

---

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b> .....	1
1.1	Ziele und Inhalte von Band 1 .....	2
1.2	Ergänzende Ziele dieses Buchs .....	4
1.3	Überblick .....	6
1.4	Notationelle Konventionen .....	7
<b>2</b>	<b>Kompakte Übersicht zur UML/P</b> .....	9
2.1	Klassendiagramme .....	10
2.1.1	Klassen und Vererbung .....	10
2.1.2	Assoziationen .....	11
2.1.3	Repräsentation und Stereotypen .....	14
2.2	Object Constraint Language .....	15
2.2.1	Übersicht über OCL/P .....	15
2.2.2	Die OCL-Logik .....	18
2.2.3	Container-Datenstrukturen .....	19
2.2.4	Funktionen in OCL .....	26
2.3	Objektdiagramme .....	28
2.3.1	Einführung in Objektdiagramme .....	29
2.3.2	Komposition .....	30
2.3.3	Bedeutung eines Objektdiagramms .....	31
2.3.4	Logik der Objektdiagramme .....	32
2.4	Statecharts .....	33
2.4.1	Eigenschaften von Statecharts .....	33
2.4.2	Darstellung von Statecharts .....	37
2.5	Sequenzdiagramme .....	44
<b>3</b>	<b>Prinzipien der Codegenerierung</b> .....	49
3.1	Konzepte der Codegenerierung .....	52
3.1.1	Konstruktive Interpretation von Modellen .....	54
3.1.2	Tests versus Implementierung .....	56
3.1.3	Tests und Implementierung aus dem gleichen Modell ..	59

3.2	Techniken der Codegenerierung .....	60
3.2.1	Plattformabhängige Codegenerierung .....	60
3.2.2	Funktionalität und Flexibilität .....	63
3.2.3	Steuerung der Codegenerierung .....	67
3.3	Semantik der Codegenerierung .....	68
3.4	Parametrisierung eines Werkzeugs .....	71
3.4.1	Implementierung von Werkzeugen .....	71
3.4.2	Darstellung von Skripttransformationen .....	73
<b>4</b>	<b>Transformationen für die Codegenerierung .....</b>	<b>77</b>
4.1	Übersetzung von Klassendiagrammen .....	78
4.1.1	Attribute .....	78
4.1.2	Methoden .....	82
4.1.3	Assoziationen .....	85
4.1.4	Qualifizierte Assoziation .....	90
4.1.5	Komposition .....	93
4.1.6	Klassen .....	95
4.1.7	Objekterzeugung .....	98
4.2	Übersetzung von Objektdiagrammen .....	102
4.2.1	Konstruktiv eingesetzte Objektdiagramme .....	102
4.2.2	Beispiel einer konstruktiven Codegenerierung .....	104
4.2.3	Als Prädikate eingesetzte Objektdiagramme .....	106
4.2.4	Objektdiagramm beschreibt Strukturmodifikation ...	109
4.2.5	Objektdiagramme und OCL .....	112
4.3	Codegenerierung aus OCL .....	112
4.3.1	OCL-Aussage als Prädikat .....	113
4.3.2	OCL-Logik .....	115
4.3.3	OCL-Typen .....	117
4.3.4	Typen als Extension .....	119
4.3.5	Navigation und Flattening .....	120
4.3.6	Quantoren und Spezialoperatoren .....	121
4.3.7	Methodenspezifikation .....	122
4.3.8	Vererbung von Methodenspezifikationen .....	125
4.4	Ausführung von Statecharts .....	125
4.4.1	Methoden-Statecharts .....	127
4.4.2	Umsetzung der Zustände .....	128
4.4.3	Umsetzung der Transitionen .....	132
4.5	Übersetzung von Sequenzdiagrammen .....	136
4.5.1	Sequenzdiagramm als Testtreiber .....	136
4.5.2	Sequenzdiagramm als Prädikat .....	138
4.6	Zusammenfassung zur Codegenerierung .....	140

- 5 Grundlagen des Testens** ..... 143
  - 5.1 Einführung in die Testproblematik ..... 144
    - 5.1.1 Testbegriffe ..... 145
    - 5.1.2 Ziele der Testaktivität ..... 147
    - 5.1.3 Fehlerkategorien ..... 149
    - 5.1.4 Begriffsbestimmung für Testverfahren ..... 150
    - 5.1.5 Suche geeigneter Testdaten ..... 152
    - 5.1.6 Sprachspezifische Fehlerquellen ..... 153
    - 5.1.7 UML/P als Test- und Implementierungssprache ..... 155
    - 5.1.8 Eine Notation für die Testfalldefinition ..... 158
  - 5.2 Definition von Testfällen ..... 161
    - 5.2.1 Operative Umsetzung eines Testfalls ..... 161
    - 5.2.2 Vergleich der Testergebnisse ..... 163
    - 5.2.3 Werkzeuge JUnit und VUnit ..... 166
  
- 6 Modellbasierte Tests** ..... 171
  - 6.1 Testdaten und Sollergebnis mit Objektdiagrammen ..... 172
  - 6.2 Invarianten als Codeinstrumentierungen ..... 175
  - 6.3 Methodenspezifikationen ..... 177
    - 6.3.1 Methodenspezifikationen als Codeinstrumentierung .. 177
    - 6.3.2 Methodenspezifikationen zur Testfallbestimmung .... 178
    - 6.3.3 Testfalldefinition mit Methodenspezifikationen ..... 181
  - 6.4 Sequenzdiagramme ..... 182
    - 6.4.1 Trigger ..... 183
    - 6.4.2 Vollständigkeit und Matching ..... 185
    - 6.4.3 Nicht-kausale Sequenzdiagramme ..... 186
    - 6.4.4 Mehrere Sequenzdiagramme in einem Test ..... 186
    - 6.4.5 Mehrere Trigger im Sequenzdiagramm ..... 187
    - 6.4.6 Interaktionsmuster ..... 188
  - 6.5 Statecharts ..... 189
    - 6.5.1 Ausführbare Statecharts ..... 190
    - 6.5.2 Statechart als Ablaufbeschreibung ..... 192
    - 6.5.3 Testverfahren für Statecharts ..... 194
    - 6.5.4 Überdeckungsmetriken ..... 196
    - 6.5.5 Transitionstests statt Testsequenzen ..... 199
    - 6.5.6 Weiterführende Ansätze ..... 200
  - 6.6 Zusammenfassung und offene Punkte beim Testen ..... 201
  
- 7 Testmuster im Einsatz** ..... 207
  - 7.1 Dummies ..... 210
    - 7.1.1 Dummies für Schichten der Architektur ..... 211
    - 7.1.2 Dummies mit Gedächtnis ..... 212
    - 7.1.3 Sequenzdiagramm statt Gedächtnis ..... 214
    - 7.1.4 Abfangen von Seiteneffekten ..... 215
  - 7.2 Testbare Programme gestalten ..... 215

7.2.1	Statische Variablen und Methoden	216
7.2.2	Seiteneffekte in Konstruktoren	219
7.2.3	Objekterzeugung	219
7.2.4	Vorgefertigte Frameworks und Komponenten	220
7.3	Behandlung der Zeit	222
7.3.1	Simulation der Zeit im Dummy	224
7.3.2	Variable Zeiteinstellung im Sequenzdiagramm	224
7.3.3	Muster zur Simulation von Zeit	227
7.3.4	Timer	228
7.4	Nebenläufigkeit mit Threads	229
7.4.1	Eigenes Scheduling	230
7.4.2	Sequenzdiagramm als Scheduling-Modell	231
7.4.3	Behandlung von Threads	232
7.4.4	Muster für die Behandlung von Threads	234
7.4.5	Probleme der erzwungenen Sequentialisierung	235
7.5	Verteilung und Kommunikation	237
7.5.1	Simulation der Verteilung	237
7.5.2	Simulation von Singletons	239
7.5.3	OCL-Bedingungen über mehrere Lokationen	241
7.5.4	Kommunikation simuliert verteilter Prozesse	242
7.5.5	Muster für Verteilung und Kommunikation	244
7.6	Zusammenfassung	245
<b>8</b>	<b>Refactoring als Modelltransformation</b>	<b>247</b>
8.1	Einführende Beispiele für Transformationen	248
8.2	Methodik des Refactoring	253
8.2.1	Technische und methodische Voraussetzungen für Refactoring	253
8.2.2	Qualität des Designs	255
8.2.3	Refactoring, Evolution und Wiederverwendung	256
8.3	Theorie der Modelltransformationen	258
8.3.1	Modelltransformationen	258
8.3.2	Semantik einer Modelltransformation	259
8.3.3	Beobachtungsbegriff	266
8.3.4	Transformationsregeln	270
8.3.5	Korrektheit von Transformationsregeln	272
8.3.6	Ansätze der transformationellen Softwareentwicklung	274
<b>9</b>	<b>Refactoring von Modellen</b>	<b>277</b>
9.1	Quellen für UML/P-Refactoring-Regeln	278
9.1.1	Definition und Darstellung von Refactoring-Regeln	280
9.1.2	Refactoring in Java/P	282
9.1.3	Refactoring von Klassendiagrammen	288
9.1.4	Refactoring in der OCL	294
9.1.5	Einführung von Testmustern als Refactoring	296

9.2	Additive Methode für Datenstrukturwechsel.....	299
9.2.1	Vorgehensweise für den Datenstrukturwechsel .....	299
9.2.2	Beispiel: Darstellung von Geldbeträgen .....	302
9.2.3	Beispiel: Einführung des Chairs im Auktionssystem... ..	306
9.3	Zusammenfassung der Refactoring-Techniken .....	314
<b>10</b>	<b>Zusammenfassung und Ausblick .....</b>	<b>317</b>
	<b>Literatur .....</b>	<b>321</b>
	<b>Index .....</b>	<b>331</b>