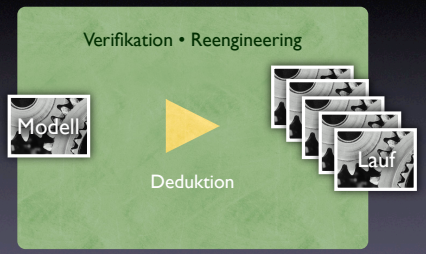
Model Mining

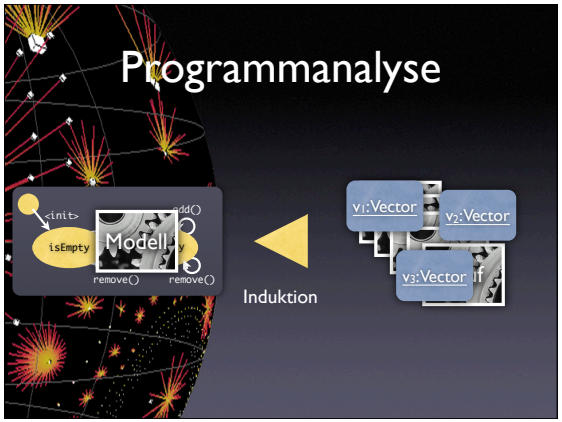
Stammvorlesung Softwaretechnik
Andreas Zeller

Programmanalyse



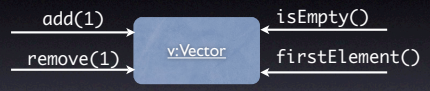
Programmanalyse

Programmanalyse



Model Mining

Objekt-Zustände

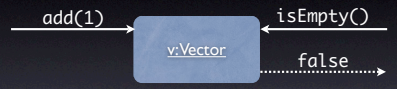


Mutatoren
ändern Zustand

Inspektoren
geben Zustand zurück

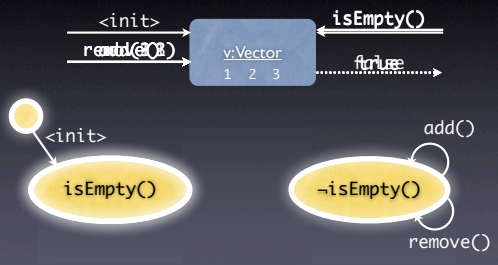
Unterscheidung per statischer Analyse

Modelle bilden

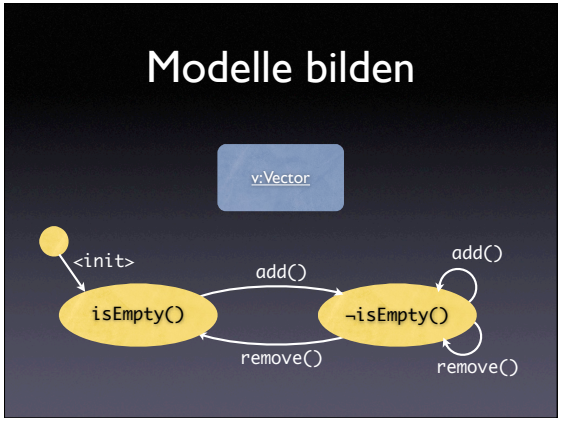


- Nach jedem Mutator-Aufruf bestimmen wir den Objekt-Zustand – mit Inspektoren
- Zustände bilden endlichen Automaten

Modelle bilden

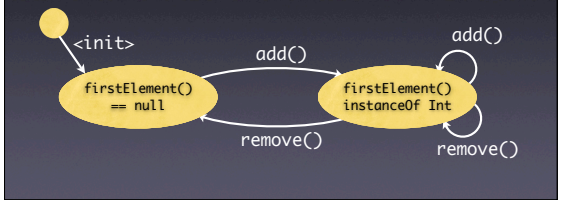


Modelle bilden

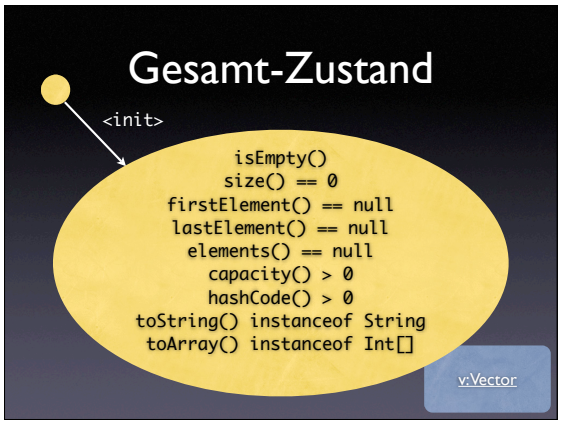


Äquivalenzklassen

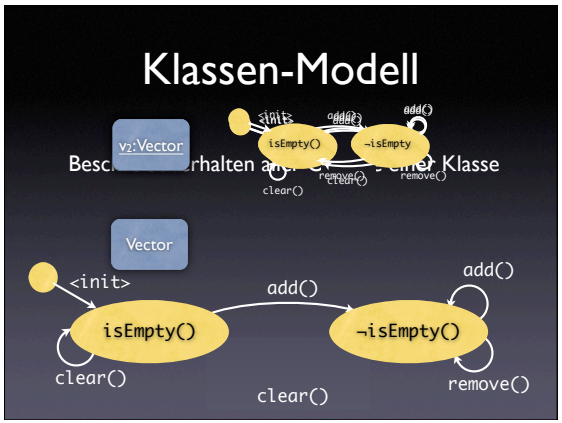
Inspektor-Typ	boolesch	numerisch	objekt
Zustände	true | false	< 0 | = 0 | > 0	null | Klasse



Gesamt-Zustand



Klassen-Modell



Formaler Aufbau

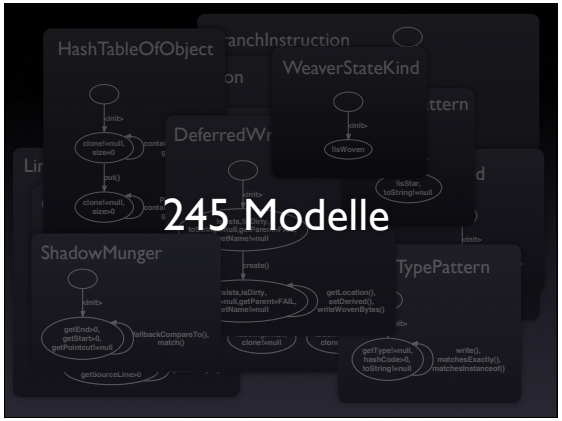
Konkreter Zustand $v \in V$ mit $v = (x_1, x_2, \ldots, x_n)$
x_i – Rückgabewert eines Inspektors

Trace $t = [(v_1, m_1, v_1'), (v_2, m_2, v_2'), \ldots]$
mit $v_i \in V$ und m_i – Name eines Mutators

Zustandsabstraktion $abs: V \to S$

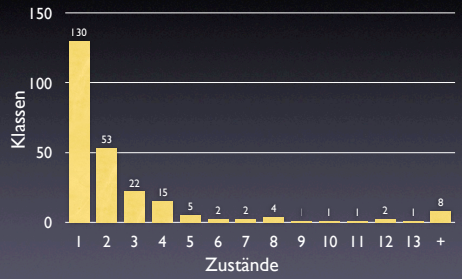
Modell mit Übergängen $s \xrightarrow{m} s'$ und Zuständen $s, s' \in S$

Übergangsbedingung $s \xrightarrow{m} s'$ mit $s, s' \in S$ gdw
$\exists (v, m, v') \in t \cdot abs(v) = s \wedge abs(v') = s'$

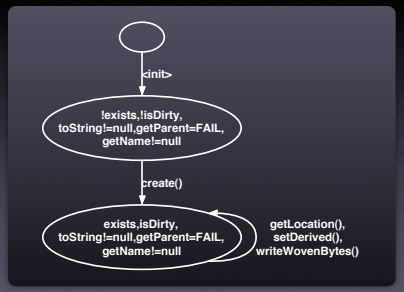


245 Modelle

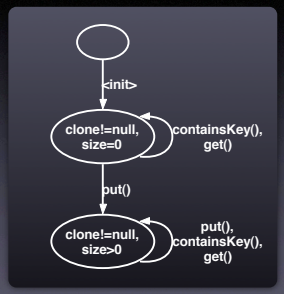
Modellgröße



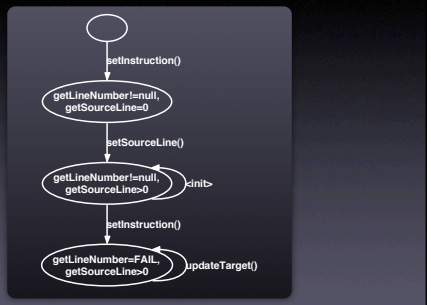
DeferredWriteFile



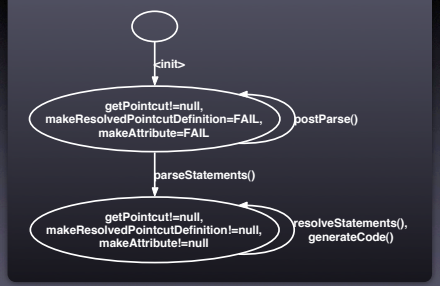
HashTableOfObject



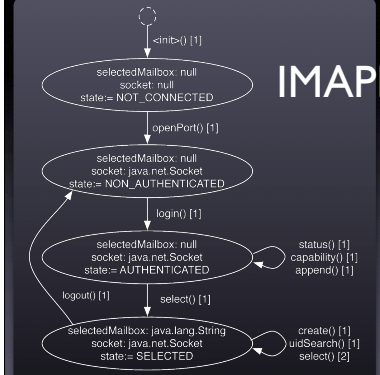
LineNumberGen



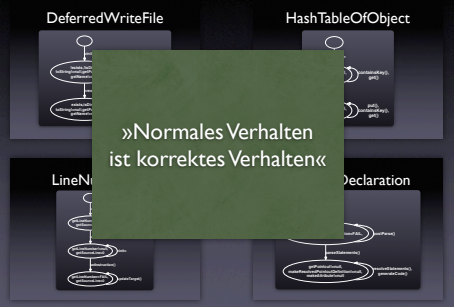
PointcutDeclaration



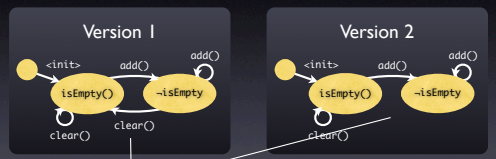
IMAPProtocol



Programme verstehen



Änderungen bewerten



Unterschiede deuten auf Fehler hin

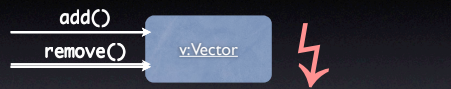
Tom Zimmermann

Verletzungen finden


Holger Cleve


Andrzej Wasylkowski

Fehlerursachen suchen



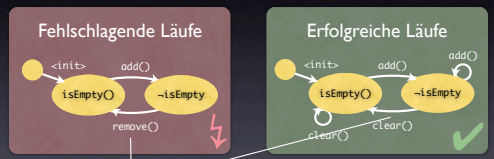
- Welche Mutatoren sind fehlerrelevant?
- Vereinfachen mit Delta Debugging

```
void testVector()
{
    v.add(1);
    v.remove(1);
    assert(v.isEmpty());
}
```


Stephan Neuhaus


Martin Burger

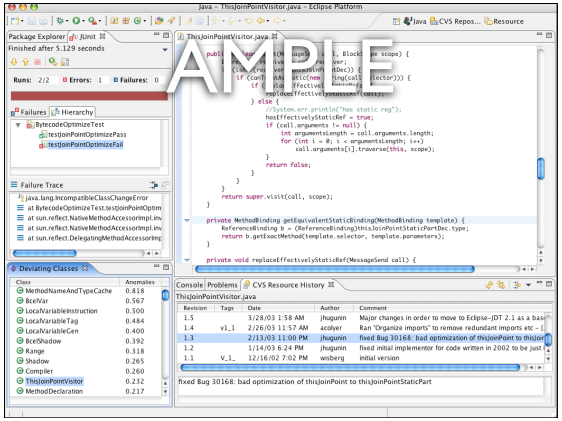
Anomalien erkennen



Unterschiede sind Hinweise auf Fehlerort


Christian Lindig


Valentin Dallmeier



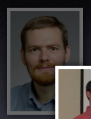
Verwandte Arbeiten

Michael Ernst, MIT
Dynamische Invarianten (TSE 2001)

Vector

After Vector.add():
this._size > 0

Verwandte Arbeiten



Michael Ernst

John Whaley, Stanford
Extraction of OO Interfaces (ISSTA 2002)

<init>

_size = 0

Verwandte Arbeiten

Michael Ernst

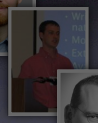
John Whaley

Ras Bodik, Berkeley
Mining Specifications (POPL 2002)

`open()` • `read()` • `close()`

Verwandte Arbeiten

Michael Ernst



John Whaley

Ras Bodik

Tom Henzinger, Stanford / EPFL
Permissive Interfaces (FSE 2005)

– statische Analyse –

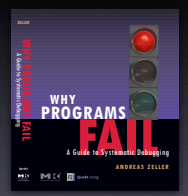
Verwandte Arbeiten

Michael Ernst

John Whaley

Ras Bodik

Tom Henzinger



Automated Debugging

- Vertiefungsvorlesung im Sommersemester 2006
- Vorlesung + Praktikum
- Einstieg für Abschluss-arbeiten am Lehrstuhl