Quasar: *Qua*litäts*s*oftware*ar*chitektur

- Architektur von Informationssystemen
- Architektur grafischer Bedienoberflächen

- Literatur

Der Begriff, das Buch

Quasar: **Qua**litäts**s**oftw**ar**earchitektur; Qualität auf 4 Ebenen

1. Ideen und Konzepte (vgl. Parnas)
2. Begriffe (klar definiert, z.B. Architektur, Komponente, Schnittstelle)
3. Standard-Architekturen und Standard-Schnittstellen für verschiedene “Themen”, z.B. Datenbankanbindung
4. Standard-Komponenten hinter definierten Schnittstellen, z.B. für Autorisierung, Persistenz

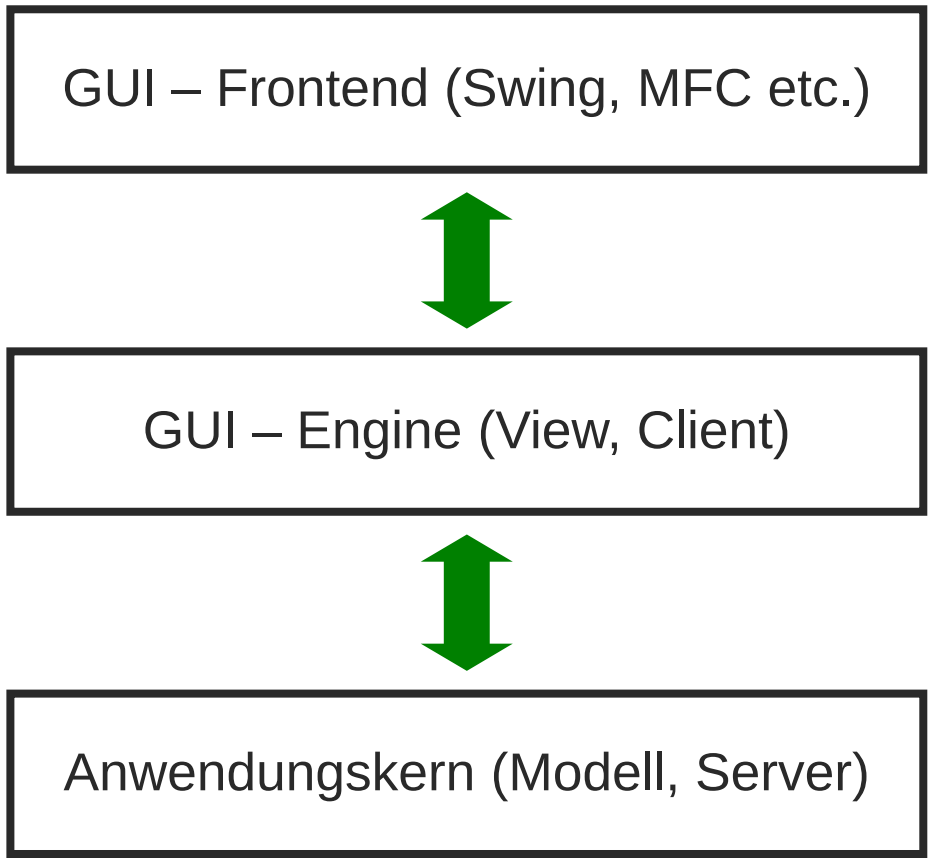
Siedersleben:
Moderne Softwarearchitektur,
dpunkt-Verlag,
ISBN: 3-89864-292-5

Signatur: E IV 15510

Komponenten von Informationssystemen (mit grafischer Bedienoberfläche)

Quasar

- IS-Architektur
- GUI-Architektur
- Exkurs: *Use Cases*
- Literatur
- ...
- ...
- ...
- ...
- ...
- ...
- ...
- ...



Architektur von Informationssystemen: Prinzipien [sehr frei nach Siedersleben]

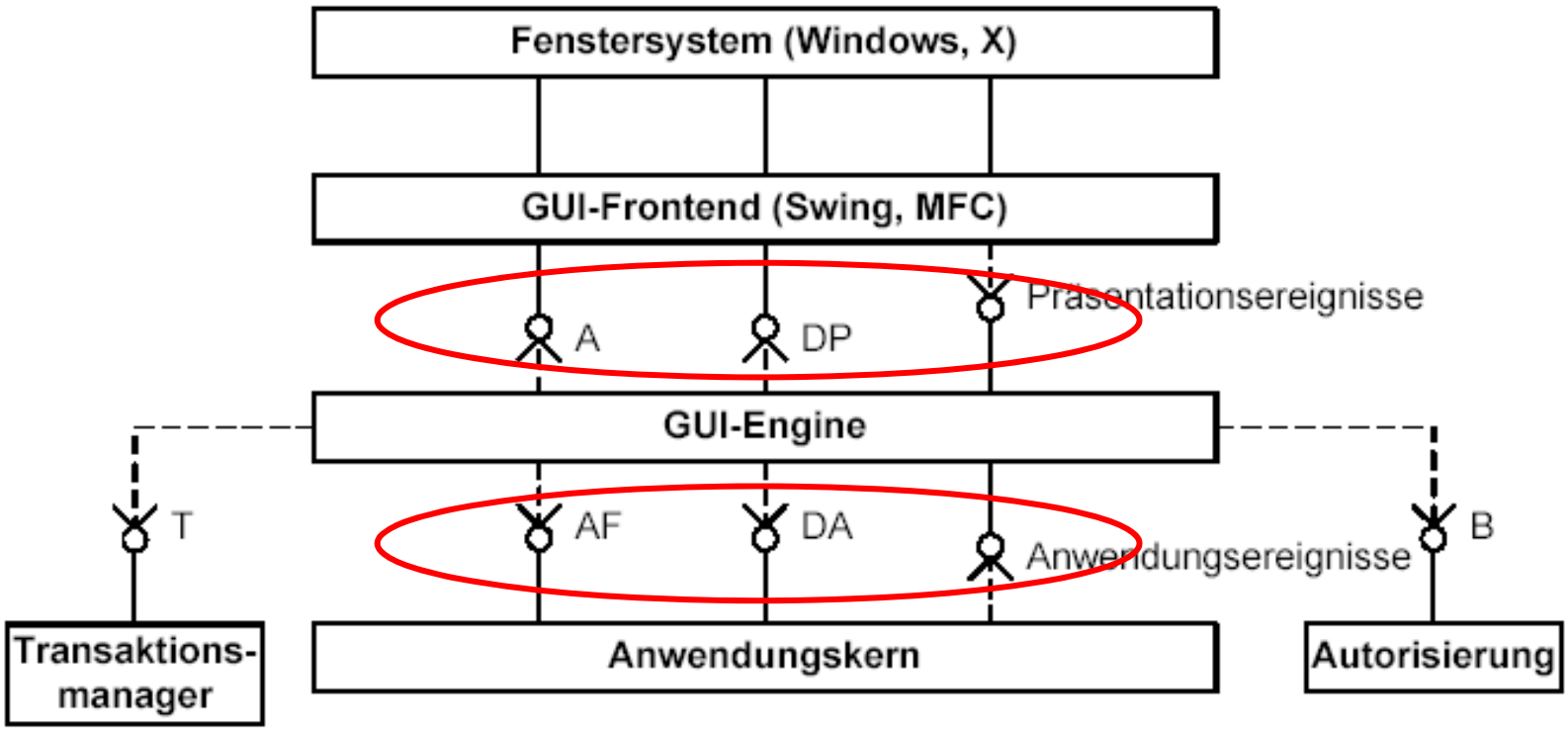
- Der **Anwendungskern** (Server) enthält *nur* die *fachlichen* Dinge (= Anwendungslogik)
  - Das (G)UI ist drumherumgebaut
  - Z.B. übernimmt eine intelligente Persistenzkomponente die Datenbankansteuerung (Technik *nicht* mit fachlichen Dingen *vermischen*)
- Prinzipien:
  Der Anwendungskern verhält sich defensiv und macht möglichst wenig Annahmen über seine Aufrufer
  - Er prüft alle Eingaben (und verlässt sich nicht darauf, dass diese an der Bedienoberfläche bereits geprüft wurden; z.B. mit *Java-Assertions* prüfen)

  Konsequenzen:
  - Das System ist robuster, leichter zu verstehen, zu ändern und zu erweitern
  - Kaum Mehraufwand in der Implementierung, aber
  - Mehr Nachdenken beim Architekturentwurf gefordert (danach wird es einfacher!)

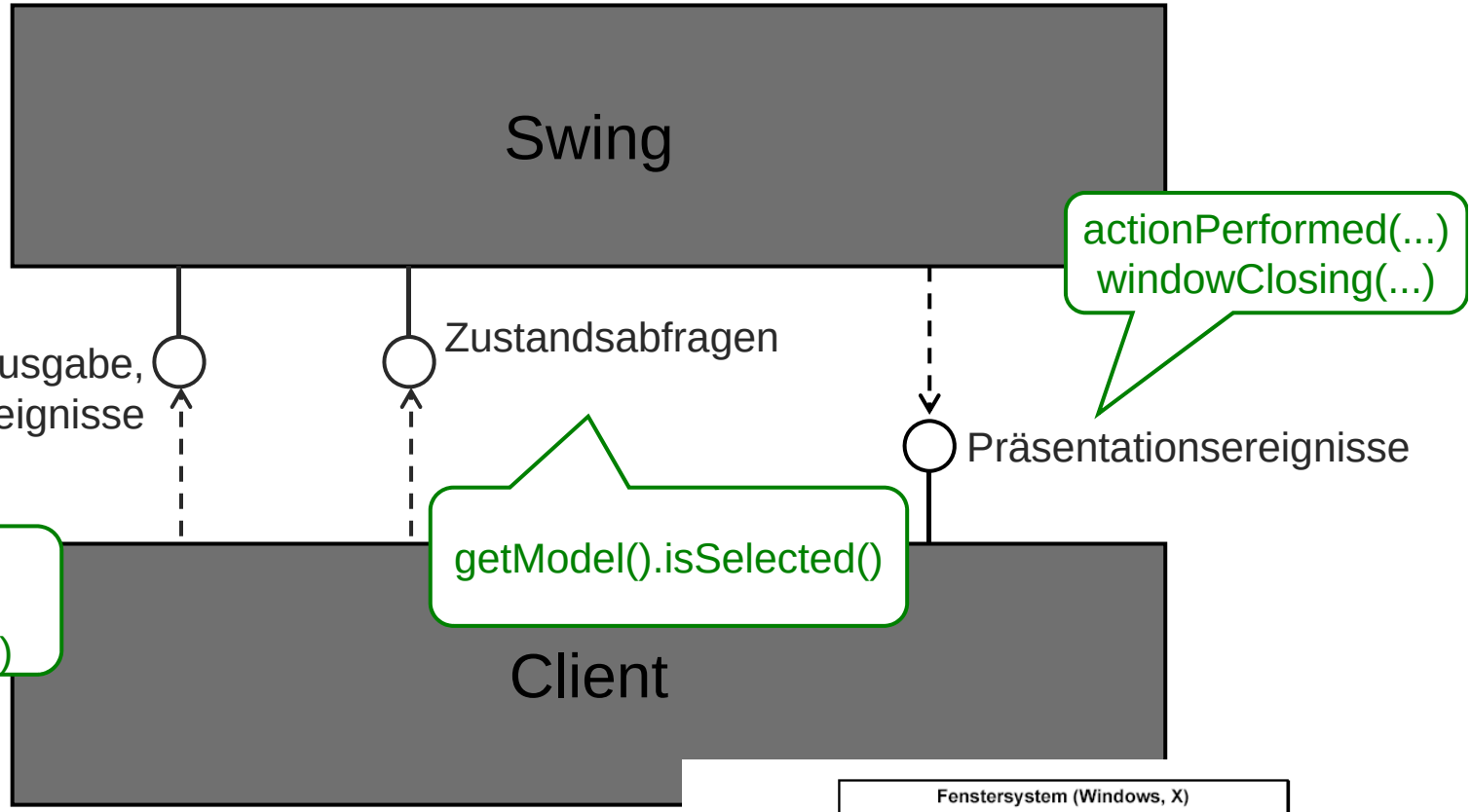
Architektur grafischer Bedienoberflächen: Übersicht [sehr frei nach Siedersleben]

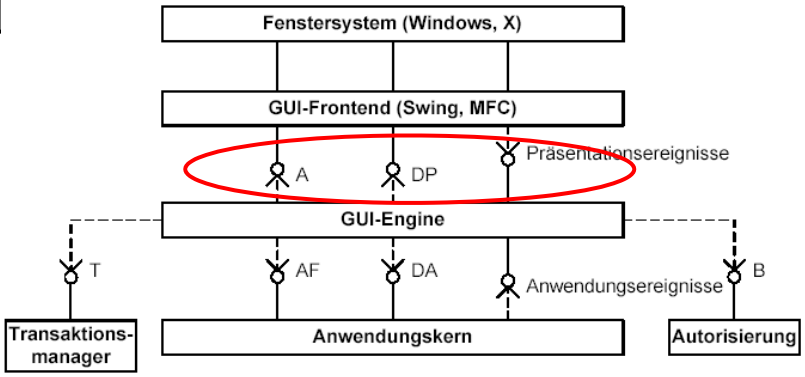
Quasar

- IS-Architektur
- GUI-Architektur
- Exkurs: *Use Cases*
- Literatur
- ...
- ...
- ...
- ...
- ...
- ...
- ...
- ...



Kommunikation zwischen Komponenten 1/2





Kommunikation zwischen Komponenten 2/2

Anwendungsfälle, Registrierung für Ereignisse

...

...

...

...

...

shootAt(...)
addPlayer(…)

keine zustandsverändernden Methoden!

Client

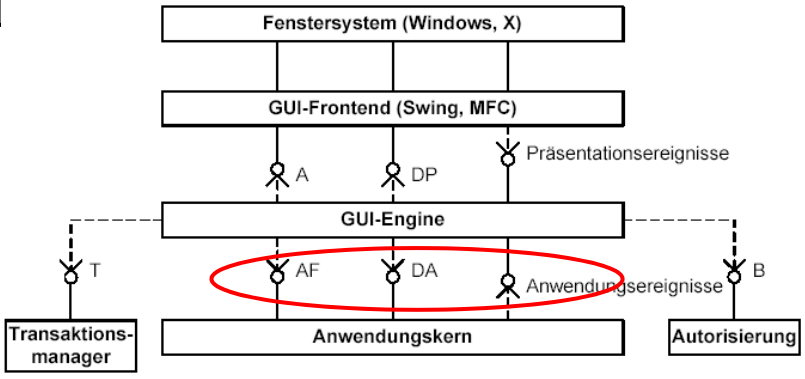
Server

getNextShipToSet()

abortByPartner()

Zustandsabfragen

Anwendungsereignisse (*update* / *notify*)



Synchronisation zwischen den Komponenten

- Der Client kennt das GUI-Frontend (Swing), aber ***nicht umgekehrt***
  - Er registriert sich z.B. bei Swing als Listener für Events
  - Er aktualisiert das Frontend bei Bedarf über die Schnittstelle *Ausgabe*

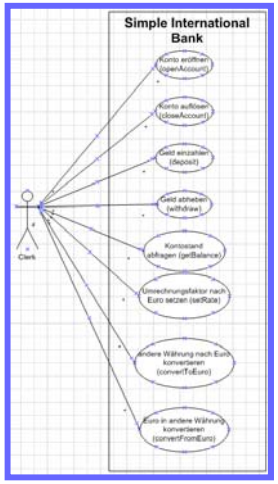
analog dazu:

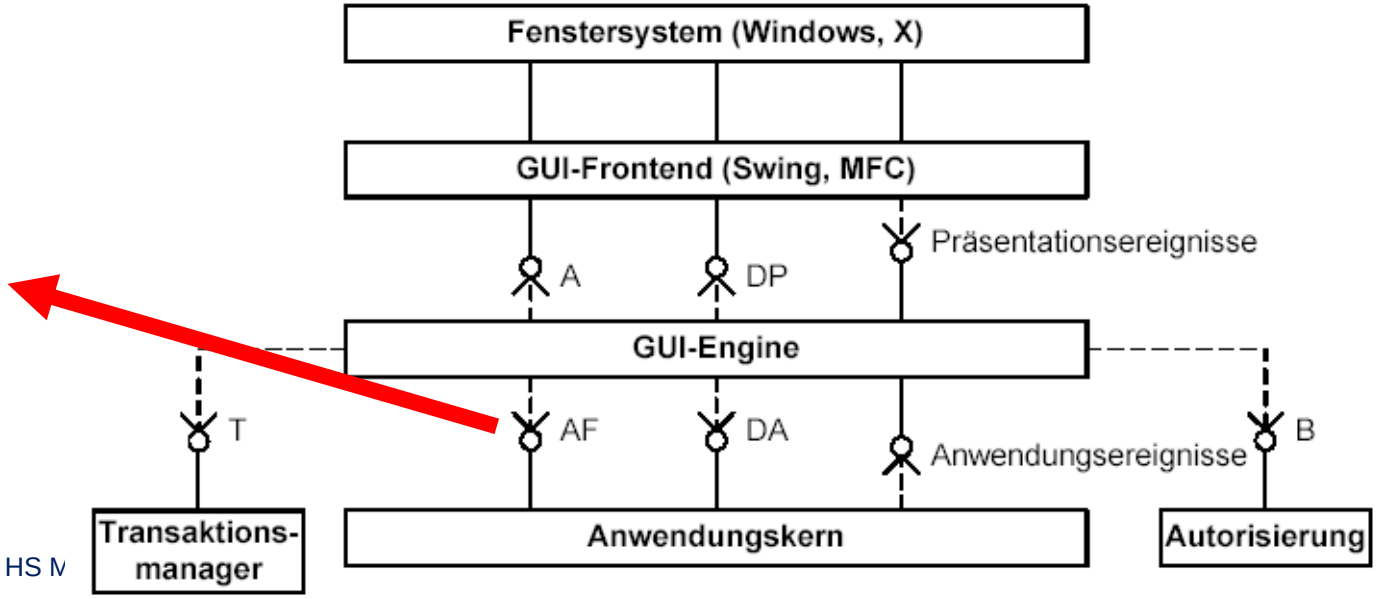
- Der Client kennt den Server, aber ***nicht umgekehrt***
  - Er ruft Anwendungsfall-Methoden über das Interface *Anwendungsfälle/Registrierung* auf, Daten werden als Parameter geliefert
  - Zustandsänderungen des Anwendungskerns werden als Rückgabewerte transportiert oder der Client fragt über das Interface *Zustandsabfragen* den aktualisierten Zustand ab
  - Wenn es Anwendungsereignisse gibt, registriert sich der Client beim Server als Beobachter (*AF/Registrierung*); alle Beobachter werden über *Anwendungsereignisse* über auftretende Ereignisse informiert (unspezifisch, also nicht an bestimmte Beobachter angepasst)

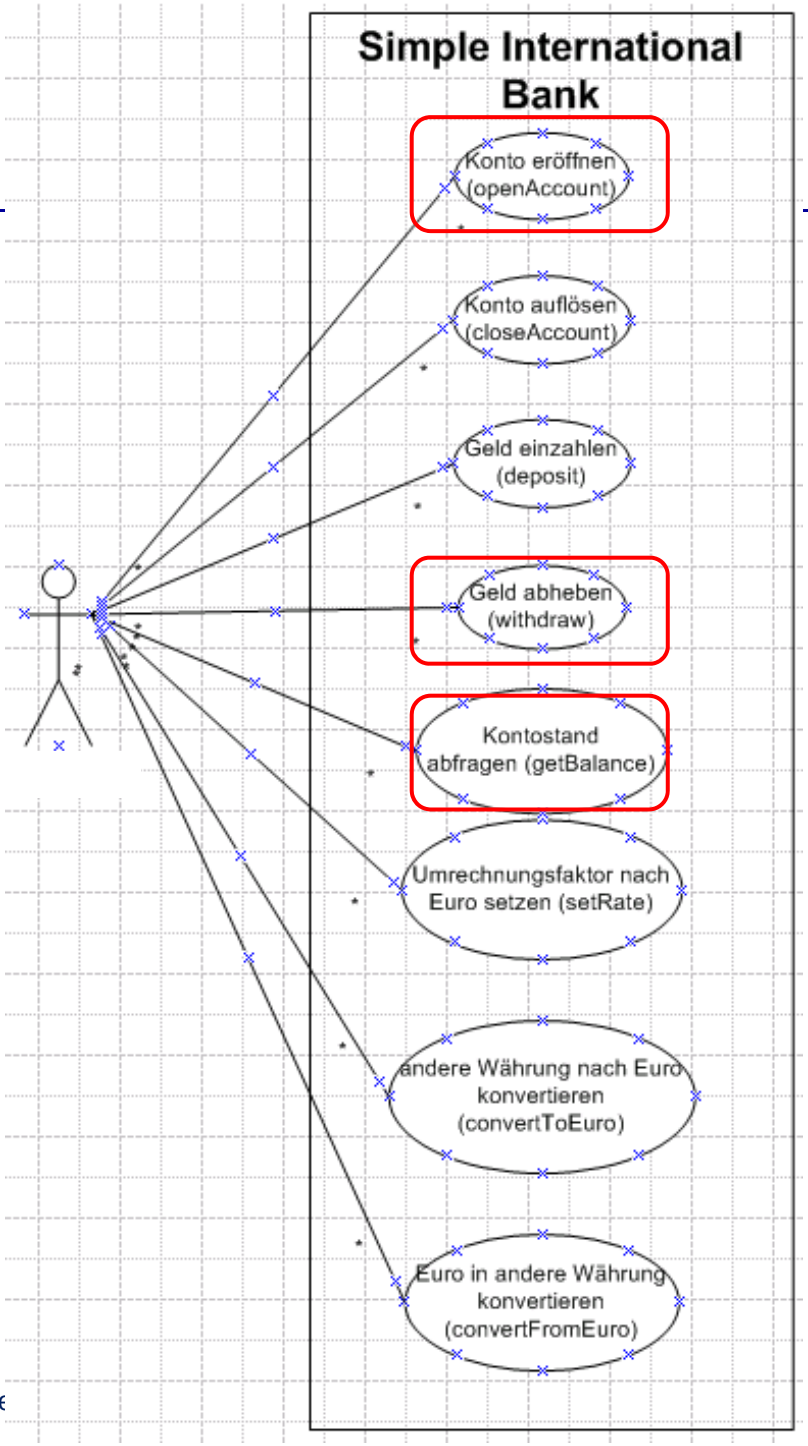
Architektur von Informationssystemen: Implementierung des Anwendungskerns

- Komponenten des Anwendungskerns (= A-Komponenten) implementieren
  - Anwendungsfälle (AF, *Use Cases*) im Sinne der UML (= Methoden einer A-Komponente)

    Beispiel "Bank"
    - → Konto eröffnen
    - → Kontostand abfragen
    - → Geld abheben
  - Registrierung für Ereignisse (→ Listener / Observer / Subscriber)







Use Case-Diagramm (Anwendungsfälle) für eine einfache Bank

Architektur von Informationssystemen: Implementierung des Anwendungskerns

- Komponenten des Anwendungskerns (= A-Komponenten) implementieren
  - Anwendungsfälle (AF, *Use Cases*) im Sinne der UML (= Methoden einer A-Komponente)

    Beispiel "Bank"
    - → Konto eröffnen
    - → ~~Kontostand abfragen~~ → *nicht zustandsverändernd!*
    - → Geld abheben
  - Registrierung für Ereignisse