
Inhaltsverzeichnis

1	Einführung	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
2	Agile und UML-basierte Methodik	9
2.1	Das Portfolio der Softwaretechnik	11
2.2	Extreme Programming (XP)	13
2.3	Ausgesuchte Entwicklungspraktiken	20
2.3.1	Pair Programming	20
2.3.2	Test-First-Ansatz	21
2.3.3	Refactoring	24
2.4	Agile UML-basierte Vorgehensweise	25
2.5	Zusammenfassung	32
3	Kompakte Übersicht zur UML/P	33
3.1	Klassendiagramme	34
3.1.1	Klassen und Vererbung	34
3.1.2	Assoziationen	35
3.1.3	Repräsentation und Stereotypen	38
3.2	Object Constraint Language	38
3.2.1	OCL/P-Übersicht	39
3.2.2	Die OCL-Logik	42
3.2.3	Container-Datenstrukturen	43
3.2.4	Funktionen in OCL	50
3.3	Objektdiagramme	52
3.3.1	Einführung in Objektdiagramme	52
3.3.2	Komposition	55
3.3.3	Bedeutung eines Objektdiagramms	55
3.3.4	Logik der Objektdiagramme	56

3.4	Statecharts	57
3.4.1	Eigenschaften von Statecharts	57
3.4.2	Darstellung von Statecharts	61
3.5	Sequenzdiagramme	67
4	Prinzipien der Codegenerierung	73
4.1	Konzepte der Codegenerierung	76
4.1.1	Konstruktive Interpretation von Modellen	78
4.1.2	Tests versus Implementierung	80
4.1.3	Tests und Implementierung aus dem gleichen Modell ..	83
4.2	Techniken der Codegenerierung	85
4.2.1	Plattformabhängige Codegenerierung	85
4.2.2	Funktionalität und Flexibilität	88
4.2.3	Steuerung der Codegenerierung	91
4.3	Semantik der Codegenerierung	92
4.4	Flexible Parametrisierung eines Codegenerators	94
4.4.1	Implementierung von Werkzeugen	95
4.4.2	Darstellung von Skripttransformationen	98
5	Transformationen für die Codegenerierung	103
5.1	Übersetzung von Klassendiagrammen	104
5.1.1	Attribute	104
5.1.2	Methoden	108
5.1.3	Assoziationen	111
5.1.4	Qualifizierte Assoziation	116
5.1.5	Komposition	119
5.1.6	Klassen	121
5.1.7	Objekterzeugung	126
5.2	Übersetzung von Objektdiagrammen	129
5.2.1	Konstruktiv eingesetzte Objektdiagramme	129
5.2.2	Beispiel einer konstruktiven Codegenerierung	131
5.2.3	Als Prädikate eingesetzte Objektdiagramme	133
5.2.4	Objektdiagramm beschreibt Strukturmodifikation ...	136
5.2.5	Objektdiagramme und OCL	139
5.3	Codegenerierung aus OCL	139
5.3.1	OCL-Aussage als Prädikat	140
5.3.2	OCL-Logik	142
5.3.3	OCL-Typen	144
5.3.4	Typ als Extension	146
5.3.5	Navigation und Flattening	147
5.3.6	Quantoren und Spezialoperatoren	148
5.3.7	Methodenspezifikation	149
5.3.8	Vererbung von Methodenspezifikationen	152
5.4	Ausführung von Statecharts	152
5.4.1	Methoden-Statecharts	154

5.4.2	Umsetzung der Zustände	155
5.4.3	Umsetzung der Transitionen	160
5.5	Übersetzung von Sequenzdiagrammen	163
5.5.1	Sequenzdiagramm als Testtreiber	164
5.5.2	Sequenzdiagramm als Prädikat	165
5.6	Zusammenfassung zur Codegenerierung	168
6	Grundlagen des Testens	171
6.1	Einführung in die Testproblematik	173
6.1.1	Testbegriffe	173
6.1.2	Ziele der Testaktivitäten	174
6.1.3	Fehlerkategorien	177
6.1.4	Begriffsbestimmung für Testverfahren	178
6.1.5	Suche geeigneter Testdaten	179
6.1.6	Sprachspezifische Fehlerquellen	180
6.1.7	UML/P als Test- und Implementierungssprache	182
6.1.8	Eine Notation für die Testfalldefinition	186
6.2	Definition von Testfällen	188
6.2.1	Operative Umsetzung eines Testfalls	188
6.2.2	Vergleich der Testergebnisse	190
6.2.3	Werkzeug JUnit	193
7	Modellbasierte Tests	197
7.1	Testdaten und Sollergebnis mit Objektdiagrammen	198
7.2	Invarianten als Codeinstrumentierungen	201
7.3	Methodenspezifikationen	203
7.3.1	Methodenspezifikationen als Codeinstrumentierung ..	203
7.3.2	Methodenspezifikationen zur Testfallbestimmung	204
7.3.3	Testfalldefinition mit Methodenspezifikationen	207
7.4	Sequenzdiagramme	208
7.4.1	Trigger	209
7.4.2	Vollständigkeit und Matching	211
7.4.3	Nicht-kausale Sequenzdiagramme	212
7.4.4	Mehrere Sequenzdiagramme in einem Test	212
7.4.5	Mehrere Trigger im Sequenzdiagramm	213
7.4.6	Interaktionsmuster	214
7.5	Statecharts	215
7.5.1	Ausführbare Statecharts	216
7.5.2	Statechart als Ablaufbeschreibung	217
7.5.3	Testverfahren für Statecharts	220
7.5.4	Überdeckungsmetriken	222
7.5.5	Transitionstests statt Testsequenzen	225
7.5.6	Weiterführende Ansätze	226
7.6	Zusammenfassung und offene Punkte beim Testen	227

8	Testmuster im Einsatz	233
8.1	Dummies	236
8.1.1	Dummies für Schichten der Architektur	237
8.1.2	Dummies mit Gedächtnis	238
8.1.3	Sequenzdiagramm statt Gedächtnis	240
8.1.4	Abfangen von Seiteneffekten	241
8.2	Testbare Programme gestalten	241
8.2.1	Statische Variablen und Methoden	242
8.2.2	Seiteneffekte in Konstruktoren	245
8.2.3	Objekterzeugung	245
8.2.4	Vorgefertigte Frameworks und Komponenten	246
8.3	Behandlung der Zeit	249
8.3.1	Simulation der Zeit im Dummy	250
8.3.2	Variable Zeiteinstellung im Sequenzdiagramm	250
8.3.3	Muster zur Simulation von Zeit	253
8.3.4	Timer	254
8.4	Nebenläufigkeit mit Threads	255
8.4.1	Eigenes Scheduling	256
8.4.2	Sequenzdiagramm als Scheduling-Modell	257
8.4.3	Behandlung von Threads	258
8.4.4	Muster für die Behandlung von Threads	260
8.4.5	Probleme der erzwungenen Sequentialisierung	261
8.5	Verteilung und Kommunikation	263
8.5.1	Simulation der Verteilung	263
8.5.2	Simulation von Singletons	265
8.5.3	OCL-Bedingungen über mehrere Lokationen	267
8.5.4	Kommunikation simuliert verteilte Prozesse	268
8.5.5	Muster für Verteilung und Kommunikation	270
8.6	Zusammenfassung	271
9	Refactoring als Modelltransformation	273
9.1	Einführende Beispiele für Transformationen	274
9.2	Methodik des Refactoring	280
9.2.1	Technische und methodische Voraussetzungen für Refactoring	280
9.2.2	Qualität des Designs	281
9.2.3	Refactoring, Evolution und Wiederverwendung	283
9.3	Modelltransformationen	284
9.3.1	Formen von Modelltransformationen	284
9.3.2	Semantik einer Modelltransformation	285
9.3.3	Beobachtungsbegriff	292
9.3.4	Transformationsregeln	297
9.3.5	Korrektheit von Transformationsregeln	298
9.3.6	Ansätze der transformationellen Softwareentwicklung	300
9.3.7	Transformations Sprachen	303

10 Refactoring von Modellen	305
10.1 Quellen für UML/P-Refactoring-Regeln	306
10.1.1 Definition und Darstellung von Refactoring-Regeln ...	308
10.1.2 Refactoring in Java/P	310
10.1.3 Refactoring von Klassendiagrammen	316
10.1.4 Refactoring in der OCL	322
10.1.5 Einführung von Testmustern als Refactoring.....	324
10.2 Additive Methode für Datenstrukturwechsel	328
10.2.1 Vorgehensweise für den Datenstrukturwechsel	328
10.2.2 Beispiel: Darstellung von Geldbeträgen	331
10.2.3 Beispiel: Einführung des Chairs im Auktionssystem ...	335
10.3 Zusammenfassung der Refactoring-Techniken	343
11 Zusammenfassung und Ausblick	347
Literatur	353
Index	369