# Anwendungssysteme in Industrie, Handel und Verwaltung

## Architekturen von Anwendungssystemen

## Sommersemester 2025


**Lehrstuhl für Wirtschaftsinformatik**
**Prozesse und Systeme**
*Universität Potsdam*

**Chair of Business Informatics**
**Processes and Systems**
*University of Potsdam*

Univ.-Prof. Dr.–Ing. habil. Norbert Gronau
*Lehrstuhlinhaber* | *Chairholder*

*Mail* August-Bebel-Str. 89 | 14482 Potsdam | Germany
*Visitors* Digitalvilla am Hedy-Lamarr-Platz, 14482 Potsdam
*Tel* +49 331 977 3322

*E-Mail* ngronau@lswi.de
*Web* lswi.de

## Lernziele

- Was wird unter Architekturen im Kontext der Wirtschaftsinformatik verstanden?
- Was versteht man unter Integration?
- Was sind typische Methoden zur Integration?
- Welche Integrationsarchitekturen gibt es und was macht sie aus?
- Aus welchen Komponenten besteht typischerweise ein ERP-System?
- Wie wird ein Geschäftsvorfall technisch abgebildet?
- Wie werden komplexe Geschäftsprozesse in ERP-Systemen abgebildet?
- Was bedeutet  Wandlungsfähigkeit im Kontext von ERP-Systemen und Systemarchitekturen?

# Quick Check 1
# Vorlesung 09: Fragerunde 1

Auditorium Quiz App

STUDENT

https://quiz.lswi.de/login

Veranstaltungs-
schlüssel:

AWS

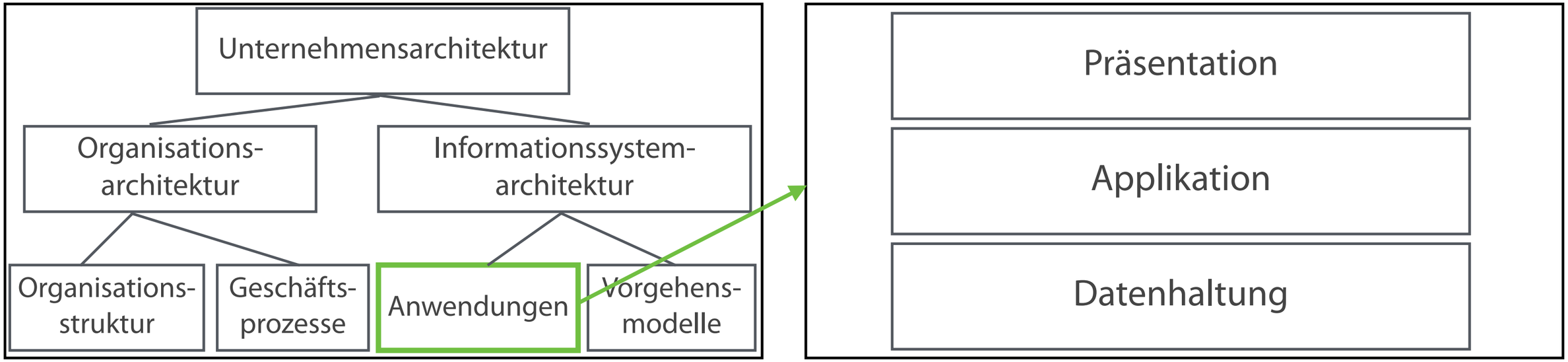

# Einführung in Architekturen

Integrationsansätze

Ausgewählte Systemarchitekturen

Wandlungsfähigkeit

# Arten von Architekturen




## Unternehmensarchitektur

- Betrachtung aller Elemente eines Unternehmens
- Unterteilung in Organisations- und IS-Architektur
- Anwendungen als Teil der IS-Architektur

## Softwarearchitektur

- Grundlegende Organisation eines Anwendungssystems
- Prinzipien, die den Entwurf und die Evolution des Systems bestimmen

Quelle: Reussner/Hasselbring 2005, S. 1 sowie Sinz 2004, S. 315

## Unternehmensarchitektur: Vom Ziel zur Technik

## Wie Strategie, Prozesse und IT ineinandergreifen

1. Unternehmensstrategie

   Ziel: Wettbewerbsvorteile sichern, Markt bedienen
   (Beispiel: "Wir wollen schneller liefern als die Konkurrenz.“)

2. Geschäftsprozesse / Business Architektur

   Umsetzung der Strategie durch Abläufe und Organisation
   *(z.B. Order-to-Cash-Prozess optimieren)*

3. Informations- und IT-Architektur

   Abbildung und Unterstützung der Prozesse durch IT-Systeme/ERP
   *(Welche Software braucht welchen Prozess?)*

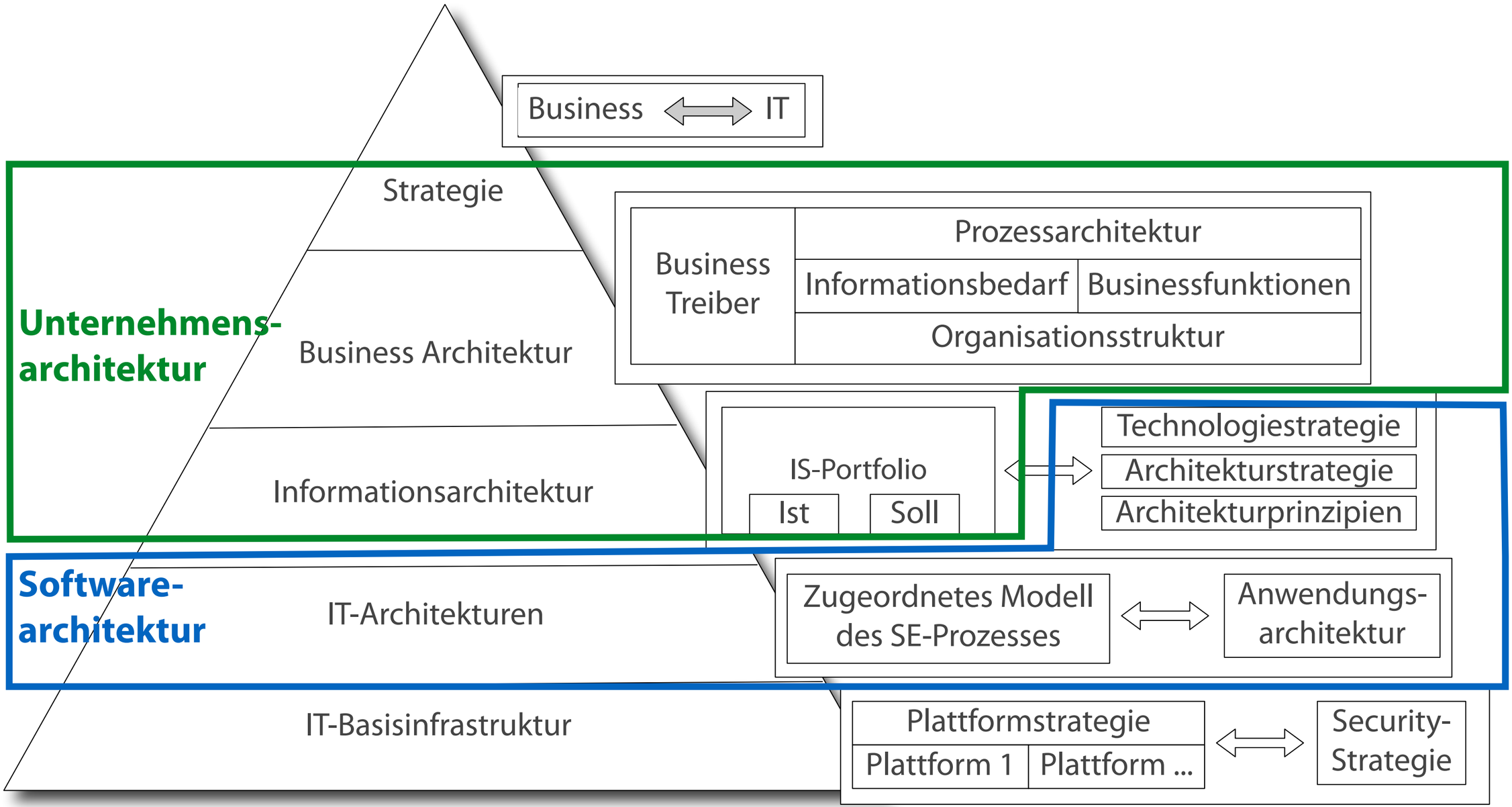
4. IT-Infrastruktur

   Technische Basis für alle Anwendungen
   *(Server, Cloud, Security)*

Quelle: vgl. Ahlemann et al. 2012, S. 17ff.

# Wie hängen Unternehmensstrategie, Geschäftsprozesse und IT/ERP-Systeme zusammen?

## Elemente der Architekturpyramide des Enterprise Architecture Management




Quelle: Dern 2006, S. 11-37

## Ziele einer Softwarearchitektur

- **Strukturierte Entwicklung:** Klare Trennung von Komponenten erleichtert Entwicklung und Wartung.
- **Langfristige Nutzbarkeit:** Das System bleibt über Jahre nutzbar und anpassbar.
- **Anpassungsfähigkeit:** Neue Anforderungen können einfach integriert werden.

## Typische Herausforderungen, die adressiert werden müssen

- Beherrschung der Komplexität
- Testbarkeit (Ist der Systemzustand validierbar?)
- Wartbarkeit/Updatefähigkeit
- Anpassbarkeit (Prozesse, Daten, Funktionen)
- Integrationsfähigkeit (Fremdanwendungen)
- Skalierbarkeit

**Die Softwarearchitektur eines Systems ist vergleichbar mit Bauplanung/-architektur von Gebäuden.**

Quelle: Vgl. Gronau 2021, S. 27ff

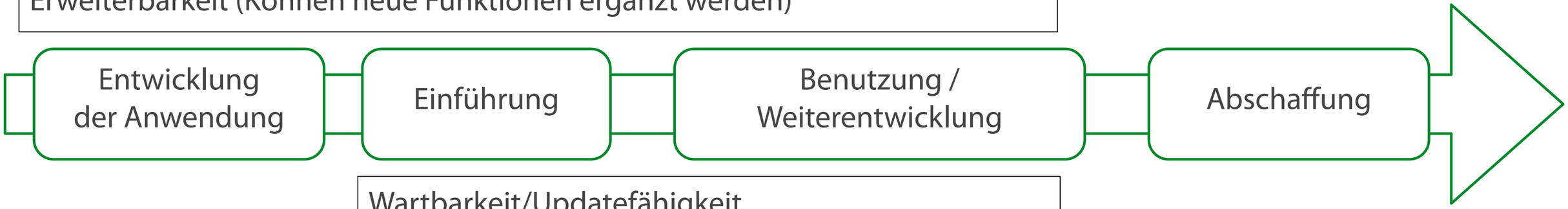
# Ziele einer Softwarearchitektur

## Warum braucht Software überhaupt eine Architektur?

Beherrschung der Komplexität

Testbarkeit (Ist der Systemzustand validierbar)

Erweiterbarkeit (Können neue Funktionen ergänzt werden)




Wartbarkeit/Updatefähigkeit

Anpassbarkeit (Prozesse, Daten, Funktionen)

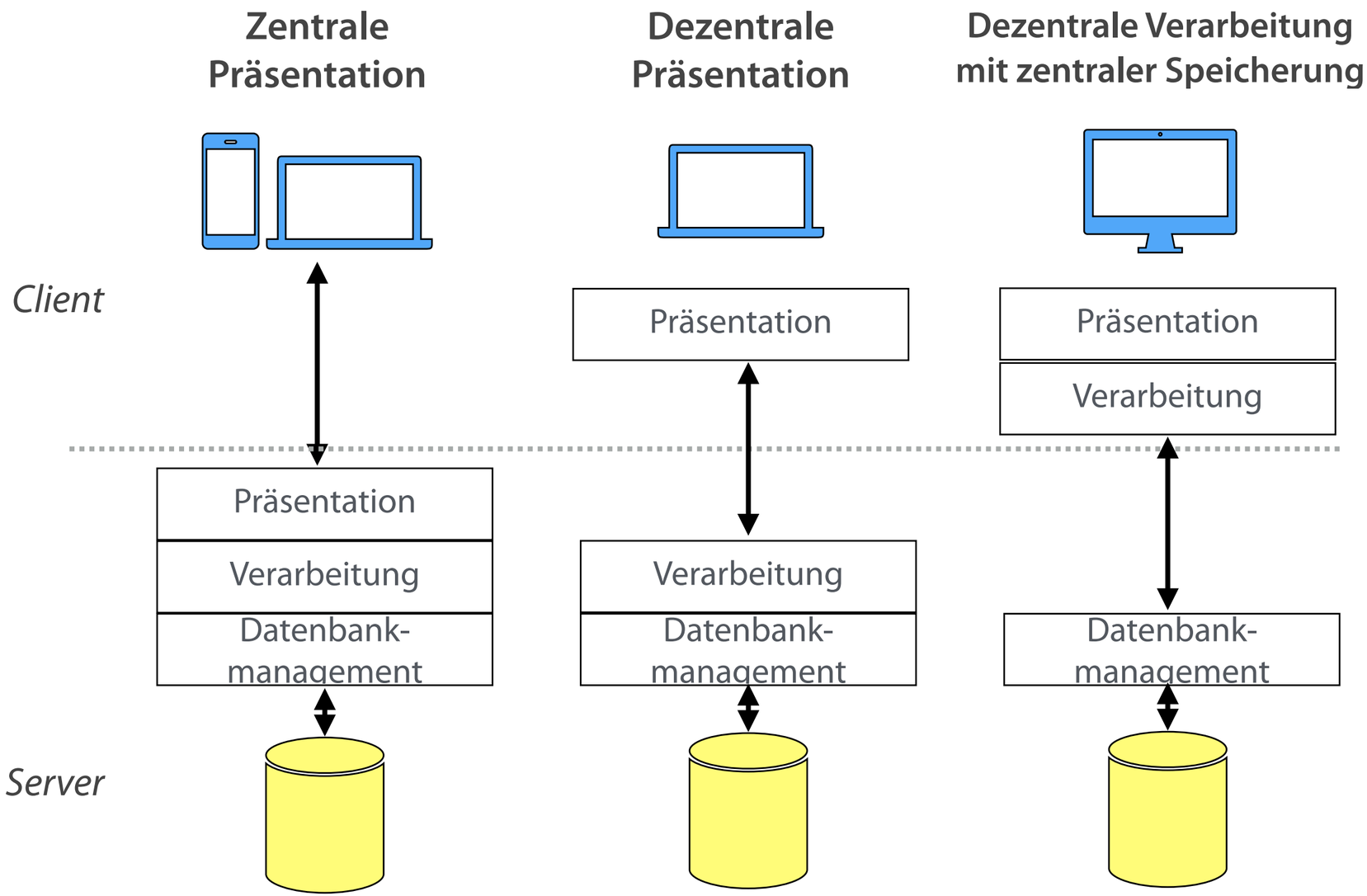
Integrationsfähigkeit (Fremdanwendungen)

Skalierbarkeit

**Die Ziele einer Softwarearchitektur werden hier aus Sicht der Standardsoftware betrachtet.**

# Verteilung der Systemfunktionen:

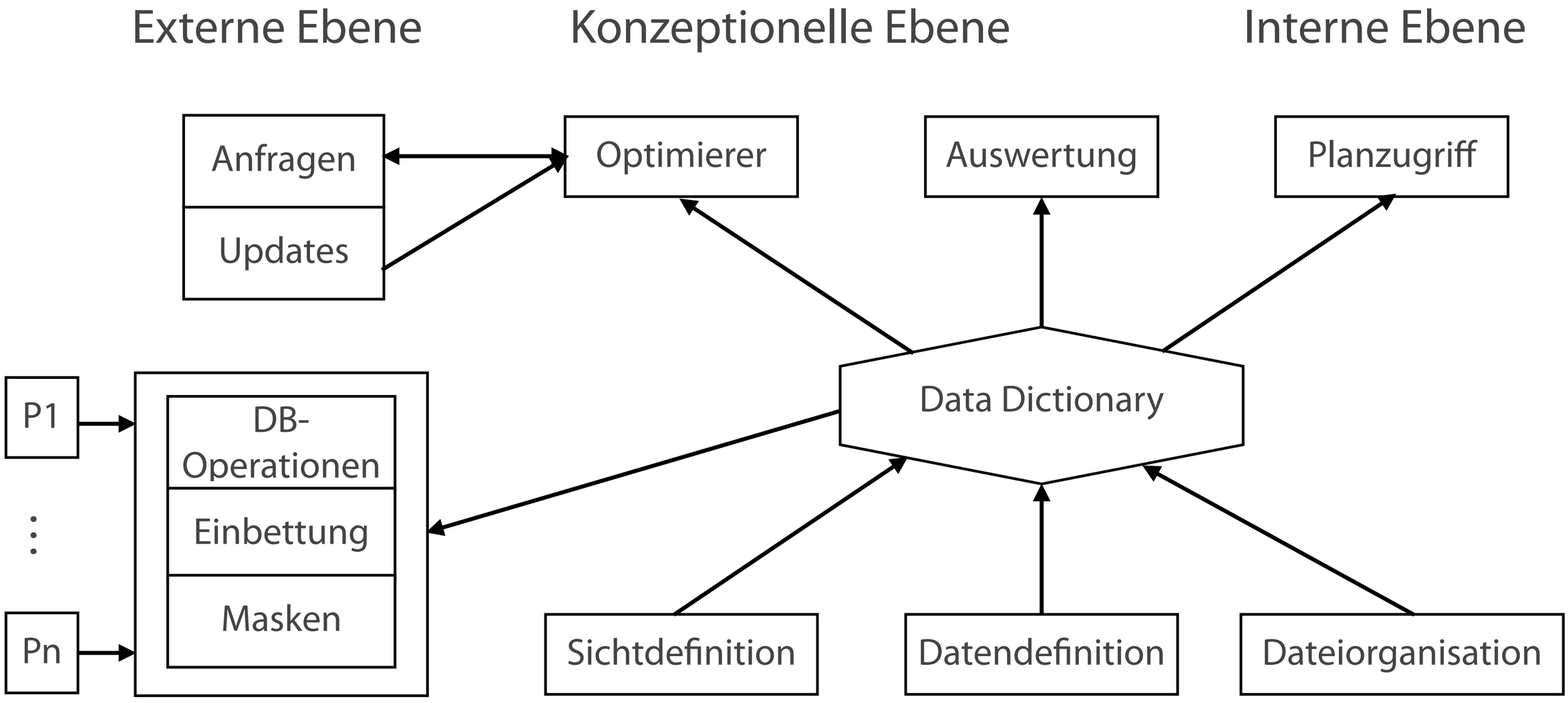
# Client-Server Computing




**Client-Server-Computing erlaubt es, die Systemfunktionen auf verschiedene Weise auf mehrere Computer zu verteilen.**

Quelle: Gronau 2021, S. 29

## Struktur und Komponenten eines Datenbank-Managementsystems (DBMS)




**Client-Server-Computing erlaubt es, die Systemfunktionen auf verschiedene Weise auf mehrere Computer zu verteilen**

Quelle: Heuer, Saake 2000, S. 31


# Einführung in Architekturen

**Integrationsansätze**

Ausgewählte Systemarchitekturen

Wandlungsfähigkeit

# Warum ist die Integration betrieblicher Anwendungssysteme entscheidend?

## Unternehmen nutzen eine Vielzahl spezialisierter Systeme

- ERP für zentrale Geschäftsprozesse
- CRM für Kundenbeziehungen
- SCM für Lieferketten
- MES für die Produktion
- BI für Auswertungen und strategische Steuerung

→ Diese Systeme müssen miteinander **vernetzt** sein, um **Datenbrüche zu vermeiden**.

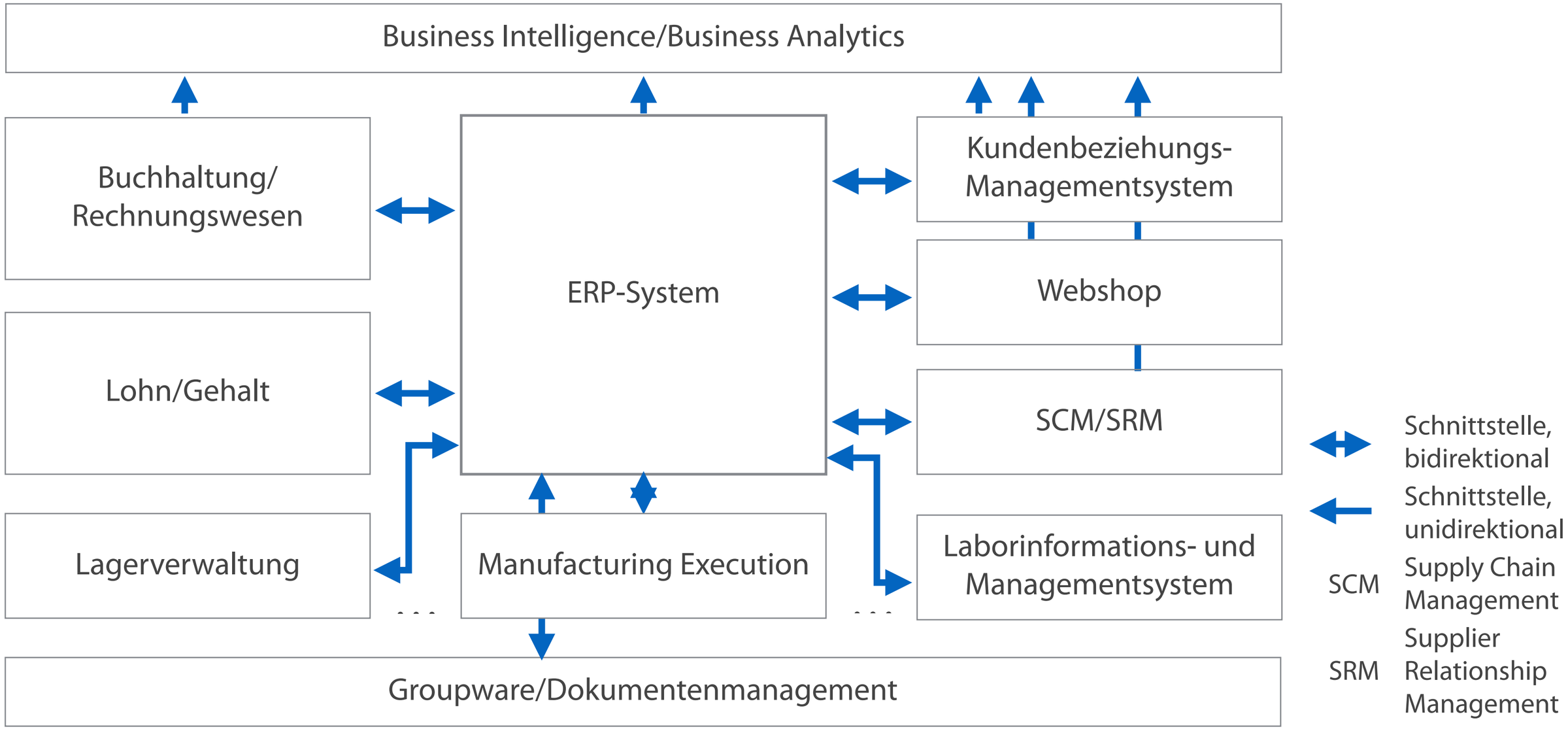
## Ziel der Integration

- Vermeidung redundanter Datenhaltung
- Automatisierung durchgängiger Prozesse
- Echtzeit-Datenfluss zur Entscheidungsunterstützung
- Höhere Effizienz, weniger Medienbrüche

**Im Allgemeinen versteht man unter Integration die Eingliederung, Einbeziehung von vielen Aspekten zu einem größeren Ganzen.**

Quelle: Vgl. Gronau 2021, S. 34

# Wie integriert ein ERP-System unterschiedliche Anwendungssysteme?




**Zwischen den Systemen findet ein äußerst intensiver Datenaustausch statt.**

Quelle: Gronau 2021, S. 10

# Integrationstypen und Beispielprozess

## Integrationstypen im Überblick

- **Bidirektionale Schnittstellen**: Gegenseitiger Datenaustausch (z. B. ERP ↔ CRM)
- **Unidirektionale Schnittstellen**: Nur eine Richtung (z. B. Webshop → ERP)
- Systeme wie **Dokumentenmanagement, BI** und **LIMS** (Laborinformationssysteme) ergänzen das ERP.

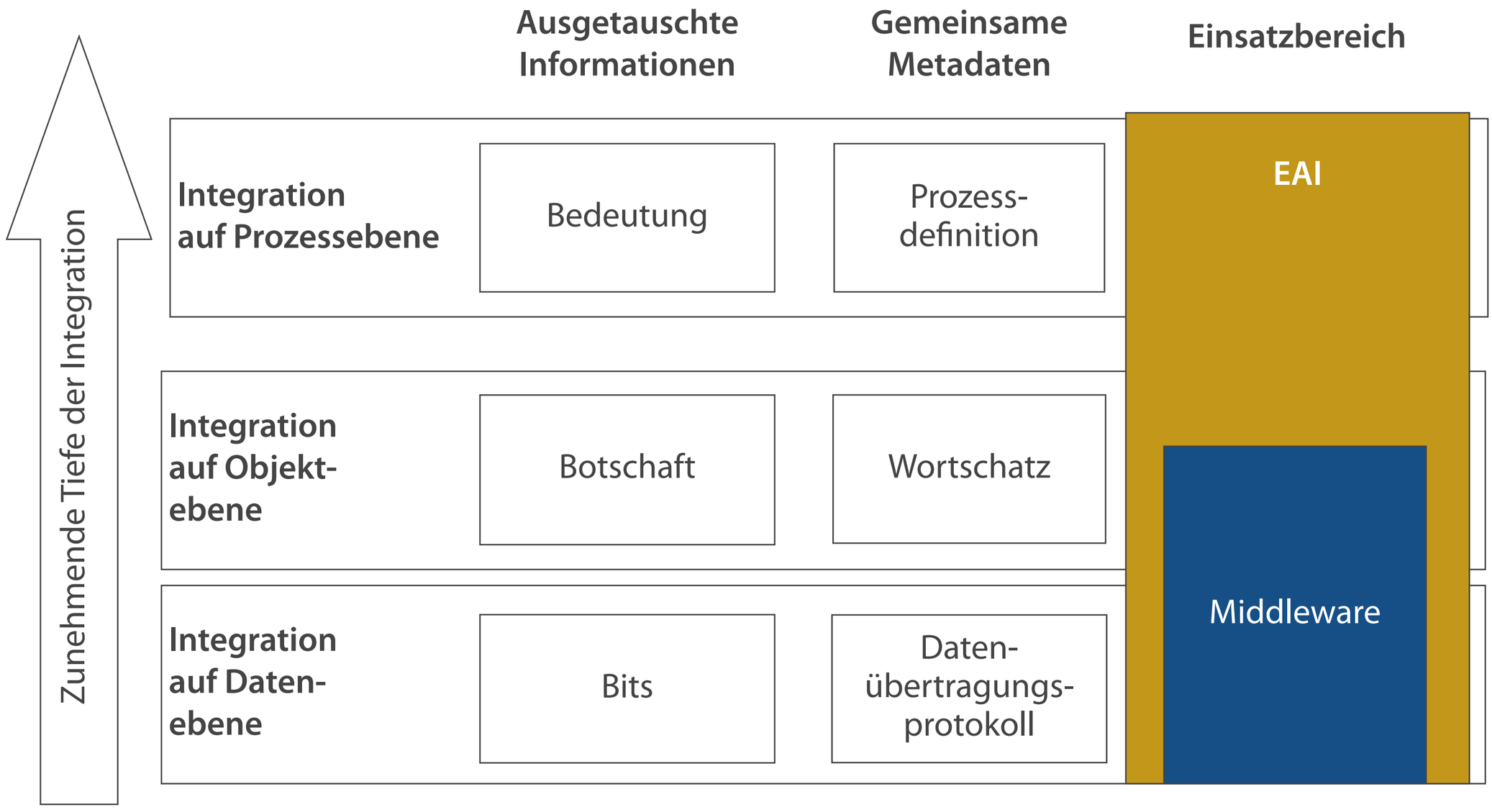
## Beispielprozess

- Kunde bestellt im Webshop → CRM & ERP erfassen die Bestellung → Lagerverwaltung prüft Verfügbarkeit → Versand wird über Manufacturing Execution organisiert → Rechnung wird automatisch im ERP erzeugt → Daten fließen ins BI-System

**Zwischen den Systemen findet ein äußerst intensiver und kritischer Datenaustausch statt – die Qualität der Integration bestimmt die Effizienz der Unternehmensprozesse.**

Quelle: Vgl. Gronau 2021, S. 34 - 39

# Integrationsansätze zwischen Anwendungssystemen




**Integration zwischen Anwendungssystemen kann auf Daten-, Objekt- und Prozessebene stattfinden.**

Quelle: Gronau 2021, S. 36

# Umsetzung der Datenintegration auf verschiedenen Integrationsebenen

| Ebene | Ausgetauschte Information | Typische Herausforderung | Technische Umsetzung |
|---|---|---|---|
| **Prozessebene** | Bedeutung, Abläufe | Hoher Abstimmungsaufwand, aber maximale Konsistenz | Workflow-Engines, BPMN-Tools (z.B. Camunda, ARIS) |
| **Objektebene** | Nachrichten (z. B. „Auftrag“) | Semantisches Mapping notwendig | REST-APIs, XML, JSON, Microservices |
| **Datenebene** | Rohdaten, Bits | Technisch einfach, aber oft ohne Kontext | Datenbanken, Dateisysteme, Streams (z.B. Kafka, SQL) |

**Je höher die Integrationsebene, desto mehr Semantik und Abstimmung ist notwendig – aber auch desto enger die Kopplung und größer der Nutzen durch automatisierte Abläufe.**

Quelle: Vgl. Gronau 2021, S. 36

## Middleware vs. EAI

## Systemintegration: Konzept vs. technische Umsetzung

- Middleware ist eine Softwarekomponente, die die technische Verbindung zwischen Systemen ermöglicht.
- EAI ist ein konzeptioneller Ansatz zur Integration von Anwendungen, Daten und Prozessen.
- Middleware wird häufig im Rahmen von EAI-Lösungen eingesetzt.

**Middleware:**

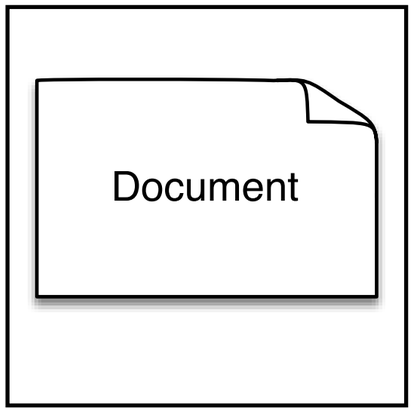
- **Ebene:** Daten- und Objektebene
- **Funktion:** Technischer Nachrichtenaustausch
- **Technologien:** REST-APIs, XML, Messaging, Datenbanken
- **Beispiele:** FTP-Übertragung, API-Kommunikation

**EAI (Enterprise Application Integration):**

- **Ebene:** Alle Ebenen, v. a. Prozessebene
- **Funktion:** End-to-End-Integration, Prozessverarbeitung
- **Technologien:** Workflow-Engines, ESB, BPMN
- **Beispiele:** Geschäftsprozessintegration (z. B. SAP PI/PO)

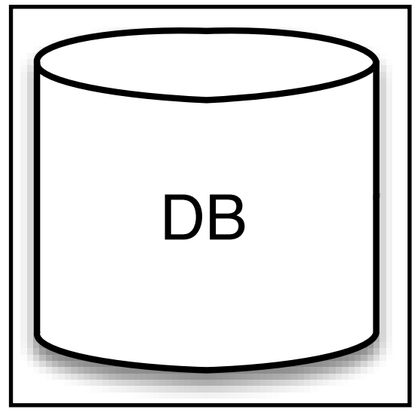
Quelle: Vgl. Hohpe/Woolf 2003, S. 13 ff. und Linthicum 2000, S. 8 ff.

# Integrationsmethoden auf Datenebene



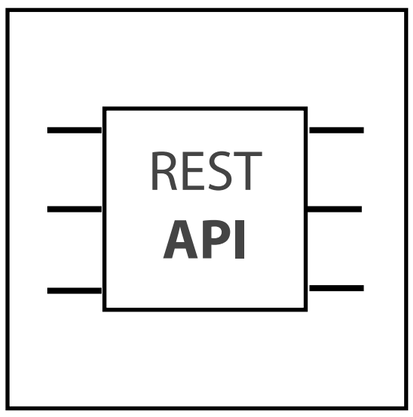

## Dateitransfer

- Klassischer Weg des Datenaustauschs zwischen Systemen – z. B. über CSV. Oft bei älteren Systemen.



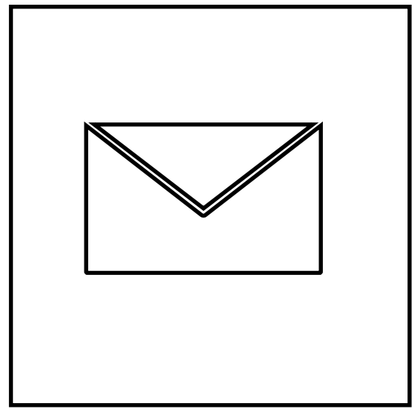

## Gemeinsame Datenbank

- Systeme greifen auf dieselbe Datenbasis zu – einfach, aber riskant bei gleichzeitigen Schreibzugriffen.




## Verteilte Aufrufe

- Remote Procedure Calls (RPC), Web Services oder REST-APIs – direkte Funktionalitäten werden systemübergreifend aufgerufen.



## Nachrichten

- Asynchrone, strukturierte Datenpakete, die zwischen Systemen ausgetauscht werden.
- Lose gekoppelte Kommunikation über Message Queues oder Event-basierte Systeme – z. B. mit Kafka

**Welche Integrationsmethode passt zu welchem Anwendungsszenario?**

**Wie unterscheiden sich die Methoden hinsichtlich Wartbarkeit, Echtzeitfähigkeit, Kopplung?**

# Austauschformate: Spezialisierungsgrade von Datenformat

## XML - generisches Datenformat

- „Extensible Markup Language“
- Tags ermöglichen flexible Datenstrukturierung
- Eigene Tags, Strukturen und komplexe Typen definierbar
- Geringe Einschränkungen, vielseitig einsetzbar (Basistechnologie)
- Vergleichbare Formate: JSON, YAML

## EDIFact - standardisiertes Datenformat

- „Electronic Data Interchange for Administration, Commerce and Transport“
- Internationaler, branchenübergreifender Nachrichtenstandard (UN)
- Standardisierte Nachrichtentypen durch definierte Kürzel in Metadaten (z. B. ORDERS)
- Format zur automatisierten B2B-Kommunikation
- Umwandlung in XML möglich

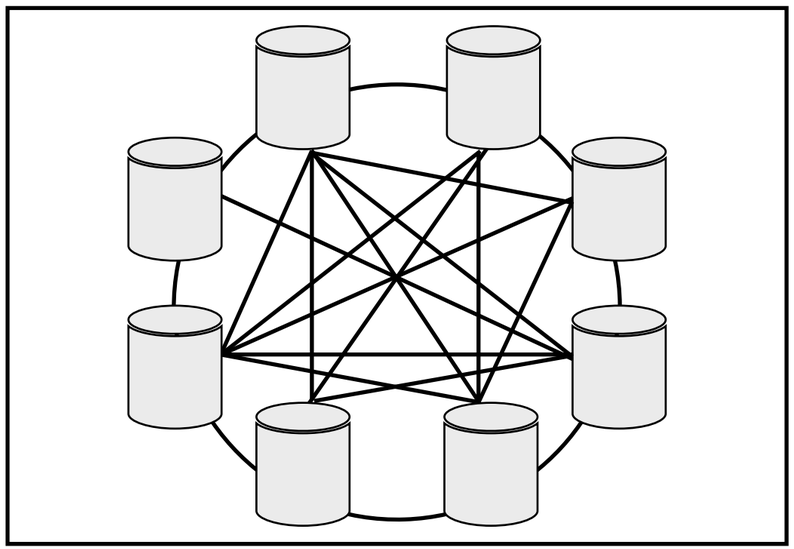
## IDoc - unternehmenspezifisches Datenformat (SAP)

- „Intermediate Document“ (SAP-intern)
- Format für den strukturierten Datenaustausch zwischen SAP-Systemen
- Basierend auf EDIFact-Grundsätzen, aber SAP-spezifisch erweitert
- Definition eigener Nachrichtentypen
- Umwandlung in EDIFact möglich (z. B. für B2B-Partner)

Quelle: Vgl. Gronau 2021, S. 39; Linthicum 2000, Kap. 14 & 17

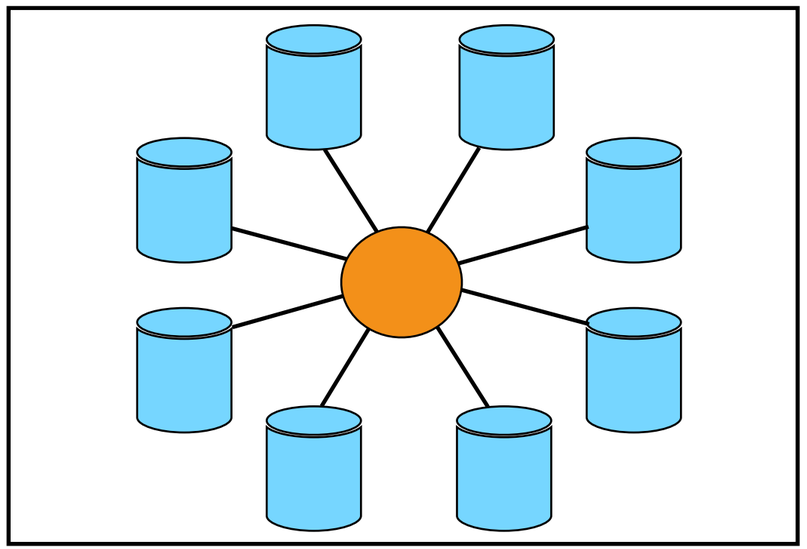
# Prinzipien von Integrationsarchitekturen

## Wie Systeme miteinander verbunden werden: Drei Integrationsarchitekturen im Vergleich



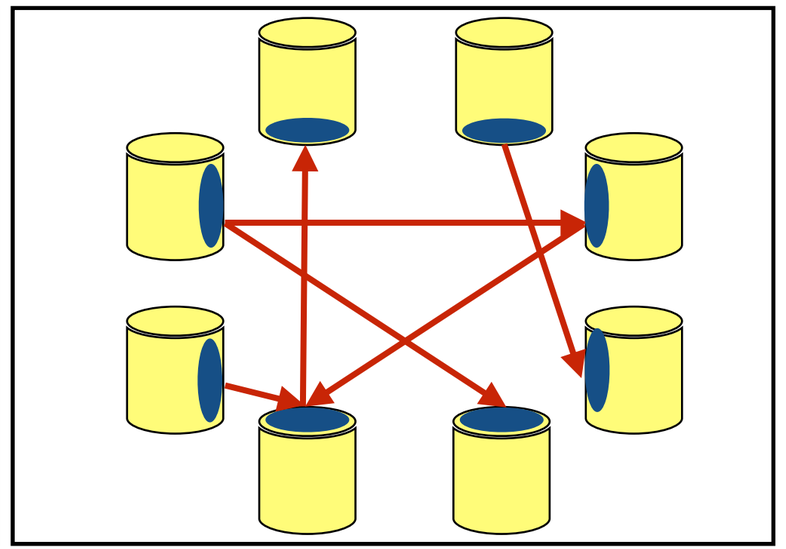
### Punkt zu Punkt

- Direkte Verbindung zwischen jedem Systempaar
- Individuell entwickelte Schnittstellen
- Feste Kopplung, schwer wartbar
- Skalierungsproblem bei vielen Systemen: Anzahl der Verbindungen = $n(n-1)/2$



### Hub and Spoke

- Zentrale Integrationsplattform (Hub) als Vermittler
- Systeme (Spokes) kommunizieren nur mit dem Hub
- Weniger Schnittstellen, aber:
- Single Point of Failure
- Transformation & Routing im Hub nötig

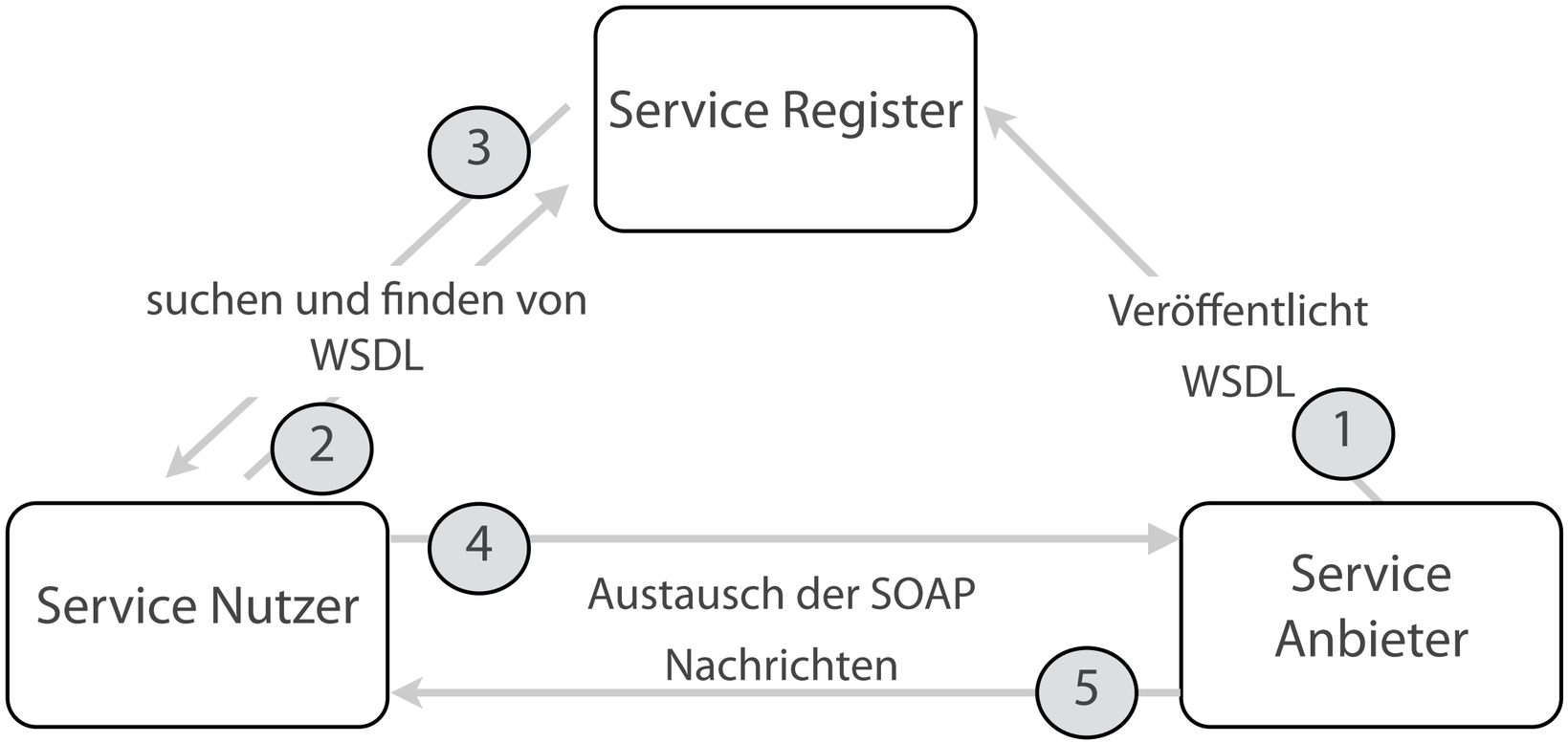


### Service-orientierte Architektur (SOA)

- Lose Kopplung durch standardisierte Dienste (z. B. Web Services, REST APIs)
- Wiederverwendbare Services für verschiedene Prozesse
- Dezentrale Kommunikation zwischen Services
- Höherer Abstimmungsbedarf, aber große Flexibilität

Quelle: Gronau 2021, S. 35

# Service orientierte Architekturen




| | |
|---|---|
| WSDL | Servicebeschreibung |
| SOAP | Stellt das Austauschformat zur Verfügung |
| UDDI | Standardisiertes Service Register Format |

Quelle: Erl 2005, S. 75

## Warum viele Unternehmen lange an Punkt-zu-Punkt festhalten – und was den Wechsel zu SOA (oder Microservices) erschwert

### Historisch gewachsen

- Systeme wurden nacheinander eingeführt, Integration geschah ad-hoc.
- Es gab keine strategische Integrationsplanung, sondern „quick fixes".

### Verfügbare Ressourcen

- Interne IT-Teams sind oft nicht auf serviceorientierte Architekturen spezialisiert.
- SOA erfordert Methodenkompetenz, Tools und Governance.

### Geringe Anfangskomplexität

- Für wenige Systeme ist Punkt-zu-Punkt schnell umsetzbar.
- Bei 2–3 Systemen erscheint der Aufwand für SOA/EAI oft nicht gerechtfertigt.

### Kosten- und Zeitdruck

- Punkt-zu-Punkt ist oft günstiger in der Erstumsetzung, vor allem in KMUs.
- Der Aufwand für Service-Design, Dokumentation und Standardisierung schreckt ab.

**Punkt-zu-Punkt wirkt kurzfristig einfach, wird aber langfristig zur technischen Schuld. Der Weg zu SOA oder Microservices erfordert strategisches Denken, Ressourcen und Kulturwandel – genau das schreckt viele Unternehmen ab.**

# Quick Check 2
# Vorlesung 09: Fragerunde 2

**Auditorium Quiz App**

STUDENT

https://quiz.lswi.de/login

Veranstaltungs-
schlüssel:

AWS


Einführung in Architekturen

Integrationsansätze

**Ausgewählte Systemarchitekturen**

Wandlungsfähigkeit

## Komponenten eines ERP-Systems

**ERP-Systeme bestehen meist aus mehreren Modulen, die zentrale Geschäftsbereiche eines Unternehmens abbilden:**

- Finanzwesen / Rechnungswesen (FI)
- Controlling (CO)
- Einkauf / Materialwirtschaft (MM)
- Lager / Logistik (WM)
- Personalwesen (HR)
- Produktion (PP)
- Business Intelligence (BI)

**Architektonischer Aufbau:**

- **Benutzungsschicht (**Benutzeroberfläche, Web-Client)
- **Applikationsschicht (**Applikationskern, Programmierumgebung, Datenbankunabhängiger & -abhängiger Teil, User Exits)
- **Datenhaltungsschicht (**DBMS, Datenbanken)
- Zusatz: Schnittstellen, Adaption

Quelle: vgl. Gronau 2021

# Technische Abbildung eines Geschäftsvorfalls im ERP-System

## Beispiel: Bestellung von Waren

1. Anlage eines Bestellbelegs im Einkauf (Materialwirtschaft)
   - Modul: Einkauf/Materialwirtschaft (MM)
   - Datenbank: Neue Zeile in der Bestelltabelle (z. B. SAP-Tabelle EBELN)

   → Erzeugung von Bewegungsdaten in der Bestelltabelle
2. Automatisierte Folgeprozesse
   - Reservierung/Änderung von Lagerbeständen
   - Integration ins Rechnungswesen (Buchung)
   - Workflow-Trigger (z. B. Freigabe, Wareneingang)
3. Status und Protokollierung
   - Jeder Schritt wird im System dokumentiert
   - Transparenz & Rückverfolgbarkeit

Quelle: Klaus, H. et al (2000) 141-162

## Abbildung komplexer Geschäftsprozesse im ERP-System

### Beispiel: Order-to-Cash-Prozess

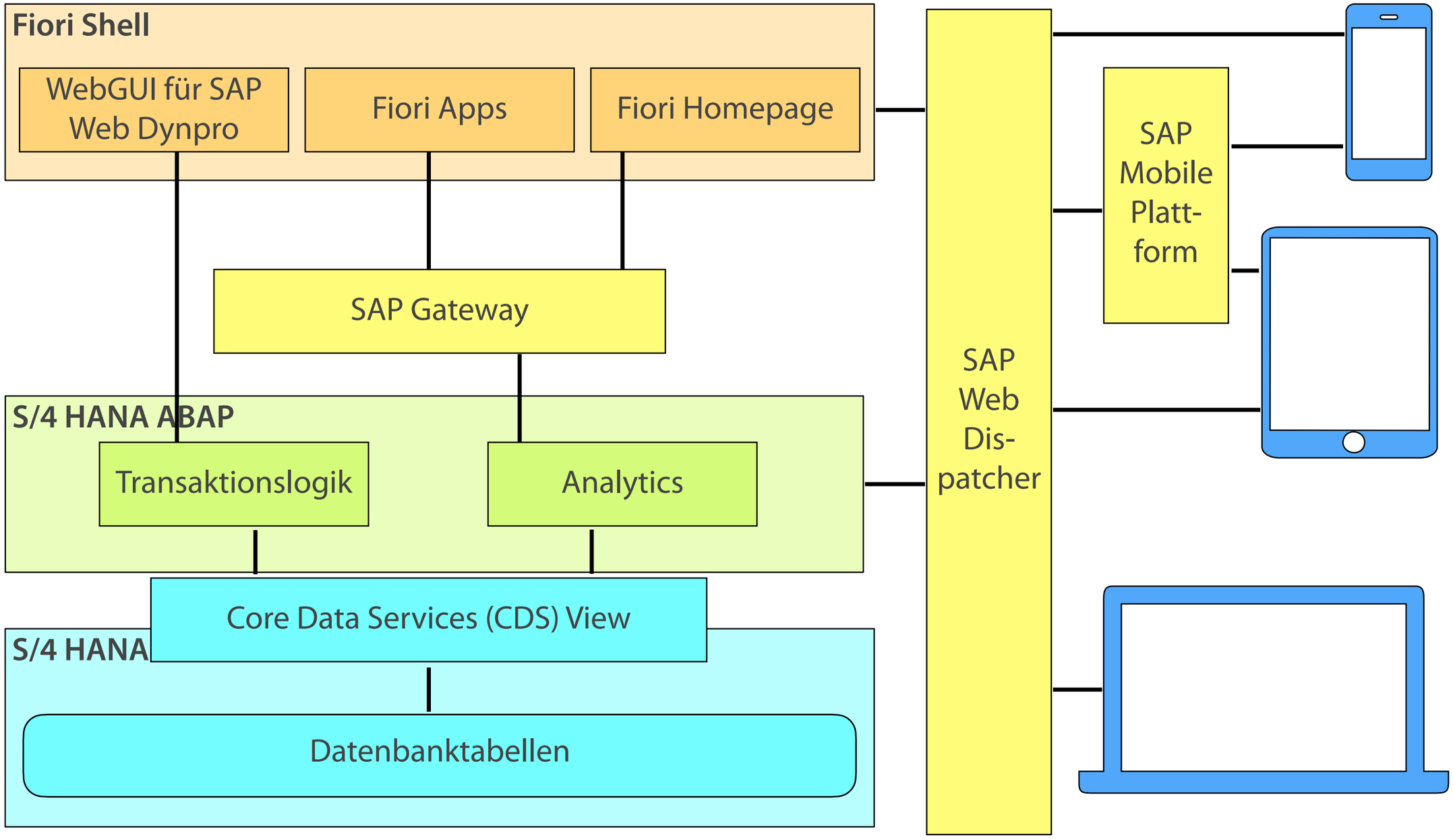
- Kundenauftrag anlegen -> Modul: Vertrieb (SD)
- Produktion/Lager prüft Verfügbarkeit -> Modul: Produktion (PP) / Lager (WM)
- Lieferung und Versand -> Modul: Logistik (LE)
- Rechnungsstellung -> Modul: Finanzwesen (FI)
- Zahlungseingang überwachen -> Modul: Finanzwesen (FI)

### ERP-Technik:

- Prozesslogik als Workflow hinterlegt (z. B. SAP Business Workflow)
- Module greifen automatisiert ineinander, Status ist systemweit sichtbar

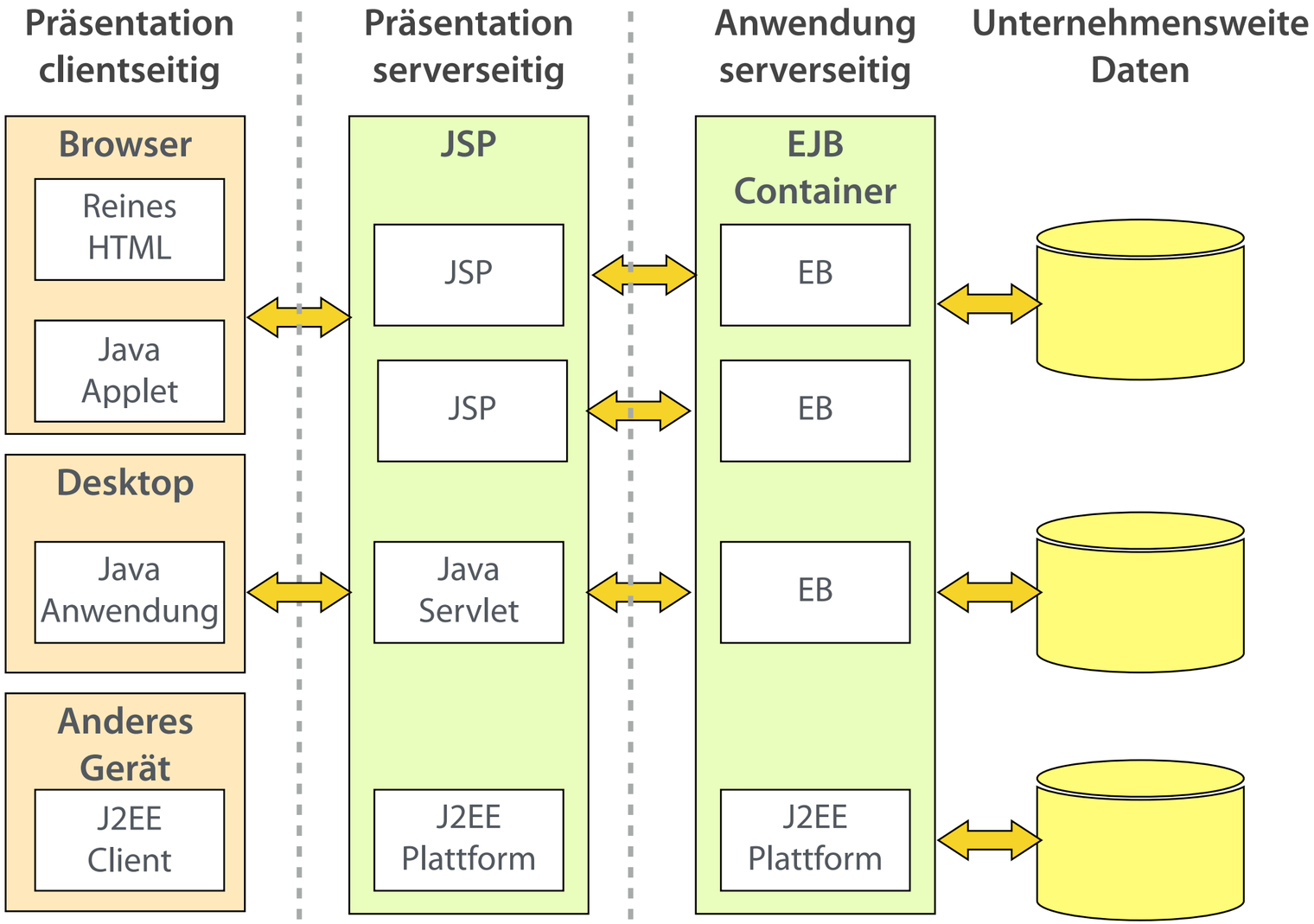
Quelle: Monk, E. et al , 2012

# Beispiel für Systemarchitektur: S/4 SAP Hana Systemarchitektur




Quelle: Gronau 2021, S. 44

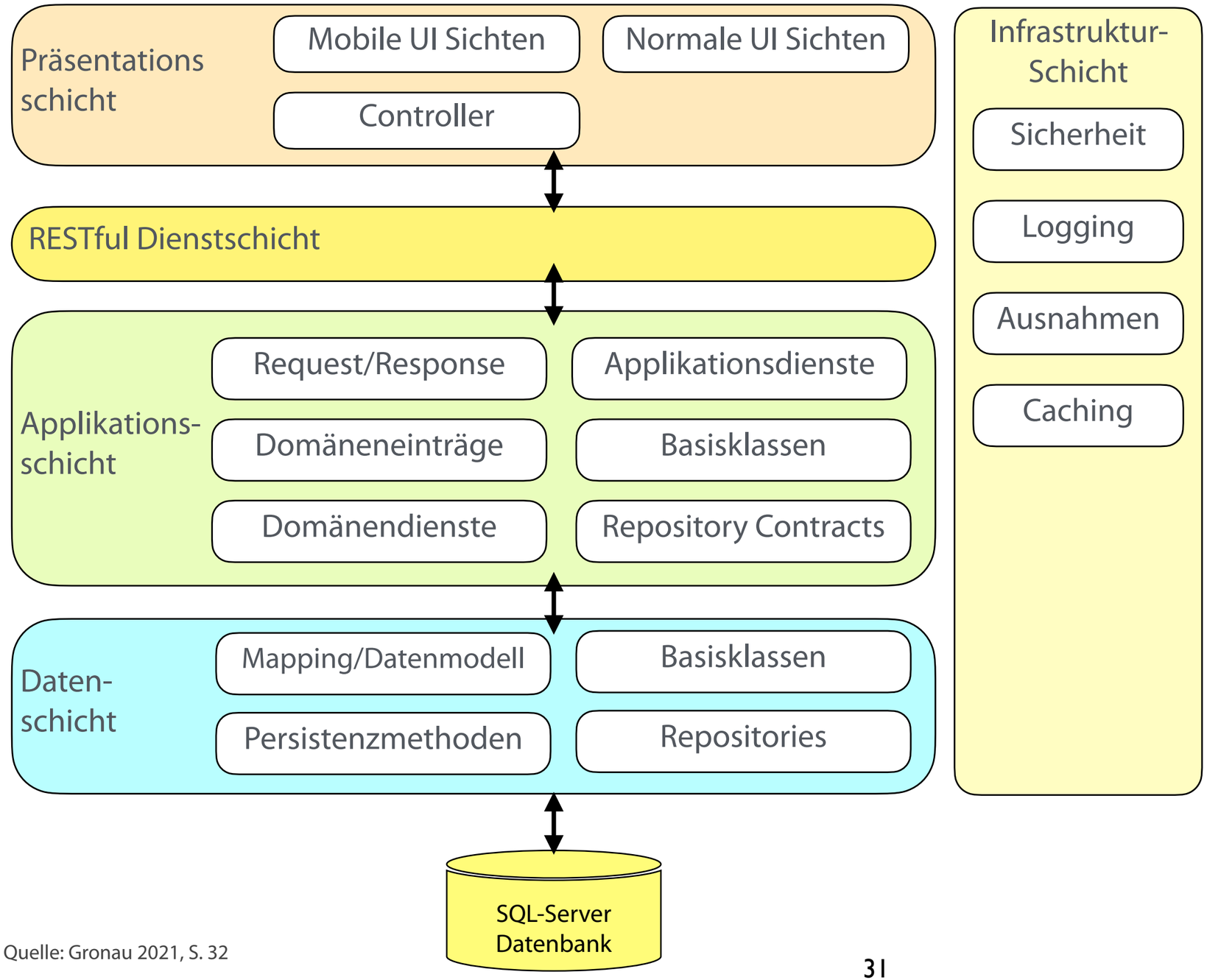
# Beispiel für Java Enterprise-Architektur basierte Systeme




## Ziele sind: Interoperabilität, Wiederverwendbarkeit und Erweiterbarkeit

Quelle: Gronau 2021, S. 31

# Beispiel für .Net basierte (ERP-)Systeme




Quelle: Gronau 2021, S. 32


Einführung in Architekturen

Integrationsansätze

Ausgewählte Systemarchitekturen

**Wandlungsfähigkeit**

# Wandlungsfähigkeit

## Warum?

- Notwendigkeit von kurzfristigen Anpassungen auf die aktuelle Marktsituation
- Schnelle und effiziente Anpassung sichert Erfolg

## Was?

- Veränderungen antizipieren und Impulse setzen
- Integration von Geschäftsprozessen, Architektur und Applikationen

## Wie?

- Technisch: Anwendungssysteme
- Geschäftsspezifisch: Unternehmensarchitektur

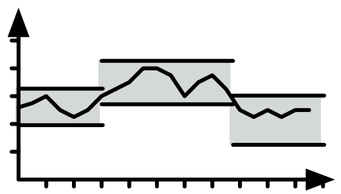
## Definition:

Wandlungsfähigkeit ist die Eigenschaft eines Systems, schnell, selbständig und effizient mit Veränderungen in seiner Umwelt umgehen zu können (Gronau/Weber 2009).

**Wandlungsfähigkeit ist ein wesentlicher Wettbewerbsvorteil.**

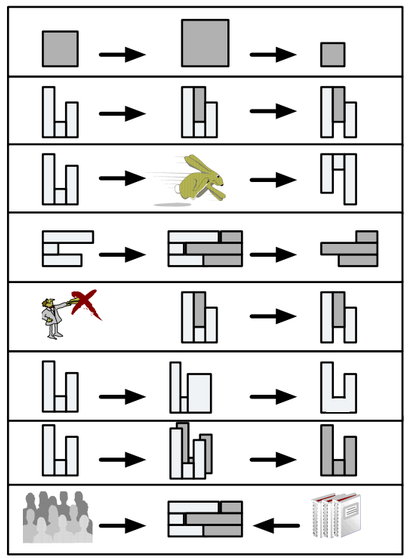
Quelle: Gronau 2021, S. 49 ff.

# Erklärungsmodelle zur Wandlungsfähigkeit



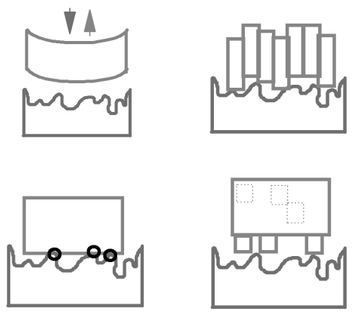
## Kapazitätsorientierter Ansatz

- Nachweis über Kapazitätsverlauf
- Fokus auf Historie-Daten zu Ressourcen und Ereignissen
- Operationalisierung über Kennzahlen



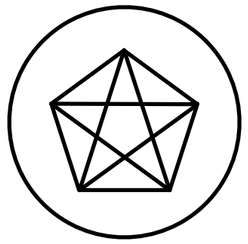
## Indikatorbasierter Ansatz

- Nachweis über Systemeigenschaften
- Fokus auf Wandlungsbefähiger
- Operationalisierung über Indikatoren



## Handlungsmusterbasierter Ansatz

- Nachweis über Handlungsspielraum
- Fokus auf Verfügbarkeit und Anwendbarkeit von Handlungsmustern
- Operationalisierung über Handlungsmuster



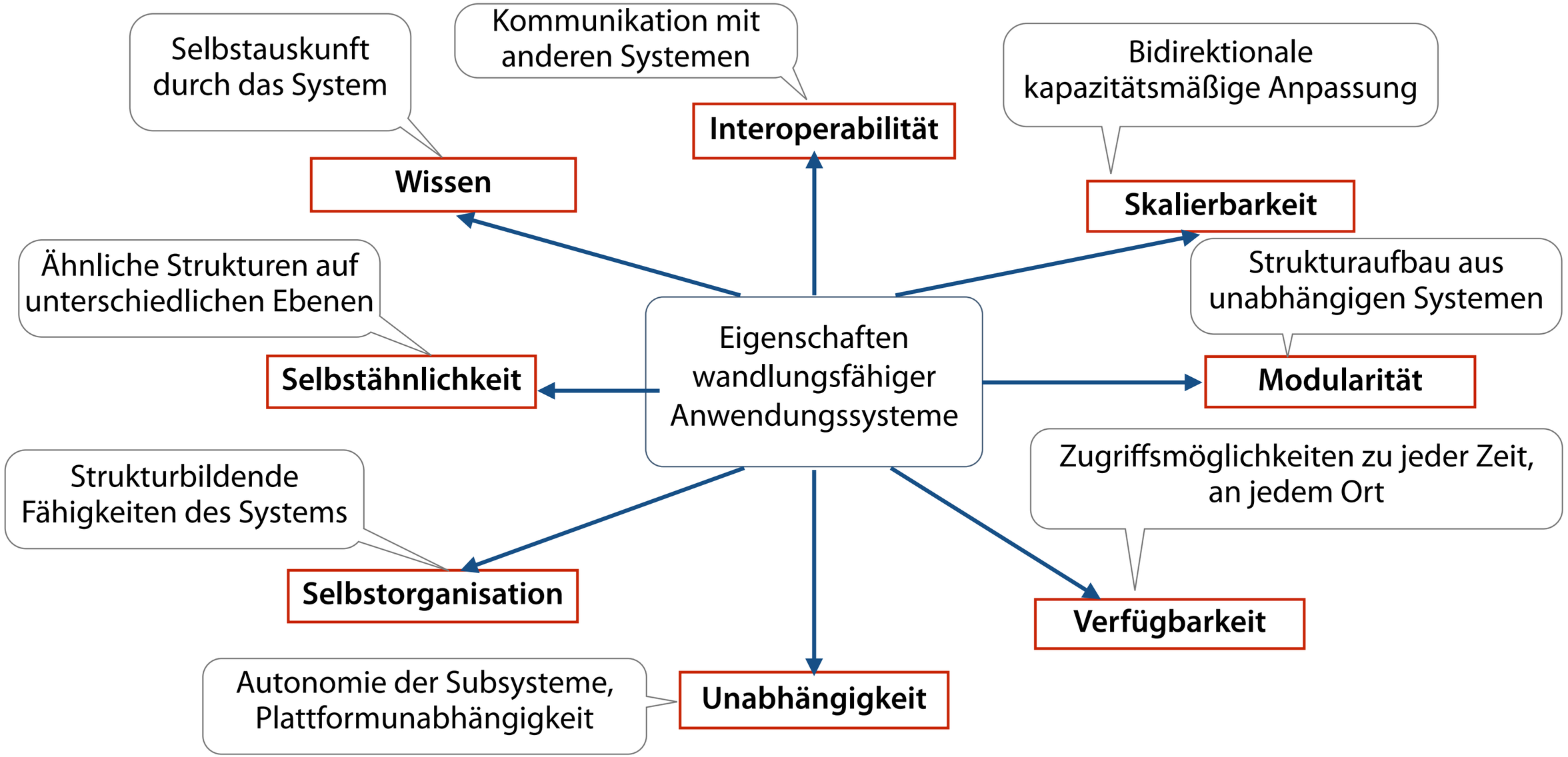
## Balanceorientierter Ansatz

- Nachweis über Ausgewogenheit des Handelns
- Fokus auf korrespondierende Handlungsmodalitäten
- Satz von fünf Handlungsmodalitäten (Können, Wollen, Müssen, Dürfen, Tun)

**Die Erklärungsmodelle stellen alternative Ansätze zur Operationalisierung von WF dar.**

**Es werden jeweils unterschiedliche Perspektiven bzw. Schwerpunkte ausgewählt.**

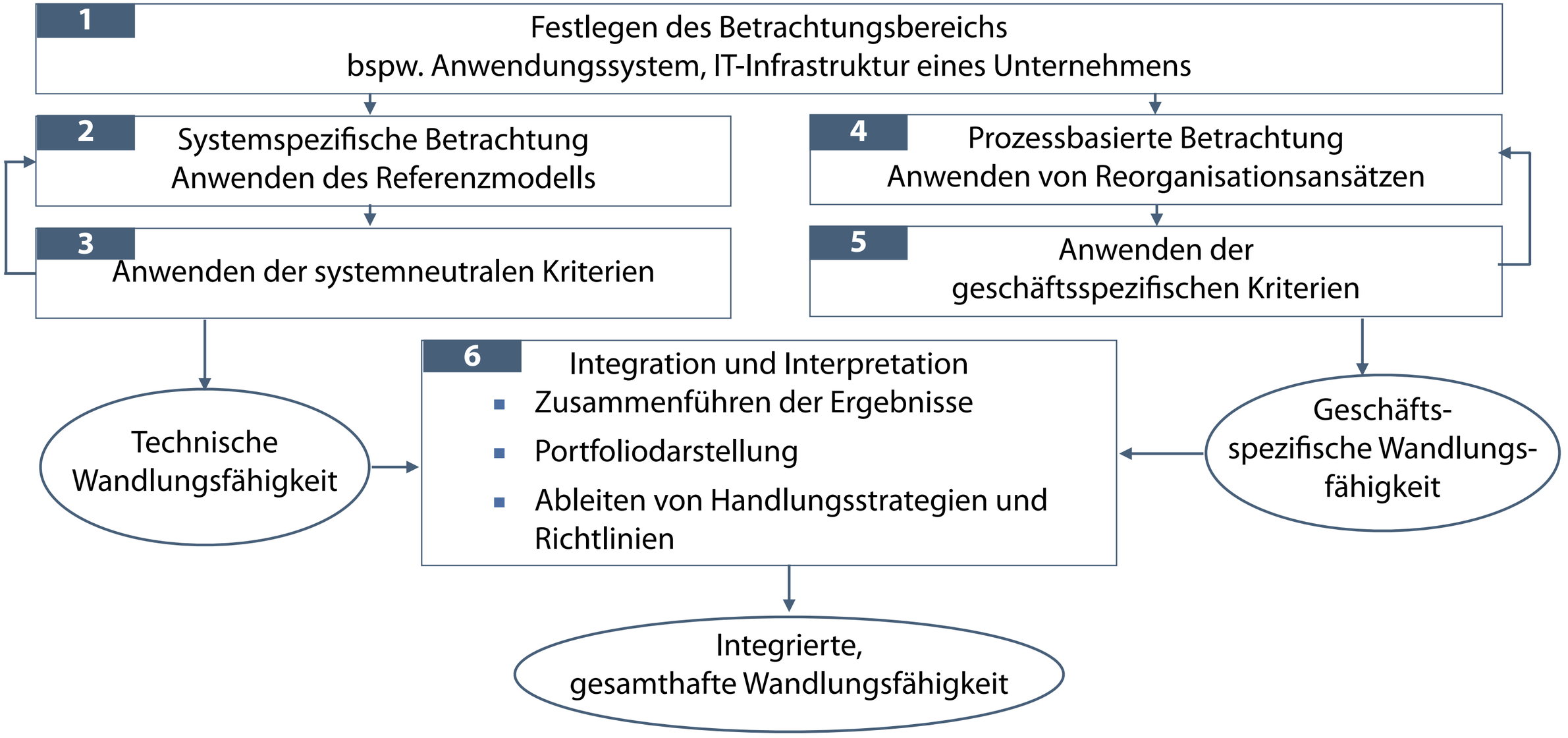
# Kriterien der Wandlungsfähigkeit




**Diese Kriterien können eingesetzt werden, um die Wandlungsfähigkeit eines konkreten Anwendungssystems zu überprüfen.**

Quelle: Gronau 2021, S. 51

# Vorgehen zur Ermittlung der Wandlungsfähigkeit

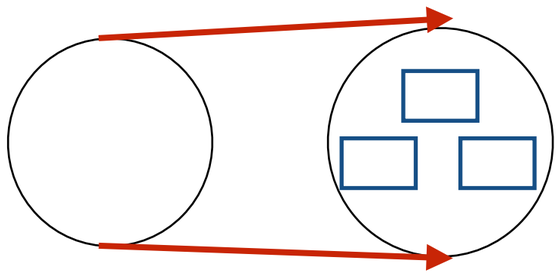



**Bewertung der Wandlungsfähigkeit in jeder Schicht des Referenzmodells mithilfe eines Fragenkataloges.**

Quelle: Gronau 2021, S. 55

# Beispiel: Wandlungsfähigkeit der Branche Handel

**Reorganisationtyp:**
**Subsystembildung**:

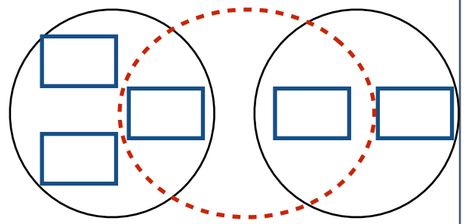


Autonome Filiale

Szenario:
Mehrere Filialen mit überregionalen und regionalen Angeboten

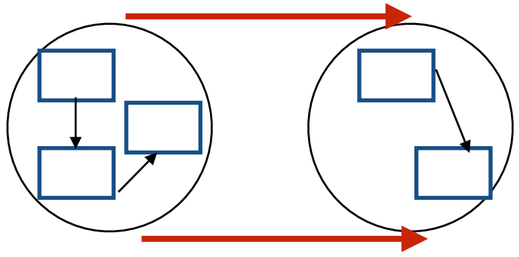
**Reorganisationtyp:**
**Auflösung der Unternehmensgrenzen**:

Mergers



Szenario:
Zusammenschluss von Unternehmen zur Erweiterung der Produktpalette

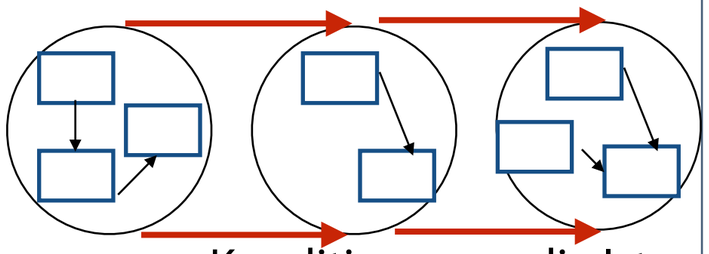
**Reorganisationtyp:**
**Prozessveränderung**:



Szenario:
Lieferantenauswahl in Echtzeit

**Reorganisationtyp:**
**Kontinuierliche Veränderung:**

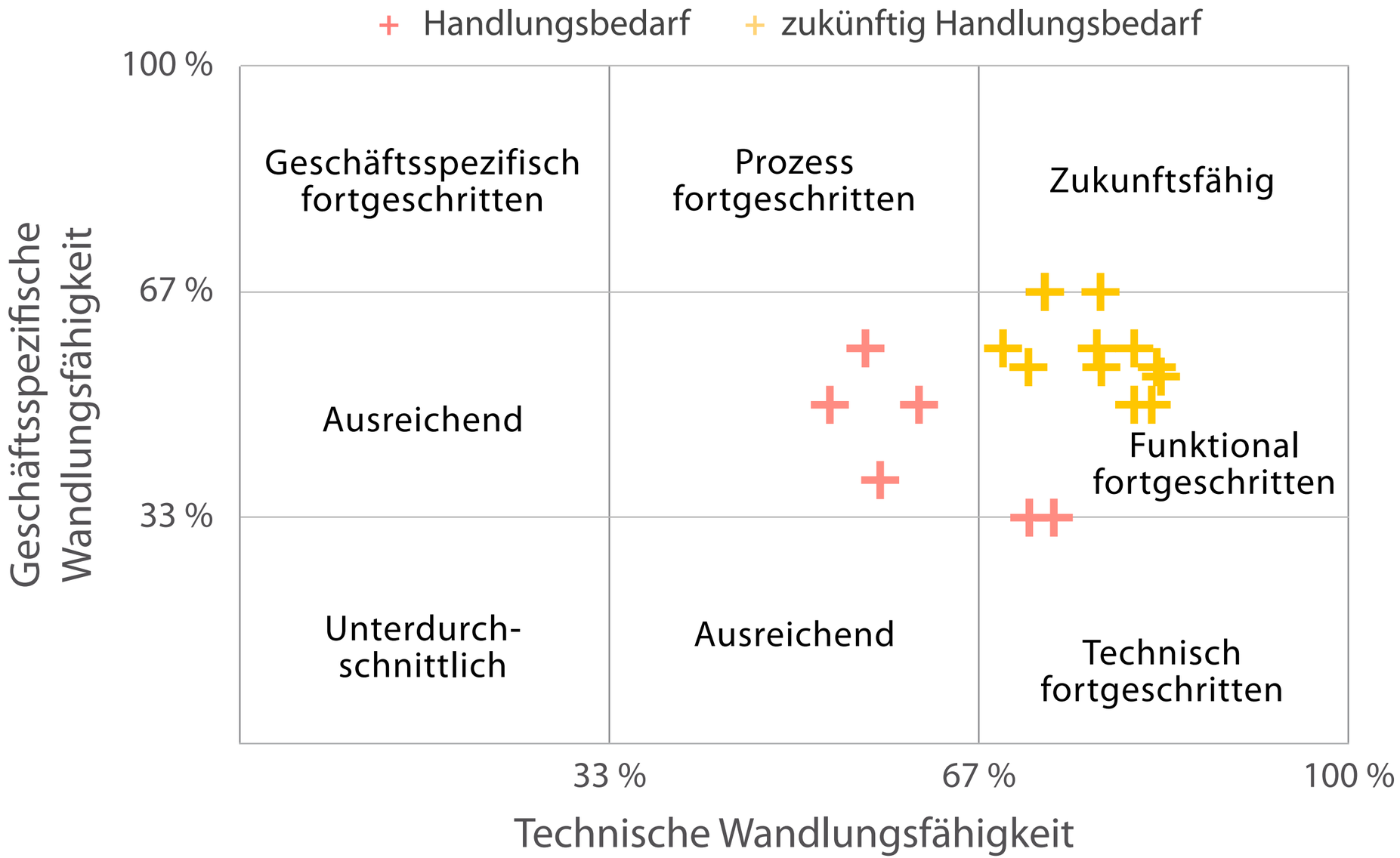
Konditionen



Szenario:
Dynamisches Anpassen von Konditionen an die Ist-Daten

Quelle: Gronau 2021 S. 56

# Bewertungsmatrix




**Einsatzmöglichkeiten bei Auswahl, Benchmarking, Schwachstellen- und Potentialanalyse.**

Quelle: Gronau 2021, S. 56

# Quick Check 3
# Vorlesung 09: Fragerunde 3

Auditorium Quiz App

STUDENT

https://quiz.lswi.de/login

Veranstaltungs-
schlüssel:

AWS

# Literatur


Ahlemann, F.; Stettiner, E.; Messerschmidt, M.; Legner, C. (2012): Strategic Enterprise Architecture Management –Challenges, Best Practices, and Future Developments. Heidelberg: Springer.

Andresen, K.; Gronau, N.; Schmid, S. (2005): Ableitung von IT-Strategien durch Bestimmung der notwendigen Wandlungsfähigkeit von Informationssystemarchitekturen. In (Ferstl, O.K.; Sinz, E.J.; Eckert, S.;

Dern, G.: Management von IT-Architekturen. Vieweg & Sohn Verlag, Wiesbaden 2006.

Erl, T.: Service-Oriented Architecture, Prentice Hall, New York 2005

Gronau  N.: ERP-Systeme, Architektur, Management und Funktionen des Enterprise Resource Planning, 4. Auflage, 2021.

Heuer, A.; Saake, G.: Datenbanken: Konzepte und Sprachen, 2. Auflage, 2000

Hohpe, G.; Woolf, B. (2003): Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Boston: Addison-Wesley.

Klaus, H., Rosemann, M., & Gable, G. G. (2000): What is ERP? Information Systems Frontiers, 2(2), 141-162.

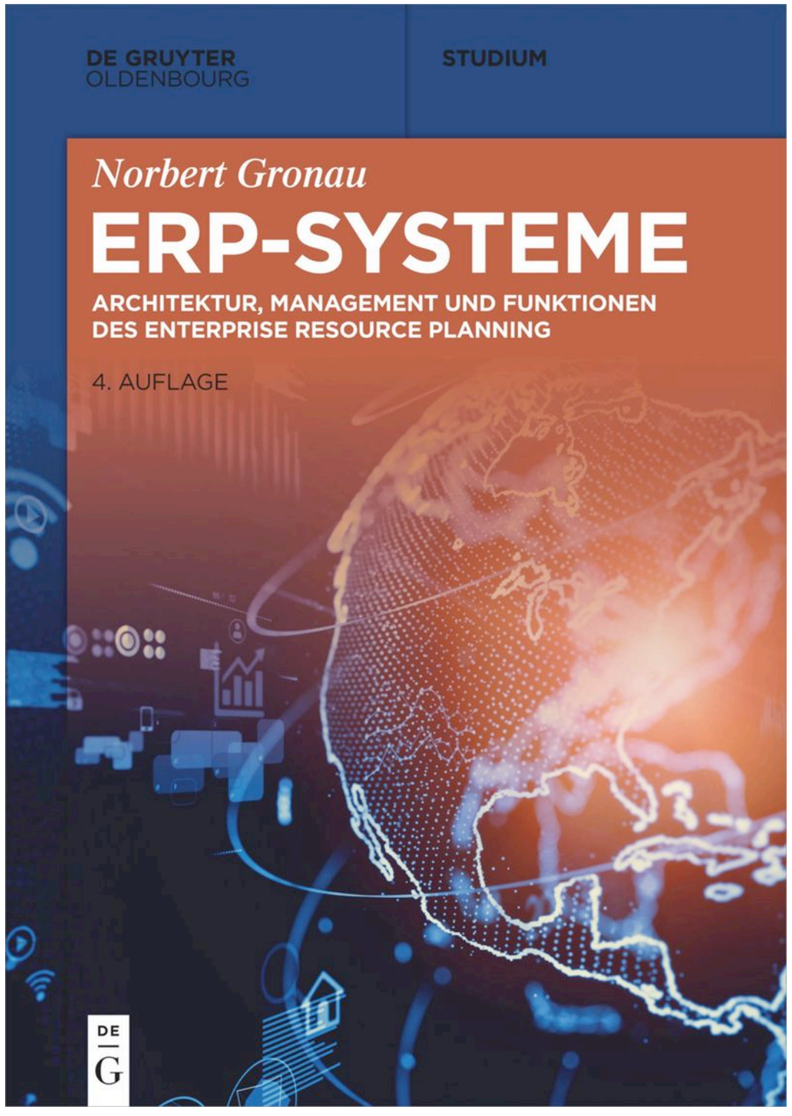
Linthicum, D. S. (2000): Enterprise Application Integration. Boston: Addison-Wesley.

Monk, E. F., & Wagner, B. J. (2012): Concepts in Enterprise Resource Planning (4th ed.), Cengage Learning.

Reussner, R.; Hasselbring, H. (2006): Handbuch der Software-Architektur. Heidelberg 2006.

Sinz, Elmar: Unternehmensarchitekturen in der Praxis – Architekturdesign am Reißbrett vs. situationsbedingte Realisierung von Informationssystemen In: Wirtschaftsinformatik, 46, 4, 2004, S. 315-316.

# Zum Nachlesen




Gronau, N.,
ERP-Systeme
Architektur, Management und Funktionen des Enterprise Resource Planning

4. Auflage, 2021

ISBN 978-3-11-066283-2

Über Verlag De Gruyter zu erwerben:

https://www.degruyter.com/document/doi/10.1515/9783110663396/html

**Kontakt**

Univ.-Prof. Dr.-Ing. Norbert Gronau

Center for Enterprise Research
Universität Potsdam
August-Bebel-Str. 89 | 14482 Potsdam
Germany

Tel. +49 331 977 3322
E-Mail ngronau@lswi.de