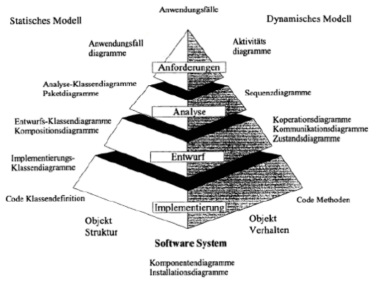
UML Klassendiagramme

- Objektorientierte Analyse
- Objektorientierung
- Klassendiagramme
- Ein Beispiel
- Anwendungsfall zu Fachmodell

Objektorientierte Analyse

Lastenheft (Anforderungen): Anwendungsfalldiagramme, Aktivitätendiagramme



*Pflichtenheft (**Analyse**)*: **Klassendiagramme**, Sequenzdiagramme, Kollaborationsdiagramme

Entwurfsphase: Paketdiagramme, **Klassendiagramme**, Zustandsdiagramme, Sequenzdiagramme

Implementierungsphase: Komponentendiagramme, Verteidigungsdiagramme

---

Die Objektorientierte Analyse - Erstellen eines **ersten Modells** des Systems (Analysemodell)

> Das Analysemodell dient dem Entwickler als Grundlage für den Entwurf

Formalisierung der Anforderungen: Entdecken, Nachverfolgen und Beseitigen von Fehlern und Problemen, Mehrdeutigkeiten, Widersprüchen und Lücken für die Anforderungsspezifikation,...

Bestandteile des Analysemodells (Fachmodell)
Statisches Fachmodell: Systemstruktur mittels **Klassendiagramm**
Dynamisches Fachmodell: Systemverhalten mittels **Sequenzdiagramm** oder **Zustandsdiagramm**

Vorgehen:
1. *Identifizieren* der Fachobjekte, Systemgrenzen und Steuerungsobjekte, z.B. als Typen von Klassen mittels **Klassendiagrammen**
2. *Zuordnen* der Objekte zu den **Anwendungsfällen** (z.B. mittels Sequenzdiagrammen)
3. *Identifizieren* von: **Beziehungen** (Abhängigkeiten, Assoziationen, Vererbung) | **Attribute** | **Methoden**

Objektorientierung

> Darstellung von Objekten der realen Welt als Objekte in der Programmierung

Objekt hat ...
1. Eine **Objektidentität**, die es von anderen Objekten unterscheidet
2. Einen **Kontext**, d.h. ein Objekt kennt andere Objekte, zu denen es in Beziehung steht
3. Einen **Objektzustand**, bestimmt durch die Werte der Attribute eines Objekts und dessen Beziehungen zu anderen Objekten
4. Ein **Objektverhalten**, bestimmt durch eine Reihe von Methoden des Objektes

Der **Zustand** eines Objekts: Umfasst die **Attribute** und ihre **aktuellen Werte** und deckt die **relativen Beziehungen** ab
Das **Verhalten** eines Objekts: Wird durch **Methoden** definiert, **Änderungen oder Abfragen** nur durch Methoden möglich

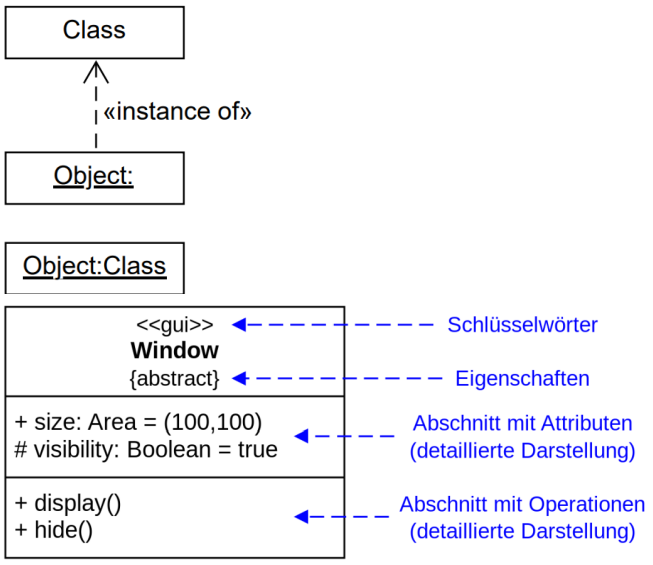
> Die Eigenschaft, die ein Objekt von allen anderen Objekten unterscheidet, wird als Identität bezeichnet.

drei Grundprinzipien der Objektorientierung: **Datenkapselung | Vererbung | Polymorphie**

Klassendiagramme

Klassendiagramme modellieren die statischen strukturellen Beziehungen zwischen den Komponenten eines Systems.

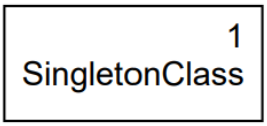
Klassendiagramme sind der Hauptbaustein der objektorientierten Modellierung.



Scopes und Multiplizität

Gültigkeitsbereich (scope):
Pfadnamen können in Klassendiagrammen verwendet werden, um den *Gültigkeitsbereich* (scope) der Klasse zu identifizieren,
z.B. org::openoffice::SpellChecker

Multiplizität von Klassen:



In der oberen rechten Ecke einer Klasse kann die Multiplizität (d.h. Anzahl der Instanzen) einer Klasse sein:
Keine Zahl in der oberen rechten Ecke: *mehr* als eine Instanz einer Klasse möglich

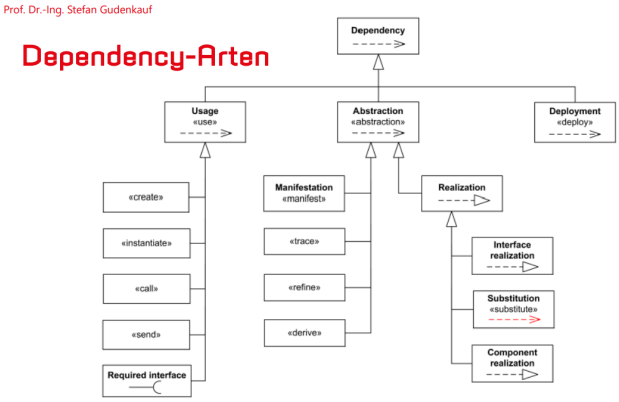
Klassenattribute stellen die Eigenschaften (properties) einer Klasse dar.

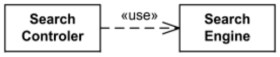
Sichtbarkeit von Attributen

+ öffentlich: Unbeschränkter Zugriff
 # geschützt: Zugriff von Klasse sowie von Unterklassen
− privat: Nur die Klasse selbst kann das Attribut sehen
~ paketsichtbar: Innerhalb des Pakets sichtbar

Arten von Beziehungen

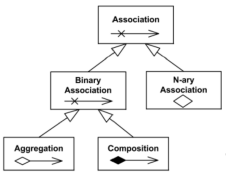
Dependency: Abhängigkeit zwischen zwei Klassen
Assoziation: lies “hat ein”
Aggregation: Spezieller Typ der Assoziation, lies “besitzt ein”
Komposition: Stärkste Beziehung zwischen Klassen, lies “besteht aus”
Generalisierung: Vererbungsbeziehung, lies “ist ein”
Realisierung: Implementierung einer Schnittstelle



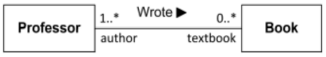


Nutzungsabhängigkeit – Angabe, dass ein Element ein anderes Element oder eine Menge von Elementen für seine Implementierung benötigt.

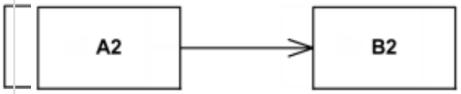
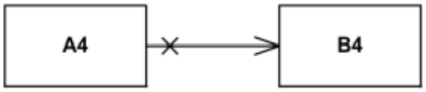
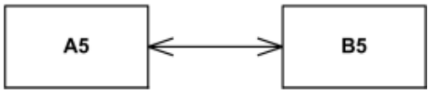
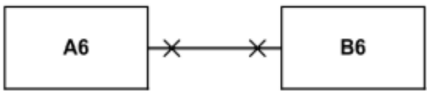
Assoziation

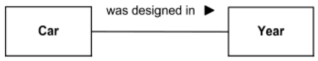


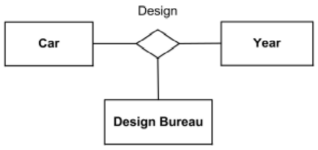
– Angabe, dass *Typinstanzen* miteinander in *Verbindung* stehen oder logisch bzw. physisch miteinander zu einer Aggregation *kombiniert* werden können. z.B: Wrote:



Navigierbarkeit von Assoziationen

Beide Enden der Assoziation haben eine nicht spezifizierteNavigierbarkeit:	A2 hat nicht spezifizierte Navigierbarkeit, während B2 von A2 aus navigierbar ist: 	A3 ist von B3 nicht navigierbar, wenn B3 nicht spezifizierte Navigierbarkeit hat:
A4 ist von B4 aus nicht navigierbar, während B4 von A4 navigierbar ist: 	A5 ist von B5 aus navigierbar und umgekehrt: 	A6 ist von B6 aus nicht navigierbar und umgekehrt 

Stelligkeit von Assoziationen: 



Binäre Assoziationen: verbindet genau **zwei** Typen bzw. Klassen.

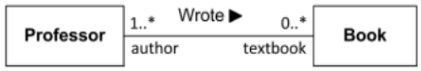
***n*-äre Assoziationen** mit mehr als zwei Assoziationsenden

---

Multiplizität (Kardinalität) - Definition Anzahl maximal Elemente durch inklusives Intervall *nichtnegativer* Ganzzahlen

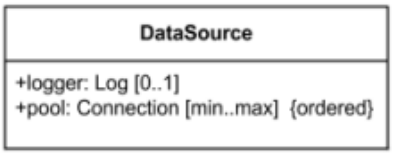
Multiplizität	Option	Bedeutung
0..0	0	Keine Instanzen (leere Menge)
0..1		Keine oder eine Instanz
1..1	1	Genau eine Instanz
0..*	*	Null oder mehr Instanzen
1..*		Mindestens eine Instanz
5..5	5	Genau fünf Instanzen
m..n		Mindestens , aber nicht mehr als Instanzen

Beispiel:



Jeder Professor kann (als author) ein oder mehrere Bücher

schreiben (kann sein lassen): 0..*
Jedes Buch (als textbook) ist von mindestens einem Professor geschrieben: 1..*

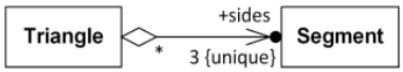


Multiplizität
Ordnung (ordered oder unordered)
Einzigartigkeit der Elemente (unique oder nonunique)
Standardgemäß sind Elemente eines Intervalls unordered und unique

Aggregation - Eine binäre Assoziation, die eine Teile-Ganzes Beziehung darstellt.

Asymmetrisch: nur ein Ende der Assoziation kann eine Aggregation sein
Transitiv: Aggregationsverknüpfungen sollten gerichteten, azyklischen Graph bilden, so dass keine Instanz Teil seiner selbst ist
Unabhängig: Wenn übergeordnete Instanzen gelöscht werden, können die untergeordneten Teile-Instanzen weiterbestehen

(shared) Aggregation

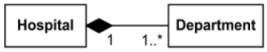


Ein Dreieck hat drei einzigartige Liniensegmente, die als Seiten bezeichnet werden.
Die Liniensegmente sind von Dreieck (Triangle) aus navigierbar.

Die Komposition

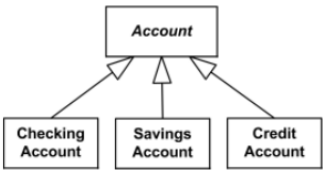
Komposition - Eine binäre Assoziation, die eine Teile-GanzesBeziehung darstellt und es gilt:

1. Ein **Teil** kann zu **einem Zeitpunkt** höchstens in einem **Ganzen** (engl. whole, composite) enthalten sein.



2. Wenn ein ***Ganzes gelöscht wird***, werden ***alle seine Teile*** mit diesem gelöscht.

Die Generalisierung

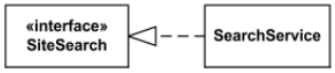


Generalisierung - Eine binäre taxonomische Beziehung zwischen einem allgemeineren Typ

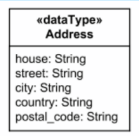
---

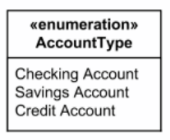
Realisierung

Realisierung odellelementen, von denen eine die Spezifikation und die andere deren Implementierung (Kunde) darstellt.

Spezielle Realisierung zwischen einem Klasse und einer Schnittstelle. 

Operation (engl. operation) -- Verhaltensmerkmal einer Schnittstelle, eines Datentyps oder Klasse

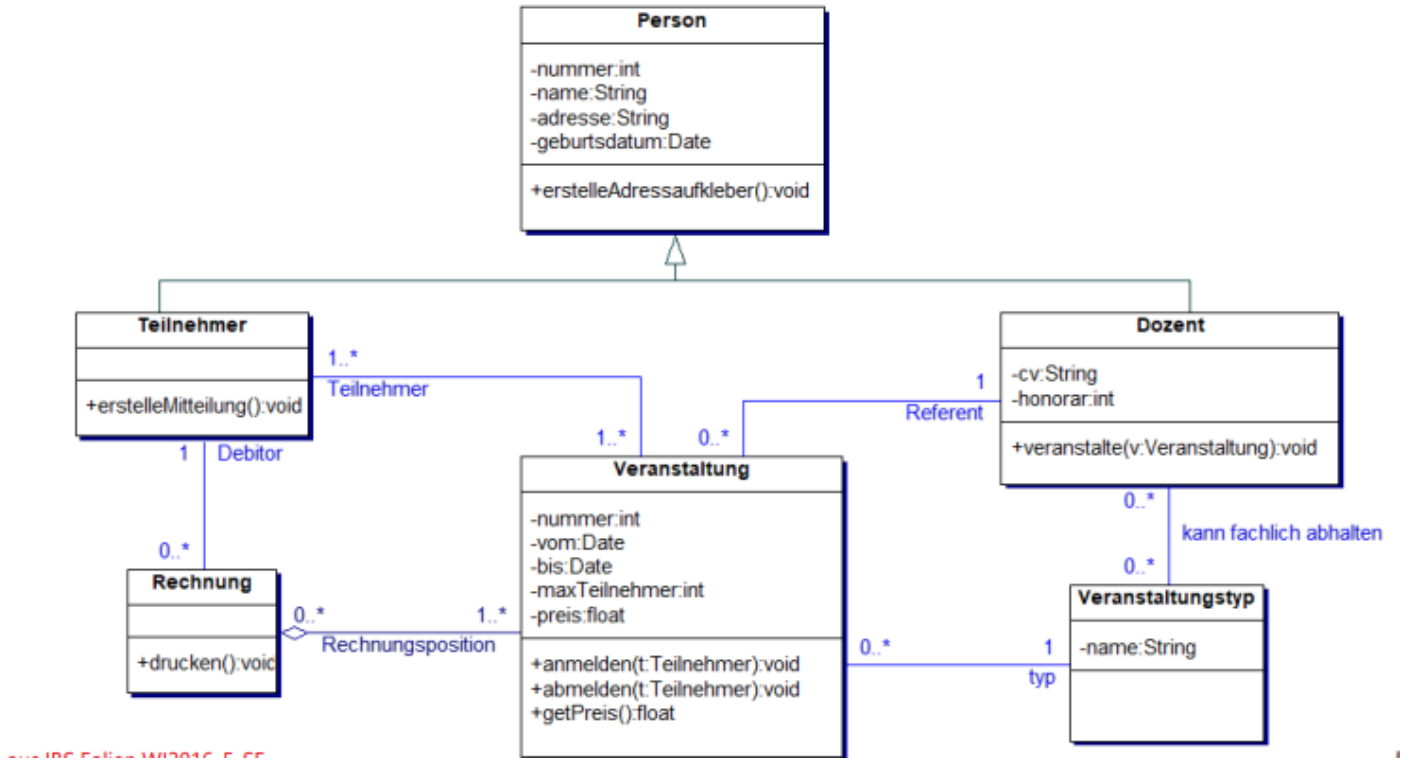
Datentyp -- Typ, dessen Instanzen 



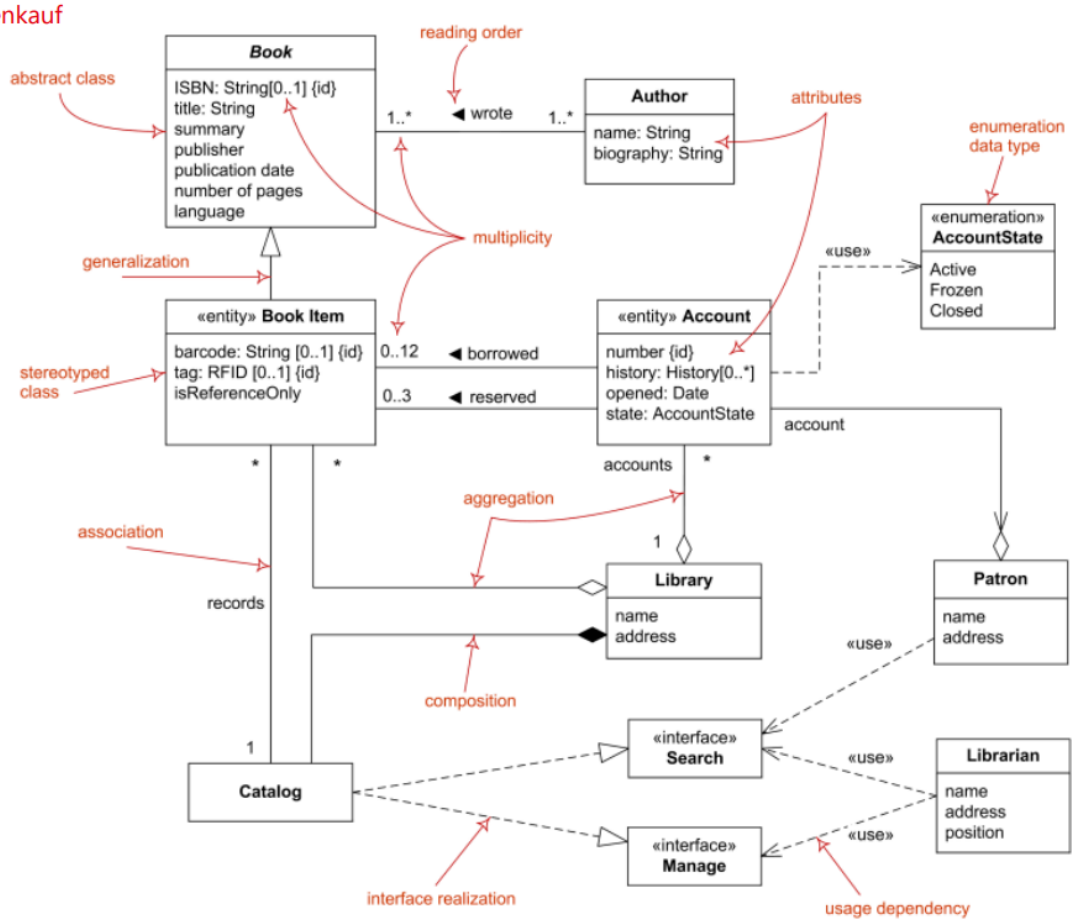
nur durch ihren Wert bestimmt sind

Aufzählung (enumeration) -- Datentyp, dessen Werte benutzerdefinierte Literale sind

Ein Beispiel



Prof. Dr.-Ing. Stefan Gudenkauf



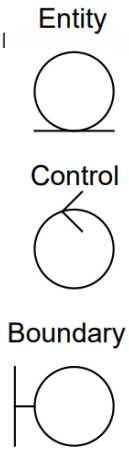
https://www.uml-diagrams.org/class-diagrams-overview.html (2019-08-26)

Anwendungsfall zu Fachmodell

Entity: Objekte, die persistentes Wissen (d.h. **Daten**) repräsentieren; häufig aus dem Domänenmodell hergeleitet.
Boundary: Objekte, die eine **Schnittstelle** zu Systemakteuren bilden (z.B. einem Benutzer oder einem externen System).
Control: Objekte, die **Prozesswissen** repräsentieren (später ggf. als separate Klassen oder einfache Funktionen implementiert).

Kommunikationsregeln im ECB-Muster
1. ***Akteure*** können nur mit **Grenzobjekten** (boundary objects) sprechen.
2. **Systemgrenzobjekte** können nur mit **Kontrollobjekten** (control objects) und **Akteuren** sprechen.
3. **Entitätsobjekte** (entity objects) können nur mit **Kontrollobjekten** kommunizieren.
4. **Kontrollobjekte** können mit **Grenzobjekten**, **Entitätsobjekten** und **Kontrollobjekten** kommunizieren, nicht mit **Akteuren**

E



jects identifizieren:

1. **Ausdrücke**, die geklärt werden müssen, damit der Anwendungsfall verständlicher wird
2. Wiederkehrende **Substantive** in Anwendungsfällen
3. **Begriffe** aus der **Fachdomäne**
4. **Aktivitäten** in der realen Welt, die befolgt werden müssen
5. **Vorhandene Datenquellen** oder Ergebnisse

Boundary Objects identifizieren
1. Jeder **Akteur** interagiert mit mindestens einem Boundary Object in jedem Anwendungsfall.
2. sammeln **Benutzerinfos** und übersetzen sie in **Strukturen**, die von Control / Entity Objects verarbeitet werden.

3. **Grobe Abstraktion** der Systemgrenze treffen, ohne Details der visuellen Darstellung
4. **Begriffe** verwenden, die der Anwender versteht
5. **Geräte** der Interaktion identifizieren, Benutzungsschnittstellen beschreiben

Kommunikation zu →	Actor	Boundary	Control	Entity
Actor		✓		
Boundary	✓		✓	
Control		✓	✓	✓
Entity			✓	✓

Control Objects identifizieren - Koordination von Entity und Boundary Objects
1. **Empfange** Informationen von den Boundary Objects.
2. **Verteile** diese in geeigneter Form an die Entity Objects.

Beispiel:

1. Der **Außenbeamte** aktiviert die **,,MeldeNotfall"-Funktion** auf dem Terminal. Ereignis ablauf

2. FRIEND antwortet durch Bereitstellung eines **Formulars**. Das **Formular** enthält Menü über Art des **Notfalls**, Ortsbeschreibung und Felder für eine Beschreibung des **Notfalls**, für Betriebsmittelanforderungen und für den Eintrag von gefährlichen Gütern.

3. **Außenbeamter** füllt das **Formular** aus durch Kurzspezifizierung des **Notfalltyps** und der übrigen Beschreibungen. **Außenbeamter** kann auch mögliche Aktionen auf die **Notfallsituation** beschreiben und spezielle Betriebsmittel erbitten. Sobald das **Formular** ausgefüllt ist, übermittelt es der **Außenbeamte** durch Drücken der,,Sende Meldung"-Taste dem **Dienstleiter**.

4. Der **Dienstleiter** überprüft die Information, die vom **Außenbeamten** übermittelt wurde, und erzeugt einen **Vorfall** in der Datenbank, indem er den Eröffne **Vorfall**-Anwendungsfall aufruft. Alle Informationen, die im **Formular** des **Außenbeamten** enthalten sind, befinden sich automatisch im **Vorfall**. Der **Dienstleiter** wählt eine Antwort aus durch Zuweisen von Betriebsmitteln an das Ereignis (mit dem WeiseBetriebs mittelZu-Anwendungsfall) und bestätigt die **NotfallMeldung** durch Übersenden einer Nachricht an den **Außenbeamten**. Die **Empfangsbestätigung** zeigt dem **Außenbeamten**, dass die **Notfall Meldung** erhalten, ein **Vorfall** erzeugt und Betriebsmittel an den **Vorfall** zugeteilt wurden. In der **Empfangsbestätigung** sind die Betriebsmittel (z.B. Löschzug) und deren voraussichtliches Eintreffen angegeben.

5. Der **Außenbeamte** erhält die Bestätigung und die ausgewählte Antwort.

Entity Objects

Dienstleiter	Polizeibeamter, der den Vorfall leitet. Ein Dienstleiter eröffnet, dokumentiert und schließt Vorfälle ab als Antwort auf Notfall Meldungen und andere Kommunikation mit dem Außenbeamten. Dienstleiter werden durch Dienstnummern identifiziert.

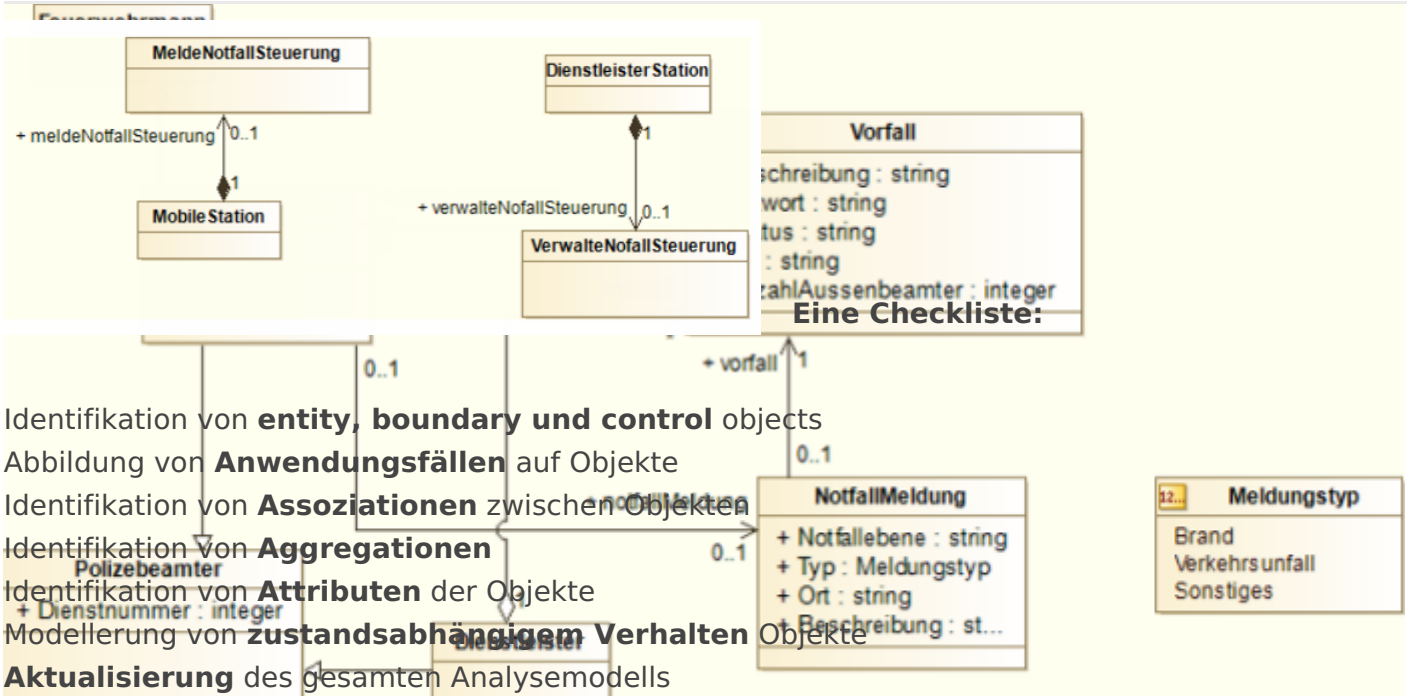
NotfallMeldung	Anfangsbericht über einen Vorfall von einem Außenbeamten an einen Dienstleiter. Eine NotfallMeldung erfordert das Erstellen eines Vor fall durch den Dienstleiter . Eine NotfallMeldung besteht aus einer Notfallebene, einem Typ (Brand, Verkehrsunfall, Sonstiges), einem Ort und einer Beschreibung.
Außenbeamter	Polizist oder Feuerwehrmann im Einsatz. Ein Außenbeamter kann zu jeder Zeit nur einem Vorfall zugewiesen sein. Außenbeamte werden durch die Dienstnummer identifiziert.
Vorfall	Ereignis, das die Aufmerksamkeit eines Außenbeamten erfordert. Ein Vorfall kann von einem Außenbeamten dem System mitgeteilt wer den. Ein Vorfall besteht aus einer Beschreibung, einer Antwort, einem Status (offen, beendet, ...), einer Örtlichkeit und einer Anzahl von zugewiesenen Außenbeamten.

Boundary Objects

Empfangsbestätigung	Bestätigung, um dem Außenbeamten den Empfang der Meldung beim Dienstleiter anzuzeigen.
DienstleiterStation	Vom Dienstleiter benutzter Rechner.
MeldeNotfall Taste	Vom Außenbeamten benutzte Taste, um den MeldeNotfall-Anwen dungsfall zu initiieren.
NotfallMeldeformular	Formular, das für die Eingabe von MeldeNotfall benutzt wird. Dieses Formular wird dem Außenbeamten auf der Mobilen Station angeboten, wenn die Melde Notfall"-Funktion gewählt wurde. Das NotfallMeldeformular enthält Felder, um alle Attribute einer NotfallMeldung zu spezifizieren und einen Knopf (oder etwas Ähnliches), um das ausgefüllt Formular abzuschicken.
MobileStation	Vom Außenbeamten benutzter mobiler Rechner.
Vorfall Formular	Formular, das für die Erstellung eines Vorfalls benutzt wird. Die ses Formular wird dem Dienstleiter auf der DienstleiterStation angeboten, wenn die NotfallMeldung empfangen wurde. Dienstleiter benutzt dieses Formular auch, um Betriebsmittel zuzuweisen und den Bericht des Außenbeamten zu bestätigen.

Control Objects

MeldeNotfall Steuerung	Verwaltet die Benachrichtigungsfunktion MeldeNotfall auf der MobilenStation. Dieses Objekt wird kreiert, wenn der Außenbeamte den,,Melde Notfall"-Knopt auswahlt. Es erzeugt dann ein Notfall Meldeformular und bietet es dem Außenbeamten ar.. Nach dem Abschi cken des Formulars sammelt dieses Objekt die Informationen des Formulars, erzeugt eine NotfallMeldung und leitet sie zum Dienst leiter. Das Steuerungsobjekt wartet dann auf eine Empfangsbestäti gung von der DienstleiterStation. Sobald die Empfangsbestätigung eingetroffen ist, erzeugt das MeldeNotfall Steuerung-Objekt eine Empfangsbestatigung und zeigt sie dem Außenbeamten an.
VerwalteNotfall Steuerung	Verwaltet die Benachrichtigungsfunktion MeldeNotfall auf DienstleiterStation. Dieses Objekt wird erzeugt, wenn eine Not fallMeldung empfangen wird. Es erzeugt dann ein Vorfall Formular und bietet es dem Dienstleiter an. Sobald Dienstleiter Vorfall erzeugt, Betriebsmittel bereitgestele und Empfangs bestätigung eingegeben hat, leitet VerwalteNotfallSteuerung die Empfangsbestätigung an MobileStation weiter



Eine Checkliste:

Identifikation von **entity, boundary und control** objects
Abbildung von **Anwendungsfällen** auf Objekte
Identifikation von **Assoziationen** zwischen Objekten
Identifikation von **Aggregationen**
Identifikation von **Attributen** der Objekte
Modellierung von **zustandsabhängigem Verhalten** Objekte
Aktualisierung des gesamten Analysemodells