

AutoMoDe

Automotive Model Based Development

Andreas Bauer, Ulrich Freund, N ria Mata, Jan Philipps, Jan Romberg, Bernhard Sch tz, Oscar Slotosch

{baueran | romberg | schaezt}@in.tum.de
Fakult t f r Informatik, TU M nchen
Boltzmannstrasse 3, D-85748 Garching

{Ulrich.Freund | Nuria.Mata}@etas.de
ETAS GmbH
Borsigstra e 14, D-70469 Stuttgart

{ philipps | slotosch }@validas.de
Validas AG
Lichtenbergstra e 8, D-85748 Garching

Kurzfassung

Software Engineering f r automobile Steuerungsanwendungen erfordert unterschiedliches Fachdisziplinwissen, um ein hohes Ma  an Korrektheit der Software sowie eine starke Optimierung der verwendeten Ressourcen zu erzielen. Der vorgestellte Ansatz unterst tzt mit Hilfe dom nenspezifischer, grafischer Beschreibungstechniken die werkzeuggest tzte Modellierung und Synthese eingebetteter Software-Systeme. Formal definierte Entwicklungsschritte wie Reengineering, Refactoring und Refinement werden werkzeugtechnisch unterst tzt. Ziel ist eine durchg ngige Werkzeugkette von dom nenspezifischen, abstrakten Modellen zu detaillierten Beschreibungen als Grundlage f r die Codesynthese.

1 Einleitung und Vorstellung des Themenkomplexes

Viele Konzepte der Softwareentwicklung der letzten 30 Jahre werden wegen hoher Echtzeitanforderungen und knappen Speicherressourcen in der Kfz-Industrie nicht in der Entwicklung eingebetteter Software verwendet.  nderungen in der Plattform bzw. den Anforderungen bedeuten meistens eine v llige Neuentwicklung der Software. Die Struktur der Softwaresysteme wird durch Blockdiagramme, ihr (Interaktions-)Verhalten auf der prozeduralen Implementierungsebene beschrieben. Aktuelle Entwicklungswerkzeuge (z.B. ASCET-SD, Titus) unterst tzen zwar diese Beschreibungsebene, bieten aber bisher keinen h heren Abstraktionsebenen. Im Gegensatz wurden in der Informatik mit der Theorie reaktiver nachrichtenbasierter

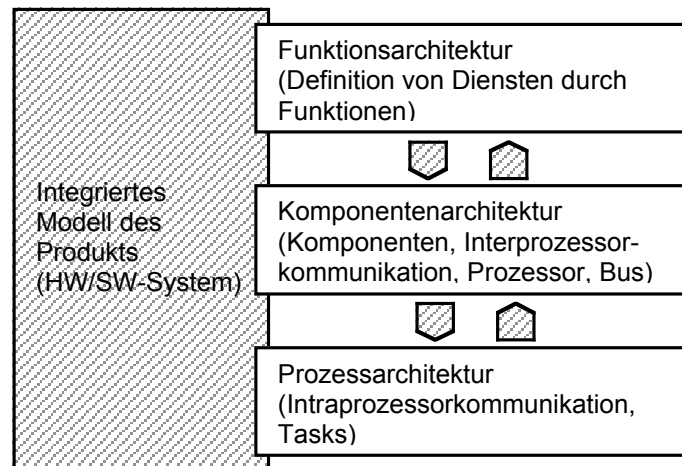


Abbildung 1: Prozessebenen und integriertes Produktmodell

Systeme ein gut verstandenes zur abstrakten Beschreibung reaktiver Systeme (Struktur und Verhalten, z.B. [BS01]) erarbeitet. Dieses Modell erlaubt die Beschreibung von eingebetteten Systemen auf hohem Niveau, mittels unterschiedlicher Sichten (z.B. Interaktionssicht, Zustandssicht), und unter Einsatz von modularen Entwicklungsschritten (z.B. Signalverfeinerung). Diese Prinzipien wurden in akademischen Werkzeugprototypen (z.B. AutoFocus [BL+00]) exemplarisch umgesetzt. Für die industrielle Anwendung bleiben jedoch viele der Sichten und Schritte zu abstrakt.

Mit dem hier vorgestellten AutoMoDe-Ansatz wird auf diesen pragmatischen und akademischen Grundlagen ein modellbasierter Ansatz [SPH+02] zur Entwicklung eingebetteter Automotive-Software erarbeitet. Die zwei wesentlichen Eigenschaften des Ansatzes sind ein *Produktmodell*, das domänenspezifische Sichten mittels formaler Konsistenzsicherung integriert, sowie ein *Prozessmodell*, das durch (werkzeuggestützte) formal fundierte Transformationsschritte realisiert wird.

1.1 Prozessebenen und Produktmodell

Für die Anwendung im Automotivesektor unterstützt AutoMoDe drei Prozessebenen mit unterschiedlich abstrakten Beschreibungsformen. Diese Ebenen werden durch ein domänenspezifisches Produktmodell integriert. Transformationsschritte mechanisieren den Entwicklungsprozess und sichern die Erstellung konsistenter Produkte.

1.1.1 Prozessebenen

Kernpunkt des AutoMoDe-Ansatzes bildet das in Abbildung 1 dargestellte integrierte Produktmodell, das die folgenden Abstraktionsstufen in einem qualitätsorientierten Entwicklungsprozess unterstützt. Zu jeder Abstraktionsstufe sind zur Illustration typische Elemente des automotive-spezifischen Produktmodells und geeignete notationelle Elemente angegeben:

- Die Funktionsarchitektur unterstützt die Analyse durch die Beschreibung des Systems als zu erbringende modulare Dienste abstrahiert von der konkreten Komponentenstruktur der Implementierungsplattform (Elemente: Systemschnittstelle, Signal, Funktion; Notationen: Strukturdiagramme, Funktionsdiagramme)
- Die Komponentenarchitektur dient dem Entwurf mit den Schwerpunkten Komponenten/Kommunikation (Elemente: SW/HW-Komponenten (Modul, Prozessor, Sensor, Aktor), Zustände; Notationen: Strukturdiagramme, Zustandsdiagramme)

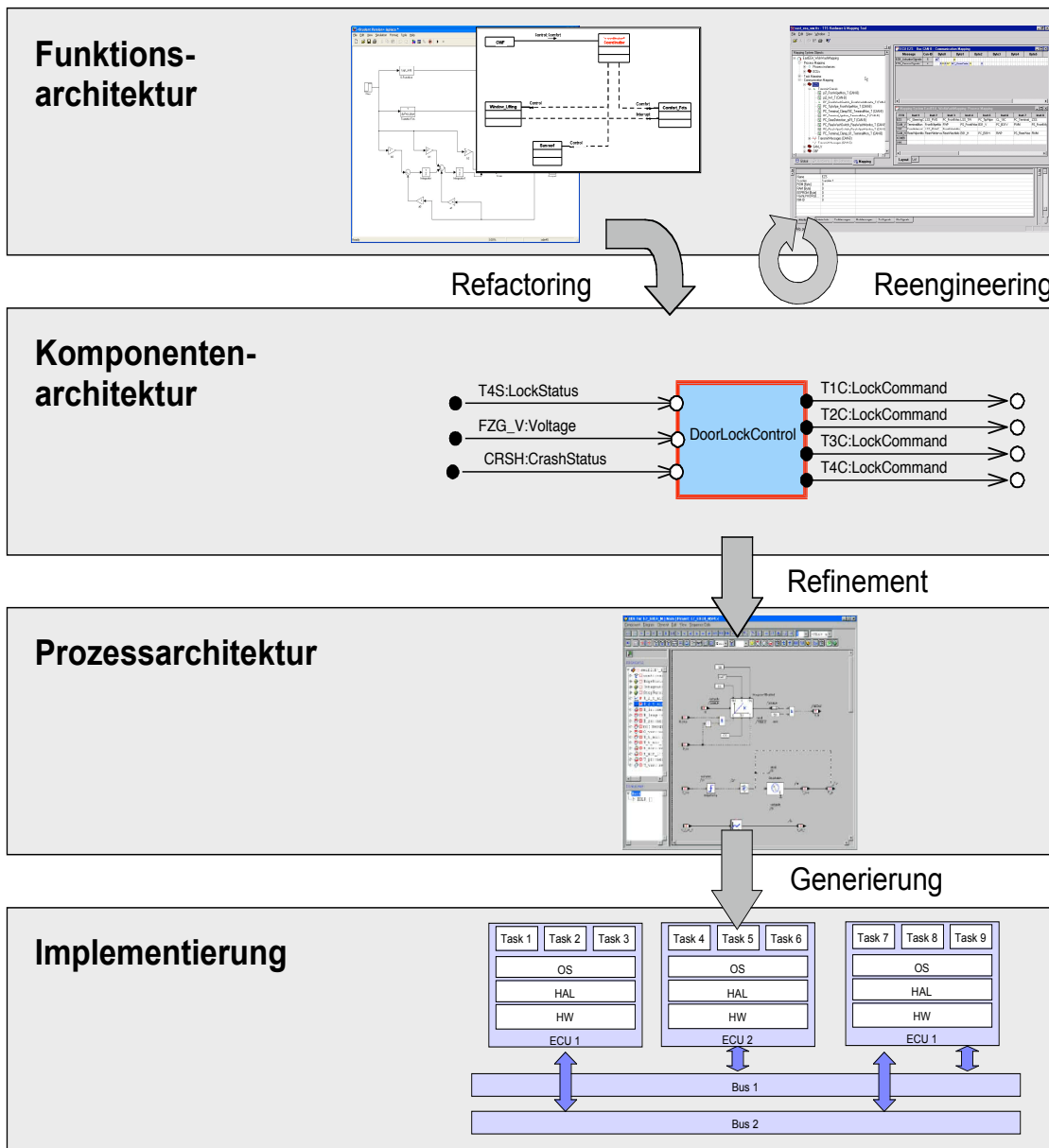


Abbildung 2: Prozessebenen und Prozessschritte

- Die Prozessarchitektur unterstützt die Implementierung mit Schwerpunkt algorithmischer Datenfluss und Scheduling (Elemente: Task, Event, Schedule, Speicher; Notationen: Strukturdiagramme, Taskdiagramme)

1.1.2 Produktmodell

Dem integrierenden Modell kommt dabei in der Prozessdefinition die entscheidende Bedeutung zu: es verbindet die verschiedenen Sichten in einer Ebene ebenso wie Abstraktionsebenen untereinander und erlaubt so den Übergang zwischen den Entwicklungsebenen. In der Anwendung kann es einerseits genutzt werden, um Validierungs- bzw. Konsistenzsicherungsmaßnahmen durch die Definition von Eigenschaften zu ermöglichen, die auf den einzelnen Abstraktionsstufen gelten müssen (z.B. Funktionsarchitektur: Vollständigkeit der Zuordnung Funktionen zu abstrakten Komponenten; Komponentenarchitektur: Schnittstellenkonsistenz zwischen Komponenten; Prozessarchitektur: Freiheit von zirkulären Abhängigkeiten im Datenfluss). Andererseits können Entwicklungsschritte als Operationen auf diesem Modell

beschrieben werden. Einfache stereotype Schritte können formalisiert, Eigenschaften nachgewiesen und ihre Ausführung durch ein Werkzeug unterstützt werden (z.B. Funktionsarchitektur: Kombination von Funktionen zu abstrakten Komponenten; Komponentenarchitektur: Abbildung Kanalkommunikation auf Busschnittstellen; Prozessarchitektur: Abbildung von Implementierungskomponenten auf Tasks).

1.1.3 Prozessschritte/Transformationen

Für die praktische Anwendung des in AutoMoDe entwickelten Werkzeugansatzes spielen drei Arten von Prozessschritten eine besondere Rolle (siehe auch Abbildung 2):

- *Reengineering*: Ziel des Reengineerings im Projekt AutoMoDe ist die Ableitung der Funktionsarchitektur aus der Elektronik-Architektur (Steuergeräte-Netzwerke) von in Produktion befindlichen Baureihen. In einem ersten Schritt werden die vorhandenen Funktionen identifiziert sowie die über Bussysteme übertragenen Signale klassifiziert. Diese Funktionsarchitektur kann durch vorhandene Spezifikationsmodelle des Funktionsverhaltens erweitert werden. Die Beschreibung dieser Funktionsarchitektur erfolgt mittels einer erweiterten AutoFocus Darstellung.

- *Refactoring*: Spezifikationsmodelle des Funktionsverhaltens (Regelungsalgorithmen) werden mittels Berechnungsnetzwerken beschrieben. Komponenten dieser Netzwerke sind elementare arithmetische Operationen auf skalaren Strömen reeller Zahlen bzw. Verzögerungskomponenten. Eine alternative Darstellung von Berechnungsnetzwerken sind Zustandsdiagramme aus AutoFocus mit einem Kontrollzustand sowie mehreren Datenvariablen. Das Berechnungsnetzwerk wird als arithmetischer Ausdruck in die Ausgabefunktion der Transition aufgenommen. Nach der Durchführung aller Refactoringschritte sind alle Berechnungsnetzwerke durch Zustandsdiagramme ersetzt worden. Im Prozessebenenmodell entspricht das Refactoring dem Übergang von der Funktionsarchitektur auf die Komponentenarchitektur.

- *Refinement*: Aus Implementierungssicht ist die Komponentenarchitektur weiterhin eine Abstrahierung des zu implementierenden Systems. Für eine effektive Implementierung der Komponentenarchitektur auf Mikrocontrollern wird u.a. die Ausführungsreihenfolge spezifiziert. Dies entspricht technisch einer Aufteilung von Komponenten auf Rechenraster und OS-Tasks. Weiterhin wird mit dieser Aufteilung die Implementierung von Konnektoren festgelegt, die wiederum die Umsetzung des AutoFocus Patternmatching-Algorithmus steuert. Im Prozessmodell entspricht das Refinement dem Übergang aus der Komponentenarchitektur in die Prozessarchitektur

1.2 Werkzeugkette

Das Produktmodell stellt das integrierende Element des AutoMoDe-Entwicklungsprozesses dar. Zur Anwendung des Produktmodells werden spezifische Editoren für die einzelnen Entwicklungsphasen erarbeitet. In der Praxis ist jedoch für eine Migration vom aktuellen Entwicklungsprozess zum AutoMoDe-Entwicklungsprozess die Anbindung zu etablierten Ansätzen und Werkzeugen unbedingt erforderlich. Dies beinhaltet sowohl die Übernahme von Modellierungskonzepten aus diesen Ansätzen ins Produktmodell als auch die unmittelbare werkzeugtechnische Anknüpfung.

1.2.1 MATLAB/Simulink

Ein (nicht nur in der Automobilindustrie) verbreitetes Modellierungswerkzeug für Regelungssysteme ist MATLAB/Simulink, das gelegentlich auch mit der Erweiterung Stateflow eingesetzt wird, um auch gemischt diskret/kontinuierliche („hybride“) Systeme beschreiben zu können. Solche Modelle erlauben insbesondere eine konzise Modellierung auch der Umwelt der Steuerung, und damit die Erarbeitung und Analyse von Regelgesetzen. Obwohl es durchaus Generatoren gibt, die aus diesen Modellen beispielsweise C-Code erzeugen, ist es bis jetzt

eher verbreitet, die Analysemodelle von Hand in Code für Steuergeräte umzusetzen. Für AutoMoDe bietet es sich aber an, die Modelle soweit wie möglich zu übernehmen: Einerseits als Eingabesprache, während es noch keine dedizierten Editoren für AutoMoDe gibt, andererseits aber auch zur raschen Erstellung von Prototypen aus den Analysemodellen.

Die Anbindung von MATLAB/Simulink an die vorgeschlagene Werkzeugkette erfolgt durch die Übersetzung von Simulink-Modellen (genauer: ihrer Regelungsanteile) in das Produktmodell von AutoMoDe (Berechnungsnetzwerke der Funktionsarchitektur). Insbesondere werden nur diskretisierte Regelungsalgorithmen nach AutoMoDe transformiert. Durch kontinuierliche Modelle beschriebene Strecken werden nicht transformiert. Die Abschätzung der Schrittweiten von diskreten Regelungsalgorithmen (und damit die Abschätzung der für die Implementierung des Systems nötigen Wiederholraten) ist eine typische Analyseaufgabe, für die sich die Simulationstechniken von MATLAB/Simulink anbieten.

1.2.2 ASCET-SD

ASCET-SD beschreibt eingebettete Kfz-Software in Form von Rechenrastern (sogenannten Prozessen), Klassen und Instanzen. Das Verhalten von Klassen wird mittels Datenflussdiagrammen und endlichen Automaten beschrieben. Innerhalb von Prozessen wird die Ausführungsreihenfolge durch die Spezifikation von Teildatenflüssen definiert. Der Schedule von Prozessen und die damit verbundene Zuordnung auf OS-Tasks bestimmt die Ausführungsreihenfolge innerhalb eines Steuergerätes. Diese Art der Softwaremodellierung erlaubt einerseits eine objektbasierte Darstellung von eingebetteter Echtzeitsoftware, andererseits eine effektive Implementierung auf Mikrocontrollern mittels automatischer C-Code-Generierung.

Im AutoMoDe-Prozess wird für jede Steuergerätekomponente die Prozessarchitektur in ein äquivalentes ASCET-SD Modell überführt. Insbesondere bleibt dabei die Struktur und das Verhalten der Klassen aus der verfeinerten Komponentenarchitektur erhalten, während Eigenschaften und Abarbeitungsreihenfolge von Prozessen der verfeinerten Prozessarchitektur entnommen sind.

1.3 Ziele

Mit der Einführung des integrierten Modells sowie der Definition von Konsistenzbedingungen bzw. Modelloperationen werden drei grundlegende Verbesserungen hinsichtlich der Entwicklung verteilter eingebetteter Steuerungssoftware ermöglicht:

- Mit dem AutoMoDe-Projekt wird eine modellbasierte Methode entwickelt, die zur Modellierung technischer Systeme der Anwendungsdomäne Automotive geeignet ist. Somit werden die Grundlagen für die Beherrschung der zunehmend komplexen HW/SW-Systeme im Automobilbereich geschaffen.
- Die Unterstützung der Wiederverwendung auf unterschiedlichen Abstraktionsniveaus (z.B. Dienste auf Funktionsebene, Komponenten auf Kommunikationsebene, Tasks auf Taskebene) zur Unterstützung eines Produktlinienansatzes, beginnend ab der Anforderungsanalyse. Die einzelnen Architekturen sind dabei als weitgehend plattformunabhängige und langlebige Darstellungen konzipiert, insbesondere auf Funktions- und Komponentenebene.
- Die Erhöhung der Korrektheit und Zuverlässigkeit verteilter Softwaresysteme durch Einsatz eines integrierten (d.h. sichten- bzw. abstraktions- sowie phasenübergreifenden) Produktmodells und qualitätssichernder Analysetechniken (z.B. Vollständigkeit von Verhaltenbeschreibungen.) Abstrakte Modelle des späteren Systems ermöglichen früh in der Entwicklung den Einsatz qualitätssichernder Maßnahmen, und erhöhen so die Funktionssicherheit bei Reduzierung des Testaufwands.

1.4 Verwandte Ansätze

Im Automotive-Projekt des Forschungsverbunds Software Engineering [BBR+01] wurde eine modellbasierte, werkzeuggestützte Vorgehensweise auf der Basis kommerzieller Werkzeuge definiert und in Form eines integrierten Produktmodells vorbereitet. Im Gegensatz zu AutoMoDe liegt der Schwerpunkt dabei auf den frühen Entwicklungsphasen. Insbesondere beschränkt sich die Werkzeugkopplung in späteren Phasen auf ein rein strukturelles Modell der verteilten Anwendung. Im ITEA-Projekt EAST-EEA [EAST] steht die Definition einer Middleware-Lösung für verteilte automobiler Anwendungen im Vordergrund. Die Beschreibung verteilter Systeme findet dabei in Form einer Architekturbeschreibungssprache (ADL) statt, die jedoch wesentliche Verhaltenseigenschaften der Anwendung nicht festlegt.

Hier wird ein weitergehender, modellbasierter Ansatz für die Modellierung und Synthese verteilter automobiler Steuerungssysteme entwickelt. Wesentliche Merkmale des Ansatzes sind die Untergliederung in Prozessebenen, ein integriertes Produktmodell, die Definition von formalisierten Prozessschritten auf dem Produktmodell, sowie die Unterstützung durch kommerzielle Werkzeuge.

2 Projektstatus

Aufgrund seiner zentralen Bedeutung für den AutoMoDe-Ansatz steht die Erarbeitung des Produktmodells im ersten Projektabschnitt im Vordergrund. Das im vorherigen Abschnitt skizzierte Produktmodell erfasst und integriert die Elemente eines Systems. Gleichzeitig strukturiert es auch den Entwicklungsprozess durch seine Gliederung in *Abstraktionsebenen*. Abstraktionsebenen definieren eingeschränkte Sichten auf Modelle um diese zu handhabbarer zu machen und Informationen zu filtern. Jede Abstraktionsebene basiert auf der darüber liegenden abstrakteren Ebene. Auf einer detaillierteren Ebene kann also auf Informationen, die in abstrakteren Ebenen enthalten sind, zugegriffen werden. Bei dem Übergang von einer abstrakten zu einer detaillierten Abstraktionsebene werden neue Informationen in das Modell aufgenommen, die die Menge der möglichen Lösungen (Freiheitsgrade) für das spezifizierte System weiter einschränken.

Die Werkzeugabhängigkeiten und damit verbundenen Übergänge zwischen den Werkzeugen erlauben es, wichtige Schritte für einen begleitenden Prozess abzuleiten. Typischerweise liegen bereits außerhalb des AutoMoDe-Werkzeugs Beschreibungen von Neu- oder Altsystemen vor, z.B. (Architekturbeschreibungen wie beim CARTRONIC-Ansatz, einem offenen Konzept zur funktionalen Strukturierung und Modellierung für Steuerungs- und Regelungsaufgaben im Kraftfahrzeug), Simulink-Entwürfe (prototypische Struktur und Verhalten), oder Kommunikationsmatrizen (Schnittstellen und Kommunikationsbeziehungen vorhandener Systeme in Form herstellerepezifischer Beschreibungen).

Als Beschreibungstechniken, die als Grundlage für die präzise Darstellung des zu entwickelnden Systems auf den eingeführten Abstraktionsebenen dienen, werden Beschreibungstechniken auf Basis der AutoFOCUS-Notation [HSE97] eingeführt:

- Systemstrukturdiagramme: Systemstrukturdiagramme (SSD) bieten eine architekturbezogene Gesamtansicht auf ein Software-System verwendet. Dabei werden Schnittstellen, Kommunikationsabhängigkeiten, und hierarchische Dekomposition hinsichtlich der Funktionsarchitektur bzw. der Softwarearchitektur beschrieben.
- Datenflussdiagramme: Datenflussdiagramme (DFD) beschreiben den algorithmischen Datenfluss von Komponenten und werden typischerweise auf allen Abstraktionsebenen eingesetzt. DFDs selbst sind wiederum aus atomaren oder hierarchischen Blöcken aufge-

baut, die ähnlich zu SSDs über gerichtete und typisierte Kanäle verbunden sind. Weiterhin können DFDs einem speziellen Modus eines Modus-Diagramms zugeordnet sein, der über die Aktivierung einer Berechnung entscheidet.

- Modus-Diagramme: Modus-Diagramme (MSD) werden auf allen Abstraktionsebenen dazu verwendet zwischen alternativen Betriebsmodi oder Konfigurationen einer Komponente zur Laufzeit hin- und herzuschalten. Abhängig vom Zustand eines MSDs können so z.B. unterschiedliche DFDs als Berechnungsvorschrift einer Komponente aktiviert sein. Darüber hinaus werden zusätzliche Beschreibungsformen entwickelt, z.B. zur Beschreibung der technischen Architektur auf Prozessebene. Für diese Diagramme wurden detaillierte Berechnungsvorschriften und Konsistenzbedingungen erarbeitet. Zusätzlich wurden bereits eine prototypische Werkzeugunterstützung zur Bearbeitung und Konvertierung dieser Beschreibungsformen erarbeitet.

3 Ausblick

Die Entwicklung einer integrierten Methodik für den Entwurf eingebetteter Software – unter Einbeziehung von regelungs- und steuerungstechnischen Modellen einerseits, sowie reaktiven Modellen andererseits – wurden als Grenzgebiet zwischen Elektro- und Informationstechnik sowie Informatik bisher kaum aus dem Blickwinkel der fundierten, modellbasierten Entwicklung angegangen. Dennoch hat dieses Teilgebiet der Informatikforschung eine hohe wirtschaftliche Bedeutung:

- Für die deutsche IT-Industrie: Da eingebettete Software einen hohen industriellen Stellenwert hat, ergibt sich für die Umsetzung der Methodik ein großer Markt, insbesondere in der Kraftfahrzeugelektronik.
- Für die deutsche Kfz-Industrie: Nur durch den Einsatz von effizienten Software-Entwicklungsmethoden kann die notwendige Produktivitätssteigerung erreicht werden, die insbesondere die deutschen Fahrzeughersteller benötigen, um sich im internationalen Wettbewerb zu differenzieren bzw. die heutige Vormachtstellung erhalten kann.

Die Bedeutung eines solchen Entwicklungsprozesses geht über die Automobilindustrie hinaus: dank ihrer Rolle als Technologietreiber wirken sich entsprechende Neuerungen in der Automobilindustrie langfristig auch auf die Produktions- und Automatisierungstechnik und damit besonders stark auf den deutschen Mittelstand aus. Um dieser zusätzlichen nachhaltigen Wirkung des Vorhabens verstärkt Rechnung zu tragen, wird im Rahmen des Projekts auch die Übernahme der Projektergebnisse in die akademische Ausbildung vorgenommen.

Literatur

- [BBR+01] M. von der Beeck, P. Braun, M. Rappl, C. Schröder, „Modellbasierte Softwareentwicklung für automobilspezifische Steuergerätenetze“, VDI Tagung Elektronik im KFZ, Baden Baden, 2001
- [BL+00] P. Braun, H. Lötzbeyer, B. Schätz, O. Slotosch Consistent Integration of Formal Methods. IN: Tool and Algorithms for the Construction and Analysis of Systems. Springer Verlag, 2000.
- [BS01] M. Broy, K. Stoelen. Specification and Development of Interactive Systems. Springer, 2001.
- [EAST] EAST-EEA Projekthomepage, <http://www.east-eea.net>
- [Hen00] T. Henzinger: „What is Giotto“, University of California, Berkeley. WWW: <http://www-cad.eecs.berkeley.edu/~fresco/giotto/what.shtml>, 11/2000
- [HSE97] F. Huber, B. Schätz, G. Einert, „Consistent Graphical Specification of Distributed Systems“, Proceedings of FME'97, LNCS 1313, pp. 122—141, Springer Verlag, 1997
- [SPH+02] B. Schätz, A. Pretschner, F. Huber, Philipps, J. Model-based Development of Embedded Systems Technical Report TUMI-0402, TU München, 2002