# Formale Modellierung
# Vorlesung vom 07.05.12: Beyond JML

Till Mossakowski & Christoph Lüth

Universität Bremen

Sommersemester 2012



## Heute im Programm

- Grenzen der JML
- Nach JML: UML (und darüber hinaus)
- Zwei Fallbeispiele



## Fallbeispiel 1: Bankautomat

**Informelle Spezifikation**

Nach dem Einlegen der Karte (cashcard o.ä.) kann der Benutzer auswählen zwischen dem Abheben von Bargeld, und der Anzeige des Kontostands.
In beiden Fällen liest der Automat von der Karte die Kontonummer des Benutzers. Beim Abheben gibt der Benutzer zuerst die gewünschte Summe und danach zur Authentifzierung sein PIN ein. Der Automat liest von der Karte die verschlüsselte PIN, und prüft ob die eingegebene mit der verschlüsselten übereinstimmt. Ist dieser der Fall, und ist das Konto gedeckt, wird das Bargeld zum Abheben bereitgestellt; danach wird die Karte zurückgegeben. Ist das nicht der Fall, wird eine entsprechende Fehlermeldung ausgegeben, und die Karte zurückgegeben. In beiden Fällen geht der Automat danach wieder in den Anfangszustand.



## Kritik JML

- Implementationsnah
- Struktur der Spezifikation = Struktur der Implementierung
  - Hier: Events = Klassen?
- Mangelnde Abstraktion
- Nächster Schritt: UML

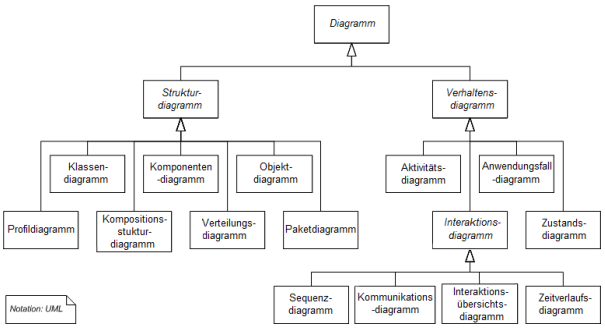


## UML als formale Spezifikationssprache

| Diagrammtyp | Modellierte Aspekte | Formal |
|---|---|---|
| Klassendiagramm | Statische Systemstruktur | Ja |
| Paketdiagramm | Pakete, Namensräume | Nein |
| Objektdiagramm | Zustand von Objekten | (Ja) |
| Kompositionsstrukturdiagramm | Kollaborationen | Nein |
| Komponentendiagramm | Dynamische Systemstruktur | (Nein) |
| Verteilungsdiagramm | Implementierungsaspekte | Nein |
| Use-Case-Diagramm | Ablauf en gros | Nein |
| Aktivitätsdiagramm | Ablauf en detail | Nein |
| Zustandsdiagramm | Zustandsübergänge | Ja |
| Sequenzdiagramm | Kommunikation | Ja |
| Kommunikationsdiagramm | Struktur der Kommunikation | (Ja) |
| Zeitverlaufsdiagramm | Echtzeitaspekte | (Ja) |



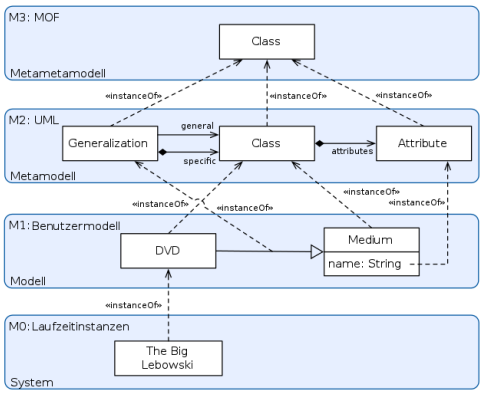
## Diagramme in UML 2.3




Quelle: Wikipedia



## Semantik der UML: Metamodellierung




Quelle: Wikipedia



## Kritik UML

- "OO built-in"
- Adäquat für eingebettete Systeme, CPS, . . . ?

## Fallbeispiel 2: omniRobot

**Informelle Spezifikation**

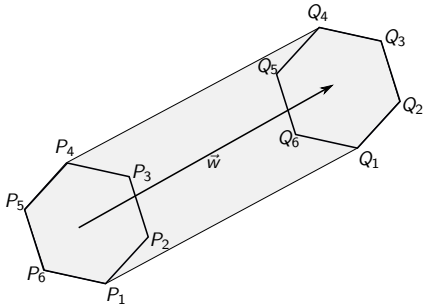
Ein omnidirektionaler, autonomer Roboter soll gegen Kollisionen mit statischen Hindernissen gesichert werden.
Dazu wird zu jedem Zeitpunkt die mit der momentanen Geschwindigkeit $\vec{w}$ beim Bremsen bis zum Stillstand überstrichene Fläche $A$ berechnet, die dann mit einer berührungslos wirkenden Schutzeinrichtung überwacht wird.

Quelle: Kuka AG



## Modellierung

- Roboter als konvexes Polygon $\mathbf{K} = \langle \vec{K}_1, \ldots, \vec{K}_n \rangle$, momentane Pos. $\mathbf{P}$
- Momentane Geschwindigkeit: Vektor $\vec{v} = (v, \phi)$
- Bremsweg $S_1$, Anhalteweg $S$, als Vektor: $\vec{w} = (S, \phi)$
- Zu berechnen: Anhaltefläche $A$






## Fazit Fallbeispiel 2

- Semantische Modellierung der realen Welt
- Nicht inhärent objektorientiert
- Modellierung in UML möglich
- Natürlicher: direkte Modellierung
- Fokus: Beweise, Manipulation von Formeln



## Zusammenfassung

- JML sehr nahe an der Implementation
- UML abstrahiert von Implementation, aber nur bedingt von Anwendungsdomäne
- Nächste VL (Montag): Fallbeispiel 2 in UML und Z