Materialsammlung - Formale Methoden: OCL und vdm++

Prof. Dr. Hans-Jürgen Buhl

Sommersemester 2006
Fachgruppe Mathematik und Informatik
Fachbereich Mathematik und Naturwissenschaften
Bergische Universität Wuppertal

Praktische Informatik
PIBUW - SS0601
April 2006
2. Auflage, 2006
Praktische Informatik 02

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

CM.516 / CM.939 / CM.940
Formale Methoden
4 V Di 12 - 14 und Do 12 - 14 in D13.08

Einordnung: Master IT; Master Wirtschaftsmathematik: Wahpflichtbereich Informatik; Diplom Mathematik; Nebenfächer und Studienschwerpunkte Informatik anderer Studiengänge

Inhalt: Formale Spezifikation von Softwaremodulen kann zu einer enormen Qualitätssteigerung in der Softwareentwicklung führen. Sie ermöglicht mit Hilfe von (mathematischen) Modellen, die (gewünschten) Eigenschaften von Programmen exakt zu definieren und überprüfbar zu machen.

Vorkenntnisse: Programmieren in C++; Nutzung von UML-Klassendiagrammen; Grundlagen der Prädikatenlogik

CM.517 / CM.941 / CM.942
Übungen zu Formale Methoden
2 Ü Mi 16 - 18 in D13.08 (Buhl)

Vorbemerkungen:

Formale Methoden

Inhalte:

1. Softwaregüte

2. Zusicherungen in Algorithmen: Konstruktoren, Modifikatoren, Observatoren und Destruktoren; Ausnahmebedingungen

3. Methodik „Programming by Contract“: Vorbedingungen, Nachbedingungen und Invarianten; Rechte und Pflichten von Softwareanbieter/Softwarenutzer

4. EBNF zur formalen Spezifikation freier Eingabesprachen, UML-Klassendiagramme

5. Startwerte, Vererbung von Klasseninvarianten, Methodenvor- und -nachbedingungen

6. Formale Spezifikation (zur Zeit in OCL2):
   UML-Klassendiagramme und „Constraints“, virtuelle Attribute und Methoden, redundante Attribte und Methoden, „Constraints“ an Attribte, Methoden und Assoziationen; Container-Typen; Frame-Regeln

7. Fallstudien von formal spezifizierter Software (Algorithmen und Datenstrukturen)

8. Von der formalen Spezifikation zum (interpretierten) Prototyp

Modulziele:

Die Studierenden lernen formale Software-Modelle lesen, verstehen und kritisieren, um formale Methoden als ein Kommunikationsmittel der Teammitglieder eines SW-Entwicklungsteams schätzen zu lernen. Sie entwickeln mit Hilfe der formalen Spezifikation Teilsystheme von realistischen Softwaremodellen selbst.

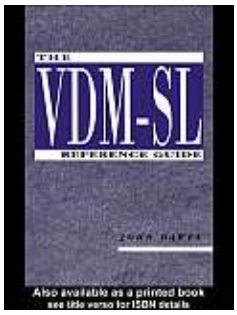
Buchempfehlungen:

Wolfgang Zuser
Software Engineering
mit UML und dem Unified Process

Sprache: Deutsch
Gebundene Ausgabe - 464 Seiten
Pearson Studium
Erscheinungsdatum: Juni 2004

Auflage: 2., überarb. Aufl.
ISBN: 3827370906

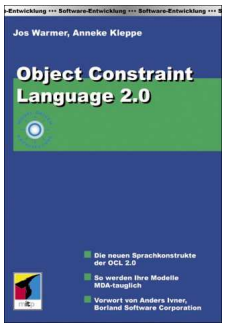
John Dawes
The VDM-SL Reference Guide



Sprache: Englisch
Taschenbuch - 200 Seiten
Routledge,an imprint of Taylor & Francis Books Ltd
Erscheinungsdatum: 12. August 1991

ISBN: 0273031511

Jos Warmer
Object Constraint Language2.0



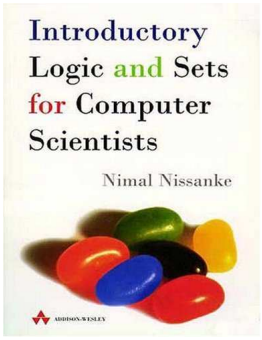
Sprache: Deutsch
Broschiert - 240 Seiten
Mitp-Verlag
Erscheinungsdatum: März 2004

Auflage: 1
ISBN: 3826614453

Nimal Nissanke
Introductory Logic and Sets for Computer Scientists.

Sprache: Englisch Broschiert - 400 Seiten
Addison Wesley
Erscheinungsdatum: Oktober 1998

ISBN: 0201179571



FOLDOC - Free-On-Line-Dictionary-Of-Computing

http://wombat.doc.ic.ac.uk/foldoc/

Eine Suche bei FOLDOC zu „formal methods“ beziehungsweise „specifikation“ ergibt folgendes:

Formale Methoden / formal methods
<Mathematik, Spezifikation> Mathematisch basierte Technick zur Spezifikation, Entwicklung und Verifikation von Software- und Harware-Systemen.

Spezifikation / spezifikation
<Jargon> Ein Dokument, das beschreibt, wie ein System arbeiten soll.

- **Formale Methoden**

  sind logikbasierte Technicken für die
  Spezifikation
  Entwicklung und
  Verifikation
  von SW- und Hardwaresystemen.

- Die **Spezifikation** eines Systems ist ein Dokument, das beschreibt, wie das System arbeiten soll.

- **Beispiele für einfachste Systeme:**

  a) Eine Funktion kann **implizit** (durch Angabe von Eigenschaften) spezifiziert werden:

  $$\begin{array}{l} max(s : \mathbb{N}_1\text{-set})m : \mathbb{N}_1 \\ \text{pre card } s \neq 0 \\ \text{post } m \in s \wedge \forall x \in s \cdot m \geq x \end{array}$$

  b) Eine Funktion kann **explizit** (durch Angabe eines Algorithmus) spezifiziert werden:

  $max : \mathbb{N}_1\text{-set} \rightarrow \mathbb{N}_1$
  $max(s)\underline{\triangle}$
  (dcl $maxBisher : \mathbb{N}_1 := getFirstElement(s)$;
  while $existsNextElement(s)$
  do let $n = getNextElement(s)$ in
  if $n > maxBisher$ then $maxBisher := n$;

  return $maxBisher$
)
  pre card $s \neq 0$

- **Vor-/Nachbedingungen**

```
subSequence(lower:Integer, upper:Integer):Sequence(T)
pre: 1 <= lower
pre: lower <= upper
pre: upper <= self→size()
post: result→size() = upper - lower + 1
post: self{lower..upper}→forAll( index : Integer |
            result→at(index - lower + 1)=
                        self→at(index))
```

- **Beispiel zur mangelhaften Spezifikation:**

  Euro-Panne bei der Deutschen Bank 24 (Update)

  Geldautomaten der Deutschen Bank 24 müssen sich wohl an den Euro erst noch gewöhnen. Wer Anfang Januar Euro-Beträge von Geldautomaten dieser Bank bezogen hat, durfte sich am heutigen Freitag wundern, dass ihm die Bank das 1,95-fache vom Konto abgebucht hat. Offensichtlich haben die Bank-Computer an Stelle der maßgeblichen Euro-Summe irrtümlich mit dem Zahlenwert des umgerechneten DM-Betrags gerechnet.

  Verunsicherte Kunden erfuhren zunächst nur, dass sogar die Angestellten der Bank dem Problem zum Opfer gefallen sind. Mit der Hoffnung auf hilfreichere Informationen mussten sie sich jedoch vorerst gedulden. Erst gegen elf Uhr konnten die Ansprechpartner an der Telefonhotline für etwas Beruhigung sorgen: "Das Problem ist bekannt, die falschen Buchungen werden automatisch zurückgezogen und korrigiert."

  Inzwischen fand die Bank heraus, dass bei einem nächtlichen Datenverarbeitungs-Lauf einige Tausend der insgesamt etwa 1,5 Millionen angefallenenen Kontobewegungen durch einen Programmfehler falsch bearbeitet worden sind. Theoretisch hätten zwar auch herkömmliche Barabhebungen am Bankschalter betroffen sein können, doch zufällig drehte es sich bei den fehlerhaften Buchungen tatsächlich nur um Abhebungen von Geldautomaten, hieß es bei der Deutschen Bank 24. Das erklärt auch, warum bei anderen Banken, die gebührenfreies Abheben von denselben Geldautomaten wie die Deutsche Bank 24 ermöglichen, keine vergleichbaren Fehler aufgetreten sind.

  Markus Block, Sprecher der Deutschen Bank 24, erklärte gegenüber heise online, alle falschen Buchungen würden bis zum Samstag korrigiert sein, sodass kein Kunde finanzielle Nachteile zu erwarten habe. (hps/c't)

  Link zu diesem Artikel bei heise-online:
  http://www.heise.de/newsticker/meldung/23747