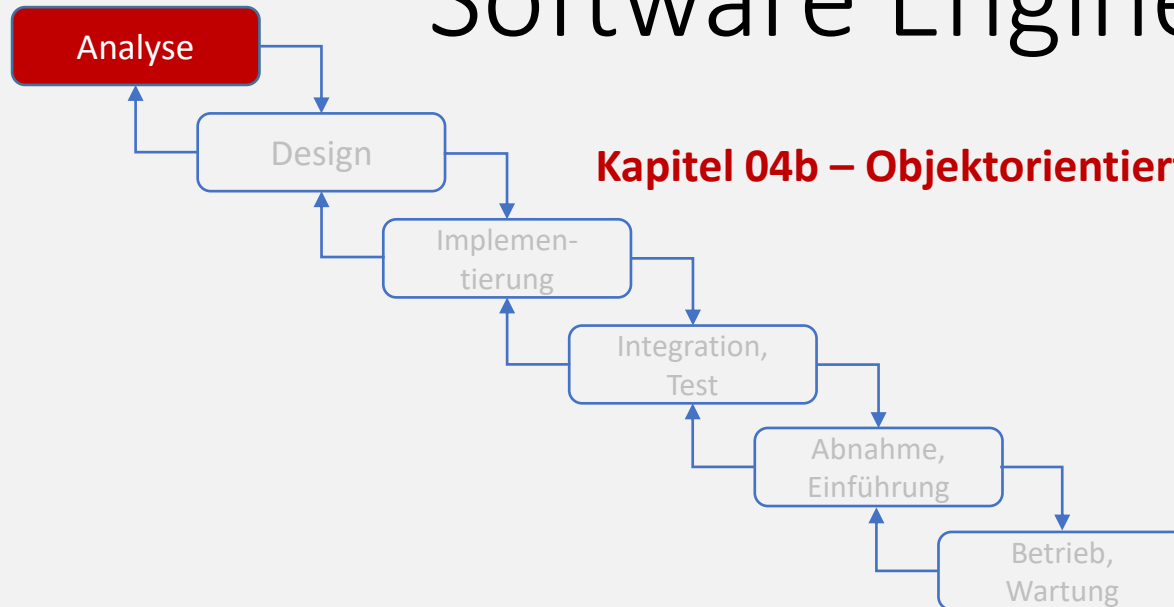




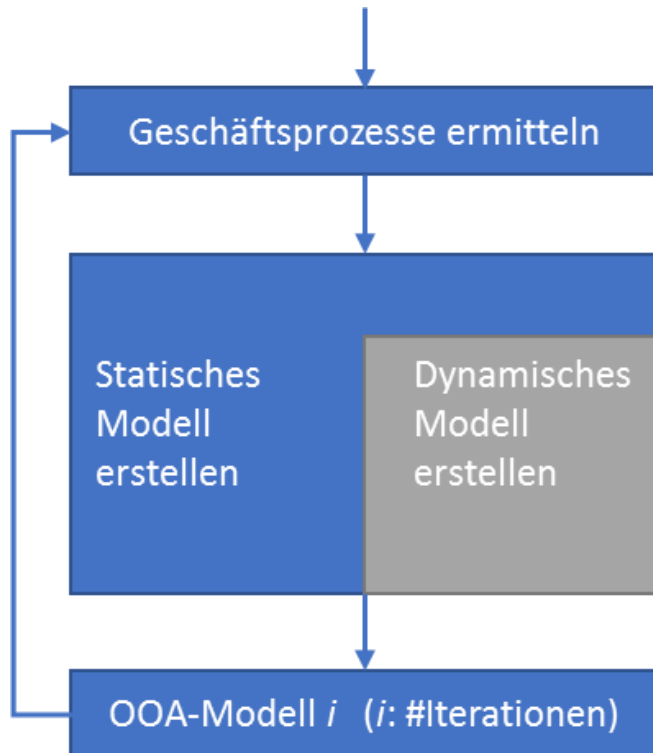
Software Engineering

Kapitel 04b – Objektorientierte Analyse (OOA)



Paketdiagramme

Erinnerung: OOA Makroprozess



Aufgaben des Makroprozesses:

- Analyse im Großen
- 6 Schritte zum statischen Modell
- 4 Schritte zum dynamischen Modell

Analyse im Großen

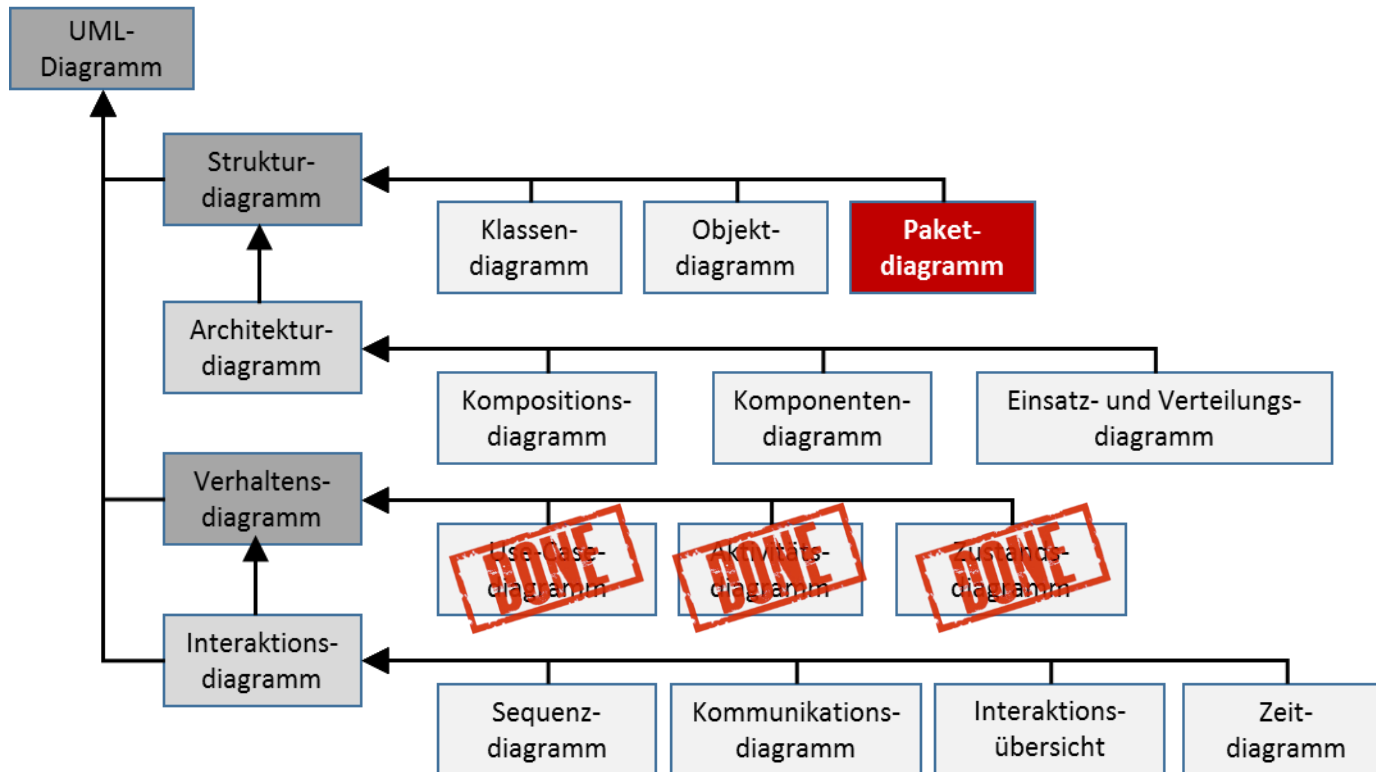
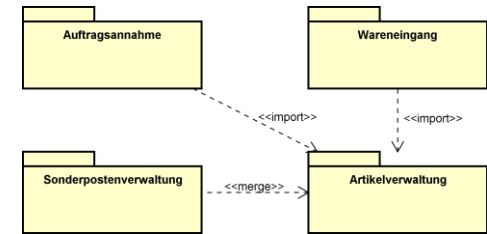
Use Case Modell aufstellen, liefert als Ergebnisse:

- Use-Case-Diagramm
- Use-Case-Beschreibung
- Aktivitätsdiagramme
- Zustandsdiagramm

Pakete bilden, beinhaltet:

- Teilsysteme festlegen (Modellelemente zu Paketen zusammenfassen)
- Bei großen Systemen erfolgt Paketbildung meist zu Anfang
- **Ergebnis: Paketdiagramm**

UML: Paketdiagramm



UML: Paketdiagramm

Motivation

- **Problem:** In großen Softwaresystemen ist es schwer den Überblick zu behalten
 - Es existieren meist hunderte von Use Cases, Aktivitätsdiagramme, Klassen etc.
- **Ziel:** Auf hoher Ebene Struktur in die Systembeschreibung bringen
- **Lösung:** Einteilung des Systems in Pakete (Abstraktion)
- **Hilfsmittel: Paketdiagramm**
 - Beantwortet Frage: *“Wie kann ich mein System so aufteilen und darstellen, dass ich den Überblick behalten?”*

UML: Paketdiagramm

Pakete

Was sind Pakete / wie bündelt man Pakete?

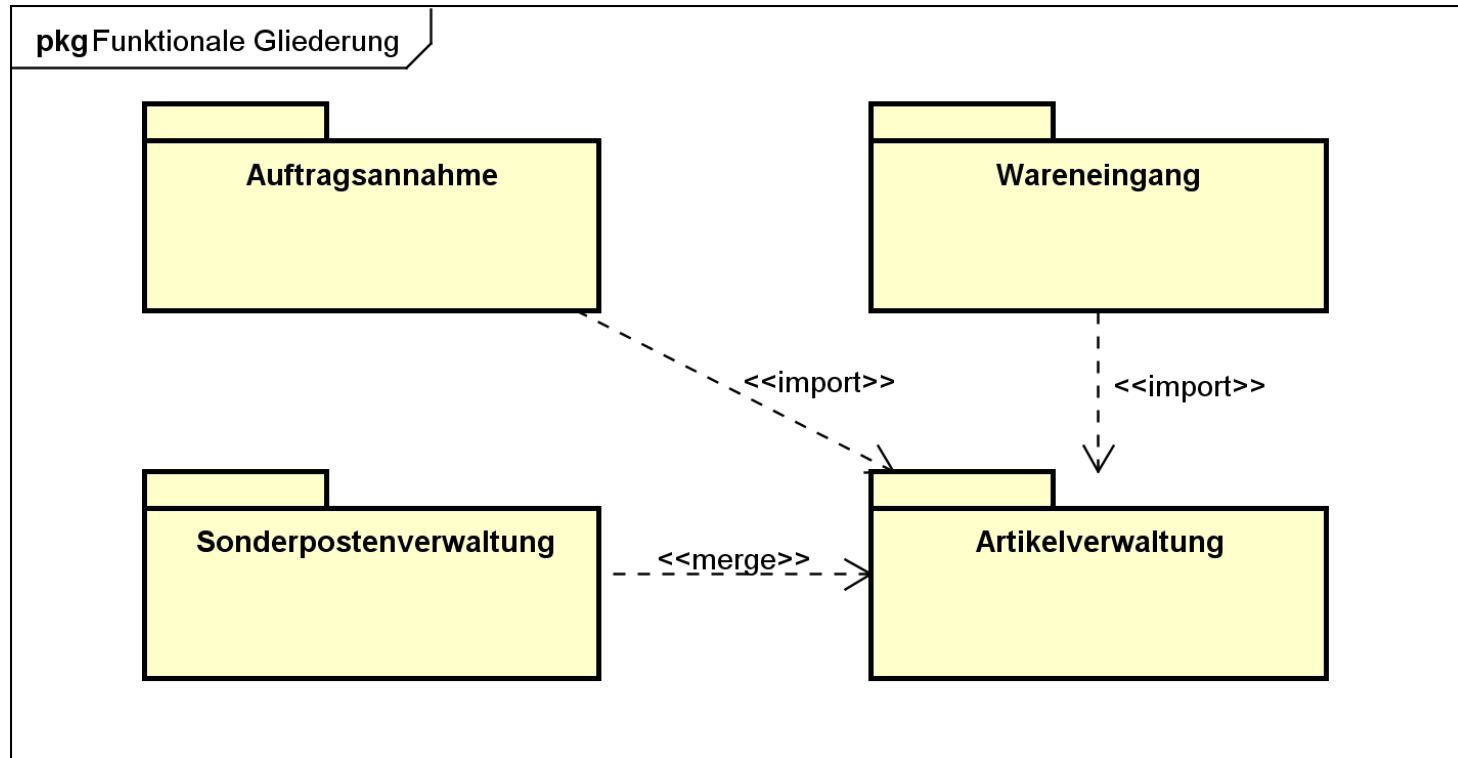
- Paket bündelt Elemente der UML sinnvoll, d.h. nach einem Zusammenhang
- Paket entspricht einem in sich geschlossenen Teilsystem
- Pakete können hierarchisch geschachtelt sein (ein Paket enthält Pakete)

Verwendung von Paketsichten:

- Paketsicht dient der Strukturierung des Systems
- Strukturierung kann nach unterschiedlichen Kriterien erfolgen:
 - Funktionale Gliederung
 - Definition von Schichten
 - ...

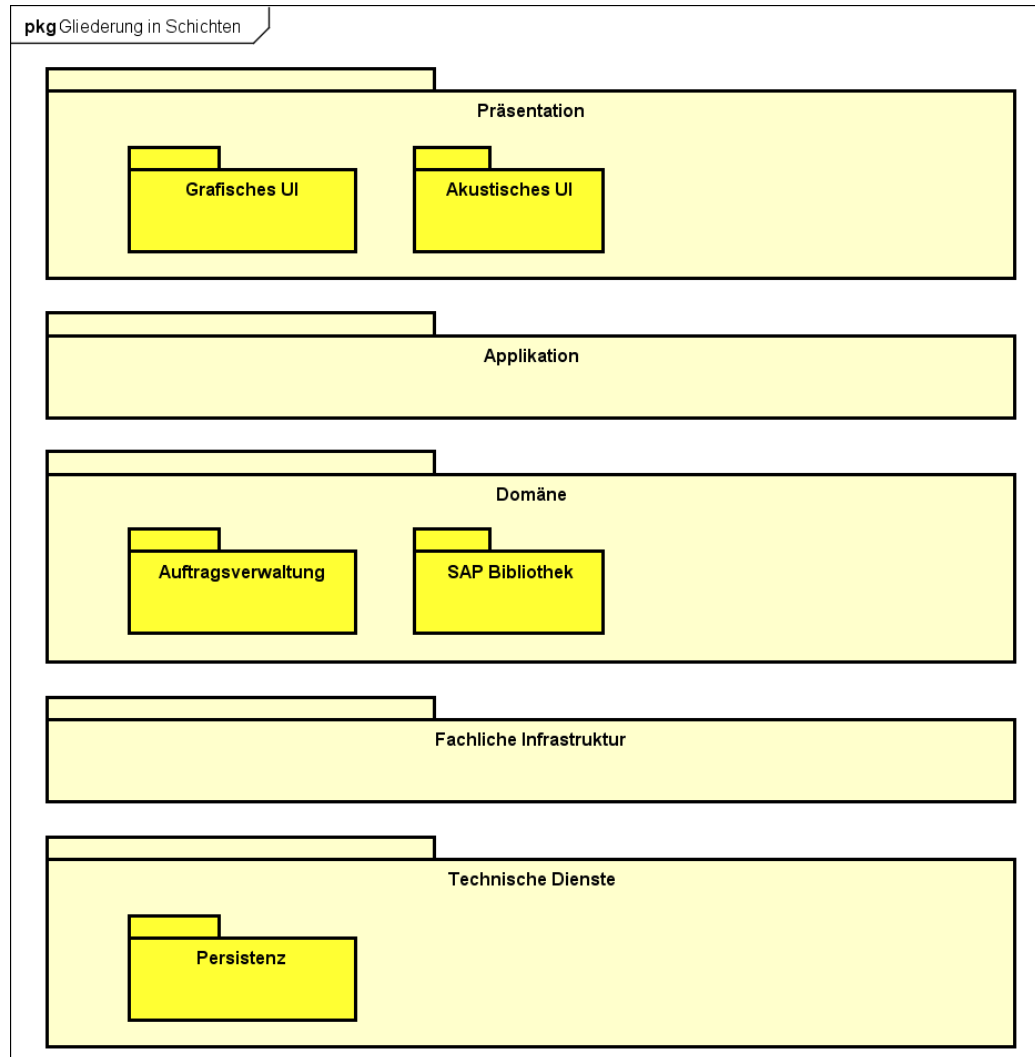
UML: Paketdiagramm

Beispiel: Funktionale Gliederung



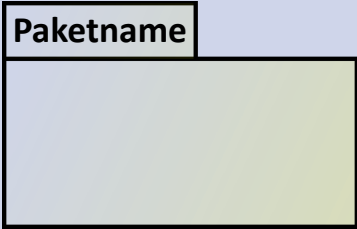
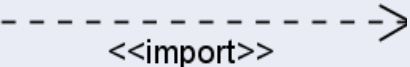
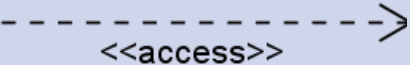
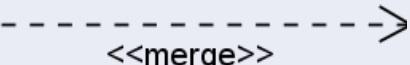
UML: Paketdiagramm

Beispiel: Gliederung in Schichten



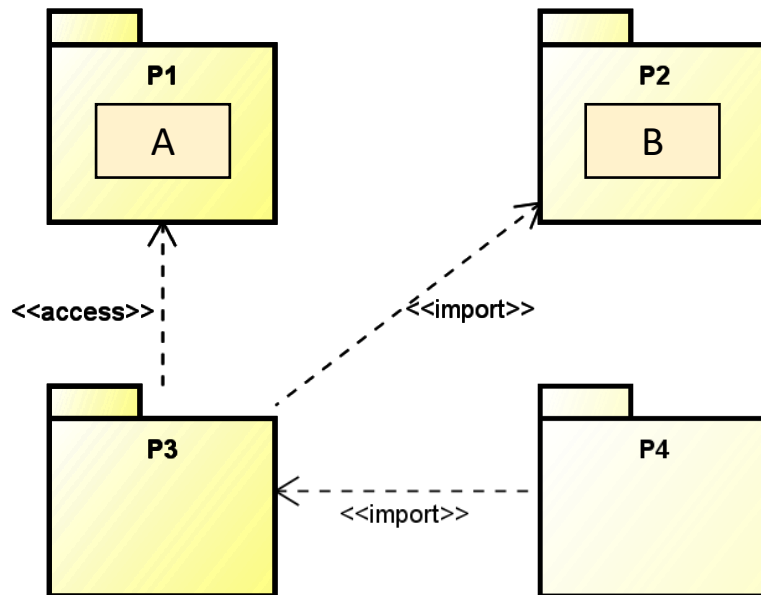
UML: Paketdiagramm

Elemente und Notation

Notation	Bedeutung, Verwendung
	<ul style="list-style-type: none"> - Paket fasst paketierbare Elemente zusammen (oft: Klassen, andere Pakete) - Paket definiert einen Namensraum - Paketname in Lasche oben links (oder oben zentriert in unterem Rechteck) - Innerhalb des unteren Rechtecks können zugehörige Elemente eingetragen werden - Sichtbarkeit der Elemente wie üblich mit "+" (public), "-" (private)
	<ul style="list-style-type: none"> - Namen aller <u>öffentlichen Elemente</u> eines Pakets werden dem importierenden System <u>als öffentlich hinzugefügt</u> (public import) - Abhängigkeit möglich zwischen: 2 Paketen, 2 Klassen, Operation und Klasse, Klasse und Schnittstelle (auch für <<access>> und <<merge>>)
	<ul style="list-style-type: none"> - Namen aller <u>öffentlichen Elemente</u> eines Pakets werden dem importierenden System <u>als privat hinzugefügt</u> (private import)
	<ul style="list-style-type: none"> - Nicht private Inhalte des Zielpakets werden in die Inhalte des Quellpakets verschmolzen

UML: Paketdiagramm

Abstraktes Beispiel (<<import>>, <<access>>)



- P3 importiert A (P1) als **private**
- P3 importiert B (P2) als **public**
- P4 importiert P3

Ergebnis:

- P4 darf auf B zugreifen
- P4 darf auf A nicht zugreifen

(Hinweis: A,B sind Classifier)

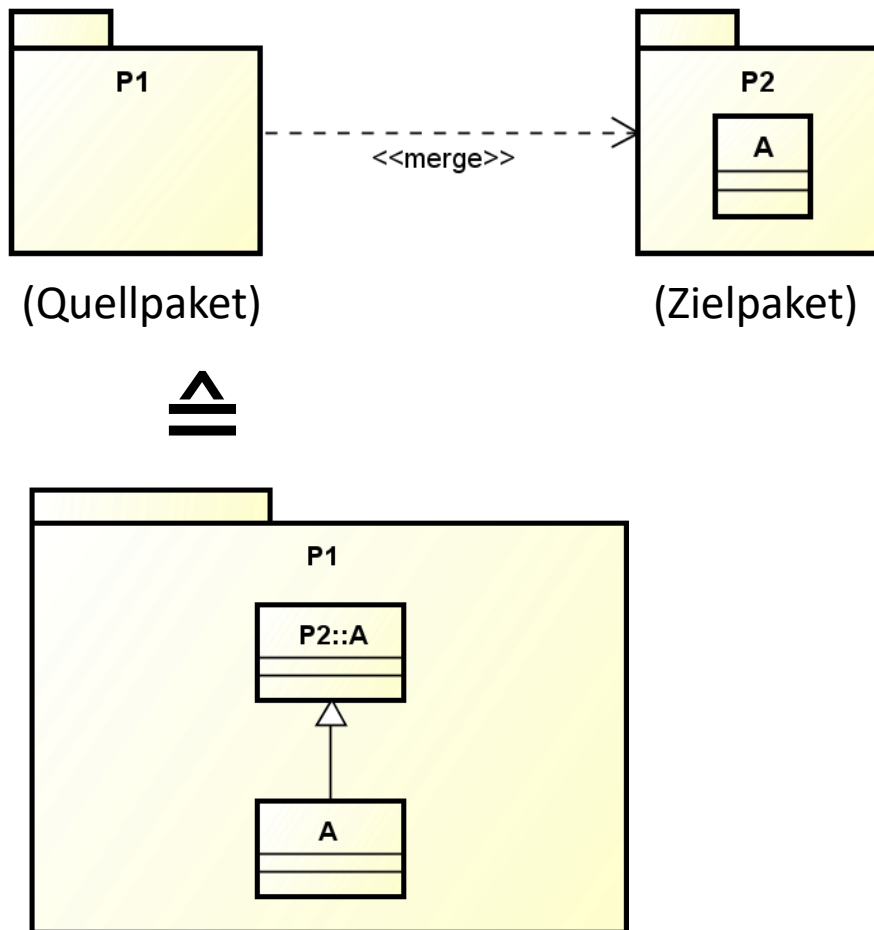
UML: Paketdiagramm (<<import>>, <<access>>)



1. Erstellen Sie ein Paketdiagramm mit den Paketen P1 und P2, die folgende Elemente enthalten:
 - P1 enthält die Klassen +K10, +K11, -K12
 - P2 enthält die Klassen +K20, +K21, -K22, -K23, +K24
 - Ein weiteres Paket P3 soll zu P1 eine <<import>>-Beziehung besitzen und zu P2 eine <<access>>-Beziehung.
 - Das Paket P4 enthält eine <<access>>-Beziehung zu P3.
2. Auf welche Klassen aus P1 und P2 kann ohne qualifizierenden Namen zugegriffen werden?
3. Welche Sichtbarkeiten besitzen diese Klassen in den entsprechenden Paketen P3 und P4?

UML: Paketdiagramm

Abstraktes Beispiel (<<merge>>)

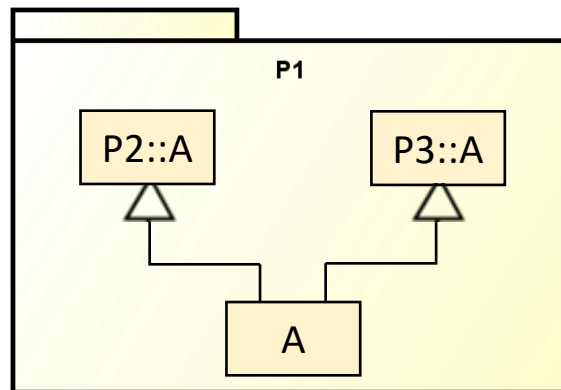
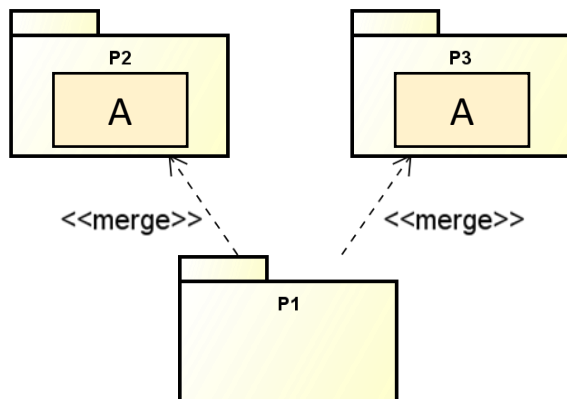
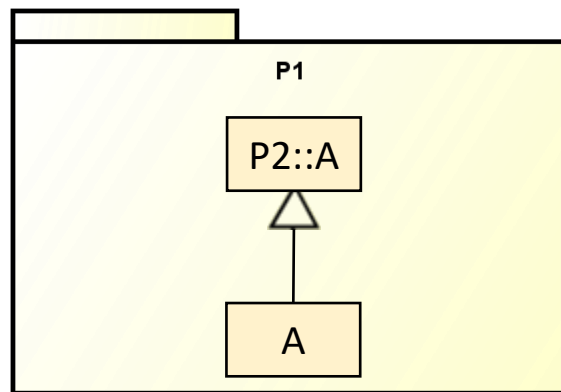
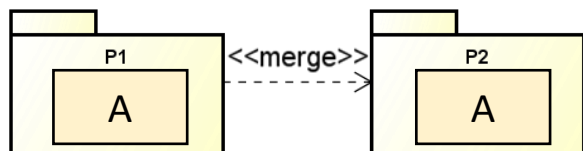


Paket-Merge:

- Anders als beim Import sind gemergte Classifier von P2 direkt im Paket P1 definiert
- (Beim Import sind sie nur sichtbar)
- Beim Merge werden also in P1 neue Elemente angelegt, die Sie dann verändern können



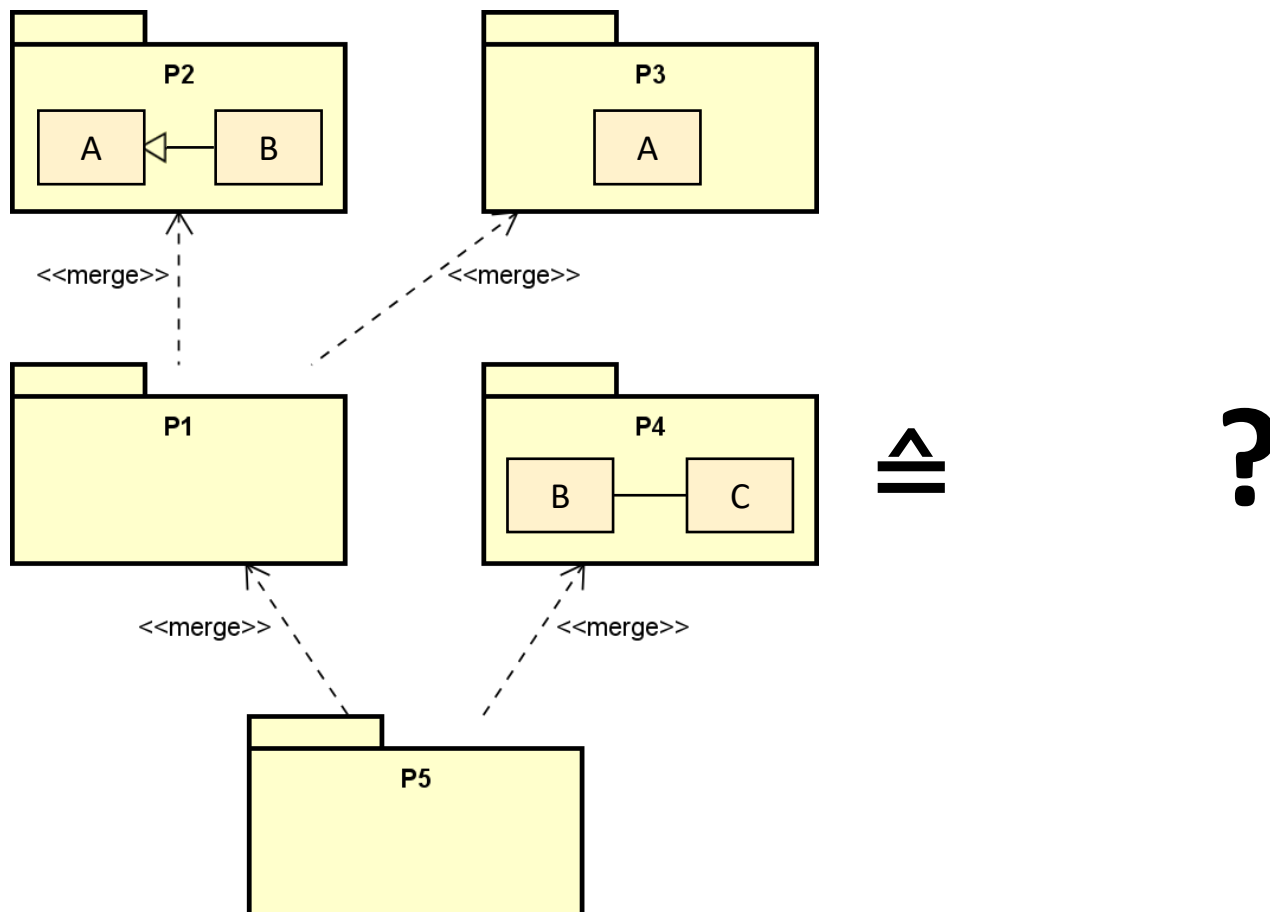
Weitere Beispiele für <<merge>>





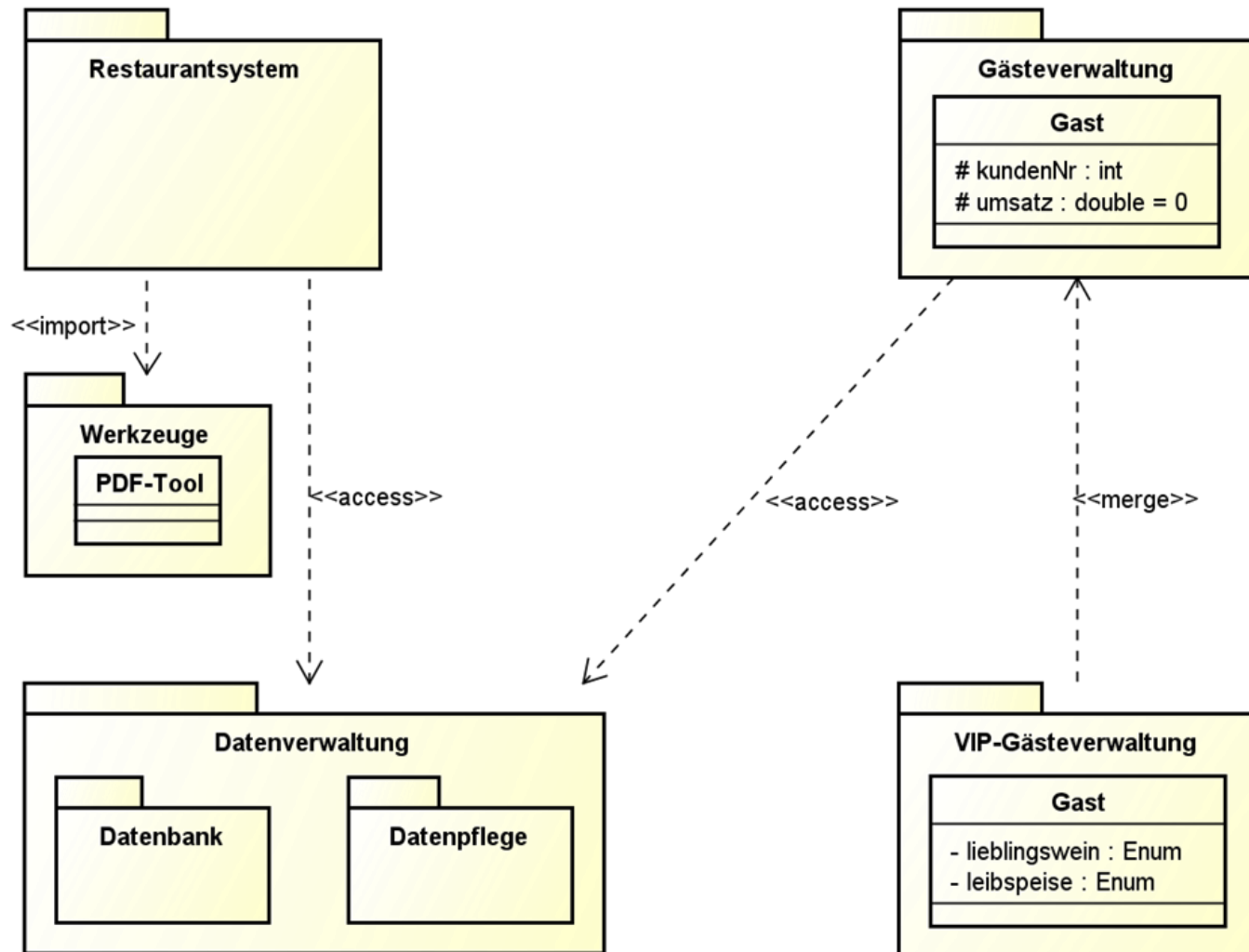
Beispiel: Mehrere <<merge>>-Stufen

Wie sieht die Auflösung der folgenden Merges in P1 und P5 aus?



UML: Paketdiagramm

Beispiel



UML: Paketdiagramm

Vorgehen bei der Erstellung eines Paketdiagramms:

- **Konstruktive Schritte zum Identifizieren von Paketen:**
 - Top-down: Bei großen Projekten noch vor der Erstellung der Use Cases
 - Bottom-up: Bei kleinen oder mittelgroßen Projekten nach der Erstellung der Geschäftsprozesse oder spätestens nach der statischen Modellierung
- **Analytische Schritte:**
 - Bildet das Paket eine abgeschlossene Einheit? (Häufig bilden Klassen mit vielen Beziehungen untereinander ein Paket)
 - Ist der Paketname geeignet?

UML: Paketdiagramm

Fehlerquellen:

- Zu kleine Pakete
- Schlechte oder falsche Bezeichner
 - Pakete müssen eindeutige Namen haben
- Bezeichnung der Beziehung fehlt (`<<import>>` (Standardfall), `<<access>>`, `<<merge>>`)
- Zirkuläre `<<merge>>`-Beziehung
 - Pakete dürfen sich nicht gegenseitig/zyklisch mergen
- Falsche Hierarchie:
 - Klassen dürfen Subklassen, aber keine Pakete beinhalten

Literatur

- *UML 2 glasklar*, Chris Rupp et al., Hanser, 2012