Grundlagen Software Engineering

Rational Unified Process (RUP)

© Prof. Dr. Liggesmeyer, 1

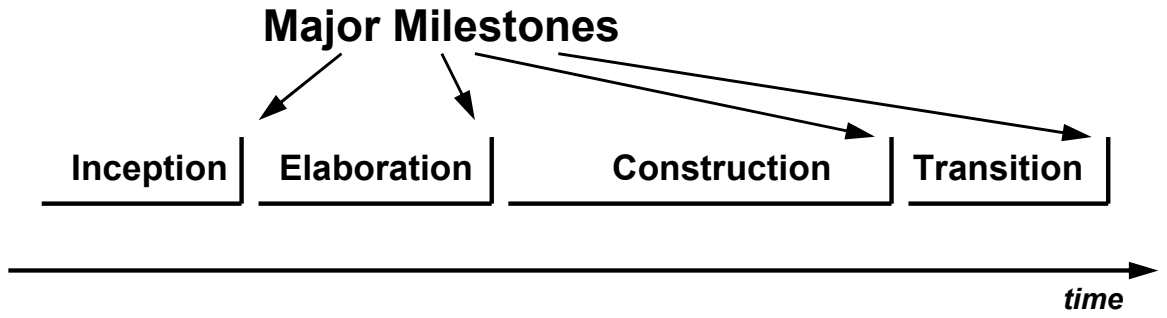
Rational Unified Process (RUP)

- Software Entwicklungsprozess
- Anpassbares und erweiterbares Grundgerüst
- Sprache der RUP ist UML
- Use-Case getrieben
  - Ausgangspunkt und Grundlage der Entwicklung sind Use-Cases
- Architektur zentriert
  - Architektur gliedert das System in Komponenten und Subsysteme
- Iterativer und inkrementeller Prozess
  - Zerlegung in kleinere Teilprojekte
  - Iterationen sind Schritte im Workflow
  - Inkremente sind Ergänzungen und Verbesserungen des Produkts

© Prof. Dr. Liggesmeyer, 2

RUP
Überblick

- Entwicklung besteht aus mehreren Zyklen
- Jeder Zyklus endet mit einer Release des Produkts; d.h. am Ende jedes Zyklus wird dem Kunde ein Produkt geliefert
- Jeder Zyklus besteht aus vier Phasen
  - Inception
  - Elaboration
  - Construction
  - Transition



- Jede dieser Phasen ist wiederum in neun Arbeitsflüsse („*Workflows*“) zerlegt

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 3

RUP
Best Practices

- Iterative Entwicklung
- Anforderungsmanagement
- Architekturzentrierte Entwicklung
- Visuelle Modellierung (mit UML)
- Qualitätssicherung
- Änderungsmanagement (Konfigurationsmanagement)

- Die „Best Practices“ sind Gestaltungsgrundlage für RUP und finden sich in den Abläufen wieder

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 4

RUP
Inception phase
Konzeptualisierung

- Erarbeitung der Produktidee, Vision
- Spezifizierung der wesentlichen Geschäftsvorfälle
- Definition des Umfangs des Projektes
- Kosten und Risiken vorhersagen
  - Vereinfachte Kostenschätzung

- **Life Cycle Objective Milestone**

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 5

RUP
Elaboration Phase
Entwurf/Design

- Spezifizierung der Produkteigenschaften
- Design der Architektur
- Planung der notwendigen Aktivitäten und Resourcen

- **Life Cycle Architecture Milestone**

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 6

RUP
Construction phase
Implementierung

- Erstellung des Produkts
- Entwicklung der Architektur
- Ergebnis: fertiges Produkt

- **Initial Operational Capability Milestone**

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 7

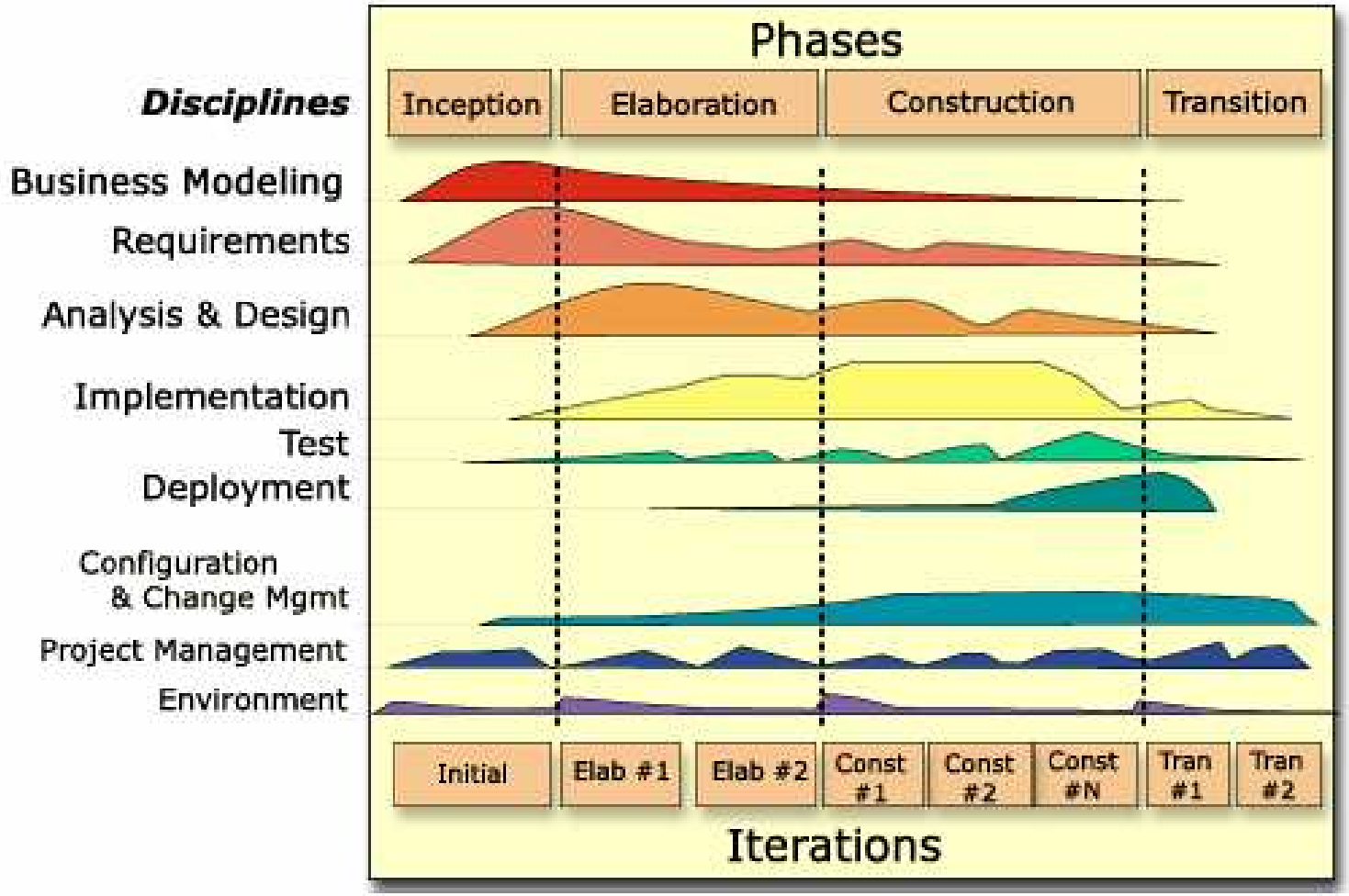
RUP
Transition phase
Produktübergabe

- Freigabe des Produkts an die Benutzer
- Überprüfung des Qualitätslevels
- Auslieferung, Training, Einsatzunterstützung, Wartung

- **Release Milestone**

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 8

RUP
Prozessstruktur



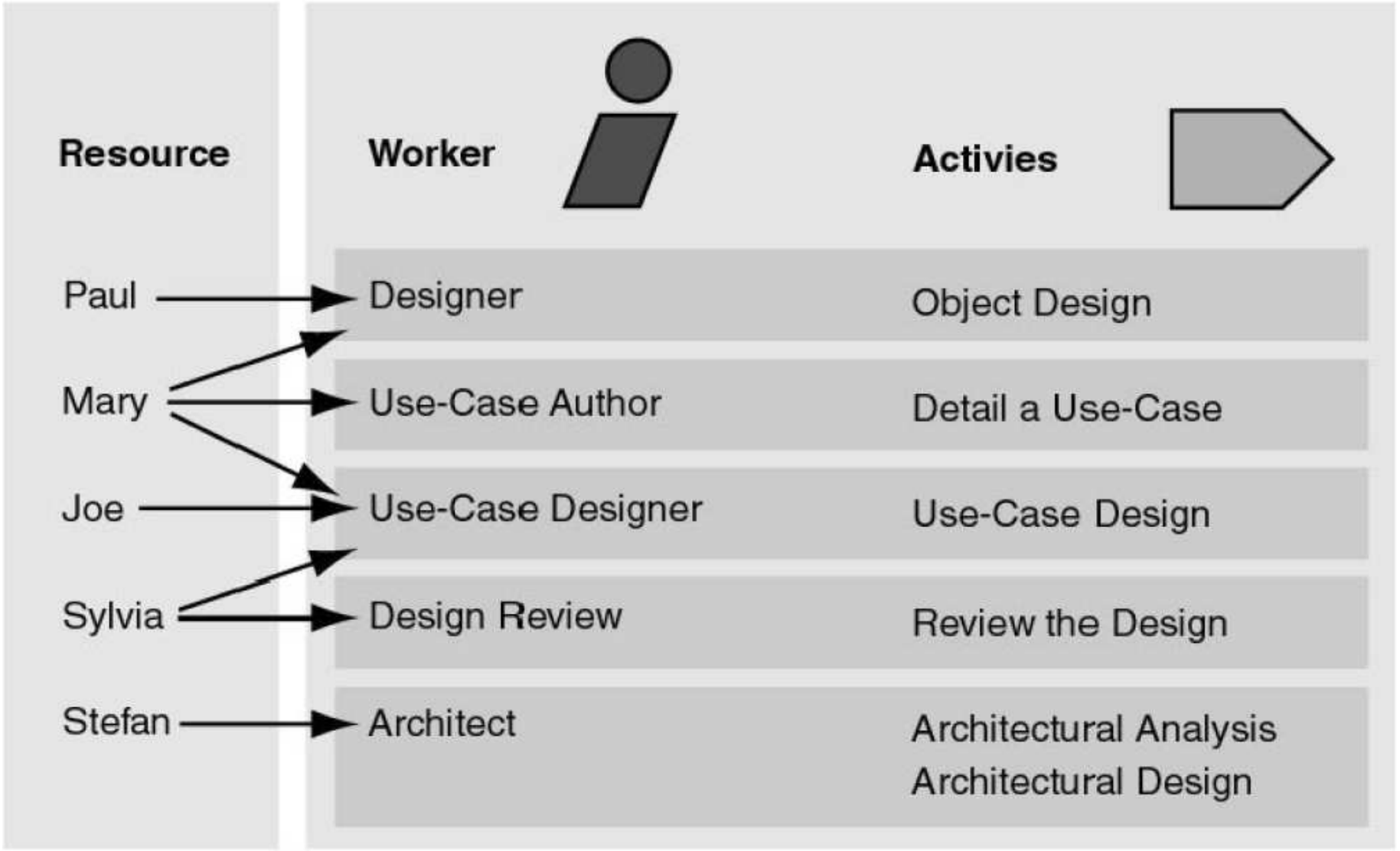
© Prof. Dr. Liggesmeyer, 9

RUP
Prozessstruktur

- Jede Phase besteht aus mindestens einer Iteration
- Eine Iteration ist aus den Workflows zusammengesetzt
- Elemente der Workflows sind Rollen („*Workers*“), Aktivitäten und Artefakte
  - Worker: „wer“
  - Artefakt: „was“
  - Aktivitäten: „wie“
  - Workflows: „wann“
- Für den Prozess ist also festgelegt „wer“, „was“, „wie“ und „wann“ tut

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 10

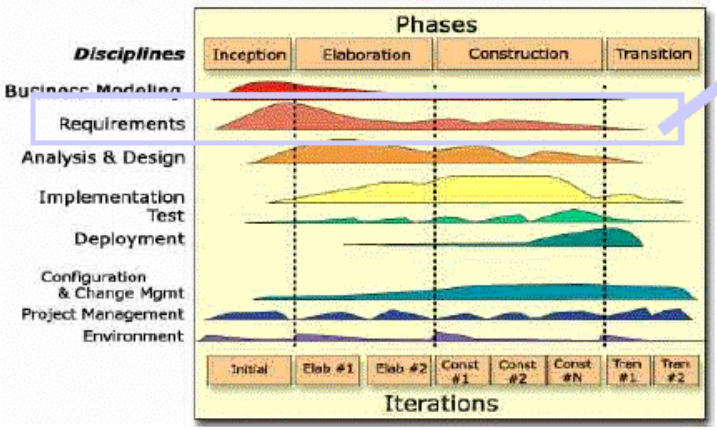
RUP
Personen und Worker

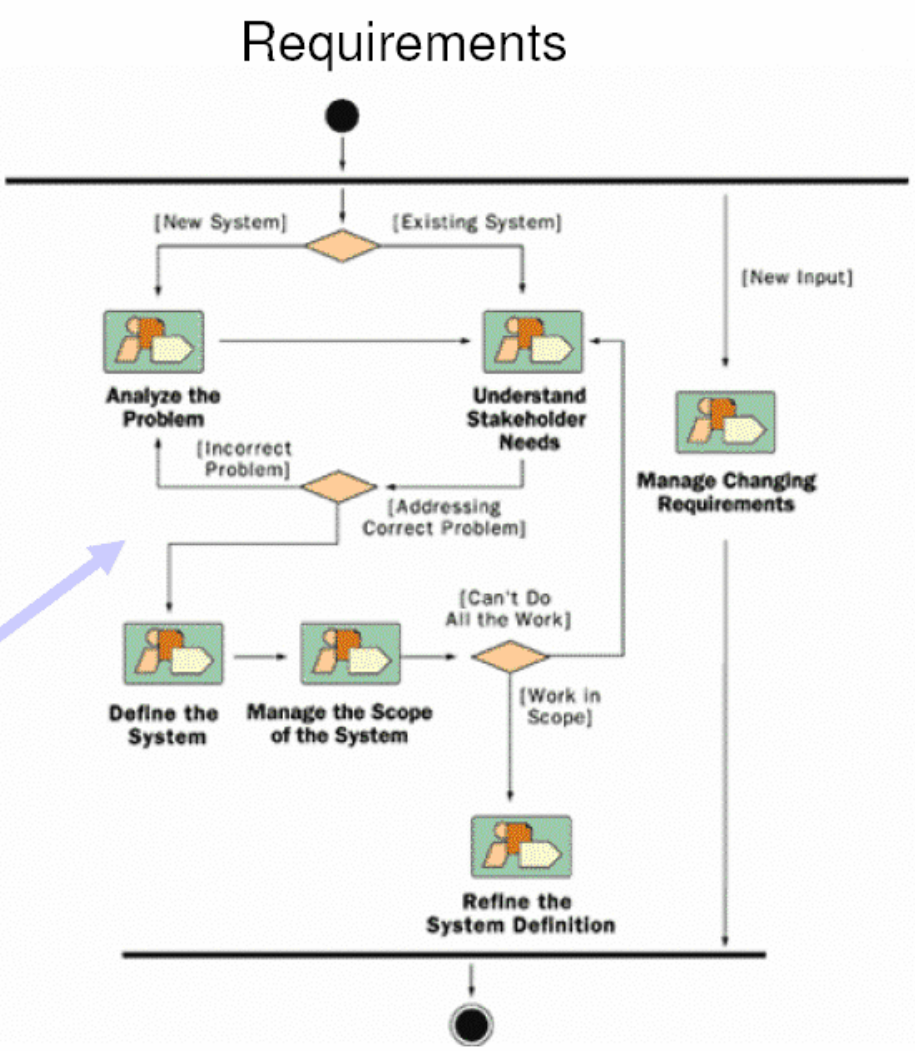


© Prof. Dr. Liggesmeyer, 11

RUP
Workflows

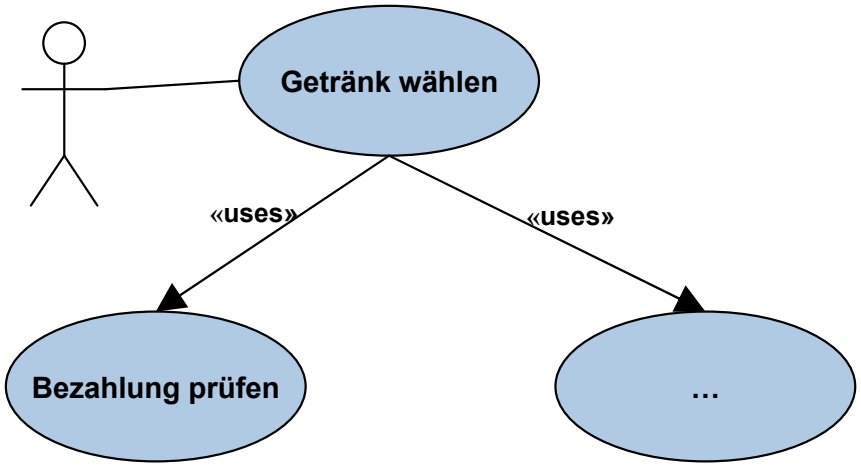
- Für jeden Workflow von Businessmodellierung über Implementierung bis zu Projektmanagement gibt es in RUP Abläufe die durch Tools unterstützt werden





RUP
Use- Case (*Anwendungsfall*) basiert

- Benutzer interagieren mit dem System; dieses führt eine Reihe von Aktionen aus
- Ein Use- Case ist die Beschreibung einer Interaktion und legt die **funktionalen Anforderungen der Benutzer** fest
- Wird durch Akteur initiiert und besteht aus einzelnen Aktivitäten
- Eine Menge von Use- Cases beschreiben die Anforderungen an das gesamte System
- Use-Cases werden in UML modelliert
- Use-Cases sind die Grundlage für alle weiteren Teile des RUP



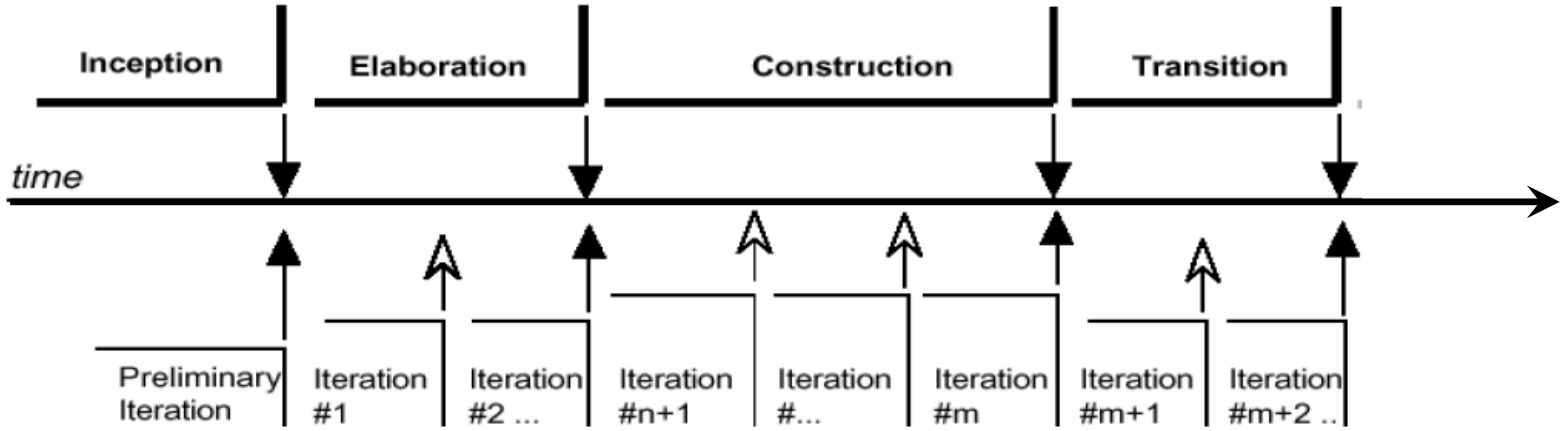
RUP
Architekturzentriert

- Die Architektur gliedert das System in Komponenten und Subsysteme
- Liefert „Sichten“ auf die statischen und dynamischen Aspekte des Systems
  - Logische Sicht
  - Implementierungssicht
  - Prozesssicht
  - Verteilungssicht
  - Use- Case Sicht
- Wird beeinflusst durch
  - Wichtige Use- Cases (funktionale Anforderungen)
  - Plattform (OS, …)
  - Wiederverwendbare Komponenten (Frameworks,…)
  - Alte Anwendungen (Integration mit Legacy Systemen,…)
  - Nicht funktionale Anforderungen (Performance, Zuverlässigkeit, …)
- Die wichtigsten Use-Cases bilden Subsysteme, Klassen oder Komponenten

ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 14

RUP
Iterativ und inkrementell

- Aufteilung des Projektes in kleinere Teilprojekte „Mini-Projekte“
- Jedes Mini-Projekt ist eine Iteration
- Iterationen sind Schritte im Workflow
- Jede Iteration führt zu einer inkrementellen Vergrößerung des Produkts
- Jede Phase besteht aus mindestens einer oder mehreren Iterationen



ENGINEERING SOFTWARE DEPENDABILITY
© Prof. Dr. Liggesmeyer, 15

RUP
Anpassbares Grundgerüst

- RUP ist sehr aufwendig zu realisieren
  - > 30 Rollen
  - > 130 Aktivitäten
  - > 100 Ergebnistypen (Artefakttypen)
- RUP kann jedoch auf die Bedürfnisse einer Firma oder eines Projektes angepasst werden
- Die Workflows können, wenn sie nicht benötigt werden, weggelassen oder verkürzt werden.

© Prof. Dr. Liggesmeyer, 16