



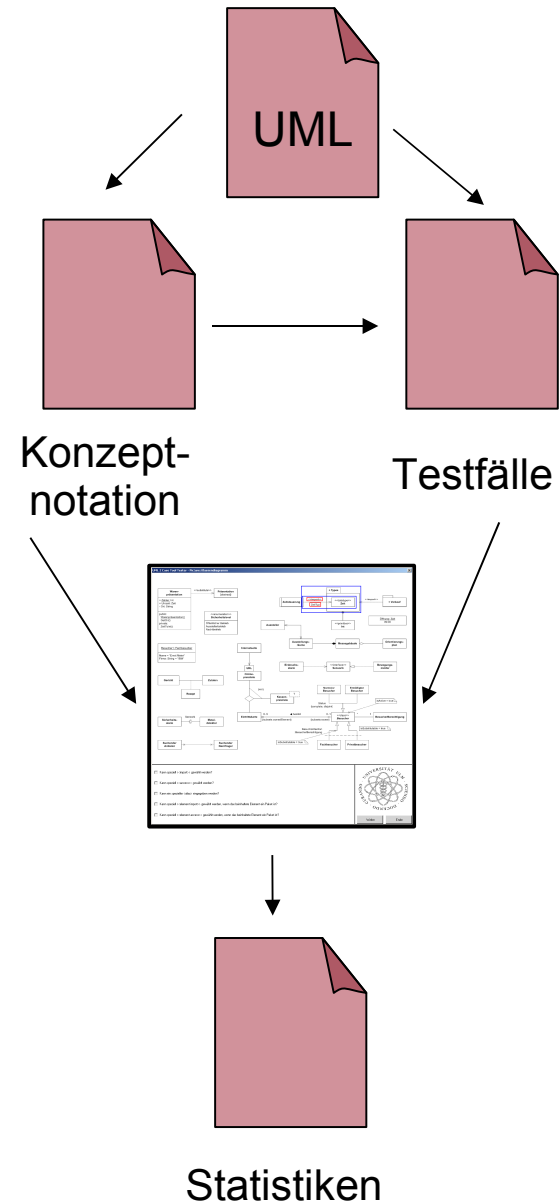
Untersuchung der Sprachkonformität und Vollständigkeit von UML 2.0 Werkzeugen

Motivation

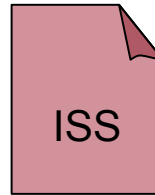
- Teilweise eklatante Missachtung der UML Spezifikationen beim Vorläufer-Standard
- Abweichung der Spezifikation nicht so einfach ersichtlich
- Es existiert eine große Menge an Unterstützungstools, welche UML 2.0 Konformität versprechen
- **Gesucht: Entscheidungsgrundlage, die es ermöglicht, bestehende UML-2-Werkzeuge auf ihre UML-2-Konformität zu überprüfen**
- Basiert auf der Masterarbeit von Daniel Bernauer

Gliederung

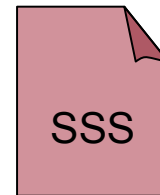
- Die UML 2.0 Spezifikation
- Testfälle erstellen
- Das Bewertungssystem
- Testdiagramme
- Test-Tool U2CTT
- Fazit



UML 2.0 Spezifikation



UML 2.0 Infrastructure Specification



UML 2.0 Superstructure Specification

- Unterteilt in 18 Kapitel
- Ab Kapitel 7 finden wir die sog. Sprachpakete, welche die 13 Diagrammarten der UML beschreiben
- Darin enthalten ein ausführlicher Erläuterungsteil der Konzepte
- Compliance Level
- Textuelle Darstellung

- Generalisierung
- Beschreibung
- Attribute
- Assoziationen
- Bedingungen
- Semantik
- Notation
- Beispiele

Testfälle

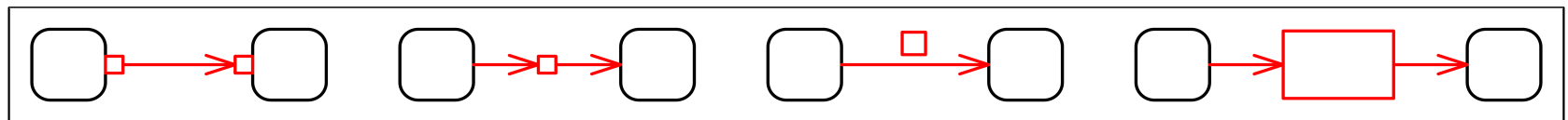
Konformitätsprüfung erfordert Kategorisierung der zu testenden Eigenschaften

Darstellungsart

- Freitextbasiert oder konzeptuell
- Sind die Typen der Konzepte eindeutig identifizierbar ?
- Bsp. Relation: „multiplicity“ <-> „property“

Vollständigkeit

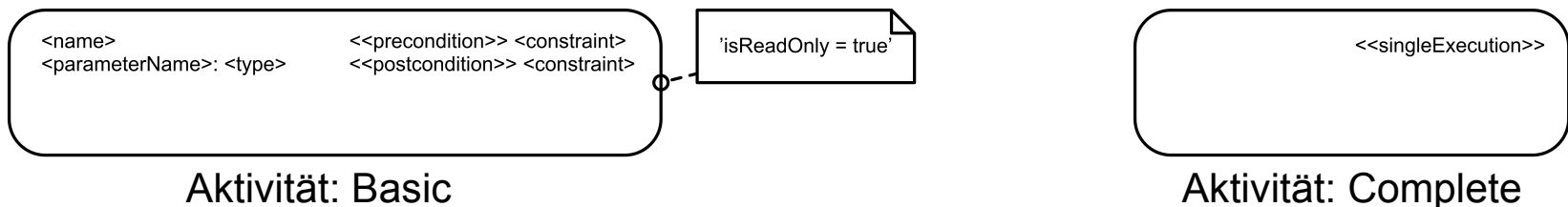
- Einige Konzepte haben alternative Darstellungsformen
- Nicht unbedingt notwendig für Konformität



Testfälle werden auf Basis der „Konzeptnotation“ bestimmt

Testfälle: Konzeptnotation

- Resultiert aus dem Studium der SSS
- Die Abschnitte der SSS-Erläuterungskapitel bilden die Konzepte
- Die Notation zu den Konzepten wird den Teilüberschriften entnommen
- Einteilung in die Pakete bzw. Compliance Level
- Beispiel Aktivität SSS (12.3.4):
 - Notationen in Paketen Basic und Complete
 - 'isReadOnly' benötigt Sonderbehandlung, <<singleExecution>> nicht
 - <<precondition>> <constraint>, <<postcondition>> <constraint> und <parameterName>: <type> werden vererbt



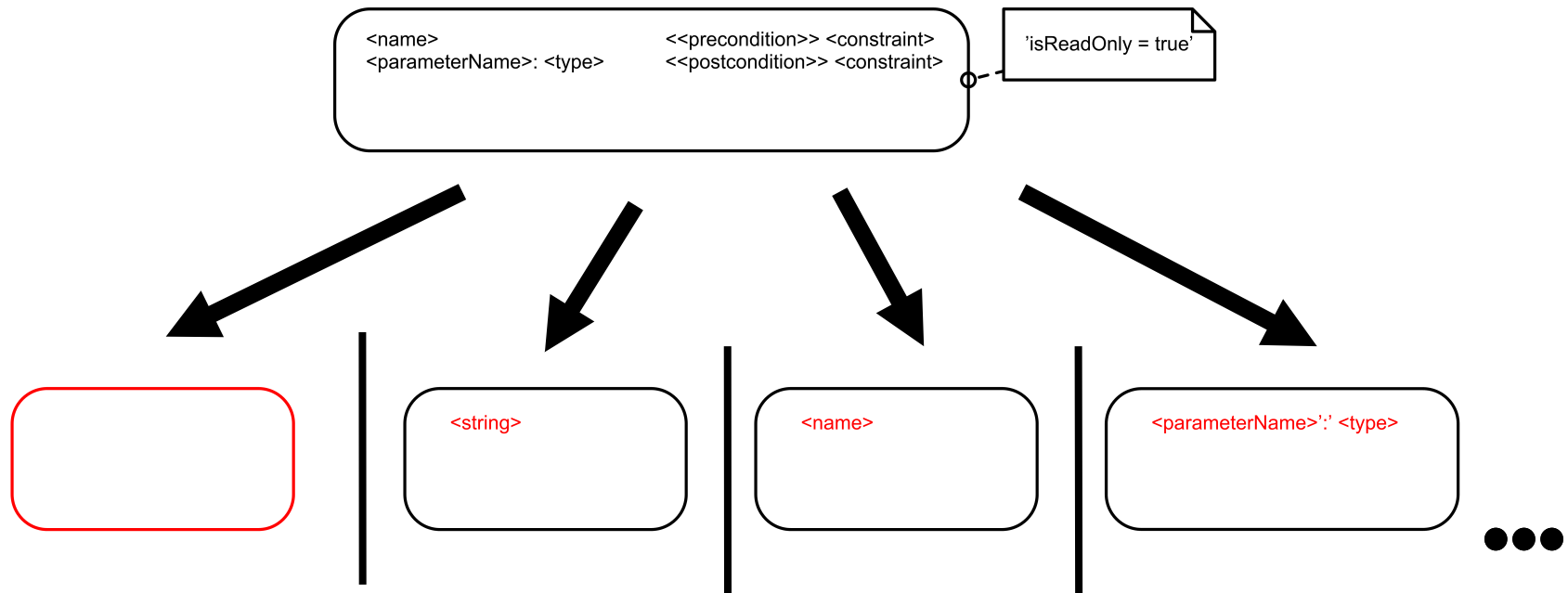
Testfälle: Bestimmung der Testfälle

Zur Bestimmung der Testfälle innerhalb der beschriebenen Konzepte werden die zugehörigen Notationen in folgende Segmente zerlegt und mit Testfällen versehen:

- Jede alternative Form erhält einen Testfall.
- Jeder Freitext an einer spezifischen Position der Form erhält einen Testfall
- Jeder Typ an einer spezifischen Position der Form erhält einen Testfall
- Jedes Symbol an einer spezifischen Position der Form erhält einen Testfall
- Jedes Schlüsselwort an einer spezifischen Position der Form erhält einen Testfall

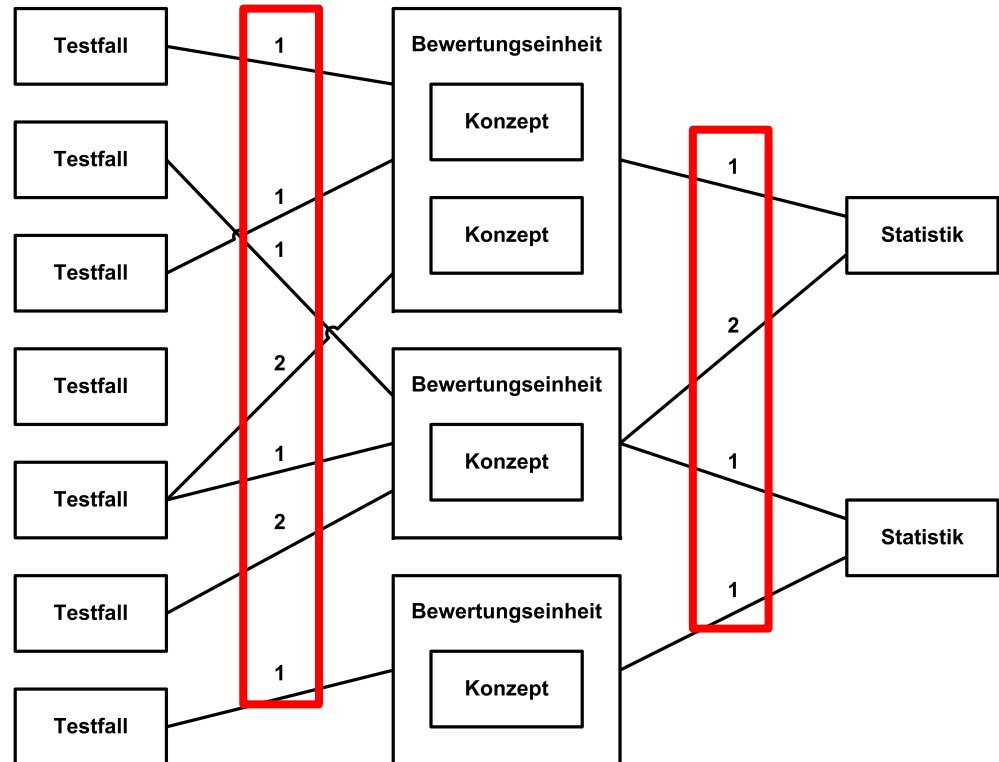
Testfälle: Bestimmung der Testfälle

- Werden nach festen Regeln aus den Konzeptnotationen abgeleitet
- Einteilung in die Bewertungsdimensionen
- Beispiel Aktivität:



Bewertungssystem

- Konzepte mit ihren Testfällen erhalten Bewertungseinheiten
- Die Bewertungseinheiten werden zu Statistiken vereint
- Die Relation Testfall zu Bewertungseinheit und die Relation Bewertungseinheit zu Statistik erhalten Gewichte



Bewertungssystem: Statistiken

Eingeführte Statistiken:

- vollständig <-> auf bestimmte Alternativen beschränkt
- freitext <-> konzeptbasiert
- Diagrammart + Compliance Level
- Bsp: Activity Charts Intermediate konzeptbasiert vollständig

$$MS_s = \frac{\sum_{b=1}^{NS_s} (FB_{s,b} \cdot \frac{\sum_{t=1}^{NB_{s,b}} (FT_{s,b,t} \cdot ET_{s,b,t})}{\sum_{t=1}^{NB_{s,b}} (FT_{s,b,t})})}{\sum_{b=1}^{NS_s} (FB_{s,b})} \quad \text{mit} \quad \begin{array}{l} NS_s > 0 \\ NB_{s,b} > 0 \\ FB_{s,b} > 0 \\ FT_{s,b,t} > 0 \\ ET_{s,b,t} \in [0, 1] \end{array}$$

MS_s = Maßzahl der Statistik s

$FB_{s,b}$ = Bewertungseinheitsfaktor der Bewertungseinheit b in Statistik s

$FT_{s,b,t}$ = Testfallfaktor des Testfalls t in Bewertungseinheit b in Statistik s

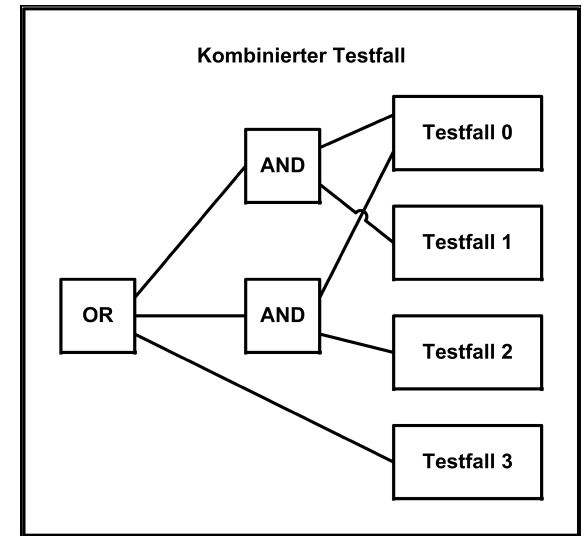
ET_t = Erfüllung von Testfall t

NS_s = Anzahl an Bewertungseinheiten in der Statistik s

$NB_{s,b}$ = Anzahl an Testfällen in der Bewertungseinheit b in Statistik s

Bewertungssystem: kombinierte Testfälle

- Alternativen werden realisiert durch kombinierte Testfälle
- Verwendung der booleschen Algebra AND, OR, Negierung



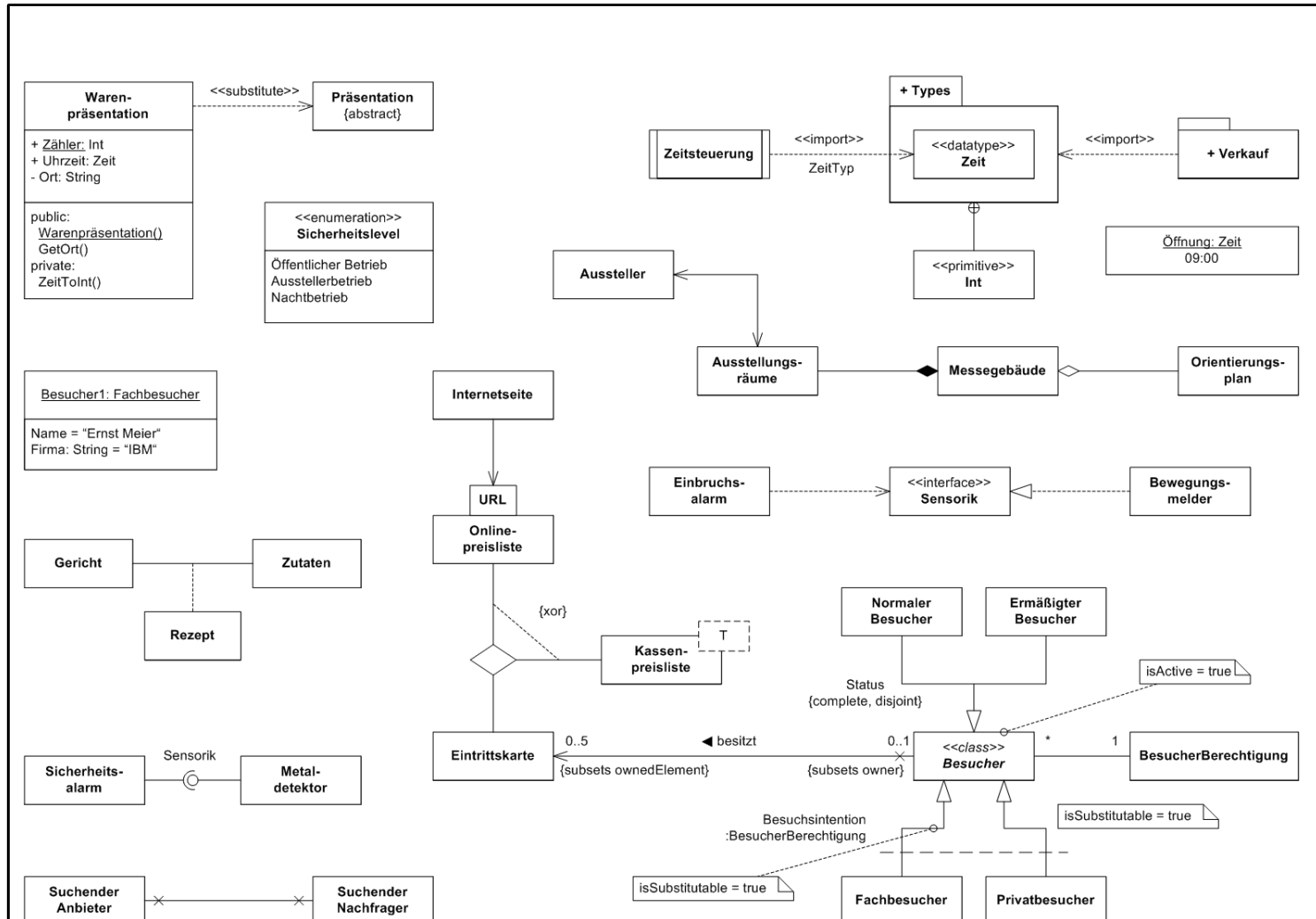
Bewertungssystem: Gewichtung

- Grössere Gewichtung für grundlegende Konzepte sinnvoll ?
Einführung von elementaren Konzepten ?
- Höhere Gewichtung von anspruchsvolleren Konzepten ->
Subjektivität ?
- Lösung durch Aufteilung in Pakete entsprechend der Compliance Level

Testdiagramm

- Aus Testfällen zusammengeführt
 - Unterteilung in Diagramme und Pakete
 - Formale oder beispielhafte Darstellung
 - Erfragung durch geeignete textbasierte oder grafische Kombination der Fragestellung
 - Diagramme sind unvollständig und nicht konsistent
 - Verhinderung von Redundanz durch Mehrfachverwendung von Beispielen
- Kann ein einfacher Text in den Eigenschaften eingegeben werden?
 - Kann speziell die Eigenschaft 'isActive = true' eingegeben werden?
 - ...

Testdiagramm: Beispiel Klassendiagramm



Test-Tool

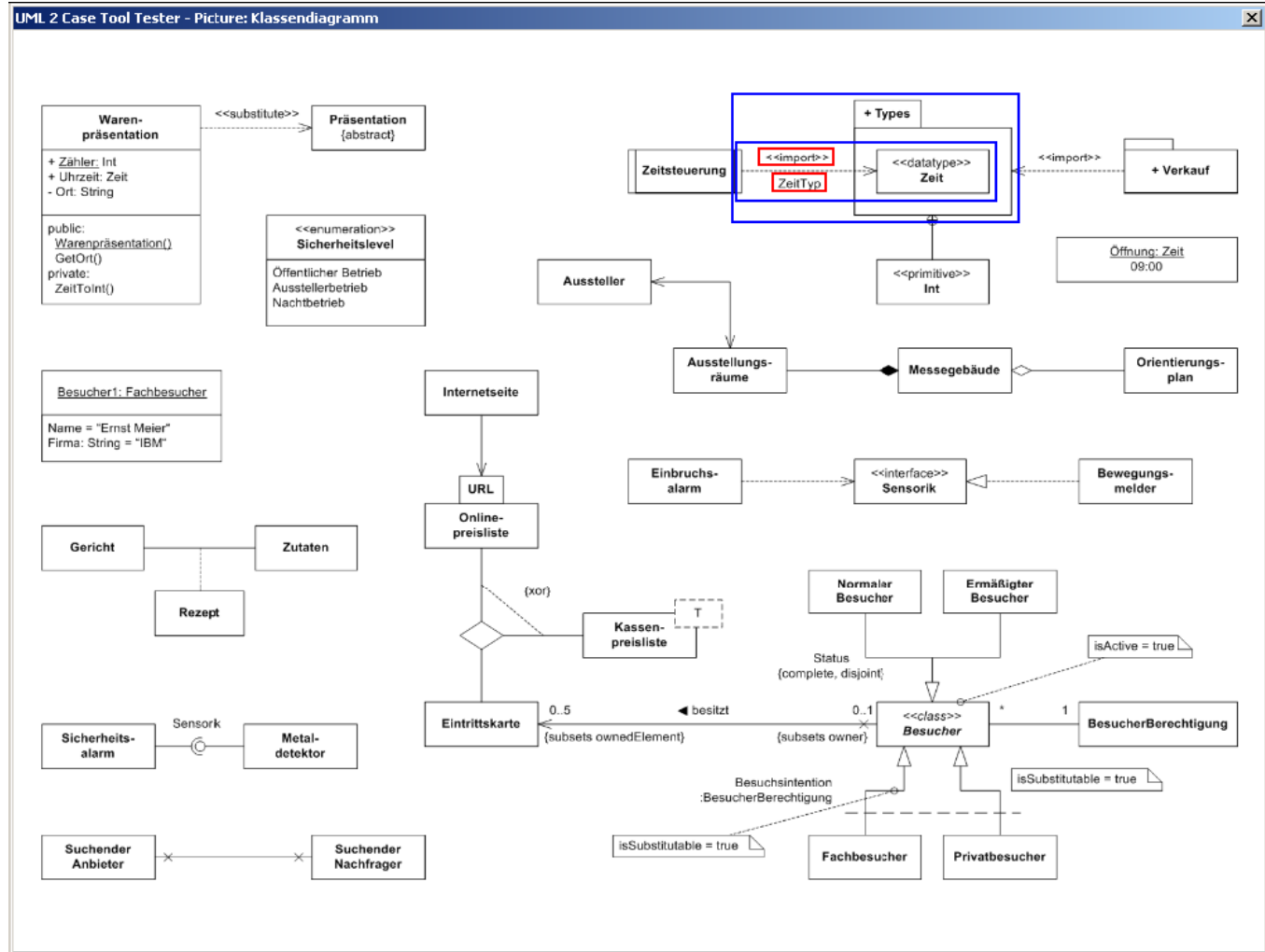
U2CTT

UML 2

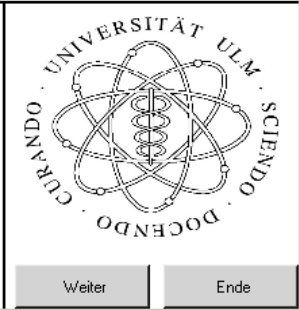
Case

Tool

Tester

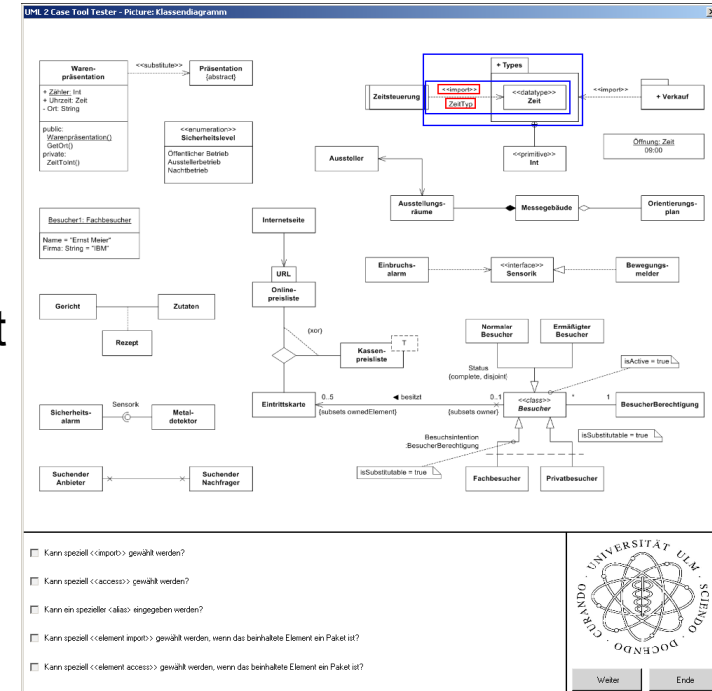


- Kann speziell <<import>> gewählt werden?
- Kann speziell <<access>> gewählt werden?
- Kann ein spezieller <alias> eingegeben werden?
- Kann speziell <<element import>> gewählt werden, wenn das beinhaltete Element ein Paket ist?
- Kann speziell <<element access>> gewählt werden, wenn das beinhaltete Element ein Paket ist?



Test-Tool: Kurzbeschreibung

- Einfaches prototypisches Programmgerüst
 - Kann Bilder (Testdiagramme) darstellen
 - Darauf Markierungen anbringen
 - Fragen darstellen
- Datenbasis in Form einer XML-Datei
- Ausgabe der Testfälle als Statistiken in HTML-File



1. Klassendiagramm Freitext

Testfälle:

Abstraction

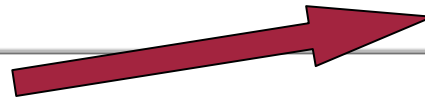
0: Gestrichelter Pfeil

1: Gestrichelter Pfeil mit `<string>`

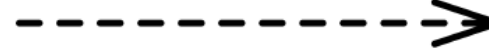
Association

28: Linie

29: Linie mit Spitze rechts



`<string>`



100,00%

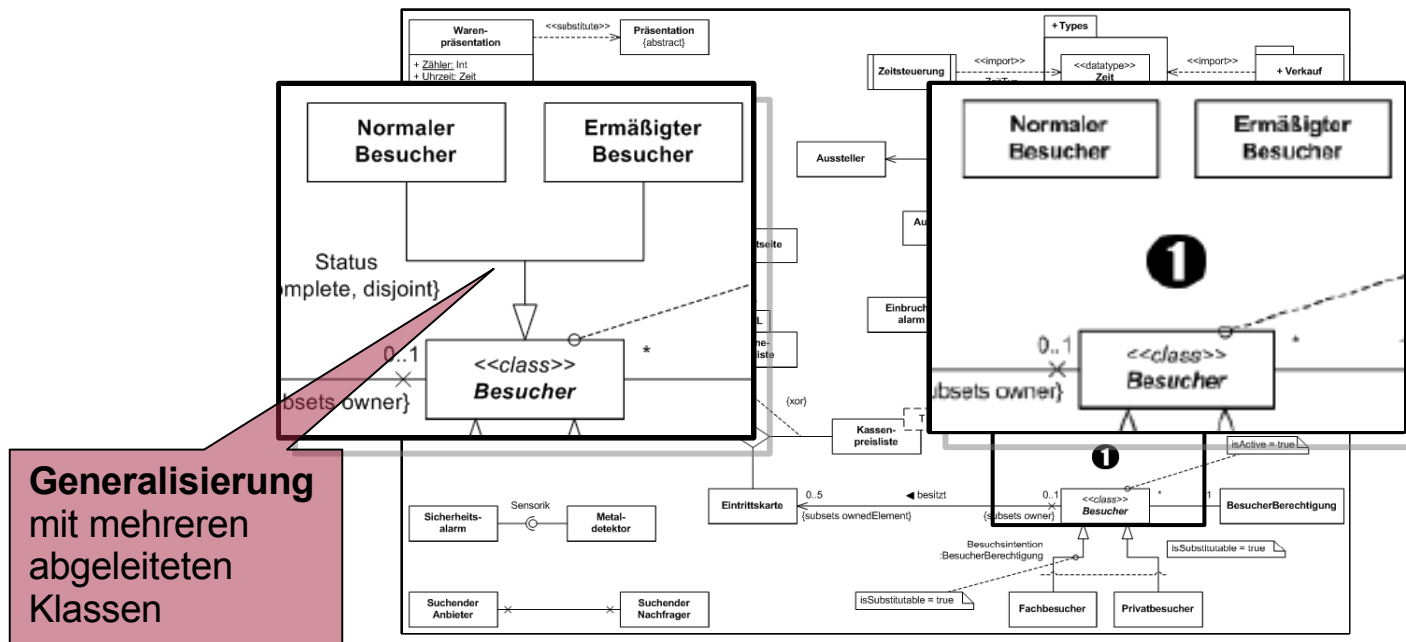
Test-Tool: Beispiel Visio 2003 mit speziellen Stencils

Klassendiagramm Freitext

100%

Klassendiagramm Freitext vollständig

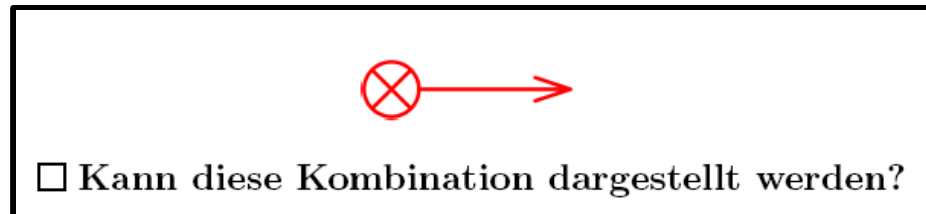
96,33%



Testdiagramm: Klassendiagramm in Visio umgesetzt

Fazit

- Vorgehensweise und Tool zum Überprüfen der UML 2.0 Konformität
- Lässt sich auch für andere Sichtweisen und Untersuchungen anpassen
 - Notationsbedingungen



- Typprüfungen
 - UML-Profiles, bsp SysML-Konformität
- Spezielle Bewertungseinheiten / Statistiken für ActiveChartsIDE
 - Konzeptnotation und Testfälle lassen sich nicht generieren
 - Gewichtungen und Bewertungseinheiten sind subjektiv