

Inhaltsverzeichnis

Abkürzungsverzeichnis	8
Kurzfassung	11
Abstract	12
1. Einleitung	13
1.1. Motivation	13
1.2. Methoden und Werkzeuge zum Testen	14
1.3. Beitrag	17
1.3.1. Beitrag durch das Framework	18
1.3.2. Beitrag durch die methodische Testsequenzerstellung	19
1.3.3. Beitrag durch die formale Testbeschreibung	20
1.3.4. Beitrag durch die Adaption an die Plattformen	20
1.3.5. Erwarteter Nutzen	21
1.4. Aufbau der Arbeit	21
2. Prozesse, Methoden und Werkzeuge	22
2.1. Aufgaben und Aufbau der Software	22
2.2. Softwareentwicklungs- und Testprozess	23
2.3. Entwicklungsparadigmen	27
2.3.1. Steuerungs- und regelungstechnische Aspekte	27
2.3.2. Kommunikation	28
2.3.3. Berücksichtigung der Hardware	28
2.3.4. Programmierung	29
2.4. Modelle in der Softwareentwicklung	30
2.4.1. Unified Modeling Language	31
2.4.1.1. Klassendiagramm	31
2.4.1.2. Zustandsdiagramm	32
2.4.2. Extensible Markup Language	33
2.4.3. Generative Programmierung	34
2.4.3.1. Model Driven Architecture	34

2.4.3.2.	Model Driven Software Development	35
2.4.3.3.	Nutzen der Vorgehensweise	37
2.4.4.	Relevanz für den Test	38
2.5.	Potenzielle Softwarefehler	39
2.6.	Testentwurfsmethoden	40
2.6.1.	Statische Testentwurfsmethoden	41
2.6.2.	Dynamische Testentwurfsmethoden	42
2.6.3.	White-box Testentwurfsmethoden	43
2.6.3.1.	Kontrollflussbezogene Kriterien	43
2.6.3.2.	Datenflussbezogene Kriterien	46
2.6.3.3.	Bewertung	46
2.6.4.	Black-box Testentwurfsmethoden	48
2.6.4.1.	Äquivalenzklassenbildung	49
2.6.4.2.	Auswahlstrategien	50
2.6.4.3.	Zustandsbasierter Test	58
2.6.4.4.	Erfahrungsbasiertes Testen	58
2.7.	Aufgabenstellung	58
3.	Testframework	61
3.1.	Artefakte aus dem Entwicklungsprozess	61
3.1.1.	Strukturartefakte	61
3.1.2.	Verhalten	62
3.1.3.	Daten	63
3.2.	Technologische Realisierung	63
3.2.1.	Das <i>Testspezifikations</i> -Meta-Modell	65
3.2.2.	Das <i>Testbericht</i> Meta-Modell	68
3.2.3.	Codegenerierung	70
3.2.4.	Zusammenfassung	71
4.	Leitanwendung	73
4.1.	Physikalische Grundlagen	73
4.2.	Software-Design und Implementierung	76
4.2.1.	Klassendiagramm	76
4.2.2.	Zustandsdiagramm	78
4.3.	Testausführungsplattform	79
4.4.	Aufgaben für die Testimplementierung	80

5. Heuristik zur modellbasierten Testsequenzgenerierung	81
5.1. Prinzip der Testsequenzgenerierung aus Zustandsmaschinen .	82
5.1.1. Generierungsstrategien	83
5.1.2. Auffindbare Fehler	83
5.1.3. Auswahl einer Strategie zur Testsequenzgenerierung .	84
5.1.4. Umgang mit erweiterten endlichen Zustandsmaschinen	85
5.1.4.1. Auflösen von Hierarchie	86
5.1.4.2. Auflösen von Orthogonalität	86
5.2. Integration in das Framework	88
5.2.1. Allgemeine Schnittstelle der Heuristik	88
5.2.2. Verfügbare Artefakte der Leitanwendung	92
5.2.3. Konzeption des <i>Exchange</i> -Meta-Modells	93
5.2.4. Instanziierung der Leitanwendung	94
5.3. Heuristik	96
5.3.1. Bedutung	97
5.3.2. Berechnung der Initialisierungs-Testsequenz	100
5.3.3. Berechnung der Transitionsüberdeckung	102
5.3.4. Vervollständigung der Transitionsfolgen	105
5.3.4.1. Umgang mit Superzuständen	106
5.3.4.2. Umgang mit orthogonalen Zuständen	109
5.3.5. Zusammenfassung der strukturellen Aspekte	114
5.3.6. Synthese von Bedutung und Struktur	114
5.4. Überführen der Testspezifikation in ausführbaren Code . . .	119
5.5. Anwendung	121
6. Einsatz des Testframeworks	124
6.1. Methoden und deren Einbindung	125
6.2. Formale Testspezifikation und Transformation	128
6.3. Nutzen und Potenziale	130
7. Zusammenfassung und Ausblick	131
A. Algorithmen zur Testsequenzinitialisierung	134
A.1. Dijkstra-Algorithmus	136
B. Entwicklung des Algorithmus zur Transitionsüberdeckung	138

Abbildungsverzeichnis	141
Tabellenverzeichnis	142
Literaturverzeichnis	144