Technische Universität Braunschweig

Institut für Flugführung

Prozessmodelle

Einführung in das Projektmanagement bei Softwareprojekten

Prof. Dr.-Ing. Peter Hecker, Dipl.-Ing. Paul Frost, 18. April 2017

Agenda

04. April	Kick-Off
11. April	Projektmanagement
18. April	**Prozessmodelle**
25. April	Versionsverwaltung und Entwicklungsumgebungen
02. Mai	Einführung Arduino/Funduino
09. Mai	Entwicklungsumgebungen und Debugging
16. Mai	Dokumentation und Testing
23. Mai	Dateieingabe und -ausgabe
30. Mai	GUI-Erstellung mit Qt
06. Juni	Exkursionswoche
13. Juni	Bibliotheken
20. Juni	Netzwerke
27. Juni	Projektarbeit
04. Juli	Projektarbeit
11. Juli	Vorbereitung der Abgabe

Teil I

Wiederholung

Motivation

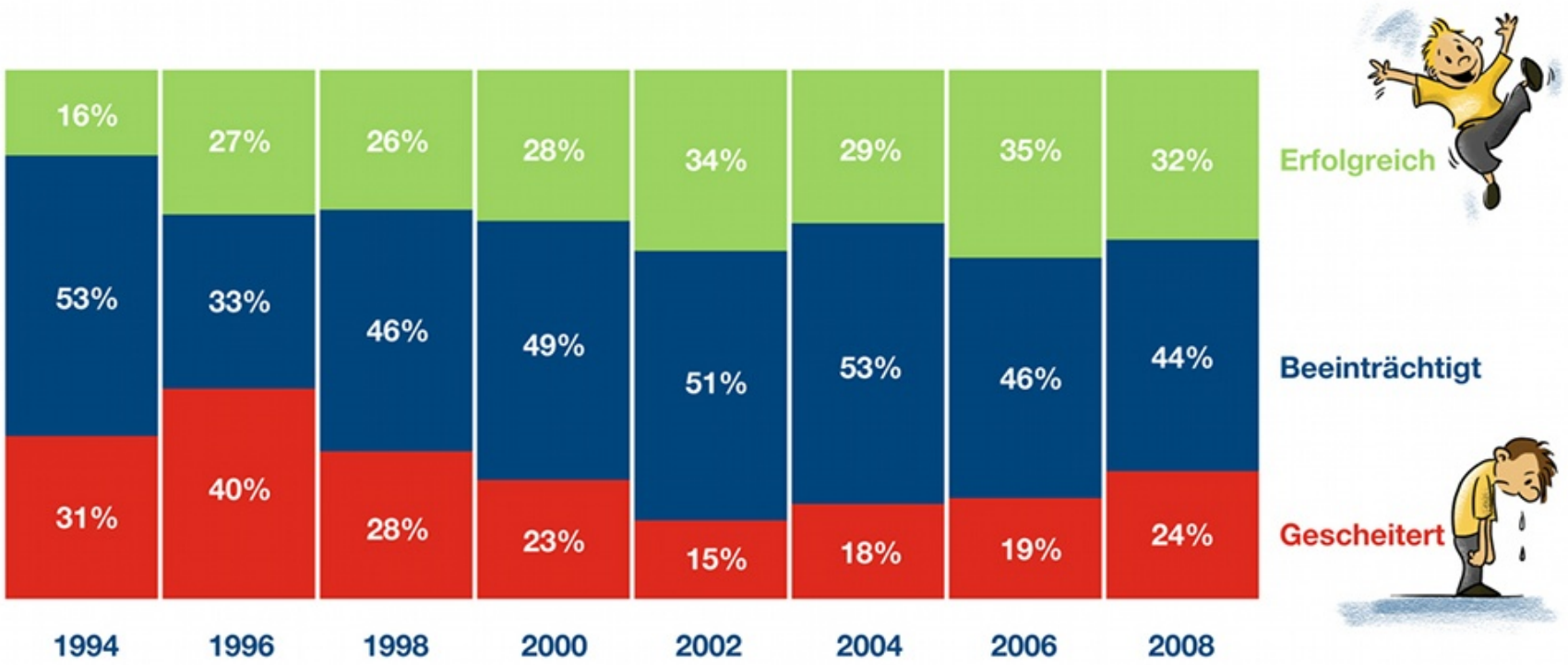


Abbildung 1: Chaos-Report[1]

[1] https://blogs.itemis.com/hs-fs/hubfs/Scrum/Scrum-Wiki/Chaos-Report-D-012.jpg?width=1000&name=Chaos-Report-D-012.jpg&t=1492504670159

Lastenheft

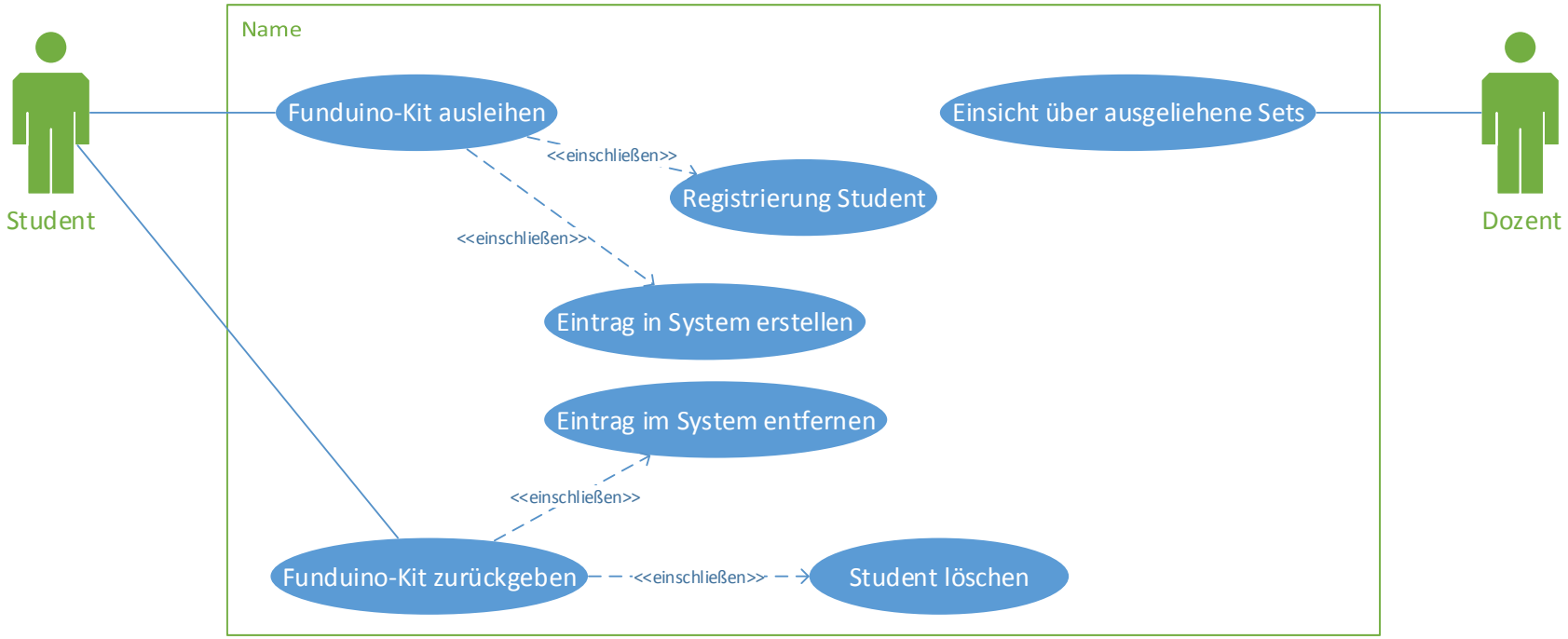
- Jedes Arduino-Kit soll durch ein RFID-Tag eine eindeutige ID erhalten.
- Das Lesegerät soll mit einem Computer verbunden sein.
- Die IDs der Arduino-Kits sollen tabellarisch auf dem Computer gespeichert sein.
- Beim Lesen eines RFID-Tags sollen die tabellarisch gespeicherten Daten des Kits mit der entsprechenden ID angezeigt werden.

Lastenheft

- Über den Computer sollen Informationen in der Tabelle eingetragen/geändert werden können. Dazu zählen:
  - Gruppennummer/-name
  - Vollständige Namen und Matrikelnummer der Studenten, an die das Arduino Kit ausgeliehen wurde
  - Eine Übersicht der Bauteile, die seit einem bestimmten Datum fehlen oder defekt sind, mit Angabe von weiteren Hinweisen
- In der Tabelle sollen neben der Kit-Nummer weitere, unveränderliche Informationen angezeigt werden.
- Über die Eingabe der Kit-Nummer sollen die Daten des entsprechenden Kits auch ohne das Lesen des RFID-Tags angezeigt und bearbeitet werden können.

Projekt

Use-Case Diagramm



Teil II

Prozessmodelle

Softwareentwicklung

Abbildung 2: Einwurf des Balls durch Luke Burgess in ein Gedränge [2]

[2] By PierreSelim - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=17336884

Ein simples Modell

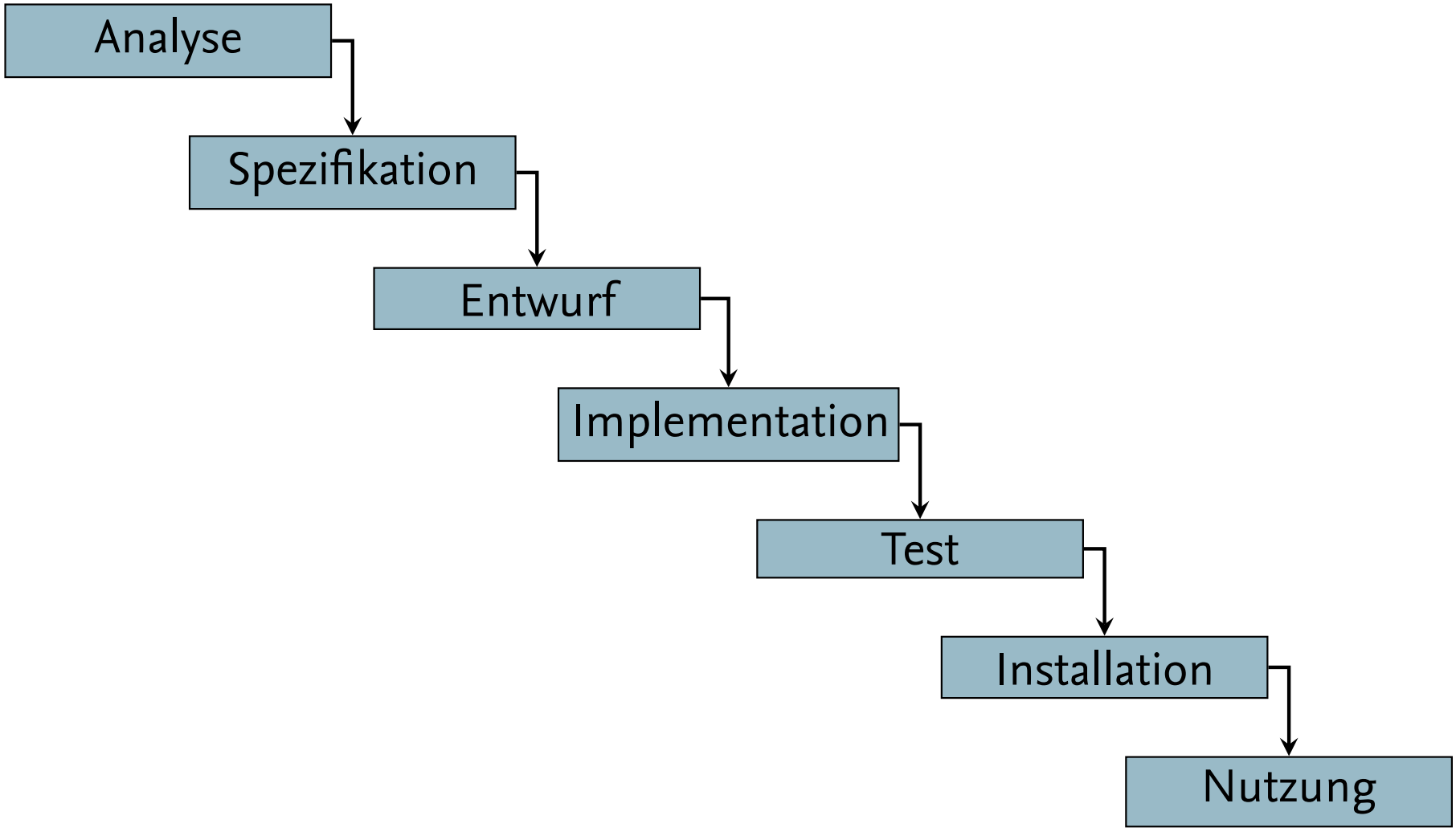
1. Quellcode schreiben
2. Fehler beheben

Nachteile:

- Fehlerbehebungen führen meistens zu Umstrukturierungen

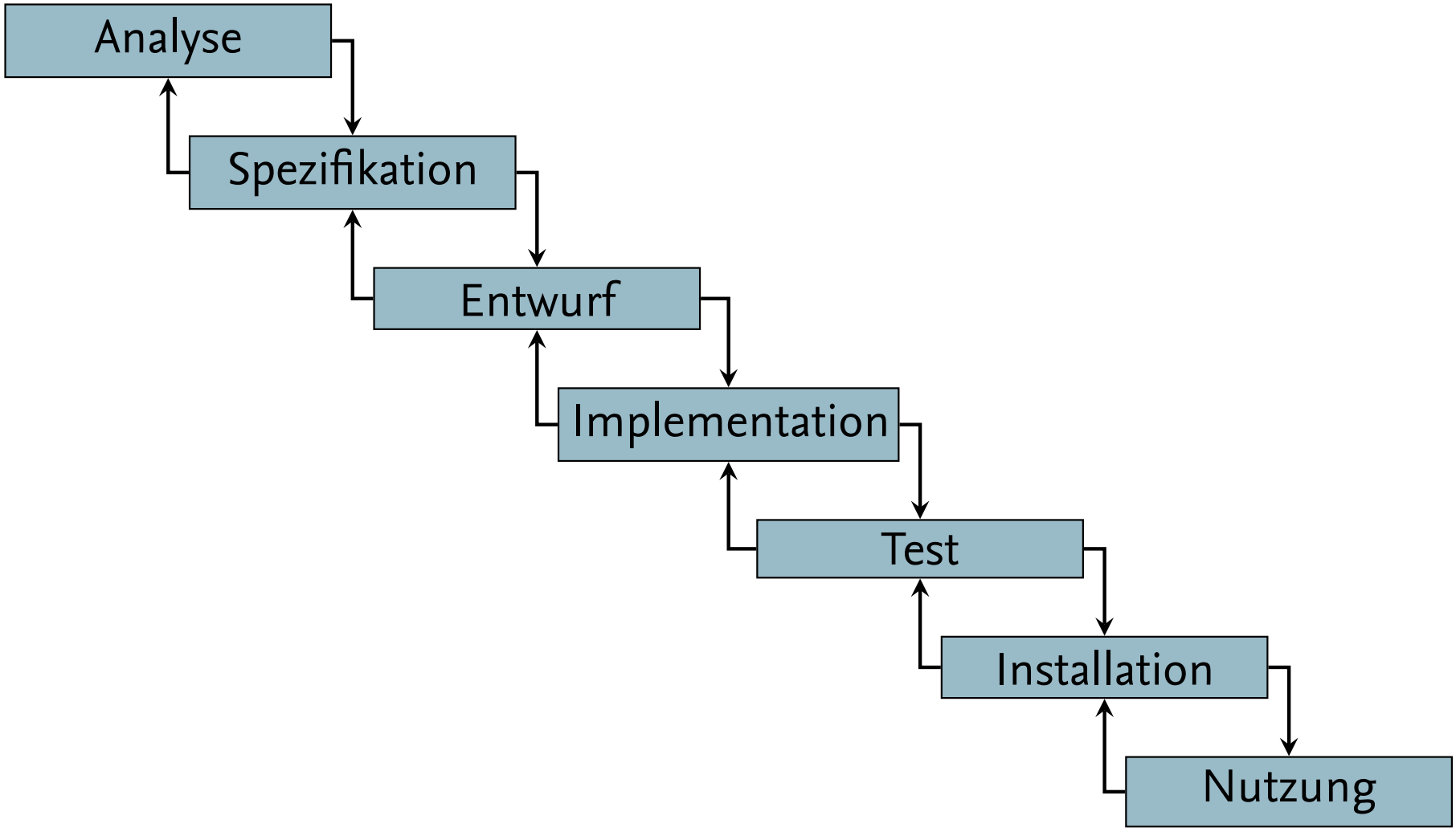
  $\Rightarrow$ Weitere Behebungen werden aufwendiger
- Wenig Akzeptanz des Produkts beim Endnutzer
- Fehleridentifikation ist sehr schwierig, weil Tests nur unzureichend vorbereitet wurden

Das Wasserfallmodell[3]



tware Engineering Economics. 1981.

Das Wasserfallmodell[3]



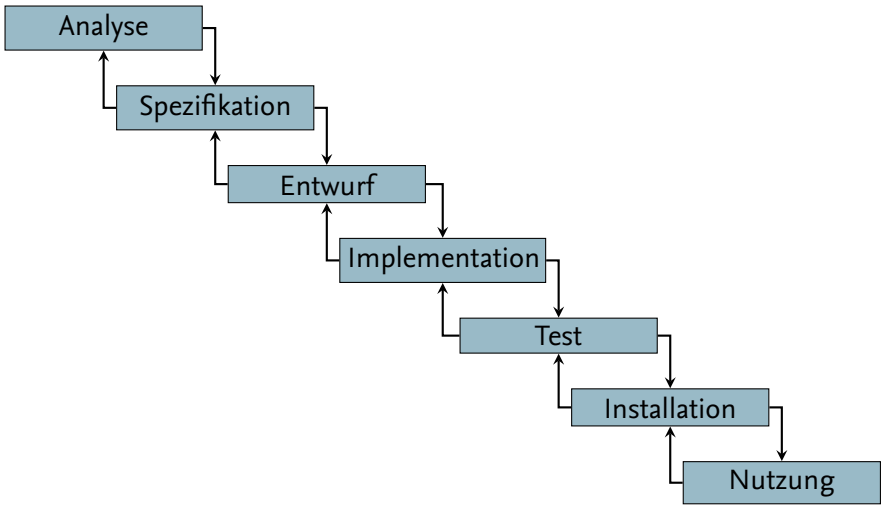
ngineering Economics.

Wasserfallmodell

Spezifikation[4]

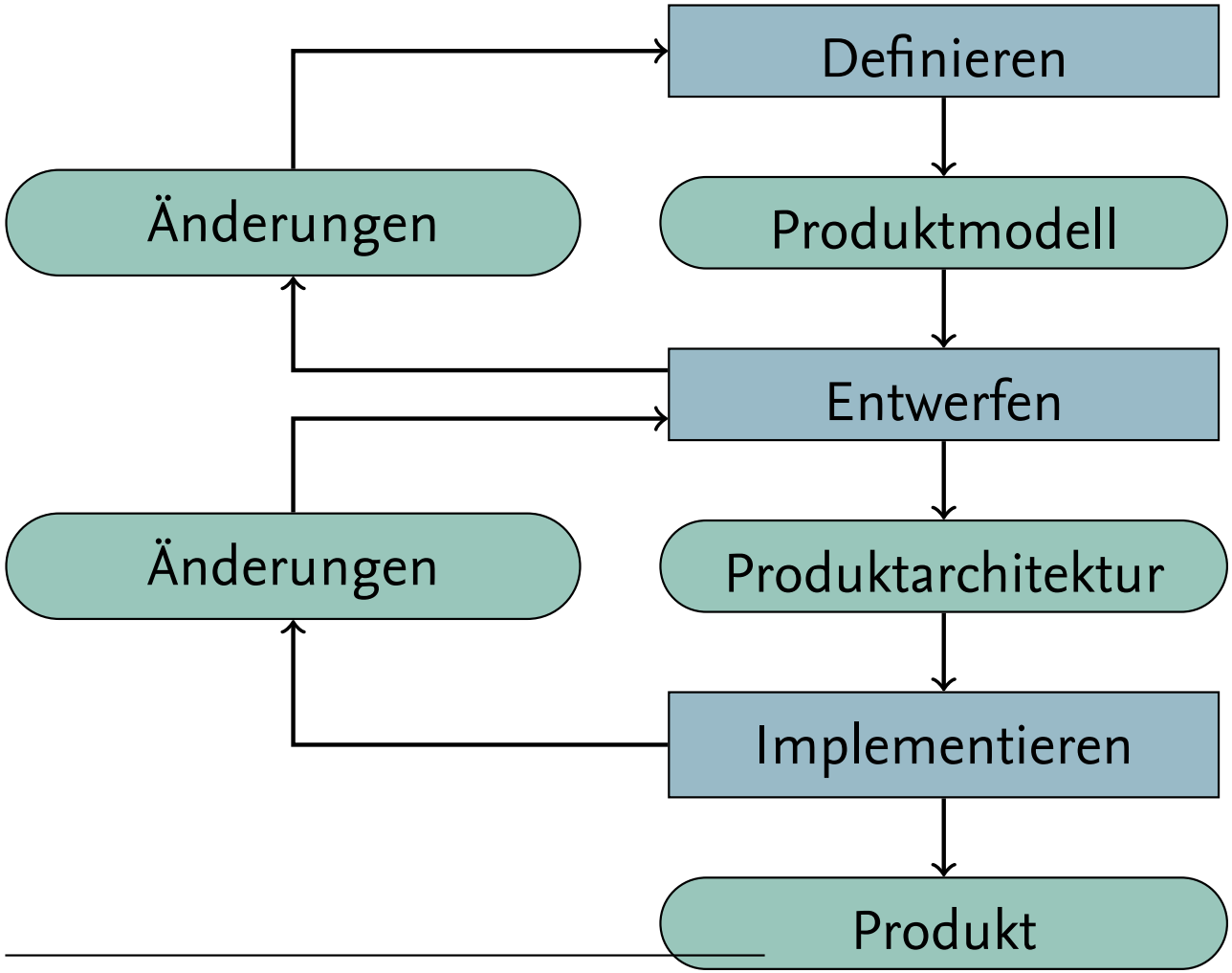
- Jede Phase wird mit einem fertigen Dokument abgeschlossen
- Sequentielle Entwicklung
- Am Top-Down-Vorgehen orientiert

  Es wird immer konkreter



[4] Helmut Balzert. Lehrbuch der Software-Technik. 1998.

Das Wasserfallmodell schematischer Ablauf[5]



[5] Balzert, Lehrbuch der Software-Technik.

Wasserfallmodell — Vor- und Nachteile[6]

Vorteile:

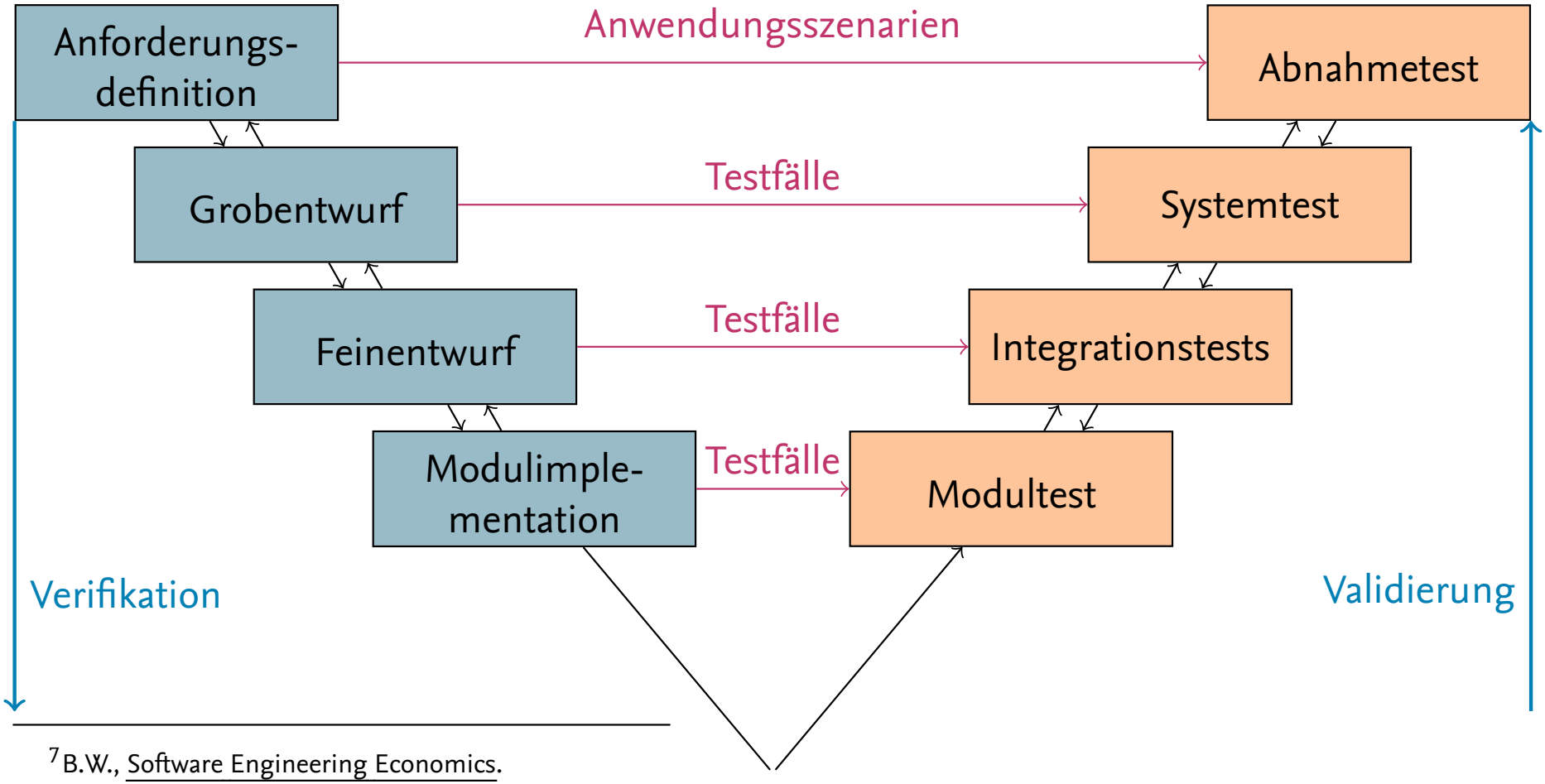
- Einfach
- Geringer Managementaufwand

Nachteile:

- Nicht immer sinnvoll Phasen komplett abzuschließen
- Nicht immer sinnvoll alle Phasen sequentiell abzuarbeiten
- Dokumente haben zum Teil eine höhere Priorität als das Produkt
- Risiken könnten vernachlässigt werden, da der festgelegte Ablauf auch in der Form durchgeführt wird

[6] Balzert, Lehrbuch der Software-Technik.

Vorgehensmodell (V-Modell)[7,8,9]



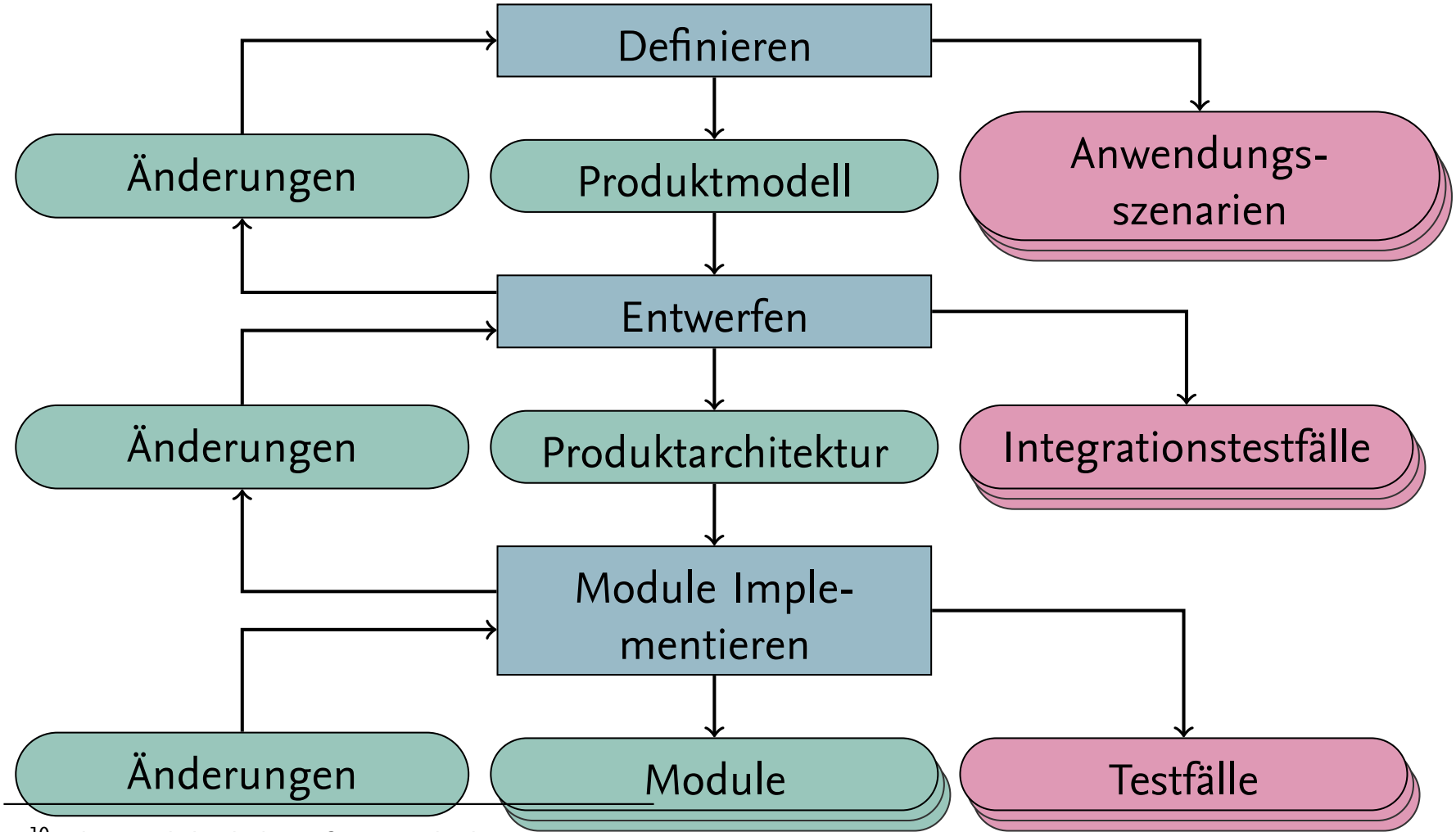
[7] B.W., Software Engineering Economics.

[8] Boehm B.W. Verifying and Validating Software Requirements and Design Specifications, in IEEE Software. 1984.

[9] Balzert, Lehrbuch der Software-Technik.

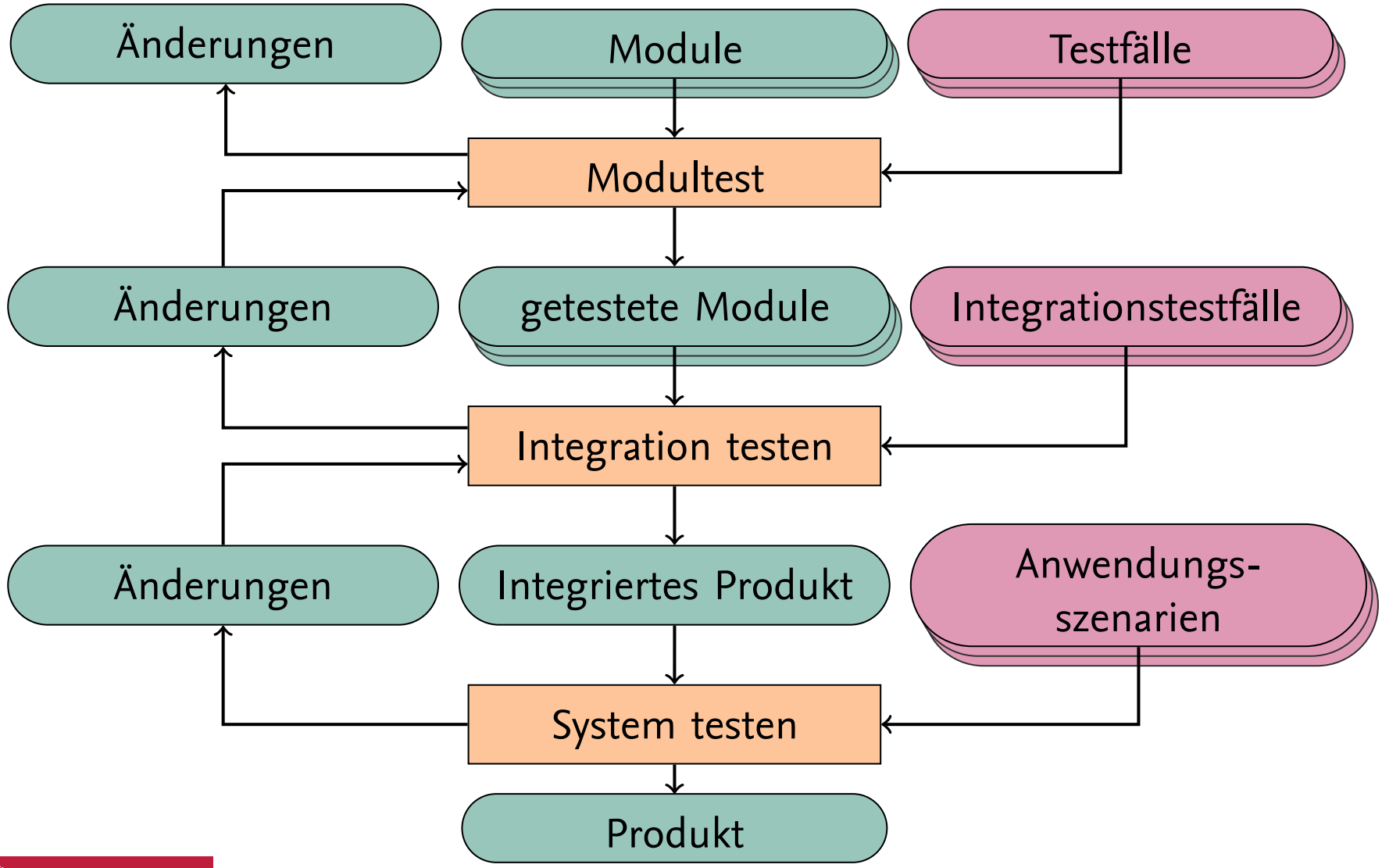
- Diente als Vorlage für die Vorgehensmodelle bei der Bundeswehr und Behörden
- Dokumentengetriebenes Modell
- Erweiterung des Wasserfallmodells um Qualitätssicherung
- Das V-Modell ist unterteilt in
  - Systemerstellung
  - Qualitätssicherung
  - Konfigurationsmanagement
  - Projektmanagement

V-Modell schematischer Ablauf 1/2[10]



[10] Balzert, Lehrbuch der Software-Technik.

V-Modell schematischer Ablauf 2/2



Vor- und Nachteile[11]

Vorteile:

- Generisches Modell, das angepasst werden kann
- Ermöglicht standardisiertes Vorgehen
- Gut geeignet für große Projekte

Nachteile:

- Sehr aufwendig
- Sehr bürokratisch
- Ohne Computer Aided Software Engineering (CASE)-Unterstützung nicht handhabbar
- Sehr viele Rollen erforderlich

[11] Balzert, Lehrbuch der Software-Technik.

Prototypen-Modell[12]

- Anforderungen herausfinden
- Diskussionsbasis und Hilfe bei Entscheidungen
- Sammlung von praktischen Erfahrungen

Unterscheidung von Prototypen

- Prototyp zur Klärung von Fragen
- Prototyp zur Erstellung der Produktdefinition
- Prototyp zur inkrementellen Weiterentwicklung

[12] Balzert, Lehrbuch der Software-Technik.

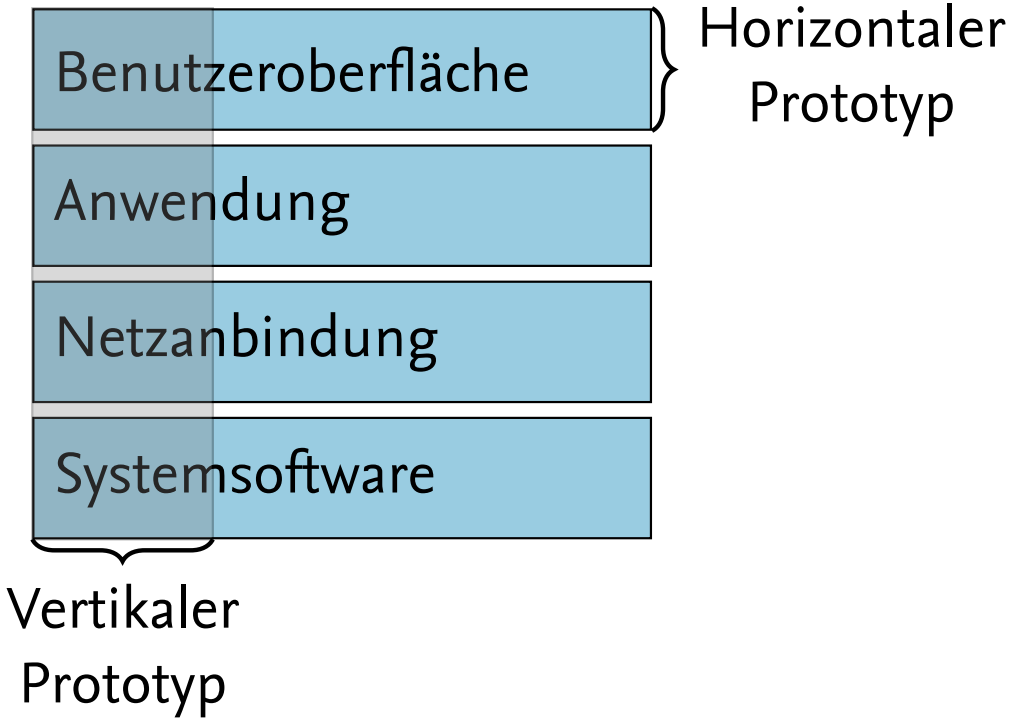
Prototypen-Modell[13]

Horizontaler Prototyp

- Realisiert spezifische Ebenen
- Ebene wird vollständig realisiert

Vertikaler Prototyp

- Umfasst alle Ebenen des Systems
- Ebenen werden partiell realisiert



[13] Balzert, Lehrbuch der Software-Technik.

Prototypen-Modell — Vor- und Nachteile[14]

Vorteile:

- Geringeres Entwicklungsrisiko
- Bessere Planbarkeit
- Unterstützung von anderen Prozessmodellen
- Benutzer können besser einbezogen werden

Nachteile:

- Mehr Aufwand
- Ein unfertiger Prototyp könnte zum Endprodukt werden
- Oft nicht eingegrenzt

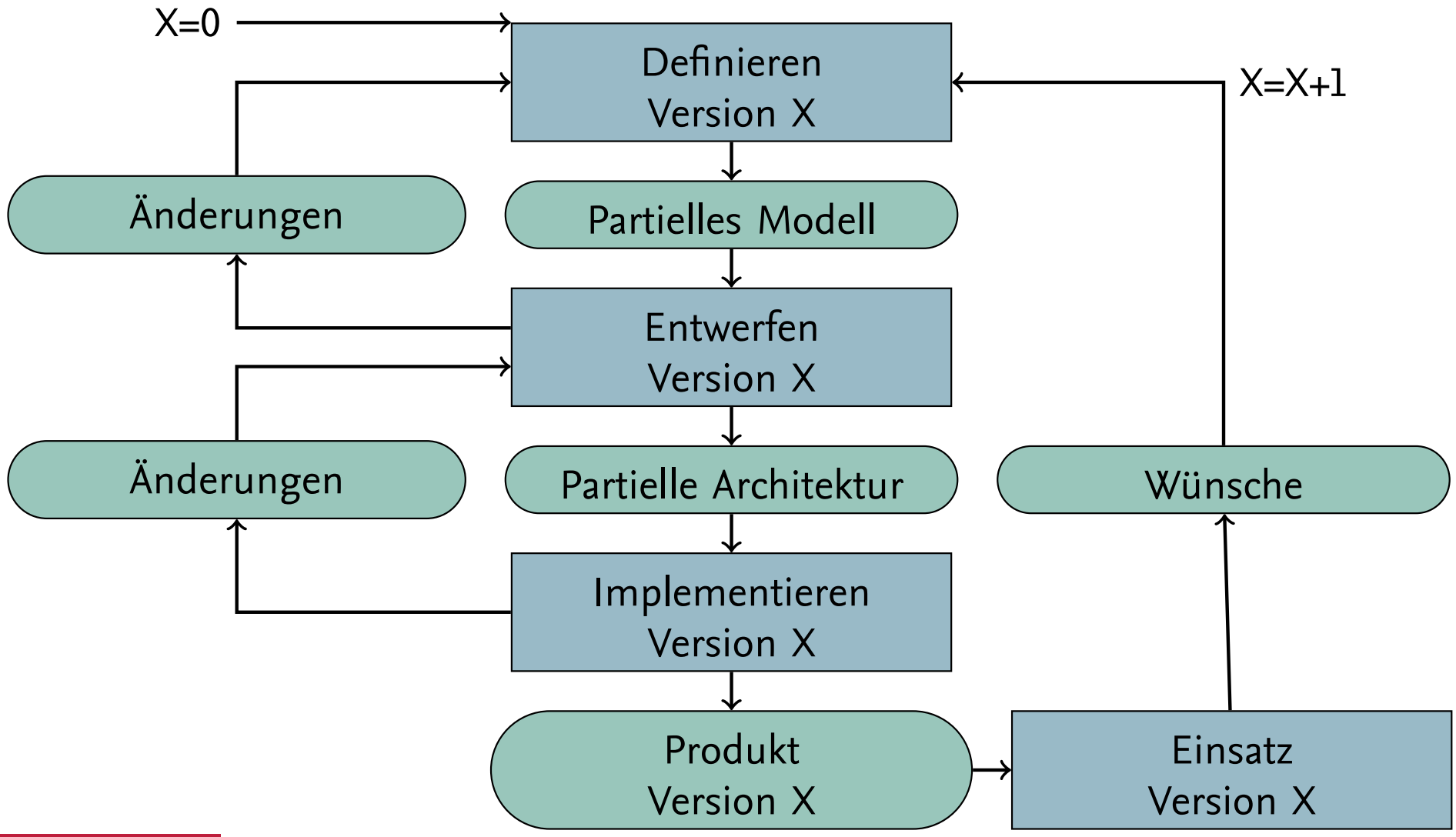
[14] Balzert, Lehrbuch der Software-Technik.

Evolutionäres Modell[15]

- Stufenweise Entwicklung
- Gut für nicht komplett generierbare Anforderungen
- Konzentration auf lauffähige Teilprodukte

[15] Balzert, Lehrbuch der Software-Technik.

Evolutionäres Modell schematischer Ablauf



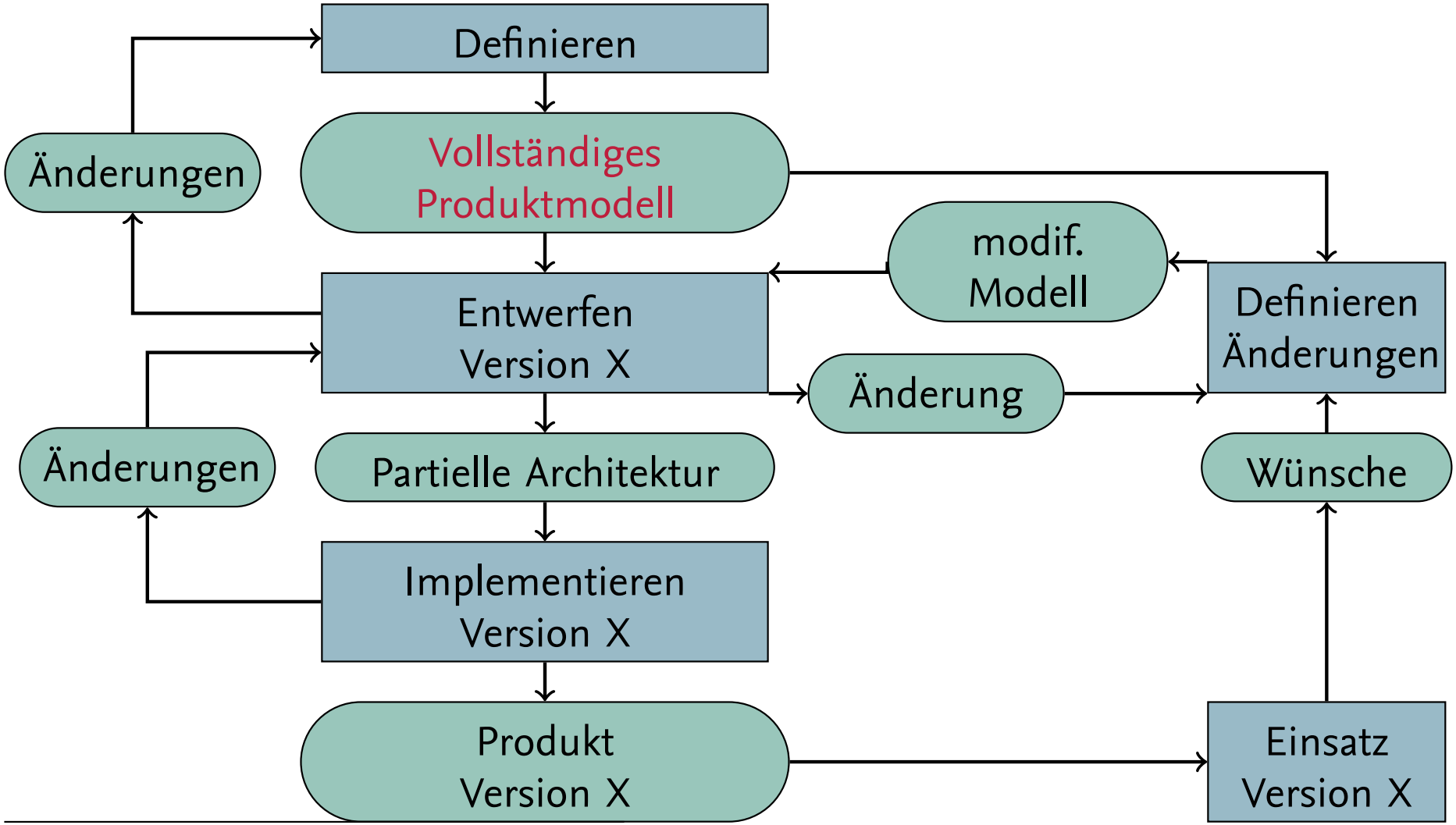
Inkrementelles Modell[16]

- Aufgebaut wie das evolutionäre Modell

  aber
- Anforderungen werden vollständig erfasst und modelliert
- Nur ein Teil der Anforderungen wird umgesetzt

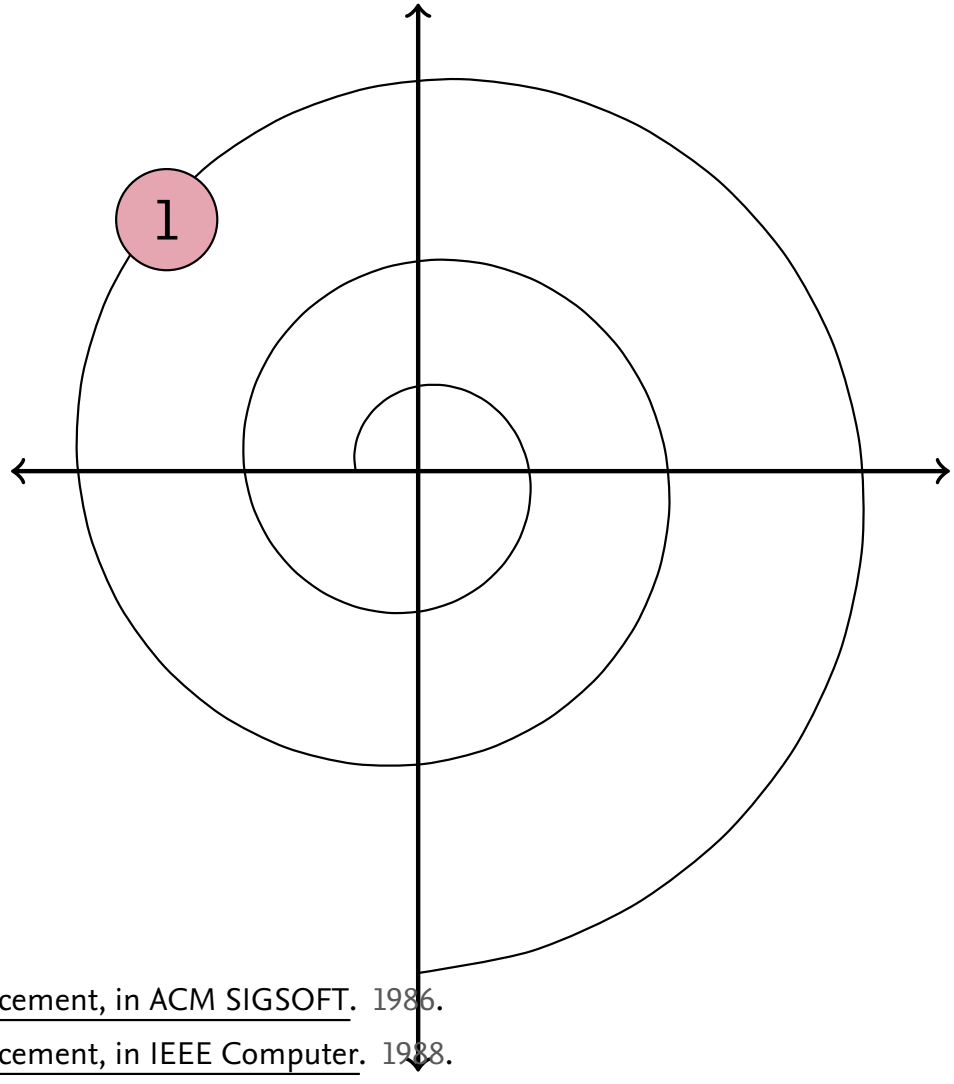
[16] Balzert, Lehrbuch der Software-Technik.

Inkrementelles Modell — Schematischer Ablauf[17]



[17] ...h der Software-Technik.

Spiralmodell[18,19]



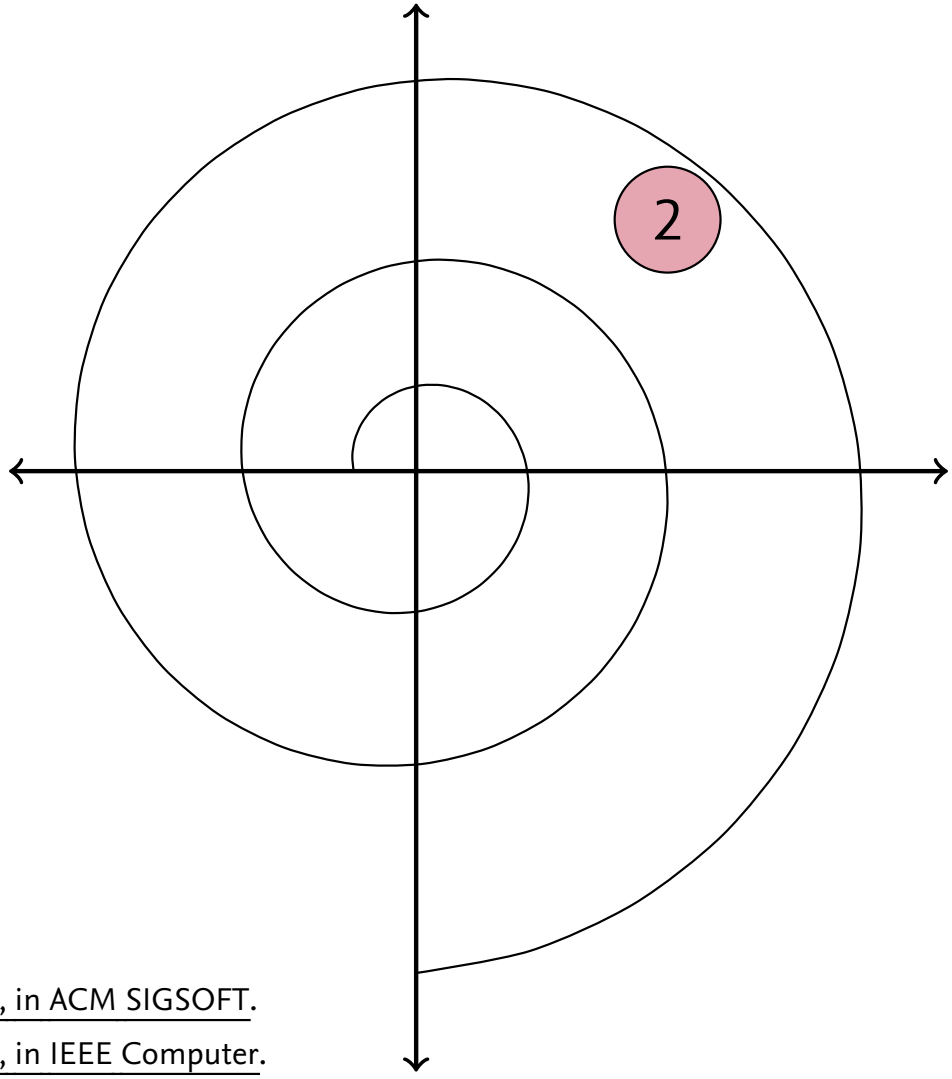
Schritt 1:

- Identifikation der Ziele
- Alternative Möglichkeiten
  - Entwurf A, Entwurf B
  - Wiederverwendung
  - Kauf
- Randbedingungen ausarbeiten
  - Kosten
  - Zeit
  - Schnittstellen

[18] Boehm B.W. A Spiral Model of Software Development and Enhancement, in ACM SIGSOFT. 1986.

[19] Boehm B.W. A Spiral Model of Software Development and Enhancement, in IEEE Computer. 1988.

Spiralmodell[18,19]



Schritt 2:

- Evaluierung der Alternativen
- Bei Risiken: Entwicklung einer Strategie
  - Prototypen
  - Simulationen
  - Benutzerbefragungen

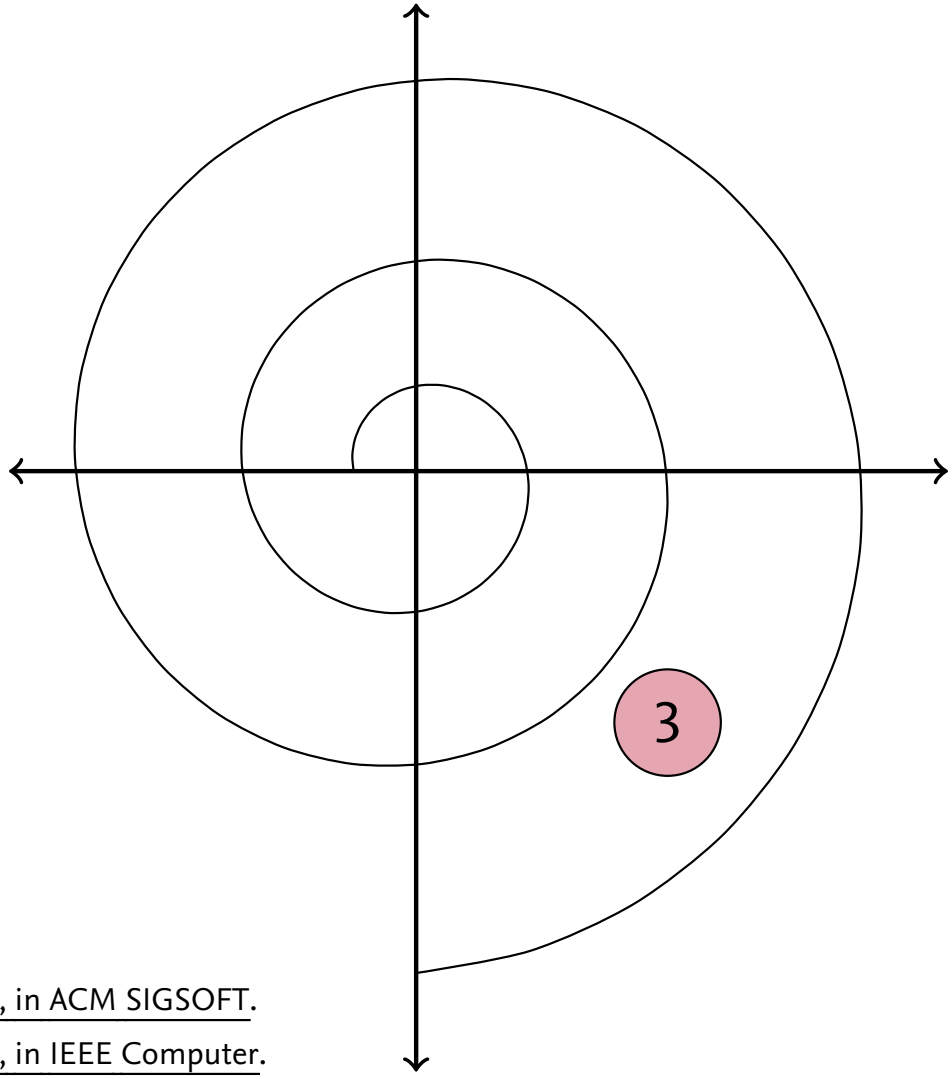
[18] B.W., A Spiral Model of Software Development and Enhancement, in ACM SIGSOFT.

[19] B.W., A Spiral Model of Software Development and Enhancement, in IEEE Computer.

Spiralmodell[18,19]

Schritt 3:

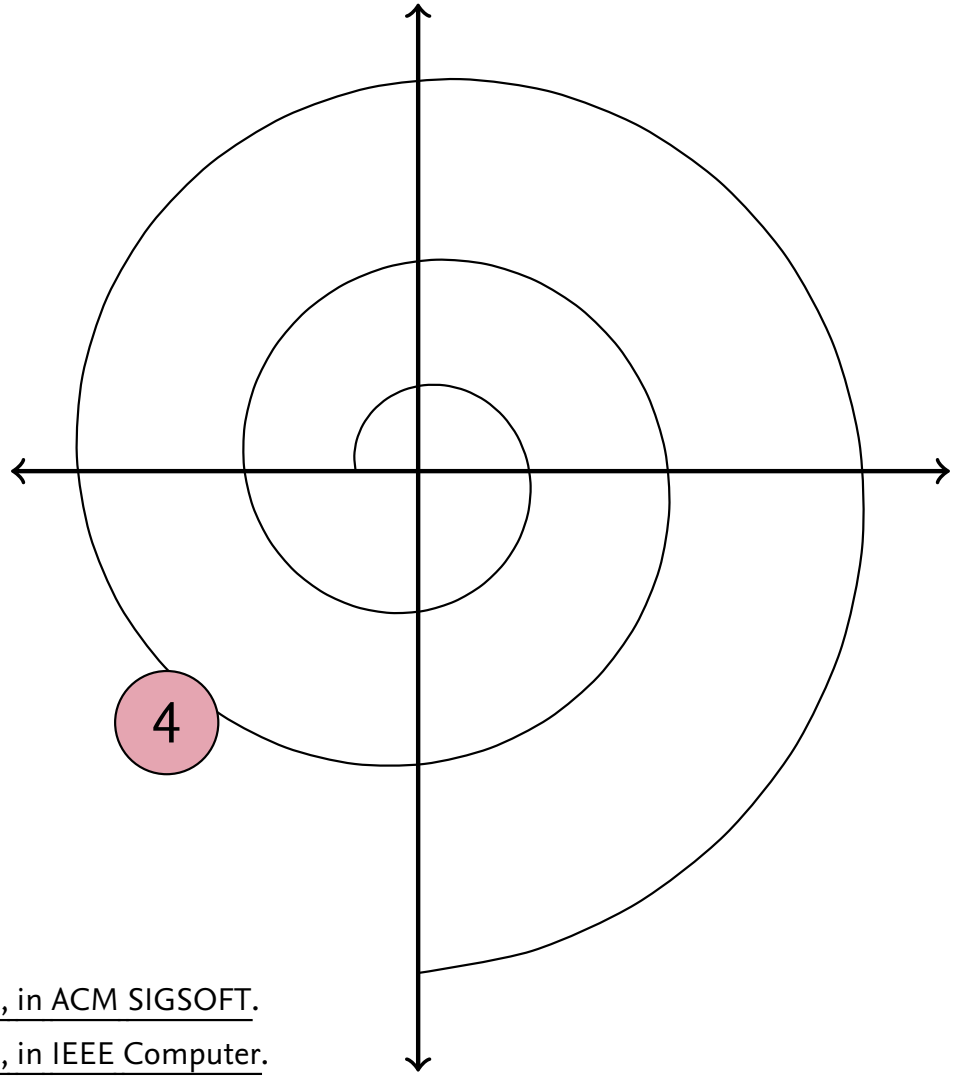
- Risiken $\Rightarrow$ Wahl eines Prozessmodells
  - evolutionäres Modell
  - Prototypenmodell
  - Wasserfall-Modell
  - ...
- Kombinationen denkbar



[18] B.W., A Spiral Model of Software Development and Enhancement, in ACM SIGSOFT.

[19] B.W., A Spiral Model of Software Development and Enhancement, in IEEE Computer.

Spiralmodell[18,19]



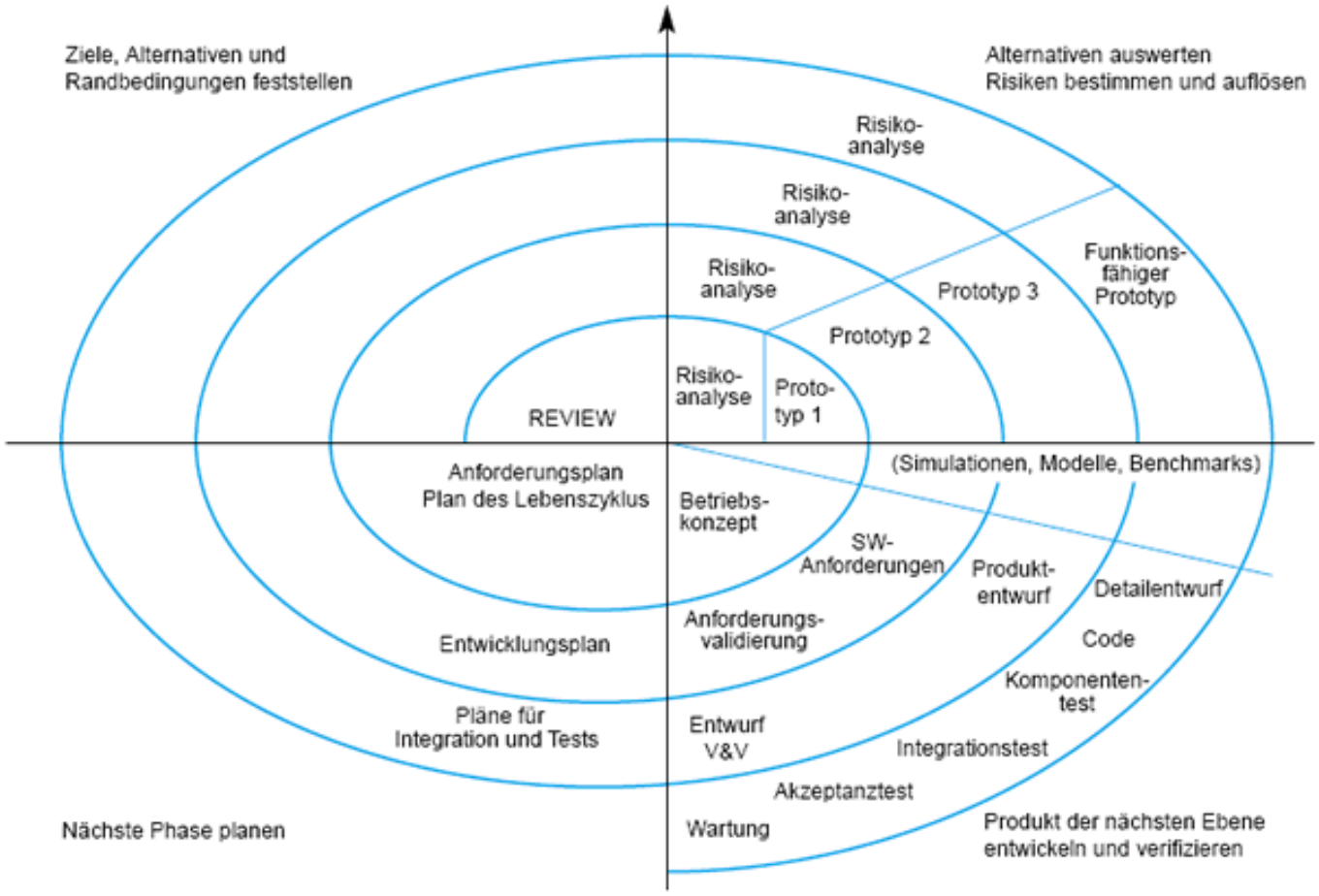
Schritt 4:

- Planung des nächsten Zyklus
  - Ressourcen
  - Aufteilung in Komponenten
- Review der Schritte 1-3
- Einverständnis über den nächsten Schritt

[18] B.W., A Spiral Model of Software Development and Enhancement, in ACM SIGSOFT.

[19] B.W., A Spiral Model of Software Development and Enhancement, in IEEE Computer.

Sprialmodell



Spiralmodell

Vor- und Nachteile[20]

Vorteile:

- Periodische Überprüfung kann ein Abdriften von Zielen und Risiken verhindern
- Je nach Zyklus kann ein geeignetes Prozessmodell ausgewählt werden
- Flexibel
- Erfahrungen können im nächsten Zyklus eingebracht werden

Nachteile:

- Hoher Managementaufwand für kleine und mittlere Projekte
- Risiken müssen identifizierbar sein

[20] Balzert, Lehrbuch der Software-Technik.

Scrum

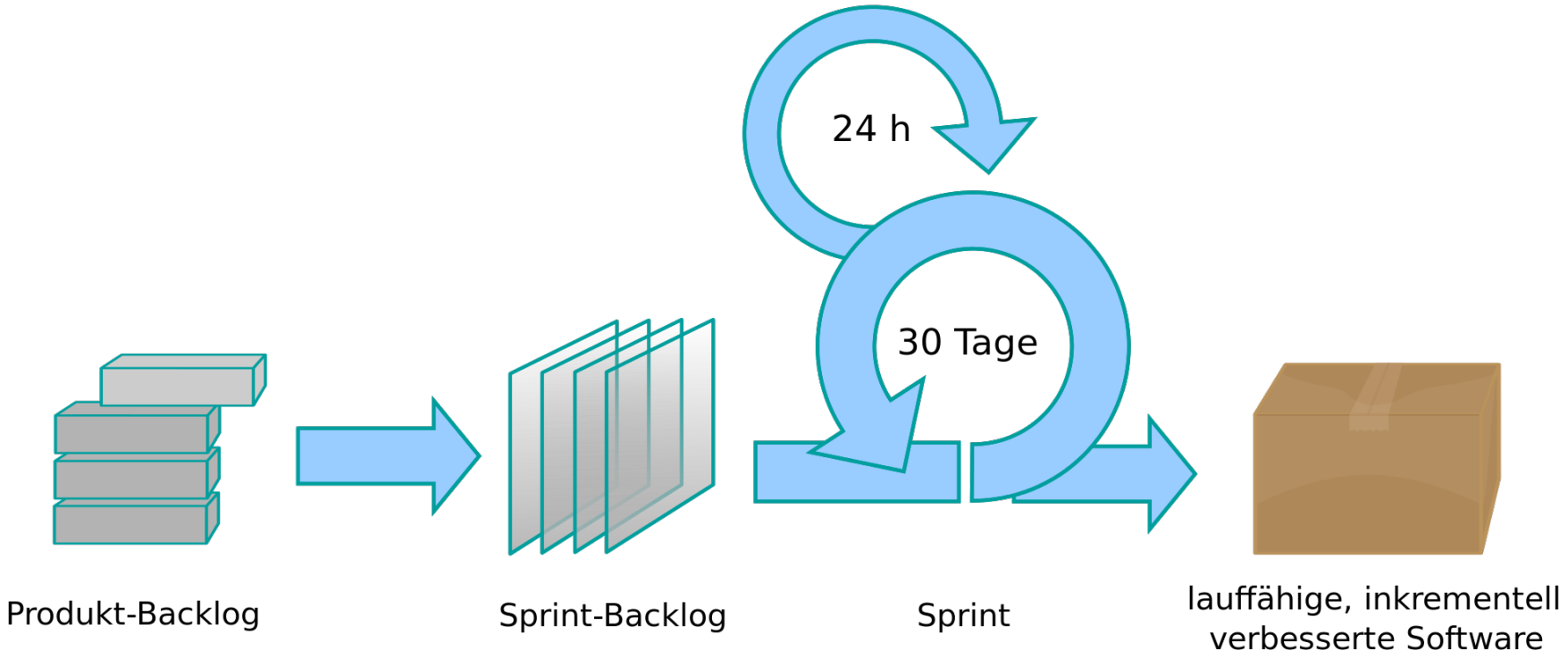


Abbildung 3: Scrum-Prozessverlauf[21]

[21] Von Scrum_process.svg: Lakeworksderivative work: Sebastian Wallroth (talk) - Scrum_process.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=10772971

Scrum

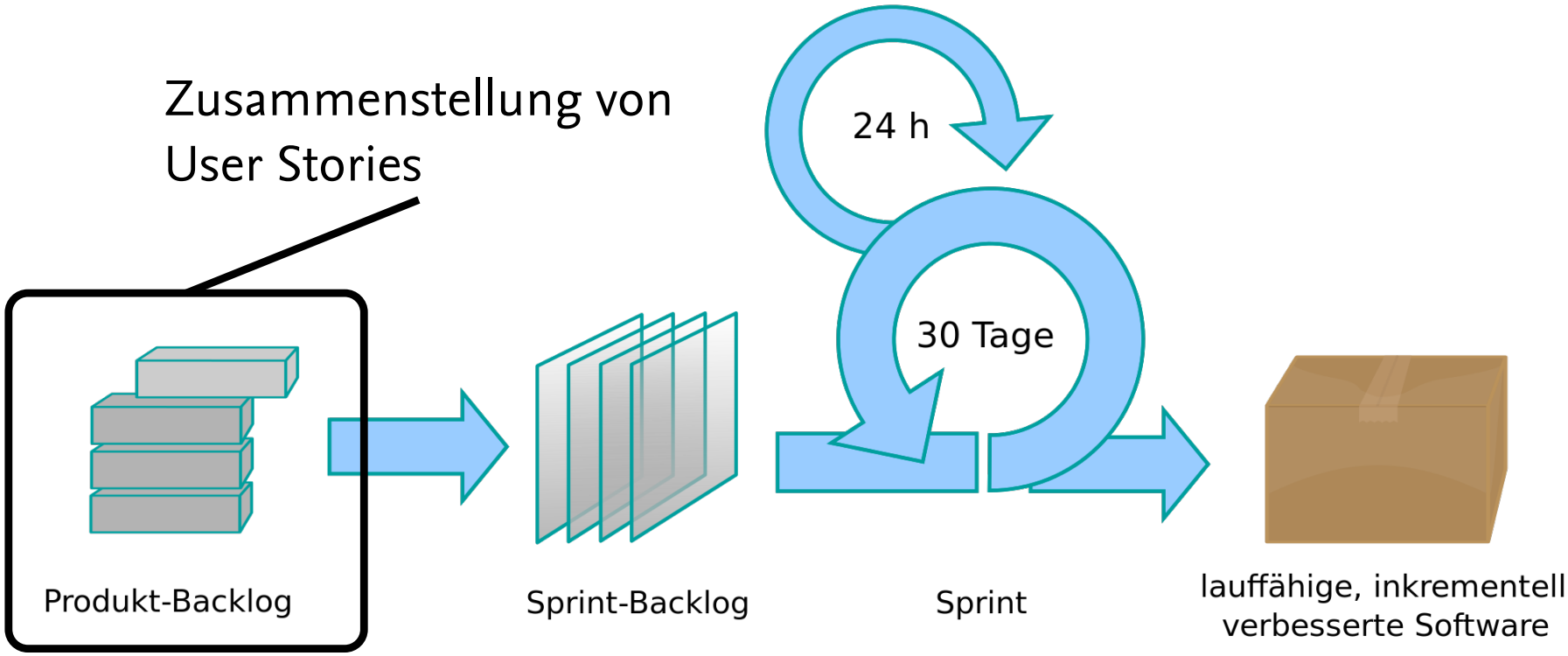


Abbildung 3: Scrum-Prozessverlauf[21]

[21] Von Scrum_process.svg: Lakeworksderivative work: Sebastian Wallroth (talk) - Scrum_process.svg, CC BY-SA 3.0, [illegible]ikimedia.org/w/index.php?curid=10772971

Scrum

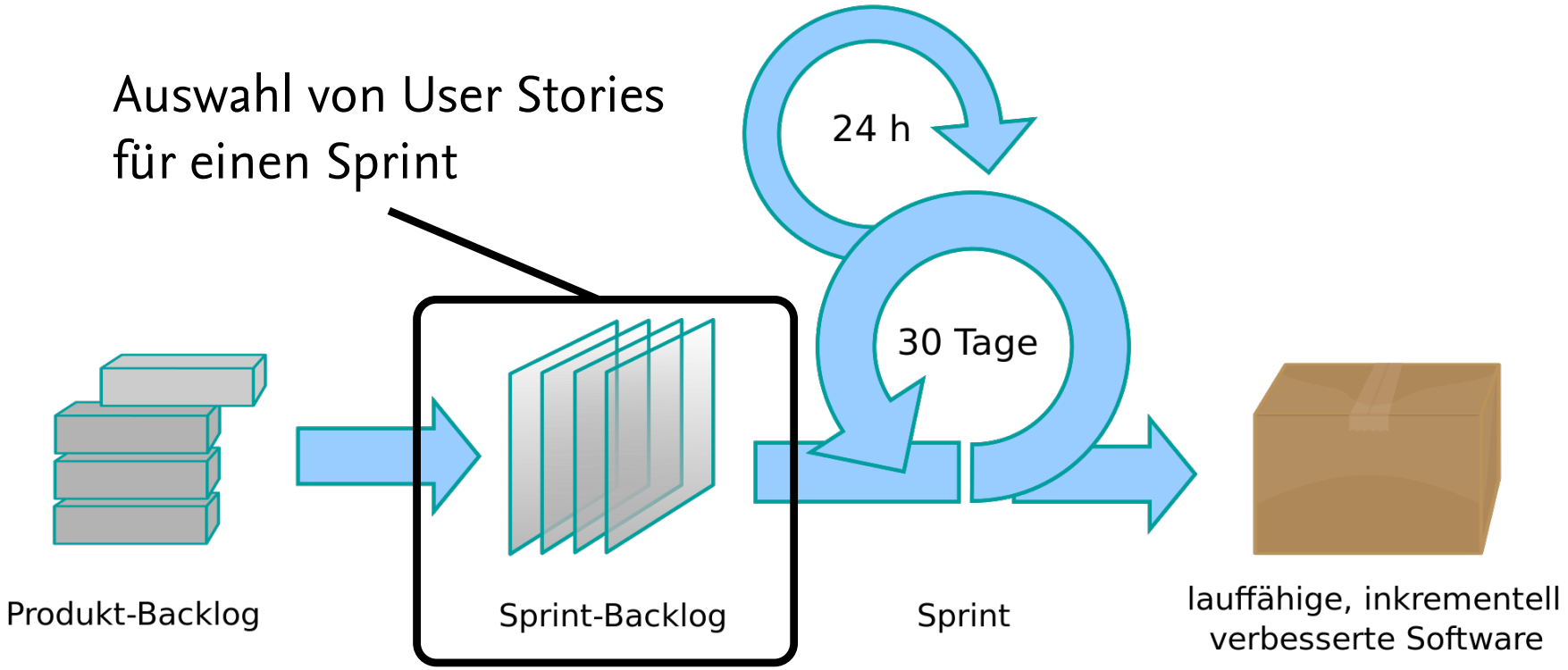


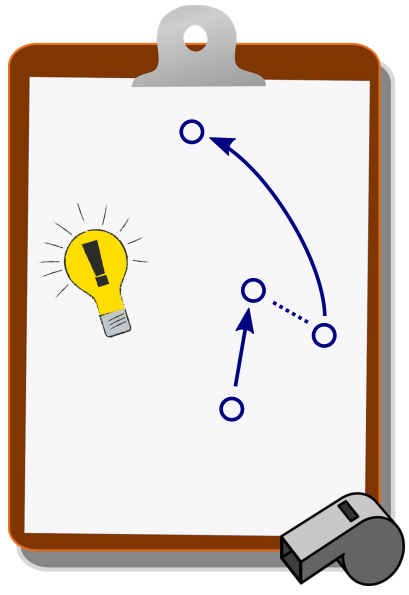
Abbildung 3: Scrum-Prozessverlauf[21]

[21] Von Scrum_process.svg: Lakeworksderivative work: Sebastian Wallroth (talk) - Scrum_process.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=10772971

Akteure: Product owner[22]

Product owner:

- Verkörpert die Projektidee
- Maximiert den Wert des Produktes
- Vermittelt die Vision an das Team
- Stellt das Team zusammen
- Erstellung eines Produkt-Backlogs
- Priorisierung des Produkt-Backlogs

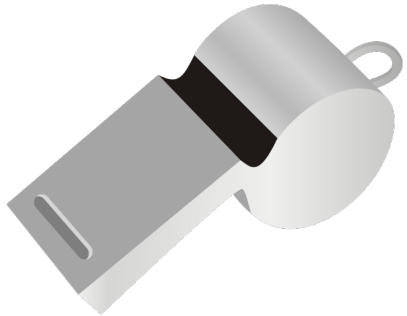


[22] Boris Gloger. Scrum Produkte zuverlässig und schnell entwickeln. 5., überarbeitete Auflage. München: Hanser, 2016. URL: http://ebooks.ciando.com/book/index.cfm/bok_id/2127529.

Akteure: Scrum Master[23]

Scrum Master:

- Treibt den Prozess
- Wahrung und Vermittlung der Scrum-Werte und -Regeln
- Schützt das Team vor Störungen
- Löst von Blockaden
- Vermittelt zwischen Team und Product Owner



[23] Gloger, Scrum Produkte zuverlässig und schnell entwickeln.

Akteure: Entwicklungsteam[24]

Entwicklungsteam (3-9 Personen):

- Selbst organisierend
- Interdisziplinär
- Liefert Produkt
- Arbeitet eigenständig an den User Storys eines Sprints
- Implementiert und testet Anforderungen

[24] Gloger, Scrum Produkte zuverlässig und schnell entwickeln.

Prozessmodell Überblick[25]

Prozessmodell	Primäres Ziel	Antreibendes Moment
Wasserfall-Modell	minimaler Managementaufwand	Dokumente
V-Modell	maximale Qualität	Dokumente
Prototypen-Modell	Risikominimierung	Code
Evolutionäres-Modell	minimale Entwicklungszeit	Code
Inkrementelles-Modell	minimale Entwicklungszeit	Code
Objektorientiertes-Modell	Zeit- und Kostenminimierung	Wiederverwendung
Spiralmodell	Risikominimierung	Risiko

[25] Balzert, Lehrbuch der Software-Technik.

API-Prozessmodell

Ziel

- Risiko minimieren
- Minimale Entwicklungszeit
- Wenig Bürokratie
- Geeignet für Neueinsteiger

Auswahl

- Ein für API angepasstes Spiralmodell
- Software-Entwicklung soll evolutionär oder inkrementell erfolgen

Teil A:

- Ziele für den kommenden Zyklus formulieren

Teil B:

- Risiken formulieren (falls vorhanden)
- Maßnahmen ausarbeiten und ggf. Risiken einbeziehen
- Kleinere Prototypen entwickeln

Teil C:

- Ziele ausarbeiten
  - Feinentwurf und/oder
  - Programmieren und/oder
  - Prototypen entwickeln

Teil D:

- Review des aktuellen Zyklus (Ziele erreicht? Qualität?)
- Planung folgenden Zyklus

API-Spiralmodell — Beispiel

Ziele	Matrikel-Nr., Vorname und Name über Studentenausweis einlesen
Risiken	Daten sind nicht auslesbar
Maßnahmen	a) Prototyp: Testaufbau erstellen b) Auslesen des Studentenausweises testen c) ggf. andere Karte auslesen
Ergebnisse	a) Testaufbau fertiggestellt b) Studentenausweis kann nicht ausgelesen werden c) Auslesen der anderen Karten erfolgt problemlos
Review	*Inhalte hierfür folgen am 16.05.17*
Nächster Zyklus	Verknüpfung zwischen Student und Funduino-Kit

Teil III

Projektarbeit

API-Spiralmodell

Aufgabe 1

1. Erstellen Sie im Wiki Ihres Projekts die Seite **Prozessmodell**
2. Beginnen Sie den ersten von drei Zyklen zu planen und füllen Sie das API-Spiralmodell so weit es geht aus
3. Passen Sie Ihren Zeitplan so an, dass drei Zyklen erarbeitet werden können

GIT Installieren

Aufgabe 2

1. Git herunterladen
   `https://git-scm.com/downloads`
2. Git installieren