

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Ziel der Arbeit . . . . .	2
1.2	Übersicht über die Arbeit . . . . .	2
<b>2</b>	<b>Elektronische Systeme im Fahrzeug</b>	<b>5</b>
2.1	Architekturüberblick . . . . .	5
2.2	Telematiksystem im Fahrzeug . . . . .	7
2.2.1	Zentrales Telematik-Steuergerät ( <i>Head-Unit</i> ) . . . . .	7
2.2.2	Anzeigesysteme . . . . .	8
2.2.3	Audiosteuergeräte . . . . .	10
2.2.4	Videosteuergeräte . . . . .	10
2.3	Abgrenzung zu anderen Domänen . . . . .	10
2.4	Bussysteme und Schnittstellen . . . . .	11
2.4.1	CAN . . . . .	11
2.4.2	MOST . . . . .	12
2.4.3	FlexRay . . . . .	17
2.4.4	LIN . . . . .	18
2.4.5	Weitere Schnittstellen . . . . .	18
2.5	Anwendungsbeispiel . . . . .	19
2.6	Fazit . . . . .	19
<b>3</b>	<b>Entwicklung und Test im Telematikbereich</b>	<b>21</b>
3.1	Entwicklungsprozesse . . . . .	21
3.1.1	Allgemeines V-Modell . . . . .	21
3.1.2	CMMI . . . . .	24
3.2	Softwaretest . . . . .	25
3.2.1	Anforderungen und Testgrundlage . . . . .	27
3.3	Testverfahren . . . . .	29
3.3.1	Manuelle Tests . . . . .	29
3.3.2	Automatisierte Tests . . . . .	29

3.4	Testbeschreibungssprachen für automatisierte Tests . . . . .	29
3.4.1	TTCN-3 . . . . .	30
3.4.2	UML2 Testing Profile . . . . .	33
3.4.3	Vector Informatik XML Testbeschreibung . . . . .	33
3.4.4	Bewertung der Testbeschreibungen . . . . .	35
3.5	Testumgebung . . . . .	36
3.6	Problematik der heutigen Testpraxis . . . . .	37
3.6.1	Geringe Testabdeckung durch informelle Spezifikationen . . . . .	38
3.6.2	Schlechte Wiederverwendbarkeit der Testfälle . . . . .	38
3.7	Anforderungen . . . . .	39
3.7.1	Übergreifende Anforderungen . . . . .	39
3.7.2	Modellanforderungen . . . . .	40
3.7.3	Testanforderungen . . . . .	41
3.8	Fazit . . . . .	42
<b>4</b>	<b>Modellierung und Modell-basiertes Testen</b>	<b>43</b>
4.1	Modell-basierte Entwicklung . . . . .	43
4.2	Modell-basierter Test . . . . .	45
4.3	Modellgetriebene Architektur (MDA) . . . . .	45
4.4	Metamodellierung . . . . .	47
4.5	Meta Object Facility (MOF) . . . . .	47
4.6	Modelltransformation . . . . .	49
4.6.1	Modelltransformationssprachen . . . . .	49
4.6.2	Query View Transformation (QVT) . . . . .	50
4.6.3	Atlas Transformation Language . . . . .	53
4.7	Modellierungssprachen . . . . .	53
4.7.1	Statecharts . . . . .	54
4.7.2	Specification and Description Language (SDL) . . . . .	55
4.7.3	Message Sequence Charts (MSC) . . . . .	57
4.7.4	Petri-Netze . . . . .	59
4.7.5	Matlab/Simulink und Ascet SD . . . . .	60
4.7.6	Unified Modeling Language, Version 2 (UML2) . . . . .	61
4.7.7	Domänenspezifische Modellierung (DSM) . . . . .	64
4.7.8	Systems Modeling Language (SysML) . . . . .	67
4.8	Auswahl der Modellierungssprache . . . . .	70
4.9	Werkzeugauswahl . . . . .	71
4.10	Fazit . . . . .	72

---

<b>5</b>	<b>Konzeption und Lösungsansatz</b>	<b>73</b>
5.1	Rahmenbedingungen	74
5.2	Modelltransformation	74
5.2.1	Erweiterung des Schlüsselkonzeptes von QVT	78
5.2.2	Zuordnung der Domänen für WHEN- und WHERE-Aufrufe	80
5.3	Konzeption des Systemmodells	80
5.3.1	Strukturmodell	81
5.3.2	Verhaltensmodell	81
5.4	Strukturierung des Modells	83
5.4.1	Funktionaler Teil	83
5.4.2	Produktspezifischer Teil	86
5.4.3	Mensch-Maschine-Schnittstelle (HMI)	86
5.5	Modell-basierter Testprozess	88
5.6	Formale Beschreibung der Modellstruktur	89
5.6.1	Funktionaler Modellteil	89
5.6.2	Produktspezifischer Teil	92
5.7	Testdurchführung	95
5.8	Verwandte Arbeiten	95
5.9	Spezifikation der Testeingabedaten	96
5.9.1	Kopplung von CTM und SysML	97
5.10	Fazit	99
<b>6</b>	<b>Generierung funktionaler Testfälle</b>	<b>101</b>
6.1	Überblick	101
6.2	Ansätze zur Testfallgenerierung	101
6.2.1	Zustandsbasierte Ansätze	101
6.2.2	Ereignisbasierte Ansätze	103
6.3	Generierung funktionaler Testfälle	105
6.3.1	Modellierung des Systemverhaltens	105
6.3.2	Simulation des Systemmodells	112
6.3.3	Gewinnung von Testfällen	115
6.4	Überdeckungskriterien	117
6.5	Diskussion des Verfahrens	117
6.5.1	Auswahl der Benutzeraktivitäten zur Simulation	117
6.5.2	Problematik der großen Zustandsmenge	119
6.5.3	Verwendung des Modells als Testorakel	119
6.5.4	Zeitliches Verhalten	120
6.5.5	Strukturiertes Vorgehen bei der Modellerstellung	120
6.6	Fazit	121

---

<b>7</b>	<b>Generierung produktspezifischer Testfälle</b>	<b>123</b>
7.1	Überblick . . . . .	123
7.2	Verwandte Arbeiten . . . . .	123
7.3	Produktspezifische Modellierung . . . . .	124
7.3.1	Verwendung der Funktionskataloge bei der Modellerstellung . . . . .	127
7.3.2	Parameter . . . . .	128
7.4	Parameteranpassung . . . . .	129
7.5	Gewinnung MOST-spezifischer Informationen aus dem Strukturmodell . . . . .	130
7.6	Äquivalentes Verhalten . . . . .	135
7.7	Transformation zu spezifischen SysML-Testfällen . . . . .	136
7.7.1	Informelle Beschreibung . . . . .	136
7.7.2	Formale Beschreibung . . . . .	138
7.8	Erzeugung der CANoe.MOST XML-Testmodule . . . . .	144
7.8.1	SysML zu XML-Transformation mit QVT . . . . .	145
7.8.2	Die QVT-Transformationsregel . . . . .	146
7.9	Fazit . . . . .	149
<b>8</b>	<b>Beispiel MOST Audio System</b>	<b>151</b>
8.1	Systemmodellierung . . . . .	151
8.1.1	Funktionaler Teil . . . . .	151
8.1.2	Produktspezifischer Teil . . . . .	157
8.2	Testfallgenerierung . . . . .	159
8.3	Fazit . . . . .	162
<b>9</b>	<b>Zusammenfassung und Ausblick</b>	<b>163</b>
9.1	Modellierung von zeitlichem Verhalten . . . . .	166
9.2	Nutzung weiterer SysML-Konzepte . . . . .	166
9.3	Varianten . . . . .	166
9.4	Automatisierte Validierung . . . . .	166
9.5	Linguistische Aspekte bei der Modellerstellung . . . . .	167
<b>A</b>	<b>QVT-Regel der SysML-zu-XML Transformation</b>	<b>169</b>
	<b>Literaturverzeichnis</b>	<b>173</b>
	<b>Index</b>	<b>181</b>