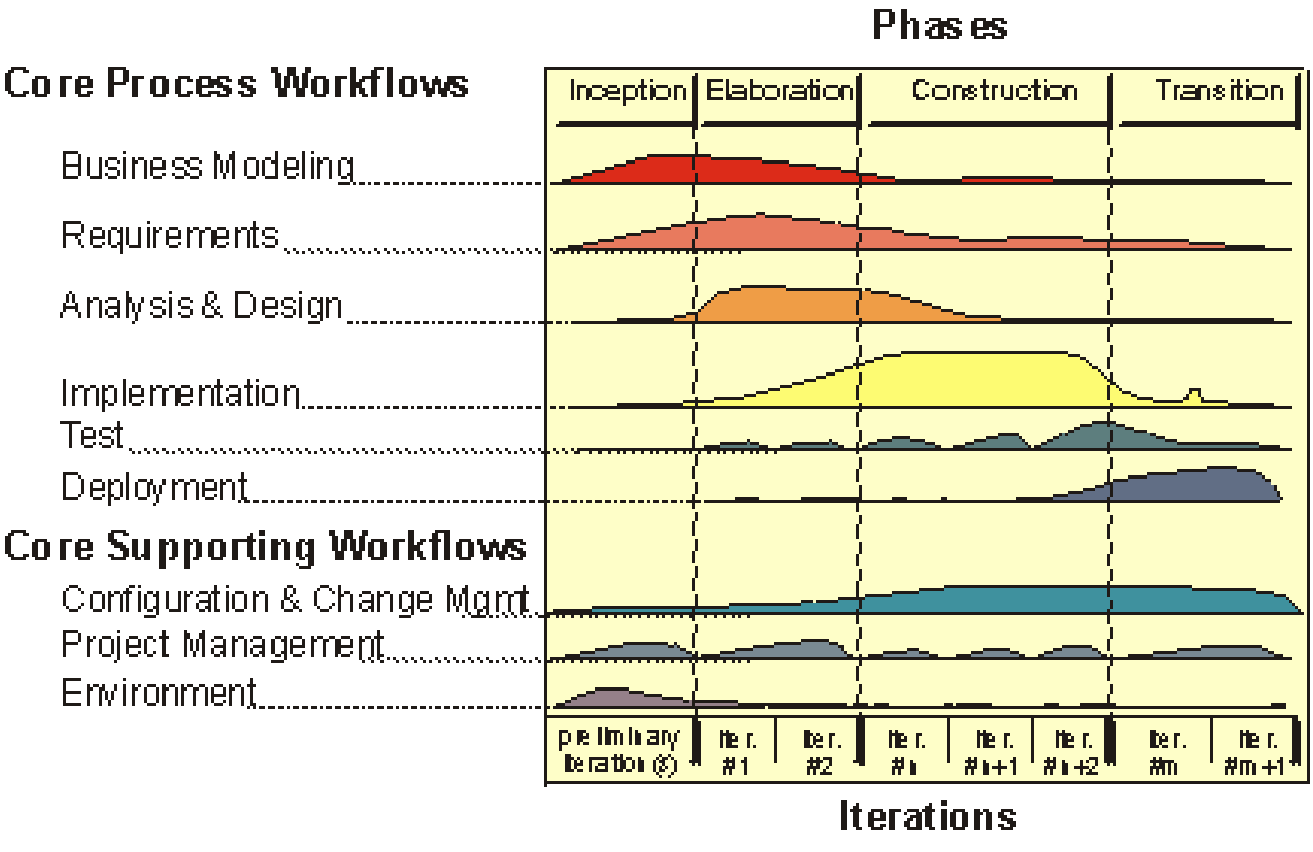
Aufwandsverteilung und Schwerpunkte

Rational Unified Process 1999 (Jacobson et al., Kruchten)

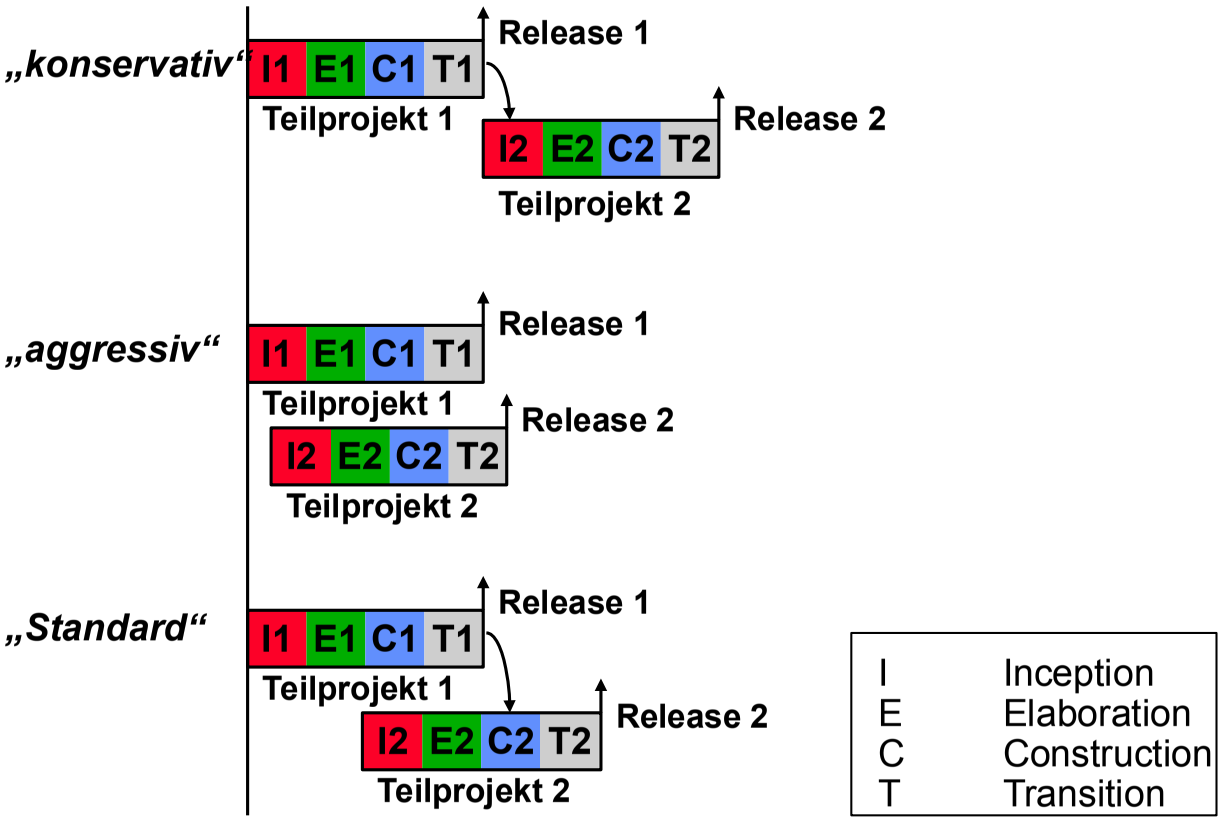


Rational Unified Process (RUP)

- von IBM Rational:



Teilprojekte und Überlappungsgrade



Vorgehen im Softwarepraktikum 3. Semester

- Echte Kunden
- Vorgehensmodell: V-Modell mit Akzeptanztests
- Einfache Inkrementalität: Kunde hat einen *Verbesserungswunsch* frei, der erst zu einem späten Zeitpunkt bekanntgegeben wird
- Intern kann ein inkrementelle Vorgehensmodell gewählt werden

Was haben wir gelernt?

- Vorgehen nach einem strukturierten Phasenmodell ist gewöhnlich besser als ad-hoc Vorgehen
- Realistische Vorgehensmodelle sind iterativ und inkrementell
- Der Ingenieur misst, entwirft, validiert und verbessert

Referenz

- Die deutschen Folien der Softwaretechnologie-Vorlesung stammen zu Teilen aus den Jahren 2000-2003 (Prof. Dr. Heinrich Hussmann, jetzt LMU München). Used by permission.