

Inhaltsverzeichnis

1 Schnittstellen von Programmierung und Modellierung	1
1.1 Papyrus-Framework zur Modellierung mit UML	3
1.1.1 UML-Aktivitätsdiagramme mit Eclipse-Papyrus	4
1.1.2 Zustandsdiagramme mit Eclipse-Papyrus	4
1.1.3 Das Erstellen von Sequenzdiagrammen mit Eclipse-Papyrus	5
1.1.4 Das Erstellen von Anwendungsfalldiagrammen mit Eclipse-Papyrus	7
1.2 Obeo UML-Designer	8
1.2.1 Visualisieren der Diagramme auf Basis von UML 2.5	8
1.2.2 Überblicke über UML-Diagramme mit Eclipse UML-Designer	13
1.2.2.1 Strukturelle Diagramme	16
1.2.2.2 Verhaltensbasierte Diagramme	16
1.2.3 Beispiele von UML-Diagrammen	17
1.2.3.1 Klassendiagramme für Stromversorgungstester	17
1.2.3.2 Komponentendiagramme für die Websystemqualität	18
1.2.3.3 Zustandsdiagramme für die Websystemqualität	19
1.2.3.4 Profildiagramme für Parallelisierungsprozesse für den Asynchronmotor	20
1.2.3.5 Verteilungsdiagramme oder Deployment-Diagramm	21
1.3 Überblick über die Erweiterung der UML-Struktur mit SyML	22
1.3.1 Blockdefinitionsdiagramme	23
1.3.2 Interne Blockdiagramme	24
1.3.3 Anforderungsdiagramme	25
1.3.4 Zusicherungsdiagramme	27
1.4 IT-Lösungen mit Programming4Modeling	28
1.4.1 Modellierung von IT-Lösungen mit Java	29
1.4.1.1 Anwendungen der Funktionalitäten der Elektronik in der Modellierung.	29

1.4.1.2	Modellierungsaspekte mithilfe der Programmierung	42
1.4.1.2.1	Aktivitätsdiagramm	42
1.4.1.2.2	Sequenzdiagramm	47
1.4.1.2.3	Kommunikationsdiagramm	47
1.4.1.2.4	Zustandsdiagramm	48
1.4.1.2.5	Inneres Klassendiagramm	50
1.4.2	Modellierungen von Java-Anwendungen mithilfe von UML	88
1.4.2.1	Modellierung von Anwendungen für den Einsatz von Resonanzelementen Kondensator und Spule mit Eclipse UML-Designer	88
1.4.2.2	Anwendung der Programmierperformance in der Berechnung der Kenngröße Wirkungsgrad des Motors.	93
1.4.2.3	Modellierung der Vermeidung der Kollision zwischen gleichnamigen Methoden aus zwei unterschiedlichen Interfaces	97
1.4.3	Von Codes zu Modellen	103
1.4.3.1	Java für die funktionelle Modellierung	103
1.4.3.1.1	Lambda-Ausdrücke zur Modellierung der Berechnungen	103
1.4.3.1.2	Modellierung der funktionellen Berechnung mithilfe der konstanten Eingaben.	108
1.4.3.2	Java für die Modellierung von Collections	111
1.4.3.2.1	Modellierung der Einfügung gezielter Elemente im Container.	111
1.4.3.2.2	Parallelisierung zum Modellieren der Funktionalität des Wirkungsgrades	117
1.4.3.2.3	Modellierung vom Algorithmus zum Berechnen der Drehzahl des Motors	122
1.4.3.2.4	Anwendung der Klasse <code>javax.swing.JFrame</code> in der Berechnung der Drehzahl des Asynchronmotors.	125
1.4.3.2.5	Anwendung der Modellierung der Parallelisierung von Vektoren in der Berechnung der Drehzahl des Asynchronmotors.	130
1.4.3.2.6	Modellierung der Funktionalität der Strukturen der WLAN-Systeme.	135
1.4.3.2.7	Anwendung der Klassen <code>java.util.Map</code> und der Lambda-Ausdrücke in der Analyse der Elemente von Sammlungen	139

1.4.3.2.8	Modellierung der Charakterisierung des Motors mithilfe von Vektoren	144
1.4.3.2.9	Modellierung der Anwendung der switch-Ausdrücke in der Drehzahlsteuerung des Asynchronmotors.	147
1.4.3.2.10	Modellierung der Funktionalität des Asynchronmotors mithilfe des Wirkungsgrades und des Schlupfs	150
1.4.3.2.11	Modellierung der Implementierung des Einordnens der Elemente in einer Sammlung mithilfe der Klasse <i>java.util.TreeSet</i> und Implementierung der Schnittstellen von <i>Collection</i> und <i>Stream</i>	153
1.4.3.2.12	Anwendungen der Record-Klasse und der switch-Ausdrücke in der Charakterisierung des Asynchronmotors.	163
1.5	Software-Architektur mit Papyrus und UML-Designer.	166
1.5.1	Modellierung eines Klassendiagramms mithilfe vom Open Source Eclipse-Papyrus zum Analysieren eines Testprogramms mit JUnit.	167
1.5.1.1	Modellierung eines Testsystems für die Energietools.	167
1.5.1.2	Modellierung eines Testsystems für WLAN-Systeme	170
1.5.2	Modellierung des Klassendiagramms zum Beschreiben der parametrisierten Systeme mit Eclipse-Ecore-Framework	172
1.5.3	Modellierungen der parallelen Implementierungen von Interfaces.	180
1.5.4	Modellierung der Funktionalitäten der Pattern-Methoden mithilfe des Klassendiagramms von Eclipse UML-Designer	191
1.5.5	Modellierung der Anwendungen des Interface <i>Collection</i> mit dem Klassendiagramm von Eclipse UML-Designer	195
1.6	Zusammenfassung	198
	Literatur.	199
2	UML-Modellierung mit der Eclipse-Umgebung	201
2.1	Modellierung des Klassendiagramms mit Obeo-UML-Designer	201
2.1.1	Vererbung.	206
2.1.2	Eigenschaften der Klassen.	207
2.1.3	Modellierung des Klassendiagramms mithilfe der Operationen	209
2.1.4	Praxis-Beispiel: Anwendung des Klassendiagramms in der Modellierung des Durchlassverhaltens des Transistors	211

2.2	Zustandsdiagramm von Obeo-UML-Designer	236
2.2.1	Überblick über Erstellungstools des Zustandsdiagramms	237
2.2.2	Notationselemente	237
2.2.3	Anwendung des Zustandsdiagramms in der Energietechnik	238
2.3	Komponentendiagramm.	239
2.3.1	Komponentenmodell von Jakarta EE.	239
2.3.2	Komponenten für Java EE	240
2.3.3	Komponenten für JSF, JPA und CDI	241
2.3.3.1	JavaServer Faces (JSF)	242
2.3.3.2	Java Persistence API (JPA)	242
2.3.3.3	Contexts and Dependency Injection(CDI)	243
2.4	Verteilungsdiagramm (Deployment-Diagramm)	244
2.4.1	Device für <i>Application Server Jakarta EE</i>	245
2.4.2	Device <i>Client</i>	246
2.4.3	Device MySQL Datenbank Server.	246
2.5	Zusammenfassung	246
	Literatur.	247
3	Eclipse-Papyrus-Framework	249
3.1	Erstellung eines UML-Klassendiagramms	249
3.1.1	Struktur des UML-Klassendiagramms.	254
3.1.2	Beispiel: Inneres Klassendiagramm.	256
3.1.2.1	Überblick über Assoziationen	258
3.1.2.2	Überblick über Generalisierung	258
3.1.2.3	Vererbungskaskade	259
3.2	Paketdiagramm.	264
3.2.1	Paketdiagramm mit dem Design Pattern Model View Controller	268
3.2.2	Überblick über Java-Codes	272
3.3	Class Tree Table	273
3.3.1	Struktur der Tabelle	273
3.3.2	Vertikale Position	277
3.3.3	Horizontale Position.	278
3.4	Sequenzdiagramme mit Eclipse-Papyrus.	283
3.5	Kommunikationsdiagramm mit Eclipse-Papyrus	290
3.6	Objektdiagramme mit Eclipse-Papyrus	296
3.6.1	Erstellen eines Klassendiagramms mit Eclipse-Papyrus	297
3.6.2	Erstellen eines Objektdiagramms mit Eclipse-Papyrus	302
3.6.2.1	Elemente des Objektdiagramms	323
3.6.2.2	Grafische Darstellung vom Objektdiagramm	323
3.7	Kompositionsstrukturdiagramm.	324
3.7.1	Komposition.	324
3.7.2	Klassifikatoren	337

3.8	Komponentendiagramm mit Eclipse-Papyrus	337
3.8.1	Praxis-Beispiel: Abhängigkeit zwischen Komponenten und Interface	338
3.8.2	Kapselung von Zustand und Verhalten.	346
3.9	Zusammenfassung	347
	Literatur.	351
4	Parallele Modellierung mit UML	353
4.1	Modellierung mit Zustandsdiagrammen	353
4.1.1	Horizontale Modellierung	354
4.1.2	Vertikale Modellierung	355
4.2	Modellierung mit Aktivitätsdiagrammen.	357
4.2.1	Vertikale Integration.	364
4.2.2	Horizontale Integration	366
4.3	Modellierung mit Klassendiagrammen	368
4.3.1	Vererbungshierarchie	368
4.3.2	Modellieren der Strukturen der Klassen	369
4.3.3	Parallelisierung der objektorientierten Modellierung	371
4.4	Modellierung mit Sequenzdiagrammen.	371
4.4.1	Darstellung der Parallelisierungsprozesse mit Objekten oder Lebenslinien.	373
4.4.2	Darstellung der Parallelisierungsprozesse mit Interaktionen.	375
4.5	Zusammenfassung	376
	Literatur.	377
5	Vom Programmieren zum Modellieren.	379
5.1	Anwendung von Java Swing in der Entwicklung der grafischen Oberfläche	380
5.2	Design Pattern Interface.	403
5.2.1	Interfaces I	403
5.2.2	Interface Pattern II	408
5.2.3	Abstract Pattern	417
5.2.4	Delegate Pattern	443
5.2.5	Factory Pattern	448
5.3	Java-Core	466
5.3.1	Modellierung der Leistung.	466
5.3.2	Modellierung der Linearität.	467
5.3.3	Modellierung der Funktionalität einer <i>Fassade</i> -Klasse zur Implementierung der Kapselung einer <i>Controller</i> -Klasse.	490
5.4	Zusammenfassung	502
	Literatur.	503
	Stichwortverzeichnis.	505