UML *Model-Checking*

Patrick Vogt (http://patrickvogt.net),
Carl-Philip Wenz (http://www.cpwenz.de)

Informatik (M.Sc.)
Hochschule RheinMain (http://hs-rm.de)

23. Mai 2012

Übersicht

Wer sind wir? Was machen wir?

Master-Projekt: UML Model-Checking unter Wiederverwendung bestehender und erprobter Model-Checker

- Julian Horn, Lars-Erik Kimmel, Patrick Vogt und Carl-Philip Wenz
- WS 2011/12
- Hochschule RheinMain in Wiesbaden
- bei Prof. Igler
- mit reger Anteilnahme von Firmen (u.a. DB-Systel, Microsoft Research, parcs-IT Consulting)

Motivation: UML Model-Checking

Model-Checking im Entwicklungskontext

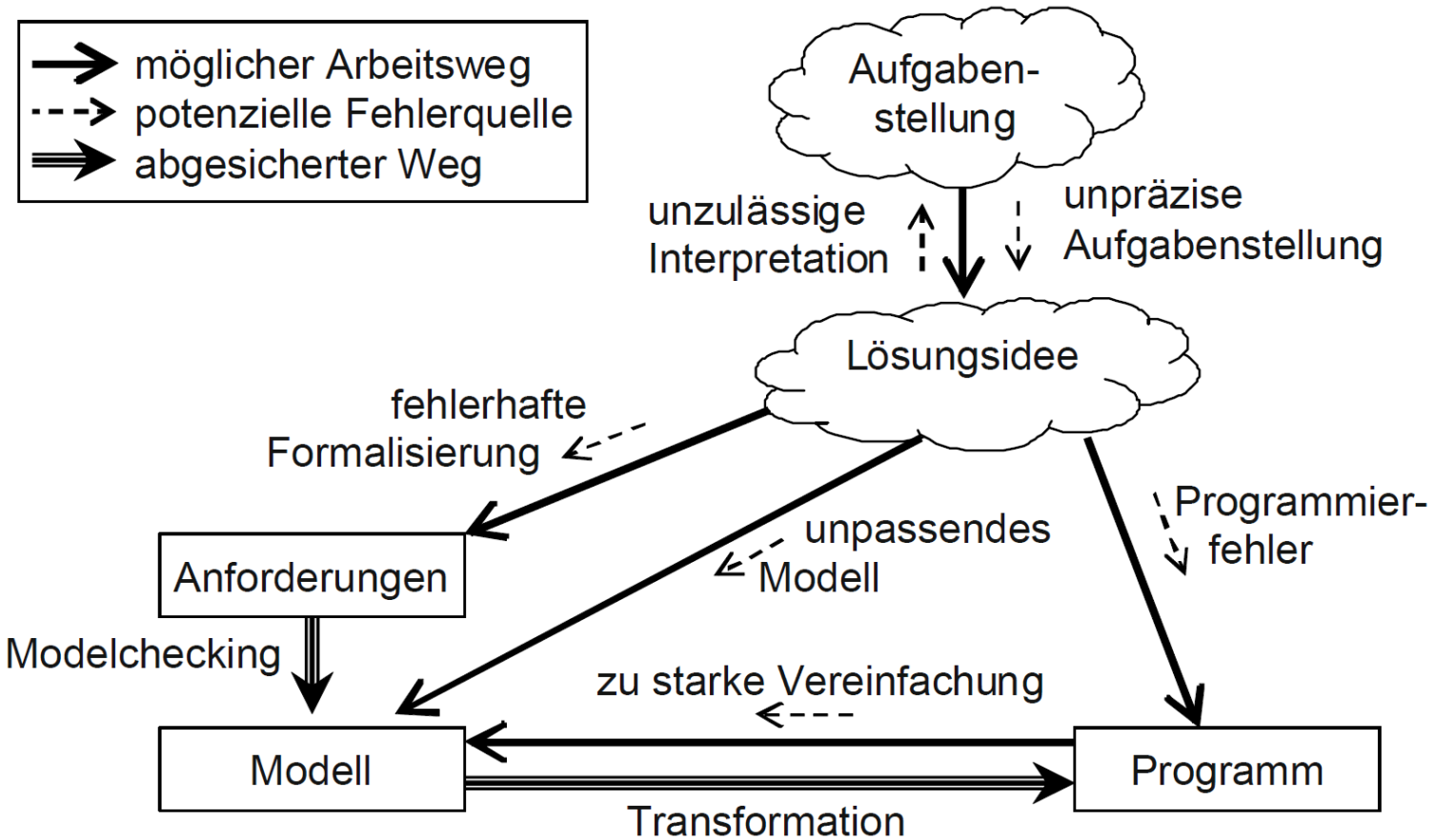


Abbildung: Fehlerquellen der Programmentwicklung (aus [Kle09])

Beispiel für UML *Model-Checking*

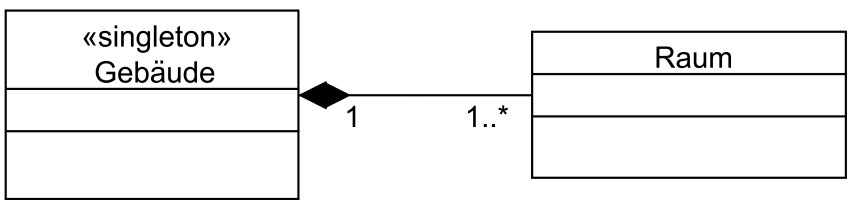


Abbildung: Beispiel-Klassenmodell

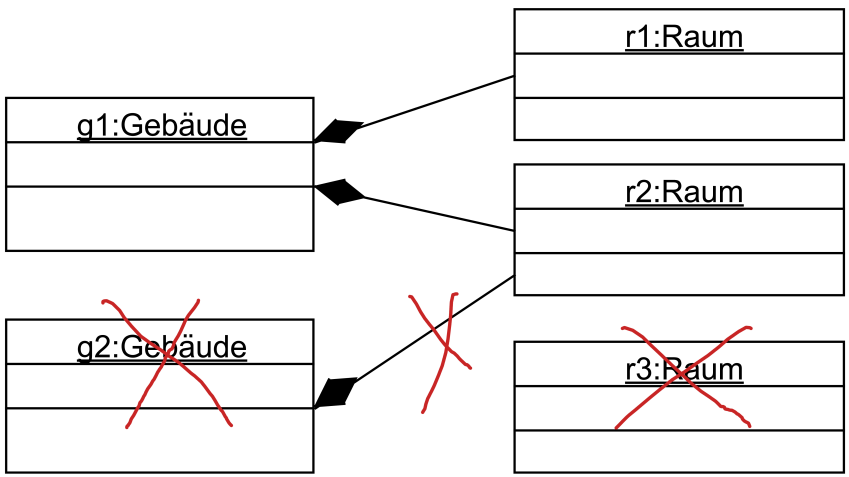


Abbildung: Beispiel-Instanzanalyse

Was ist ein Modell und Model-Checking?

Beispiele für Modelle:

- Zustandsautomaten (für Abläufe/Prozesse)
- UML Klassendiagramme (für Datenstruktur eines Softwaresystems)

Model-Checking: Modell aufstellen, dann Anforderungen spezifizieren und validieren, durch:

$\Rightarrow$ Finden von Instanzen des Modells
$\Rightarrow$ Finden von Gegenbeispielen zu aufgestellten Behauptungen

Beispielwerkzeuge für *Model-Checking*

Alloy Analyzer: für „Datenstrukturen“ als Relationen spezifiziert

SPIN Model Checker: für (parallele und konkurrierende) Programmabläufe

Wo werden solche Werkzeuge eingesetzt?

Model-Checking wird im klassischen SW-Entwicklungsprozess größtenteils ignoriert [BA08, Som10].
. . . aber . . .

- im Automotive-Bereich
- in SW-Systemen für Flugzeugsteuerungen

UML *Model-Checking*: Was gibt es schon?

USE: UML Specification Environment

+ überprüft Objektdiagramme gegen Klassendiagramme
- keine automatische Instanziteration
- keine Option für Verifikation von Behauptungen

UML2Alloy

- transformiert UML Klassendiagramme und OCL nach Alloy
+ zeigt Instanzen als UML-Objektdiagramme
- Beta-Status und nicht lauffähig auf x64-Systemen

Ansatz: Formula

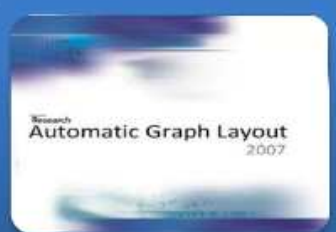
Technologien

Entwicklungsumgebung

- Visual Studio 2010 Ultimate

Model Checking System

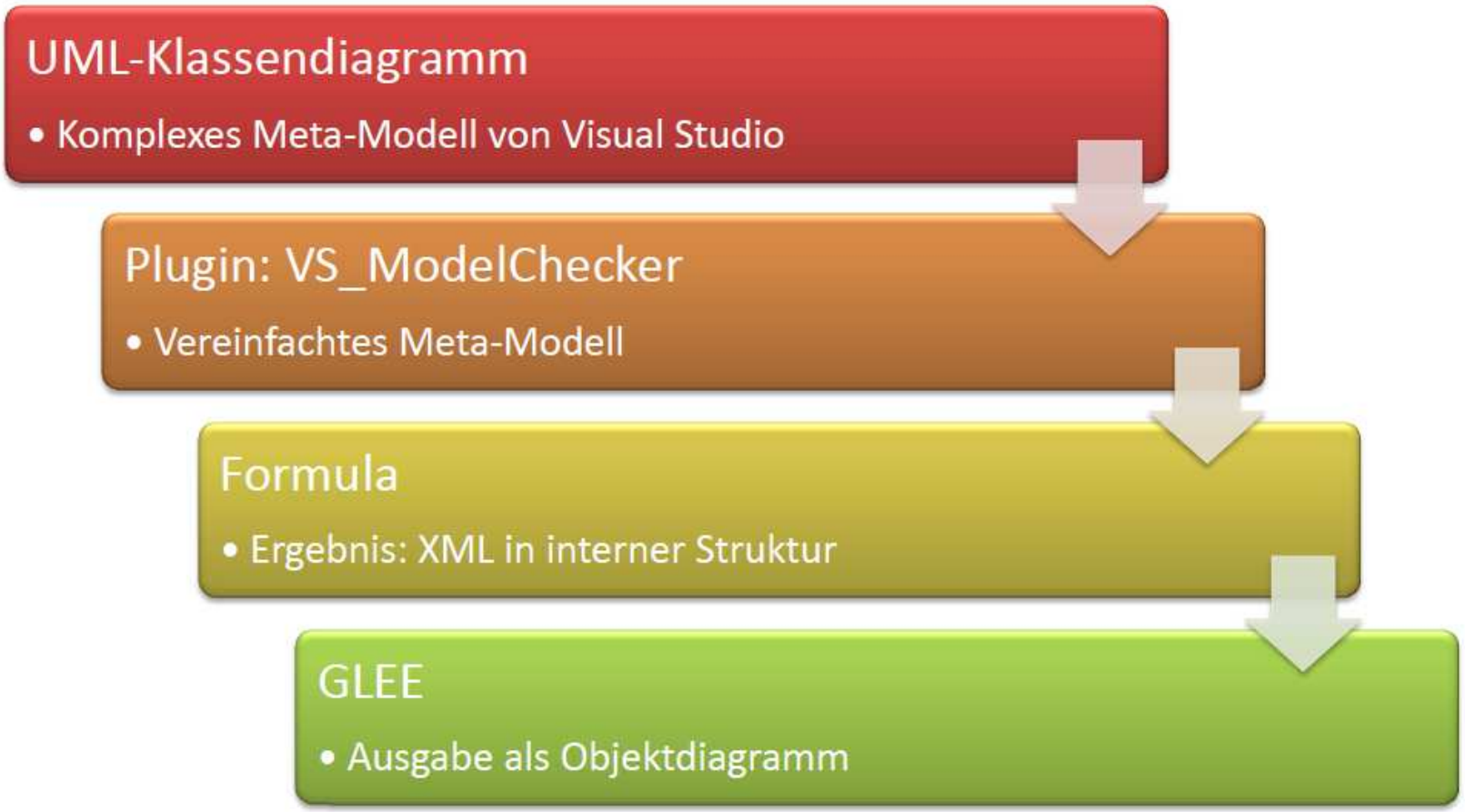
- Microsoft Research Formula



Visualisierung

- GLEE / AGL

Klassendiagramm → Objektdiagramm



Transformation

FORMULA
Modeling Foundations.

Klassendiagramm

- Person
  - Name
- Firma
- Person <-> Firma

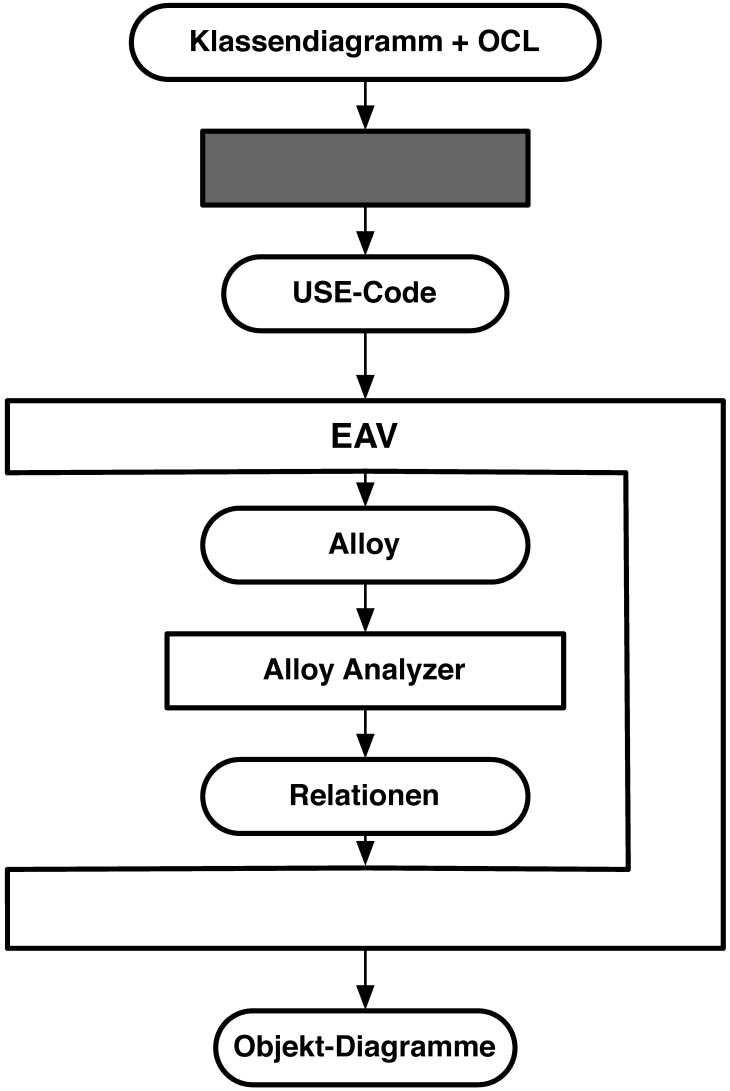
Constraint Logic Programming

- **Person(id:Integer)**
- PropName(Person,String)
  - *Query: 1 Person = 1 PropName*
- **Firma(id:Integer)**
- AssoPersonFirma(Person,Firma)
  - *Query: 1 zu 1*

Fähigkeiten

Ansatz: Alloy/KodKod

EAV - Erste Analyse Versicherung:



Transformation in Prädikatenlogik

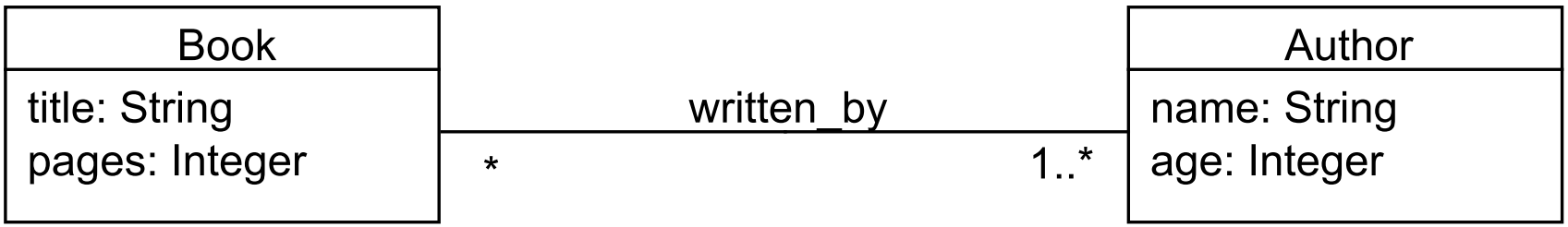


Abbildung: Beispiel-Klassenmodell

- Typzusicherung für Attribute:
  $$\forall x, y.\ \mathit{title}(x, y) \rightarrow \mathit{Book}(x) \wedge \mathit{String}(y)$$
- Multiplizität von Attributen:
  $$\forall x.\ \mathit{Book}(x) \rightarrow (1 \leq \#\{y \mid \mathit{title}(x, y)\} \leq 1)$$

Transformation in Prädikatenlogik

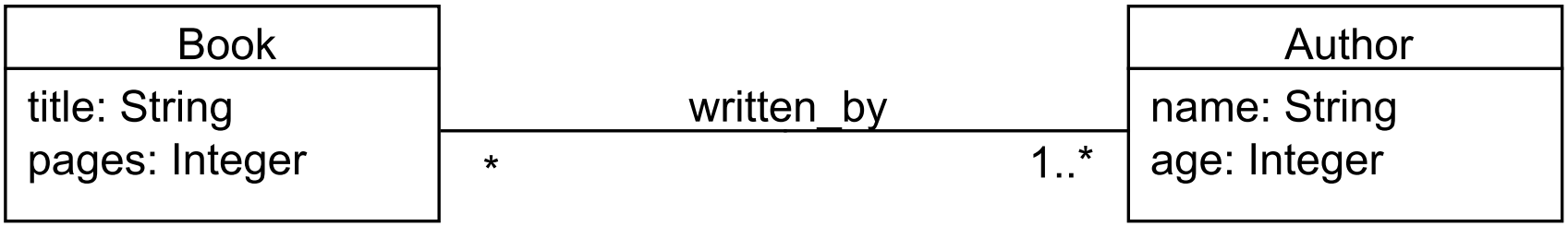


Abbildung: Beispiel-Klassenmodell

- Typzusicherung für Assoziationen:
  $\forall x, y.\ \mathit{written\_by}(x, y) \rightarrow \mathit{Book}(x) \wedge \mathit{Author}(y)$
- Multiplizität von Assoziationen:
  $\forall x.\ \mathit{Book}(x) \rightarrow (1 \leq \#\{y \mid \mathit{written\_by}(x, y)\})$

Transformation in Relationale Logik

Was geht:

- Datenstruktur eines Klassendiagramms
- atomare Operationen/Methoden (OCL Vor- und Nachbedingungen)
- Teilmenge von OCL-Operationen

Was geht (noch) nicht:

- Unterstützung von Bag's und Aggregatsfunktionen
- Unterstützung von temporaler Logik
- Symmetrien filtern $\Rightarrow$ Ausblick
- nicht-atomare Operationen $\Rightarrow$ Ausblick
- rekursive OCL-Operationen $\Rightarrow$ Ausblick
- alle OCL-Operationen unterstüzten

Fazit und Ausblick

UML Model-Checking möglich!

- Prototypen erstellt → vollständige Anwendungen
  - Unterstützung der wichtigsten OCL-Ausdrücke
  - Packaging
  - Lizenz-Situation klären
  - ...
- Ansätze evaluieren
  - Komplexität
  - Effizienz der Trafos
  - ...
- Anwenden!!!
  - Beispiele aus der Praxis bearbeiten
  - Integration in bestehende SW-Entwicklungs-Prozesse
  - ...

Literatur


Ben-Ari, Mordechai:
Principles of the Spin Model Checker.
Springer, 2008. –
ISBN 978–1–84628–769–5


Kleuker, Stephan:
Formale Modelle der Softwareentwicklung.
Vieweg+Teubner, 2009


Sommerville, Ian:
Software Engineering.
9th.
Addison Wesley, 2010. –
ISBN 978–0137035151


Renz, Burkhardt ; Asmussen, Nils:
Kurze Einführung in Alloy.

http://homepages.thm.de/~hg11260/mat/alloy-intro.pdf, Abruf: 22. Oktober 2011

Renz, Burkhardt:
Wie baut man zuverlässige Software? Model Checking - Das Konzept und ein Beispiel.
http://www.mni.fh-giessen.de/thm_repository/reposforsch
Abruf: 10. Mai 2012