Informatik Forschung und Entwicklung manuscript No.
(will be inserted by the editor)

Andreas Bauer, Manfred Broy, Jan Romberg, Bernhard Schätz, Peter Braun, Ulrich Freund, Nuria Mata, Robert Sandner, Pierre Mai, Dirk Ziegenbein

Das AutoMoDe-Projekt

Modellbasierte Entwicklung softwareintensiver Systeme im Automobil

14. August 2006

Zusammenfassung Die Entwicklung eingebetteter Software für Automobile ist inhärent komplex und vereint verschiedene Entwicklungsphasen, mehrere fachliche Disziplinen, sowie verschiedene Akteure in beteiligten Unternehmen. Der AutoMoDe-Ansatz zur Entwicklung automobiler Software beschreibt Systeme auf verschiedenen Abstraktionsebenen und definiert schrittweise Übergänge zwischen diesen Ebenen. Neben der Definition geeigneter Ebenen werden zur Modellierung von Echtzeitsystemen ein einheitliches Berechnungsmodell sowie domänenspezifische Beschreibungstechniken verwendet. Automatisierte Anbindungen für Analyse und Synthese komplexer Softwaresysteme mit dem Ziel eines konsistenzbetonten Entwicklungsprozesses wurden realisiert. Die beschriebenen Techniken wurden in den Werkzeugprototypen AutoFOCUS integriert und im Zusammenspiel mit einer Werkzeugkette demonstriert.

Schlüsselwörter Automotive Software Engineering · Eingebette Software · Synchrone Sprachen · AutoMoDe

Dieses Projekt wurde vom Bundesministerium für Bildung und Forschung (BMBF) unter dem Kennzeichen 01ISC08 gefördert

A. Bauer, M. Broy, J. Romberg, B. Schätz
Software & Systems Engineering, Institut für Informatik
Technische Universität München
Boltzmannstr. 3, D-85748 Garching b. München

P. Braun
Validas AG
Lichtenbergstr. 8, D-85748 Garching b. München

U. Freund, N. Mata
ETAS GmbH
Borsigstr. 14, D-70469 Stuttgart

R. Sandner
BMW AG
D-80788 München

P. Mai
PMSF IT Consulting
Ludwig-Thoma-Str. 11, D-87724 Ottobeuren

D. Ziegenbein
Robert Bosch GmbH
Postfach 30 02 40, D-70442 Stuttgart

Abstract Development of embedded automotive software is inherently complex and involves different stakeholders, phases, and disciplines. The AutoMoDe approach to automotive software development defines distinct levels of abstraction for integrated development, and defines stepwise transitions between the levels. Along with the definition of suitable abstraction levels, to support modeling of real-time systems, a homogeneous operational model along with domain-specific notations are used. Automated backend functionalities for analysis and synthesis of complex software systems, with the goal of a consistent development process, were devised. The techniques described have been integrated into the tool prototype AutoFOCUS and have been demonstrated by the construction of a tool chain.

Keywords Automotive software engineering · embedded software · synchronous languages · AutoMoDe

CR Subject Classification D.2.2

1 Einleitung

Die ständig steigende Anzahl von Funktionalitäten, die von eingebetteten Systemen im Automobilbereich zur Verfügung gestellt werden, sowie die wachsende Tendenz, Funktionalitäten zu interoperierenden Netzwerken zu integrieren, hat die Komplexität softwareintensiver Systeme im Automobil drastisch erhöht. Die vorherrschende Verwendung von etablierten, stark an konkreten Implementierungsmechanismen sowie an Subsystemen orientierten Ansätzen zur Entwicklung eingebetteter Systeme führt zu einem steigenden Problemdruck, unter anderem in Bezug auf Integration und Qualitätssicherung.

Die Komplexität stellt somit hohe Anforderungen an eine Entwicklungsmethodik für Automotive-Software. Zum einen soll die Interoperabilität von Funktionen bereits in frühen Stadien der Entwicklung evaluiert werden können. Hierzu ist eine explizite Modellierung der Abhängigkeiten zwischen Funktionen, der Schnittstellen zwischen Modulen,

sowie eine Festlegung der Verhaltenssemantik notwendig. Des weiteren sollen Funktionsmodule zunächst unabhängig von der physischen Verteilung modelliert werden können, um Freiheitsgrade im Entwurf bezüglich des Mappings von Funktionen auf Steuergeräte zu erhalten. Diese Freiheitsgrade können dann später im Deployment, z. B. spezifisch nach unterschiedlichen Ausstattungsumfängen, kostenoptimal ausgenutzt werden. Schließlich soll durch geeignete methodische Vorgaben und wohldefinierte Schnittstellen- und Moduldefinitionen ein hoher Wieder- und Mehrfachverwendungsgrad von Funktionsmodulen erreicht werden. Über die genannten Punkte hinaus hinaus stellt sich mit dem Zusammenwachsen der Anwendungsgebiete diskreter Ereignissysteme und synchroner zeitgesteuerter Systeme die Aufgabe, einen einheitlichen Modellierungsansatz für beide Arten eingebetteter Software zur Verfügung zu stellen, um beide Aspekte eines eingebetteten Systems in Kombination beschreiben, analysieren, und bis hin zu einer Implementierung entwickeln zu können.

In diesem Artikel werden die Ergebnisse des AutoMoDe-Projektes abschließend vorgestellt, die von den Projektpartnern BMW AG, Robert Bosch GmbH, ETAS GmbH, Technische Universität München, sowie Validas AG im Zeitraum Oktober 2003 bis März 2006 erzielt wurden. Der Ansatz wird unter anderem durch das AutoFOCUS-Werkzeug [7] unterstützt. Konkret bietet AutoFOCUS dem Entwickler eine Anzahl graphischer und textueller Beschreibungstechniken für verschiedene Abstraktionsebenen (siehe Abschnitt 3.1). Damit ist die Modellierung von Struktur und Verhalten von Software durchgängig möglich: angefangen von der Erfassung funktionaler Abhängigkeiten, bis hin zur Verteilung von Applikationen auf Prozesse, Tasks und reale ECUs im Bordnetzverbund des Fahrzeugs. Zusätzlich existieren eine Reihe von Anbindungen für Techniken der Konsistenzanalyse, formalen Verifikation sowie Testfallgenerierung [6], sowie Code-Generatoren für die Programmiersprachen C und Ada.

Um die praktische Umsetzung der Konzepte zu überprüfen wurde eine Werkzeugkette bestehend aus etablierten Werkzeugen der ETAS aufgebaut. Anhand dieser konnte gezeigt werden, dass AutoFOCUS-Modelle in effiziente Implementierungen für eingebettete Hardware umgesetzt werden können.

Aufbau. Abschnitt 2 diskutiert verwandte Arbeiten zu den in AutoMoDe geleisteten Beiträgen. In Abschnitt 3 wird ein kurzer Überblick des AutoMoDe-Ansatzes und der betrachteten Abstraktionsebenen gegeben. Abschnitt 4 erläutert kurz das AutoFOCUS zugrunde liegende Berechnungsmodell, bevor im darauffolgenden Abschnitt 5 die eingesetzten AutoFOCUS-Notationen beschrieben werden. Als wesentlicher Bestandteil von AutoMoDe werden in Abschnitt 6 verschiedene Transformationen im Entwicklungsprozess beschrieben. Das Beispiel der Modellierung einer Antischlupfregelung in Abschnitt 7 dient dazu die Ergebnisse zu veranschaulichen und deren praktische Umsetzung zu überprüfen. Der Übergang zu der Implementierung des Beispiels ist Gegenstand des Abschnitts 8. Der Stand der Werkzeugunterstützung des AutoFOCUS-Werkzeugs wird in Abschnitt 9 kurz diskutiert, bevor dieser Artikel mit einer Zusammenfassung endet.

2 Verwandte Ansätze und Beiträge von AutoMoDe

Der Beitrag von AutoMoDe kann wie folgt in zwei Teilaspekte aufgespalten werden: Zum einen wurde als Ergebnis des Projekts ein *methodisches* Rahmenkonzept für die Entwicklung automobiler Softwaresysteme definiert. Zum anderen wurden *technische* Beiträge in den Bereichen Modellierungssprachen und deren semantischer Fundierung, Transformationssprachen für Software-Modellierungswerkzeuge, sowie verteilte Implementierung synchroner Datenflusssprachen durch das Projekt eingebracht.

Methodischer Beitrag. Es existieren eine Reihe von verwandten Methoden für modellbasierten Entwurf automobiler Softwaresysteme [16, 17, 1, 27]. Neben Unterschieden im Detail verwenden alle zitierten Ansätze eine Anzahl von definierten Abstraktionsebenen sowie verschiedenen unterstützenden Werkzeugen und Notationen, mit dem Ziel, einer inkrementellen Entwurfsmethodik für automobile Software. Durch die Verwendung heterogener Notationen und Werkzeuge im Stadium des Entwurfs gelingt diesen vorangegangenen Ansätzen jedoch typischerweise keine enge Kopplung von Syntax- und Semantikkonzepten über Werkzeug-, Notations-, sowie Phasengrenzen hinweg: diese enge Kopplung ist in einem heterogenen Ansatz naturgemäß schwierig. AutoMoDe verwendet dagegen ein einheitliches und semantisch fundiertes Domänenmodell, das durch das AutoFOCUS-Werkzeug unterstützt wird [7]. Durch die Homogenität des Domänenmodells können neuartige technische Beiträge wie die semantikerhaltende Transformation von Modellen durch Transformationssprachen, eingebracht werden.

Von den genannten Ansätzen stützt sich AutoMoDe speziell auf Vorgängerprojekte Automotive [27] sowie EAST-EEA [1]. Dabei wurden verschiedene Defizite der Vorgänger adressiert: im Vergleich zu Automotive, dass auf der UML 1.x-Notation basierte, wird in AutoMoDe mit der AutoFOCUS-Notation ein explizites Modell für Komponenten und ihre (nachrichtenbasierten) Schnittstellen eingeführt, sowie Unterstützung zur Modellierung regelungstechnischer Algorithmen. AutoFOCUS ist in wesentlichen Konzepten eng verwandt zu der Komponentendarstellung in UML 2.0, so dass die weiterführende Möglichkeit einer UML-standardkonformen Darstellung nicht als kritisch angesehen wird. Als Vorteil gegenüber einer rein standardbezogenen Darstellung können die Arbeiten mit AutoFOCUS je-

doch auf ein unzweideutiges und für die Domäne geeignetes semantisches Modell abgestützt werden. Im Vergleich sowohl mit dem Automotive-Projekt als auch mit EAST-EEA wurde in AutoMoDe ein stärkerer Schwerpunkt auf die Modellierung und Erhaltung von *Verhaltenseigenschaften* eingebetteter Systeme gelegt. Mit dem gleichzeitigen Start der AUTOSAR-Entwicklungspartnerschaft [20] konnte AutoMoDe zudem von den technischen Ergebnissen her auf die „Virtual Functional Bus"-Architektur von AUTOSAR abgestimmt werden.

Technischer Beitrag. Gegenüber regelungstechnisch motivierten Werkzeugen wie Matlab/Simulink [15] oder ASCET [9] sowie aus anderen Anwendungsgebieten (z. B. Telekommunikation) stammenden Ansätzen wie UML 2.0 Komponentendiagrammen [24] bietet der AutoFOCUS-gestützte Ansatz den Vorteil von domänenspezifische Konzepten wie z. B. Betriebsmodus, Funktion, Periode.

In diesem Artikel wird unter anderem die explizite Modellierung von *Betriebsmodi* als ein Mittel der grobgranularen Dekomposition eingebetteter Systeme als Ergebnis des AutoMoDe-Projekts beschrieben. Diese Idee wurde auch von anderen Autoren beschrieben, so z. B. [14]. Zusätzlich zur reinen Verwendung einer neuen Notation setzt unser Ansatz Modi über mehrere Abstraktionsebenen hinweg ein, und untersucht speziell Transformationen zwischen verschiedenen strukturierten, jeweils auf einen Einsatzzweck hin optimierte Modus-Hierarchien.

Die Idee, zeit- und ereignisgesteuerte Takte als Boolsche Ausdrücke (Clocks) zu definieren, sowie ein flankierendes Verfahren zur Inferenz und Überprüfung von Clocks, stammt aus dem Gebiet der synchronen Datenflusssprachen [4]. Das entworfene Inferenzverfahren unterscheidet sich in Details von den bekannten Verfahren, um die Skalierbarkeit des Verfahrens sowie die Anschaulichkeit der inferierten Clocks mit einer ausreichenden Ausdrucksmächtigkeit der Modellierungssprache zu verknüpfen.

3 Überblick

Der AutoMoDe-Ansatz vereint verschiedene Aspekte einer modellbasierten, durchgängigen Entwicklungsmethodik:

Domänenmodell: In AutoMoDe wurde ein integriertes Domänenmodell zur Entwicklung softwareintensiver eingebetteter Systeme im Automobilbereich definiert.

Notationen: Das Domänenmodell integriert Modellierkonzepte, die in der Entwicklersicht durch eine Anzahl von problemorientierten grafischen und textuellen Notationen repräsentiert werden. Dabei bieten unterschiedliche Diagramme, zum Beispiel für Verhalten und Struktur, jeweils alternative Sichten auf ein integriertes Modell.

Abstraktionsebenen und Transformationsschritte: Um den Ansatz insgesamt zu strukturieren, werden verschiedene Abstraktionsebenen eingeführt. Formalisierte Transformationsschritte dienen dem Übergang zwischen Abstraktionsebenen sowie der Umformung von Modellen auf gleicher Ebene mit dem Ziel der Erhaltung wesentlicher Eigenschaften.

Werkzeugkette: Der AutoMoDe-Ansatz wird durch eine Werkzeugkette unterstützt, die Editieren, Analyse, sowie Transformation von Modellen auf verschiedenen Abstraktionsebenen zulässt.

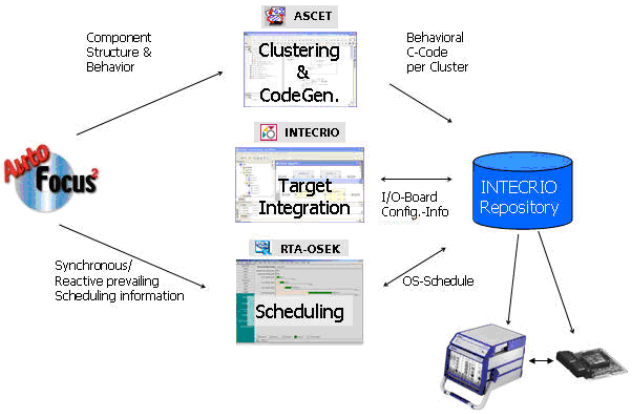

Abb. 1 AutoMoDe-Werkzeugkette

Die AutoMoDe-Entwurfsmethodik basiert stark auf einer einer Gliederung der Modellierungsartefakte in domänenspezifische Abstraktionsebenen. Diese sind an die in [1] vorgeschlagenen angelehnt (siehe Abb. 2). Zur Veranschaulichung des Gesamtansatzes werden für jede Abstraktionsebene jeweils die AutoMoDe-Werkzeuge erwähnt, die primär zur Repräsentation der jeweiligen Entwurfsartefakte eingesetzt werden. Die gesamte AutoMoDe-Werkzeugkette ist in Abb. 1 illustriert (siehe Abschnitt Abschnitt 8). Der Zusammenhang von Werkzeugen zu Abstraktionsebenen ist unkompliziert: Für die abstrakteren Modellierungsebenen wird das AutoFOCUS-Werkzeug verwendet, das die in Abschnitt 4 beschriebenen Notationen für die abstrakteren Ebenen unterstützt. Nebem dem Editieren und Validieren der so entstandenen Modelle unterstützt AutoFOCUS verschiedene Arten von Modelltransformationen, die in Abschnitt 6 näher erläutert werden. In Richtung auf die konkreteren und stärker implementierungsbezogenen Abstraktionsebenen schließt die AutoMoDe-Werkzeugkette die kommerziellen Werkzeuge ASCET [9], RTA OSEK Planner [13], sowie INTECRIO [10] mit ein. Die letztgenannten Werkzeuge zur implementierungsbezogenen Entwicklung und Synthese automobiler Softwaresysteme sind in dieser Domäne praktisch etabliert.

3.1 Abstraktionsebenen

Functional Analysis Architecture (FAA). Die abstrakteste der betrachteten Ebenen ist die Functional Analysis Architecture. Sie ist eine fahrzeugweite und bzgl. der Struktur

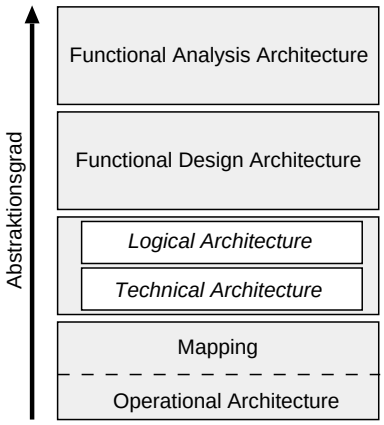

Abb. 2 AutoMoDe-Abstraktionsebenen

und der funktionalen Abhängigkeiten vollständige Beschreibung. Übergreifendes Ziel der Beschreibung auf FAA-Ebene ist vor allem die Identifikation wesentlicher Abhängigkeiten und eventueller Konflikte zwischen Funktionalitäten, sowie die Konzeptvalidierung anhand von unvollständigen oder prototypischen Verhaltensbeschreibungen. Der Begriff der *Funktionalität* bezeichnet eine benutzersichtbare Elementarfunktion auf FAA-Ebene. Die Beschreibung auf FAA-Ebene konzentriert sich dabei auf Kernelemente der eigentlichen Systemanwendung bzw. des Regelalgorithmus: Im Sinne der Anwendung untergeordnete Schichten wie z.B. Sensor- und Aktuatorvorverabeitung, Diagnosefunktionalitäten, sowie Systemüberwachung und Plausibilisierung werden nicht betrachtet. Die folgende Darstellung konzentriert sich zudem auf in Software realisierte Funktionalitäten, während die FAA insgesamt unabhängig von der Realisierungsentscheidung (HW/SW) ist.

Systeme sind auf FAA-Ebene typischerweise nach funktionalen Gesichtspunkten und somit vor allem nach Sichtbarkeit durch den Benutzer gruppiert. Als Beispiel für eine solche Gruppierung kann man unter dem Begriff „Längsdynamiksteuerung“ alle diejenigen Funktionalitäten versammeln, die einen Einfluss auf die (durch den Benutzer wahrgenommene) Längsdynamik eines Fahrzeuges haben, also z. B. die Vortriebs-/Motorsteuerung, die Steuerung der Bremse, sowie die Längsdynamik beeinflussende Funktionen wie eine Traktionskontrolle. Im Vergleich zur FDA-Ebene kann dabei ein und dieselbe Komponente mehrfach in verschiedenen Kontexten ohne weitere Einschränkung vorkommen (als Typ bzw. Referenz).

FAA-Darstellungen sind typischerweise *strukturorientiert*: während die Strukturierung in Funktionalitäten und Abhängigkeiten bzw. Datenflüsse essentiell für die FAA-Modellierung ist, spielt das Verhalten der Einzelfunktionalitäten eine untergeordnete Rolle. Somit wird die FAA in AutoFOCUS primär durch die in Abschnitt 5 beschriebenen Systemstrukturdiagramme (SSDs) repräsentiert, und in untergeordneter Weise durch die AutoFOCUS-Notationen für detailliertes Verhalten. Mit AutoFOCUS können auch vielfältige, FAA-bezogene Konsistenzbedingungen überprüft und durchgesetzt werden: so kann z.B. die Strukturvollständigkeit auf FDA-Ebene als Vollständigkeit der Kommunikationsverbindungen interpretiert und automatisch überprüft werden: jeder eingehende Konnektor muss dabei z. B. mit einem Signal belegt sein. Verhaltensvollständigkeit bezeichnet in AutoFOCUS die Festlegung eines eindeutigen (deterministischen) Verhaltens für jede Komponente.

Functional Design Architecture (FDA). Die *Functional Design Architecture* stellt bereits eine bzgl. der Struktur und des Verhaltens vollständige Beschreibung von Software-Systemen im Automobil dar. Der methodische Schwerpunkt liegt dabei auf einer Verifikation des Softwareverhaltens sowie auf der Identifikation gemeinsamer und wiederverwendbare Softwarekomponenten. Gegenüber der FAA steht eine stärker realisierungsorientierte und weniger nutzerorientierte Strukturierung des Systems im Vordergrund. So kann z. B. die obengenannte nutzerbezogene FAA-Strukturierung von Fahrdynamikfunktionen in „Längsdynamiksteuerung“ und „Querdynamiksteuerung“ auf FDA-Ebene durch eine stärker technische, organisatorische, sowie wiederverwendungsbezogene Strukturierung ersetzt werden.

Bezüglich der Vielfachheit von Komponenten gilt, dass auf der FDA-Ebene eine Minimierung des wiederholten Auftretens einer gegebenen Komponente angestrebt wird. Der methodische Nutzen ist dabei, dass durch die Zusammenfassung identischer Funktionalitäten in einmalig auftretenden Komponenten auf FDA-Ebene Potenziale für Mehrfachverwendung aufgedeckt werden können, die sich beim Übergang von einer funktionsorientierten Strukturierung (FAA) auf eine an der Implementierung orientierten Struktur (LA/TA) ergeben. Ein häufig genanntes Beispiel für solche Potenziale ist z. B. die zunächst häufig mehrfach eingeplante Vorverarbeitung und Aufbereitung von Sensorsignalen, die in der späteren Realisierung an einer Stelle aufbereitet werden sollten, um anschließend verteilt kommuniziert und verwendet zu werden.

FDA-Komponenten werden in AutoFOCUS durch Komponenten in Systemstrukturdiagrammen beschrieben: sie können somit wiederum hierarchisch durch andere Komponenten aufgebaut sein, und sind über typisierte, gerichtete Kanäle und Konnektoren miteinander verbunden. Die Verhaltensdarstellung ist in AutoFOCUS mit den in Abschnitt 5 beschriebenen Notationen zur Verhaltensdarstellung möglich. Die Überprüfung FDA-spezifischer Konsistenzbedingungen wie z. B. Verhaltensvollständigkeit kann ebenfalls durch AutoFOCUS vorgenommen werden.

Logical/Technical Architecture (LA/TA). Die detaillierteste AutoMoDe-Abstraktionsebene gliedert sich in Logical Architecture und Technical Architecture. In der LA werden die Software-Komponenten in sog. Cluster gruppiert: dabei repräsentiert ein Cluster eine kleinste verteilbare Einheit, z. B.

im Sinne einer Task oder einer Routine. Als Strukturierungskriterium treten somit noch stärker als in der FDA technische Eigenschaften wie gemeinsame Aktivierungsfrequenz, Priorität und Kritikalität in den Vordergrund. Technische Informationen wie z. B. Takte/Perioden oder Implementierungstypen müssen auf der Logical Architecture vollständig spezifiziert sein.

Die Technical Architecture (TA) repräsentiert alle für die Zuordnung von logischen Clustern zu technischen Ressourcen relevanten Konzepte, wie z. B. Steuergeräte, Busse und Slots bzw. Nachrichten der Kommunikationsmedien. Die eigentliche Zuordnung von Clustern zu Steuergeräten bzw. Datenflüssen zu Kommunikationsmedien erfolgt in der hier nicht weiter behandelten *Operational Architecture*, die dem aus den implementierungsorientierten Werkzeugen synthetisierten HW/SW-System entspricht.

Die LA kann auszugsweise in AutoFOCUS modelliert werden: zu diesem Zweck sind vor allem die Modellierungskonzepte Clocks (für heterogene Aktivierungsfrequenzen) sowie Implementierungstypen hilfreich. In vollständiger Ausprägung können LA und TA in den Werkzeugen ASCET, INTECRIO, sowie RTA OSEK Planner modelliert werden.

4 Berechnungsmodell

Das dem AutoMoDe-Projekt zugrundeliegende Berechungsmodell von AutoFOCUS verwendet eine nachrichtenbasierte, taktsynchrone Semantik mit uniformem, systemweitem Takt [8]. Dabei werden Rechenschritte innerhalb eines Taktes nicht weiter zeitlich aufgelöst. Modelle in AutoFOCUS werden als Netzwerke von *Komponenten* bzw. *Blöcken* definiert, die *Nachrichten* mit der Umgebung und untereinander über explizite Schnittstellen (Nachrichtenports) sowie Konnektoren zwischen Schnittstellen (Nachrichtenkanäle) austauschen. Nachrichten besitzen im abstrakten Modell einen Zeitstempel in Bezug auf den globalen Takt. Dieses datenflussorientierte Berechnungsmodell unterstützt einen hohen Grad an Modularität dadurch, dass Komponentenschnittstellen vollständig und explizit durch syntaktische Konstrukte abgebildet werden. Das Berechnungsmodell verringert den Grad an Komplexität in Echtzeitsystemen: Das Nachrichtenparadigma mit diskreter Zeitbasis abstrahiert von Implementierungsdetails wie detailliertem Timing, kausaler Abfolge, und verwendeten Kommunikationsmedien.

Beispiele für solche Implementierungsdetails beinhalten die Reihenfolge der Ankunft von Implementierungsnachrichten mit demselben logischen Zeitstempel, oder die genaue Dauer des Datentranfers auf einem Kommunikationsmedium. Somit werden Echtzeitintervalle der Implementierung durch logische Zeitintervalle abstrahiert. Das nachrichtenorientierte, zeitdiskrete Paradigma kann sowohl periodische als auch sporadische Berechnungen abbilden, wie es z. B. für die Modellierung von zeitgesteuertem und ereignisgesteuertem Verhalten notwendig ist.

Modellierung von Echtzeitverhalten. Um die heterogenen (a)periodischen Takte bzw. Frequenzen typischer Automotive-Anwendungen komfortabel modellieren zu können, werden in AutoFOCUS sog. *Clocks* [5] verwendet. Jeder Nachrichtenstrom in AutoFOCUS ist mit einer Clock assoziiert: Die Clock ist dabei ein Boolsche Ausdruck, welcher die Anwesenheit bzw. Abwesenheit einer Nachricht eines Nachrichtenstrom beschreibt. Zur Laufzeit evaluiert der Ausdruck zu logisch wahr wenn der der Nachrichtenstrom eine Nachricht enthält.

Mit expliziten *Sampling-Operatoren* zwischen Komponenten bzw. Clustern kann ein Datenfluss von einer Clock auf eine andere überführt werden. Mit Hilfe von strukturell über den Elementaroperatoren der Sprache definierten Regeln ist es dann möglich, für jedes AutoFOCUS-Konstrukt *Vorbedingungen* sowie *Inferenzregeln* bezüglich der Clocks anzugeben. Das AutoFOCUS-Werkzeug kann sowohl die Korrektheit des Designs in Bezug auf die Clocks (Erfüllung aller Vorbedingungen) überprüfen, als auch auf sämtliche interne und ausgabeseitige Clocks eines Systems schließen (Anwendung der Inferenzregeln).

5 AutoFOCUS-Notationen

AutoFOCUS bietet eine Anzahl von verschiedenen Notationen, die sich bei der Modellierung eingebetteter Systeme bewährt haben. Diese werden im Folgenden beschrieben.

Systemstrukturdiagramme (SSD). Systemstrukturdiagramme bieten eine architekturbezogene Gesamtsicht auf ein Software-System und werden für die Abstraktionsebenen FAA und FDA verwendet. Dabei werden Schnittstellen, Kommunikationsabhängigkeiten und hierarchische Dekomposition hinsichtlich der Funktionsarchitektur (FAA) bzw. der Softwarearchitektur (FDA) beschrieben. Ähnlich zu anderen visuellen Architekturbeschreibungssprachen oder der UML-RT wird das System dabei als Netzwerk von *Komponenten* beschrieben, die Nachrichten und Signale über gerichtete *Kanäle* untereinander austauschen. Komponenten sind entweder atomar, oder setzen sich hierarchisch aus Subkomponenten, die in einem eigenen SSD spezifiziert sind, zusammen.

Die Strukturierung auf SSD-Ebene hat dabei auch semantische Auswirkungen: Kommunikation zwischen SSD-Komponenten erfolgt mit einer Verzögerung um eine Periode der jeweiligen Clock. Durch diese Festlegung werden bereits auf hoher Abstraktionsebene “Sollbruchstellen” für die spätere Partitionierung im Rahmen des Deployments eingeführt, ohne diese Partitionierung im Detail vorwegzunehmen.

Abb. 3 zeigt ein typisches SSD für die Darstellung der in Abschnitt 7 beschriebenen Antischlupfregelung. Ein