Software Management

(Schwerpunkt)

- Zusammenfassung -

Prof. Dr. K.-P. Fähnrich

19.07.2006

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Übersicht der Vorlesung

1. **Grundlagen**
2. **Planung**
3. **Organisation: Gestaltung**
4. **Organisation: Prozess-Modelle**
5. **Personal**
6. **Leitung**
7. **Innovationsmanagement**
8. **Kontrolle: Metriken, Konfigurations- und Änderungsmanagement**
9. **CASE**
10. **Wiederverwendung**
11. **Sanierung**

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Einführung

- Erfolgreiche Software-Erstellung von der Güte des Software-Managements abhängig.

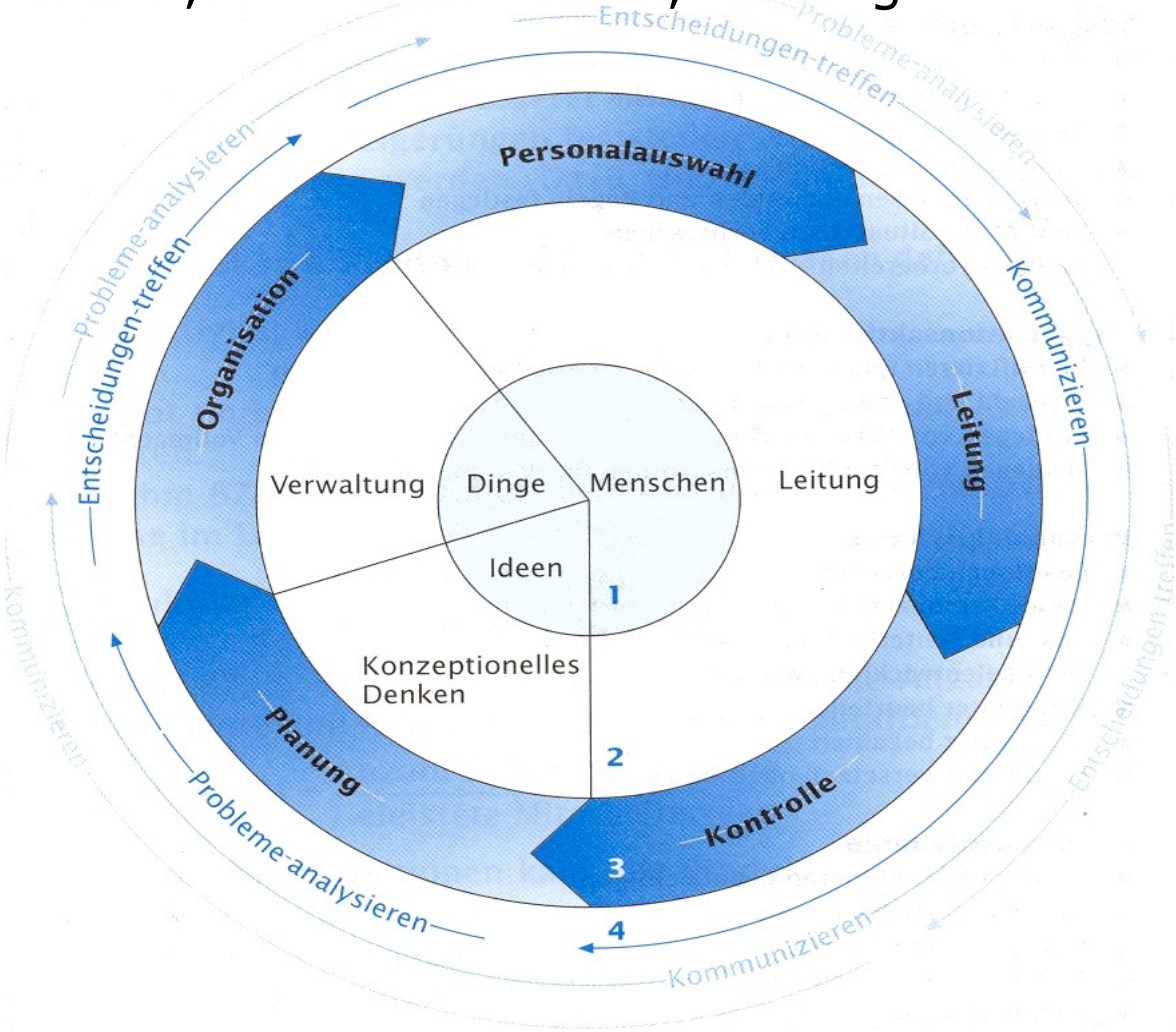
- Ziele des Software-Managements:
  - Erhöhung der Produktivität,
  - Sicherstellung einer definierter Qualitätssicherung, und
  - Senkung der Kosten.

- 3 primäre Managementstrategien (nach [Grady92])
  - Maximierung der Kundenzufriedenheit,
  - Minimierung des Aufwands und der Zeit der Software-Erstellung,
  - Minimierung von Fehlern.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

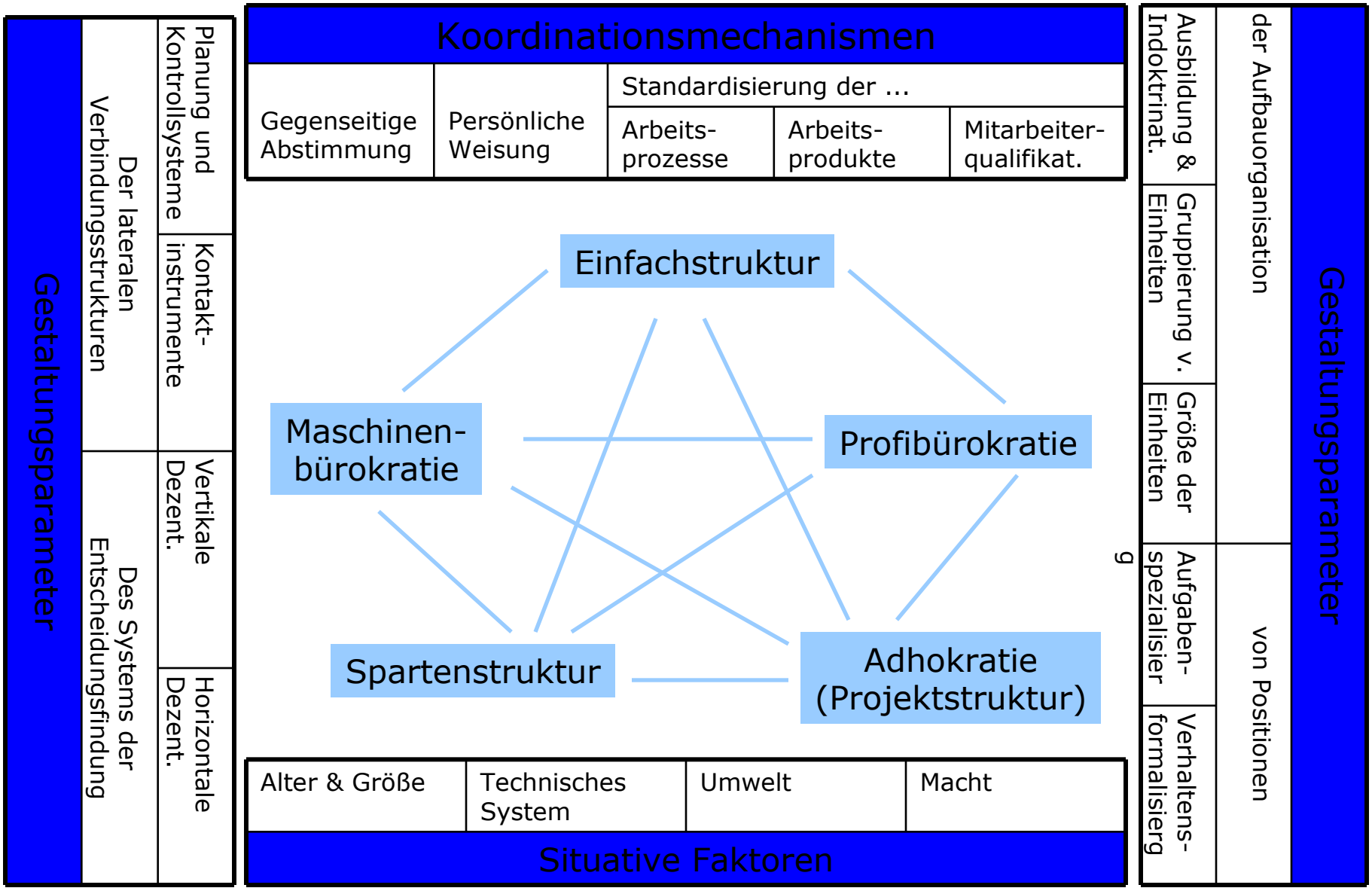
Managementfunktionen

- 5 universelle Managementfunktionen: Planung, Organisation, Personalauswahl, Leitung und Kontrolle.



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Grundlagen



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Hauptaktivitäten des Managements

Planungsaktivitäten

- Ziele setzen,
- Strategien und Taktiken entwickeln,
- Termine festlegen,
- Entscheidungen treffen,
- Vorgehensweisen auswählen und Regeln festlegen,
- Zukünftige Situationen vorhersehen

Organisation-saktivitäten

- Identifizieren und Gruppieren der zu erledigenden Aufgaben,
- Auswahl und Etablierung organisatorischer Strukturen,
- Festlegen von Verantwortungsbereichen und disziplinarischen Vollmachten,
- Festlegen von Qualifikationsprofilen für Positionen

Personalaktivitäten

- Positionen besetzen,
- Neues Personal einstellen und integrieren,
- Aus- und Weiterbildung von Mitarbeitern,
- Personalentwicklung planen,
- Mitarbeiter beurteilen, bezahlen,
- Mitarbeiter versetzen und entlassen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Hauptaktivitäten des Managements

Leitungsaktivitäten

- Mitarbeiter führen und beaufsichtigen,
- Kompetenzen delegieren,
- Mitarbeiter motivieren,
- Aktivitäten koordinieren,
- Kommunikation unterstützen,
- Konflikte lösen,
- Innovationen einführen

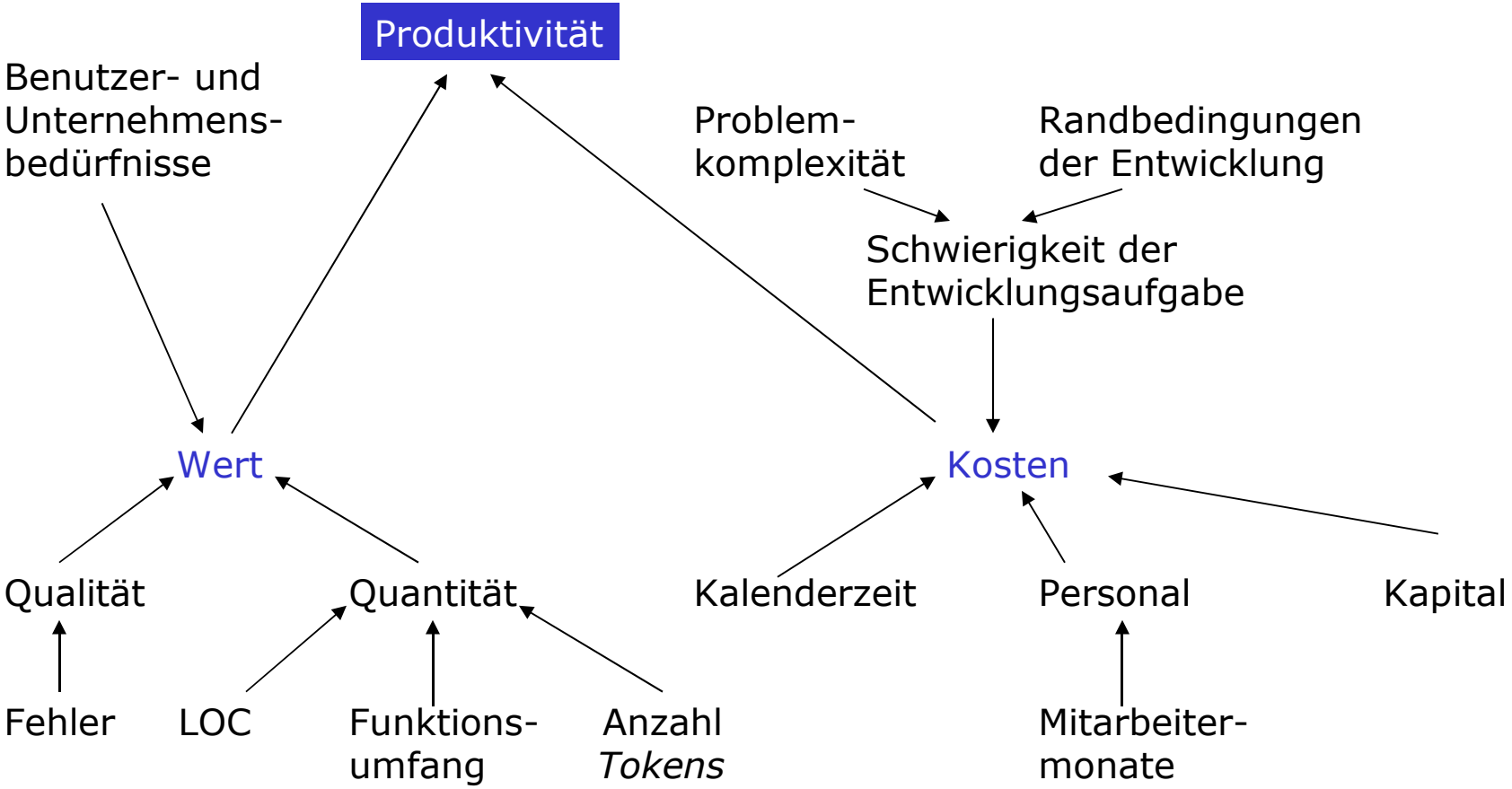
Kontrollaktivitäten

- Prozess- und Produktstandards entwickeln,
- Berichts- und Kontrollwesen etablieren,
- Prozesse und Produkte vermessen,
- Korrekturaktivitäten vermessen,
- Loben und Tadeln

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Produktivität

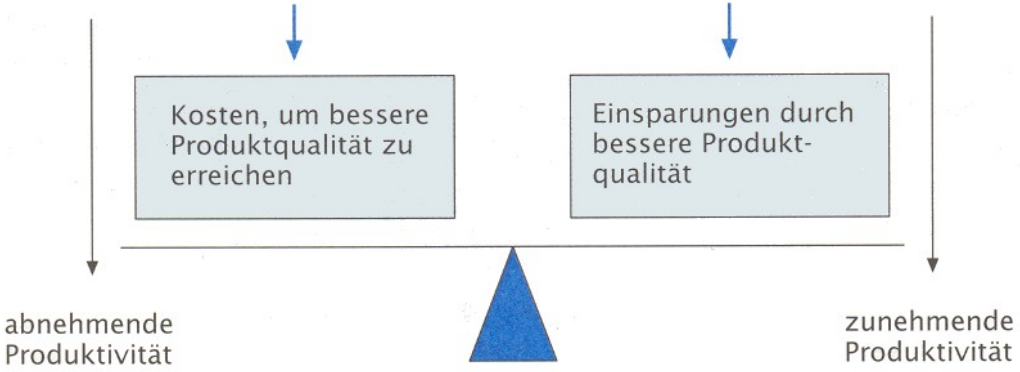
Produktivität und ihre Einflussfaktoren nach Basili



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Produktivität und Qualität

- Produktivität und Qualität beeinflussen sich.
- Zwei konträre Meinungen:
  - Hohe Qualitätsanforderungen verringern die Produktivität.
  - Hohe Qualitätsanforderungen verbessern die Produktivität.



- 2/3 aller Lebenszykluskosten entfallen auf die Wartung und Pflege und nur 1/3 auf die eigentliche Entwicklung.
- Probleme bei den Einsparungen:
  - Wie quantifiziert man Software-Qualität?
  - Wer bezahlt die Wartung?
- Software-Qualität schlecht quantifizierbar und messbar.
- Als pragmatischer Ansatz werden die Wartungskosten ermittelt.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Literaturangaben

[DeMarco,Lister91]
DeMarco T., Lister T., Wien wartet auf Dich! Der Faktor Mensch im DV-Management, 1991

[Grady92]
Grady R.B., Practical Software Metrics for Management and Process Improvement, 1992

[Mackenzie69]
Mackenzie R.A., The management process in 3-D, 1969

[Maxwell,Wassenhove,Dutta96]
Maxwell K.D, Wassenhove L.V., Dutta S., Software Development Productivity of European Space, Military, and Industrial Applications, 1996

[Sneed87]
Sneed H.M., Software-Management, 1987

[Wallmüller90]
Wallmüller E., Software-Qualitätssicherung in der Praxis, 1990

Begleitliteratur: Helmut Balzert, Lehrbuch der Software-Technik
Quelle der Grafiken und Tabellen: Helmut Balzert, Lehrbuch der Software-Technik, wenn nicht anders angegeben

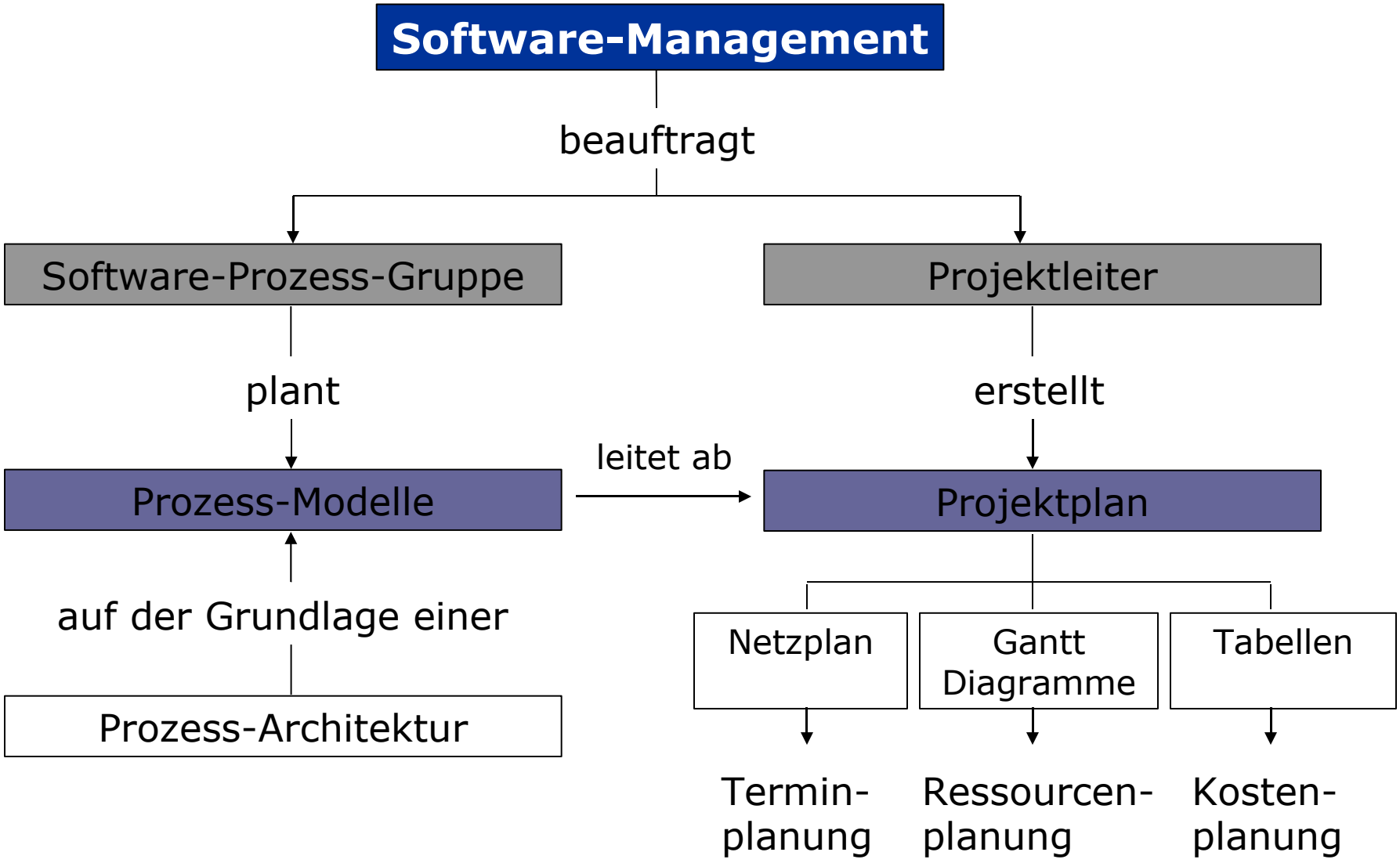
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. Einführung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Planung: Vorbereitung zukünftigen Handelns

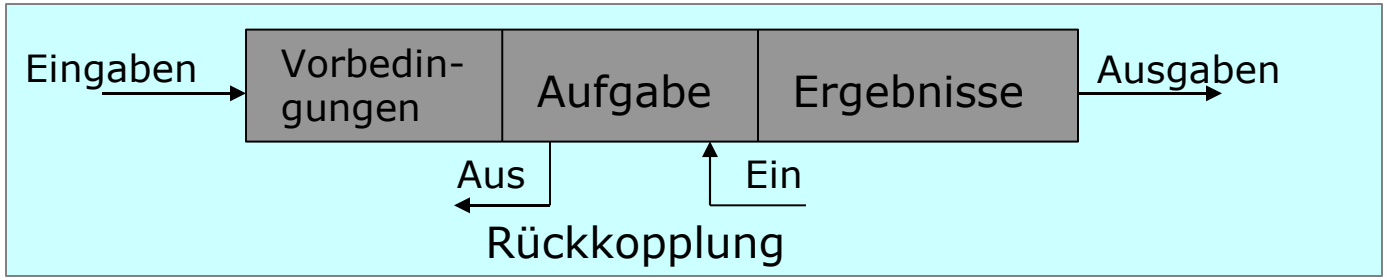
- Ist keine einmalige Angelegenheit.
- Flexible und dynamische Anpassung ist notwendig, wenn sich die Umgebung oder die Entwicklung ändert.

3 Abstraktionsebenen

- **Prozess-Architektur:** Beschreibung der Spezifikation von Software-Entwicklungen, von Standard-Prozesselementen und deren Zusammenwirken.
- **Prozess-Modell:** Festlegung des generellen Vorgehens beim Entwickeln von Software-Produkten.
- **Projektplan:** Instanz eines Prozess-Modells.

Prozess-Architektur

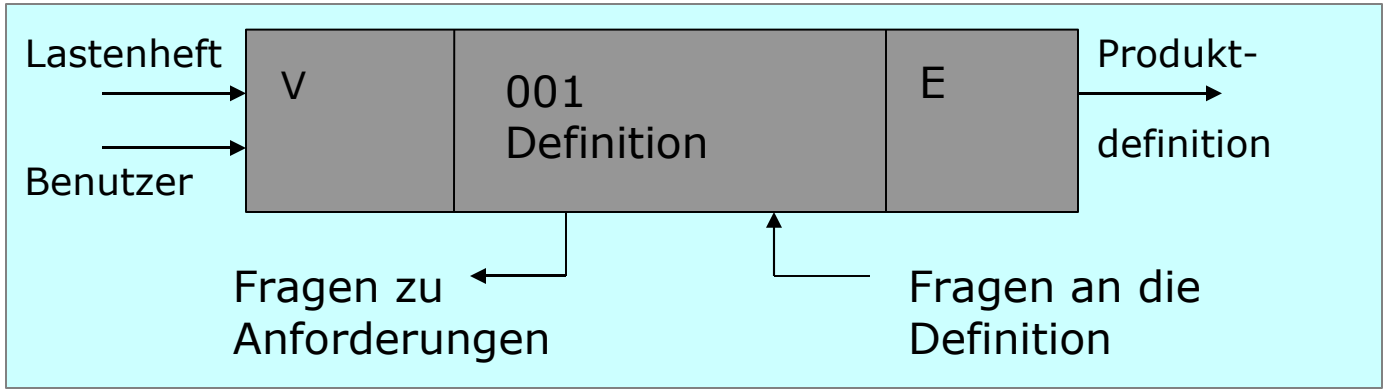
- Spezifikation für den Ablauf der SW-Entwicklung
- besteht aus Standard-Menge von fundamentalen Prozessschritten
- Ein Prozess beschreibt Aktivitäten, Methoden und Verfahren, die zur Software-Entwicklung benötigt werden.



- Bestimmung der Beschreibung von Prozessen und deren Beziehungen durch Regeln.
- Durch geeignetes Zusammenschalten von Standard-Prozesselementen entstehen Prozess-Modelle.

Prozess-Modell

- Prozess-Modelle sind Vorgehens-Modelle
- Prozess-Modell: Ergebnis aus Planung und Prozess-Architektur
- Prozess-Modell: definierbar auf verschiedene Abstraktionsebenen
- Verschiedene Produkt-Klassen bedingen mehrere Prozess-Modelle.



- Prozess-Modell ist Meta-Plan für den Projekt-Plan

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Aufbau von Projektplänen (1)

Projektplan

- verfeinert, konkretisiert und ergänzt ein ausgewähltes Prozess-Modell
- wird projekt- und planspezifisch verfeinert
- Im Prozess zu erledigende Aufgaben werden in Vorgänge unterteilt.

Für jeden **Vorgang** ist festzulegen:

- Name des Vorgangs,
- erforderliche Zeitdauer,
- Zuordnung von Personal- und Betriebsmittel,
- Kosten und Einnahmen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Aufbau von Projektplänen (2)

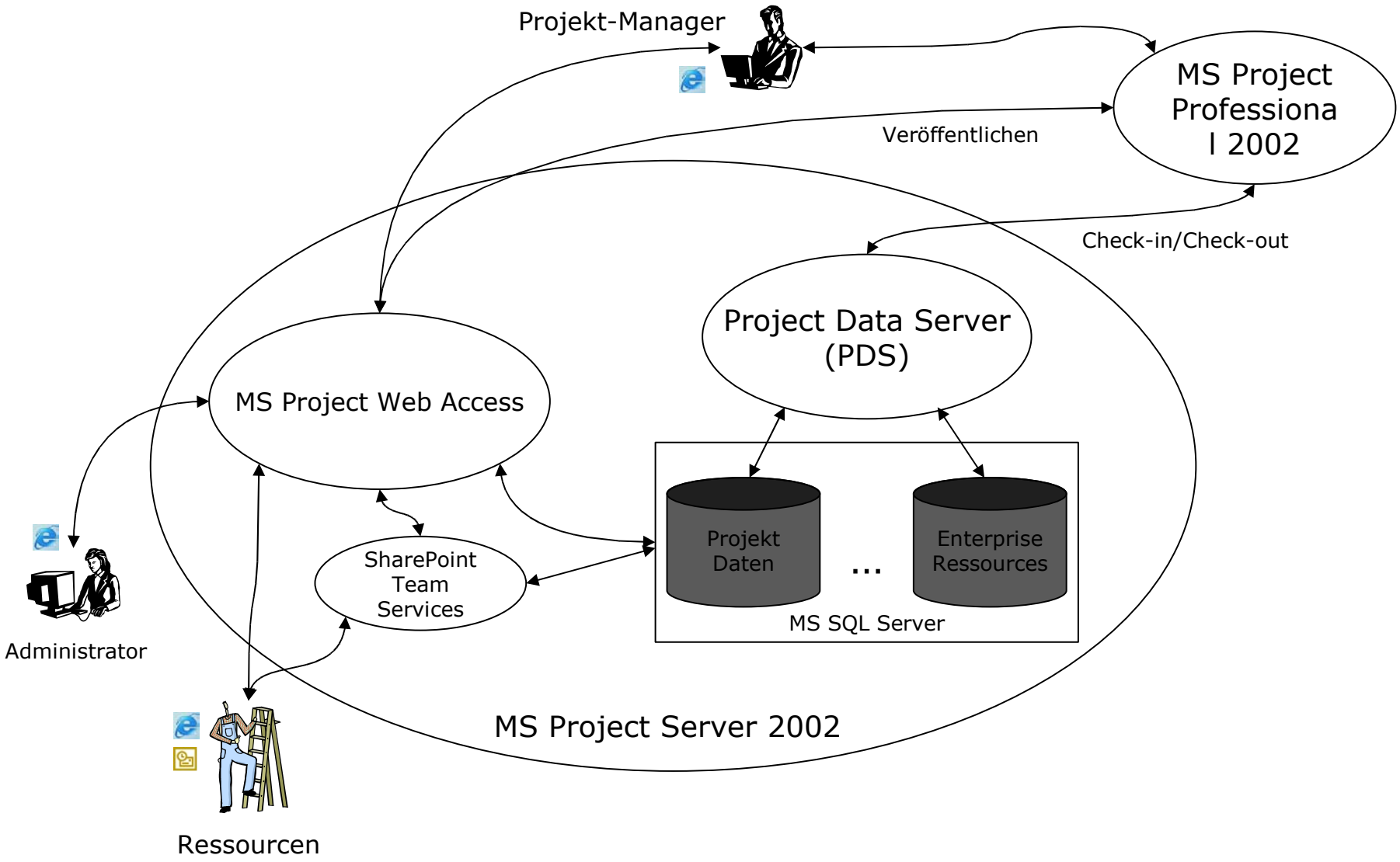
Projektplan

- **Phase**: Zusammenfassung von Vorgängen aus einem globalen Abschnitt
- **Meilenstein(e)**:
  - stellt einen Zeitpunkt dar (Ggs.: Vorgang → Aktivität)
  - dienen der Projektüberwachung
  - können den
    - Beginn und das Ende eines Projekts,
    - den Abschluss jeder Phase und
    - den Abschluss einer Gruppe von Vorgängen einer Phase

    kennzeichnen
  - Müssen folgende Anforderungen erfüllen:
    - Überprüfbarkeit
    - Kurzfristigkeit
    - Gleichverteilung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Architektur



Einsatzmittel

- Werden bei der Durchführung von Vorgängen benötigt.
- Es sind:
  - Personal-,
  - Betriebsmittel und
  - Geldmittel.
- **Ressourcen = Personalmittel + Betriebsmittel**
- **Aufgabe** der Einsatzmittelplanung:
  - Vorhersagen des Bedarfs an Einsatzmitteln,
  - Einsatzoptimierung durch Aufzeigen von Engpässen und Leerläufen,
  - auslastungsoptimale Verteilung der Einsatzmittel auf die einzelnen Vorgänge und Projekte

Einsatzplanung bei Multiprojekten

- Multiprojektplanung ist nötig, wenn sich mehrere Projekte ein bestimmtes Einsatzmittel oder einen beschränkten Vorrat teilen.
- Planabstimmung der vorhandenen Ressourcen mit Prioritätsvergabe erforderlich.
- Wichtige Aspekte:
  - bestimmte Mitarbeiter sollen zeitparallel in mehreren Projekten arbeiten,
  - eine feste Mitarbeiteranzahl steht als Summe für mehrere Projekte zur Verfügung und soll fachgerecht aufgeteilt werden,
  - ein vorgegebenes Budget soll auf die einzelnen Projekte aufgeteilt werden,
  - eine beschränkte Menge eines bestimmten Betriebsmittels soll fair auf mehrere Projekte aufgeteilt werden

Verschiedene Kostenarten

- **Projektkosten** werden bottom-up ermittelt und die **Budgets** top-down von der Geschäftsleitung festgelegt.
- **Gemeinkosten**: Indirekte Kosten, die nicht direkt einem Projekt zugeordnet werden können (Mietkosten, Lohn und Gehalt für Verwaltungspersonal).
- **Fixe Kosten**/**Erlöse**: Einmalige, mit einem Vorgang zusammenhängende Kosten/Erlöse (Prämien)
- **Ressourcenkosten**: Laufende, mit einer Ressource zusammenhängende Kosten (Stundensatz eines MA).
- **Cash-flow**: Kassenzufluss, d.h. Überschuss der einem Unternehmen nach Abzug der Kosten bleibt (Beurteilung d. finanziellen Situation).
- **Budgetierung**: Zweckgebundene Zuweisung von Etats und Ressourcen für einen definierten Zeitraum (Resultat der Aufteilung der Mittel des Wirtschaftsplans auf die Teilbereiche des Unternehmens).

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. Einführung

Ziel des Softwaremanagement : Software-Produkte entwickeln lassen.

- Zwei grundlegende Ziele des SWM in der Organisation:
  - **Festlegung des Arbeitsablauf**: **Ablauforganisation** oder **Prozess-Modell;**
  - **Koordination einzelner Aufgaben**: **Aufbauorganisation.**
- Mit Aufbauorganisation verbunden sind organisatorische Positionen, Verantwortungsbereiche und disziplinarische Vollmachten sowie Qualifikationsprofile.

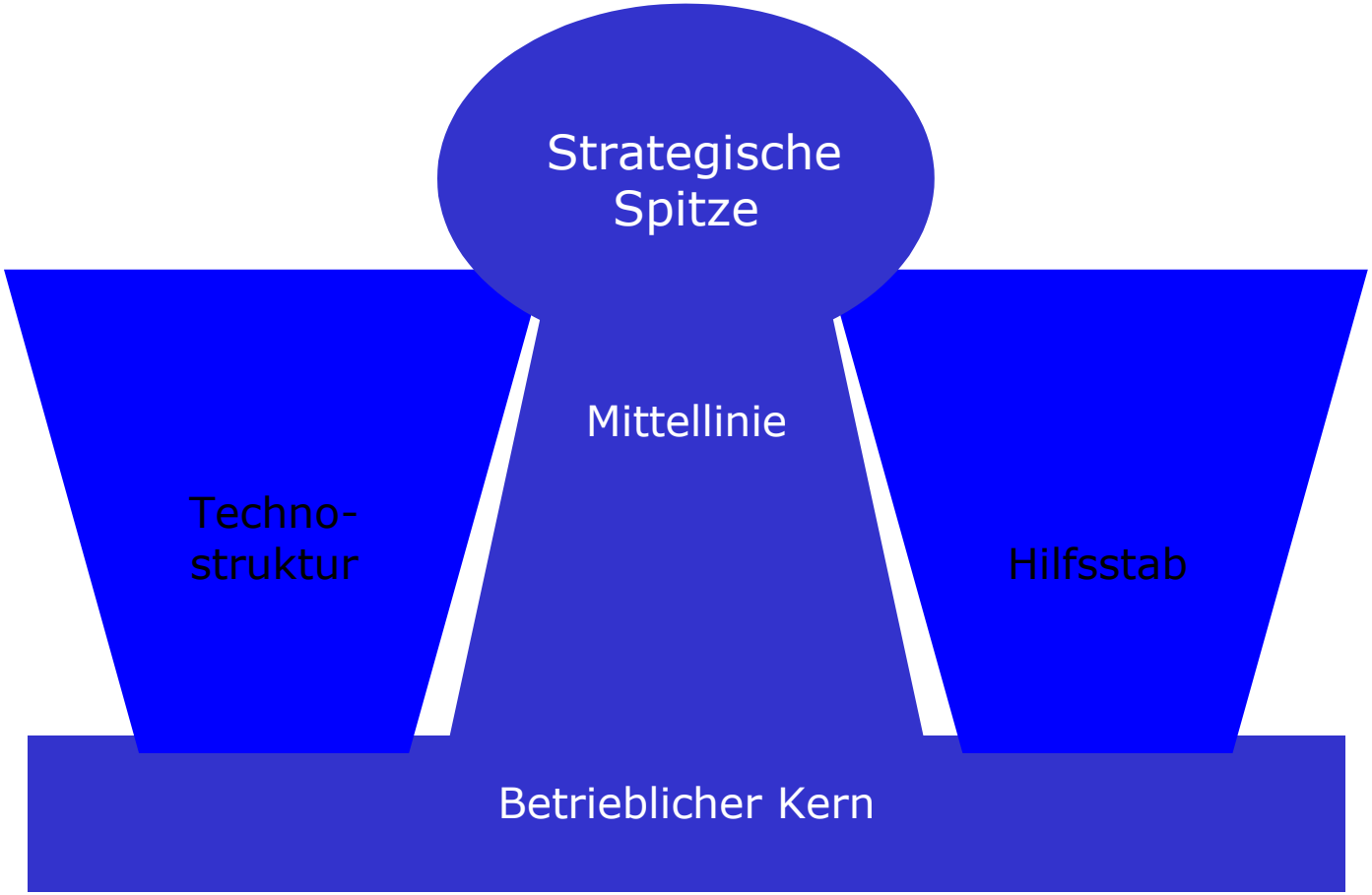
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Grundlagen

- Unterschiede zwischen Software-Management und Management anderer Ingenieurbereiche [Sneed87]:
  - Das Produkt ist immateriell.
  - Der Entwicklungsfortschritt ist objektiv nicht zu ermitteln.
  - Eine Software-Entwicklung verläuft nicht-deterministisch.
  - Es gibt noch kein klares Verständnis vom Entwicklungsprozess.
  - Große Software-Systeme tendieren dazu, einmalige Entwicklungen zu sein.
  - Unteilbarkeit der Arbeit.
  - Die Software-Technik ist keine Naturwissenschaft.
  - Hoher Grad an Abstraktion, bei gleichzeitig niedrigem Grad an Normierung.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.2. Teile einer Organisation



Die fünf Teile einer Organisation

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.2. Teile einer Organisation

- **Betrieblicher Kern**: Hier befinden sich die ausführenden Mitarbeiter.
- **Strategische Spitze**: Auch bekannt als Top Management. Dafür verantwortlich, dass die Organisation ihren Auftrag effektiv erfüllt.
  - 3 Aufgabenbereiche:
    - Persönliche Weisung,
    - Vertretung der Organisation nach außen,
    - Strategische Planung der Organisation.
- **Mittellinie**: Verbindet die strategische Spitze mit dem betrieblichen Kern. Verläuft meistens linear von oben nach unten.
- **Technostruktur**: Wird gebildet von „Analytikern", die für einen Teil der Standardisierung verantwortlich sind. Diese
  - gestalten,
  - planen und
  - verändern den betrieblichen Ablauf.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.2. Teile einer Organisation

- **Hilfsstab**: Unterstützt mit seinen Diensten die Organisation außerhalb des betrieblichen Arbeitslaufs (Telefonzentrale, Kantine, Rechtsabteilung,...).
- „**Mittleres Management**": Führungskräfte, die weder zur strategischen Spitze noch zum betrieblichen Kern gehören.
- „**Linie**": Hierarchie der Führungskräfte von der strategischen Spitze bis zum betrieblichen Kern.
  Linienpositionen sind üblicherweise mit bestimmten formalen Entscheidungsbefugnissen ausgestattet.
- „**Stab**": bezieht sich auf die Technostruktur und den Hilfsstab.
  Stabpositionen sind Leitungshilfestellen mit Unterstützungscharakter.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

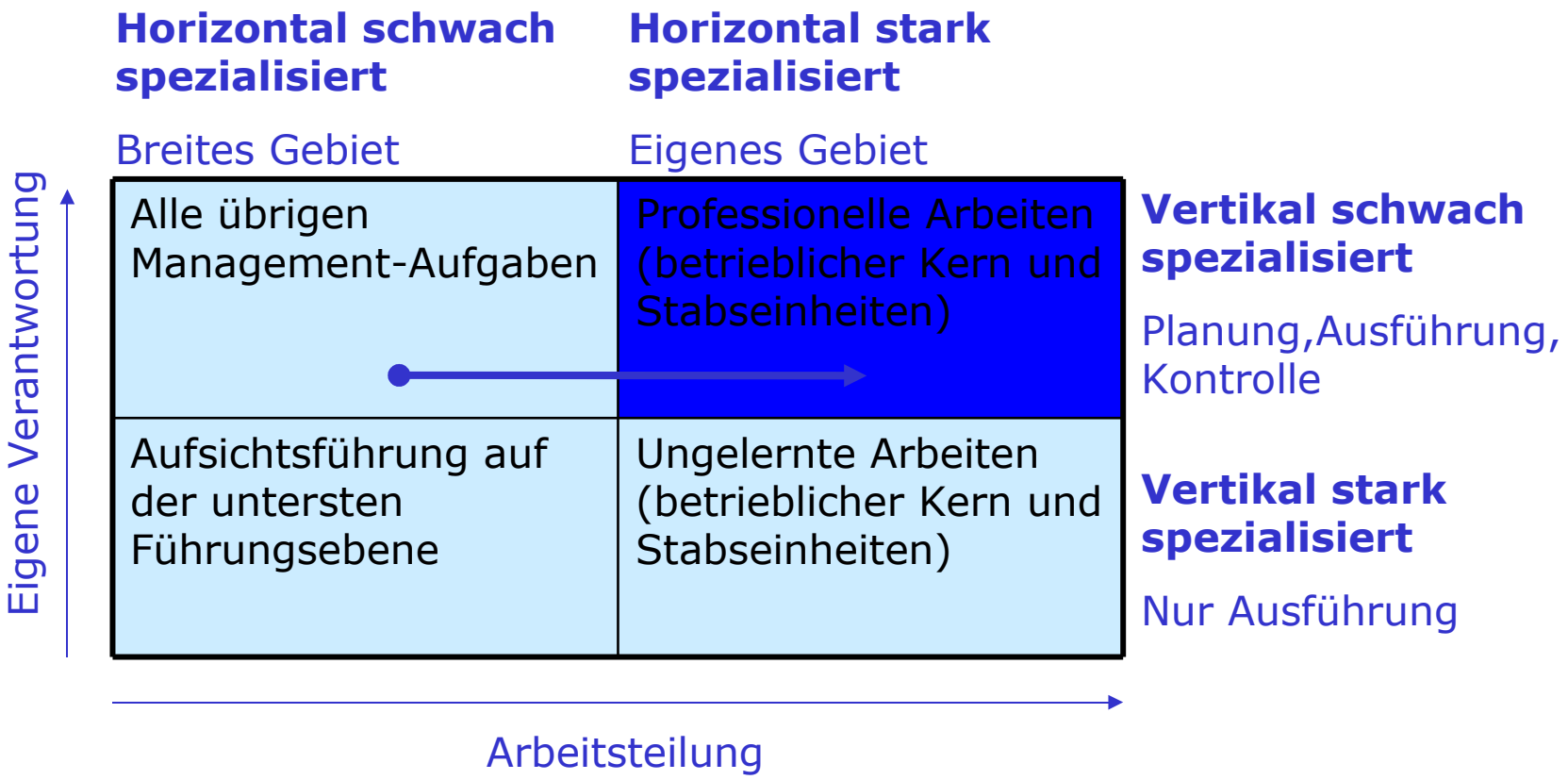
2.3. Gestaltung von Positionen

- **Positionsgestaltung** hängt von der Aufgabenspezialisierung, der Verhaltensformalisierung bei der Arbeit sowie der dazu erforderlichen Ausbildung und Indoktrination ab.
  - **Aufgabenspezialisierung**: Vertikale und Horizontale Spezialisierung von Arbeitsbereichen.
  - **Formalisierung von Verhaltensweisen**: Verhaltensvorschriften zur Einschränkung der Entscheidungsfreiheit von Mitarbeitern.
  - **Indoktrination**: Erwerb organisatorischer Normen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

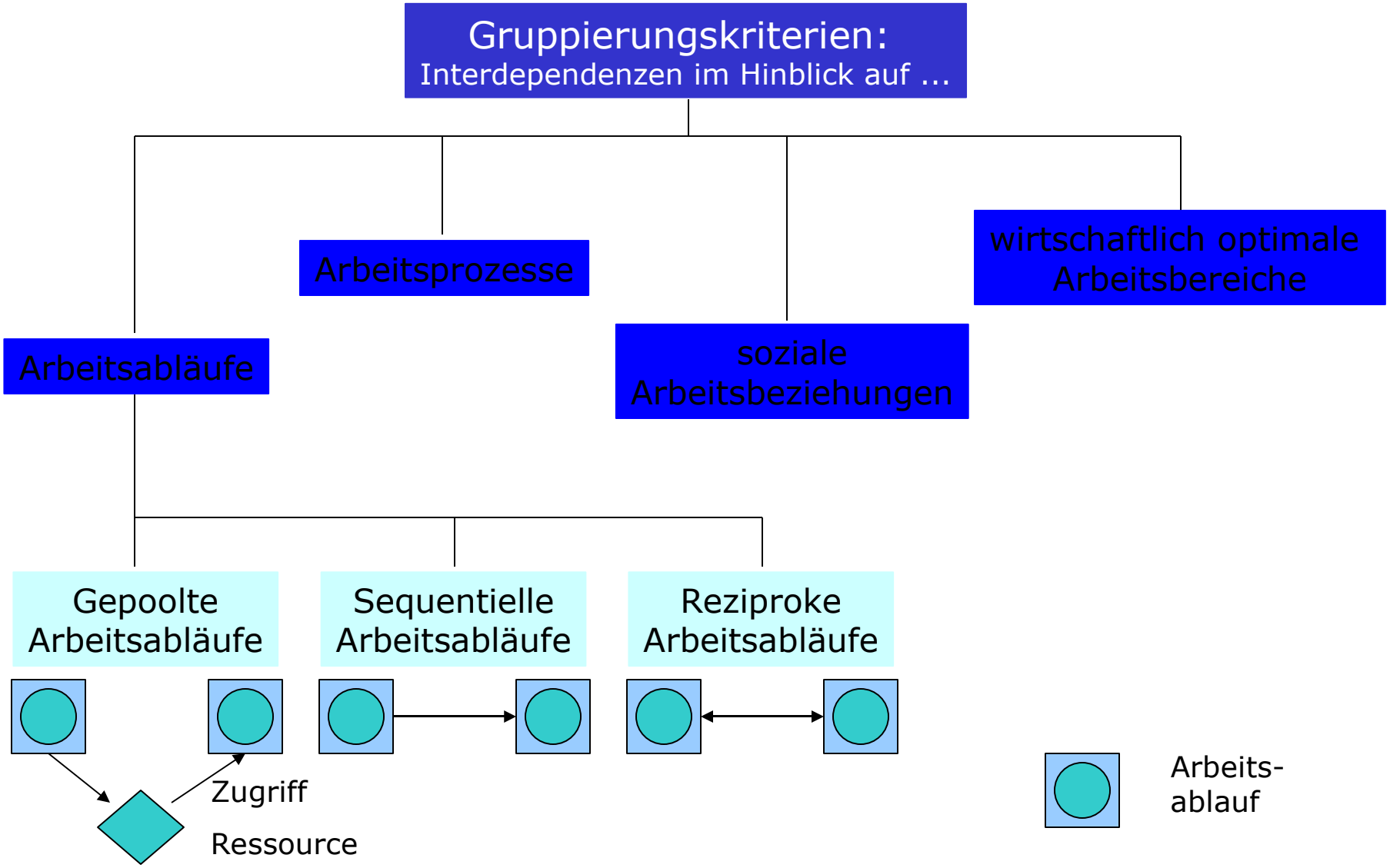
2.3. Gestaltung von Positionen

Dimensionen der Aufgabenspezialisierung



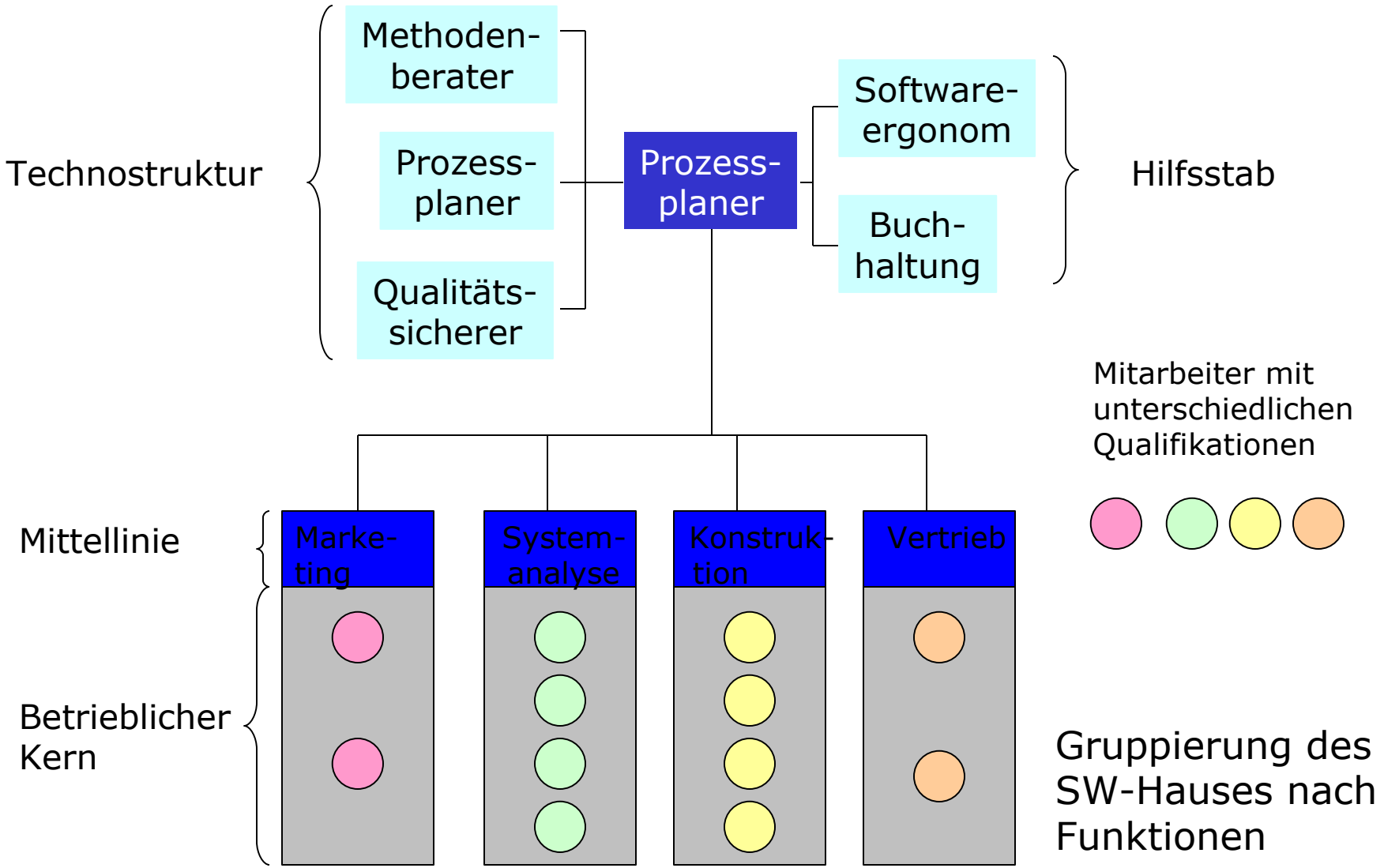
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Gestaltung der Aufbauorganisation



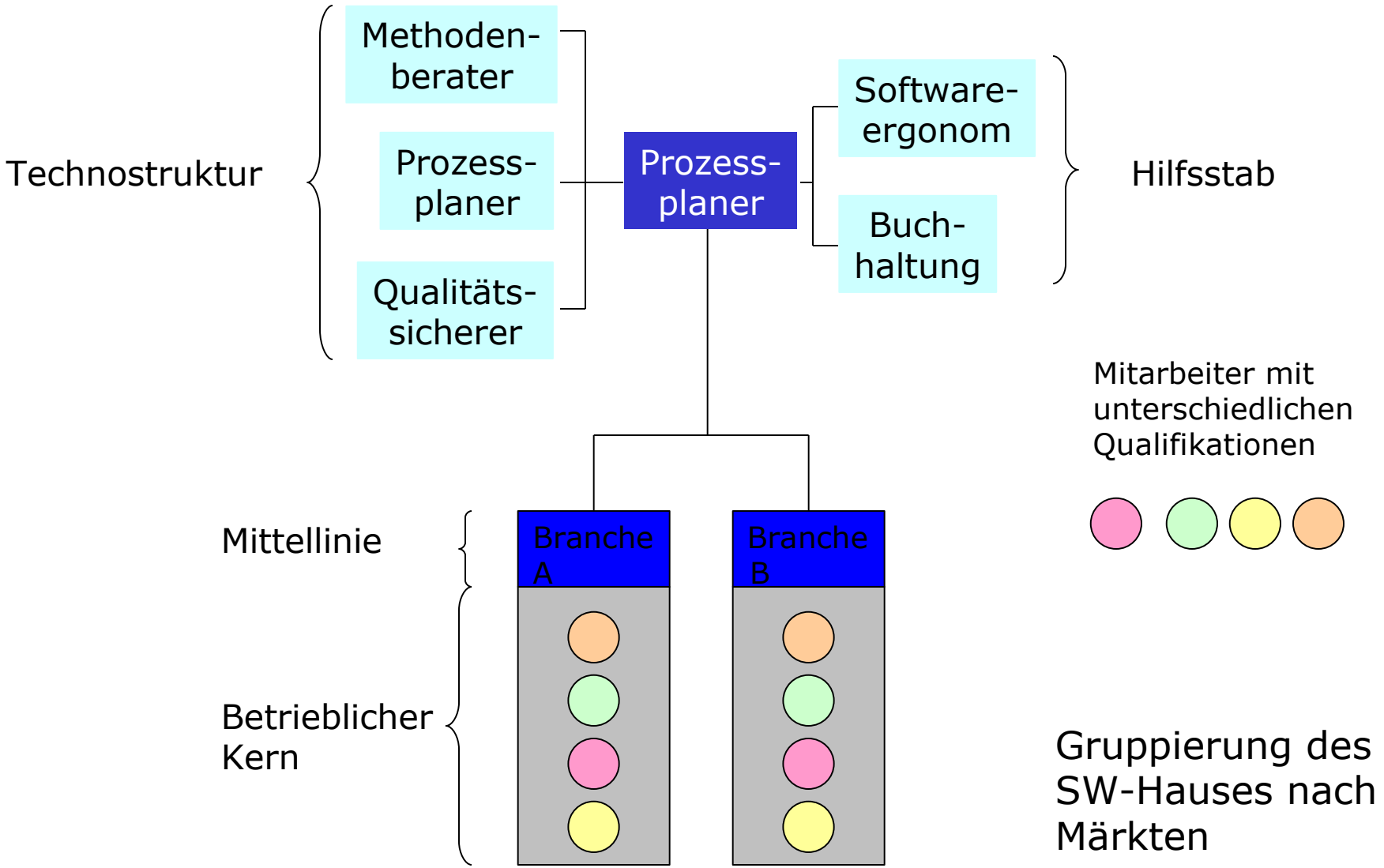
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Gestaltung der Aufbauorganisation



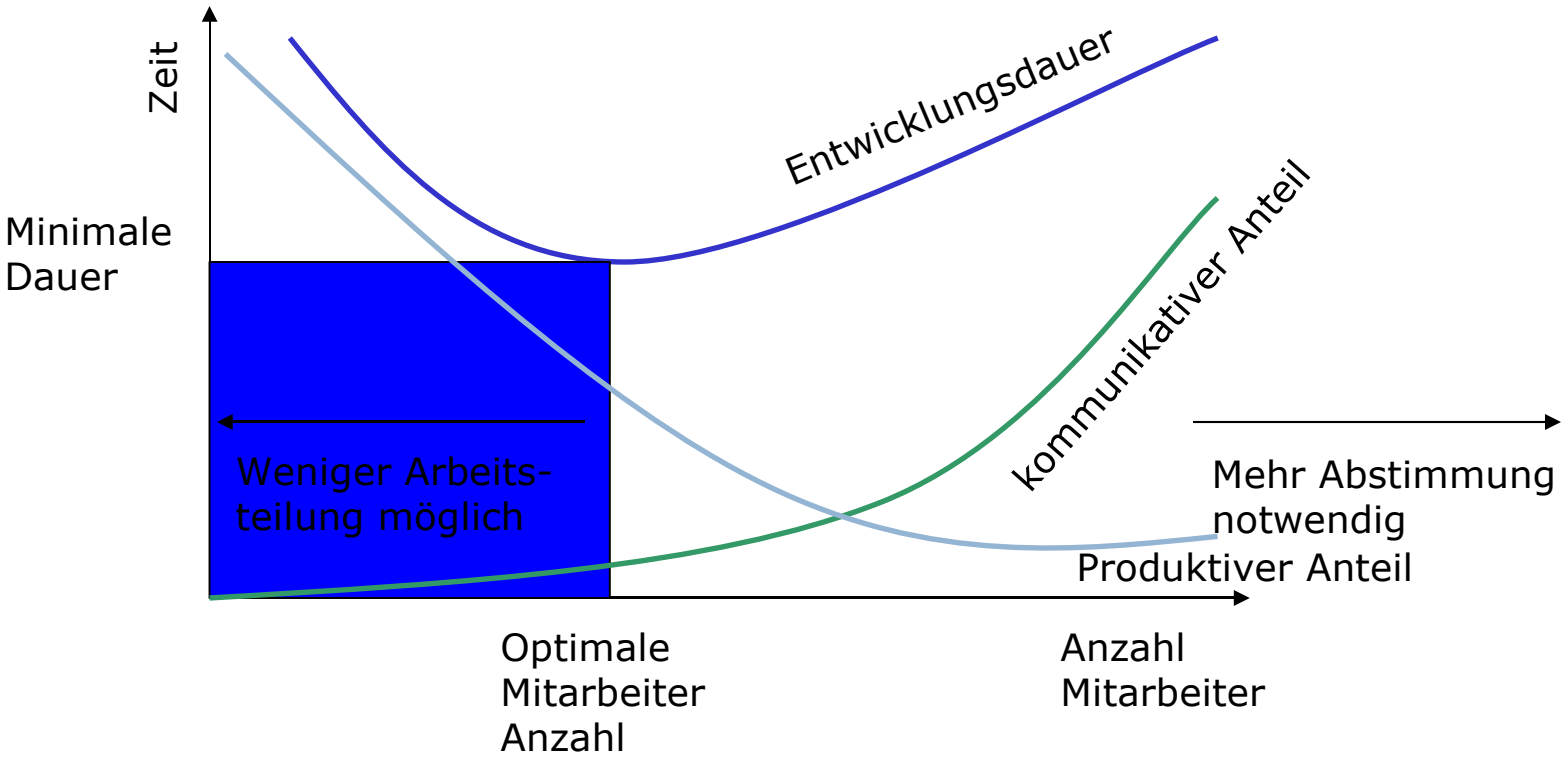
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Gestaltung der Aufbauorganisation



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.4. Gestaltung der Aufbauorganisation



Einfluss des Kommunikationsaufwand

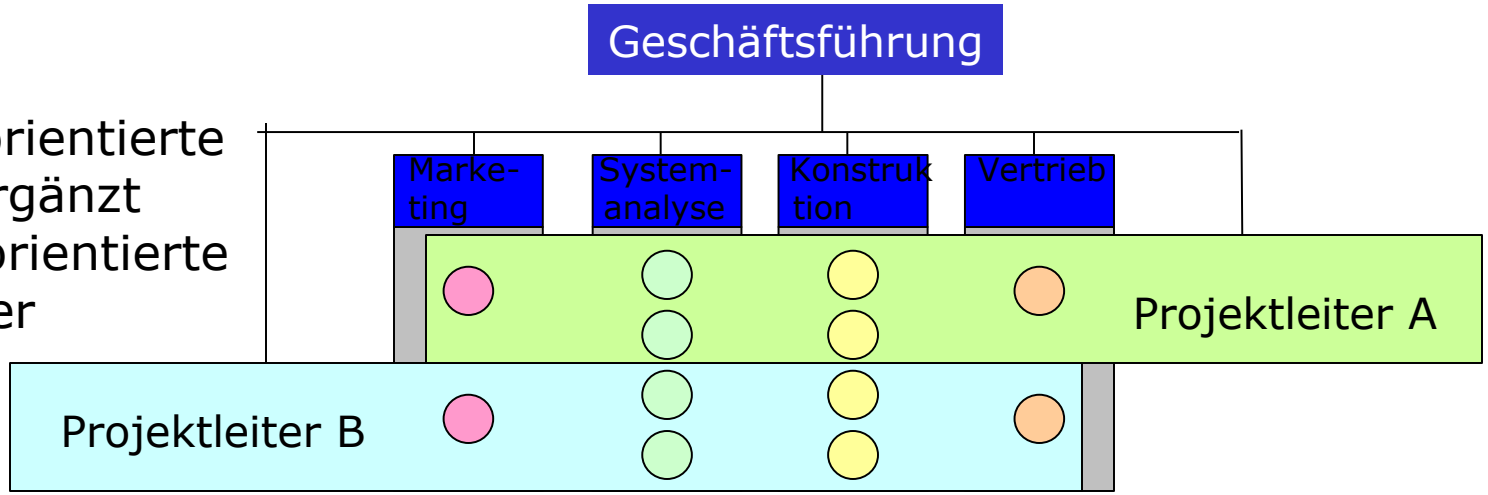
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Projektleiter und Matrixstrukturen

Projektleiter

- Zunächst unbeteiligte Führungskraft, die Teile übernimmt, die zuvor von den verschiedenen Einheiten wahrgenommen wurden.
- Erhält formale Machtbefugnisse (meist fachliche Entscheidung).
- Marktorientierte Projektleiter können funktionsorientierte Strukturen vorangestellt werden und umgekehrt.

Funktionsorientierte Struktur ergänzt um marktorientierte Projektleiter



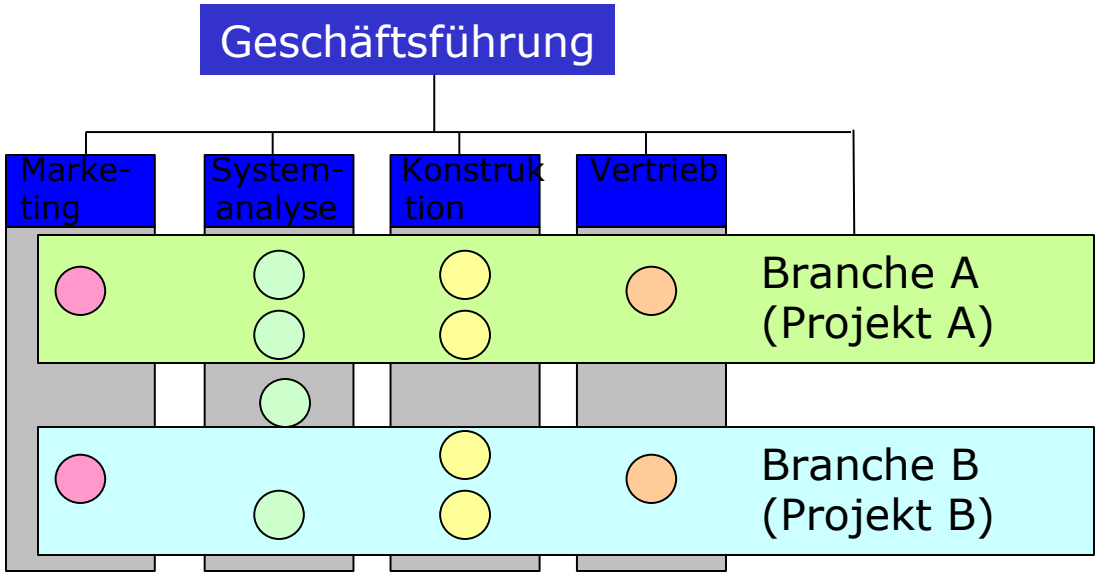
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2.5. Projektleiter und Matrixstrukturen

Matrixstruktur

- Behält beide Gruppierungsalternativen (funktions- und marktorientiert) bei.
- Zwei Arten von Matrixstrukturen:
  - Permanente Matrixstrukturen
  - Projektorientierte Matrixstrukturen

Permanente oder projektorientierte Matrixstruktur



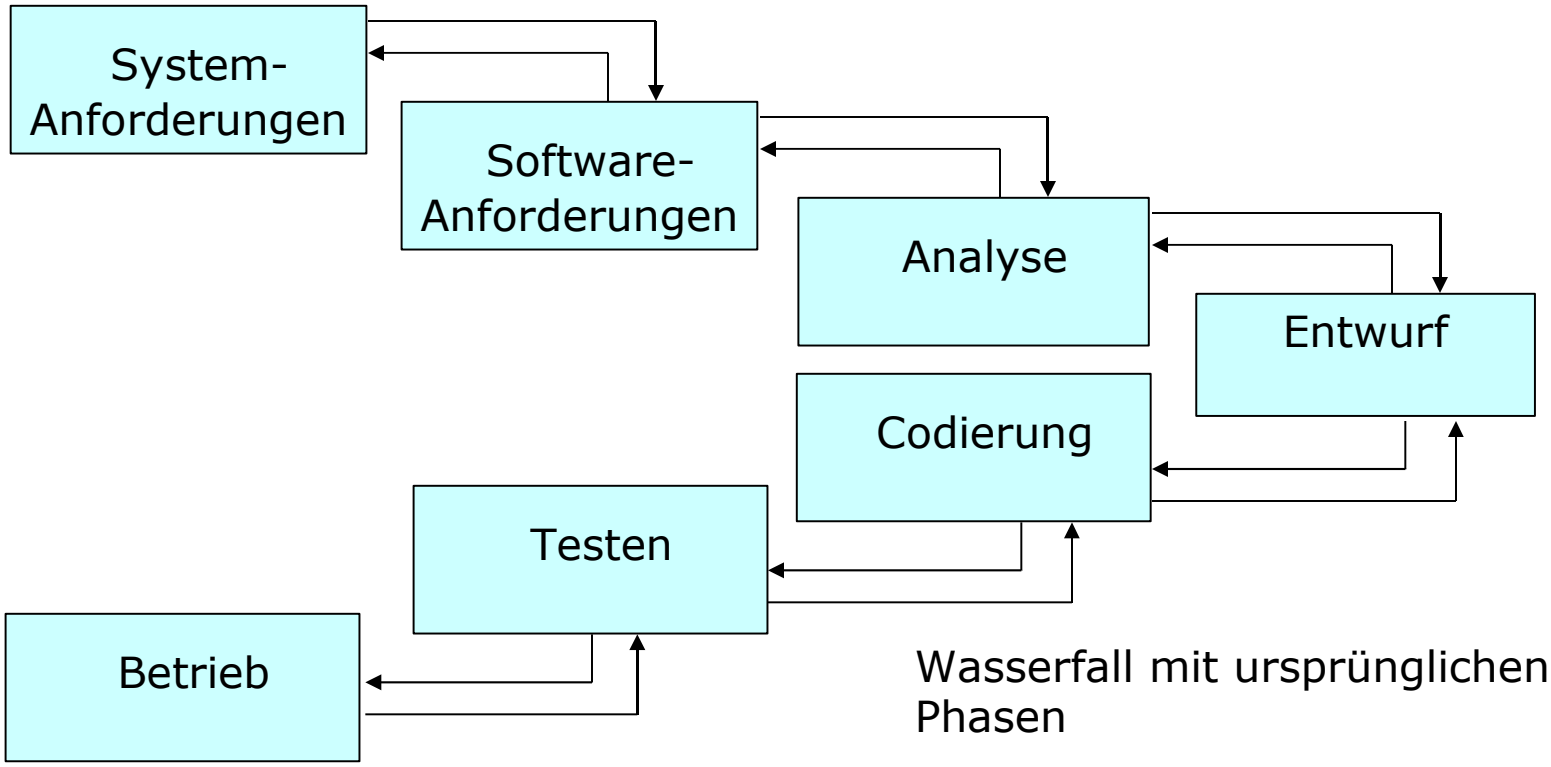
Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Das Wasserfall-Modell

- **Wasserfallmodell**: Weiterentwicklung des „stagewise models" [Benington 56].
- Wasserfallmodell von [Royce 70] erweitert das „stagewise model" um Rückkopplungsschleifen zwischen den Stufen.



Wasserfall mit ursprünglichen Phasen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Das V-Modell

- Legt die Aktivitäten und Produkte des Entwicklungs- und Pflegeprozesses fest.
- Produktzustände und logische Abhängigkeiten zwischen Aktivitäten und Produkten werden dargestellt.
- Vier Submodelle:
  - Systemerstellung (SE),
  - Qualitätssicherung (QS),
  - Konfigurationsmanagement (KM),
  - Projektmanagement (PM).

➔ Erzeugnisstruktur des V-Modells

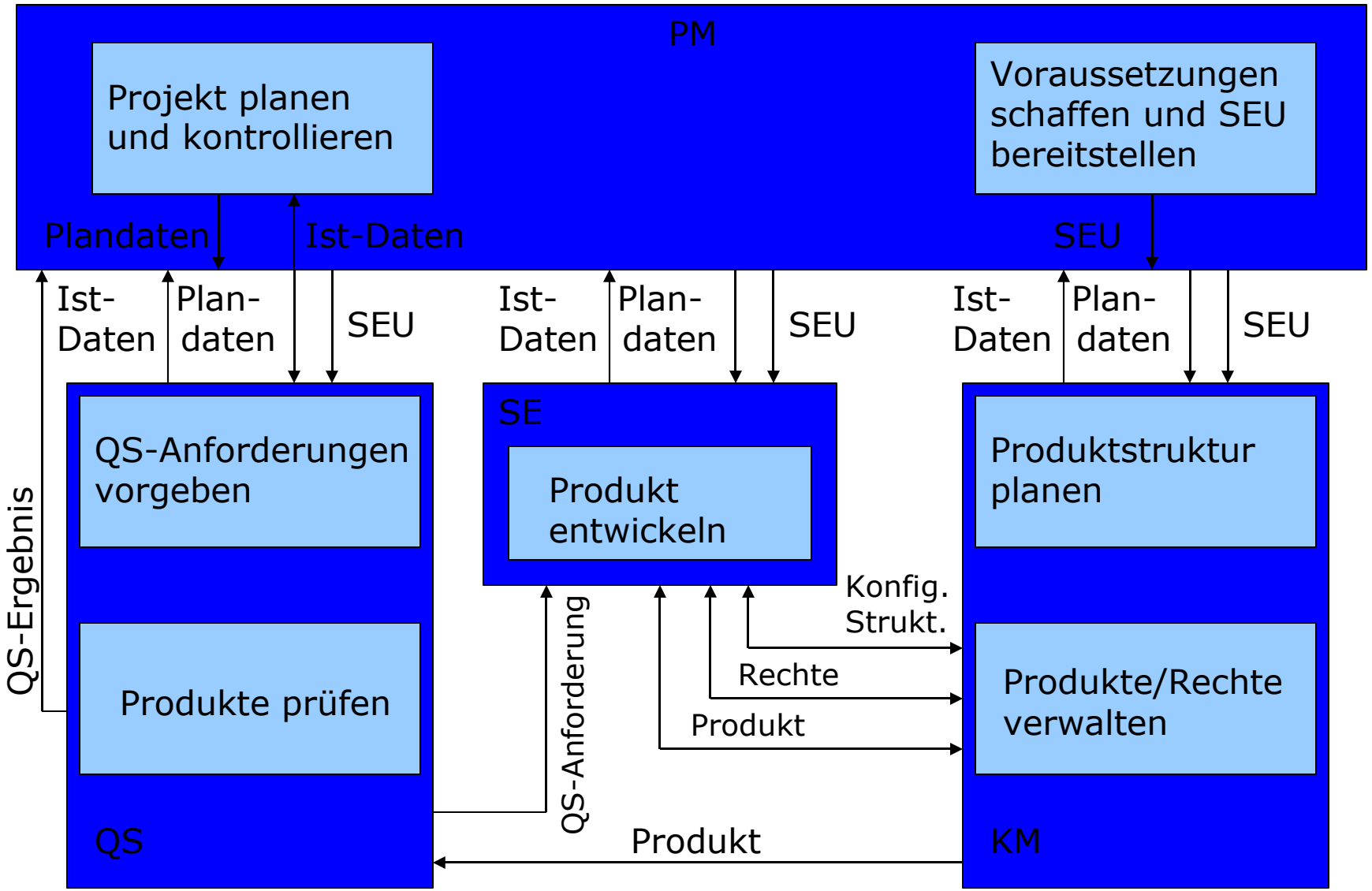
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Das V-Modell

- Grundelemente des V-Modells sind Aktivitäten und Produkte.
- **Aktivität**: Tätigkeit, die bezogen auf ihr Ergebnis und ihre Durchführung genau beschrieben werden kann.
- **Produkt**: Ergebnis bzw. Bearbeitungsgegenstand einer Aktivität.
- Ziel einer Aktivität:
  - Erstellung eines Produkts,
  - Änderung des Zustands eines Produkts,
  - Änderung des Inhalts eines Produkts.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Das V-Modell



Rollen im V-Modell

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

	Manager	**Verantwortliche**	**Durchführende**
PM	Projektmanager	Projektleiter Rechtsverantwortlicher Controller Projektleiter	Projektadministrator
SE	Projektmanager IT-Beauftragter Anwender	Projektleiter	Systemanalytiker Systemdesigner SW-Entwickler HW-Entwickler Technischer Büro SEU-Betreuer Datenadministrator IT-Sicherheitsbeauftragter Datenschutzbeauftragter Systembetreuer
QS	Q-Manager	QS-Verantwortlicher	Prüfer
KM	KM-Manager	KM-Verantwortlicher	KM-Administrator

Rollen im V-Modell

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

V-Modell XT - Überblick

- **V-Modell XT - das Vorgehensmodell für IT-Projekte**
- **Nachfolger des V-Modells 92 und 97**
- **Standard in der Bundesverwaltung**
- **Für Auftraggeber, Auftragnehmer und Projektorganisationen nutzbar**
- **Ca. 700 Seiten Umfang, liegt als HTML und PDF vor**

V-Modell XT - Historie

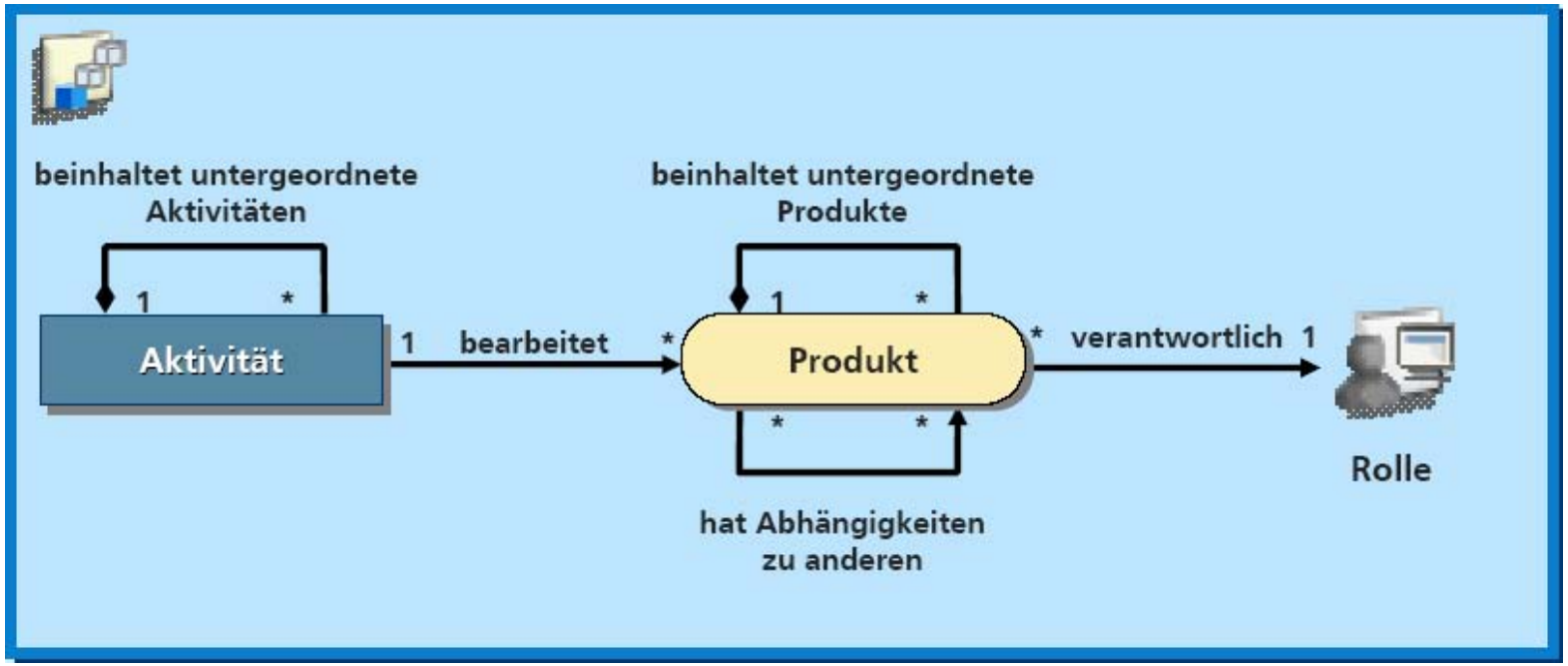
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Das V-Modell XT - ein modulares Vorgehensmodell
- XT= „eXtreme Tailoring“
- WEIT-Projekt 2002-2004
  - KBSt/IT-AmtBw
  - TU München / TU Kaiserslautern und Partner
- Kosten ca. 4 Mio. €, ca. 30 beteiligte Personen
- Nach WEIT geht’s immer WEITER...
- Version 1.2 seit dem 1. Februar 2006

V-Modell XT - Grundlagen

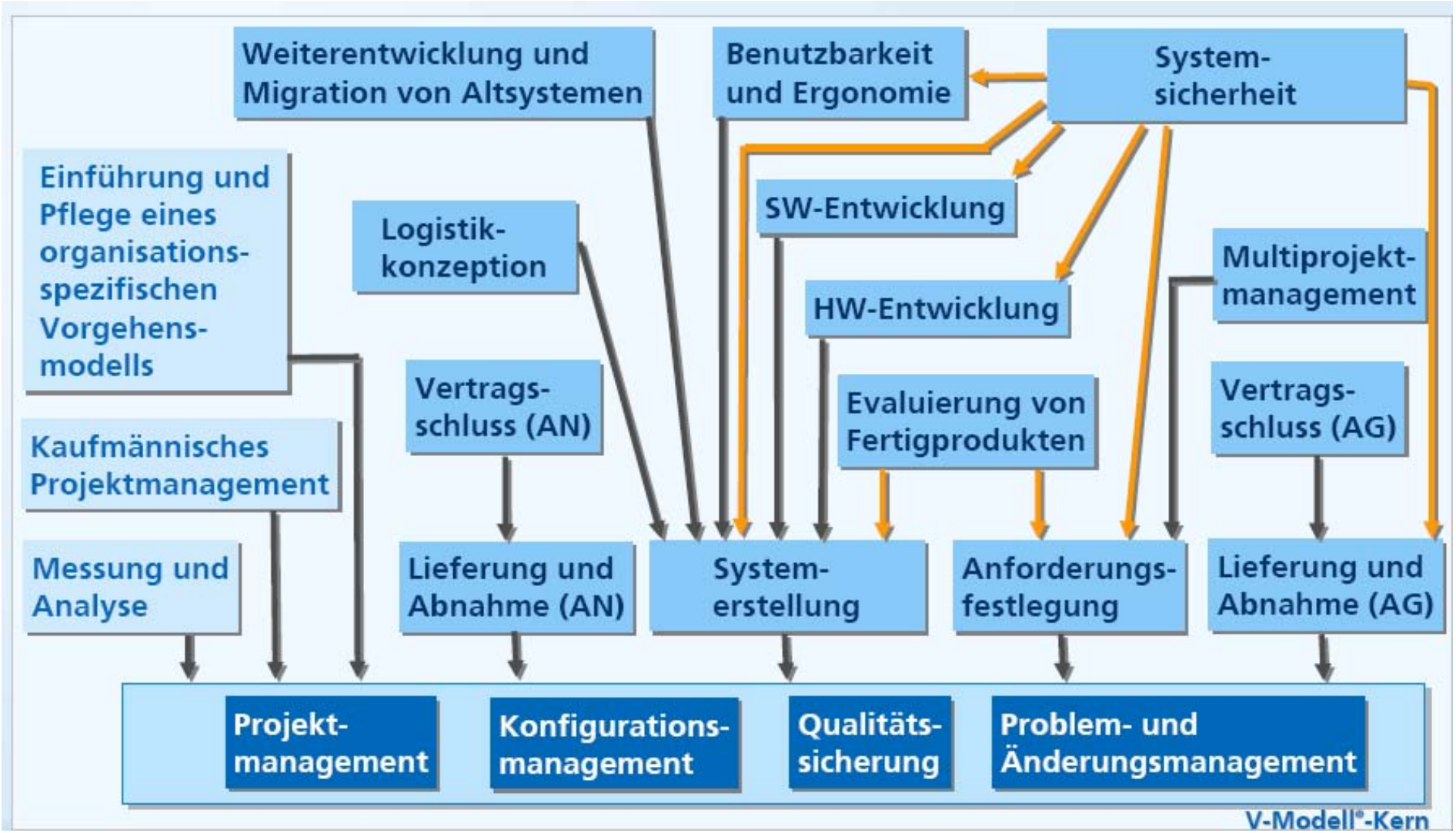
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- **Vorgehensbausteine**
  - **Kapseln Aktivitäten, Produkte und Rollen**
  - **sind die modularen Einheiten für das Tailoring**
  - **können von anderen Vorgehensbausteinen abhängen**



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

V-Modell XT - Vorgehensbausteinkarte



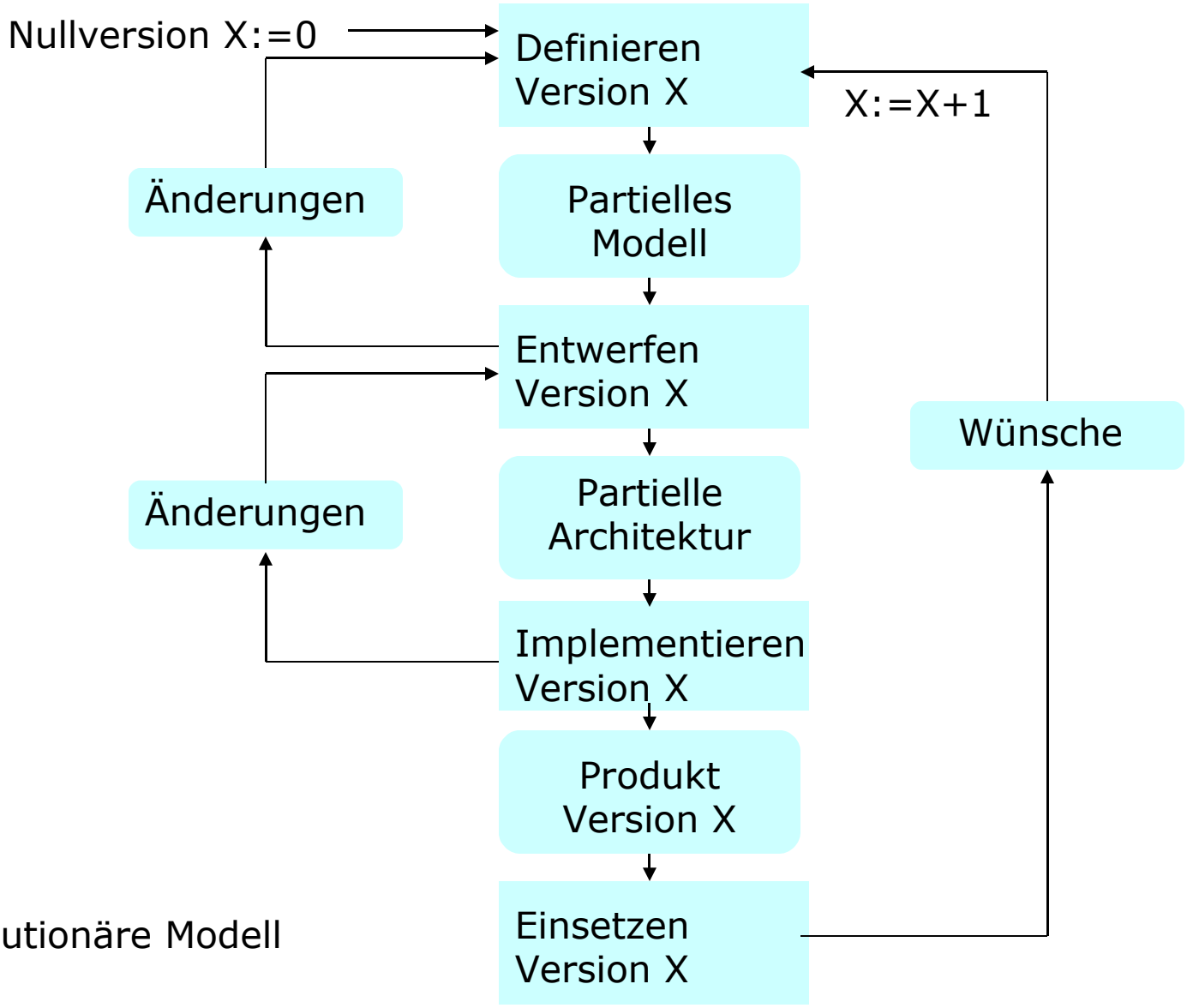
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

4. Das Prototypen-Modell

- Unterschiede zwischen einem Software-Prototypen und einem Prototypen in anderen Ingenieursdisziplinen. Ein Software-Prototyp:
  - ist nicht das „erste Muster" einer großen Serie von Produkten (da Vervielfältigung kein Ingenieursproblem).
  - zeigt ausgewählte Eigenschaften des Zeilproduktes im praktischen Einsatz.
- Ähnlichkeiten in der Anwendung der Prototypen:
  - Sie werden verwendet, um relevante Anforderungen oder Entwicklungsprobleme zu klären.
  - Sie dienen als Diskussionsbasis und helfen bei Entscheidungen.
  - Sie werden für experimentelle Zwecke verwendet und um praktische Erfahrungen zu sammeln.
- **Prototypen-Modell** unterstützt die Erstellung ablauffähiger Modelle des zukünftigen Produkts.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

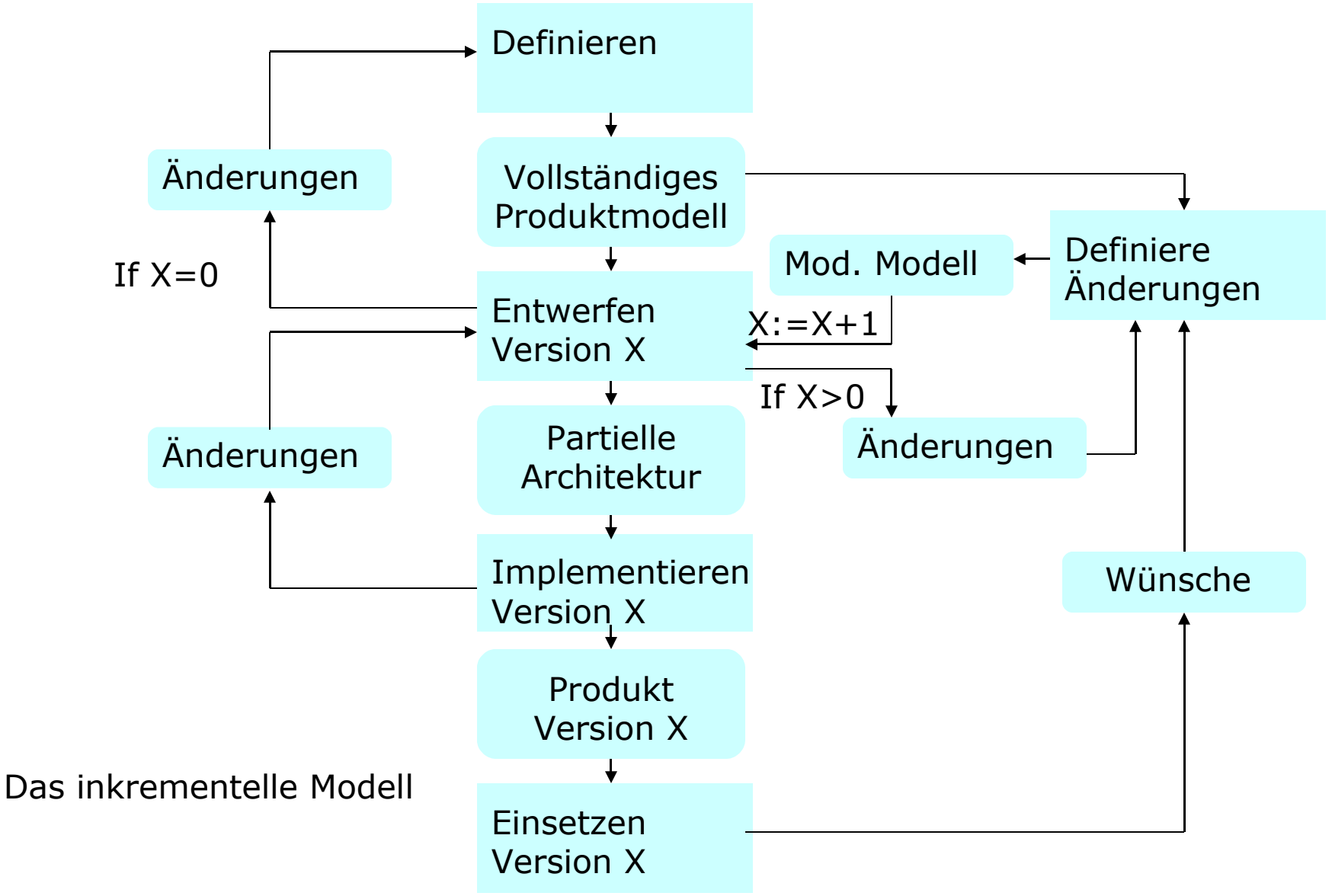
5. Das evolutionäre/inkrementelle Modell



Das evolutionäre Modell

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

5. Das evolutionäre/inkrementelle Modell



Das inkrementelle Modell

Probleme bei der Software-Entwicklung

- Häufige Änderungen an Software-Elementen verursachen ein Chaos. Bereits korrigierte Fehler tauchen wieder auf, es ist unklar von wem welche Änderungen durchgeführt wurden.
  → *Reduktion des Problems durch Aufzeichnung der Historie.*
- Es ist unklar, ob ein Fehler bereits behoben wurde oder nicht.

  → *Verknüpfung von Änderungswünschen und vorgenommenen Änderungen.*

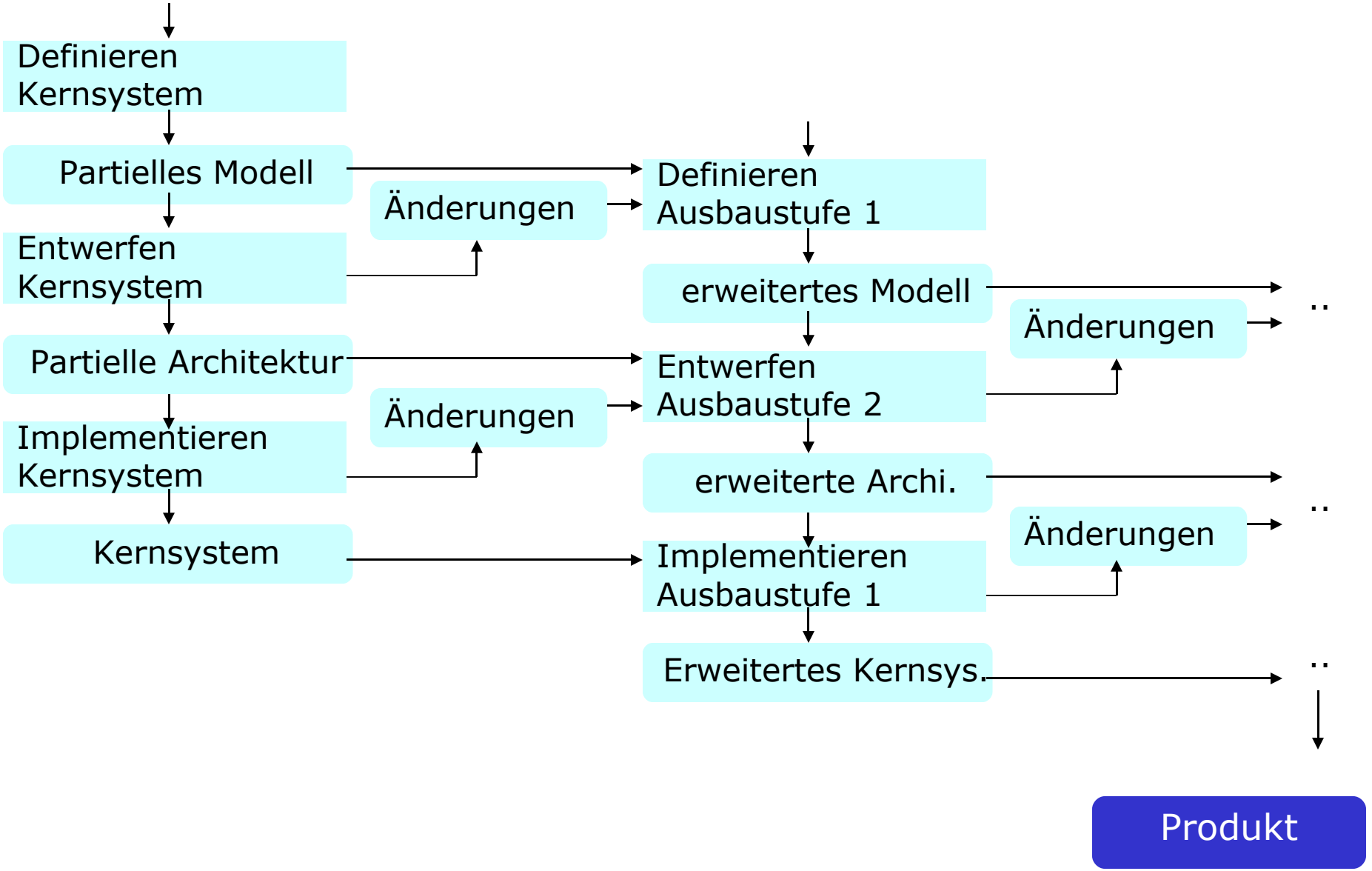
  → *Überwachung des Änderungsprozesses*
- Es ist schwierig, das System so zu konfigurieren, dass alle Fehlermeldungen bis vor zwei Wochen berücksichtigt sind. Die letzten „Verbesserungen" waren Fehlerhaft, sie müssen aus der Konfiguration entfernt werden.

  → *Automatische Versions- und Konfigurationsselektion.*
- Man ist unsicher, ob alles neu übersetzt wurde und ob die Kunden die neueste Freigabe haben.

  → *Software-Konfigurationsmanagement sorgt dafür, dass kein Arbeitsschritt bei der Vor- und Nacharbeitung von Software-Elementen vergessen wird.*

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

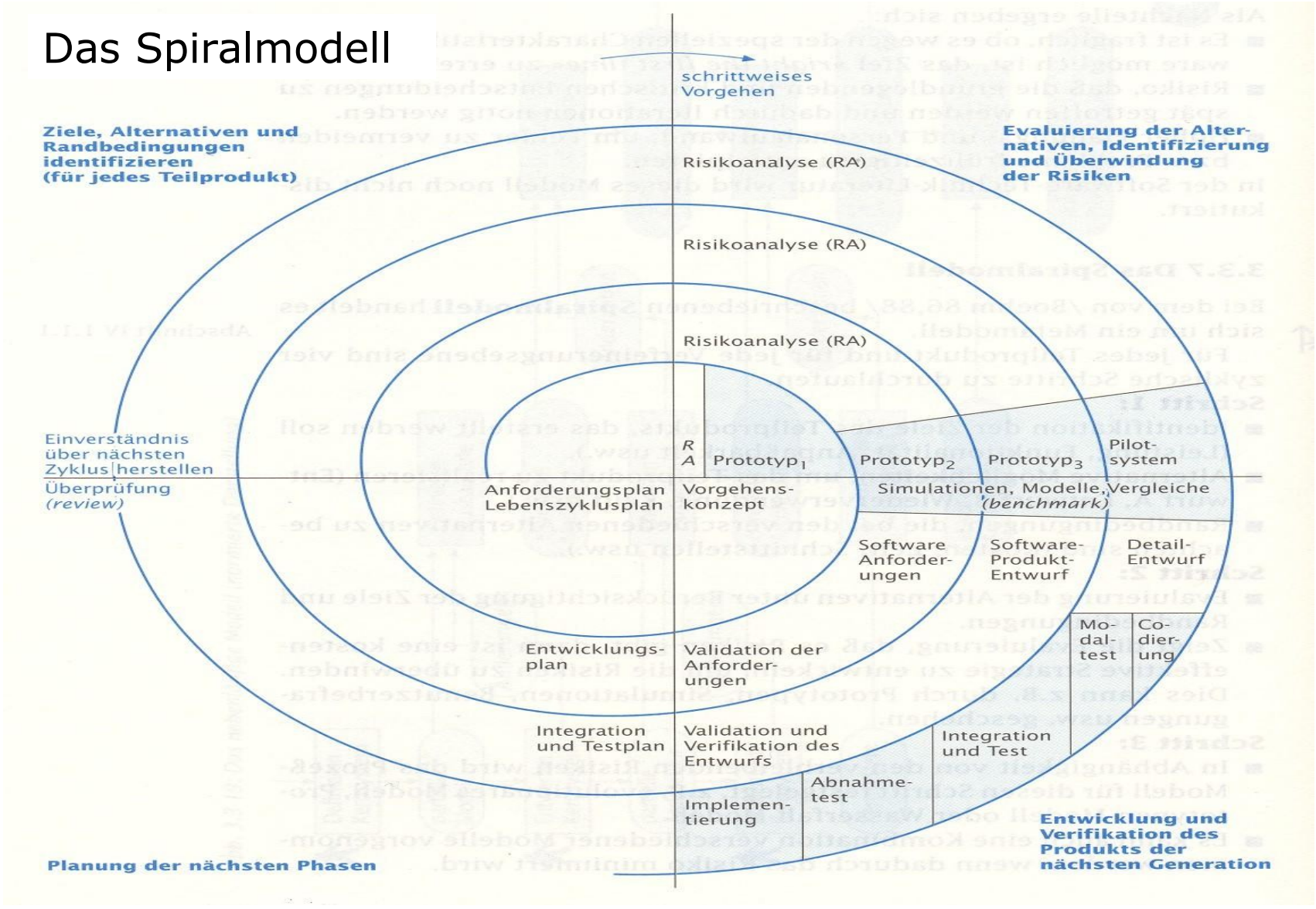
7. Das nebenläufige Modell



8. Das Spiral-Modell

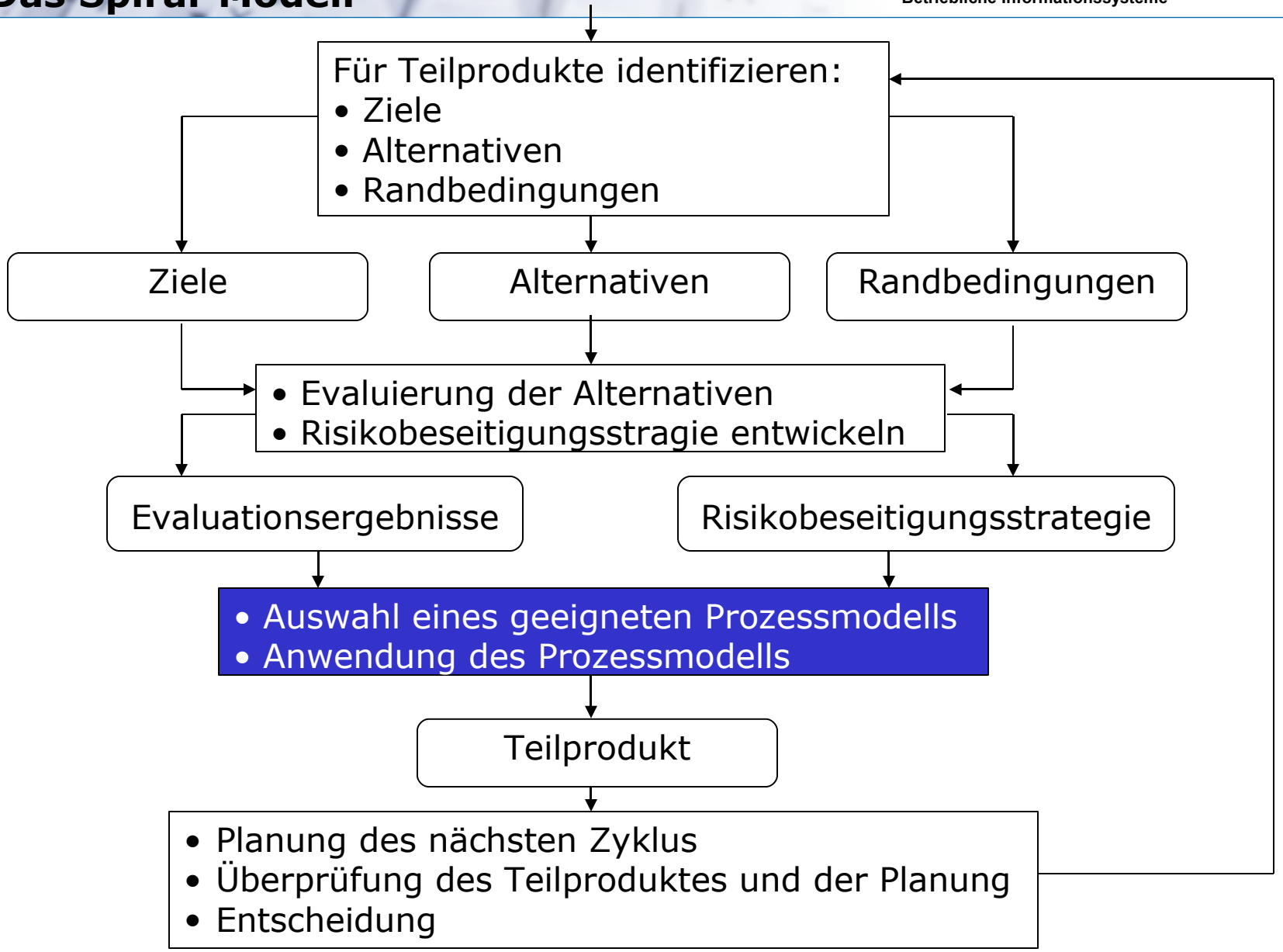
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Das Spiralmodell



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

8. Das Spiral-Modell



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zusammenfassung

Prozessmodell	Primäres Ziel	Antreibendes Moment	Benutzerbeteiligung	Charakteristika
Wasserfall-Modell	min. Managementaufwand	Dokumente	Gering	Sequentiell, volle Breite
V-Modell	max. Qualität	Dokumente	Gering	Sequentiell, volle Breite, Validation, Verifikation
Prototypen-Modell	Risikominimierung	Code	Hoch	Nur Teilsysteme
Evolutionäres Modell	min. Entwicklungszeit	Code	Mittel	Sofort: nur Kernsystem
Inkrementelles Modell	min. Entwicklungszeit und Risiko	Code	Mittel	Volle Definition dann zunächst nur Kernsystem

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zusammenfassung

Prozessmodell	Primäres Ziel	Antreibendes Moment	Benutzerbe-teiligung	Charakteristika
OO-Modell	Zeit und Kosten-minimierung	Wiederver-wendbare Komponenten	?	Volle Breite in Abh. Von WV-Komponenten
Nebenläufiges Modell	min. Ent-wicklungszeit		Hoch	Volle Breite, nebenläufig
Spiralmodell	Risikominimie-rung		Mittel	Entscheidung pro Zyklus über weiteres Vorgehen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Literatur

- [Benington 56]
  Benington H.D., Production of large computer programs
- [Boehm 81]
  Boehm B.W., Software Engineering Economics, Prentice Hall
- [Boehm 84]
  Boehm B.W., Verifying and validating Software Requirements and Design Specification, IEEE Software
- [Boehm 86]
  Boehm B.W., A Spiral Model of Software Development and Enhancement, ACM SIGSOFT
- [Boehm 88]
  Boehm B.W., A Spiral Model of Software Development and Enhancement, IEEE
- [Royce 70]
  Royce W.W., Managing the development of large software systems, IEEE
- [Spectrum 91]
  Special Report: Concurrent Engineering, IEEE Spectrum

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

1. Grundlagen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- [DeMarco, Lister 91]: „Die großen Probleme bei unserer Arbeit sind keine technologischen Probleme, sondern soziologische Probleme".
- Erfolgreiche Software-Entwicklungen auf gute menschliche Zusammenarbeit zurückzuführen.
- Produktive Einsetzung von Mitarbeitern setzt
  - allgemeine Qualifikationen, die eine Tätigkeit im Bereich der Software-Technik generell erfordert und
  - spezielle Qualifikationen, die für ihre Tätigkeit innerhalb der Software-Technik notwendig sind

voraus.

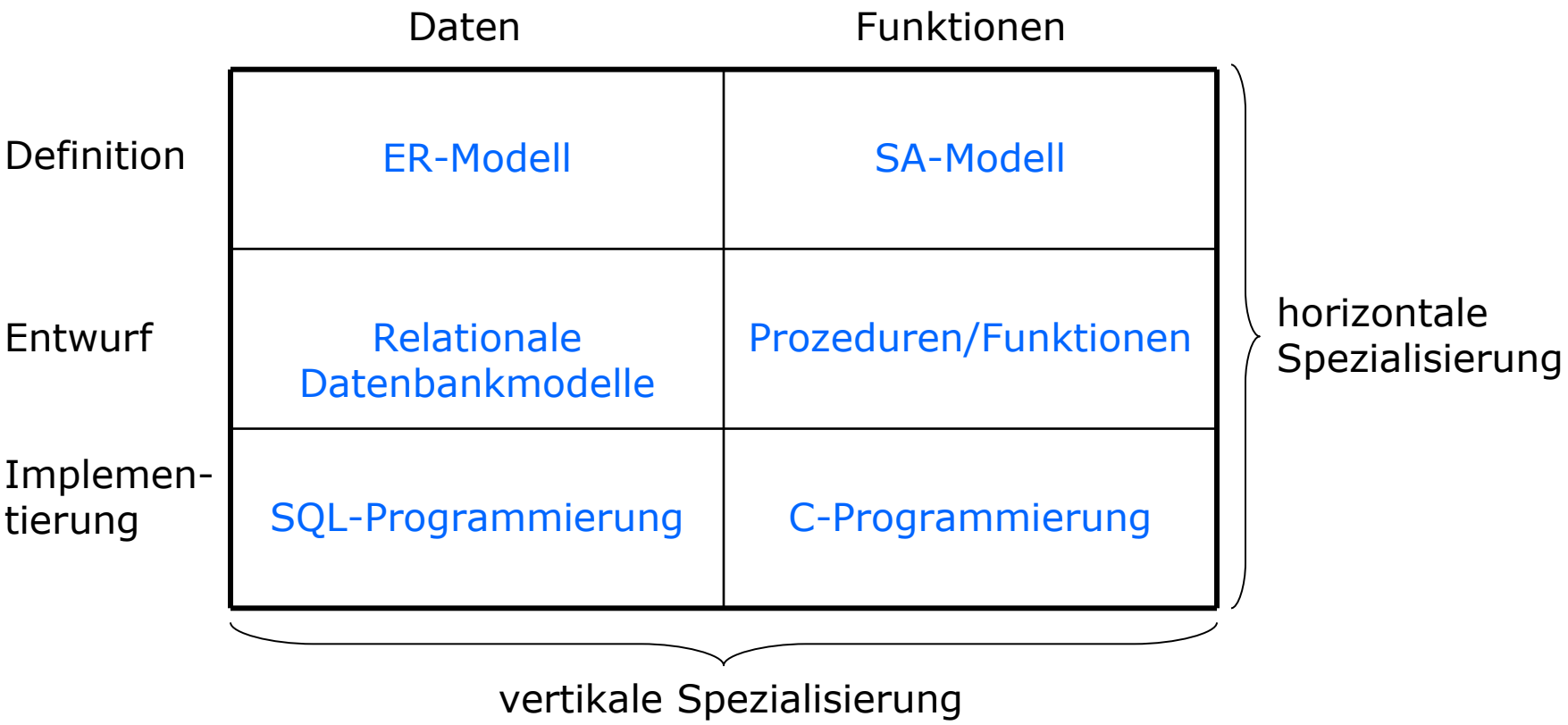
1.1. Allgemeine Qualifikation

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zur Software-Erstellung notwendige Qualifikationen	• Fähigkeit zum Abstrahieren. • Sprachliche und schriftliche Kommunikationsfähigkeit. • Teamfähigkeit. • Wille zum lebenslangen Lernen. • Intellektuelle Flexibilität und Mobilität. • Kreativität. • Hohe Belastbarkeit. • Englisch sprechen und lesen. • Schreibmaschine schreiben.

1.2. Spezialisierung

Horizontale vs. Vertikale Spezialisierung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Ziele und Aufgaben

Ziel: Für die Durchführung der aktuellen und der zukünftigen Entwicklungsaufgaben in ausreichender Qualität und Quantität Personal zu wirtschaftlichen Kosten zur Verfügung stellen

- Zwei Problembereiche:
  - Fluktuation von Mitarbeitern;
  - Qualifikationsprobleme bei nicht forlbildungsfähigen Mitarbeitern.
- Lösungsmöglichkeiten:
  - Gute personenabhängige Dokumentation;
  - Unabhängige Qualitätssicherung;
  - Attraktive Arbeitsbedingungen;
  - Realistische Erwartungen bei der Personalplanung;
  - Gute Personalförderung;
  - Mittelfristige Personalplanung.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Literatur

- [DeMarco, Lister 91]
  Wien wartet auf dich! Der Faktor Mensch im DV-Management, Carl Hanser Verlag

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Hochqualifizierte Mitarbeiter führen

Führung: Einwirkung auf die Mitarbeiter, so dass vorgegebene Ziele erreicht werden.

- **Management by Objectives**
  - Erfordert Festlegung der Unternehmensziele. Aus diesen werden die Ziele der verschiedenen Abteilungen abgeleitet.
  - Vorraussetzungen:
    - Mitarbeitern wird ein bestimmter Aufgabenbereich delegiert;
    - Mitarbeitern wird erläutert, wie ihre Ziele in übergeordnete eingebettet sind,was von der nächsten Periode erwartet wird und welche Unterstützung die Führungskraft bereitstellt;
    - Festlegung, voran bzw. wie die Leistung gemessen wird.
- **Management by Results**
  - Dezentrale Führungsorganisation bei der die Ergebnisse vorgegeben, gemessen und kontrolliert werden.
  - Grundsätze:
    - Die Abteilungen konzentrieren sich auf wenige, möglichst quantitative Entscheidungsmaxima;
    - Die Ziele sollen motivieren;
    - Die Führungskräfte werden auf allen Hierarchieebenen ausreichend über die von ihnen erwarteten Verhaltensweisen informiert.

2. Hochqualifizierte Mitarbeiter führen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- **Management by Delegation**
  - Aufgaben und Befugnisse werden soweit wie möglich an die Mitarbeiter und auf untere Hierarchieebenen übertragen.
  - Vorraussetzungen:
    - Klare Aufgabendefinitionen;
    - Kompetenzabgrenzungen.
- **Management by Participation**
  - Starke Betonung der Mitarbeiterbeteiligung an den sie betreffenden Zielentscheidungen.
  - Grundidee:
    - Die Identifikation der Mitarbeiter mir den Unternehmenszielen wächst, je stärker sie an der Formulierung dieser Ziele mitwirken.
- **Management by Alternatives**
  - Für jedes Problem sind Alternativlösungen zu entwickeln. Erst nach der Bewertung der Alternativen wird eine Entscheidung gefällt.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Hochqualifizierte Mitarbeiter führen

- **Management by Exception**
  - Normal- und Routinefälle werden von der mittleren und unteren Führungsebene völlig selbstständig bearbeitet und entschieden.
  - Vorgesetzte werden nur dann hinzugezogen, wenn Ausnahmnefälle vorliegen.
  - Vorraussetzungen:
    - Klare Definition der übertragenen Aufgaben;
    - Umfassende Richtlinien für die Entscheidungen der einzelnen Stellen;
    - Übertragung von Vollmacht und Verantwortung.
- **Management by Motivation**
  - Aufgabe des Managers besteht darin,
    - die Bedürfnisse, Interessen, Einstellungen und persönliche Ziele der Mitarbeiter zu erkennen und
    - sie mit den Unternehmenszielen und betrieblichen Erfordernissen zu verbinden, so dass die Mitarbeiter Spaß an der Arbeit haben.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Teams

In einem **Team** arbeiten Mitarbeiter unterschiedlicher Qualifikationen miteinander, um eine gemeinsame Aufgabe zu erledigen.

- Charakteristika der Teamarbeit:
  - Regelmäßige und kontinuierliche Kommunikation;
  - Von Fall zu Fall gegenseitige Abstimmung;
  - Gleichberechtigte Mitbestimmung aller Teammitglieder;
  - Alle Teammitglieder sind und agieren gleichrangig;
  - Verschiedene Teammitglieder übernehmen zeitweise die Führungsrolle.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

4. Kreativität fördern

Kreativität: Fähigkeit, Wissens- und Erfahrungselemente aus verschiedenen Bereichen unter Überwindung verfestigter Strukturen und Denkmuster zu neuen Problemlösungen bzw. zu neuen Ideen zu verschmelzen.

Kreativitätstechniken: Wenden heuristische Prinzipen in formalisierter Form an.

Heuristische Prinzipien:

- Assoziieren;
- Kombinieren;
- Variieren.

➔ Klassifizierung der Kreativitätstechniken

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

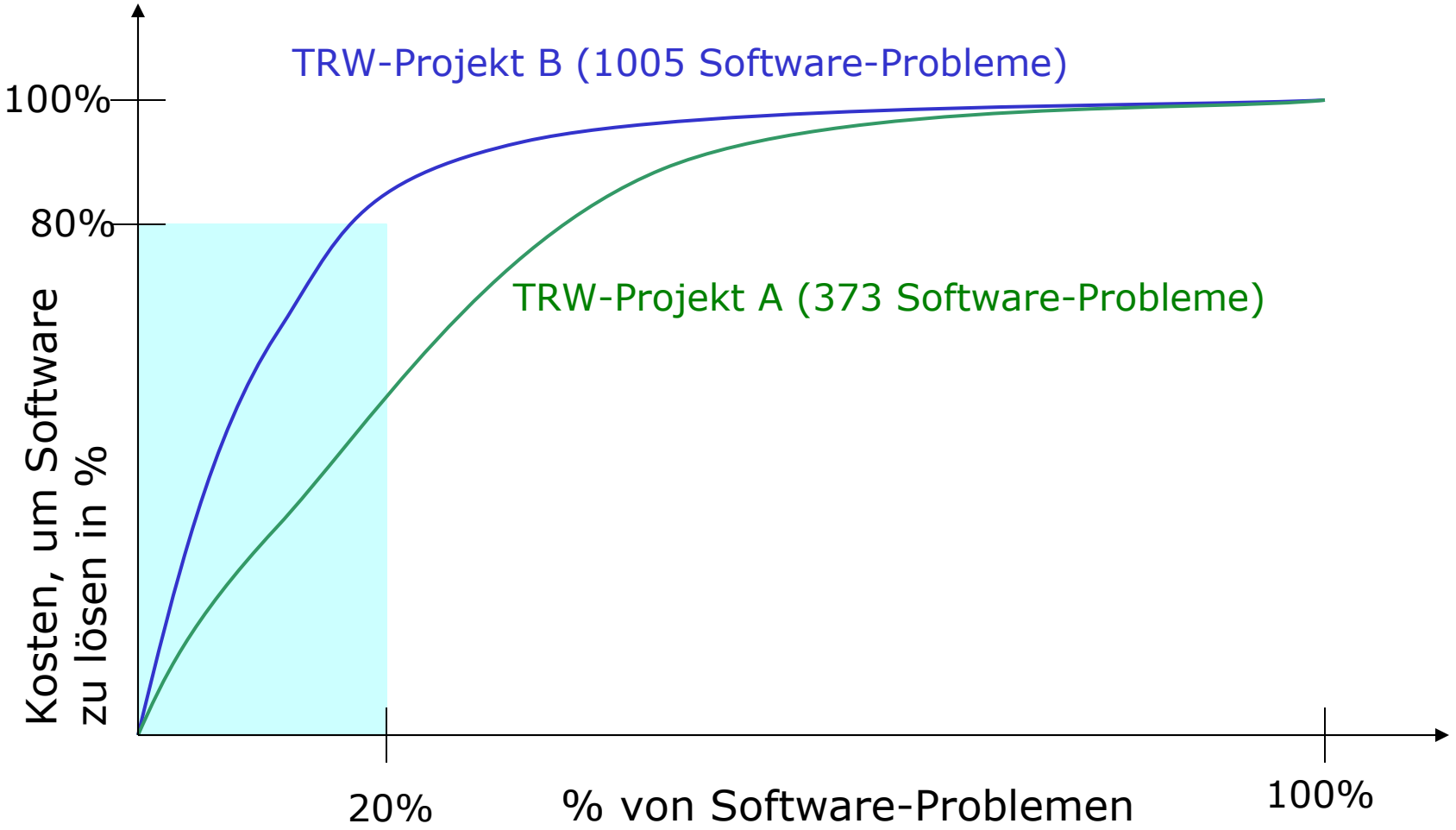
4. Kreativität fördern

Förderung der Kreativität

- Führung zu Kreativität heißt [Schlicksupp 85]:
  - Freiräume für Experimente gewähren;
  - Initiativen anerkennen;
  - Ungewöhnliches positiv diskutieren;
  - Erfolgsziele eindeutig definieren, aber die Wege zu den erfolgen weitgehend offen lassen.
- **Metaplan-Technik**
  - Wichtigstes Arbeitsmittel für Gruppen- und Teambesprechungen.
  - Einsatz von Pinnwänden, Flip-Charts, verschiedenartig geformte, farbige Kanten, Stecknadeln, Klebepunkte, und Filzstifte.
  - Dienen der Visualisierung, Strukturierung und Gewichtung von Ideen.

5. Risiken managen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme



Überarbeitungskosten von Software

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

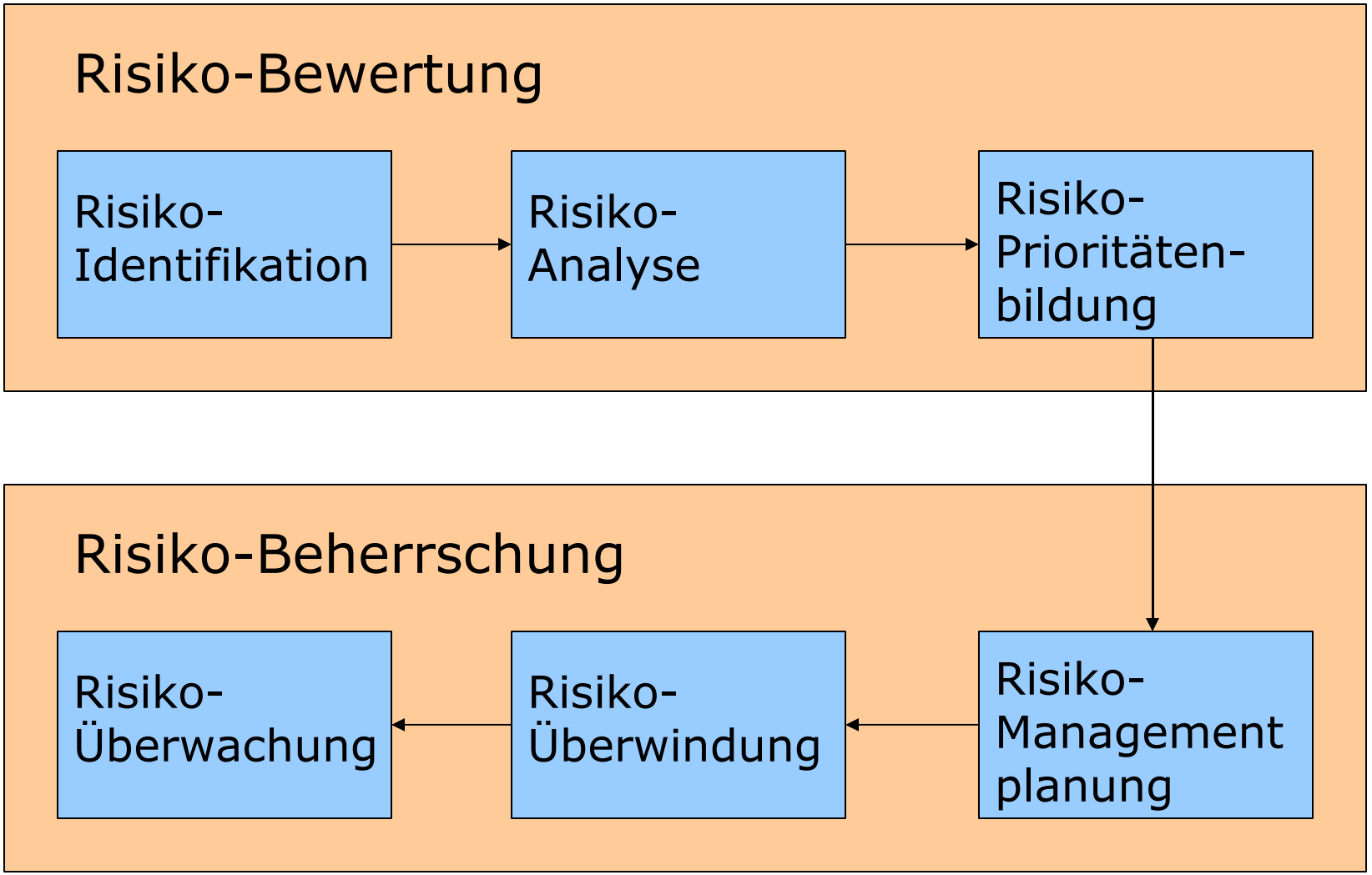
5. Risiken managen

Risikomanagement

- 20% der Probleme sind risikoreiche Probleme.
- Ziel des Software-Risikomanagement:
  - Formalisierung der Wechselbeziehungen zwischen Risiken und Erfolg und
  - deren Umsetzung in anwendbare Prinzipien und Praktiken.
- Aufgabe des Risikomanagement:
  - Identifizieren,
  - Ansprechen und
  - Beseitigen von Risiken.
- Risiko: beschreibt die Möglichkeit, dass eine Aktivität einen körperlichen oder materiellen Verlust oder Schaden zufolge hat.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

5. Risiken managen



Die sechs Schritte des Risiko-Managements

Literatur

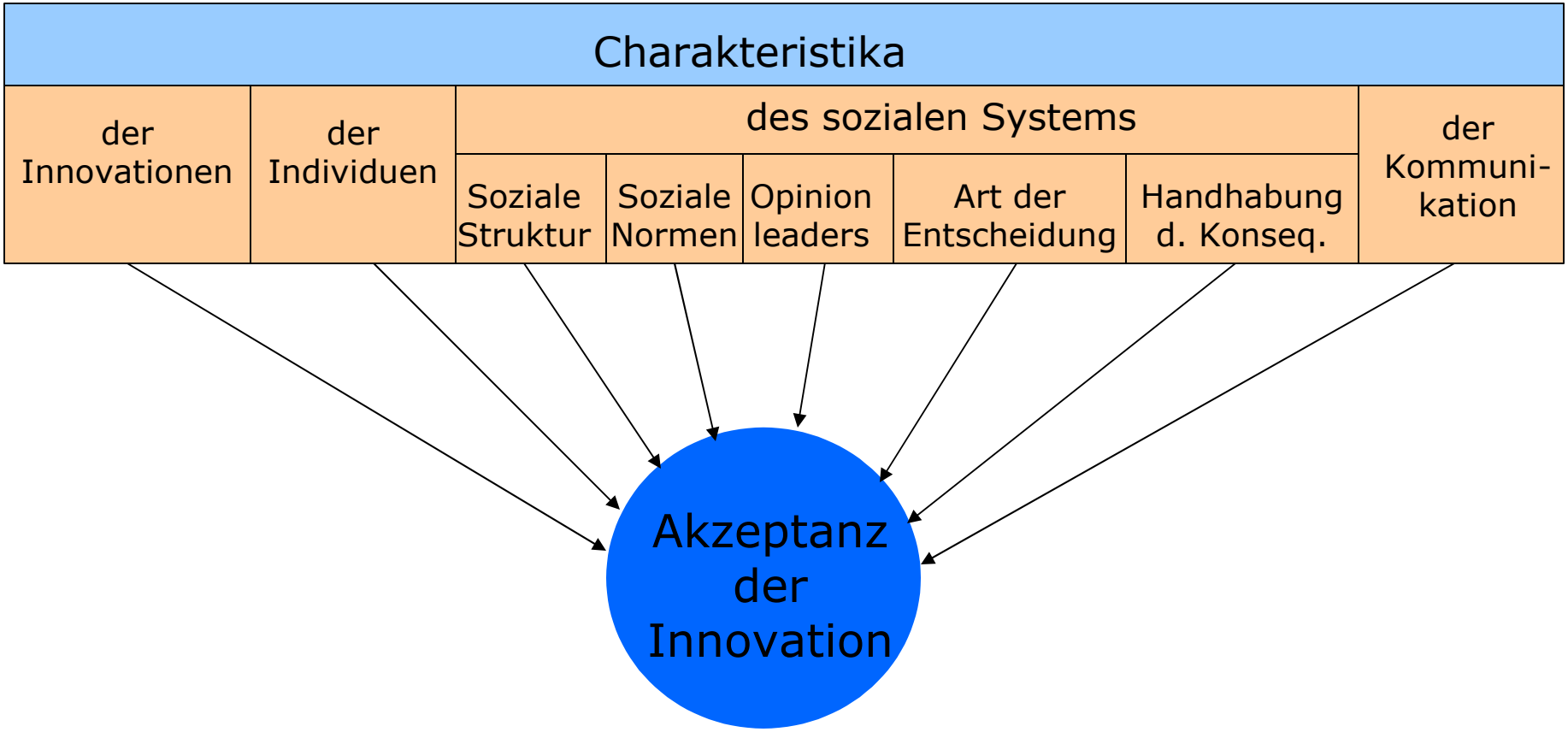
- [Peters,Walterman 82]
  Peters, T.J., Waterman R.H.: In search of Excellence, Lessons from America‘s Best-Run Companies, Harper&Row

- [Schlicksupp 85]:
  Schlicksupp H.: Jedem nacht es Spaß zu denken, in: Management Wissen

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

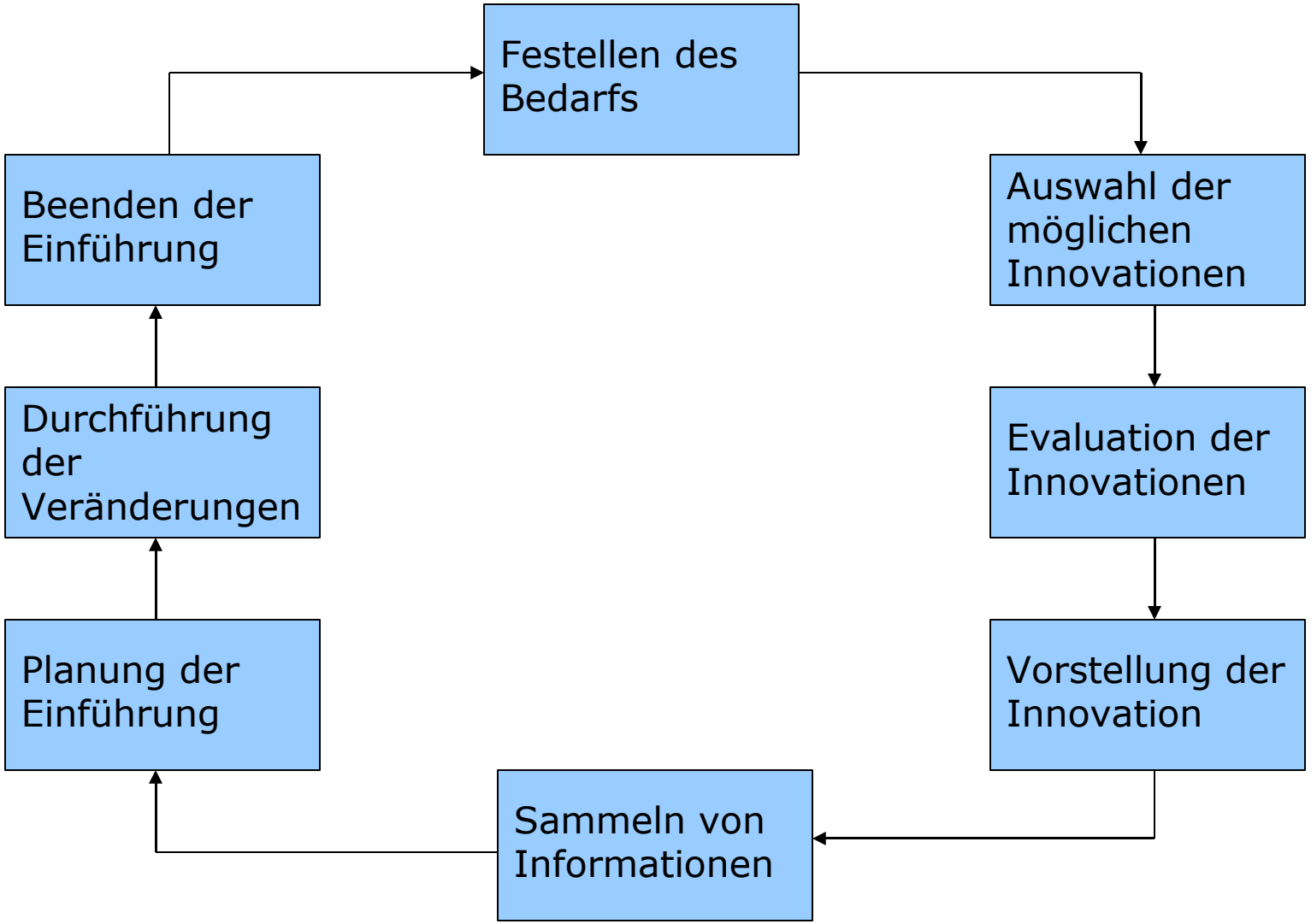
Bestimmende Faktoren des Technologie-Transfers



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Der Lebenszyklus

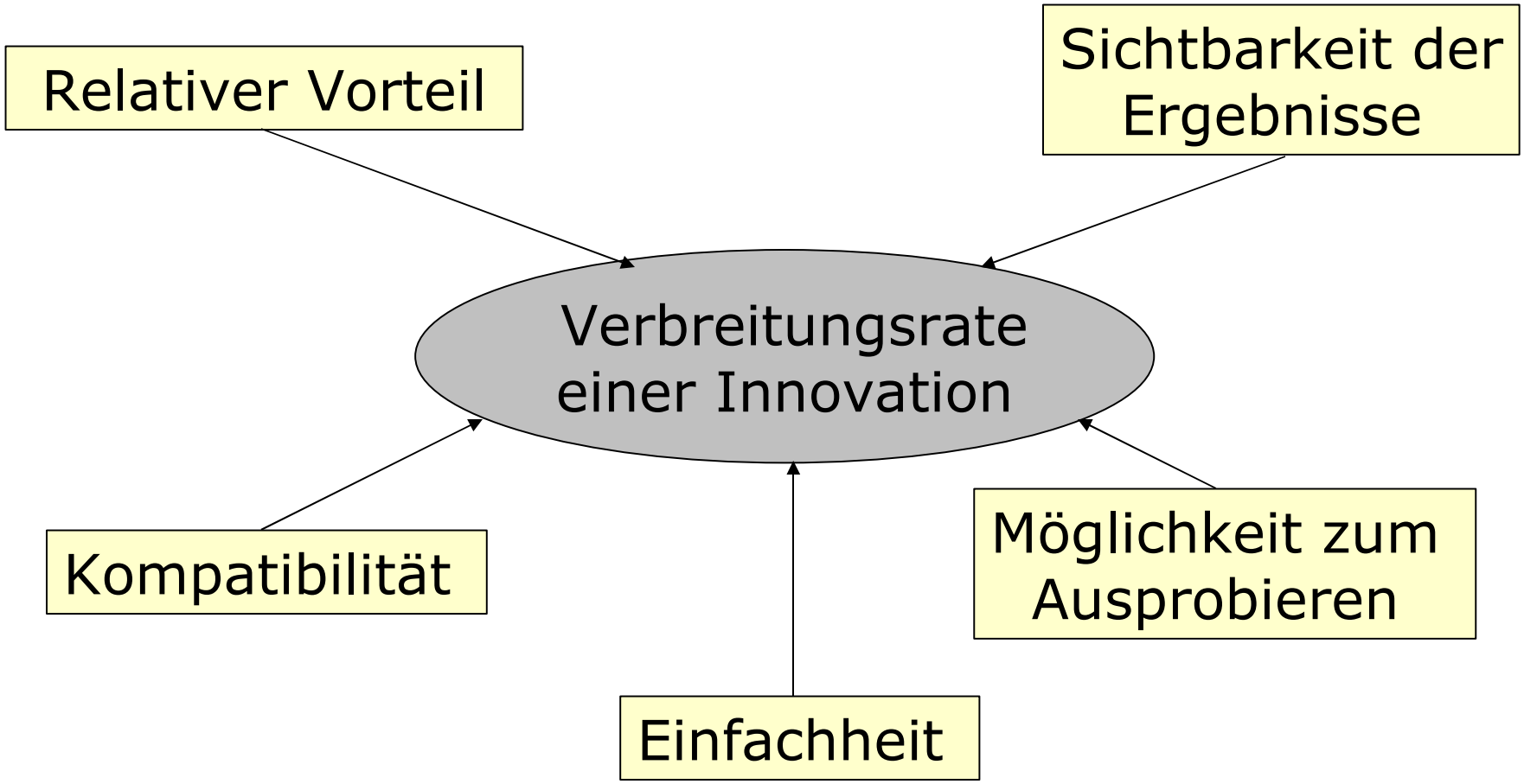
Lebenszyklus einer Innovationseinführung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3. Charakteristika einer Innovation

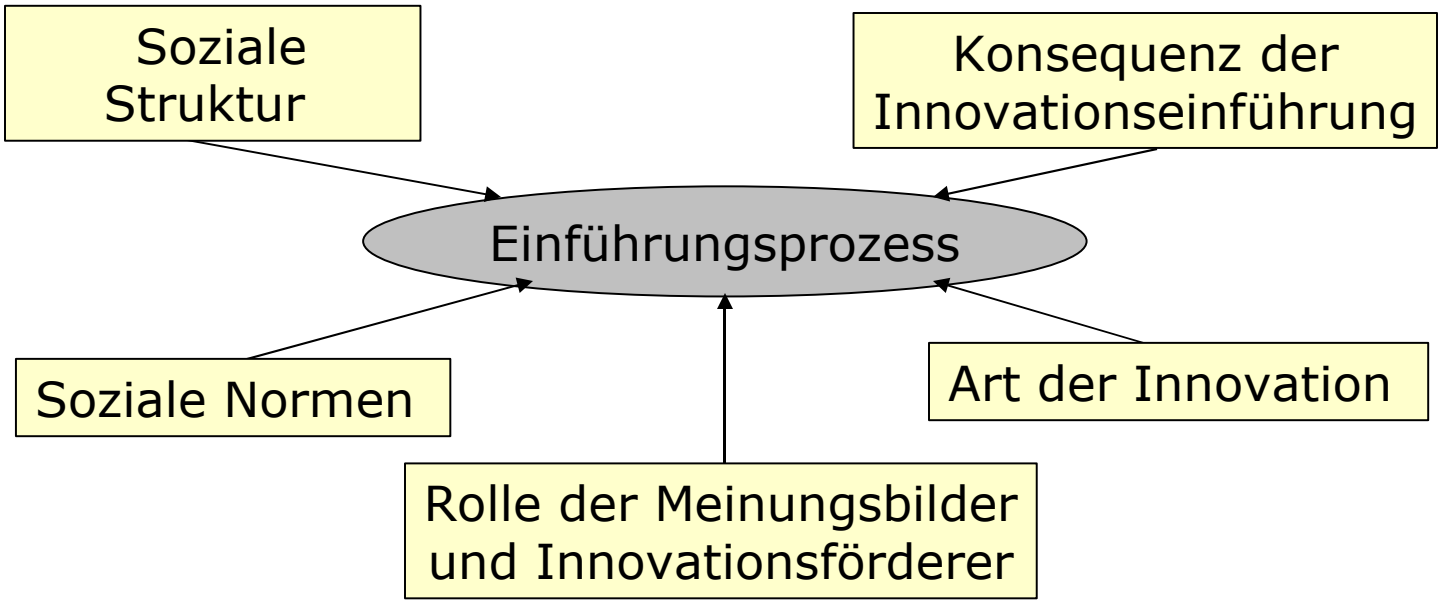
Beeinflussende Faktoren nach [Rogers 83]



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

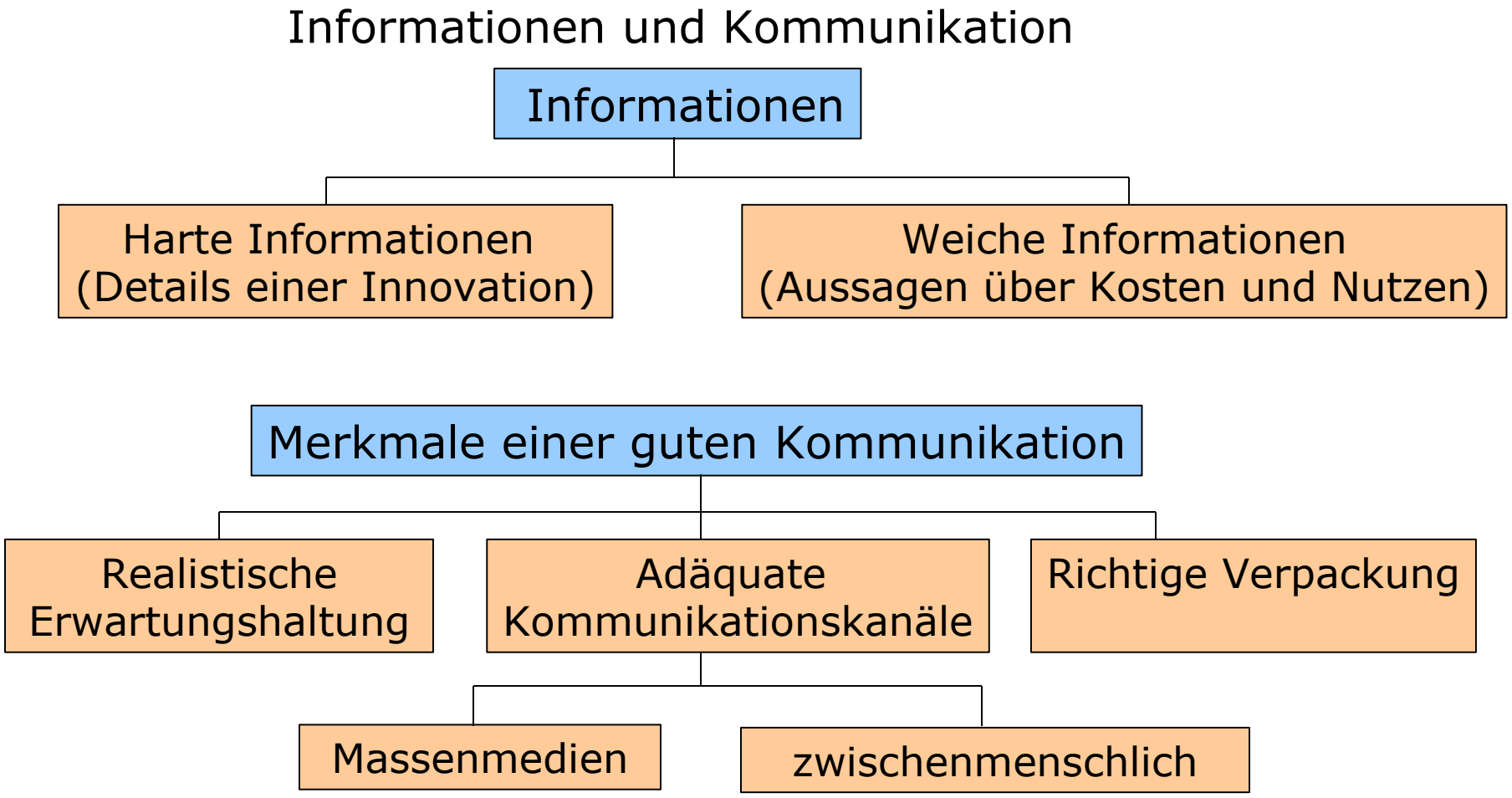
5. Charakteristika des sozialen Systems

Soziales System: Umgebung, in der der Einführungsprozess stattfindet



→ beeinflusst

6. Charakteristika des Kommunikationsprozesses



- Die Kommunikation wird auch durch die Gleichartigkeit der Zielgruppe beeinflusst.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Literatur

- [Asthana 95]
  Asthana P., Jumping the Technology S-Curve, IEEE Spectrum
- [Fisher 88]

  Fisher A, CASE- Using Software Development Tools, John Wiley & Sons
- [Kemerer 92]
  Kemerer C.F., Hoe the Learning Curves Affects CASE Tool Adaption, IEEE Software
- [Offerbein 93]
  Offerbein, T., Erreichen von Wiederwendbarkeit und Erweiterbarkeit von Klassenbibliotheken am Beispiel eines Leitstandes, ReUse Konferenz
- [Rogers 83]

  Rogers E.M. Diffusion of Innovations, Free Press
- [Yourdon 86]

  Youdon E., Managing the Structured Techniques- Strategies for Software Development in the 1990‘s, Yourdon Press

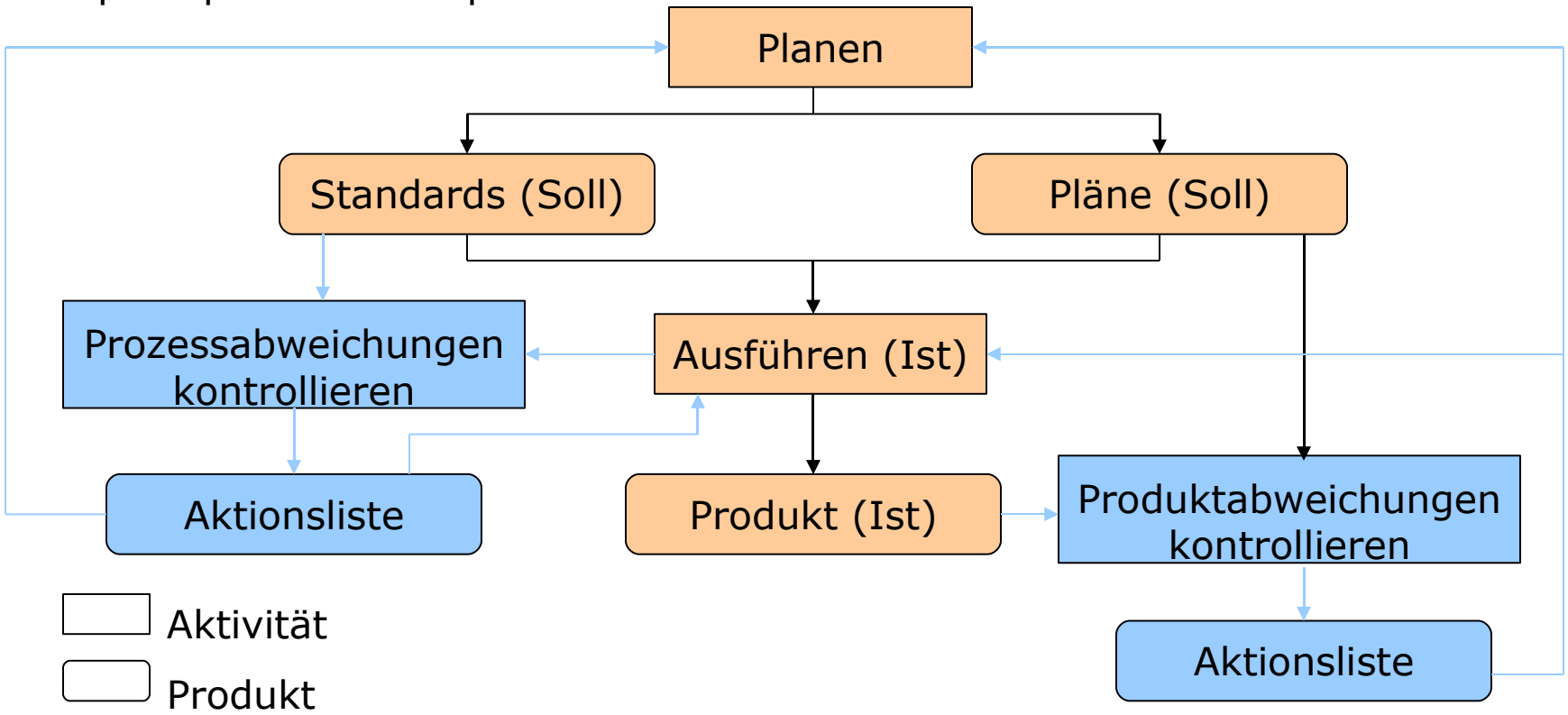
Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. Grundlagen

- Zur **Kontrolle** gehören alle Management-Aktivitäten, die sicherstellen, dass die laufenden Tätigkeiten mit dem Plan übereinstimmen.
- **Pläne** legen Produktanforderungen, Zeit und Kosten fest.
- Unter **Standards** werden für die Ausführung benötigte Prozessmodelle, Richtlinien, Methoden und Werkzeuge zusammengefasst.
- Der prinzipielle Kontrollprozess:



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

1. Grundlagen

- Methoden und Werkzeuge zur Kontrolle müssen:
  - objektiv,
  - anpassbar und
  - ökonomisch sein.
- Abweichungen vom Plan und den Standards müssen ohne Rücksicht auf die betroffenen Mitarbeiter und Stellen aufgezeigt werden.
- Kontrolle muss zu korrigierenden Aktionen führen, entweder um den Prozess zu den Standards und den Plan zurückzubringen, die Standards und den Plan zu ändern oder den Prozess zu beenden.
- Mit der Kontrolle sind folgende Probleme verbunden:
  - Viele Kontrollmethoden nehmen die bisher benötigte Zeit und die bisher verbrauchten Kosten als Maßstab für die Entwicklungsfortschritt.
  - Standards für Entwicklungsaktivitäten sind entweder nicht vorhanden oder nicht schriftlich fixiert.
  - Eine Software-Messtechnik, die Software-Maße über den Entwicklungsprozess und das Produkt bereitstellt, ist noch nicht voll entwickelt.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Aufgaben eines Software-Managers (1)

1. Standards entwickeln und festlegen

- Standards können für eine gesamte Firma oder für jeweils ein Produkt verbindlich vorgeschrieben werden.
- Werden selbst entwickelt oder von Organisationen oder anderen Firmen übernommen.
- Folgende Teilaktivitäten sind durchzuführen:
  - Entwicklung von Qualitäts- und Quantitätsstandards,
  - Festlegen des Prozessmodells,
  - Festlegen der QS-Methoden,
  - Entwicklung von Produktivitäts-, Qualitäts- und Prozessmetriken,
  - Standards sollten immer immer daraufhin überprüft werden, ob sie folgende Vorteile mit sich bringen:

Vorteile	• Senkung der Einarbeitungs- und Umschulungskosten. • Verbesserung der Kommunikation zwischen Teammitarbeitern. • Erleichterung des Personalaustauschs zwischen Projekten. • Erfahrungen können besser weitergegeben werden. • Einheitliche Anwendung der besten Erfahrungen erfolgreicher Projekte. • Vereinfachung der Wartung und der Pflege. • Kontrolle der Standards.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Aufgaben eines Software-Managers (2)

2. Kontroll- und Berichtssystem etablieren

- Kontroll- und Berichtssysteme müssen ausgesucht oder entwickelt werden, um den Entwicklungsprozess zu überwachen und jederzeit den Entwicklungsstatus bestimmen zu können.
- Für Entwicklungsbereichte ist der Typ, die Häufigkeit, der Ersteller und der Empfänger festzulegen.
- Die Art und der Umfang von Kontroll- und Berichtssystemen hängen von folgende Parametern ab:
  - Verwendeter Führungsstil (Management-by ...),
  - Verwendete Aufbau- und Ablauforganisation (Prozessmodelle),
  - Umfang der Entwicklungszeit (Zeit, Anzahl, Mitarbeiter),
  - Eingesetzte CASE-Umgebung.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Aufgaben eines Software-Managers (2)

- Zur Aufwandsermittlung pro Phase kann eine einfache Strichliste verwendet werden, die ökonomisch ausgefüllt werden kann. Die Auswertung trägt wesentlich dazu bei, Aufwandsschätzungen für neuere Projekte genauer vorzunehmen.
- Hat man die Projektplanung mit Hilfe eines Planungssystem vorgenommen, kann dieses auch für die Projektkontrolle verwendet werden.
- Terminpläne können detailliert überwacht werden:
  - Anfangs- und Endtermine sowie Dauer von Vorgängen,
  - Prozentsatz, zu dem Vorgänge bereits abgeschlossen sind,
  - Kosten des Projekts, einzelner Vorgänge und Ressourcen,
  - Arbeitsstunden, die von jeder Ressource ausgeführt wurden.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Aufgaben eines Software-Managers (3, 4)

3. Prozess und Produkte vermessen

- Für die Qualitätsüberprüfung sowohl des Prozesses als auch des Produktes ist die Qualitätssicherung zuständig.
- Sind entsprechende Messverfahren etabliert, so muss das Software-Management sicherstellen, dass die Messungen konsequent durchgeführt werden.

4. Korrigierende Aktionen initiieren

- Plan- und Standard-Änderungen können zu zusätzlichen Budget-Anforderungen, zusätzlichen Mitarbeitern oder leistungsstärkeren Arbeitsplatzrechnern führen.
- Manchmal ist es möglich einen Teil des Plans einzuhalten, wenn die Zeit für die Einhaltung eines anderen Plans vergrößert wird.
- Als letzte Maßnahme können auch Anforderungen an das Produkt geändert werden.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Aufgaben eines Software-Managers (5)

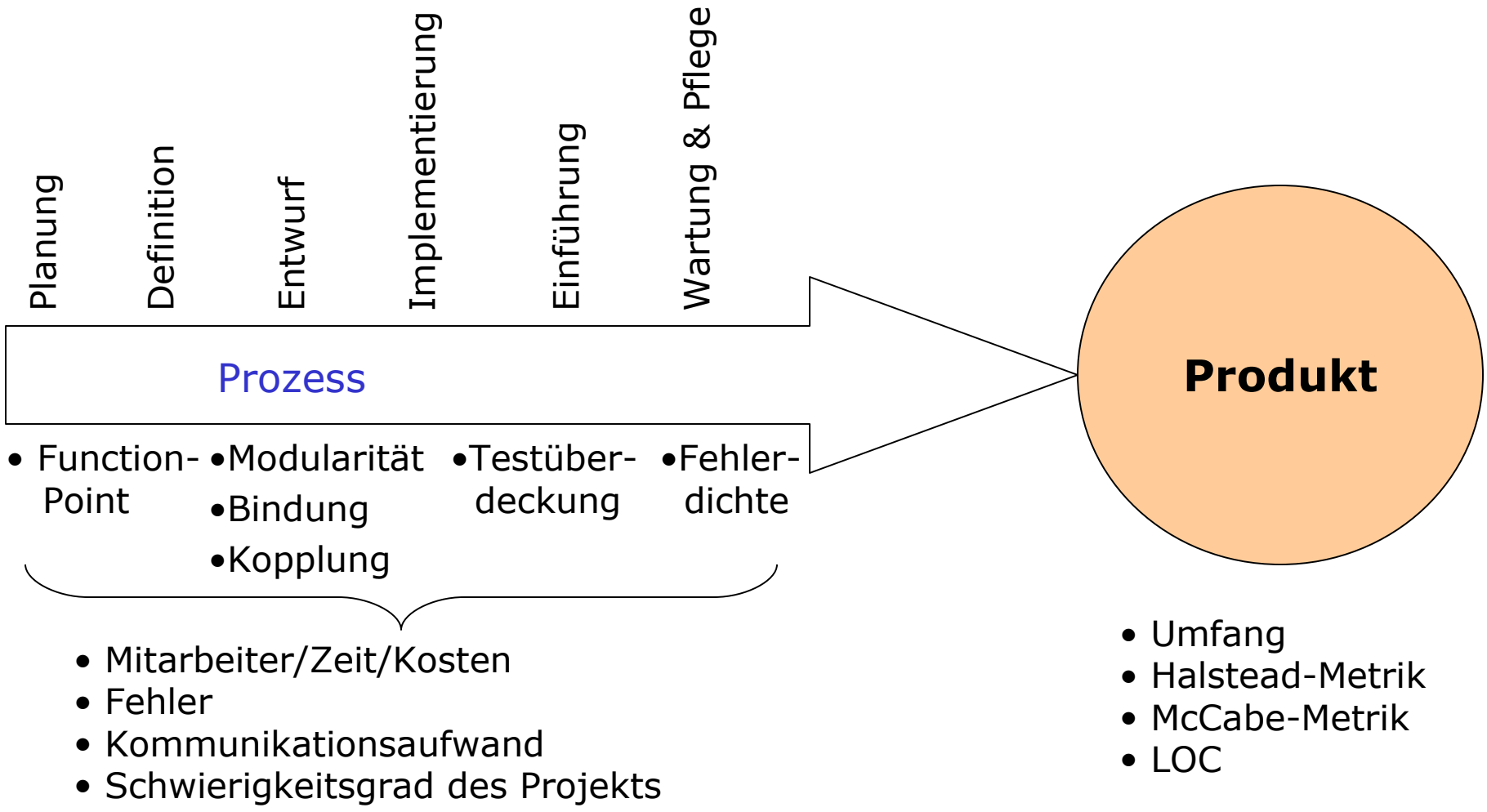
5. Loben und Tadeln

- Mitarbeiter, die den Plan und die Standards einhalten, sollten vom Manager gelobt, die anderen getadelt werden.
- Lob sollte sich unsbesondere durch nichtmonetäre Belohnungen ausdrücken, z.B. einen zusätzlichen freien Tag, eine Kongressreise u.ä..

- Die Aufgaben **1** und **2** sind in der Regel einmal für die gesamte Softwareentwicklung durchzuführen.
- Die Aufgaben **3** bis **5** sind permanent für jede Software-entwicklung durchzuführen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

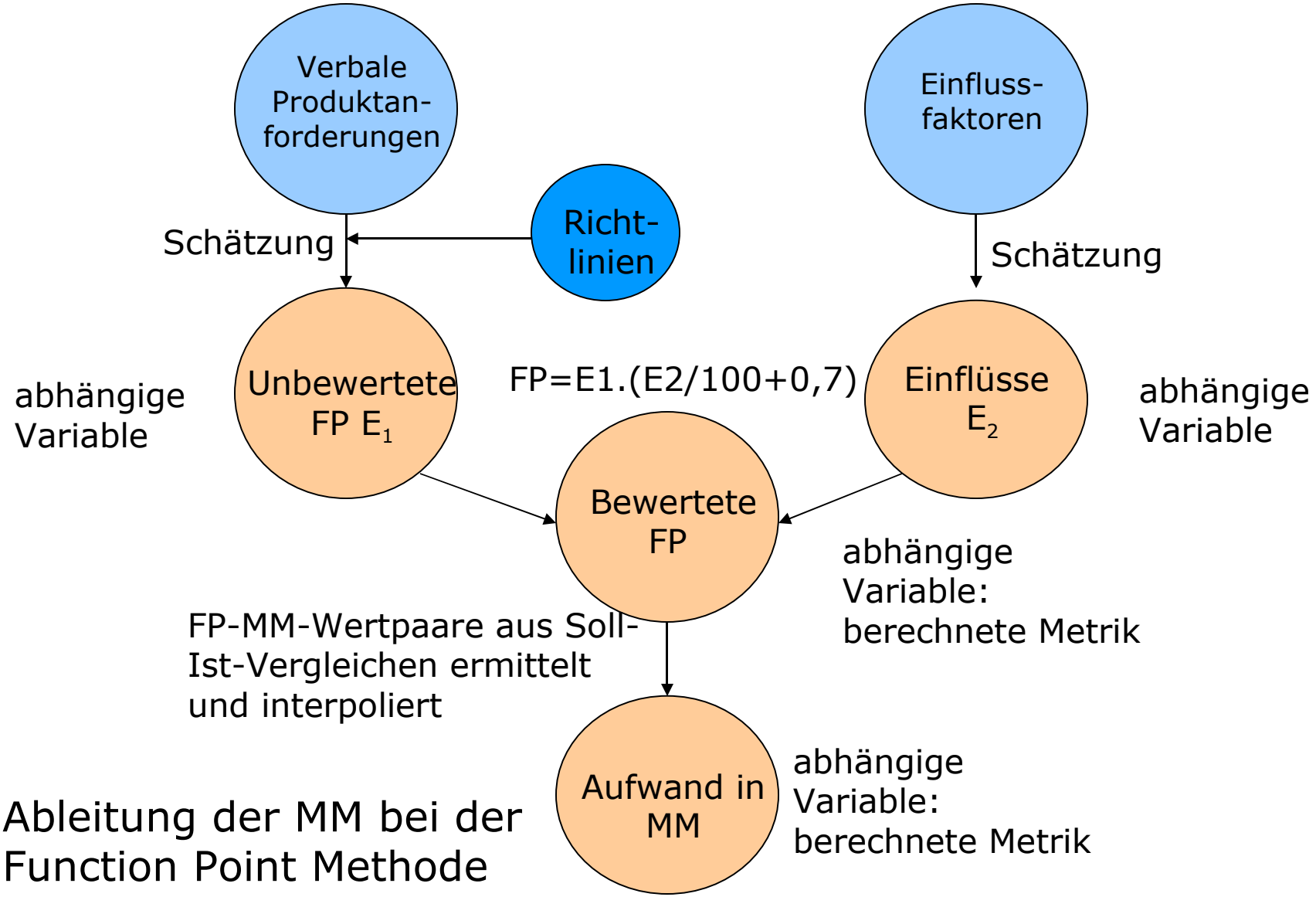
2. Metriken definieren, einführen und anwenden



Typische Kenngrößen von Software-Prozessen und Produkten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Function Point Methode (3)



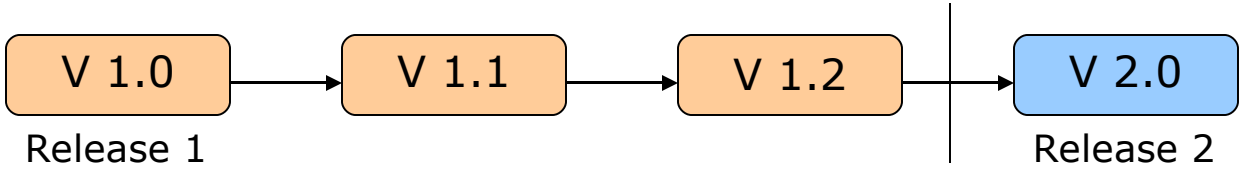
Ableitung der MM bei der Function Point Methode

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3.2 Versionen und ihre Verwaltung

Version: Ist gezeichnet durch die Ausprägung eines Software-Elements zu einem bestimmten Zeitpunkt. Unter Versionen werden zeitlich nacheinander liegende Ausprägungen eines Software-Elements verstanden und in der Regel durch eine Nummer beschrieben.

- Bestandteile einer Versionsnummer:
  - Release-Nummer
  - Level-Nummer



- Verbreitetstes Modell zur Versionenverwaltung: **Checkin/Checkout-Modell**. Software-Elemente werden in diesem Modell in Archiven gesammelt.
- Checkout-Operation: Holt eine Kopie des Elements aus dem Archiv und reserviert sie für den Ausbucher.
- Checkin-Operation: Befördert die geänderte Kopie ins Archiv und löscht die Reservierung und erledigt die Geschichtseinschreibungsoperationen (Autor, Zeit, ...).

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3.3 Varianten

- Verschiedene Versionen eines Software-Elements führen zu einem sequentiellen Versions-Stamm. Für die Praxis reichen solche Versions-Stämme aber nicht aus.
- Durch Varianten werden zusätzliche Anforderungen abgedeckt.
- Varianten können:
  1. Zeitlich nebeneinander liegende Ausprägungen von SW-Elementen;
  2. Verwendet werden, um parallele Entwicklungslinien darzustellen;
  3. Unterschiedliche, d.h. variante, Implementierung derselben Schnittstelle sein;
  4. Sich durch bedingte Übersetzungen unterscheiden;
  5. Auf unterschiedliche Hardware- und/oder Software-Konfigurationen zugeschnitten sein;
  6. Ab einem Abstraktionsniveau nicht mehr unterschieden werden.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3.4 Konfigurations- und Änderungsmanagement

Software-Konfigurationsmanagement: unterstützt die Identifikation, Initiierung und effiziente Verwaltung von Softwarekonfigurationen, durch geeignete Methoden, Werkzeuge und Hilfsmittel, um durch kontrollierte Änderungen die Entwicklung und Pflege eines Softwareprodukts zu erleichtern und transparent zu machen.

- Ziele des Software-Konfigurationsmanagement:
  - Sicherstellen der Sichtbarkeit, Verfolgbarkeit und Kontrollierbarkeit eines Produkts und seiner Teile im Lebenszyklus.
  - Überwachung der Konfigurationen während des Lebenszyklus.
  - Sicherstellung, dass jederzeit auf vorangegangene Versionen zurückgegriffen werden.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

3.4 Konfigurations- und Änderungsmanagement

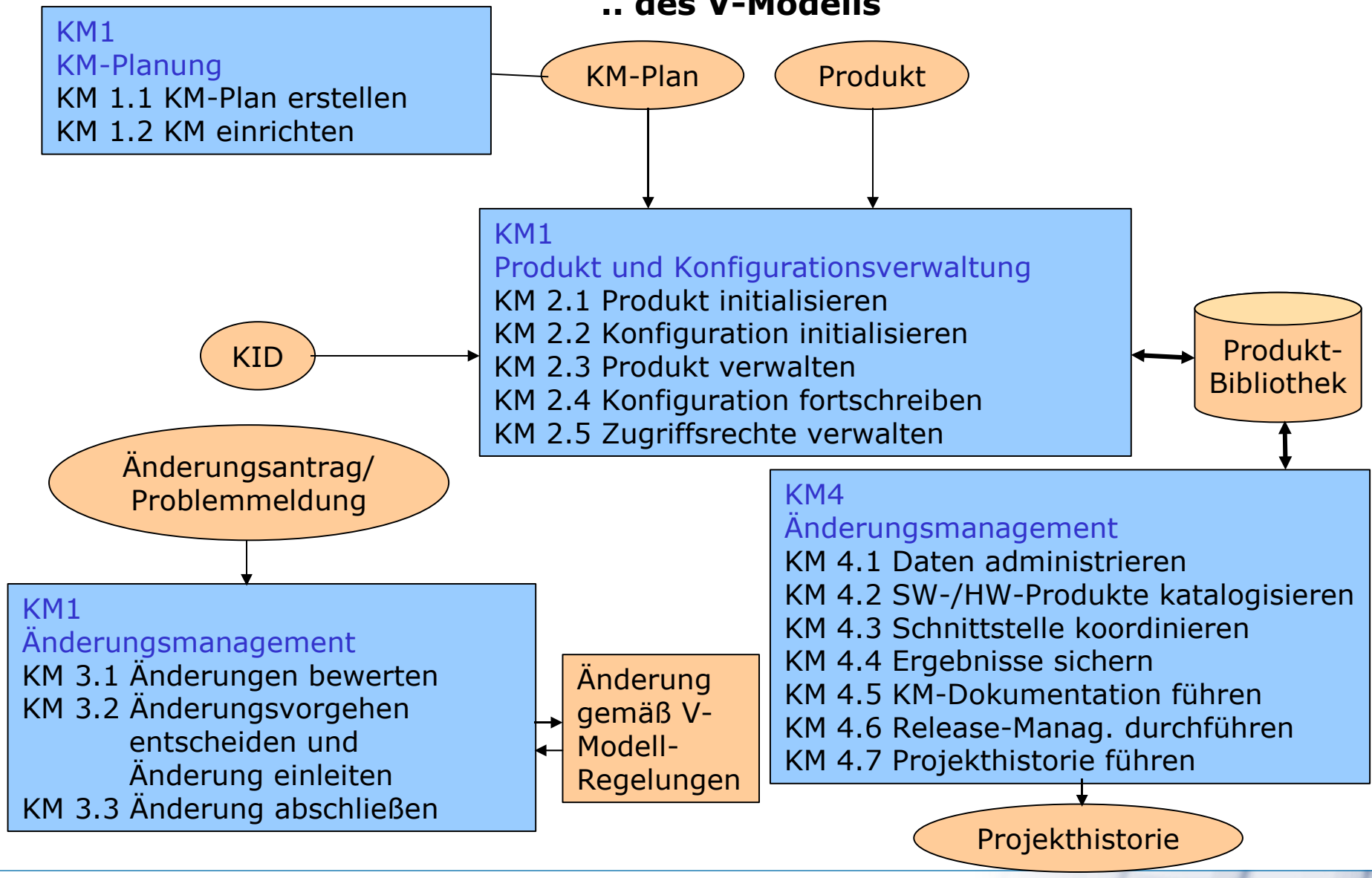
Aufgaben

1. Planung des Konfigurationsmanagement
2. Produkt- und Konfigurationsverwaltung
3. Änderungsmanagement
   a. Erfassung und Verwaltung eingehender Fehlermeldungen
   b. Entscheidung über die Bearbeitung von Änderungsanträgen
   c. Abschluss der Änderungen und Information aller Betroffenen
4. Konfigurationsmanagement-Dienste
   a. Daten Verwalten
   b. Soft-/Hardware-Produkte katalogisieren
   c. Schnittstelle koordinieren
   d. Ergebnisse sichern
   e. KM-Dokumentation führen
   f. Release-Management durchführen
   g. Projekthistorie führen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Das Submodell Konfigurationsmanagement ..

.. des V-Modells



Literatur

- [Boehm 91]
  Boehm B.W., Software Risk Management: Principles and Practices, IEEE Software
- [Itzfeld 83]
  Itzfeld W., Methodische Anforderungen an Software-Kennzahlen in Angewandte Information
- [Itzfeld, Schmidt, Timm 84]
  Itzfeld W., Schmidt M., Timm M.. Spezifikation von Verfahren zur Validierung von SW-Qualitätsmaßnahmen in Angewandte Informatik
- [Sommerville 89]
  Sommerville I., Software-Engineering –3rd ed. Addison-Wesley
- [Thayer 90]
  Thayer R.H., Tutorial Software Engineering Project Management, IEEE Society Press
- [Wallmüller 90]
  Wallmüller E., Software-Qualitätssicherung in der Praxis, Carl Hanser Verlag
- [Zuse 85]
  Zuse H., Messtheoretische Analyse von statischen Softwarekomplexitätsmaßen, Tu Berlin

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

Computer Aided Software Engineering

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

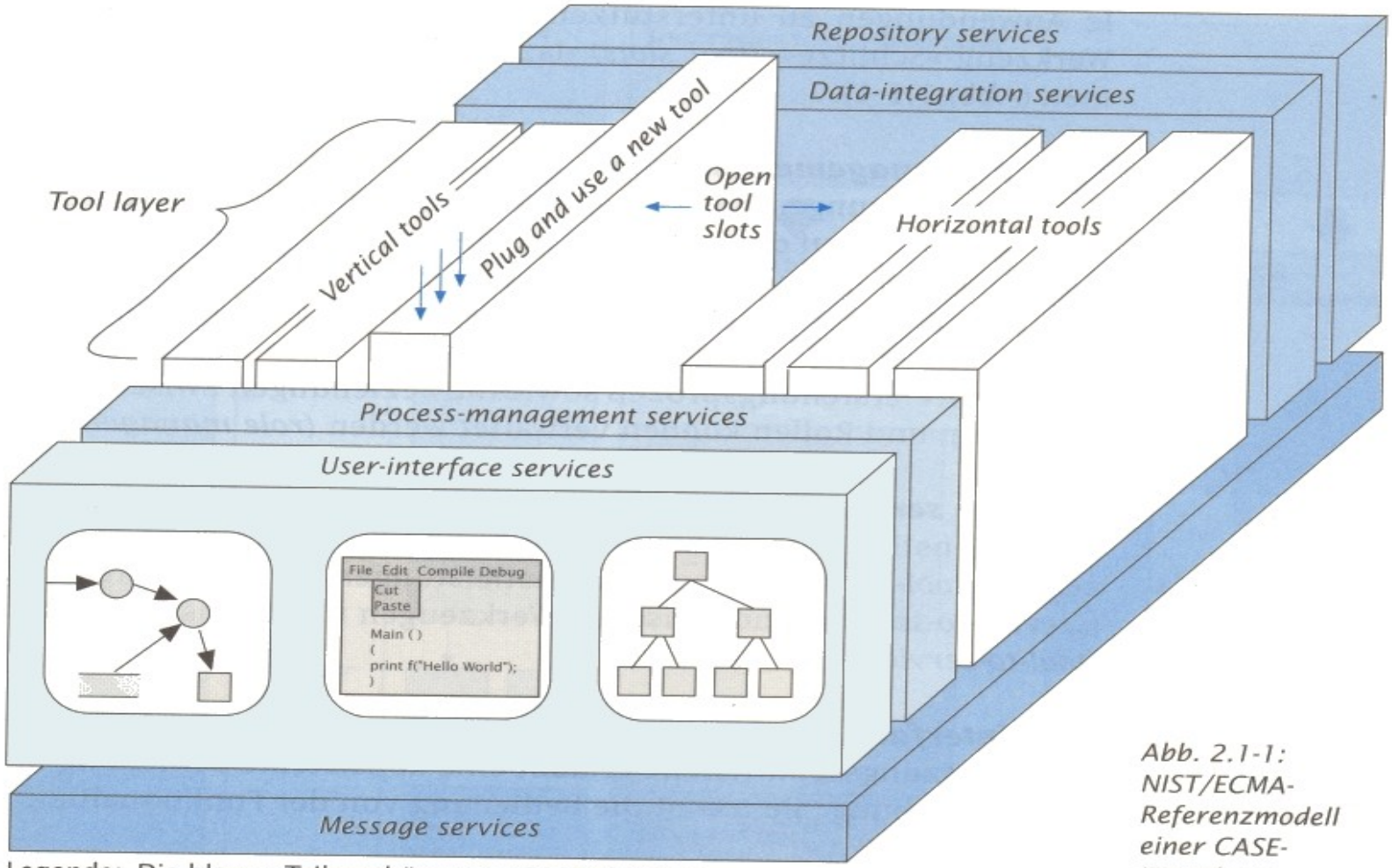
- **CASE**: Computer Aided Software Engineering

CASE befasst sich mit allen computerunterstützten Hilfsmitteln, die dazu beitragen, die SW-Produktivität und die SW-Qualität zu verbessern sowie das SW-Management zu erleichtern.

- CASE lässt sich gliedern in:
  - CASE-Werkzeuge:
    - Software-Produkte, die zumindest einzelne bei der SW-Erstellung benötigte Funktionen bzw. Dienstleistungen zur Verfügung stellen.
  - CASE-Plattformen:
    - Stellen Basisdienstleistungen, ein Repository und einen Nachrichtendienst zur Verfügung.
- Beide zusammen ergeben CASE-Umgebungen, auch Software-Entwicklungsumgebungen (SEU) genannt.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

NIST/ECMA- Referenzmodell



Legende: Die blauen Teile gehören zur CASE-Plattform
Vertical tools: Werkzeuge, die den gesamten Lebenszyklus begleiten, wie das Konfigurationsmanagement
Horizontal tools: Phasenorientierte Werkzeuge, z.B. SA-Werkzeug

Abb. 2.1-1: NIST/ECMA-Referenzmodell einer CASE-Umgebung (»Toaster«-Modell) /Chen, Norman 92, S. 19/

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Engineering

- Beim **Forward Engineering** ist das fertige Software-System das Ergebnis des Entwicklungsprozesses.
- Zum **Reverse Engineering** gehört das Extrahieren von Konstrukten und das Erstellen oder Synthetisieren von Abstraktionen.

  Reverse Engineering kann in jeder Entwicklungsphase gestartet werden. Das vorhandene SW-System ist der Ausgangspunkt der Analyse.
- **Re-Engineering** umfasst die Überprüfung und den Umbau des vorhandenen Systems, so dass eine Wiederherstellung in neuer Form erreicht wird.
- **Round Trip Engineering** bedeutet, an einer beliebigen Stelle als Ausgangspunkt beginnen und an einer beliebigen Stelle enden können (Einsatz von Forward- und Reverse Engineering).

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Basisanforderungen an CASE-Umgebungen

1. Vollständigkeit

- Eine vollständige CASE-Umgebung unterstützt die zwei Tätigkeitsgruppen:
  - Tätigkeiten, die bei fast allen Prozessmodellen auftreten,
  - Prozessmodellspezifische Tätigkeiten.
- Eine CASE-Umgebung heißt partiell vollständig, wenn sie einen Teilbereich der Software mit semantisch integrierten Hilfsmitteln unterstützt.

2. Inkrementeller Einsatz

- Die überwiegende Mehrzahl der CASE-Umgebungen sind nur partiell vollständig und nicht inkrementell einführbar.
- Viele Anwender stellen sich eigene, nicht-integrierte CASE-Werkzeuge aus verschiedenen Werkzeuggruppen zusammen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Ziele von CASE

Technische Ziele

- Generatoren einsetzen, um Arbeitsschritte zu eliminieren.
- Methoden einsetzen, die Konsistenz- und Redundanz-überprüfungen ermöglichen.
- Qualität der Dokumentation verbessern.
- Wiederverwendbarkeit erleichtern.
- Verwalten von Konfigurationen und Änderungen.
- Hinweise auf mögliche Schwachstellen (Metriken).

Wirtschaftliche Ziele

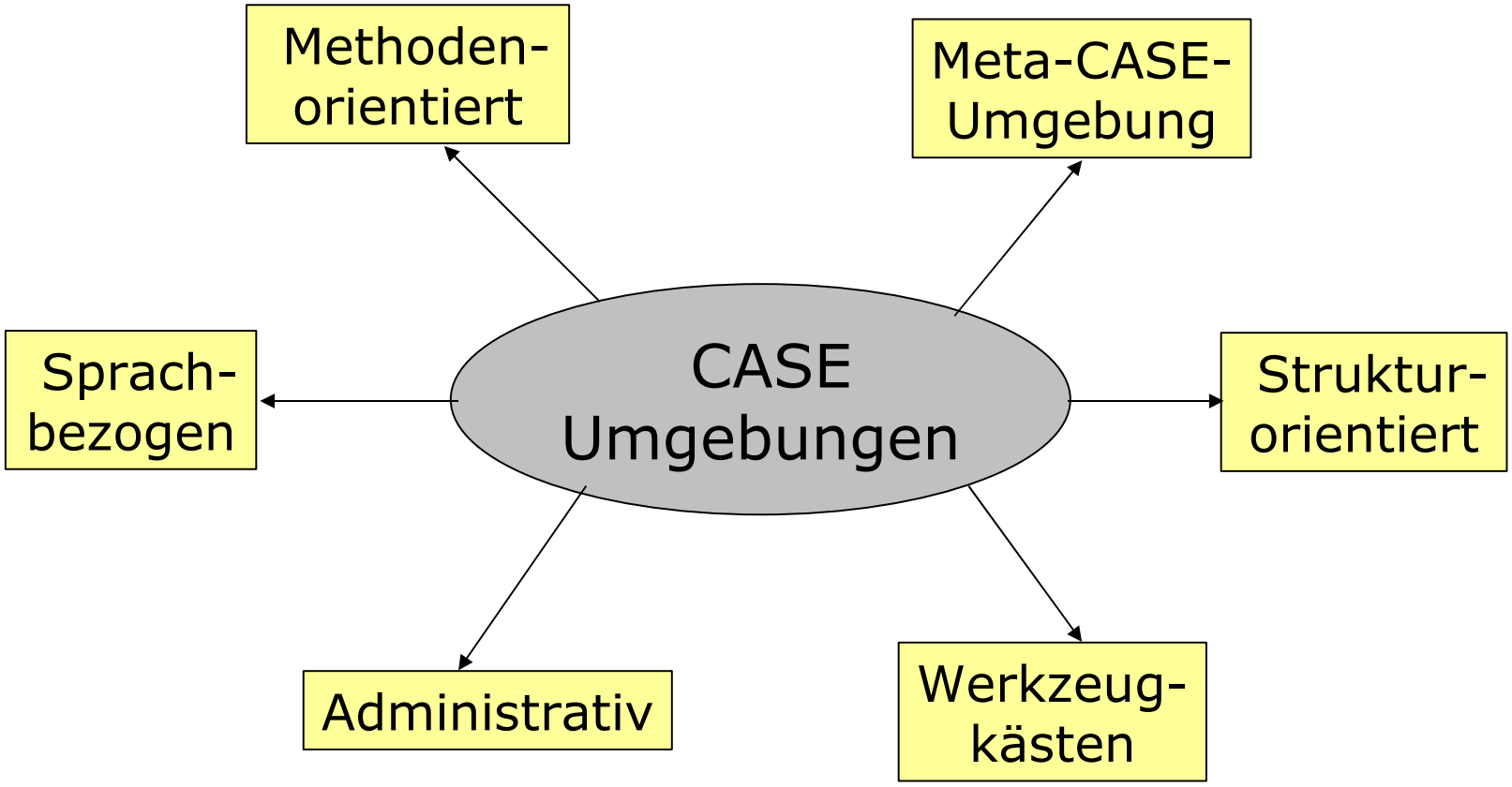
- Effektivität erhöhen.
- Produkte schneller entwickeln.
- Qualität erhöhen.
- Wartungsaufwand reduzieren.
- Personenanbhängigkeit verringern.
- Änderungen noch kurz vor Markteinführung ermöglichen.

Organisa-torische Ziele

- Unterstützung des gewählten Prozessmodells.
- Verbesserung der Entwicklungsmethoden und –verfahren.
- Erhöhung der Standardisierung.
- Jederzeit Information über den Ist- und Soll-Zustand.
- Flexible Anpassung an geänderte Rahmenbedingungen.

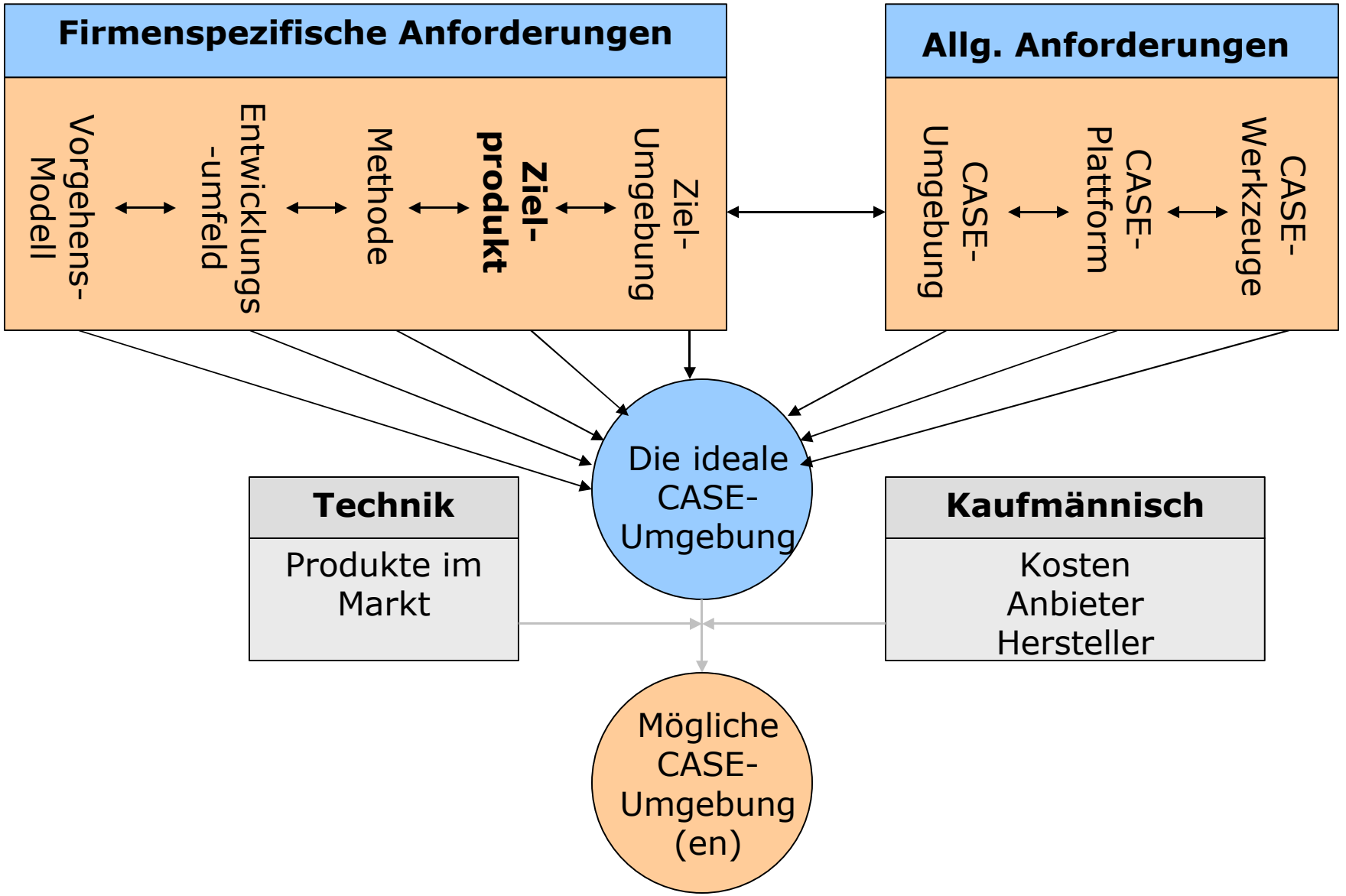
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Klassifikation von CASE-Umgebungen



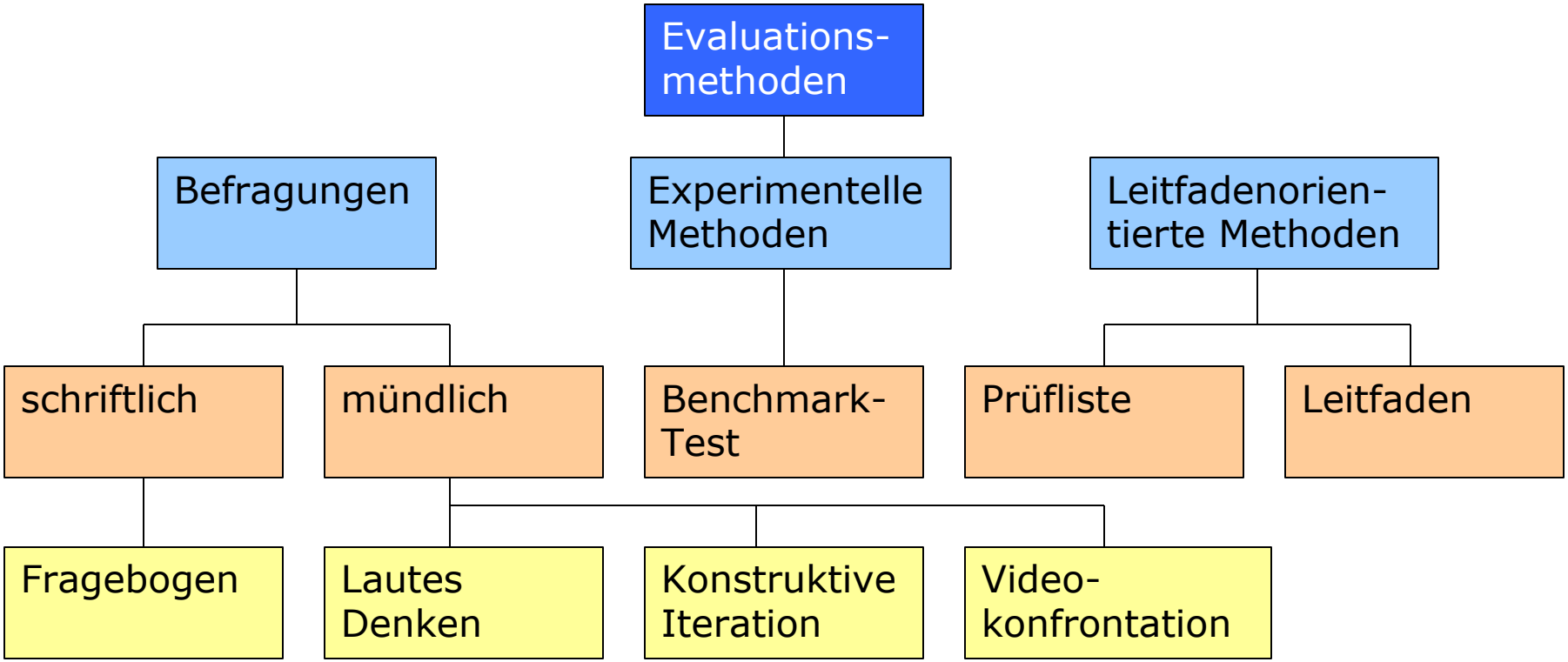
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Bestimmende Faktoren der CASE-Auswahl



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Überblick über Evaluationsmethoden



Überblick über Evaluationsmethoden [Oppermann et al. 92]

Literatur

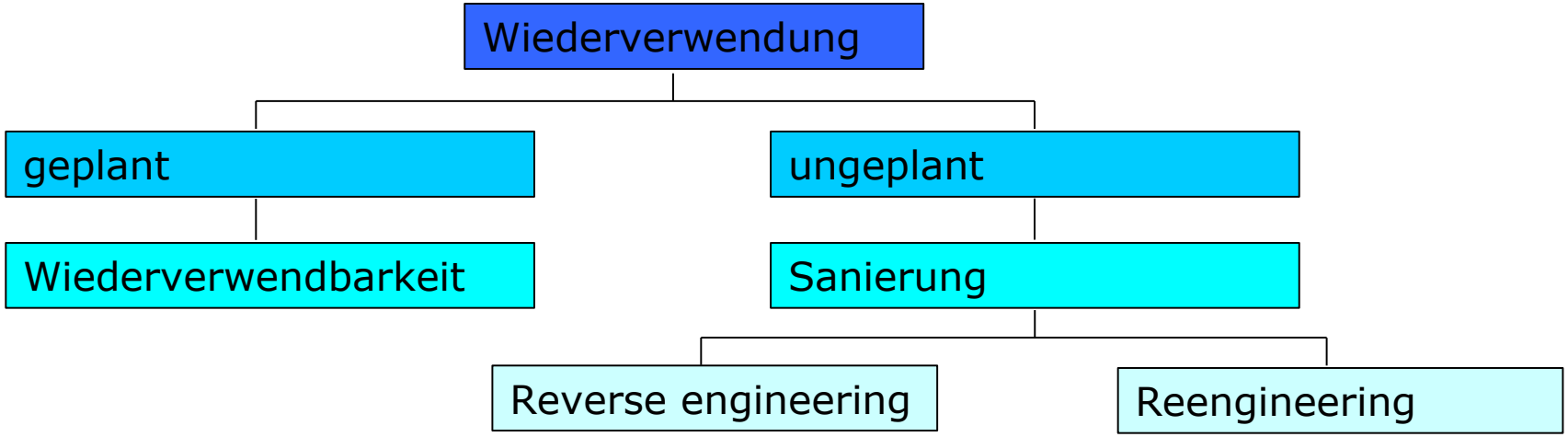
- [Herzwurm, Hierholzer, Kunz 93]
  Produktivität stieg bis um 600%-Uni Köln: CASE-Tools auf dem Prüfstand, in: Computer Zeitung
- [Ramming, Steinmüller 93]
  Ramming F.J., Steinmüller B., Frameworks und Entwurfsumgebungen, in: Informatikspektrum, 15, 1992, S. 33-43
- [Microtool 96]
  Microtool, Eine bedarfsgerechte SEU aus Komponenten – die Architekturlösung von Microtool, Microtool GmbH
- [Oppermann et al. 92]
  Oppermann R., Murchner B., Reiterer H., Koch M. Software-ergonomische Evaluation- Der Leitfaden EVADIS II, Walter de Gruyter
- [Wassermann 90]
  Wassermann A.I.. Tool Integration in Software Engineering Environments, in Software Engineering Environments, Springer-Verlag

Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

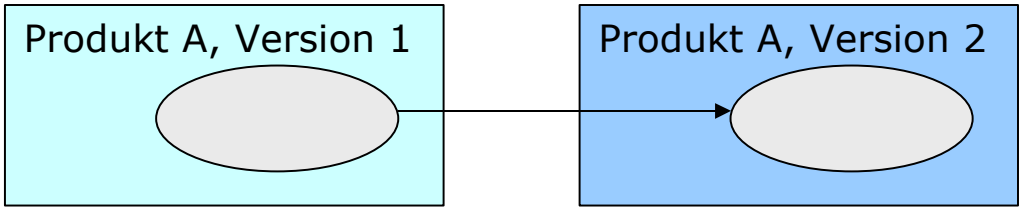
Arten der Wiederverwendung



- Üblich ist die ungeplante Wiederverwendung.
- Es wird versucht, Software-Komponente alter Systeme mit Methoden des Reverse Engineering, des Reengineering und der Sanierung zu identifizieren und aufzubereiten.

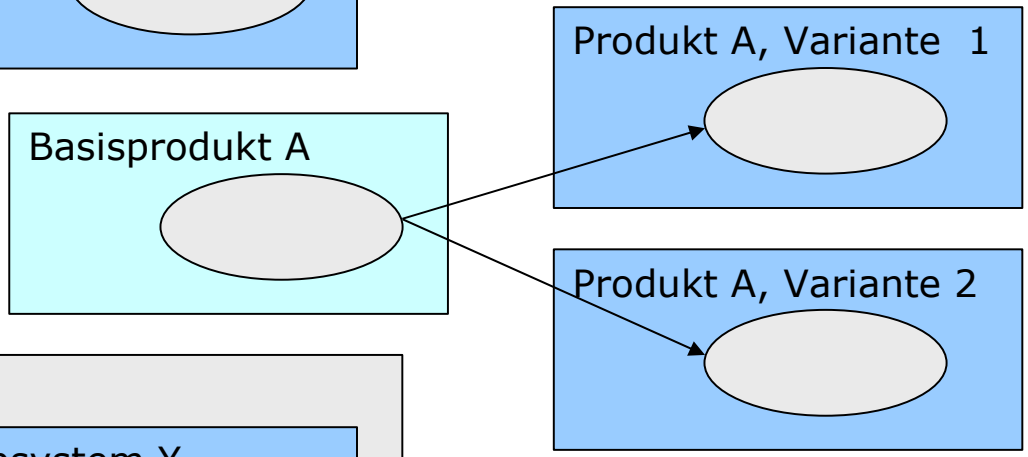
2. Wiederverwendbarkeit und Wiederverwendung

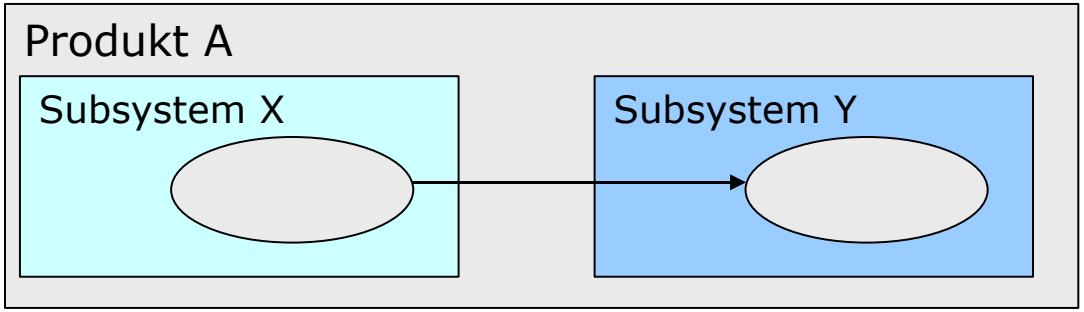
- Bei der geplanten Wiederverwendung geht es darum, Software-Komponenten von vornherein wiederverwendbar so zu gestalten.
- Gute wiederverwendbare Software-Komponenten:
  - besitzen einen hohen Allgemeinheitsgrad,
  - sind qualitativ hochwertig und
  - sind gut dokumentiert.
- 4 Typen der Wiederverwendung:
  - Wiederverwendung bei Versionsentwicklung,
  - Wiederverwendung bei Variantenentwicklung,
  - intra-produktorientierte Wiederverwendung,
  - inter-produktorientierte Wiederverwendung.



Wiederverwendung bei Versionsentwicklung

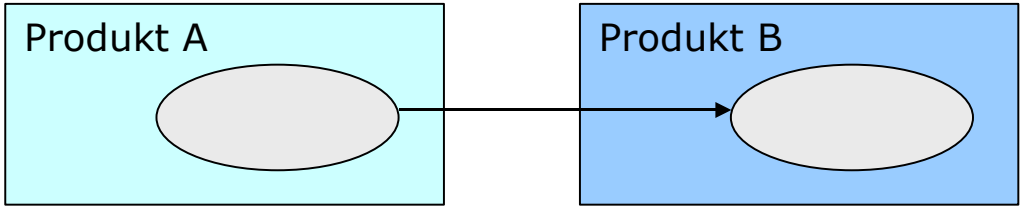
Wiederverwendung bei Variantenentwicklung





Intraproduktorientierte Wiederverwendung

Interproduktorientierte Wiederverwendung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Literatur

- [Herzwurm, Hierholzer, Kunz 93]
  Produktivität stieg bis um 600%-Uni Köln: CASE-Tools auf dem Prüfstand, in: Computer Zeitung
- [Ramming, Steinmüller 93]
  Ramming F.J., Steinmüller B., Frameworks und Entwurfsumgebungen, in: Informatikspektrum, 15, 1992, S. 33-43
- [Microtool 96]
  Microtool, Eine bedarfsgerechte SEU aus Komponenten – die Architekturlösung von Microtool, Microtool GmbH
- [Oppermann et al. 92]
  Oppermann R., Murchner B., Reiterer H., Koch M. Software-ergonomische Evaluation- Der Leitfaden EVADIS II, Walter de Gruyter
- [Wassermann 90]
  Wassermann A.I.. Tool Integration in Software Engineering Environments, in Software Engineering Environments, Springer-Verlag

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

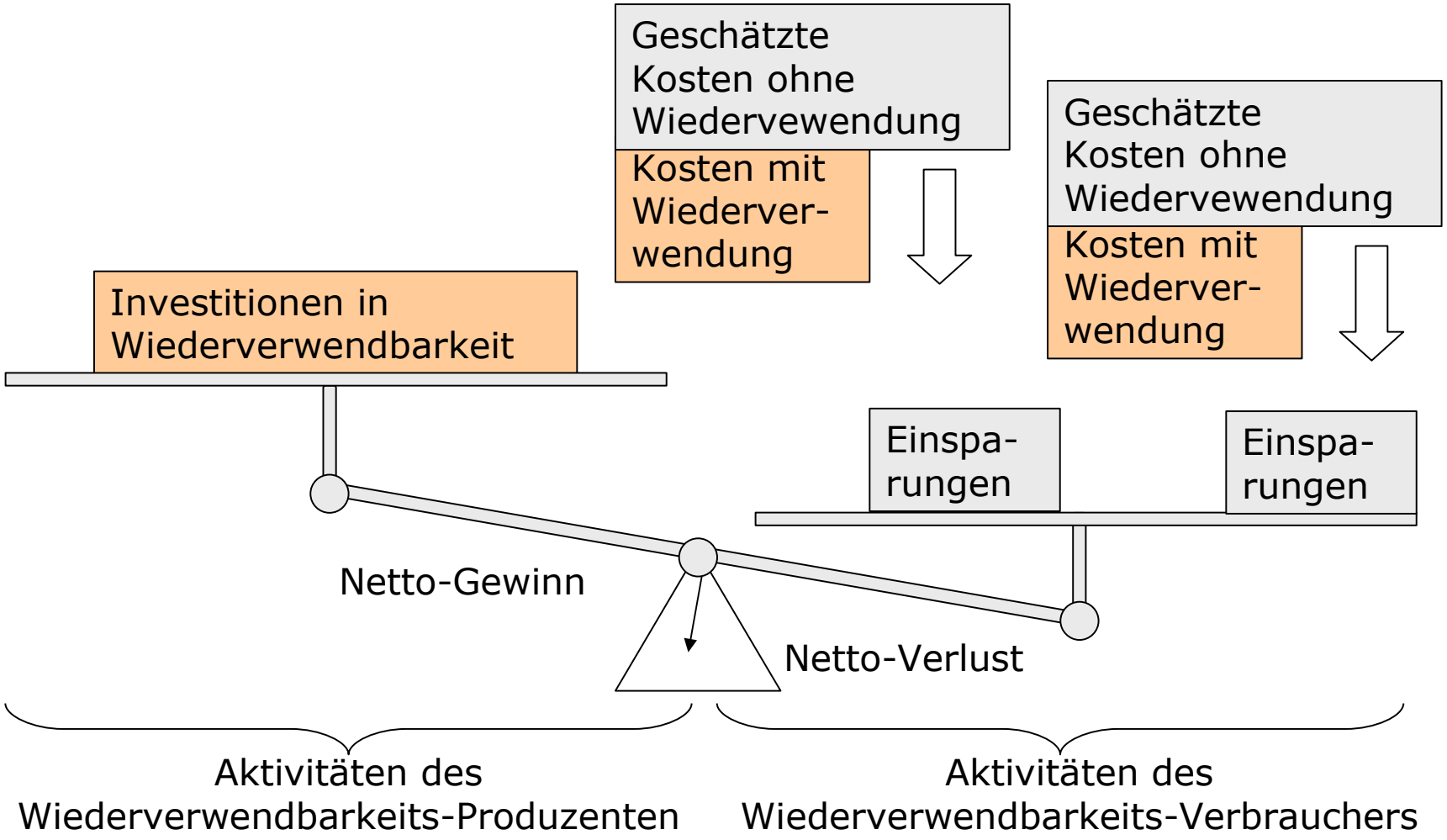
Faustregeln

- *Formel 3* (nach [Lanergan, Grasso 84], [Biggerstaff 94]):
  - Software muss dreimal eintwickelt werden, bevor sie wirklich wiederverwendbar entwickelt werden kann.
  - Bevor die Früchte der Wiederverwendung geerntet werden können, muss Software dreimal wiederverwendet werden.
- Nach [Levine 93] sind die Erstellungskosten von wieder-verwendbaren Komponenten 60% höher. Diese sind folgendermaßen aufgeteilt:
  - 25% für zusätzliche Verallgemeinerung,
  - 15% für zusätzliche Dokumentation,
  - 10% für zusätzliches Testen,
  - 5% Mehraufwand für Archivablage und Wartung.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

6. Kosten/Nutzen der Wiederverwendung

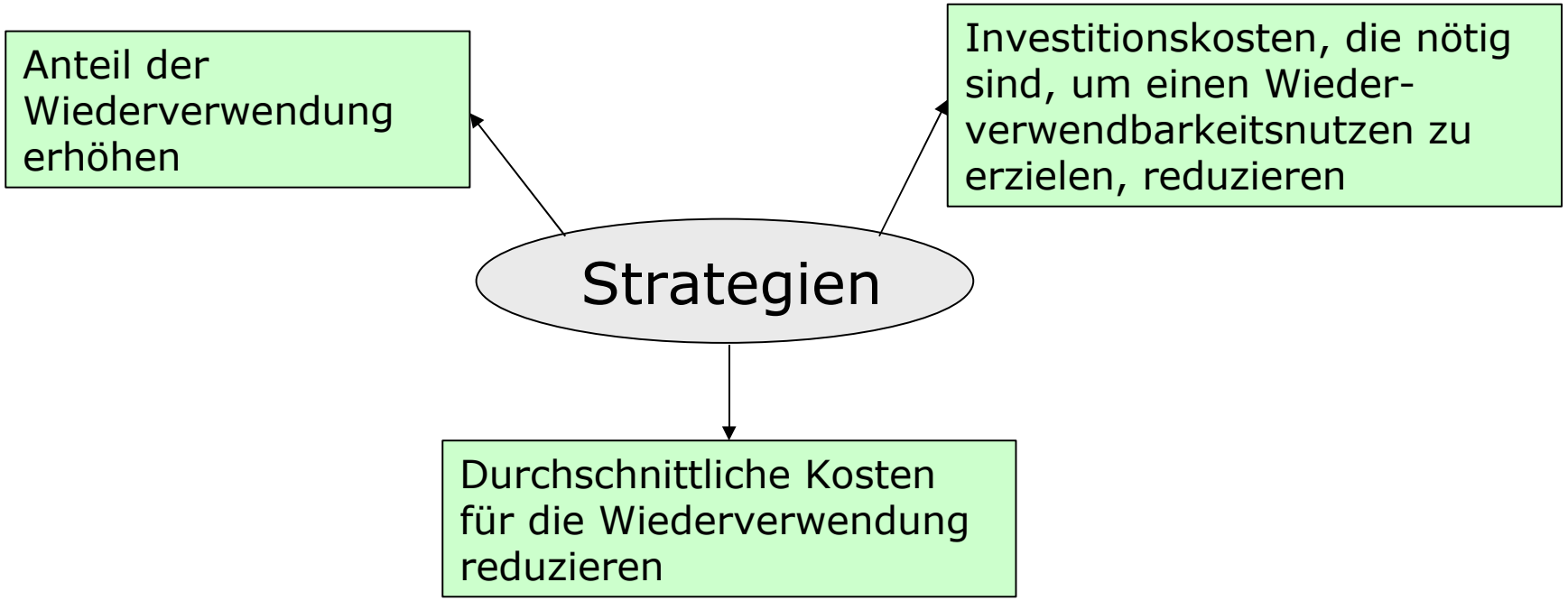
Kosten/Nutzen Relation der Wiederverwendung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Kosteneffektive Wiederverwendung

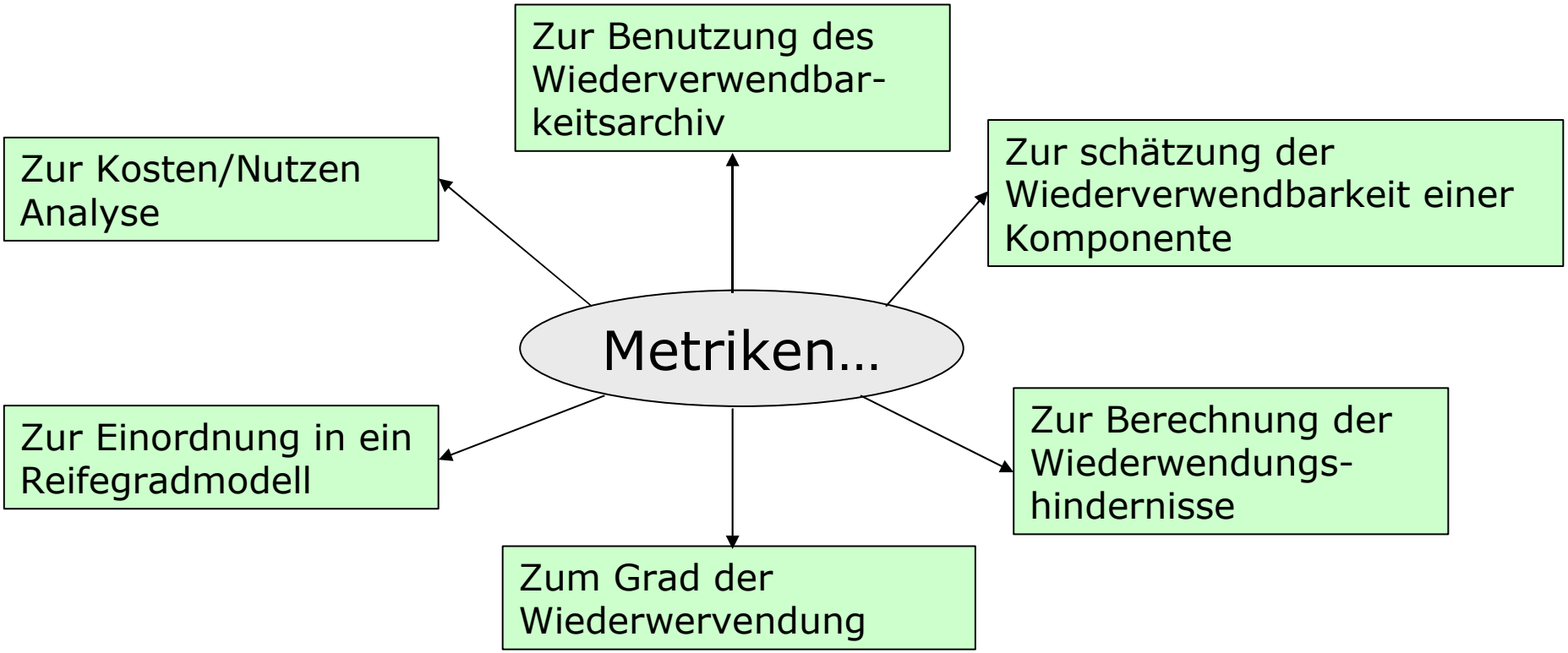
- Um die Wiederverwendung kosteneffektiv zu machen, muss R möglichst groß gemacht werden.
- 3 mögliche Strategien:



6. Kosten/Nutzen der Wiederverwendung

Kosteneffektive Wiederverwendung (2)

- 6 mögliche Metrikkategorien zum Messen verschiedener Aspekte der Wiederverwendung nach [Frakes, Terry 96].

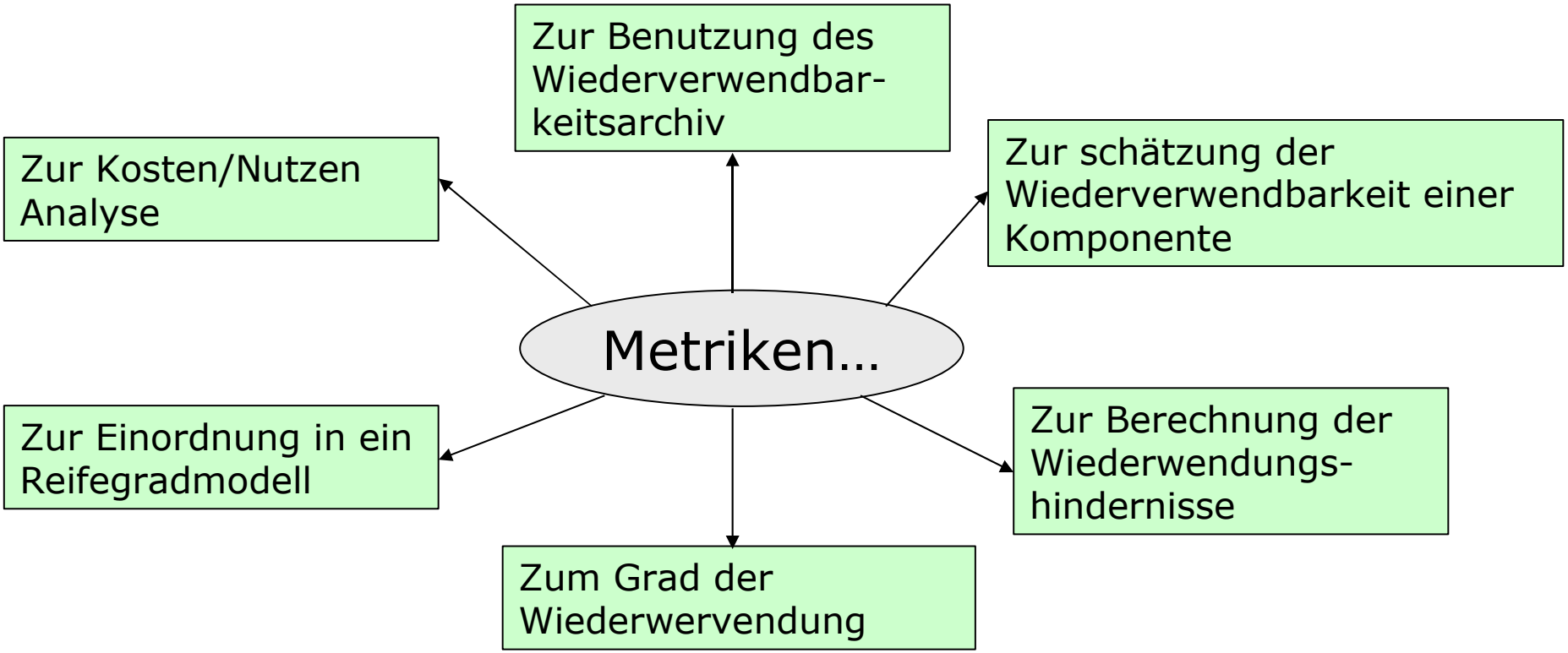


UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

6. Kosten/Nutzen der Wiederverwendung

Kosteneffektive Wiederverwendung (2)

- 6 mögliche Metrikkategorien zum Messen verschiedener Aspekte der Wiederverwendung nach [Frakes, Terry 96].



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Fakten (nach [Sneed 92a])

- Die Wartungskosten eines durchschnittlichen Anwender-unternehmens liegt zwischen 50 und 75% des gesamten DV-Budgets.
- Nur 30% sind individueller, problemspezifischer Natur. Der Rest ist softwaretechnisch identisch und ließe sich mit entsprechenden Standards abdecken.
- Ein typischer Anwender in den USA hat durchschnittlich 2200 Programme mit 1,15 Millionen Anweisungen, die 40 bis 50 Anwendungen abdecken.
- Ein typisches Programm in den USA ist meistens in COBOL geschrieben, lebt 5 bis 7 Jahre und enthält 700 Anweisungen.
- Ein Programm, das ein hierarchisches oder netzwerkorientiertes DBS benutzt, wird jährlich zwischen 10 und 20-mal angepasst.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

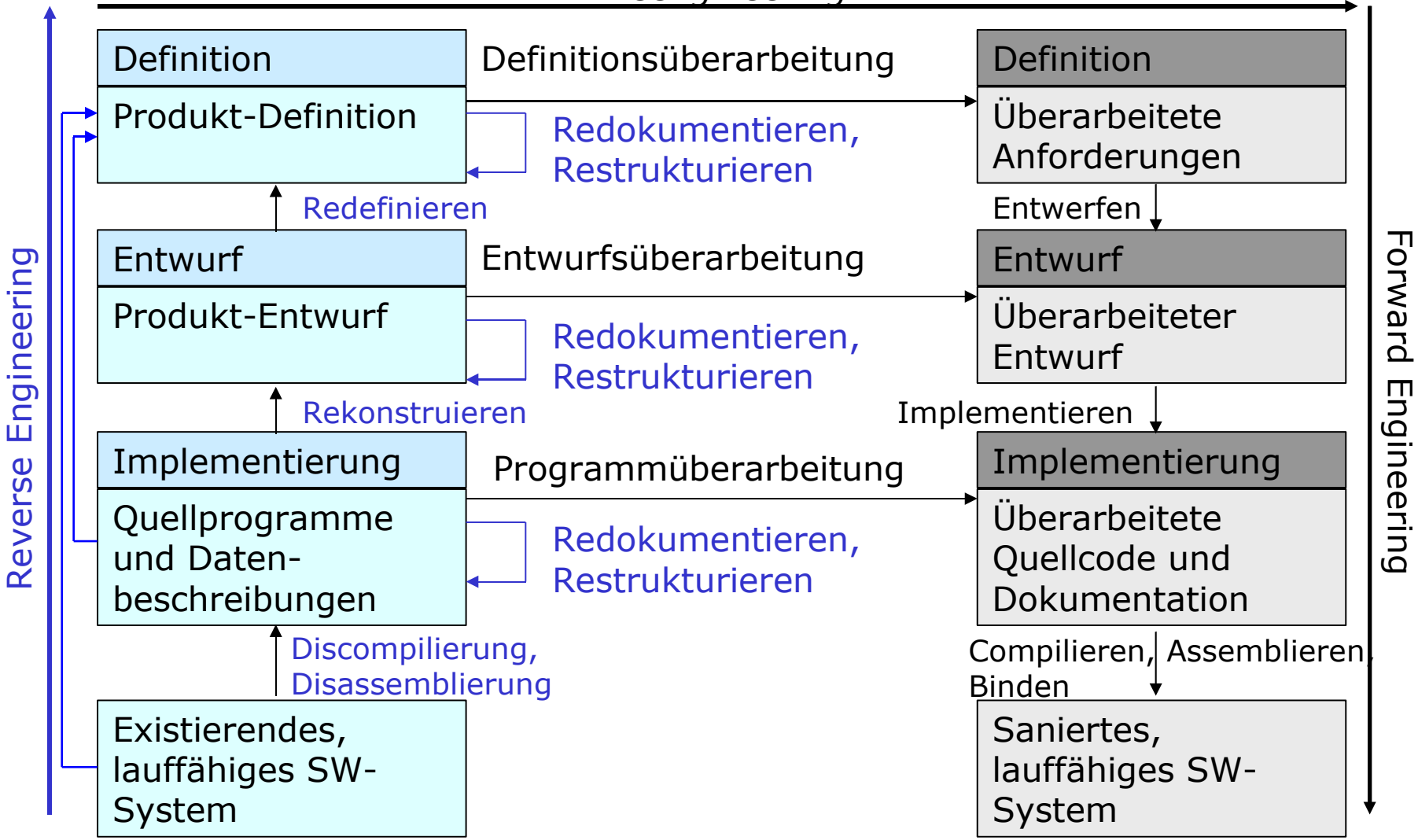
Fakten (nach [Sneed 92a]) (2)

- Deutsche Cobol-Programme sind zu 80% monolithisch, zu 77% unstrukturiert und enthalten zu 93% überflüssige, redundant gehaltene Daten.
- Ein Cobol-Programmierer verwaltet ca 1100 Data-Division-Anweisungen, 2000 Procedure-Division-Anweisungen und 270 Sprunganweisungen.
- Die Komplexität eines unstrukturierten Programms erhöht sich nach einer Korrektur um 4%, nach einer Änderung um 17% und nach einer Erweiterung um 26%.
- Ein Wartungsprogrammierer benötigt 47% seiner Zeit für die Programmanalyse, 15% für die Programmierung, 28% für den test und 9% für die Dokumentation.
- Die Wartungskosten je geänderte Zeile sind bei kleinen Anwendungen am höchsten.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

2. Konzepte und ihre Terminologie

Terminologie der Software-Sanierung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Definitionen

Forward Engineering: stellt den normalen Entwicklungsablauf bei der Neuentwicklung von Softwaresystemen dar.

Reverse Engineering: Hierbei wird versucht, aus einem fertigen Produkt, z.B. einem Prozessor oder einem Schaltkreis, die Produktspezifikation abzuleiten.

Re-engineering (Renovierung): Umfasst alle Aktivitäten zur Änderung von Software-Altsystemen, um sie in einer neuen Form wieder implementieren zu können.

Sanierung: Umfasst alle notwendigen Reverse Engineering; Reengineering und Forward Engineering-Maßnahmen, um ein saniertes lauffähiges System zu erhalten, das definierte neue Ziele erfüllt.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Verfahren der Sanierung

- Kategorien von Verfahren zur Realisierung der verschiedenen Konzepte der Sanierung:
  - Verpacken (*wrapping*),
  - Verstehen und
  - Verbessern von Altsystemen.
- Hat man mittels Reverse Engineering ein ausreichendes Verständnis und eine angemessene Dokumentation der jeweiligen Abstraktionsebene erreicht, dann kann auf dieser Grundlage das Altsystem verbessert werden.
- Eine Konvertierung eines Altsystem in eine neue Programmiersprache ohne Veränderung der Systemarchitektur erfordert nur die Transformierung des Codes in eine neue Programmiersprache

Möglichkeiten

- Prinzipiell gibt es 3 Möglichkeiten:
  - Verstehen des Systems als Black box,
  - Verstehen des Systems als White box,
  - Mischformen der ersten beiden Möglichkeiten.
- Das Verstehen des Systems als Blackbox ist in vielen Fällen ausreichend. Dazu werden 3 Engineering-Formen verwendet:
  - Reverse Engineering: Erstellen eines OOA-Modells des Altsystems
  - Reengineering: Überarneitung, Verbesserung und Erweiterung des entstandenen OOA-Modells.
  - Forward Engineering: Entwicklung eines neuen Systems ausgehend vom OOA-Modell.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

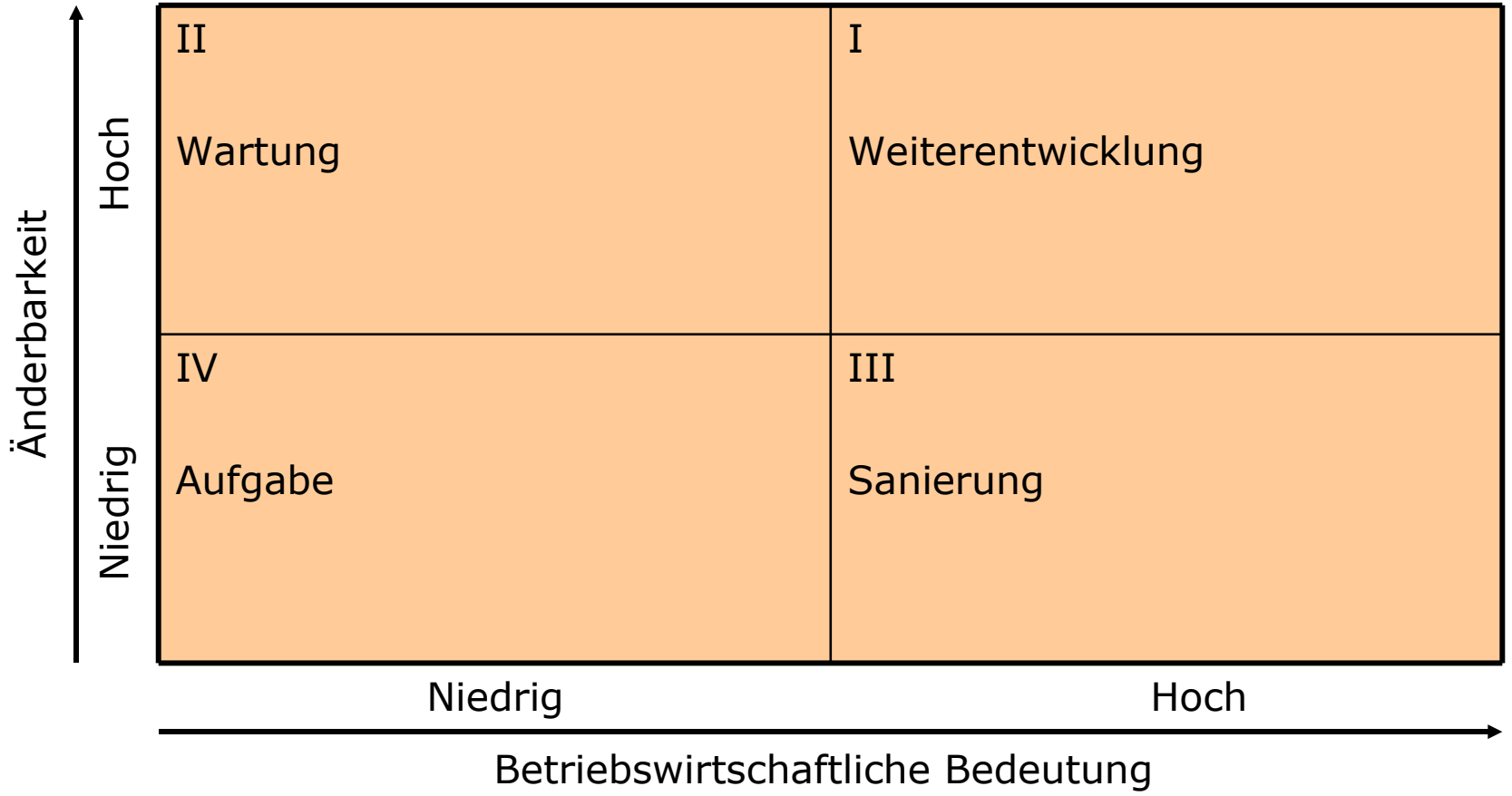
Verfahren der Sanierung

- Kategorien von Verfahren zur Realisierung der verschiedenen Konzepte der Sanierung:
  - Verpacken (*wrapping*),
  - Verstehen und
  - Verbessern von Altsystemen.
- Hat man mittels Reverse Engineering ein ausreichendes Verständnis und eine angemessene Dokumentation der jeweiligen Abstraktionsebene erreicht, dann kann auf dieser Grundlage das Altsystem verbessert werden.
- Eine Konvertierung eines Altsystem in eine neue Programmiersprache ohne Veränderung der Systemarchitektur erfordert nur die Transformierung des Codes in eine neue Programmiersprache

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

4. Kosten/Nutzen der Sanierung

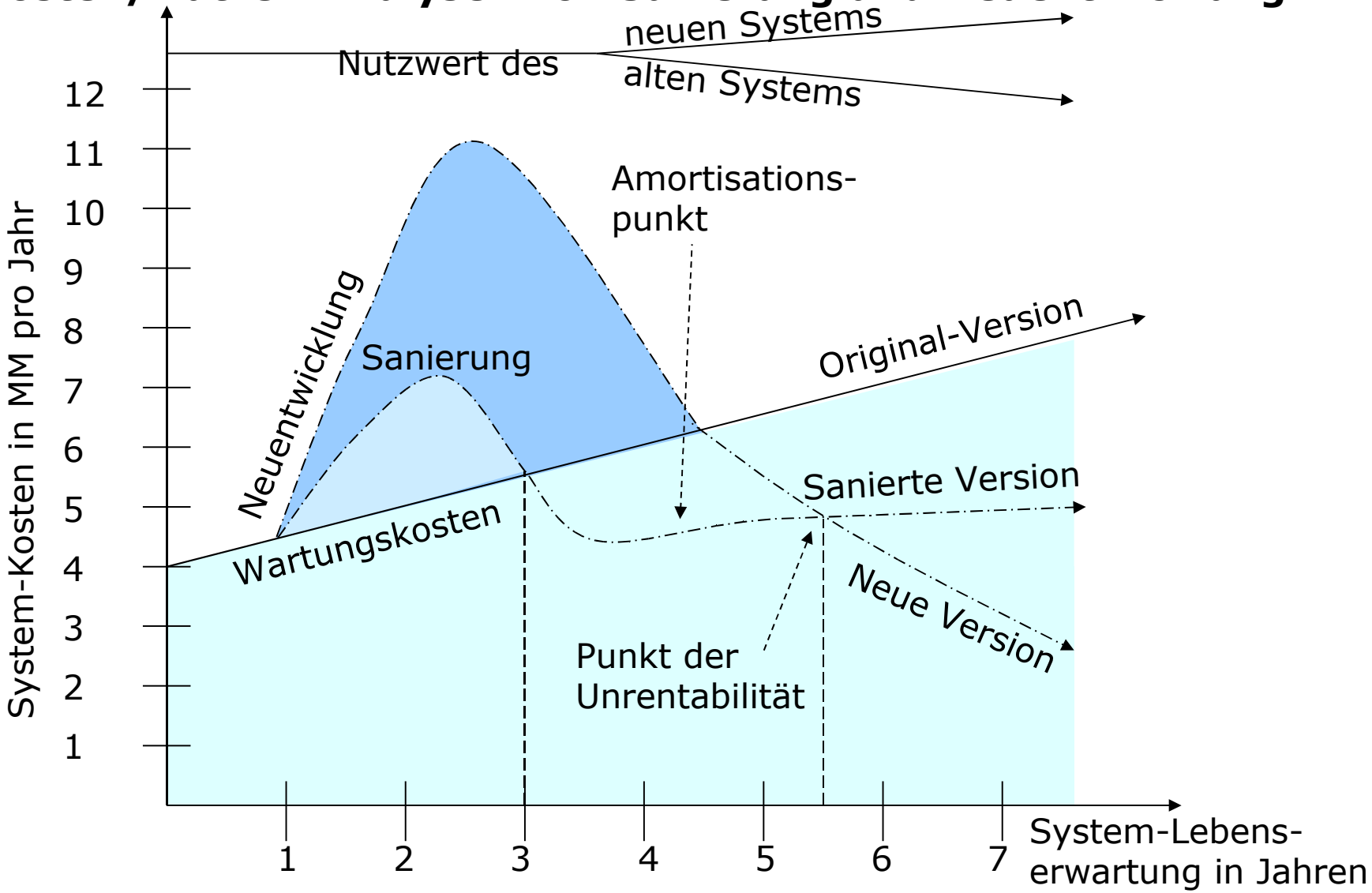
Portfolioanalyse der Altsysteme (nach [Jacobson, Lindström 91])



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

4. Kosten/Nutzen der Sanierung

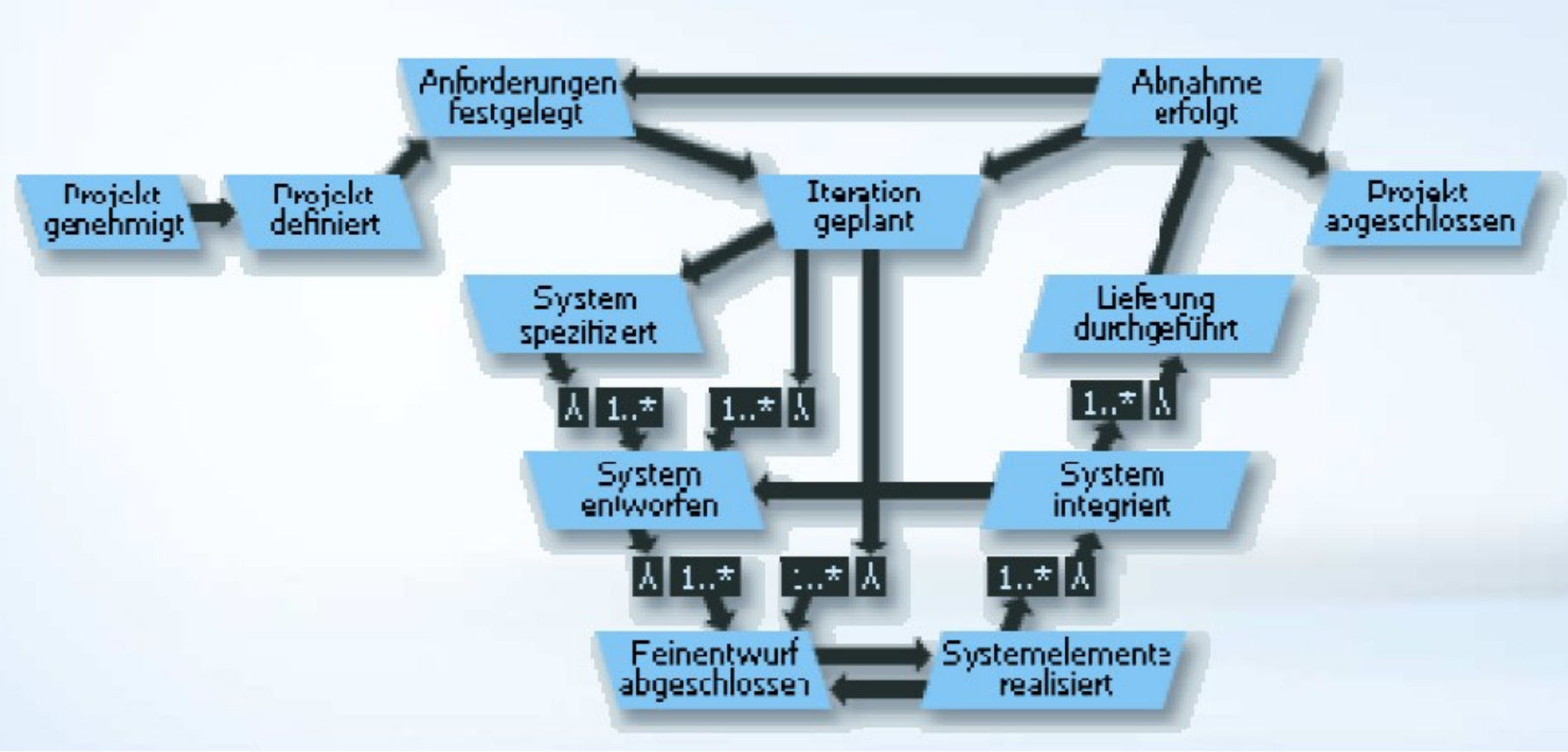
Kosten/Nutzen-Analysen von Sanierung und Neuentwicklung



Systementwicklung AG/AN

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

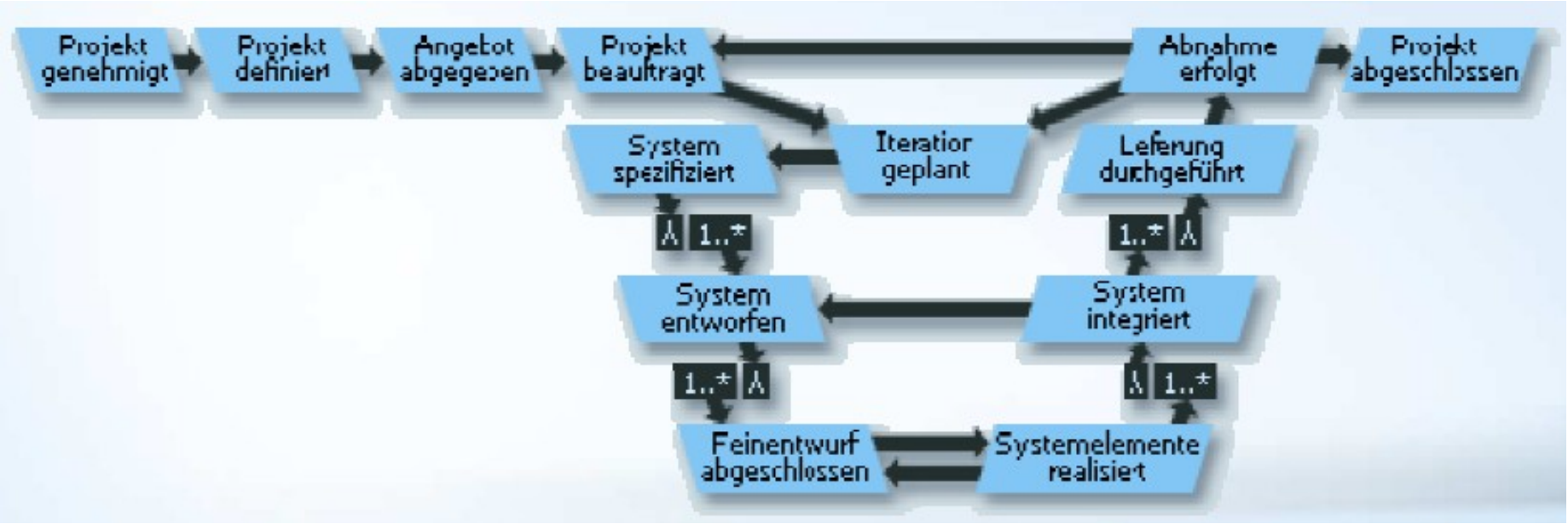
- Wartung und Pflege von Systemen (AG/AN)



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Projektdurchführungsstrategien

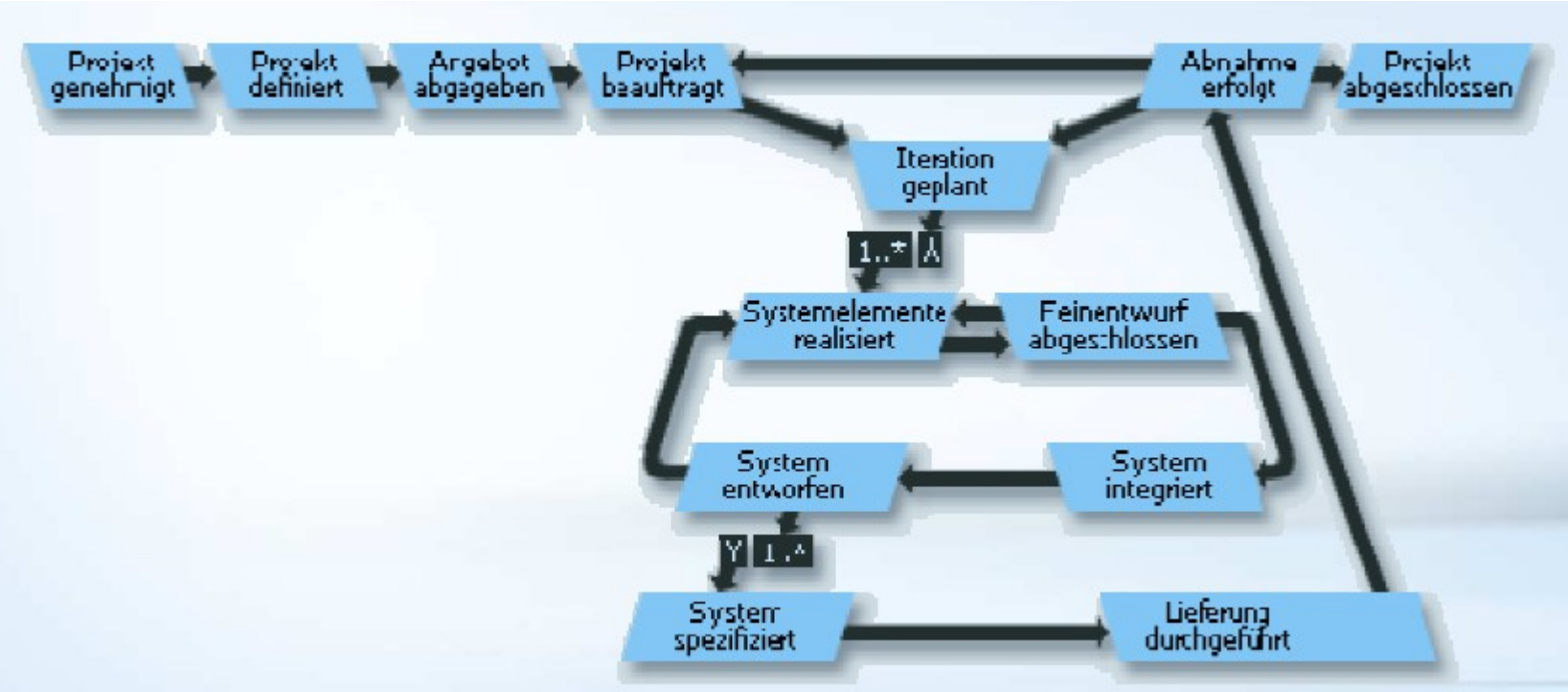
- Inkrementelle Systementwicklung (AN)



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Projektdurchführungsstrategien

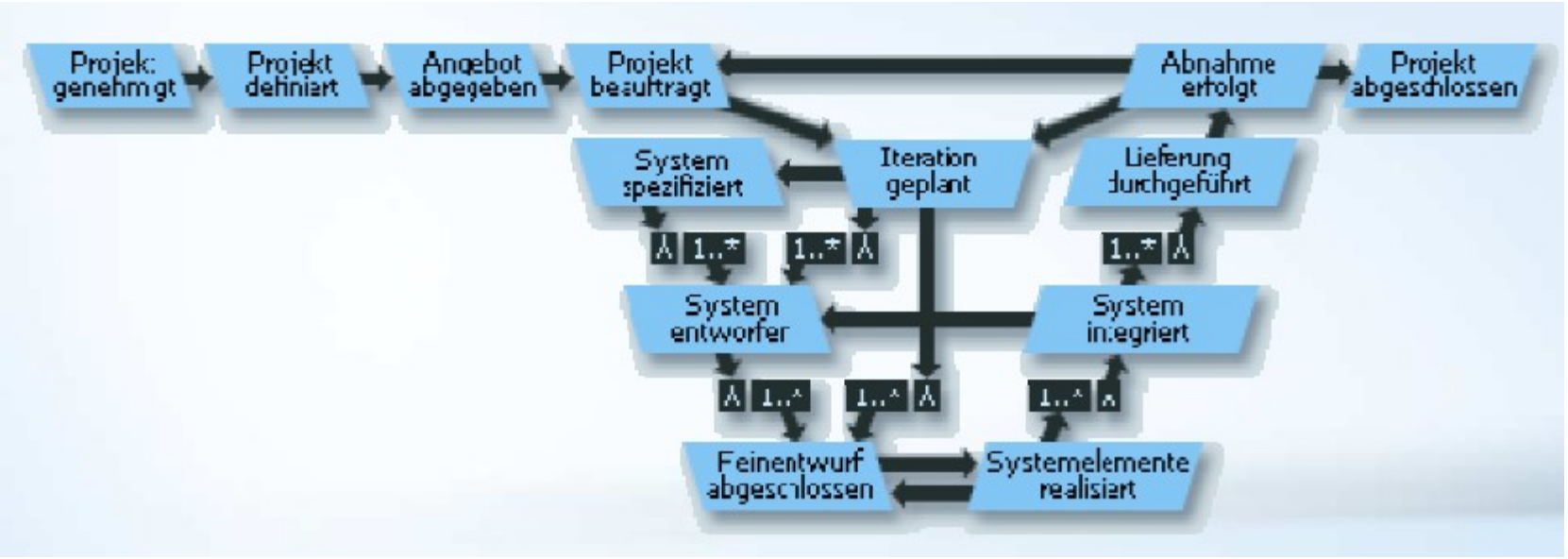
- Agile Systementwicklung (AN)



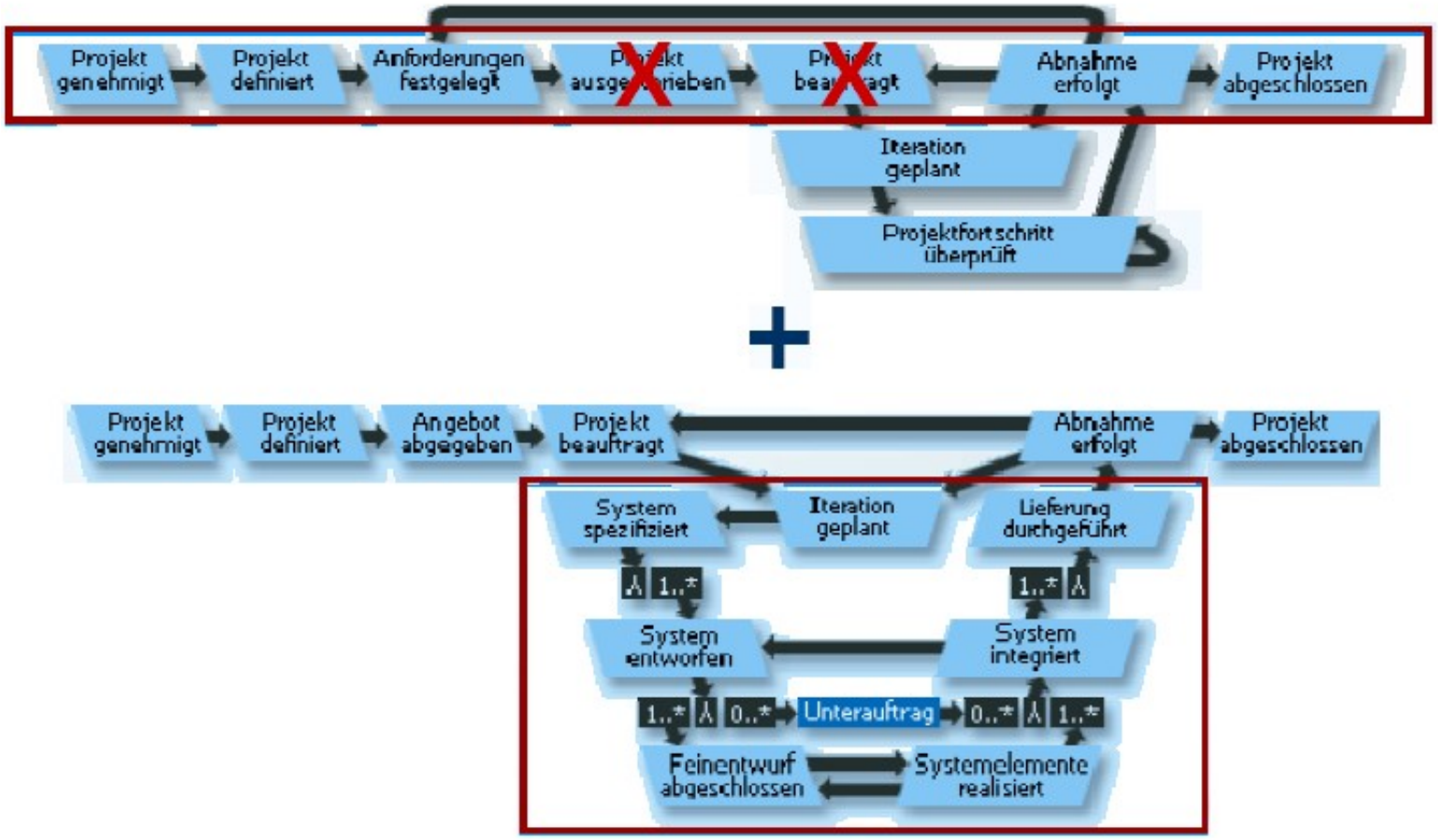
Projektdurchführungsstrategien

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Wartung und Pflege von Systemen (AN)



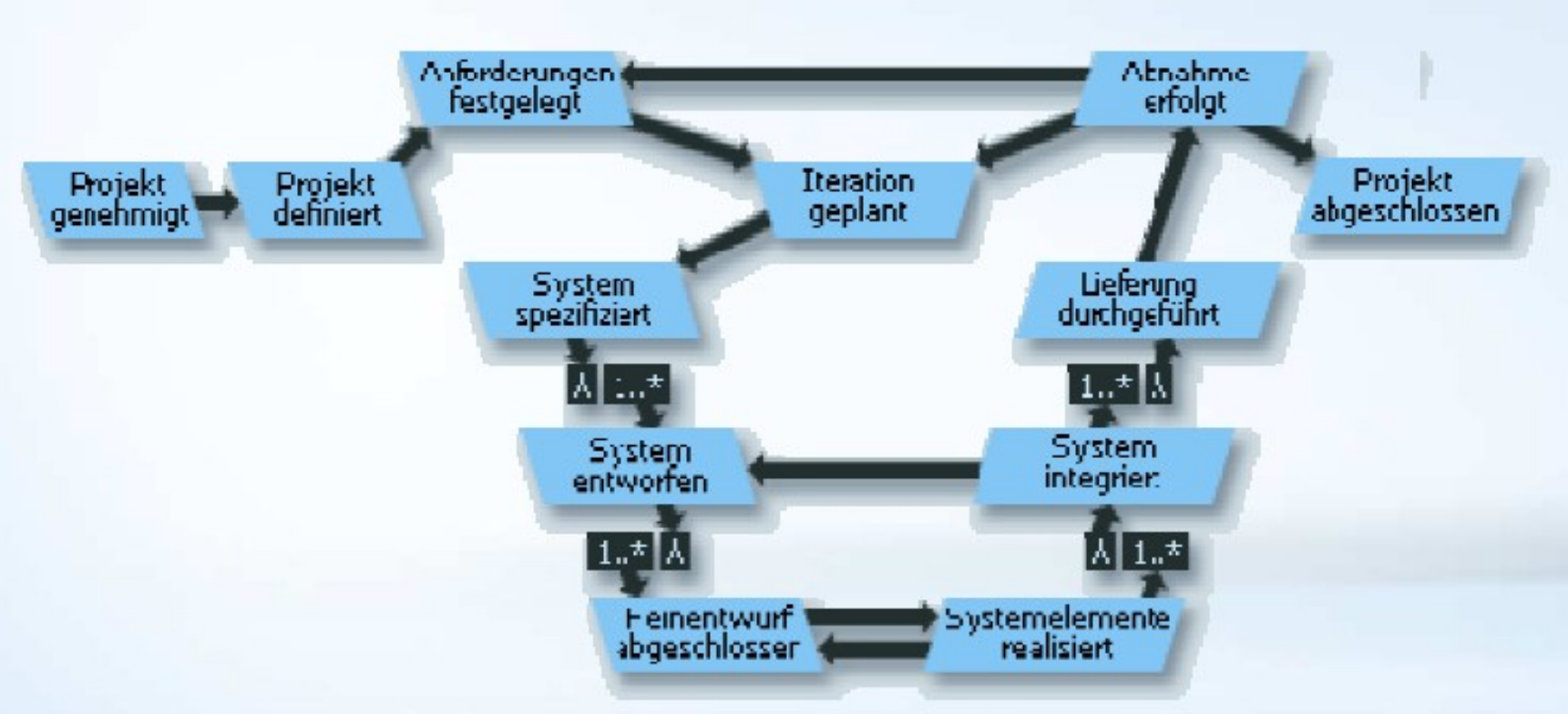
- Die Projektdurchführungsstrategien des Projekttyps „Systementwicklung (AG/AN)“ als Kombinationslösung



Systementwicklung AG/AN

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

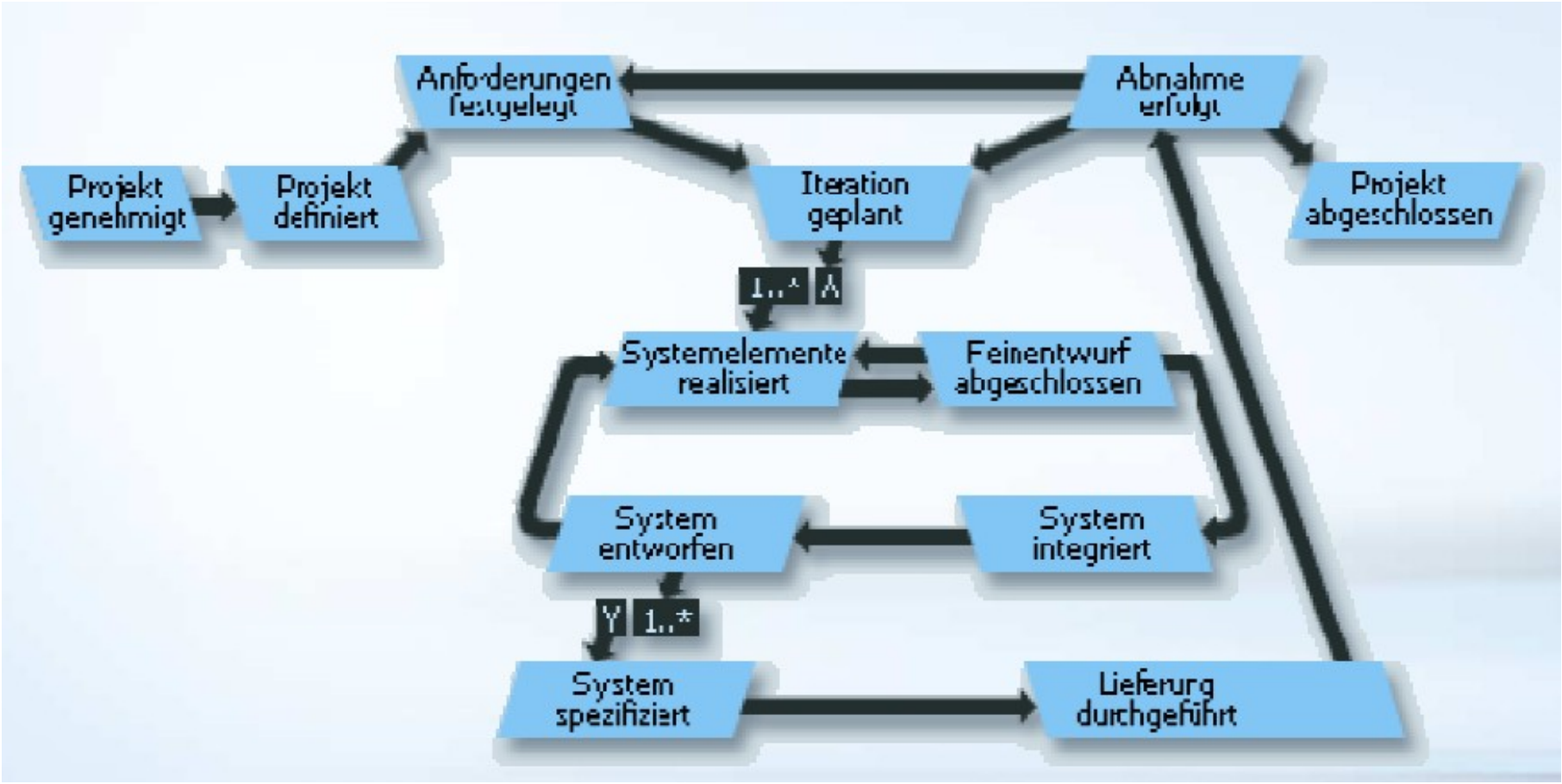
- Inkrementelle Systementwicklung (AG/AN)



Systementwicklung AG/AN

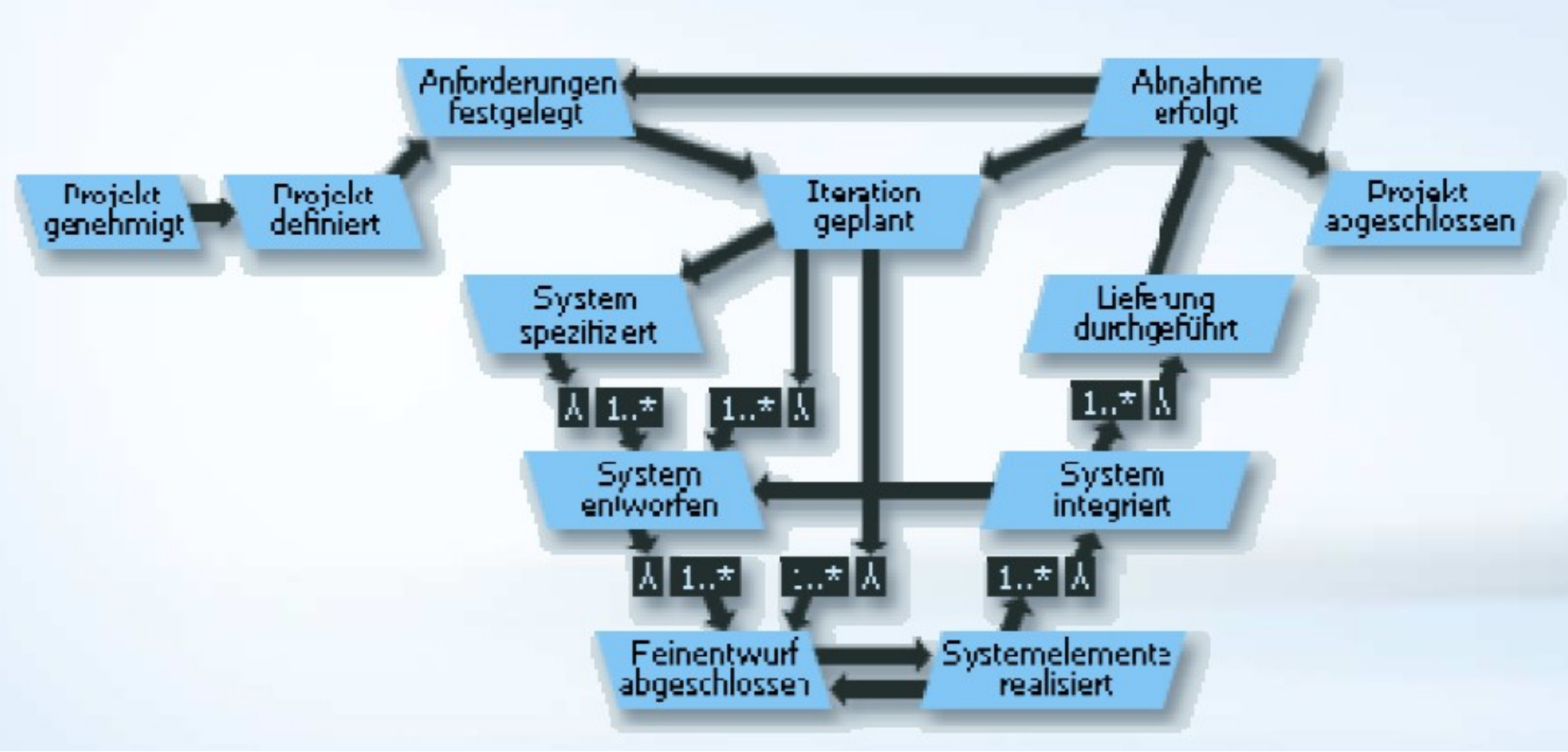
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Agile Systementwicklung (AG/AN)



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

- Wartung und Pflege von Systemen (AG/AN)



Lehrevaluation

- Zeitraum der Evaluation: 16.06.2006 und 31.07.2006
- Lehrveranstaltung: Software-Managment

- Veranstaltungskennung: kpf-swm-06
- Passwort: ee5q6xaj
- Link zum Fragebogen: https://www.umfragen.uni-bonn.de/leipzig/lehre2