

Simulation und Verifikation von UML-basierten Stellwerken

Beispiele und Gegenbeispiele im Zeichen von Verifikation und Validierung

M. Kollmann

Institut für Informationssysteme
Technische Universität Braunschweig

8.11.2006 / Bieleeschweig 8

Gliederung

Einleitung

Software Engineering

Beispiele und Gegenbeispiele

Zusammenfassung und Ausblick

Entwicklung rechnergestützter Systeme

Software

anfangs: Umsetzung von mathematischen Algorithmen → Rechner

- ▶ kurze, überschaubare Programme
- ▶ Verzicht auf Benutzeroberflächen
- ▶ wenige Entwickler

1968: Softwarekrise

- ▶ Hardwarekapazitäten nehmen rasant zu (Rechenleistung, Speicher, ...)
- ▶ Softwareentwicklung kann nicht mithalten

Entwicklung rechnergestützter Systeme

Softwarekrise(n)

Beobachtungen:

- ▶ Gordon E. Moore (1965): Cramming more components onto integrated circuits – Moore's Law
- ▶ Ray Kurzweil (2001): The Law of Accelerating Returns
- Hardware/Systeme wird/werden immer leistungsfähiger (exponentielles Wachstum).
- ▶ **Usability-Probleme** – vgl. Fraunhofer-Institut FIT (DIE ZEIT 21.09.2006 Nr. 39)
 - ▶ 56% aller Menschen, die mit Computerprogrammen zu tun haben, erhalten nicht die notwendigen Bedienungshinweise.
 - ▶ 27% aller Kaufversuche im Internet werden abgebrochen, weil die Käufer nicht finden, was sie suchen, obwohl der Anbieter das Produkt im Sortiment führt.
- ▶ **Methodenvielfalt** zur Modellbildung, zur Verifikation, ...

Software Engineering

Anforderungen an sicherheitskritische Systeme

Sicherheit wird unterschieden in:

- ▶ Sicherheit gegen unauthorisierten Zugriff (security)
- ▶ **technische Sicherheit (safety)**

Die Sicherheit eines Systems wird

- ▶ definiert / erfordert durch Normen (z.B. IEC 61508) / Gesetze (z.B. AtG) und
- ▶ aus der Versagenswahrscheinlichkeit der Sicherungsmechanismen abgeleitet.

Allgemeine Grundsätze:

- ▶ Ein Nachfolgesystem muss mindestens so sicher wie das Vorgängersystem sein.
- ▶ Unabhängigkeit von sicherheitskritischen / nicht-sicherheitskritischen Funktionen ist zu zeigen. Ist dies nicht möglich, ist das gesamte System als sicherheitskritisch einzustufen.
- ▶ **Die technische Sicherheit ist mit geeigneten Methoden nachzuweisen.** Hierzu sind beim V-Modell Verifikation und Validierung durchzuführen.

Software Engineering

Verifikation und Validierung

Verifikation

- ▶ *Nachweis, dass ein Entwicklungsschritt korrekt verlaufen ist.* Zu zeigen ist, dass
 - ▶ die Eingangsgrößen in einen Entwicklungsabschnitt richtig in dessen Ausgangsgrößen überführt wurden.
 - ▶ Richtig bedeutet hier, dass sämtliche Anforderungen, die die Eingangsgrößen erfüllen, auch von den Ausgangsgrößen erbracht werden.
 - ▶ **Beispiel:** Erhalt von Integritätsbedingungen
- ▶ **Vorgehensweisen:** Theorem Proving, Model Checking, ...

Validierung

- ▶ *Nachweis, dass das richtige Produkt entwickelt wurde.* Zu zeigen ist, dass
 - ▶ das Produkt den in einem Lastenheft festgehaltenen Anforderungen bezüglich
 - ▶ Funktionalität,
 - ▶ Anwendbarkeit,
 - ▶ Sicherheitskriterien,
 - ▶ ...genügt.
- ▶ **Vorgehensweise:** Testen

Verwendung von Beispielen und Gegenbeispielen

für Verifikation und Validierung

Verifikation

- ▶ benötigt einen Verifikationsgegenstand,
- ▶ benötigt zu verifizierende (formalisierte) Anforderungen und
- ▶ eine Instanz, die beweisen oder widerlegen kann, dass der Verifikationsgegenstand den Anforderungen genügt.

Schwierigkeiten:

- ▶ Formulierung / Konstruktion des Verifikationsgegenstands,
- ▶ der Anforderungen und
- ▶ das Finden einer geeigneten Verifikationsinstanz.

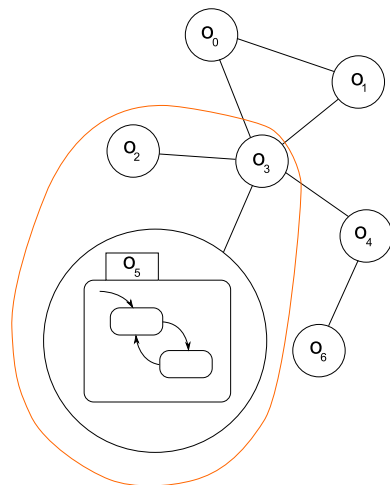
Das Erlernen komplizierter Techniken / Notationen erfolgt in der Regel an Beispielen, so dass eine große Anzahl von Lernenden einen (schnellen / schnelleren) Zugang zu der Methode erhält. Je höher die Problematik des Erwerbs einer Fähigkeit ist, desto einfacher sollte sie vermittelt werden.

Verwendung von Beispielen und Gegenbeispielen

Verifikation

Beispiel: Multi-Object Checking

- ▶ Zu erlernen sind die Erstellung von Zustandsübergangssystemen, die Extraktion von Nachrichten zwischen Zustandsübergangssystemen und die Formulierung von Anforderungen mittels temporallogischer Formeln in einer Multi-Objekt Logik.
- ▶ Die Problematik der Erstellung der einzelnen Zustandsübergangssysteme entspricht der des Model Checking.
- ▶ Die Erstellung semantisch korrekter Formeln erfordert
 - ▶ in einer Formel für jedes einzelne Objekt die Fähigkeiten zur Formulierung von Anforderungen mittels temporallogischer Formeln und
 - ▶ die richtige Schachtelung der Formeln für die einzelnen Objekte.



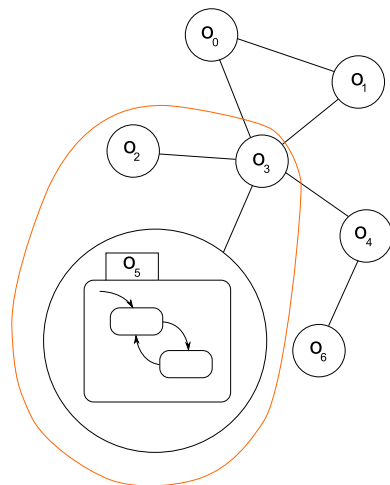
$o_3.(\dots o_2!(\dots) \dots o_5?(\dots) \dots)$

Verwendung von Beispielen und Gegenbeispielen

Verifikation

Beispiel: Multi-Object Checking

- ▶ Zu erlernen sind die Erstellung von Zustandsübergangssystemen, die Extraktion von Nachrichten zwischen Zustandsübergangssystemen und die Formulierung von Anforderungen mittels temporallogischer Formeln in einer Multi-Objekt Logik.
 - ▶ Die Problematik der Erstellung der einzelnen Zustandsübergangssysteme entspricht der des Model Checking.
 - ▶ Die Erstellung semantisch korrekter Formeln erfordert
 - ▶ in einer Formel für jedes einzelne Objekt die Fähigkeiten zur Formulierung von Anforderungen mittels temporallogischer Formeln und
 - ▶ die richtige Schachtelung der Formeln für die einzelnen Objekte.
 - ▶ **Bisheriger Schwachpunkt:** Das Gegenbeispiel ist nur partiell.
 - ▶ Zustandsfolge des Zustandssystems des äußersten Objektes
- (+) Kommunikationsangaben zu dessen Kommunikationspartnern.



$o_3.(\dots o_2!(\dots) \dots o_5?(\dots) \dots)$

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk



konkrete Objekte:

- ▶ Gleisabschnitt
- ▶ Weiche
- ▶ Signal

abstrakte Objekte:

- ▶ Fahrstraße
- ▶ Fahrweg
- ▶ ...

Anforderungen:

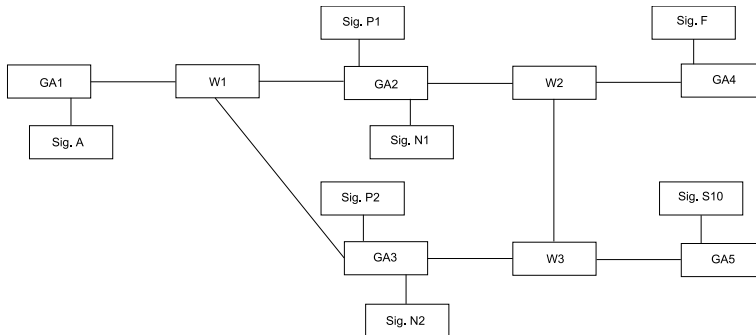
- ▶ Stellbarkeit von Fahrstraßen
- ▶ sicherer Betrieb
- ▶ ...

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

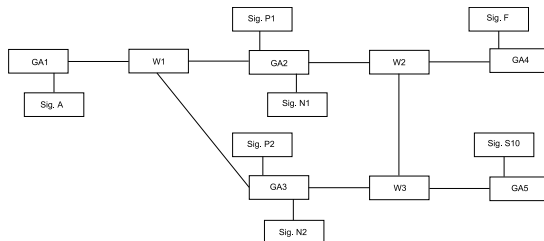


► Objekt- und Assoziationsidentifikation konkreter Objekte

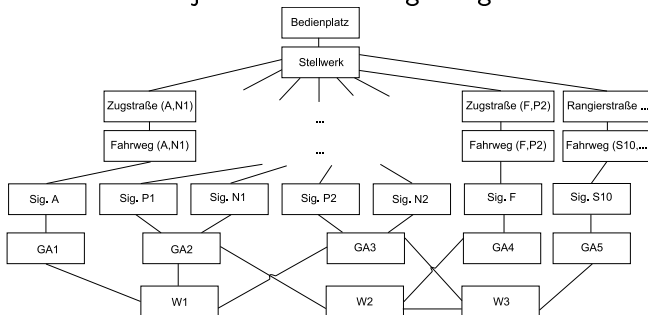


Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

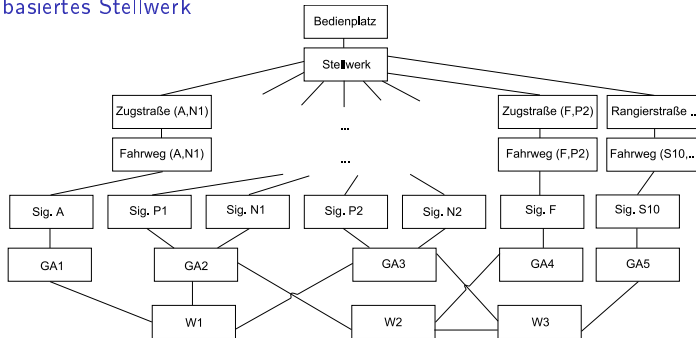


- ▶ Objekt- und Assoziationsidentifikation konkreter Objekte
- ▶ Identifikation abstrakter Objekte und der Umgebung



Verwendung von Beispielen und Gegenbeispielen

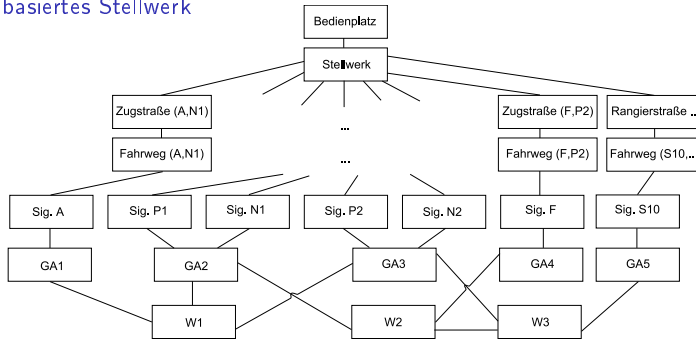
Fallstudie – Ein UML-basiertes Stellwerk



- ▶ Objekt- und Assoziationsidentifikation konkreter Objekte
- ▶ Identifikation abstrakter Objekte und der Umgebung
- ▶ Modellierung des Verhaltens der einzelnen Objekte mit UML-Statecharts und deren
- ▶ Simulation

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

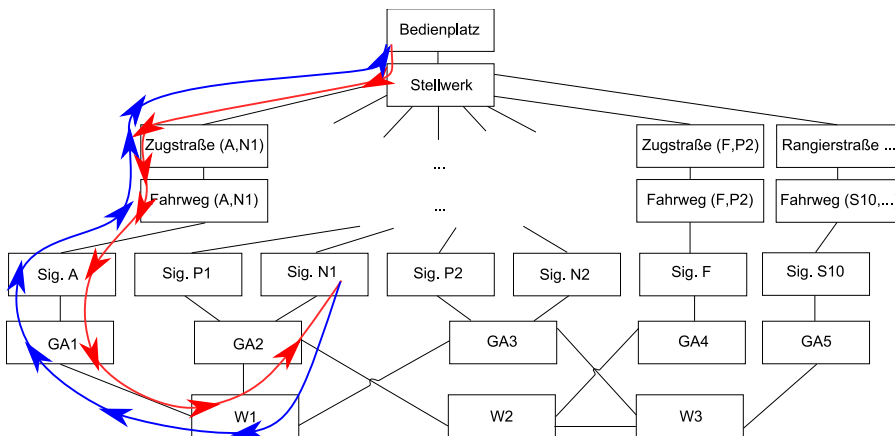
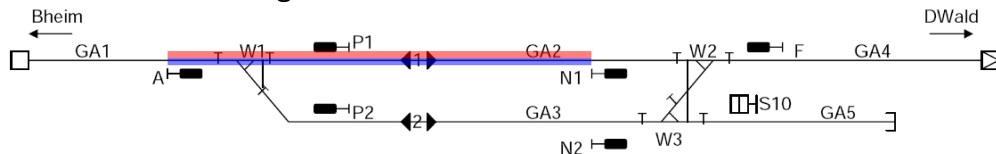


- ▶ Objekt- und Assoziationsidentifikation konkreter Objekte
- ▶ Identifikation abstrakter Objekte und der Umgebung
- ▶ Modellierung des Verhaltens der einzelnen Objekte mit UML-Statecharts und deren
- ▶ Simulation
- ▶ Verifikation von Anforderungen, die einzelne Objekte betreffen, durch Model Checking
- ▶ Verifikation komplexer Anforderungen durch Multi-Object Checking

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

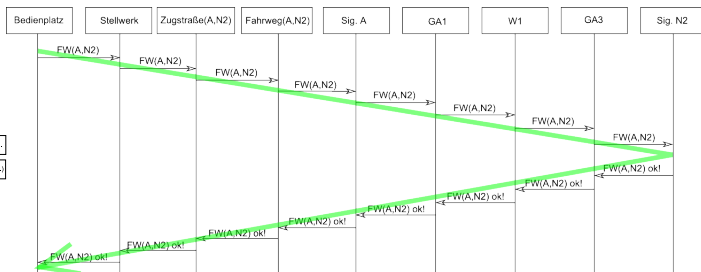
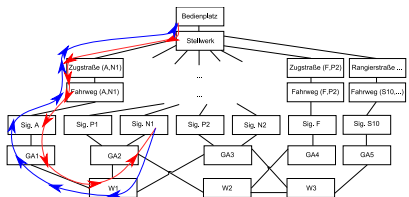
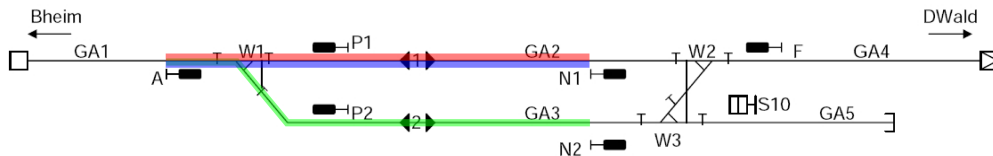
Verifikation – Bildung einer Fahrstraße – 1



Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

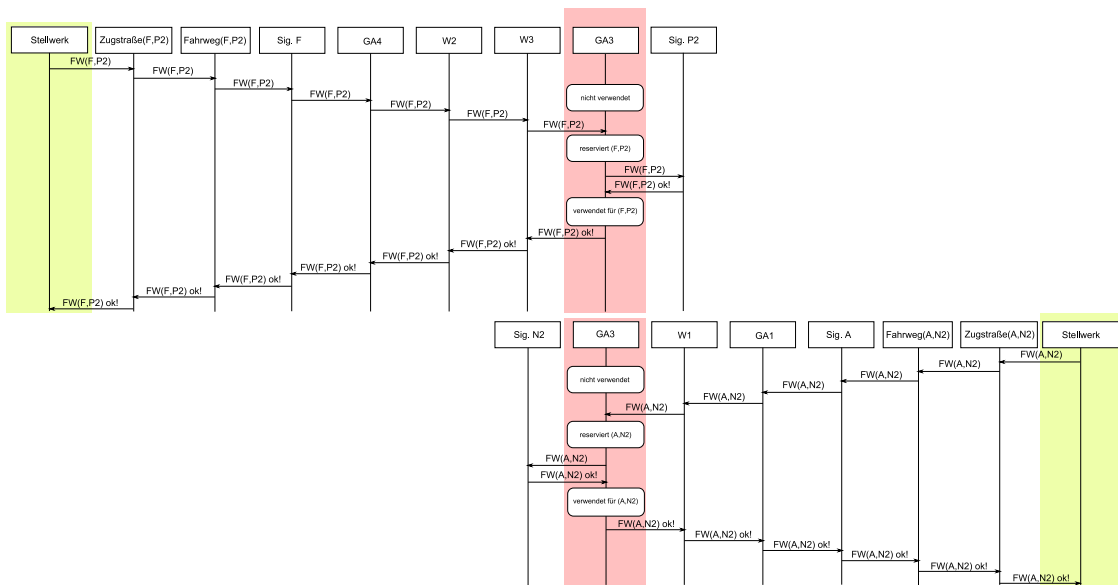
Verifikation – Bildung einer Fahrstraße – 1



Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

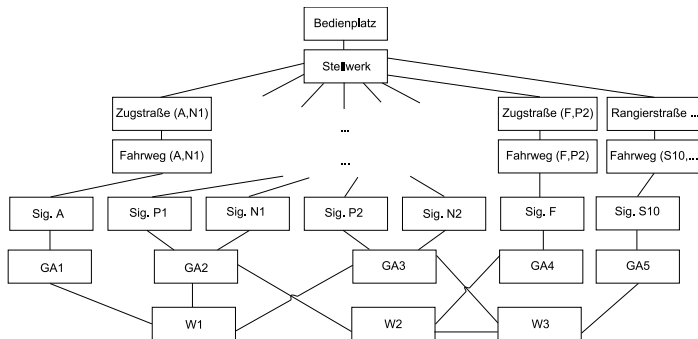
Verifikation – Gegenseitiger Ausschluß von Fahrstraßen



Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

Verifikation – Gegenseitiger Ausschluß von Fahrstraßen



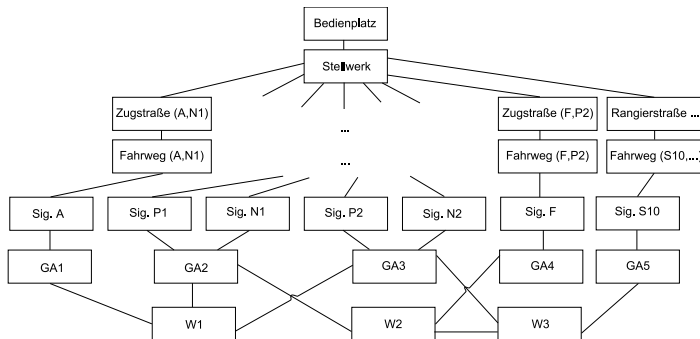
Beispiel:

$Stellwerk.(G \neg F(Zugstra\beta e(F,P2)!*(aktiv) \wedge Zugstra\beta e(A,N2)!*(aktiv)))$

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

Verifikation – Gegenseitiger Ausschluß von Fahrstraßen



Beispiel:

$Stellwerk.(G \neg F(Zugstra\beta e(F,P2)!*(aktiv) \wedge Zugstra\beta e(A,N2)!*(aktiv)))$

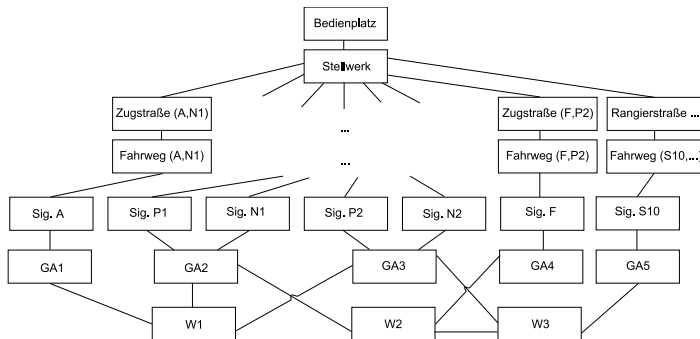
...

GA3 in Fahrweg für (A,N2): $Stellwerk.(G \neg F(... GA3!*(reserviert(A,N2) \vee verwendet\ f\ur(A,N2)) \dots \wedge \dots GA3!*(reserviert(A,N2) \vee verwendet\ f\ur(A,N2)) \dots))$

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

Verifikation – Gegenseitiger Ausschluß von Fahrstraßen



Beispiel:

$Stellwerk.(G \neg F(Zugstra\beta e(F,P2)!*(aktiv) \wedge Zugstra\beta e(A,N2)!*(aktiv)))$

...

GA3 in Fahrweg für (A,N2): $Stellwerk.(G \neg F(... GA3!*(reserviert(A,N2) \vee verwendet\ f\ur(A,N2)) \dots \wedge \dots GA3!*(reserviert(A,N2) \vee verwendet\ f\ur(A,N2)) \dots))$

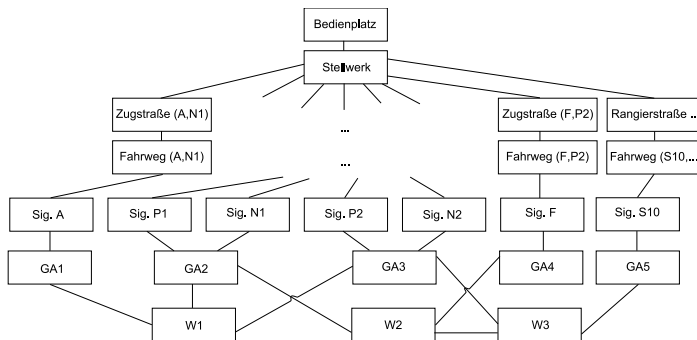
...

TRUE

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

Verifikation – Bildung einer Fahrstraße – 2



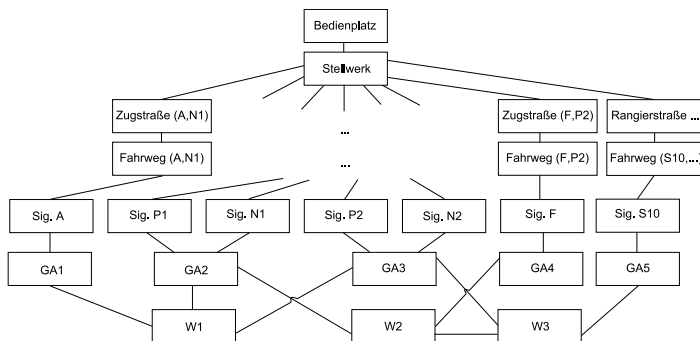
Beispiel:

Prüfbedingung: Wenn GA3 nicht verwendet wird, kann immer der Fahrweg von A nach N2 benutzt werden.

Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

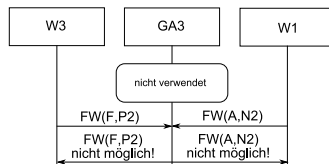
Verifikation – Bildung einer Fahrstraße – 2



Beispiel:

Prüfbedingung: Wenn GA3 nicht verwendet wird, kann immer der Fahrweg von A nach N2 benutzt werden.

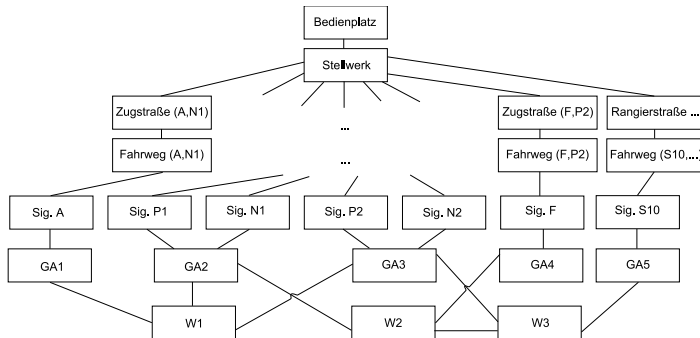
Auswertung: FALSE



Verwendung von Beispielen und Gegenbeispielen

Fallstudie – Ein UML-basiertes Stellwerk

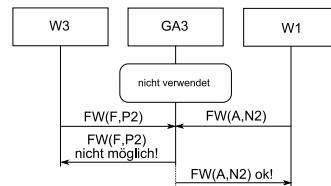
Verifikation – Bildung einer Fahrstraße – 2



Beispiel:

Prüfbedingung: Wenn GA3 nicht verwendet wird, kann immer der Fahrweg von A nach N2 benutzt werden.

Konsequenz: Priorisierung eines Fahrweges bei gleichzeitiger Anfrage

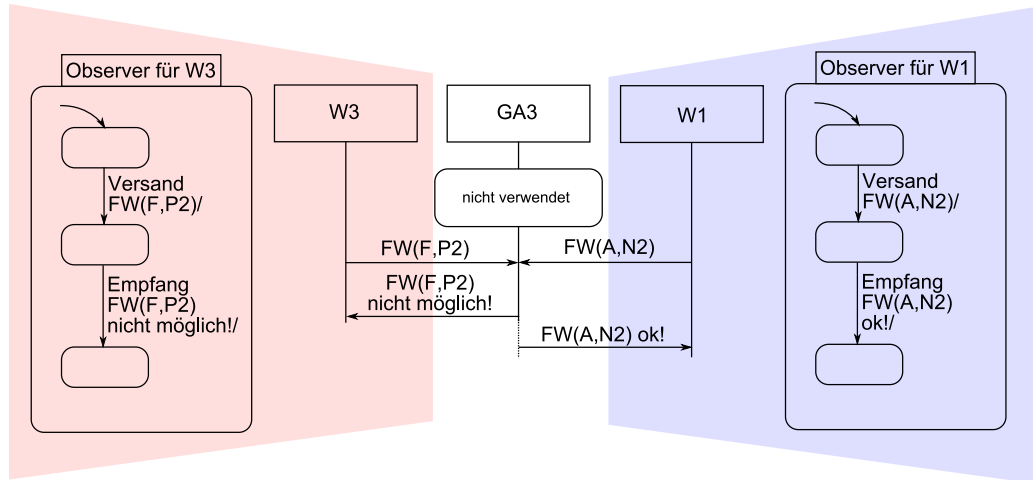


Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

Idee:

1. Erstelle für jeden Kommunikationspartner, der im (initialen) Gegenbeispiel angesprochen ist, einen Observer, der die Kommunikation überwacht.

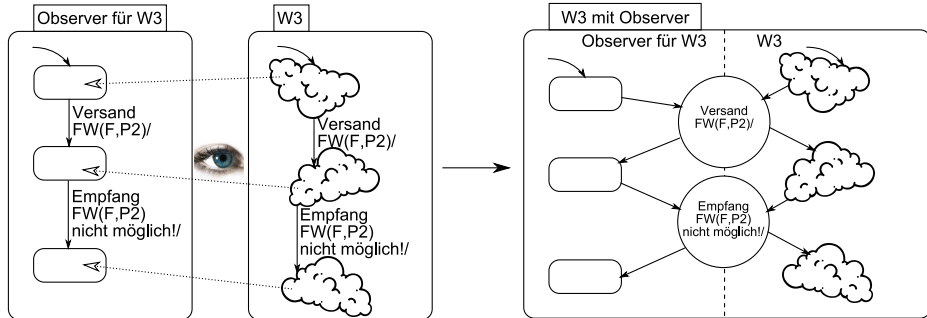


Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

Idee:

1. Erstelle für jeden Kommunikationspartner, der im (initialen) Gegenbeispiel angesprochen ist, einen Observer, der die Kommunikation überwacht.
2. Kombiniere den Zustandsautomaten eines jeden Kommunikationspartners mit allen bis dahin für diesen Kommunikationspartner ermittelten Observern.

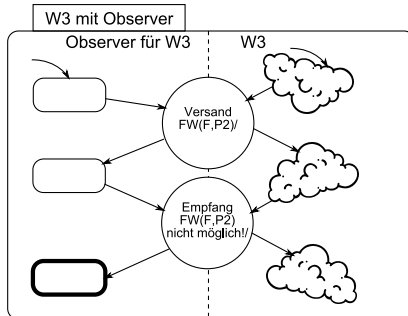


Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

Idee:

1. Erstelle für jeden Kommunikationspartner, der im (initialen) Gegenbeispiel angesprochen ist, einen Observer, der die Kommunikation überwacht.
2. Kombiniere den Zustandsautomaten eines jeden Kommunikationspartners mit allen bis dahin für diesen Kommunikationspartner ermittelten Observern.
3. Beginnend mit dem Kommunikationspartner, mit dem ein Objekt zuletzt kommuniziert hat, wird nun für jeden Kommunikationspartner überprüft, ob die in den Observern geforderte Kommunikation erfolgen kann.



Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

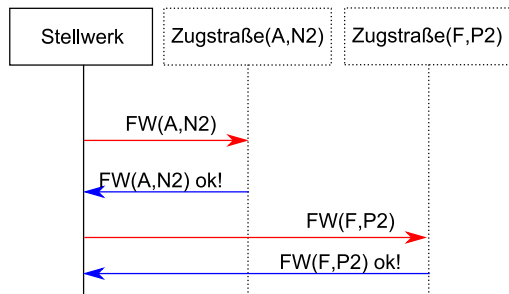
Idee:

1. Erstelle für jeden Kommunikationspartner, der im (initialen) Gegenbeispiel angesprochen ist, einen Observer, der die Kommunikation überwacht.
2. Kombiniere den Zustandsautomaten eines jeden Kommunikationspartners mit allen bis dahin für diesen Kommunikationspartner ermittelten Observern.
3. Beginnend mit dem Kommunikationspartner, mit dem ein Objekt zuletzt kommuniziert hat, wird nun für jeden Kommunikationspartner überprüft, ob die in den Observern geforderte Kommunikation erfolgen kann.
4. Dieses Verfahren wird solange angewendet, bis dass alle (auch eventuell neu angeforderte) Kommunikationspartner abgearbeitet wurden.
5. Behauptet wird jeweils, dass die vom Observer überwachte Kommunikation **nicht** erfolgen kann, so dass eine auf die Kommunikation passende Zustandsfolge für den Kommunikationspartner als (lokales) Gegenbeispiel generiert wird.
6. Wird ein Objekt mehrfach nachgefragt, ist die Erzeugung eines lokalen Gegenbeispiels zu wiederholen, falls zusätzliche Observer hinzugekommen sind.
7. Die Komposition aller lokalen Gegenbeispiele ergibt ein globales Gegenbeispiel.

Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

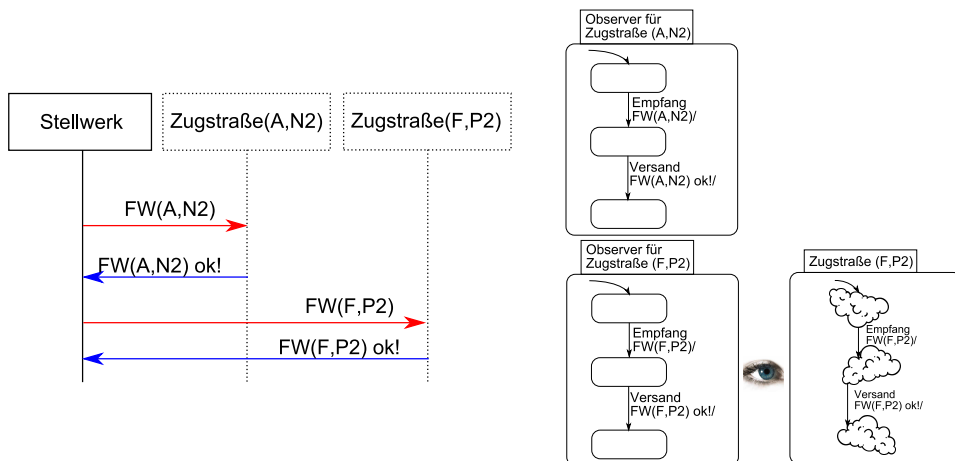
Beispiel: Überprüft werde, dass die Zugstraßen (A,N2) und (F,P2) nacheinander eingestellt werden können. Hierzu wird behauptet, dass dies nicht möglich sei. Das für das Objekt Stellwerk erzeugte Gegenbeispiel beinhalte Kommunikation mit Zugstraße(A,N2) und Zugstraße(F,P2), wobei zuerst Zugstraße (A,N2) eingestellt wurde. Diese wurde dann aufgelöst und Zugstraße (F,P2) eingestellt.



Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

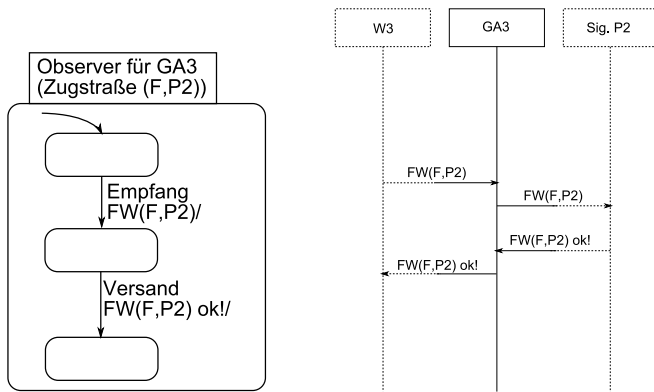
Beispiel: Für die Objekte Zugstraße(A,N2) und Zugstraße(F,P2) werden die entsprechenden Observer generiert und mit Zugstraße(F,P2) fortgefahren. Zugstraße(F,P2) erfordere wiederum Kommunikation mit dem Startsignal F ...



Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

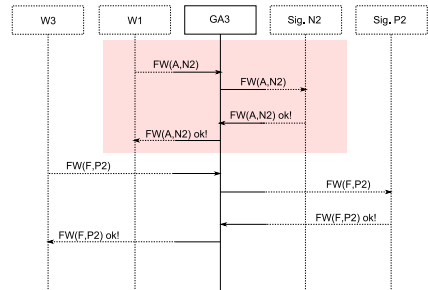
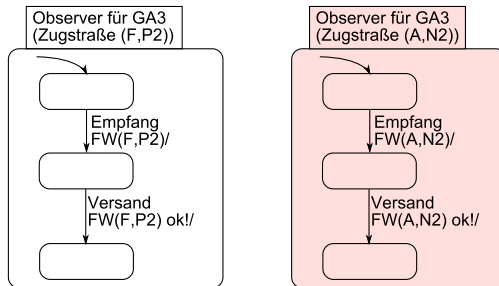
Beispiel: Gleisabschnitt GA3 ist für beide Zugstraßen zu betrachten. Zuerst wird ein Observer ausgehend von Zugstraße(F,P2) generiert und ausgewertet.



Verwendung von Beispielen und Gegenbeispielen

Gegenbeispiele für Multi-Object Checking

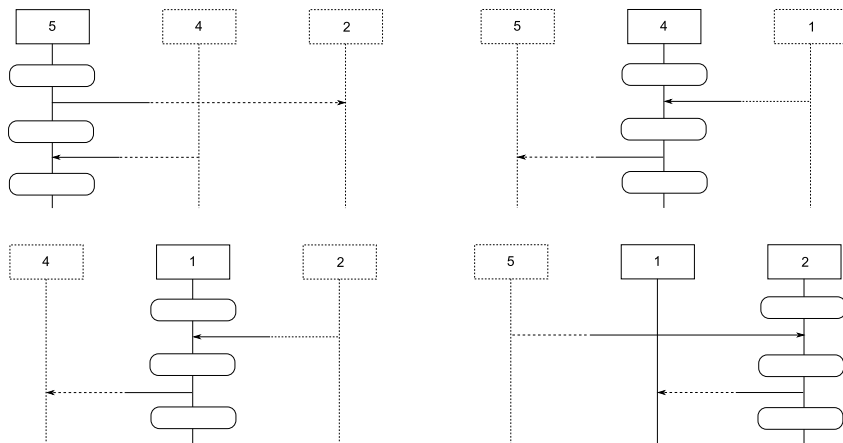
Beispiel: Wenn die Bearbeitung von Gleisabschnitt GA3 für Zugstraße(A,N2) erfolgt, werden schließlich beide Observer mit GA3 kombiniert. Das bisherige lokale Gegenbeispiel wird verworfen.



Verwendung von Beispielen und Gegenbeispielen

Darstellung der Gegenbeispiele für Multi-Object Checking

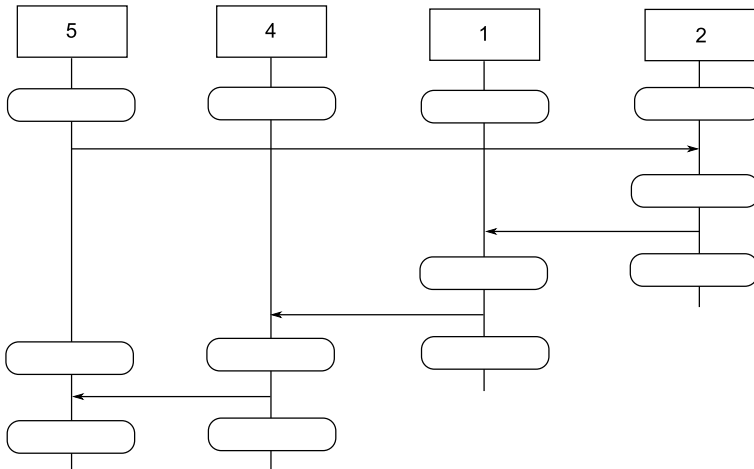
- ▶ Jede lokale Zustandsfolge definiert ein partielles / abstraktes Sequenzdiagramm.



Verwendung von Beispielen und Gegenbeispielen

Darstellung der Gegenbeispiele für Multi-Object Checking

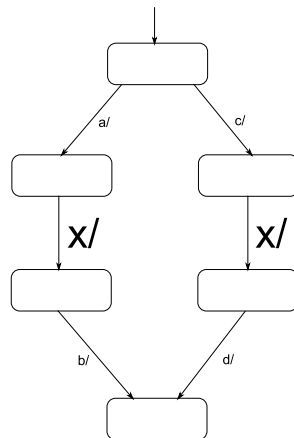
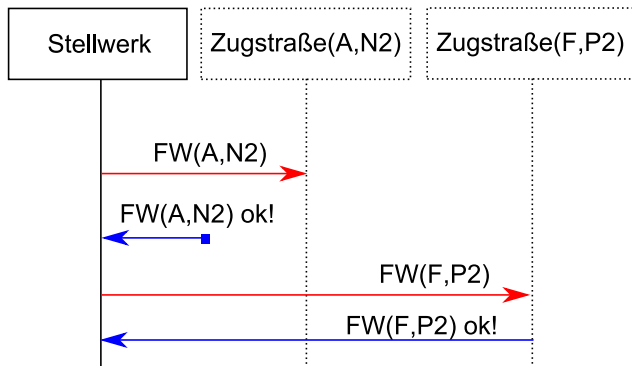
- ▶ Jede lokale Zustandsfolge definiert ein partielles / abstraktes Sequenzdiagramm.
- ▶ Die Komposition aller dieser Sequenzdiagramme ist ein Sequenzdiagramm, welches das Verletzen der Anforderung unter Berücksichtigung der Kommunikation illustriert.



Verwendung von Beispielen und Gegenbeispielen

Partielle vs. vollständige Gegenbeispiele für Multi-Object Checking

Problem: Eine lokale Zustandsfolge, die die Kommunikation widerspiegelt, ist nicht zwingend existent bzw. eindeutig. Dann besteht keine Möglichkeit der automatischen Generierung eines vollständigen Gegenbeispiels durch den beschriebenen Algorithmus.



Verwendung von Beispielen und Gegenbeispielen

Partielle vs. vollständige Gegenbeispiele für Multi-Object Checking

Problem: Eine lokale Zustandsfolge, die die Kommunikation widerspiegelt, ist nicht zwingend existent bzw. eindeutig. Dann besteht keine Möglichkeit der automatischen Generierung eines vollständigen Gegenbeispiels durch den beschriebenen Algorithmus.

Lösungsstrategien:

1. Interaktion durch den Benutzer

- ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, durch den der Benutzer den Fehler (oder einen anderen) bereits beheben kann.
- ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, in dem der Benutzer z.B. Kommunikation unterdrücken kann.

Verwendung von Beispielen und Gegenbeispielen

Partielle vs. vollständige Gegenbeispiele für Multi-Object Checking

Problem: Eine lokale Zustandsfolge, die die Kommunikation widerspiegelt, ist nicht zwingend existent bzw. eindeutig. Dann besteht keine Möglichkeit der automatischen Generierung eines vollständigen Gegenbeispiels durch den beschriebenen Algorithmus.

Lösungsstrategien:

1. Interaktion durch den Benutzer

- ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, durch den der Benutzer den Fehler (oder einen anderen) bereits beheben kann.
- ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, in dem der Benutzer z.B. Kommunikation unterdrücken kann.
 - Observerbeeinflussung

Verwendung von Beispielen und Gegenbeispielen

Partielle vs. vollständige Gegenbeispiele für Multi-Object Checking

Problem: Eine lokale Zustandsfolge, die die Kommunikation widerspiegelt, ist nicht zwingend existent bzw. eindeutig. Dann besteht keine Möglichkeit der automatischen Generierung eines vollständigen Gegenbeispiels durch den beschriebenen Algorithmus.

Lösungsstrategien:

1. Interaktion durch den Benutzer
 - ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, durch den der Benutzer den Fehler (oder einen anderen) bereits beheben kann.
 - ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, in dem der Benutzer z.B. Kommunikation unterdrücken kann.
 - Observerbeeinflussung
2. Wenn erforderliche Kommunikation nicht beobachtet werden kann, wird für den initiiierende Kommunikationspartner versucht, eine alternative Zustandsfolge zu bestimmen.

Verwendung von Beispielen und Gegenbeispielen

Partielle vs. vollständige Gegenbeispiele für Multi-Object Checking

Problem: Eine lokale Zustandsfolge, die die Kommunikation widerspiegelt, ist nicht zwingend existent bzw. eindeutig. Dann besteht keine Möglichkeit der automatischen Generierung eines vollständigen Gegenbeispiels durch den beschriebenen Algorithmus.

Lösungsstrategien:

1. Interaktion durch den Benutzer

- ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, durch den der Benutzer den Fehler (oder einen anderen) bereits beheben kann.
- ▶ Präsentation des bis dahin ermittelten Teil eines möglichen Systemlaufes, in dem der Benutzer z.B. Kommunikation unterdrücken kann.
 - Observerbeeinflussung

2. Wenn erforderliche Kommunikation nicht beobachtet werden kann, wird für den initiiierende Kommunikationspartner versucht, eine alternative Zustandsfolge zu bestimmen.

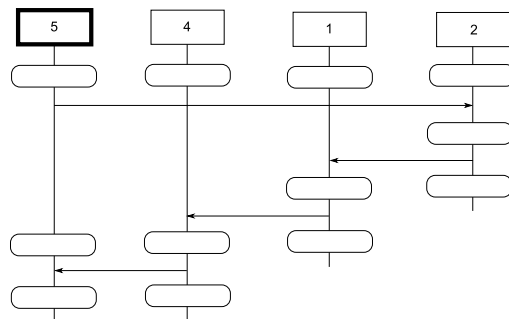
- Observerbeeinflussung

Verwendung von Beispielen und Gegenbeispielen

Multi-Object Checking Gegenbeispiele für die Validierung

Ein Test kann von einem Multi-Object Checking Gegenbeispiel abgeleitet werden aus:

- ▶ einem Testobjekt (System, Objekt, Methode, ...),

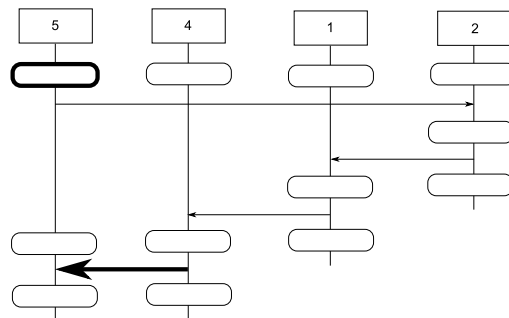


Verwendung von Beispielen und Gegenbeispielen

Multi-Object Checking Gegenbeispiele für die Validierung

Ein Test kann von einem Multi-Object Checking Gegenbeispiel abgeleitet werden aus:

- ▶ einem Testobjekt (System, Objekt, Methode, ...),
- ▶ einer Menge von Eingabedaten,

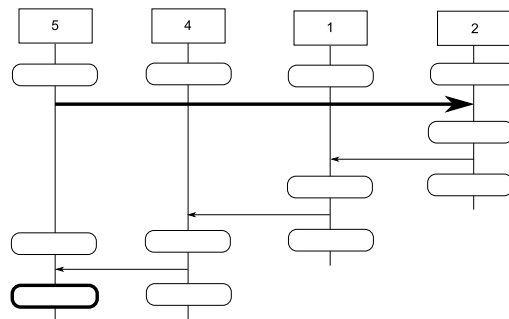


Verwendung von Beispielen und Gegenbeispielen

Multi-Object Checking Gegenbeispiele für die Validierung

Ein Test kann von einem Multi-Object Checking Gegenbeispiel abgeleitet werden aus:

- ▶ einem Testobjekt (System, Objekt, Methode, ...),
- ▶ einer Menge von Eingabedaten,
- ▶ einer Menge erwarteter Ausgabedaten,

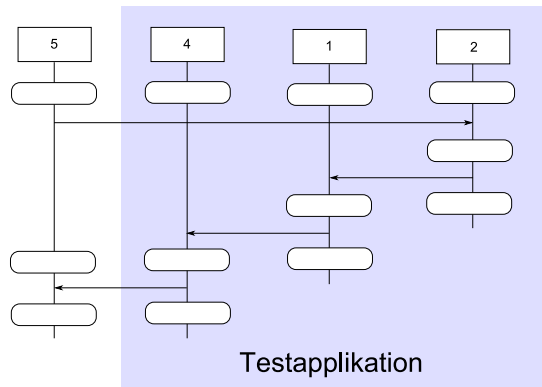


Verwendung von Beispielen und Gegenbeispielen

Multi-Object Checking Gegenbeispiele für die Validierung

Ein Test kann von einem Multi-Object Checking Gegenbeispiel abgeleitet werden aus:

- ▶ einem Testobjekt (System, Objekt, Methode, ...),
- ▶ einer Menge von Eingabedaten,
- ▶ einer Menge erwarteter Ausgabedaten,
- ▶ einer Instanz, die den Test durchführt und

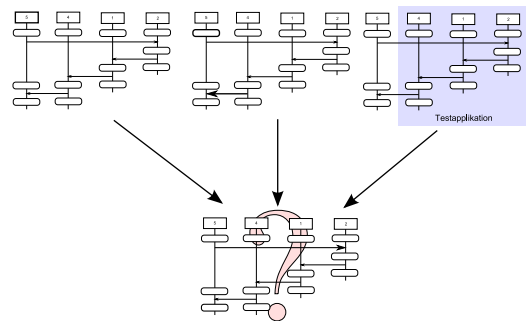


Verwendung von Beispielen und Gegenbeispielen

Multi-Object Checking Gegenbeispiele für die Validierung

Ein Test kann von einem Multi-Object Checking Gegenbeispiel abgeleitet werden aus:

- ▶ einem Testobjekt (System, Objekt, Methode, ...),
- ▶ einer Menge von Eingabedaten,
- ▶ einer Menge erwarteter Ausgabedaten,
- ▶ einer Instanz, die den Test durchführt und
- ▶ einer Instanz, die das Testergebnis auswertet.



Zusammenfassung

Beispiele und Gegenbeispiele dienen zum (besseren) Verständnis problematischer Sachverhalte.

Beispiele und Gegenbeispiele können die Entwicklung (sicherheitskritischer) Systeme unterstützen

- ▶ bei der Erlernung von Methoden und Techniken,
- ▶ bei der Erkennung geeigneter Methoden und Techniken für die einzelnen Entwicklungsphasen,
- ▶ zur Erkennung und Behebung von Fehlern während Verifikationsprozessen,
- ▶ zur Bereitstellung von Testabläufen und Testdaten,
- ▶ ...

Die frühzeitige Erkennung von Fehlern durch Beispiele und Gegenbeispiele reduziert Entwicklungszeiten und spart damit Kosten.

Ausblick

Implementierung des Algorithmus zur Erzeugung von Gegenbeispielen und dessen Demonstration an einer Fallstudie

Entwicklung eines Tools zur automatischen Zerlegung von komplexen Zustandmaschinen einzelner Objekte

- ▶ Parallelität
- ▶ Hierarchie

Analyse der automatischen Anpassung der Prüfbedingungen, die als Formeln in einer Multi-Objekt Logik vorliegen

Vielen Dank für Ihre Aufmerksamkeit!

Sie haben Fragen?