

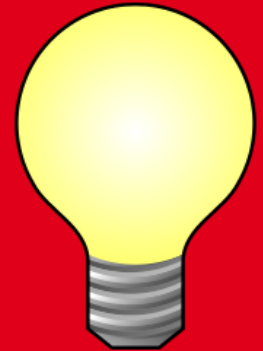


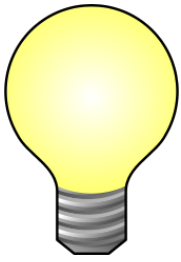
Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

14.01.2021

Programmieren im Großen III

Grobentwurf (Architektur)





Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

AGENDA

Einführung ins Thema

Begriffsabgrenzungen

Erste Schritte

Fundamental Modeling Concepts

Die 3-Schichten-Architektur

Den Grobentwurf überprüfen (Robustness Analysis)

Wie geht's dann weiter?

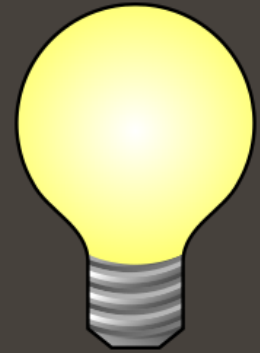
Fazit



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

01 EINFÜHRUNG INS THEMA

Ziel:
Die Eckpunkte des Themas kennenlernen

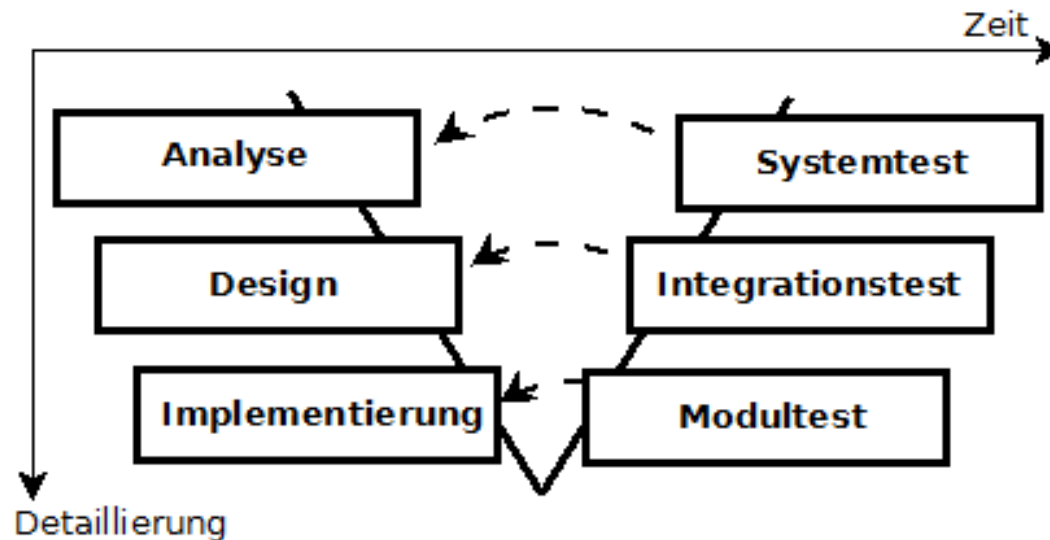


WORUM GEHT'S?



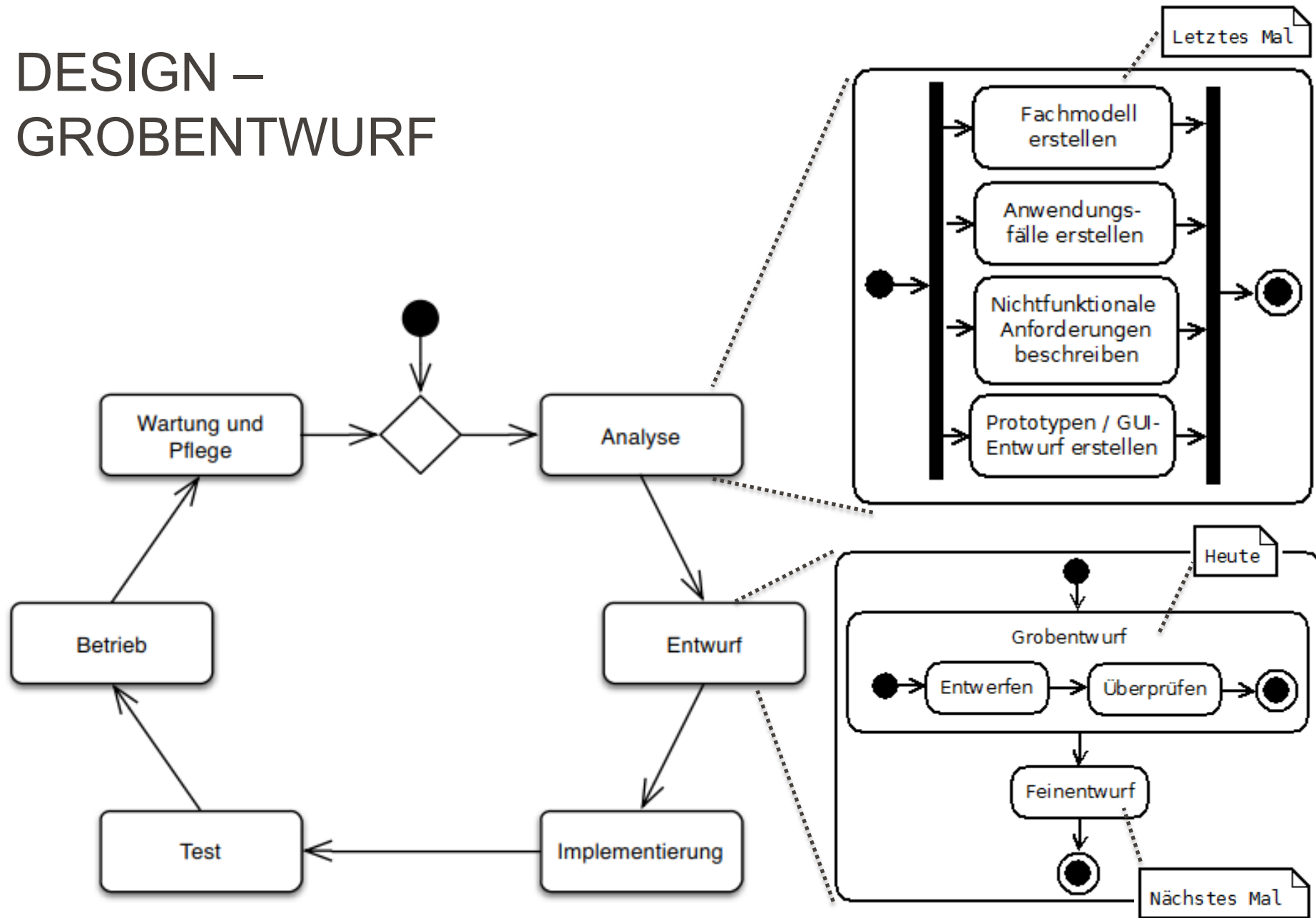
Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Wir brauchen ein Vorgehensmodell
 - Verschiedene Phasen:



- Heute besprechen wir:
 - Design: Grobentwurf (Architektur)

DESIGN – GROBENTWURF



ENTWURF (DESIGN) – WORUM GEHT‘S?



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

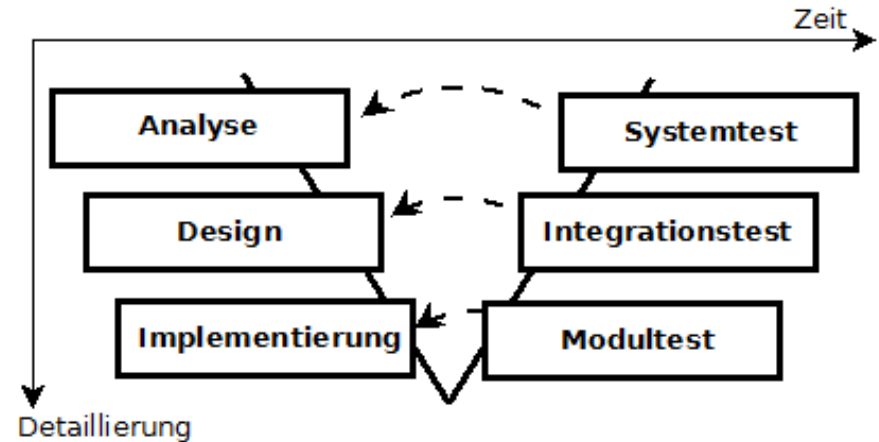
- Grundlegende Frage
 - Wie soll das zu bauende System sein?
- Entwurf (Design)
 - Tätigkeiten
 - Grobentwurf (Architektur) → heute
 - Feinentwurf (Detailed Design) → nächstes Mal
 - Sprechweise im (R)UP:
 - „Analysis and Design“
 - Sprechweise im V-Modell:
 - „Design“

HINWEIS: ANALYSE IN V-MODELL & RUP



Hochschule RheinMain
University of Applied Sciences
Wiesbaden Rüsselsheim

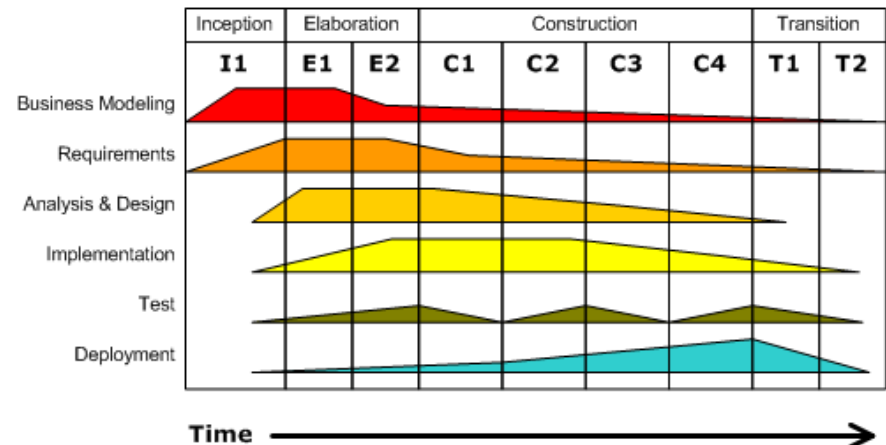
- Analyse im V-Modell:
 - Anforderungserhebung & Analyse (letzte Einheit)



- Analyse im (R)UP:
 - Grobentwurf erstellen (Architektur)
 - + Grobentwurf im Hinblick auf Anforderungen überprüfen (Robustness Analysis)

Iterative Development

Business value is delivered incrementally in time-boxed cross-discipline iterations.



GROBENTWURF – WORUM GEHT'S?



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Ziele:
 - Grobentwurf (Architektur) für das zu bauende System
 - Anforderungen sind vollständig und widerspruchsfrei
 - Grobentwurf erfüllt Anforderungen
 - System kann gebaut werden (build-able)



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

02

Begriffsabgrenzungen

Ziel:
Missverständnisse Vorbeugen
→ Begriffe sauber abgrenzen





VORSICHT: EIN GERÜCHT GEHT UM



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Manche Autoren behaupten . . .
 - Fachmodell (aus der Analyse)
 - + Typen
 - + Methoden
 - ⇒ OO-Programm

 Das funktioniert nur in einfachen Fällen!

→ Meistens:

- Mehrstufiger Entwurfsprozess
- Weitere/andere Klassen
- Umcodierung der Daten (andere Datenstruktur)
- . . .



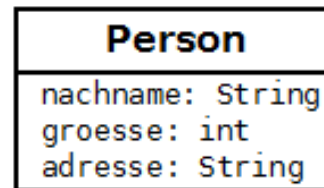
VORSICHT: EIN GERÜCHT GEHT UM



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

→ ***Fachliche Sicht betrachtet alles ohne technische Details***

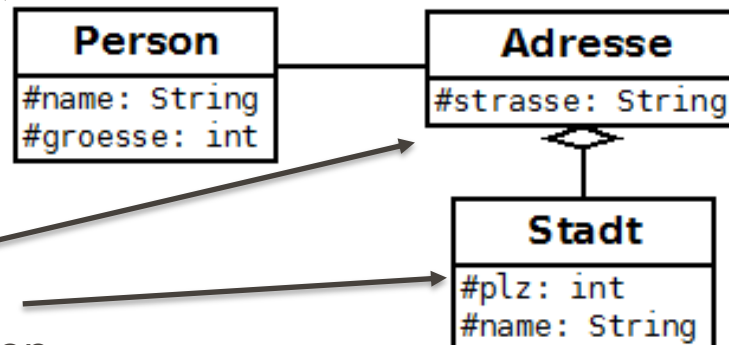
- Wird in der Analyse verwendet, um **NUR** die fachlichen Anforderungen zu ermitteln → Fachmodell



Hier funktioniert es
schon mal nicht!!

→ ***Technische Sichten***

- In Architektur, Detailed Design, Implementierungsdoku verw.
- Es werden auch die für die Lösung **relevanten** technischen Details dargestellt:



Technische Lösung
z.B. für spezifische Suche
nach Städten oder Adressen

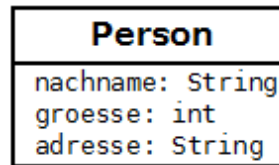
BEGRIFFE ZUR KLAREREN ABGRENZUNG:



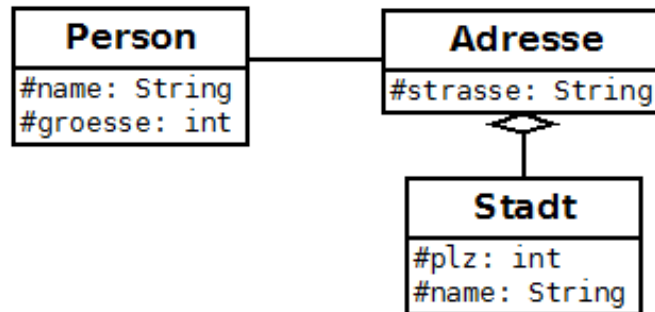
Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

Zur klareren Abgrenzung verwenden wir folgende Begriffe:

- In (Anforderungs-)Analyse:
 - **Fachmodell**: Klassenmodell zur fachl. Beschreibung der Anf:



- In Architektur (Grobdesign) und Detailed Design (Feindesign):
 - **Domänenmodell**: techn. Klassenmodell der Fachklassen



Vorsicht: Wir nennen das so, leider werden in der Literatur oft die Begriffe für Fachmodell, Domänenmodell, fachl. Datenmodell, ... durcheinandergewürfelt

BEGRIFFE ZUR KLAREREN ABGRENZUNG:



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

Generell gilt damit:

- In (Anforderungs-)Analyse:
 - **Fachmodell***: Klassenmodell zur fachl. Beschreibung der Anf.
 - + Dynamische Aspekte (z.B. Use Cases)
- In Architektur (Grobdesign) und Detailed Design (Feindesign):
 - **Domänenmodell***: techn. Klassenmodell der Fachklassen
 - + Dynamische Aspekte (Geschäftsregeln)
 - Meist als Methoden in Klassen des Domänenmodells
 - Oder eigener Controller (siehe später MVC-Muster)
 - + Sehr viele weitere Klassen für das „DrumHerum“ (GUI, Persistenz, ...)

} Geschäfts-
logik

*Unsere Terminologie, damit Sie es besser unterscheiden können



Vorsicht: Auch Geschäftsregeln (Business Rules), Geschäftslogik, ...
werden oft im Zusammenhang mit Anforderungsanalyse
benutzt und es gibt auch hier keine klare Abgrenzung!



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

03

Erste Schritte

Ziel:
Erste Schritte anhand eines Beispiels

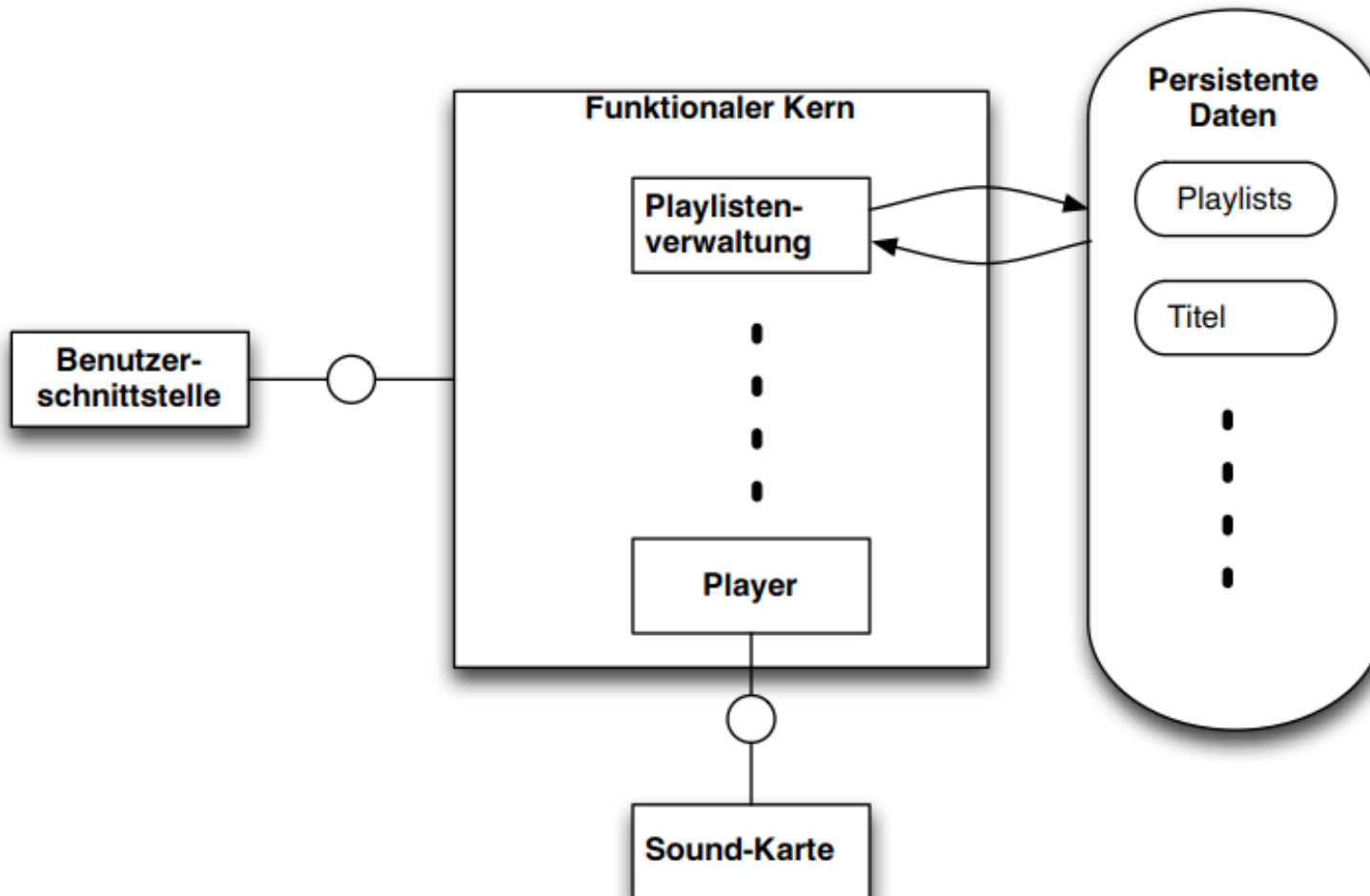


BEISPIEL GROBENTWURF



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- BSP: Grobentwurf MP3-Player (unvollständig):

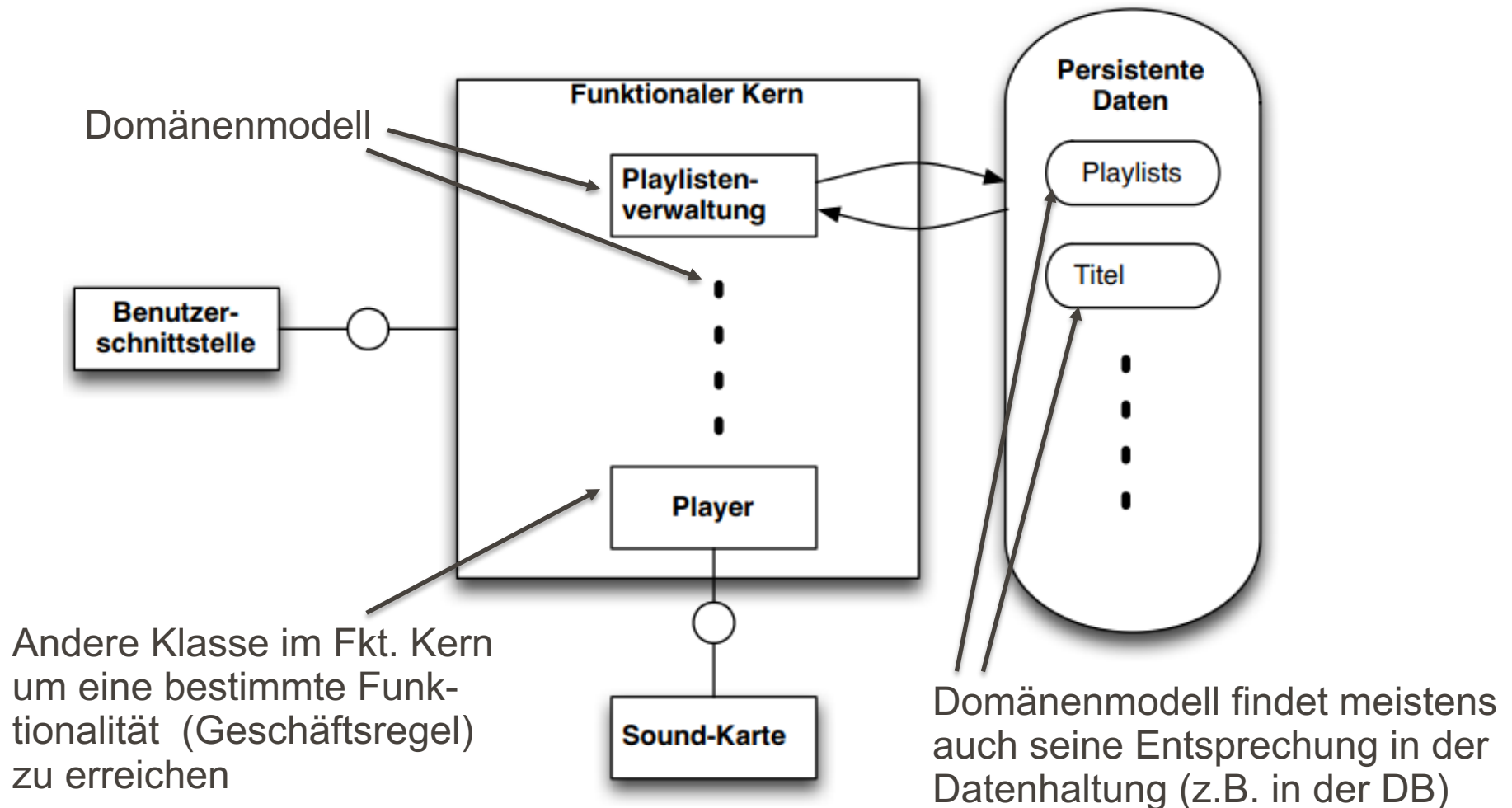


BEISPIEL GROBENTWURF



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- BSP: Grobentwurf MP3-Player (unvollständig):



ARCHITEKTUR (GROBENTWURF)



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Ziel: Grundlegenden Aufbau der Anwendung festlegen
- Input:
 - Anforderungen
 - Andere Entscheidungen (dokumentieren!)
- Muss für Architektur:
 - erfüllt Anforderungen
 - „solide“ (Architektur verträgt z.B. problemlos Änderungen an den funktionalen Anforderungen)
 - Leider schwer die Änderungen vorherzusehen
 - Leider auch so schwer zu überprüfen
 - Erfahrung



Vorsicht:

- Architektur-Entscheidungen haben weitreichende Konsequenzen!
- Genau analysieren und die wichtigsten Dokumentieren!



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

04

Fundamental Modeling Concepts (FMC)

Ziel:

Die Architekturentwurfssprache FMC kennenlernen



WIE KANN MAN DIE ARCHITEKTUR MODELLIEREN?



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Mit UML
 - Klar, ist das die Intention der UML
 - Allerdings gibt es ein paar Defizite:
 - Konzentriert sich sehr auf die Datenmodellsicht (Z.B. Klassen)
 - Datenspeicherebene wird vernachlässigt
 - Zugriff auf Datenspeicher wird nicht thematisiert
 - Rel. frühe Phase → Manchmal noch gar nicht klar, ob ein Konzept wirklich eine Klasse wird (oder eher Attribut, Methode, oder ...)
 - Kompositionsstruktur-Diagramme wären prinzipiell geeignet
 - ABER für den Zweck recht umständlich
 - Verteilungsdiagramme scheinen sich anzubieten
 - ABER für Software-Architektur nur bedingt geeignet (Fokus: Verteilung der Software auf Hardware)
 - UML ist nicht so gut geeignet

WIE KANN MAN DIE ARCHITEKTUR MODELLIEREN?



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Hilfreiche **und** einfache Notation für Architektur nötig
 - FMC-Block-Diagramme
- BEM: Auch über Profilmechanismus könnte man UML auch „umbiegen“ eine ähnliche Syntax & Semantik wie FMC-Blockdiagramm zu haben
 - Technical Architecture Management (TAM) → FMC + UML
(<http://www.fmc-modeling.org/>)

FMC – ALLGEMEINE INFOS



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

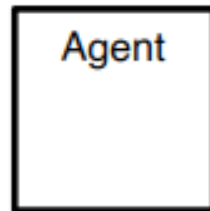
- FMC = Fundamental Modeling Concepts
 - Siehe: <http://www.fmc-modeling.org/>
 - Im Umfeld von SAP (Hasso-Plattner-Institut) entwickeltes Modellierungskonzept für Architekturen
 - Ursprünge schon aus den 1970ern in Uni Kaiserslautern entwickelt
- FMC besteht eigentlich aus 3 Diagrammarten:
 1. FMC-Block-Diagramme → Struktur der Software
 2. Petri-Netze → Dynam. Abläufe / „Geschäftsprozesse“
 3. Wertebereichsdiagramme (WBD) → Erweiterte Entity-Relationship-Diagr.
- Vorschlag hier:
 - 2 & 3 kann man auch sehr gut durch UML abdecken (vgl. TAM)
 - 2. Petri-Netze → Aktivitätsdiagramme (oder BPMN, ...)
 - 3. WBD → UML-Klassendiagramm (→ siehe Domänenmodell)
 - Block-Diagramm bringt aber Vorteile für die Architekturübersicht

FMC-BLOCKDIAGRAMM – KOMPONENTEN:



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Agent (= aktive/interagierende Komponente):



- Kommunikationskanal (= passive Komponente ohne Gedächtnis):



→ Verbindet Agenten

- Speicher (= passive Komponente mit Gedächtnis):

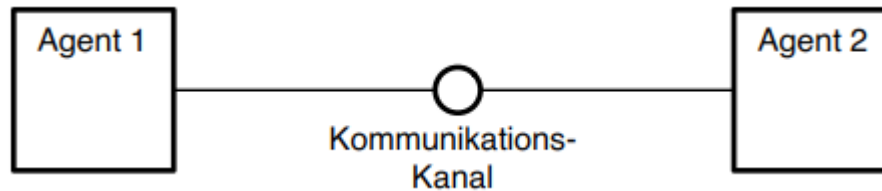


NOTATION FMC-BLOCKDIAGR. – KOMMUNIKATION

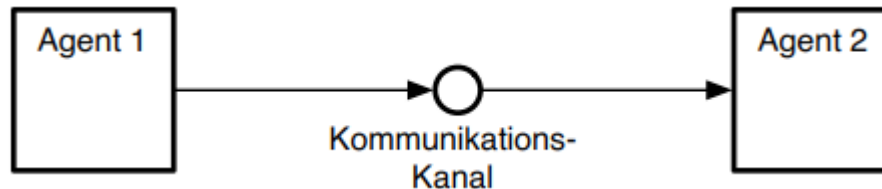


Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

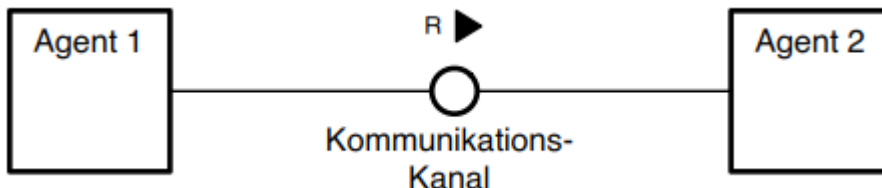
- Kommunikation zwischen Agenten:
 - Richtung unspezifisch (unbekannt/unwichtig):



- Unidirektional (nur eine Richtung):



- Anfrage/Antwort:

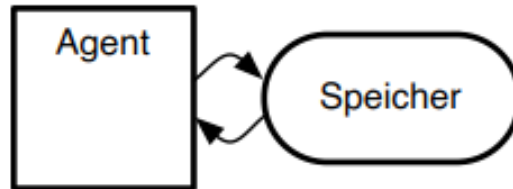


NOTATION FMC-BLOCKDIAGR. – KOMMUNIKATION

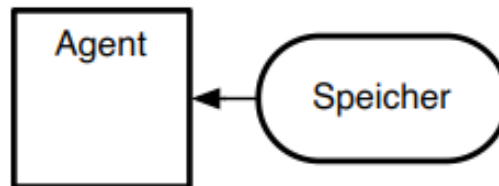


Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

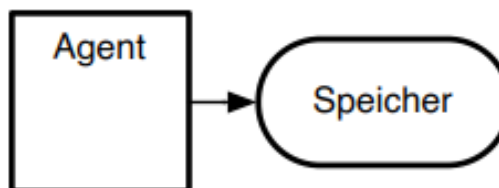
- Speicherzugriff :
 - lesen + schreiben (Normalfall):



- nur lesen (häufiger Fall):



- bei jedem Zugriff erst löschen, dann schreiben (seltener Fall):

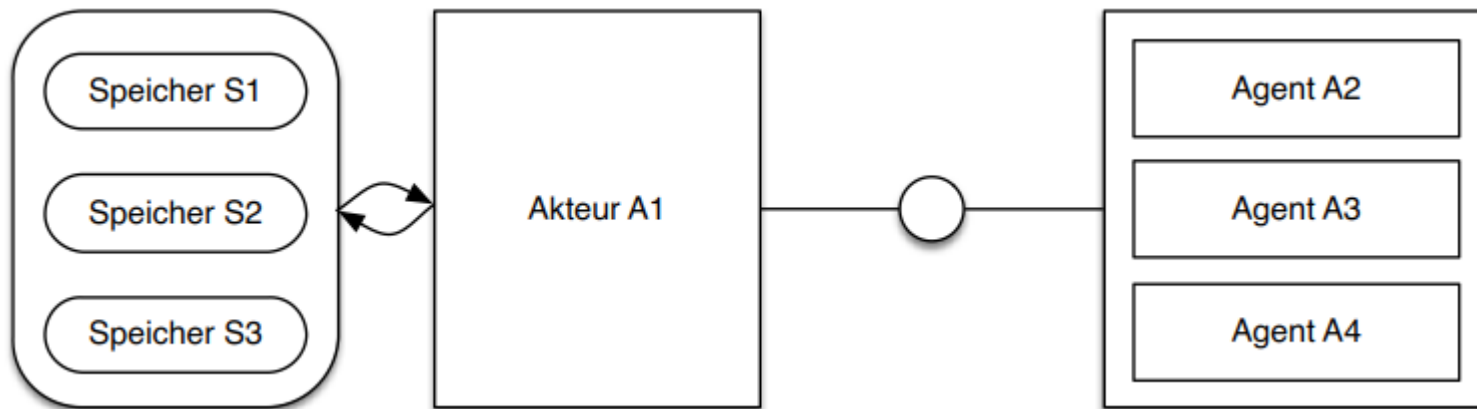


NOTATION FMC-BLOCKDIAGR. – KURZSCHREIBWEISE

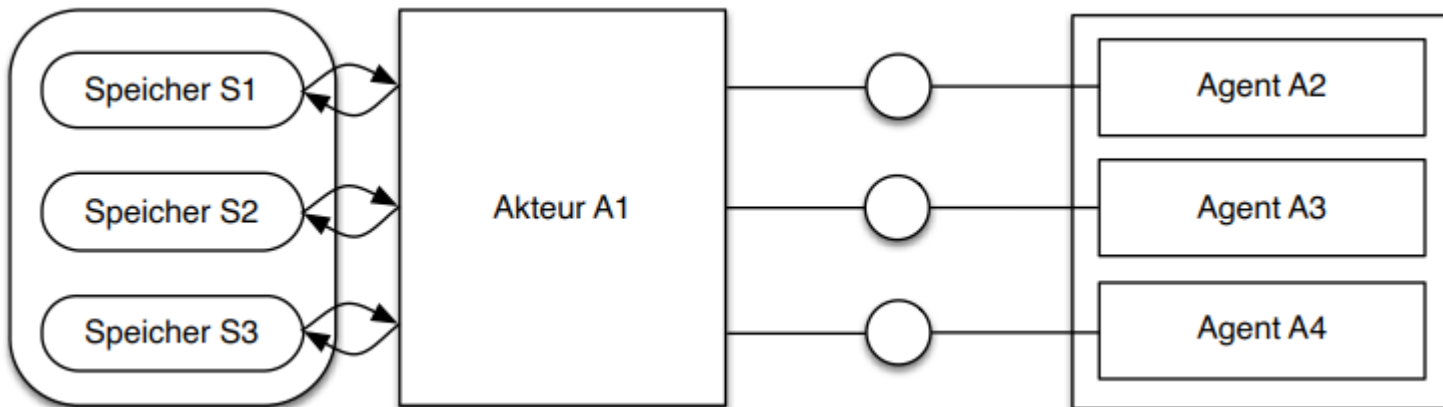


Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Zwei gleichwertige FMC-Block-Diagramme:
 - „Kurzschreibweise“:



- „Ausgeschrieben“:





Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

05

Die 3-Schichten Architektur

Ziel:

Das Architekturmuster 3-Schichten-Architektur
kennenlernen

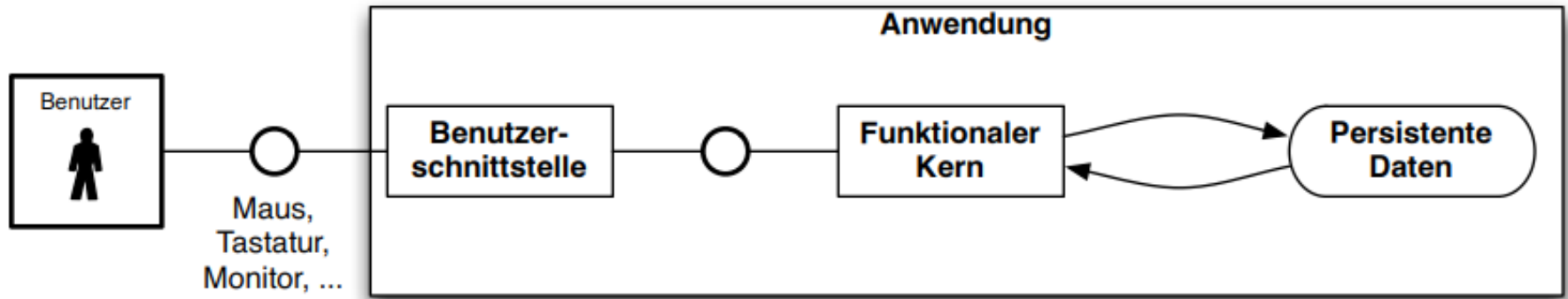


DIE 3-SCHICHTEN-ARCHITEKTUR



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- 3-Schichten-Architektur (Engl.: 3-Tier Architecture):



→ Beliebter Architektur-Ansatz (Muster)

- Aufgabenteilung in 3 Schichten:
 - Benutzerschnittstelle:
 - Darstellung + Benutzereingaben
 - Funktionaler Kern:
 - Domänenmodell + Anwendungs-/Geschäfts-Logik
 - Datenhaltung: siehe Vorlesung „Datenbanksysteme“

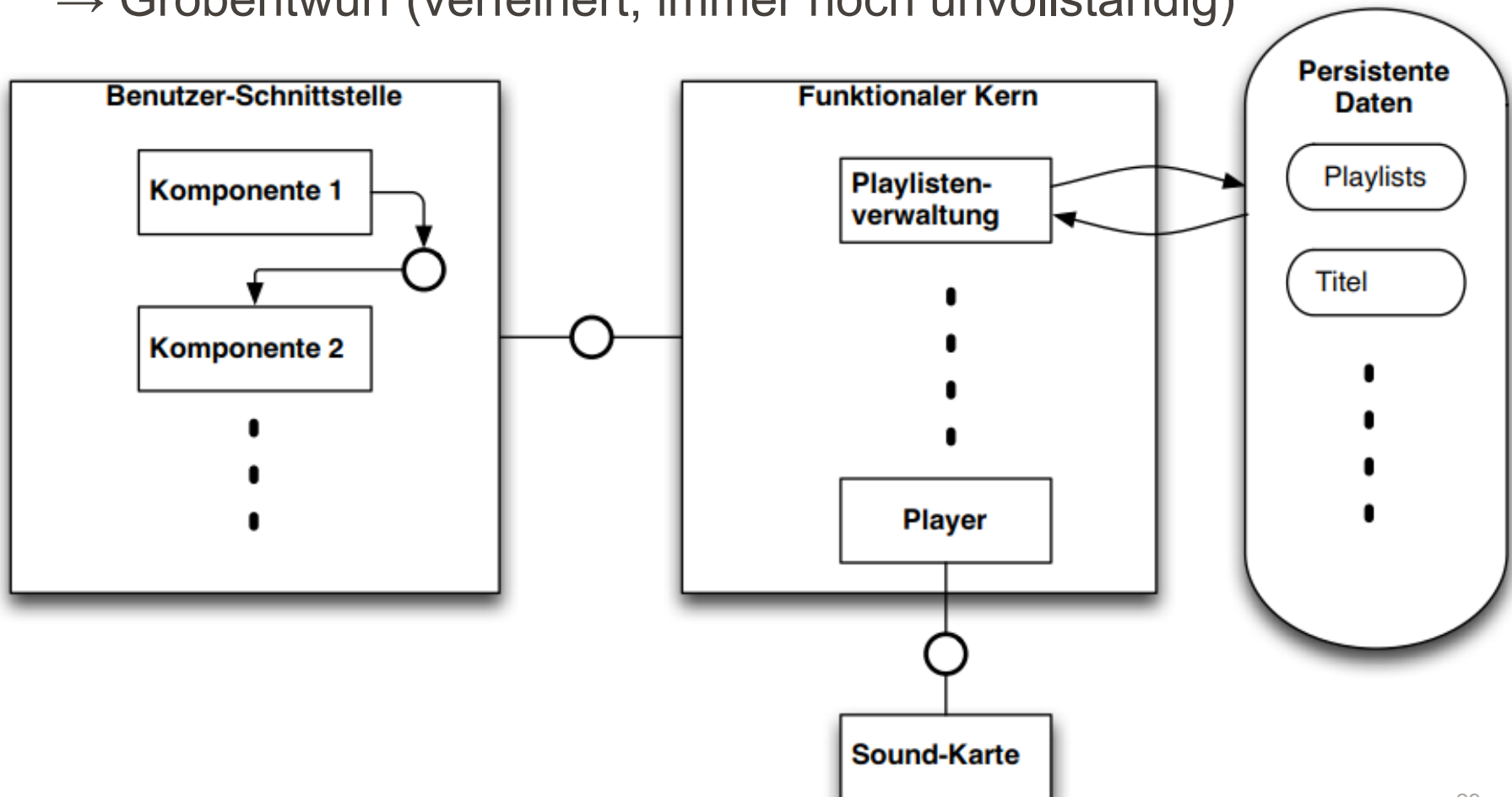
BEISPIEL

3-SCHICHTEN-ARCHITEKTUR



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Beispiel: MP3-Player in 3-Schichten-Architektur:
→ Grobentwurf (verfeinert, immer noch unvollständig)



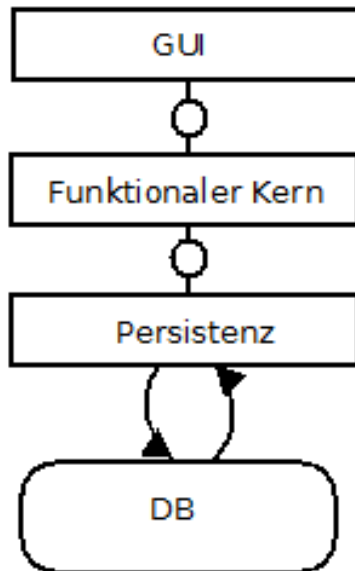
HINWEIS ZUR 3-SCHICHTEN-ARCHITEKTUR



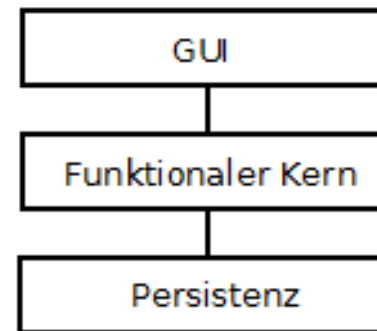
Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Wird oft eher umgekehrt von oben nach unten dargestellt
→ Schichten

So (z.B. FMC):



Oder auch so (z.B. UML):



- Oft wird DB weggelassen und als Teil der Persistenzschicht betrachtet
- Kann aber auch ohne DB sein (z.B. dann XML-Dateien, ...)
 - Deshalb: Besser auch die DB, ... darstellen



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

06

Den Grobentwurf überprüfen

Ziel:

Den Grobentwurf mittels Reviews überprüfen

→ Am Beispiel der Robustness Analysis (aus RUP)



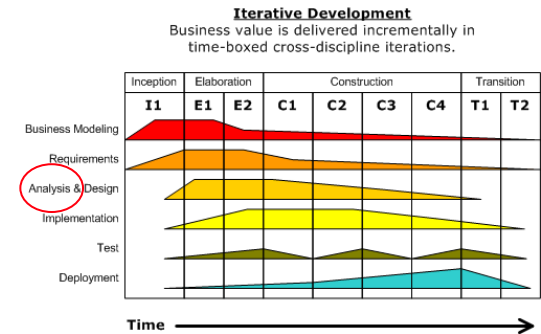
GROBENTWURF ÜBERPRÜFEN – GRUNDIDEE



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Typische Fragen / Probleme beim Grobentwurf
 - Sind alle Anforderungen abgedeckt?
 - Haben wir etwas vergessen?
 - Kann das funktionieren?
 - . . .
- Wie herausfinden? → Robustness Analysis
 - Grobentwurf „zum Leben erwecken“:
 - Anwendung als Maschine betrachten
 - Mit der Maschine konkrete Szenarien durchspielen
 - Sequenz-/Kommunikations-Diagramme zeichnen
 - Welche Szenarien?
 - Siehe: Anwendungsfälle (Use Cases)

EXKURS – DAS RUP ANALYSIS PROFILE:



- Es gibt auch noch ein Profil für die Analyse im RUP
→ Erweiterung der UML über Stereotypen, ...
- Elemente einer Anwendung können in folgende Elemente untergliedert werden:
 - ⋈ «Actor» – Akteure (siehe UseCase-Diagramm)
 - ⊖ «Boundary» – Schnittstellen-Komponenten (z.B.: UI)
 - ⦿ «Control» – Programmlogik / Kontrollfluß
(z.B.: Geschäftslogik / Veränd. von Entity-Objekten)
 - ⊙ «Entity» – Datenelemente
(z.B. Klassen im Fach-/Domänenmodell)
- Mögliche Darstellung in UML:







ROBUSTNESS ANALYSIS DURCHFÜHREN



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Vorgehensweise:

1. Zerlegung der Anwendung in

-  Actor-Objekte (Benutzer, Externe Systeme)
-  Boundary-Objekte (Schnittstellen-Komponenten)
-  Control-Objekte (Programmfluss)
-  Entity-Objekte (Fachliche Daten)

} \triangleq Architektur

2. Für jeden Anwendungsfall durchspielen:

- Standardablauf (= 1 Szenario)
- Alternative Szenarien (Alternative Schritte)

3. Bei Fehlern/Unstimmigkeiten:

- Anforderungen korrigieren und/oder
- Grobentwurf korrigieren/verfeinern
- Zurück zu 2.

ROBUSTNESS ANALYSIS DURCHFÜHREN



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Durchspielen
 - BSP Gemeinsam an der Tafel / Dokumentenkamera
 - FMC-Entwurf
 - Szenarien
 - Sequenz-/Kommunikations-Diagramme
 - Siehe dazu PDF auf der Homepage

ROBUSTNESS ANALYSIS – WOZU?



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Vorteile
 - + frühzeitig Fehler erkennen
 - + besseres Verständnis der Architektur
 - + Risiko verringern

(Architektur wird auch gern für weitere Planung der Arbeitspakete verwendet, ...)
- Nachteil
 - (zunächst) höherer Aufwand

→ Beispiel für eine Art von Architekturreview

→ Immer empfohlen!



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

07

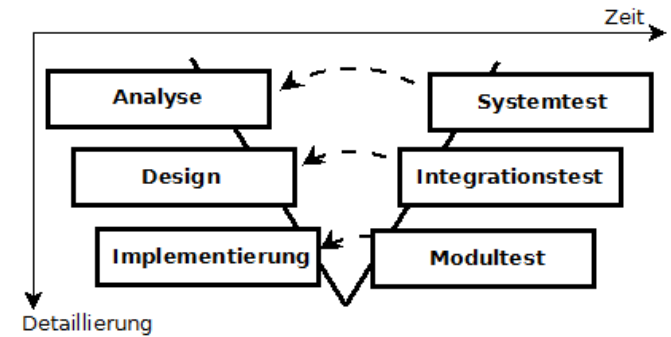
Wie geht's dann weiter?

Ziel:

Was passiert im Anschluß?



TESTVORBEREITUNG

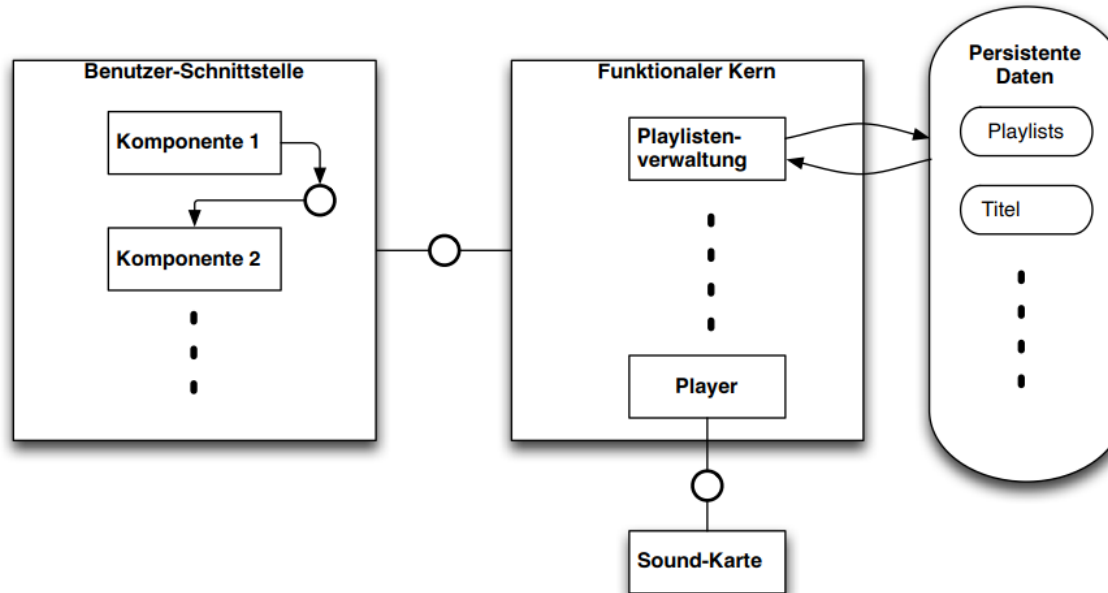


- Weiterer Bestandteil der Qualitätssicherung:
→ Testvorbereitung
- Vorbereitung des Integrationstests (Details später)
 - Input: Grobentwurf
 - Welche Komponenten gibt es?
 - Wie kommunizieren diese miteinander?
 - Output:
 - Grundlegender Ansatz (bottom-up, top-down, . . .)
 - Top-Level-Testfälle
(Zusammenstecken der letzten Verbindungen.)

VORSCHAU: FEINENTWURF



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim



→ Feinentwurf/Implementierung

- Wie werden FMC-Komponenten dann im Detail umgesetzt?
- Mehrere Möglichkeiten:
 - FMC-Komponente → Klasse(n), Attribut(e), Methode(n), . . .
- UND:
 - Spätestens jetzt entsteht das Domänenmodell



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

08

Fazit

Ziel:
Was haben wir damit gewonnen?



WAS HABEN WIR GELERNT?



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

- Entwurfsphase teilt sich in zwei Unterphasen:
 - Grobentwurf
 - Feinentwurf
- Grobentwurf
 - Festlegen der Architektur
 - FMC-Notation für Blockdiagramme
 - 3-Schichten-Architektur
 - Architekturmuster → Siehe spätere Vorlesung über Muster
 - Wahrscheinlich populärste Architektur für PC-Softwaresysteme
- Grobentwurf überprüfen
 - Z.B.: Robustness Analysis
- Ab hier kann man auch den Integrationstest vorbereiten



- Object-Oriented Analysis and Design:
 - Kleuker: Grundkurs Software-Engineering mit UML
[<http://dx.doi.org/10.1007/978-3-8348-9843-2>]
 - Zuser et al: Software-Engineering mit UML und dem Unified Process [BF 500 92]
 - C. Larman: Applying UML and Patterns [30 BF 500 78]
- FMC: Compositional structures and block diagrams
 - P. Tabelaing: Home of Fundamental Modeling Concepts
[<http://www.fmc-modeling.org/>]
 - Knoepfel et al: Fundamental modeling concepts [30 BF 000 158]



Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

AUF GEHT'S!!

SELBER MACHEN UND LERNEN!!

