ITEA project 00009

EAST-EEA
Embedded Electronic Architecture

Report Typ	Bericht
Report Name	Schlussbericht
Berichtszeitraum	01.05.2001 – 31.07.2004

Report Status	Öffentlich
Versionsnummer	Version 1.0
Erstellungsdatum	14.02.2005

Zuwendungsempfänger	Förderkennzeichen
ZF Friedrichshafen AG	*01 IS A01 C*
Laufzeit des Vorhabens	*01.05.2001 – 31.07.2004*

Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung, Wissenschaft, Forschung und Technologie unter dem Förderkennzeichen 01 IS A01 C gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

E-A-S-T ▶ E-E-A

I T E A
INFORMATION TECHNOLOGY
FOR EUROPEAN ADVANCEMENT

Dieser Bericht wurde erstellt durch die ZF Friedrichshafen AG.

Beiträge wurden verfasst von

Joachim Eisenmann DC AG
Dr. Roland Geiger ZF Friedrichshafen AG

© 2004 EAST-EEA Konsortium

Weitergabe sowie Vervielfältigung dieser Unterlage, Verwertung und Mitteilung ihres Inhalts sind nicht gestattet, soweit nicht ausdrücklich zugestanden. Zuwiderhandlungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patenterteilung oder Gebrauchsmustereintragung vorbehalten. Für Fehler oder Auslassungen wird keine Haftung übernommen.

Versionsübersicht

Version	Datum	Grund
0.1	09.12.2004	Vorlage von J. Eisenmann, DCAG
0.9	25.01.2005	Kurzdarstellung von J. Eisenmann, DCAG
1.0	14.02.2005	Eingehende Darstellung von R. Geiger, ZF

Inhaltsverzeichnis

Versionsübersicht iv
Inhaltsverzeichnis iv
Abbildungsverzeichnis iv
1 Kurzdarstellung 4
1.1 Aufgabenstellung 4
1.2 Voraussetzungen des Vorhabens 4
1.3 Planung und Ablauf des Vorhabens 4
1.4 Wissenschaftlich-technischer Stand 4
1.5 Zusammenarbeit mit anderen Stellen 4
2 Eingehende Darstellung 4
2.1 Erzieltes Ergebnis 4
2.2 Voraussichtlicher Nutzen 4
2.3 Fortschritt auf diesem Gebiet bei anderen Stellen 4
2.3 Erfolgte und geplante Veröffentlichungen 4
Literaturverzeichnis 4
Anlage 1: Erfolgskontrollbericht 4
Anlage 2: Kurzfassung 4

Abbildungsverzeichnis

Abbildung 1: ZF-Produkte 4
Abbildung 2: Erfahrungen mit Getriebesteuerungen 4
Abbildung 3: Erfahrungen im Software-Sharing mit OEM's 4
Abbildung 4: ZF-Beteiligung am EAST-EEA Projekt 4
Abbildung 5: Strukturierung WT1.5 4
Abbildung 6: Architektur der Powertrain-Domäne 4
Abbildung 7: Neustrukturierung von WP2 4
Abbildung 8: Beispiel Transparente Kommunikation 4
Abbildung 9: Beispiel Parameter-Kalibrierung 4
Abbildung 10: Beispiel Operation Mode Management 4
Abbildung 11: Middleware Konfigurations- und Build-Prozess 4
Abbildung 12: Validator für spezifische Validierung 4
Abbildung 13: Abgestimmter Aufbau des Powertrain Validator 4
Abbildung 14: Abgestimmte Komponenten-Verantwortlichkeiten 4
Abbildung 15: Szenario 1 als Basisszenario 4
Abbildung 16: Szenario 2 als Verschiebeszenario 4
Abbildung 17: Szenario 3 als Austauschszenario 4
Abbildung 18: Logische Architektur des Powertrain Validator 4
Abbildung 19: Strukturiertes Simulink® Modell 4
Abbildung 20: ECE Fahrzyklus als Referenz-Stimulation 4
Abbildung 21: Modellrechner 4
Abbildung 22: Testumgebungen 4
Abbildung 23: Validator-Konzept mit Steuergeräten 4
Abbildung 24: Software-Struktur in DaVinci 4
Abbildung 25: Konzept zur Integration des ISG-Steuergeräts 4
Abbildung 26: Middleware-Integration am Beispiel Kalibrierung 4
Abbildung 27: Implementierung am Beispiel Mode Management 4
Abbildung 28: Integrierte Powertrain-Steuergeräte 4
Abbildung 29: Referenztest 4
Abbildung 30: Referenz Simulation – Test PC-Modell 4
Abbildung 31: Referenz Szenario 1 – Test Szenario 2 4
Abbildung 32: Aufbau des Validators zur Abschlußpräsentation 4
Abbildung 33: Powertrain Steuergeräte-Verbund mit PC 4
Abbildung 34: Beteiligung am HMI Validator 4
Abbildung 35: XML-Beschreibung für die Einparkhilfe 4
Abbildung 36: Standardisierung als Treiber 4

1 Kurzdarstellung

1.1 Aufgabenstellung

Ziel dieses Projektes ist die Schaffung einer standardisierbaren, offenen Softwarearchitektur im Kfz, die auf existierenden Standards aufsetzt und die eine hardwareunabhängige Verteilung von Software-Modulen in vernetzten Kfz-Systemen zulässt.

Einerseits kann der Kfz-Hersteller dadurch Software für seine „embedded real-time systems“ als Lieferantenteil mit „Teilenummer“ auf die im Fahrzeug verbaute Hardware adaptieren, unabhängig vom jeweiligen Zulieferanten. Andererseits können neue Kfz-Funktionen allein in Software durch ineinander greifende Verknüpfungen ebenfalls Zulieferanten übergreifend realisiert werden.

Diese offene Softwarearchitektur soll durch eine geschichtete Softwarestruktur erreicht werden, die auf existierenden Lösungen (z.B. OSEK/VDX) aufsetzt. Fokus ist eine so genannte Middleware-Schicht, die sowohl die Partitionierung der Funktionssoftware auf eine verteilte und vernetzte Hardware unterstützt, als auch die Interoperabilität der unterschiedlichen SW-Module garantiert.

Die Vorteile dieser offenen Systemarchitektur sind:

- für die Kfz-Hersteller:

  Gemeinsame Wieder- und Mehrfachnutzung von nicht wettbewerbsrelevanten Softwaremodulen. Wettbewerbsrelevante Funktionen können jeweils separat / spezifisch entwickelt werden.

- für die Kfz-Zulieferer:

  Reduktion der Variantenvielfalt, die entsteht, wenn jeder Kfz-Hersteller seine eigenen Standards setzt.

  Damit kann die Effizienz in der Funktionsentwicklung gesteigert und die internationale Wettbewerbsfähigkeit gesteigert werden. Neue Geschäftsmodelle für Zulieferanten und die Senkung der Einstiegsschwelle für neue Firmen sind zu erwarten.

- für Toolhersteller:

  Einheitliche Schnittstellen zu den Entwicklungsprozessen. Transparente Schnittstellen fördern auch hier eine Senkung der Einstiegsschwelle für neue Firmen.

Insgesamt erfordert oder stimuliert die Kfz-Industrie somit Lösungen, die einen wesentlichen Beitrag zur Entwicklung der deutschen IT-Industrie (im europäischen Markt) bringt.

Zurzeit finden viele neue Funktionen Eingang in das Kfz, z.B. Navigation, Verkehrstelematik, Fahrwerksteuerung, adaptive Temporegelung usw. Diese Funktionen sind noch weitgehend autark. Eine wichtige technische Aufgabe der beteiligten Kfz-Hersteller ist die Integration unterschiedlicher Elektroniksysteme und deren Komponenten verschiedener Zulieferer zu einem kostengünstigen Netzwerk im Fahrzeug. Um die Entwicklungskosten zukünftiger Systeme möglichst niedrig zu halten, ist das Verfahren zur Integration für Anwendungs- und Kommunikations-Schnittstellen und Entwicklungsumgebungen möglichst herstellerunabhängig zu formulieren.

Dies soll über die Definition einer offenen Systemarchitektur erreicht werden. Wesentliches Element dieser Architektur ist ein Schichtenmodell unter Einschluss einer Middleware-Schicht, die Schnittstellen und Dienste bereitstellt, welche die Portabilität von eingebetteten Software-Modulen unter hohen Qualitätsstandards gewährleistet.

Die Projektarbeit wird in den folgenden Teilprojekten (Work Packages) abgearbeitet:

WP1: General Aspects

In diesem Teilprojekt werden die Grundlagen für die weitere Projektarbeit bereitgestellt. Folgende Arbeitspakete (Work Tasks, WT) werden behandelt:

WT1.1 Zukünftiges technisches Szenario

Hier wird eine Benutzersicht bezüglich der zukünftig zu erwartenden Integration elektronischer Systeme im Kraftfahrzeug unter Marketingaspekten, d.h. ohne Erläuterung technischer Hintergründe, dargestellt. Ergebnis ist die Vision des zukünftigen Umfeldes der im Projekt zu entwickelnden Fahrzeugarchitektur.

WT1.2 Definition der Allgemeinen Anforderungen (Requirement-Analyse)

Hier werden die Anforderungen der Projektpartner in Form von Anwendungsfällen (Use Cases) gesammelt. Hieraus werden zum einen die technischen Kriterien zur Validierung der beispielhaften Implementierungen abgeleitet und zum anderen die Rohdaten zur Ermittlung der Referenzarchitektur (siehe WT1.5) erzeugt.

WT1.3 Existierende Lösungen und ein Fahrplan zu ihrer Harmonisierung

Ziel der Arbeiten ist generell die Vermeidung teurer und unnötiger Neuentwicklungen durch Rückgriff auf existierende Lösungen. Hierbei sollen sowohl Anwendungen in den verschiedenen Domänen (Karosserieelektronik, Triebstrang, Telematik usw.) als auch bei der Bustechnologie berücksichtigt werden.

WT1.4 Verfahren zum Umgang mit geistigen Eigentumsrechten und zukünftige Geschäftsmodelle

Die im Rahmen des EAST/EEA-Projekts zu erarbeitenden Mechanismen sollen das Zusammenwirken von Softwaremodulen ermöglichen, die von unterschiedlichen Entwicklungspartnern (Fahrzeughersteller und Zulieferer) erstellt werden. Diese Vorgehensweise erfordert neue Verfahren und Methoden zum Umgang mit geistigen Eigentumsrechten. Dabei sind zwei verschiedene Aspekte zu berücksichtigen:

- Technische Aspekte:

  Es muss ein Modell entwickelt werden, das es ermöglicht, die unterschiedlichen Rollen bei der Softwareentwicklung für Kfz-Steuergeräte (z.B. Systemintegrator, Anwendungsentwickler oder Steuergeräte-Integrator) sowie deren Schnittstellen untereinander technisch zu definieren. Dies kann auf Basis moderner kryptographischer Verfahren erfolgen. Der Informationsfluss über diese Schnittstellen muss dabei bestimmten noch festzulegenden Regeln gehorchen.

Dies stellt sicher, dass jeder Entwicklungspartner sein spezifisches Know-how schützen kann und den anderen Partnern nur genau die Informationen zur Verfügung stellen muss, die diese von ihm benötigen. Die in diesem Arbeitspaket erarbeiteten Methoden sollen in die im WP3 zu spezifizierenden Entwicklungstools integriert werden.

- Rechtliche Aspekte:

  Hier müssen die Auswirkungen dieser verteilten Softwareentwicklung auf zukünftige Geschäftsmodelle der einzelnen Entwicklungspartner untersucht werden. Das Ergebnis umfasst sowohl Randbedingungen für die Vertragsgestaltung zwischen den Partnern mit besonderem Augenmerk auf Haftungsfragen als auch mögliche Vorschläge an den Gesetzgeber für die zukünftige Gestaltung von gesetzlichen Rahmenbedingungen, sofern diese notwendig erscheinen.

WT1.5 Ableitung der zukünftigen Referenzarchitektur

In diesem Arbeitspaket soll die allgemeine Referenzarchitektur ausgehend von den Ergebnissen aus WT1.2 und WT1.3 abgeleitet werden. Dies erfolgt im Domain Engineering Prozess:

Eine *Domänenarchitektur* wird als Beschreibung der Analyse der spezifischen Domänen des Fahrzeugs bestimmt.

Eine *Subsystem-Architektur* wird aus der Domänenarchitektur abgeleitet durch Spezifikation kooperierender funktionaler Subsysteme.

Die *Referenzarchitektur* enthält weitere detaillierte Definitionen in bestimmten Bereichen der Subsystem-Architektur. Wesentliches Element dieser Architektur ist ein Schichtenmodell unter Einschluss einer Middleware-Schicht, die Schnittstellen und Dienste bereitstellt, welche die Portabilität von Embedded Software-Modulen unter hohen Qualitätsstandards gewährleistet.

WP2: Interoperability and Portability

In diesem Teilprojekt soll die Spezifikation eines Middleware- und Kommunikationskonzepts erarbeitet werden. Die Middleware bietet dabei den Anwendungsfunktionen Dienste in Form eines APIs (Application Programmers Interface) an, das eine abbildungstransparente Interaktion zwischen verschiedenen Anwendungsfunktionen im Fahrzeug erlaubt (Interoperabilität). Die Kommunikationsschicht stellt der Middleware Basiskommunikationsdienste zur Verfügung, die über Device-Treiber an die Fahrzeugnetzwerke (z.B. CAN, MOST, IEEE1394) adaptiert werden.

Das Teilprojekt unterteilt sich in die folgenden Arbeitspakete:

WT2.1 General Implementation Framework

In diesem Arbeitspaket werden die Anforderungen an die Middleware und die Kommunikationsschicht erarbeitet. Dazu gehört sowohl die Erarbeitung subsystemspezifischer Anforderungen als auch die Berücksichtigung allgemeiner Anforderungen. Das Ergebnis des Arbeitspakets ist eine Beschreibung der Struktur sowie der von den einzelnen Schichten zur Verfügung gestellten Dienste.

WT2.2 Middleware Specification

Aus den Ergebnissen von WT2.1 wird die Spezifikation der Middlewareschicht einschließlich den der Anwendung bereitgestellten Diensten erarbeitet.

WT2.3 Middleware Implementation

Prototypische Implementierung der in WT2.2 erarbeiteten Spezifikation der Middleware zur Validierung der Spezifikation sowie als Basis für die in WP4 zu erstellenden Demonstratoren.

WT2.4 Communication Layer Specification

Aus den Ergebnissen von WT2.1 wird die Spezifikation der Kommunikationschicht einschließlich den der Middleware bereitgestellten Diensten erarbeitet.

WT2.5 Communication Layer Implementation

Prototypische Implementierung der in WT2.4 erarbeiteten Spezifikation des Communication Layers zur Validierung der Spezifikation sowie als Basis für die in WP4 zu erstellenden Demonstratoren.

WP3: Development and Validation Tools

Dieses Teilprojekt behandelt die Entwicklung und Validierung von Architekturen und Funktionsmodulen. Es berücksichtigt den vollständigen Entstehungsprozess von der Spezifikation über die Simulation, Implementierung, Einzeltests und Integrationstests bis zur Validierung. Ziel ist, auf bereits existierenden Ansätzen aufzusetzen. Daher werden den jeweiligen Definitionsphasen grundsätzlich Untersuchungen bestehender Lösungsansätze vorgeschaltet.

WT3.1 Architecture, Modelling Language and Exchange Format

Hier geht es um die Definitionen einer Beschreibungssprache für "Embedded Electronic"- Architekturen (ADL = Architecture Description Language) für eine offene Systemarchitektur, einer Interface-Sprache zur Nutzung bestehender Funktionsmodule, die Anforderungen an ein XML- Austauschformat, sowie Analyse- und Modifikations-Tools zur Struktur-Optimierung.

WT3.2 Functional Models/Modules Test & Validation

WT3.3 Architecture Allocation and Validation, Performance and Safety

WT3.4 Integration Test & Validation

Die Tasks 3.2, 3.3 und 3.4 behandeln die Test- und Validierungsphasen für die Funktionsarchitektur, die Elektronikarchitektur und die Integration. Es werden die erforderlichen Tools, Prozesse, Prozeduren und Methodiken definiert, um eine hohe Produktqualität hinsichtlich Anforderungserfüllung, Sicherheit, Leistung und Robustheit sicherzustellen.

WT3.5 Automatic Code Generation

Hier werden die Anforderungen an einen automatischen Code-Generator für Applikationen mit verteilten Systemstrukturen untersucht.

WP4: Domain Specific Implementation and Validation

Das Ziel dieses Teilprojekts ist es, die in WP2 erarbeiteten Konzepte unter Verwendung der in WP3 erarbeiteten Methoden an Hand

domänenspezifischer Demonstratoren zu validieren. Dabei handelt es sich um Labordemonstratoren, die allerdings, soweit dies im Rahmen dieses Projekts möglich ist, auf Basis seriennaher Steuergeräte aufgebaut werden sollen. Die Erfahrungen aus der Validierung durch die Demonstratoren sollen zur Optimierung der in den anderen Teilprojekten erarbeiteten Lösungen dienen.

Beim Aufbau der Demonstratoren soll gezeigt werden, dass es durch den Einsatz der in EAST-EEA erarbeiteten Konzepte und Methoden möglich ist, die spezifischen Herausforderungen bei der Entwicklung von Software für die einzelnen Fahrzeugdomänen zu meistern.

WT4.1 Body Electronics

Bei der heutigen Elektronikentwicklung werden die Steuergeräte vielfach als Einheit von Hardware und darauf laufender Software, also als „Black Box“, betrachtet. Die in EAST-EEA erarbeiteten Methoden erlauben eine Trennung von Soft- und Hardware, d.h. eine separate Betrachtung aller für die Softwareentwicklung notwendigen Aspekte unabhängig von der zugrunde liegenden Hardware. Durch den Demonstrator sollen die folgenden Vorteile der EAST-EEA-Vorgehensweise gezeigt werden:

- Software-Funktionen und Steuergeräte-Hardware können durch unterschiedliche Zulieferer entwickelt werden.
- Software-Funktionen eines Zulieferers können auf unterschiedliche Steuergeräte anderer Zulieferer verteilt werden.

Die in WP2 spezifizierte Middleware stellt den Funktionen unabhängig von der Verteilung einen Satz von vereinheitlichten Diensten zur Verfügung.

Die in WP3 erarbeitete Entwicklungsmethode erlaubt den Aufbau von Funktionsbibliotheken sowie den Austausch von Softwaremodulen zwischen Fahrzeugherstellern und Zulieferern unter Beachtung des Rechts auf geistiges Eigentum.

WT4.2 Powertrain

Die Implementierung der Middleware- und Communication-Layer aus WP 2 und die Methoden und Tools aus WP3 bilden die Grundlage für die Realisierung einer EAST-SW-Architektur. Die Teilnehmer an der WT4.2 haben sich auf folgende zu untersuchende Antriebsstrangkomponenten geeinigt:

- Verbrennungsmotor
- Starter/Generator
- Automatikgetriebe und/oder automatisiertes Getriebe.

Hauptaufgabe ist die Untersuchung und Validierung der Echtzeitfähigkeit und der Konfigurationsmöglichkeiten der in den vorhergehenden WP’s definierten Architekturen, Tools und Kommunikationsmechanismen an ausgewählten Applikationsbeispielen. Die Ergebnisse werden an firmenspezifischen Demonstratoren dargestellt. Zusätzlich besteht die Option Teilergebnisse an einem firmenübergreifenden gemeinsamen Demonstrator zu erproben.

WT4.3 Telematics

Mit Multimedia, Internet, Telekommunikation und Telematikdiensten sind Technologien ins Fahrzeug zu integrieren, die zum größten Teil

von IT- und Consumerindustrie getrieben werden und von sehr verschiedenen Lieferanten kommen können. Typischerweise sind deren Produktzyklen deutlich kürzer als diejenigen von Fahrzeugen. Fahrzeuge stellen nur einen kleinen Markt, der jedoch technologisch hohe Anforderungen bei beherrrschbaren Kosten stellt.

Die Leistungsfähigkeit der im EAST-EEA Projekt erarbeiteten Architekturkonzepte und Methoden sollen für die Telematik-Domäne durch die Realisierung eines oder mehrerer Szenarios zu Download und Installation von Software ins Fahrzeug untersucht werden. Hierzu stehen verschiedene Szenarien zur Auswahl:

- Szenario Nutzerprofile:

  Der Benutzer eines Fahrzeuges wird anhand einer Chipkarte oder eines biometrischen Merkmals erkannt, sein Profil von seinem Home-Server geladen und so stets die vertraute Umgebung eingestellt.

- Szenario Neuer Telematikdienst:

  Die Software zu einem bisher nicht im Fahrzeug verfügbaren Telematikdienst wird über eine Funkschnittstelle ins Fahrzeug geladen und automatisch installiert. Den Fahrzeuginsassen steht anschliessend eine neue Funktionalität zur Verfügung.

- Szenario Funktionserweiterung:

  Über Mikrowellensensoren kann ein Hindernis um das Fahrzeug detektiert werden. Dies kann sowohl für die Einparkhilfe als auch für Abstandswarnung und Precrash-Meldung verwendet werden. Diese Funktionen werden mit denselben Sensoren ermöglicht. Eine zunächst nur als Einparkhilfe verkaufte Lösung kann durch Softwareupgrade um zusätzliche Funktionen erweitert werden. Eine Anbindung in das Fahrzeug-Netzwerk ist über CAN geplant.

Zu Beginn der Arbeiten in WT4.3 sind diese Szenarien bzgl. ihrer Leistungsfähigkeit zur Validation der Konzepte aus WP2 und WP3 zu analysieren. Eine den verfügbaren Ressourcen und adäquate Auswahl ist zu treffen.

WT4.4 Human Machine Interface

Ein Demonstrator für Human Machine Interface ist in Deutschland nicht geplant. Dieser wird im Gesamtprojekt unter italienischer Führung (Fiat, Magneti Marelli) unter Beteiligung einzelner deutscher Partner realisiert. Damit ist hinsichtlich der Validierung ein Rückfluss ins deutsche Konsortium gewährleistet.

WT4.5 Chassis

Der Validator im Bereich der Fahrwerkselektronik hat zur Aufgabe die Ergebnisse der Arbeitspakete WP2 und WP3 in Bezug auf die Erfüllung der Anforderungen aus diesem Technologiebereich zu validieren. Der Wesentliche nachzuweisende Punkt wird hierbei in einer offenen Systemarchitektur gesehen, in der die Verschiebbarkeit von Funktionsnetzwerken zwischen unterschiedlichen Plattformen (z.B. Steuergeräten) möglich ist. Eine rechtzeitige Bereitstellung von Architekturbeschreibungsmitteln (Sprache und Werkzeugumgebung) sowie einer domänenübergreifenden, einheitlichen Schnittstellenbeschreibung und einer zumindest prototypischen Implementierung der

spezifizierten Middleware als Ergebnisse der Arbeitspakete 1, 2 und 3 ist dabei Voraussetzung.

Um den Nachweis der Verschiebbarkeit von Softwarekomponenten zu erbringen, wird der Demonstrator von 3 verschiedenen deutschen Projekt-Partnern realisiert, wobei unterschiedliche Entwicklungsprozesse zum Einsatz kommen: BMW wird eine Überlagerungslenkung (Steer-by-wire), DaimlerChrysler eine fehlertolerante, redundant ausgeführte Pedaleinheit und Siemens VDO eine elektromechanische Bremse (EMB, brake-by-wire) realisieren.

Der Demonstrator wird in der Realisierung als Laboraufbau konzipiert sein und dadurch ausschließlich diesen Anforderungen genügen.

WP5: Project Management

Die technischen Teilprojekte WP1 bis WP4 werden ergänzt durch das WP5. Dieses Teilprojekt beinhaltet die interne und externe Koordination, das Dokumentenmanagement, das Glossar, die Arbeitsqualität, die Ergebnisverwertung sowie die Sammlung offener Punkte.

1.2 Voraussetzungen des Vorhabens

Um die ambitionierten Ziele zu erreichen, ist eine europäische Zusammenarbeit von Kfz-Herstellern, Zulieferern und Forschungseinrichtungen notwendig. Damit wird eine breite Basis für eine eventuell nachfolgende Standardisierung geschaffen.

Aus diesem Grund ging die Initiative zum Projekt auch von einem europäischen Teilnehmerkreis aus. Das deutsche Konsortium ist Teil des ITEA-Projekts „EAST-EEA“, an dem insgesamt 23 Partner aus Deutschland, Frankreich, Italien und Schweden beteiligt sind.

1.3 Planung und Ablauf des Vorhabens

Die Arbeiten im Vorhaben wurden im Wesentlichen gemäß dem in der Vorhabensbeschreibung vom 10.12.2001 enthaltenen Arbeitsplan durchgeführt. Vor Allem in der Anfangsphase des Projekts gab es zahlreiche Treffen auf Work-Package- und Work-Task-Ebene. Die Ergebnisse wurden in 29 Deliverables dokumentiert.

Zur Steuerung des Projekts wurden die folgenden Gremien installiert:

- Steering Committee (STC):

  Diesem Steuerkreis gehören alle Projektpartner an. Er tagt zweimal jährlich und stellt die höchste Instanz innerhalb des Projekts dar.

- Steering Support Committee (STSC):

  Es unterstützt das STC bei der Steuerung des Projekts und tagt vierteljährlich. Ihm gehören von deutscher Seite die Firmen DaimlerChrysler, BMW, Bosch und SiemensVDO an.

- Externe Projektkoordination:

  Für die Aufgaben des Projektmanagements wurde die Firma Irion Management Consulting (IMC) beauftragt.

Im Rahmen von ITEA wurden insgesamt 3 Projektreviews am 25.09.2002, 22.05.2003 und am 22.06.2004 durchgeführt. Am

21./22.06.2004 fand auf der Flugwerft Schleißheim die Abschlussveranstaltung des Projekts statt.

Daneben war das Projekt EAST-EEA auf den ITEA-Konferenzen 2002 in Amsterdam, 2003 in Leuven und 2004 in Sevilla vertreten.

1.4 Wissenschaftlich-technischer Stand

Im Bereich der Kraftfahrzeug-Elektronik gab es zu Beginn der 90er-Jahre die ersten Ansätze zur Standardisierung von Bussystemen, Protokollen und Standard-Software. Im Jahr 1991 kam das erste Fahrzeug mit einem CAN-Bus (Controller Area Network) [1] auf den Markt. Heute wird dieses Bussystem mit Datenraten bis zu 1 MBit/s von fast allen Kraftfahrzeug-Herstellern zur Vernetzung von Antriebsstrang- und Innenraum-Steuergeräten eingesetzt.

Im Bereich der Telematik hat sich der MOST-Bus (Media Oriented Systems Transport) [2] etabliert, ein optisches Bussystem mit Datenraten bis zu 22 MBit/s. Damit können sowohl Audio- als auch Video-Signale digital zwischen Telematik-Geräten (z.B. Headunit, CD-Wechsler, Soundsystem) übertragen werden.

Für sicherheitskritische Systeme im Fahrwerksbereich ergab sich die Notwendigkeit für ein neues deterministisches und fehlertolerantes Bussystem, das einerseits durch eine Zeitsteuerung die Realisierung verteilter Regelungen über den Bus ermöglicht, auf der anderen Seite durch Mechanismen wie Kanalredundanz und Bus-Guardian die Kommunikation auch bei Leitungsfehlern oder fehlerhaften Steuergeräten aufrecht erhält. Hier hat sich FlexRay [3] mit Datenraten bis zu 10 MBit/s zum De-Facto-Standard entwickelt.

Parallel zu der Entwicklung der Bussysteme gab es ab 1990 erste Überlegungen zur Neustrukturierung der Software auf Steuergeräten. Das erste Ziel war dabei, auch auf den im Kfz eingesetzten Microcontrollern mit beschränkten Ressourcen (8 bis maximal 16 Bit, maximal 2kByte RAM) eine Basisfunktionalität zur Verfügung zu stellen, die aus den Teilen Betriebssystem, Kommunikation und Netzmanagement besteht. Im Jahr 1993 wurde u. A. von BMW, Bosch, DaimlerChrysler, Opel und Siemens das OSEK-Konsortium gegründet, dem ein Jahr später auch die französischen Partner PSA und Renault beitraten. In diesem Rahmen wurde gemeinsam mit anderen Partnern daraus ein heute in der gesamten europäischen Automobilindustrie etablierter Standard geschaffen [4]. Zur Unterstützung zeitgesteuerter Architekturen wurde in 2001 die erste Version eines zeitgesteuerten Betriebssystems (OSEKTime) sowie eines fehlertoleranten Kommunikationsmoduls (OSEK FTCom) vorgestellt.

Interoperability and Portability

Als Folge der sich stark verändernden Randbedingungen (steigende Komplexität, zunehmender Kostendruck, kürzere Entwicklungszeiten, mehr Baureihen, weniger Entwickler) bei der Elektronik-Entwicklung für neue Kraftfahrzeuge ergibt sich die Notwendigkeit, Systeme mehr als in der Vergangenheit Baureihen übergreifend wieder zu verwenden. Eine wichtige Voraussetzung für die Wiederverwendung ist eine Strukturierung der Anwendungen in Teilfunktionen, die über standardisierte Kommunikationsprotokolle miteinander kommunizieren.

Diese Portabilität und Interoperabilität soll durch eine so genannte Middleware-Schicht erreicht werden. Dazu existieren unterschiedliche Ansätze, die im Rahmen von EAST-EEA untersucht und harmonisiert werden sollen.

- **TITUS**

  Auf OSEK aufbauend wurde von DaimlerChrysler die TITUS Software-Architektur entwickelt, die auf einem Client-Server-Modell basiert und sich für den Entwurf von Steuerungsfunktionen im Kraftfahrzeug eignet [5, 6, 7]. Diese Funktionen können gemäß der TITUS-Methodik in einem Werkzeug entworfen werden und in einem separaten Prozessschritt auf eine, für die jeweilige Fahrzeugbaureihe optimierte Steuergerätetopologie abgebildet werden [8].

- **EEA (Embedded Electronics Architecture)**

  Von den französischen Projektpartnern PSA, Renault, Valeo und SiemensVDO (F) existieren ebenfalls der Entwurf einer Middleware für vernetzte Fahrzeugsysteme sowie der Vorschlag für eine Architektur-Beschreibungssprache „Architecture Implementation Language (AIL)“ [9].

- **Multimedia-Middleware**

  Aus der Unterhaltungs- und Computerindustrie sind verschiedene Ansätze zur Vereinheitlichung und Abstraktion von Kommunikationssystemen gegenüber den darauf zugreifenden Applikationen bekannt. Beispiele für derartige Architekturen finden sich mit JINI auf der Basis der Programmiersprache JAVA, mit HAVi auf Basis des Kommunikationsstandards IEEE1394 (FireWire) und mit UPnP auf Basis der Metasprache XML.

  Im ITEA-Projekt VHE-Middleware wird die Konzeption, Entwicklung und Erprobung einer Software-Infrastruktur, bestehend aus kooperierenden Diensten, für den Heimbereich verfolgt [10]. Dies erfolgt unter besonderer Berücksichtigung der Mobilität des Benutzers und mobiler Endgeräte (z.B. Handy, Palm, PocketPC, PDA, Notebook, ...).

Weitere für das Projekt relevante Ansätze sollen im Verlauf der Projektarbeit identifiziert und bewertet werden.

Development and Validation Tools

Im Bereich der Entwicklungstools für eingebettete elektronische Systeme existieren heute Werkzeuge, die bestimmte Aspekte des Entwicklungsprozesses wie z.B. Requirements Engineering oder Test abdecken. Zur Realisierung einer durchgängigen Entwicklungs-Toolkette fehlt insbesondere eine Architekturbeschreibungssprache, die über alle Phasen des Entwicklungsprozesses eine konsistente Datenhaltung sicherstellt und entsprechende Austauschformate zwischen den Tools zu Verfügung stellt.

Auf diesem Gebiet sind einige interessante Ansätze bekannt:

- **Produktlinien-Ansatz**

  Eine der Grundlagen für den Produktlinien-Ansatz ist „The Product Line Practice Initiative“ des Software Engineering Institutes (SEI) der Carnegie Mellon Universität (CMU) in den USA [11]. Das SEI hat in [12] eine Reihe von Informationen zum Produkt-

familien-Engineering zusammengestellt. Es hat so genannte „Practice Areas“ definiert und beschrieben, die für Produktfamilien wichtig sind. Diese Beschreibungen sind ausführlich, allerdings geben sie nur sehr wenige Verfahren und Methoden an und zählen nur auf, welche Dinge wichtig sind.

- **ITEA-ESAPS**

  Das europäische ESAPS-Projekt ist das bisher größte Projekt zum Thema Produktfamilien, und die dort getätigte Arbeit repräsentiert im Wesentlichen den Stand der Technik in diesem Bereich. Dies wurde u.a. auf der First Product Line Conference [13] im August 2000 bestätigt, die einen guten Überblick über den State-of-the-Art in der Produktfamilienentwicklung bot.

- **ITEA-DESS**

  Ziel dieses Projektes ist die Definition einer innovativen, komponentenbasierten Software-Entwicklungsmethodik für eingebettete Echtzeitsysteme auf der Basis von UML-Konzepten, die Erstellung unterstützender Werkzeugumgebungen durch die Integration moderner (UML-)Werkzeuge und der Nachweis der Zweckmäßigkeit der Methodik durch die Implementierung mehrerer Testfälle zur Validierung [14]. Unter eingebetteten Systemen werden in diesem Projekt hauptsächlich Systeme verstanden, die aus Microcontrollern mittlerer bis höherer Leistungsfähigkeit (16-32bit) bestehen und deren Anwendung in den meisten Fällen Echtzeit-Charakter hat, also das Einhalten von festen, periodischen Terminen im Bereich von 1-100ms.

- **Projekt SETTA**

  Im Rahmen der Definition neuer Architekturen für X-by-wire-Systeme, d. h. sicherheitskritischer Fahrwerkssysteme ohne mechanische Rückfallebene, arbeiteten u. A. die Projektpartner DaimlerChrysler, SiemensVDO (D) und Renault in einem EU-Projekt namens SETTA [15, 16], welches sich mit dem Entwurfsprozess verteilter, zeitgesteuerter Systeme beschäftigt. Die Arbeiten beschränken sich in dem Projekt im Wesentlichen auf den simulationsbasierten Entwurf. Mit Hilfe von EAST-EEA sollen offene Punkte, wie z.B. das Austauschformat von Designinformationen und die Integration der Modelle in reale Hardware geschlossen werden.

Über diese Aktivitäten in konkreten Forschungsprojekten hinaus gibt es zahlreiche Initiativen zur Integration von unterschiedlichen Entwicklungstools zu einer durchgängigen Toolkette.

1.5 Zusammenarbeit mit anderen Stellen

Soweit es zur Informationsgewinnung bezüglich der oben genannten Projekte und Initiativen über deren Publikationen hinaus notwendig war, wurden Gespräche mit deren Vertretern geführt.

Eine darüber hinausgehende Zusammenarbeit mit anderen Stellen fand nicht statt.

2 Eingehende Darstellung

Während Kapitel 1 die Kurzdarstellung aus der Sicht des Gesamtprojekts gibt, steht in diesem Kapitel der spezifische Projektanteil der ZF Friedrichshafen AG im Vordergrund.

Die Rolle von ZF in der Automobilindustrie ist historisch die eines international agierenden Komponenten-Zulieferers. Zunächst waren wir im Bereich Antriebsstrang speziell mit Getrieben und Achsen vertreten. Seit mehreren Jahren ist das Unternehmen aber auch mit Komponenten wie Lenkungen, Dämpfern oder Stabilisatoren auf dem Markt vertreten und spielt damit auch zunehmend als Komponenten-Zulieferer in der Domäne Fahrwerk eine Rolle.

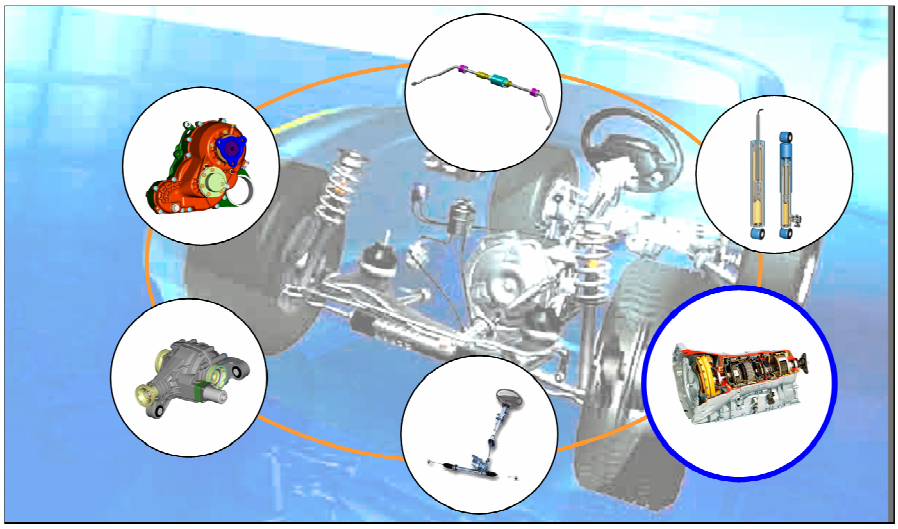

Abbildung 1: ZF-Produkte

Dementsprechend haben wir im Elektronik-Bereich bis an die Anfänge zurückreichende Erfahrungen mit Getriebesteuerungen.

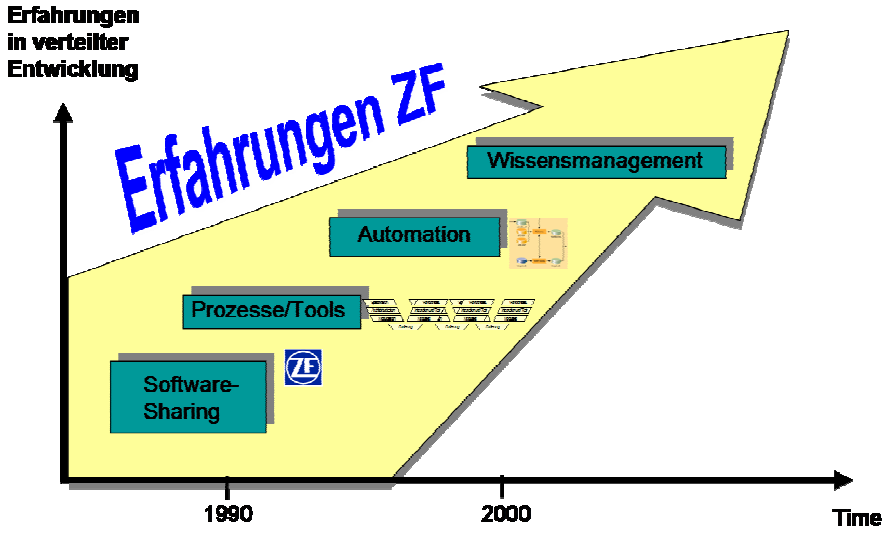

Abbildung 2: Erfahrungen mit Getriebesteuerungen

Seit längerem schon ist ein Anstieg des Elektronik- und speziell des Software-Anteils im Fahrzeug zu beobachten. Im Bereich Antriebsstrang-Elektronik war dies in der Vergangenheit vor allem auf

die Erweiterung von Einzelaggregat-Steuerungen (Motor, Getriebe), sowie auf die Entwicklung von neuen, elektronisch gesteuerten Aggregaten (Differenzialsperre, Verteilergetriebe, Starter-Generator), und auf die zunehmend erforderliche Synchronisation der Einzelsteuerungen, zurückzuführen. Ein Kennzeichen dieser Phase war, dass die Einzelsteuerungen im Wesentlichen noch unabhängig voneinander entwickelt werden konnten. Neuere Trends erfordern, auch für ZF, eine zunehmende Vernetzung der Funktionen und der Steuergeräte. Prognosen stellen auch weiterhin einen steigenden Software-Anteil im Fahrzeug in Aussicht. Diese Zukunftsperspektive bedeutet, dass einerseits die Entwicklungsaufgaben stärker unter den an der Entwicklung eines Kfz beteiligten Firmen abzustimmen sind, und dass andererseits auch mit einer weiteren Verschärfung des Spannungsdreiecks Zeit – Kosten – Qualität zu rechnen ist. Denn trotz der mit zunehmendem Software- und Kommunikations-Anteil stark ansteigenden Komplexität müssen Zeit und Kosten in einem annehmbaren Rahmen bleiben, und zusätzlich muss die erforderliche Qualität der Software über die Lebensdauer eines Kfz sichergestellt sein. Speziell im Antriebsstrang wird mit weiterreichenden Anforderungen bezüglich Interaktion der Aggregat-Funktionen, sowie mit der Vernetzung mit den anderen Domänen gerechnet. Um den damit verbundenen Anforderungen gerecht zu werden, müssen neue Technologien zum Einsatz kommen. Herstellerübergreifende Standards, die Individualisierung und Wertschöpfung eines Produktes nicht einschränken, bilden hierfür eine wichtige Grundlage. Nicht zuletzt, weil sie einen Hersteller übergreifenden Einsatz von Entwicklungswerkzeugen im Prinzip erst ermöglichen.

Vor diesem Hintergrund hat sich die ZF Friedrichshafen AG an dem Verbund-Forschungsprojekt EAST-EEA beteiligt, um einerseits die notwendige standardisierbare, offene Software-Architektur gemeinsam mit den Projektpartnern auf den Weg zu bringen, andererseits aber auch, um die speziellen Anforderungen aus dem Antriebsstrang-Bereich in das Projekt einzubringen und zu vertreten.

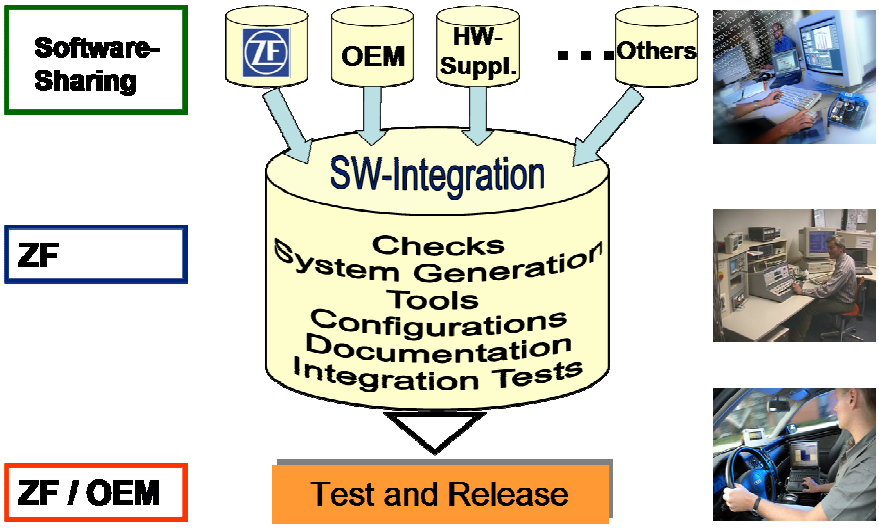

Abbildung 3: Erfahrungen im Software-Sharing mit OEM's

In der Software-Entwicklung hat die ZF mittlerweile langjährige Erfahrungen in der Objekt- und Komponenten-orientierten

Programmierung [17, 20, 22], mit Software-Sharing, mit virtuellem und rapid Prototyping [18, 19], mit Test und Prozessen [21, 23], sowie mit dem Einsatz von Entwicklungstools. Dabei stehen die besonderen Bedingungen von Echtzeitsystemen stets im Vordergrund.

In der Vorbereitungsphase wurde die Projektbeteiligung entsprechend den vorhandenen Erfahrungen geplant. Insbesondere legten die Erfahrungen im Bereich der Umsetzung für Powertrain-Aggregate zwingend eine starke Powertrain bezogene Beteiligung in den Teilprojekten WP2 und WP4 nahe, wo es letztlich um die Architektur-Spezifikation und deren Integration und Test ging.

In diesen Teilprojekten übernahm ZF schließlich auch leitende Rollen in den Arbeitspaketen. Demgegenüber sah man mitwirkende Rollen in den Teilprojekten WP1 und WP3 vor.

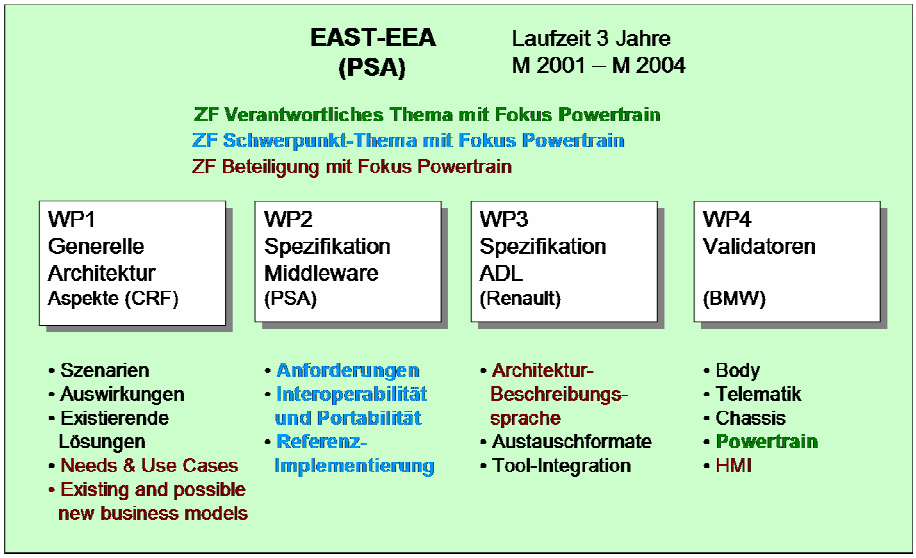

Abbildung 4: ZF-Beteiligung am EAST-EEA Projekt

Die Beteiligungen an den einzelnen Teilprojekten können dem Bild oben entnommen werden. Projektmanagement-Aufgaben wurden im Zuge der internen Koordination, der Mitarbeit im STC und der Leitung von Arbeitspaketen im Rahmen des WP5 wahrgenommen.

Die inhaltliche Bearbeitung stellte in verschiedenerlei Hinsicht eine große Herausforderung für ZF, aber auch für alle anderen Beteiligten, dar.

- Das Projekt integrierte alle Fahrzeug-Domänen
    - Body
    - Powertrain
    - Chassis
    - Telematik
    - HMI

- In den Domänen wurde das Thema Software-Architektur entsprechend den wechselnden Anforderungen sehr unterschiedlich behandelt. Die Unterschiede zwischen Body, Powertrain und Chassis (komplexes Hardware-Netzwerk, hohe Echtzeit- und Sicherheitsanforderungen) sind ebenso wie die zwischen Telematik und HMI (Bedienerfreundlichkeit durch z. B. Plug&Play, hoher Datendurchsatz) jeweils eher im Detail zu suchen und auch zu finden. Hingegen gibt es deutliche Unterschiede zwischen diesen beiden Domänen-Gruppierungen, die sich natürlich auch in den Anforderungen an die Software-Architektur und deren Realisierung wieder finden.

- Am Projekt waren verschiedene Interessengruppen beteiligt:
  - Fahrzeughersteller
  - Zulieferer
  - Tool-Hersteller
  - Forschungseinrichtungen
- Die internationale Beteiligung erforderte ein hohes Maß an terminologischer Synchronisation, da zwar eine Reihe von Vorarbeiten, überwiegend national ausgerichtet, vorhanden war, diese aber im Detail doch stark voneinander abwichen. Zu den technischen Randbedingungen kamen durch die internationale Beteiligung auch sprachliche Hürden, die für eine Verständigung zu zahlreichen Details zwischen Deutschen, Franzosen, Italienern und Schweden zu überwinden waren.

Im Folgenden werden Ziele und Beiträge der ZF an den Arbeitspaketen detaillierter beschrieben.

Neben den Arbeiten zu den Arbeitspaketen wurden Beiträge für eine Sammlung von Begriffsdefinitionen erbracht, mit der letztlich ein gemeinsames Verständnis unter den Partnern zur Sache entwickelt wurde. Das Ergebnis ist öffentlich verfügbar unter dem Titel [Glossary_73_040218.pdf].

Alle Ergebnisse von EAST-EEA sind in den in Kapitel 1.3 erwähnten Deliverables dokumentiert. An einem Teil der Deliverables aus den Teilprojekten WP1, WP2 und WP3 hat ZF mitgearbeitet, der Großteil der auf die Powertrain-Domäne bezogenen Deliverables aus den Teilprojekten WP2 und WP4 wurde mit großer Beteiligung seitens ZF erstellt. Da alle Deliverables jeweils Informationen von mehreren Partnern beinhalten, unterliegt deren Veröffentlichung den im Rahmen des Projekts vereinbarten Restriktionen.

Referenzen auf Deliverables werden im Folgenden in der Form [Dx.y] angegeben. Darin entspricht x der Nummer des Teilprojekts (Work Package (WP)) und y einem laufenden Index, aber i. a. nicht der Nummer des Arbeitspakets (Work Task (WT)).

2.1 Erzieltes Ergebnis

WP1: General Aspects

Die Beteiligung von ZF am Arbeitspaket WP1 erfolgte mit der Zielsetzung, gemeinsam mit den Partnern Rahmenbedingungen für die Einführung der Architektur zu erarbeiten und die Auswirkungen einer Einführung der Architektur abzuschätzen. Insbesondere beteiligten wir uns an der Anforderungsanalyse, an der Analyse von Verfahren zum Umgang mit geistigen Eigentumsrechten und zukünftigen Geschäftsmodellen und an der Ableitung der zukünftigen Referenzarchitektur. Auf die Arbeiten und Ergebnisse wird in den folgenden Arbeitspaket-Betrachtungen etwas tiefer eingegangen.

WT1.2 – Definition der Allgemeinen Anforderungen (Requirement-Analyse)

In diesem Arbeitspaket verfolgten wir das Ziel, Anforderungen aus Anwendungsfällen zu erarbeiten, die heute typischerweise im Bereich des Antriebsstrangs-Management von Fahrzeugsystemen auftreten. Neben der Berücksichtigung der Prüfbarkeit der einzelnen Komponenten in Entwicklung und Serie sollte insbesondere den hohen Anforderungen an Echtzeit und den Sicherheitsaspekten besondere Bedeutung beigemessen werden.

Dementsprechend konzentrierte sich auch unsere Arbeit im Arbeitspaket WT1.2 auf die Definition von Anwendungen, die einerseits für Getriebesteuerungen typisch sind und die andererseits auch die besonderen Anforderungen an Echtzeitfähigkeit und Sicherheit unterstreichen. In den Ergebnissen wurden Anforderungen an die Architektur aus der Sicht eines Zulieferers formuliert. Wesentliche Anforderungen ergaben sich aus dem Wunsch nach einheitlichen Schnittstellen zu verschiedenen OEM-Systemen, um den Adaptionsaufwand zu verringern, wobei aber die Diversifikation der Implementierungen nicht eingeschränkt werden sollte. Um die unterschiedlichen Rollen der Projektbeteiligten zu verdeutlichen und damit auch die Notwendigkeit und Relevanz von spezifischen Anforderungen zu untermauern, haben wir uns an der Analysephase hierzu aktiv beteiligt. Die Ergebnisse sind dokumentiert in [D1.2].

WT1.4 – Verfahren zum Umgang mit geistigen Eigentumsrechten und zukünftige Geschäftsmodelle

Gemeinsam mit den Partnern setzten wir uns hier das Ziel, Auswirkungen auf bestehende Geschäftsmodelle zu untersuchen und daraus einen Vorschlag für ein neues Geschäftsmodell abzuleiten. Gründe dafür wurden vor allem gesehen in dem zu erwartenden größeren Austausch von Funktionssoftware, von notwendigen Tools und von Know-How über mehrere an einer Kfz-Entwicklung beteiligte Software-Sharing-Partner hinweg. Der Rolle von ZF entsprechend sollten hier insbesondere die Interessen der Zulieferindustrie mit vertreten werden.

Im Vordergrund der Aktivitäten standen bei der Umsetzung die Analyse von heute in der Automobilindustrie üblichen

Geschäftsmodellen, und daran anschließend die beispielhafte Erarbeitung eines neuen Geschäftsmodells, welches den geänderten Randbedingungen gerecht werden und die Interessen der Beteiligten berücksichtigen sollte. Dabei wurden technische und rechtliche Aspekte berücksichtigt. Als Haupttreiber wurde die Kommerzialisierung von Software gesehen, da erwartet wird, dass Software zunehmend als eigenständiges Produkt auf dem Markt erscheinen wird. Die Ergebnisse sind dokumentiert in Deliverable [D1.8], das die Analyse zusammenfasst, und in Deliverable [D1.10], welches das Beispiel für ein zukünftiges Geschäftsmodell enthält.

WT1.5 – Ableitung der zukünftigen Referenzarchitektur

Unter Berücksichtigung von bereits vorliegenden Ergebnissen sollte in diesem Arbeitspaket die zukünftige Referenzarchitektur beschrieben werden. Dazu wurden gemeinsam mit den Partnern bekannte Technologien aus anderen Software-Bereichen (z. B. CORBA), aus dem Automotive-Bereich bekannte nichtfunktionale Randbedingungen (z. B. Sicherheitsanforderungen) und bekannte funktionale Anforderungen (z. B. das Mode Management) und sonstige Treiber (z. B. die Kostenanforderungen) identifiziert. Von allen diesen Größen nahm man an, dass sie die zukünftige Architektur maßgeblich beeinflussen würden. Ansatz war die Betrachtung des Gesamtfahrzeugs aus verschiedenen Sichten, z. B. aus der eines Fahrzeugbesitzers oder aus der Sicht der Domänen.

Die nachfolgend abgebildete Struktur fasst die Verbindungen zu anderen Teilprojekten und Arbeitspaketen zusammen und gibt eine grobe Einteilung der Inhalte von WT1.5 wieder.

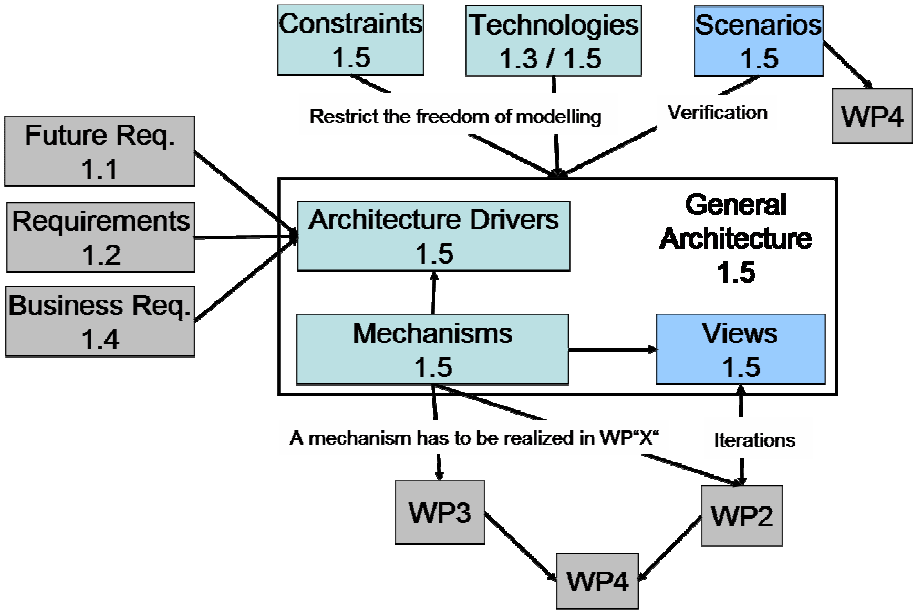

Abbildung 5: Strukturierung WT1.5

Unsere Beiträge wurden aus der Sicht von typischen Powertrain Anwendungsfällen erarbeitet und finden sich in dem Deliverable [D1.5] wieder.

WP2: Interoperability and Portability

Die gemeinsam definierten Ziele in diesem Arbeitspaket zielten zunächst darauf ab, eine Anforderungsspezifikation für eine fahrzeugweite Middleware- und Kommunikations-Schicht zu erstellen, deren Schnittstellen zu spezifizieren, die Ergebnisse mit existierenden Lösungsansätzen zu harmonisieren, und darauf basierend eine prototypische Implementierung vorzunehmen.

Für die Spezifikation betrachtete man es als essenziell, realistische Annahmen über die Einsatzumgebung zu treffen. So sollte als System beispielsweise nicht nur das Getriebe allein betrachtet werden, sondern das gesamte Fahrzeug. Wir verfolgten hier das Ziel, uns an der Spezifikation der gemeinsamen Middleware, unter Berücksichtigung der besonderen Anforderungen des Antriebsstrangs, zu beteiligen.

WT2.1 General Implementation Framework

Als Basis wurde hier zunächst eine Liste mit Grundfunktionalitäten für Software von eingebetteten Systemen im Fahrzeug erarbeitet. Diese Grundfunktionalitäten wurden zu Gruppen wie Datenaustausch, Software Download oder System-Konfiguration und weiteren Gruppen zusammengefasst. Bezüglich des Datenaustausches ergaben sich dann wesentliche Anforderungen wie

- Transparenz zwischen Steuergeräte-interner und Steuergeräte-externer Kommunikation oder
- Transparente Kommunikation zwischen Software-Komponenten auf Ebene der Anwendung innerhalb einer Domäne.

Durch eine Analyse der Grundfunktionalitäten war es möglich, die Anforderungen an die Architektur zu formulieren. Um diese Arbeit effektiv durchführen zu können, hat man kleine Arbeitsgruppen gebildet. In den Arbeitsgruppen wurde basierend auf den vorhandenen Ergebnissen zum einen Domänen übergreifend auf die Anforderungs-Spezifikation und zum anderen Domänen spezifisch auf die Architektur-Spezifikation hingearbeitet.

Nachdem die wesentlichen Anforderungen gesammelt waren, stellte sich in den Detail-Betrachtungen zunehmend heraus, dass die Anforderungen sehr verschieden waren und in erheblichem Maße auch Domänen spezifische Ausprägung hatten. Mit einer Kategorisierung der Anforderungen wurde versucht, dies auf eine gemeinsame Basis zu stellen, aber es wurde schnell offensichtlich, dass an eine gemeinsame Detail-Spezifikation einer Middleware mit dem zusätzlichen Ziel der Implementierung nicht zu denken war. Als Beispiel möge die Verschiebung von Software-Komponenten zwischen Steuergeräten dienen. Während für die Domänen Chassis und HMI die Möglichkeit einer Verschiebung zur Laufzeit (dynamische Verschiebung), erforderlich für z. B. Plug&Play-Anwendungen, vorausgesetzt wurde, musste dies in den anderen Domänen u. A. aus Sicherheits- und Ressourcengründen von vornherein ausgeschlossen werden. Die Verschiebung von Software-Komponenten sollte hier durch eine Re-Konfiguration der Software

erfolgen (statische Verschiebung). In weiteren Details traten auch zunehmend Unterschiede zwischen den Domänen Body, Antriebsstrang und Chassis zu Tage. Diese waren wesentlich auf die Domänen-spezifisch realistisch gewählten Anwendungsfälle zurückzuführen (Chassis: Ausfallsichere Echtzeit-Kommunikation, Body: Viele Steuergeräte und Software-Download über eine Telematik-Anwendung, Powertrain: Echtzeitanforderungen bei CAN-Kommunikation, sowie die Unterstützung für Mode Management und Kalibrierung). Die Beiträge von ZF finden sich im Deliverable [D2.1] wieder, welches die Anforderungen an die Middleware für alle Domänen zusammenfasst.

Bezüglich der Architektur-Spezifikationen ergab sich folgende Situation. In den Domänen Antriebsstrang, Body und Chassis wurden i. W. die OSEK Standards entsprechend dem jeweils benötigten „Service-Level“ (Antriebsstrang, Body: OSEK-COM 3.0, OSEK-OS; Chassis: OSEK-FTCOM, OSEKtime-OS) in die Betrachtungen als existierende Lösungen einbezogen. Als Langzeitperspektive wurde auch für Powertrain eine Option auf OSEK-FTCOM und OSEKtime-OS offen gehalten, die aber im Rahmen des Projekts nicht weiter verfolgt wurde.

Weitere existierende Lösungen, die als Standards oder Quasi-Standards schon vorhanden sind, wurden ebenfalls, aber Domänen spezifisch berücksichtigt. In der Domäne Powertrain waren das CCP, OSEK TL, OSEK NM, KWP2000, wie dargestellt in der folgenden Skizze der Architektur:

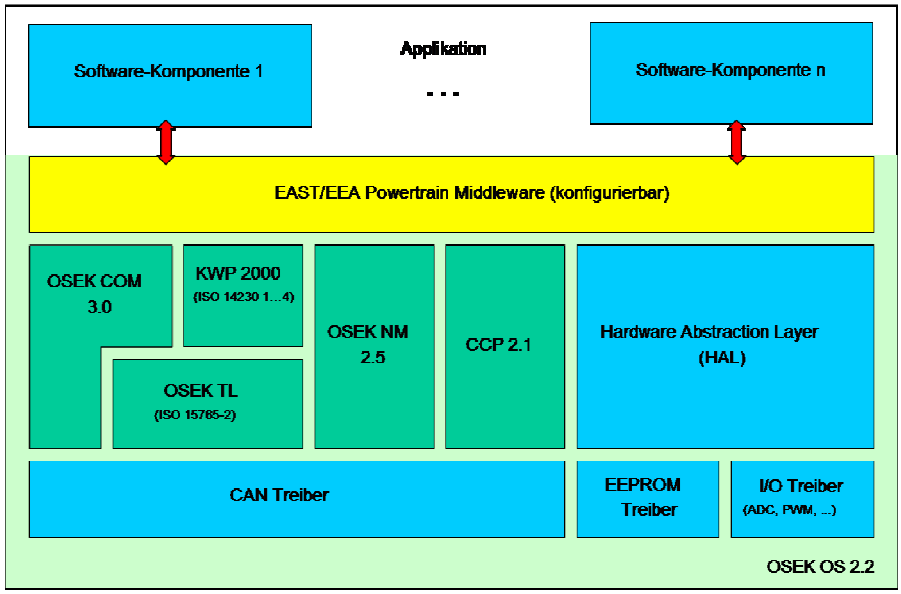

Abbildung 6: Architektur der Powertrain-Domäne

Als neue Schicht kommt eine konfigurierbare Middleware hinzu, die die Software-Komponenten der Anwendung von der ECU-spezifischen Basis-Software (bezeichnet die Middleware und alle Komponenten darunter) abstrahiert. Wesentlich ist, dass alle Applikations-Software-Komponenten ausschließlich auf Schnittstellen der Middleware zugreifen und nicht direkt interagieren oder direkt Services z. B. der Treiber-Komponenten verwenden.

Dabei ist es unerheblich, ob z. B. die Middleware Schnittstellen eines Standard-Betriebssystems zur Verfügung stellt oder diese direkt verwendet werden. Die Ergebnisse wurden für das gesamte Projekt in Deliverable [D2.2] zusammengefasst.

Neustrukturierung des Teilprojekts WP2

Wegen der klar ersichtlichen Domänen-Ausrichtung der Anforderungen (siehe Beschreibung WT2.1) hat das STC mit dem Abschluss der Anforderungsanalyse-Phase beschlossen, die Middleware zunächst in einer ersten Phase für die einzelnen Domänen zu spezifizieren und zu implementieren und die Ergebnisse daran anschließend hinsichtlich deren Domänen übergreifenden Anwendbarkeit in einer zweiten Phase zu analysieren, um schließlich die Domänen-Spezifikationen zu harmonisieren. Auf diese Weise konnten die Differenzen ausgeglichen und eine funktionierende Arbeitsteilung hergestellt werden.

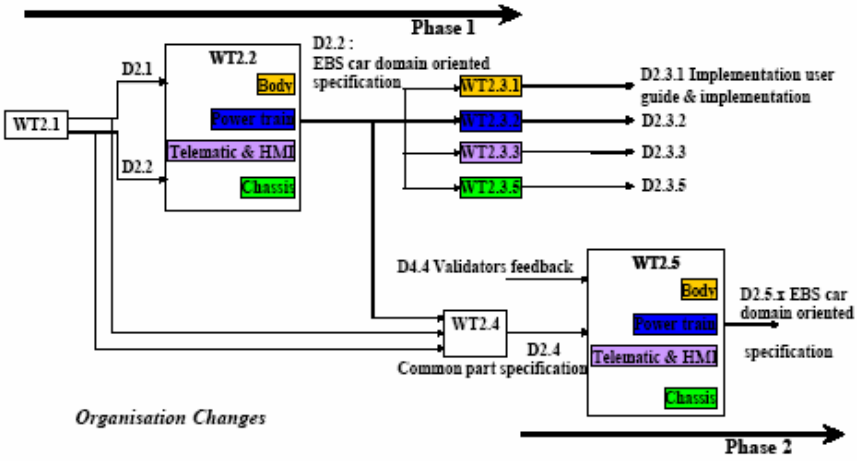

Abbildung 7: Neustrukturierung von WP2

Mit der Integration von existierenden Lösungen konnte das ursprünglich vorgesehene Arbeitspaket „Communication Layer Specification“ mit der Middleware-Spezifikation zusammengefasst werden. Mit der Implementierung wurde in gleicher Weise verfahren. Um die Powertrain-Ergebnisse in die gemeinsame Spezifikation einfließen zu lassen, sahen wir eine entsprechende Beteiligung an dem neuen Arbeitspaket WT2.4 „Common Part Specification“, sowie die abschließende Überarbeitung der Middleware-Spezifikation vor.

WT2.2 Middleware & Communication Layer Specification

Der Beitrag von ZF zu diesem Arbeitspaket sollte die im Powertrain relevanten Aspekte beleuchten. Als wesentlich wurden die Echtzeiteigenschaften erachtet, weil diese die regelungstechnischen Anwendungsfälle in einem verteilten System erst ermöglichen. Da man letztlich die Domänengrenzen nur exemplarisch überschreiten wollte (Download-Szenario zwischen Telematik und Body), war die Betrachtung eines heterogenen Netzwerks, in dem die Kommunikation eine höhere Komplexität aufweist, nicht erforderlich.

Aufgrund der Festlegungen aus Arbeitspaket WT2.1 sollte die Middleware zunächst so spezifiziert werden, dass Interoperabilität und Portabilität für Software-Komponenten der Powertrain-Applikation erzielt werden. Mit dem Ergebnis wollte man dann in die Domänen übergreifende Analyse der Ergebnisse eintreten.

Basierend auf den Ergebnissen des Arbeitspakets WT2.1 wurde für die Domäne Powertrain eine Middleware-Spezifikation erarbeitet. Dazu wurden mehrere Zuständigkeiten der Middleware als Themen definiert. An allen Themen hat sich ZF wesentlich beteiligt:

- Transparente Kommunikation
- Kalibrierung von Applikationsparametern
- Applikationsweites Mode Management
- Transparenter Zugriff auf persistente Speicher
- Zentrale Diagnose-Instanz

Die Integration der Ergebnisse in das Spezifikationsdokument erfolgte durch ZF.

Die transparente Kommunikation ermöglicht einen transparenten Datenaustausch zwischen Applikations-Software-Komponenten.

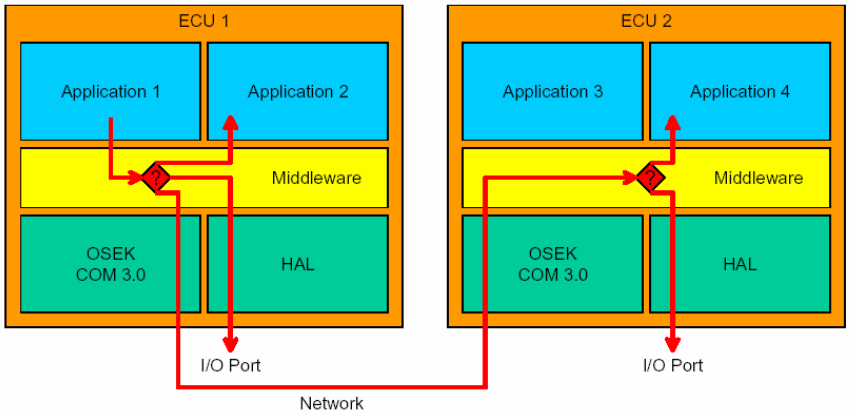

Abbildung 8: Beispiel Transparente Kommunikation

Als Funktionalität der Middleware stellt die transparente Kommunikation beispielsweise sicher, dass ein Signal über fest definierte Schnittstellen von einer zu einer anderen Applikations-Software-Komponente gelangt, und zwar unabhängig davon, auf welchem Steuergerät diese Komponenten lokalisiert sind.

Das Kalibrieren dient der Anpassung von Software an die reale Einsatzumgebung. Eine genauere Beschreibung findet sich in [A]. Für eine Kalibrierung müssen Ablage und Zugriff auf Applikationsparameter unabhängig vom verwendeten Applikationsverfahren sein. In der Architektur wurde für die Kalibrierung als existierende Lösung das CAN Calibration Protocol (CCP) vorgesehen, das mittlerweile quasi Standard ist. In der nachfolgenden Abbildung sind die logischen Zusammenhänge, dargestellt durch rote Pfeile, und die beteiligten Architekturkomponenten zu sehen. Insbesondere sind die logischen Zusammenhänge zwischen MCD-System, Kalibrierungs-Treiber,

Applikations-Software-Komponenten und Speicher sichtbar, wie sie in heutigen Lösungen schon existieren.

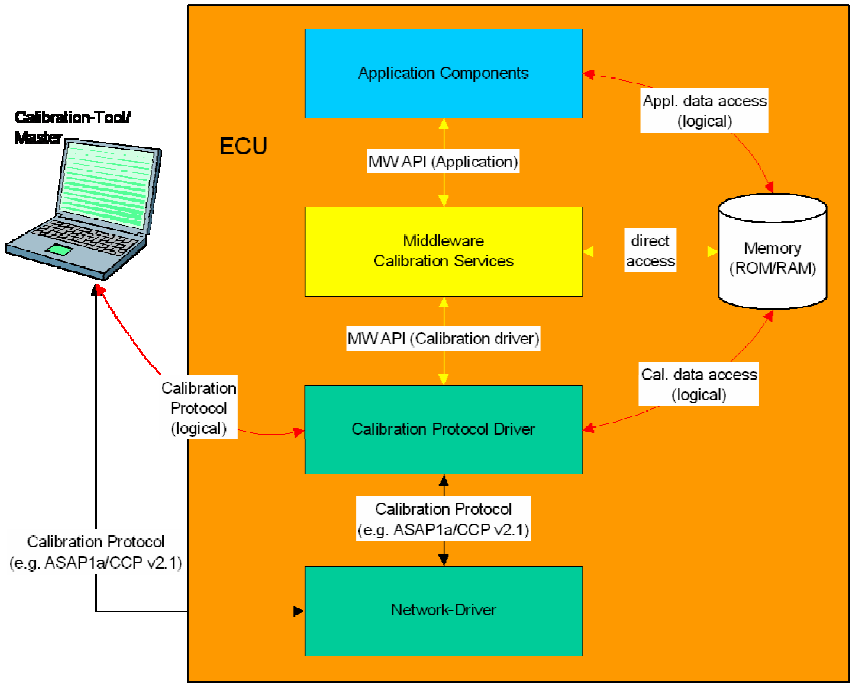

Abbildung 9: Beispiel Parameter-Kalibrierung

Die Middleware vereinheitlicht den Zugriff auf den Speicher, in dem die Parameter liegen. Hierzu bietet die Middleware der jeweiligen Applikations-Software-Komponente Schnittstellen, mit denen sie auf ihre Parameter zugreifen kann (MW API - Anwendung). Dem Kalibrierungs-Treiber muss der Zugriff auf die Parameter gewährt werden, damit er die vom MCD-System angeforderten Aufgaben ausführen kann. Auch hierzu werden von der Middleware Schnittstellen bereitgestellt (MW API - Kalibrierungstreiber), die aber neben dem Zugriff auf die Parameter, auch den Zugriff auf den restlichen Speicherbereich ermöglichen. Dies ist nötig, da das CCP auch für das Monitoring von Messgrößen verwendet wird.

Heutige Applikationen lassen i. d. R. noch keine Synchronisation des Betriebsmodus auf klar definierte Weise zu.

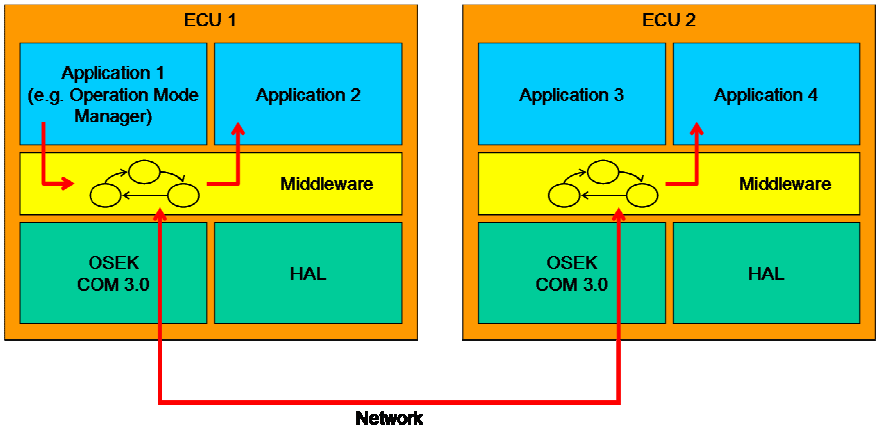

Abbildung 10: Beispiel Operation Mode Management

Die Implementierung eines gemeinsamen Operation Mode Management für alle Anwendungs-Software-Komponenten ermöglicht die Koordination des Ausführungsmodus der Gesamt-Anwendung. Ein Harmonisierungs-Mechanismus, den OSEK NM zur Verfügung stellt, dient dazu, einen Schlafmodus bezüglich der Netzwerk-Kommunikation zu aktivieren bzw. zu deaktivieren. Das ermöglicht zwar in gewissen Grenzen eine Koordination, erlaubt aber beispielsweise nicht, eine Startup-Phase zwischen Teilen der Anwendung zu synchronisieren.

Die OSEK COM als Communication Layer bietet in der Version 3.0 schon eine gewisse Kommunikations-Transparenz mit zusätzlichen Eigenschaften wie „Bit-Ordering“, „Notification“ oder Filterung. Deshalb waren folgende Details der Spezifikation besonders zu beachten:

- Die Transparenz hat man nur für die Kommunikation zwischen Applikations-Software-Komponenten und nicht etwa für einen transparenten Speicherzugriff.
- Die Daten-Repräsentation nutzt einen „byte-ordering“-Mechanismus, deshalb werden implizit entsprechende Anforderungen an die Kohärenz von Datenspezifikationen gerichtet.
- Client/Server-Mechanismen werden nicht unterstützt.

Dementsprechend war die Middleware mit den für die Domäne Powertrain geforderten Grundfunktionalitäten unter Berücksichtigung der festgestellten Eigenschaften der OSEK COM zu spezifizieren.

Für das Hardware Abstracion Layer existierte eine Spezifikation der HIS, die IO-Library. Diese liefert lokalen Anwendungs-Software-Komponenten eine Abstraktion für typische Steuergeräte IO wie Digitale/Analoge Ports oder PWM. Für Komponenten wie Peripherie-Treiber gab es zum selben Zeitpunkt noch keine standardisierte Spezifikation, aber firmenspezifische Lösungen, die als Basis-Software-Komponenten integriert werden konnten. Detaillierte Ausführungen zu den oben skizzierten Themen sind in den Deliverables [D2.3-Powertrain] und [D2.7-Powertrain] enthalten. Für die weiteren Themen wie den transparenten Zugriff auf persistente Speicher oder die Organisation einer zentralen Diagnose-Instanz wurden ebenfalls entsprechende Spezifikationen ausgearbeitet.

WT2.3 Middleware & Communication Layer Implementation

Die Implementierung sollte einerseits der Überprüfung der gefundenen Konzepte dienen, andererseits führt eine Implementierung auch dazu, dass eine Spezifikation einen Level erreicht, der i. d. R. eine Erst-Implementierung erlaubt.

Die Middleware realisiert eine Vermittlungsschicht zwischen Applikations-Software und Basis-Software und bietet dadurch eine standardisierte Laufzeitumgebung für die Applikations-Software. Die Applikations-Software-Komponenten haben nur noch eine Schnittstelle zur Middleware. Insbesondere dürfen keine direkten

Zugriffe von Applikations-Software-Komponenten auf Hardware oder Treiber-Funktionen und kein direkter Datenaustausch zwischen Applikations-Software-Komponenten stattfinden.

Die folgenden allgemeinen Forderungen werden an eine implementierte Middleware gerichtet:

- Die Kommunikationszeiten dürfen gegenüber heutigen Implementierungen nicht nennenswert ansteigen, um die Echzeitfähigkeit der Architektur sicher zu stellen.
- Es muss eine Ressourcen schonende Implementierung möglich sein.
- Der Overhead beim Zugriff auf die Basis-Software muss gering sein.

Daraus ergeben sich die folgenden speziellen Anforderungen an die Realisierung der Middleware:

- Die Middleware ist statisch konfiguriert.
- Die Middleware ist automatisch generiert aus der Gesamt-Systembeschreibung.
- Die Middleware ist keine vorkompilierte Library, sondern auf jedem Steuergerät läuft eine individuelle Middleware-Variante.

Weiterhin setzt die Middleware auf Standardkomponenten auf, deren Verfügbarkeit entsprechend Spezifikation aber zumindest zum Zeitpunkt der Bearbeitung nicht gegeben war, wie z. B. die OSEK COM 3.0. Dieses Problem wurde dadurch gelöst, dass man Komponenten des für die Integration verwendeten Tools (DaVinci Tool Suite der Fa. Vector, siehe [24]) erweitert und angepasst hat.

Der Konfigurations- und Build-Prozess, der für die Middleware-Implementierung aufgesetzt wurde, ist 2-stufig.

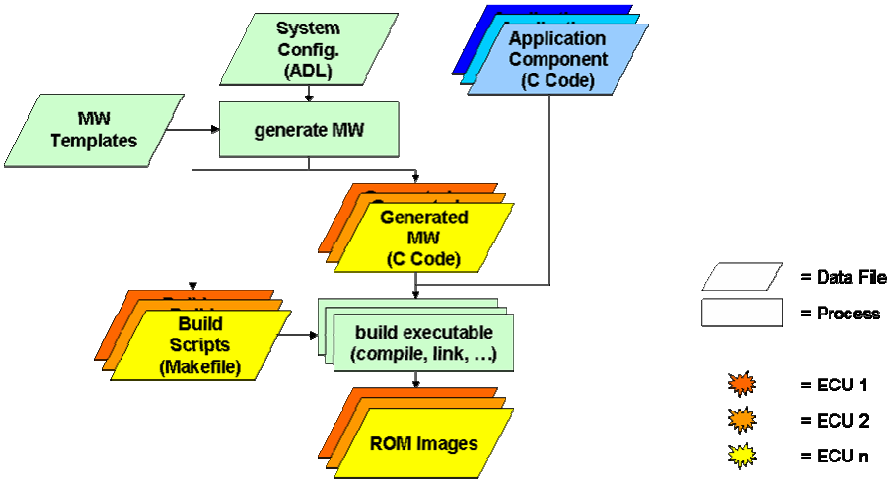

Abbildung 11: Middleware Konfigurations- und Build-Prozess

In der ersten Stufe wird der C-Code der Middleware aus der System-Konfiguration generiert. Dies ergibt i. a. für jedes Steuergerät

separaten Code. In der zweiten Stufe wird dann das ausführbare Programm für jedes Steuergerät generiert.

Die Integration der Implementierungen erfolgte direkt im Powertrain-Validator, der im Arbeitspaket WT4.2 bei ZF aufgebaut wurde. Dazu waren insbesondere einige Anpassungen an die dort herrschenden Randbedingungen erforderlich. Weiterhin definierte ZF die Code-Templates, die als Vorlage zur Code-Generierung für die verschiedenen Powertrain-Middleware-Services dienten. Es wurden nur die für die Validation im Powertrain notwendigen Dienste der Middleware implementiert. Berücksichtigt wurden hierbei die mit den Powertrain-Partnern festgelegten Zuständigkeiten und die für die transparente Kommunikation bereits im verwendeten Tool vorhandenen, einsetzbaren Komponenten.

ZF wirkte an der abschließenden Erstellung eines User Guide für die Verwendung der Prototypen-Implementierung der Powertrain-Middleware aktiv mit. Der User Guide fasst Erfahrungen und Erkenntnisse zusammen, die im Zuge der Entwicklung der Middleware Services gesammelt wurden. Der User Guide wurde als Deliverable [D2.6-Powertrain] erstellt.

WT2.4 Common Part Specification

Die Ziele für dieses Arbeitspaket wurden mit der Struktur-Änderung in WP2 dahingehend festgelegt, dass man die Domänen-Ergebnisse einer Analyse hinsichtlich deren Domänen übergreifenden Anwendung von Konzepten zuführt, die Ergebnisse des Arbeitspakets dokumentiert, sowie die Middleware-Spezifikationen entsprechend harmonisiert, ausführliche Angaben finden sich in [A].

Hier haben wir uns als Powertrain-Verantwortliche ebenfalls aktiv beteiligt. Es konnte eine Liste mit Domänen übergreifenden Themen erarbeitet und deren Inhalte unter verschiedenen Gesichtspunkten bezüglich ihrer Vorzüge und Innovationsfähigkeit kategorisiert werden. Diese Liste kann als Grundlage für eine fahrzeugweite Standardisierung eingeordnet werden. ZF hat hier an konkreten Beispielen wie Mode Management, Fehlermanagement und Diagnose mitgearbeitet.

Die Ergebnisse wurden im Deliverable [D2.4] dokumentiert und Deliverable [D2.7-Powertrain] überarbeitet.

WP3: Development and Validation Tools

Als ein wichtiges Ziel in diesem Arbeitspaket wurde von allen Partnern erachtet, die für den gesamten Entwicklungsprozess erforderliche Tool-Umgebung zu spezifizieren. Da wir keine Tools entwickeln, andererseits aber zahlreiche Erfahrungen mit der Anwendung von Entwicklungstools sammeln konnten, war das Ziel hier, die Domänenanforderungen aus dem Bereich der Software-Entwicklung in die Tool-Entwicklung einzubringen und die spezifischen Anforderungen für die Tool-Entwicklung kontrollierend mit zu verfolgen.

Dazu war zunächst eine Beteiligung in allen Arbeitspaketen von WP3 vorgesehen, die sich aus 2 Gründen dann aber auf die Mitwirkung in WT3.1 beschränkte:

1. Das Arbeitspaket WT3.1 selbst hat einen größeren Umfang, als ursprünglich veranschlagt, angenommen.

2. Wegen der Festlegung auf Domänen-spezifische Architekturen wurde eine Verschiebung von Ressourcen nach WP2 notwendig, und

WT3.1 Architecture, Modelling Language and Exchange Format

Wesentliches Ziel für dieses Arbeitspaket war eine gemeinsame Spezifikation für eine Modellierungssprache für typische Anwendungen zu definieren. Aufgrund der enormen Beteiligung wurde erst durch eine Teilung in Kernteams für Sprache und Artefakte eine effiziente Arbeit möglich. Ausführliche Angaben finden sich in [A].

Ein weiteres Ziel war, die Tool-Entwicklung zu begleiten, um beurteilen zu können, ob eine Alternative zu dem Einsatz von kommerziellen Tools entstehen würde. Das war letztlich nicht so, da das GME-Tool erst gegen Ende des Projekts verfügbar war. Davor hätte es nur zur grafischen Spezifikation eingesetzt werden können.

Zu Testzwecken wurde eine in Simulink® erstellte Interface-Spezifikation des WT4.2 Validator in WP3 eingebracht. Weiterhin wurden die Anforderungen an Code-Generierung und Varianten-Management aus Sicht der Domäne Powertrain definiert und zur Verfügung gestellt.

Die Ergebnisse wurden im Deliverable [D3.6] dokumentiert.

WP4: Domain Specific Implementation and Validation

Die gemeinsame Zielsetzung aller Projektpartner sah in diesem Arbeitspaket die prototypische Implementierung, den Test und die Verifikation der Middleware und der Kommunikation anhand von ausgewählten Szenarien in den einzelnen Domänen vor.

Als Ergebnis einer vorbereitenden Abstimmung unter den deutschen Partnern sollte zunächst anhand von ausgewählten, getriebe-spezifischen Applikationen die Anforderungen in einer „Software in the Loop“ und einer „Hardware in the Loop“ - Umgebung validiert werden.

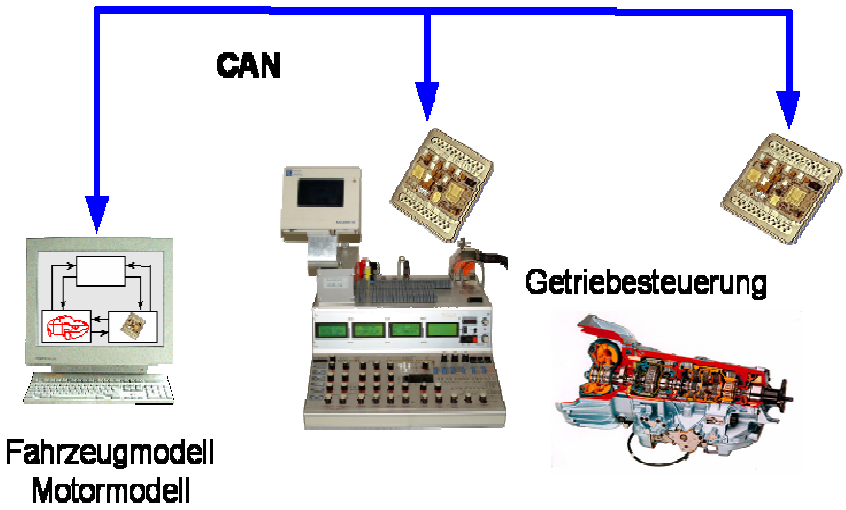

Abbildung 12: Validator für spezifische Validierung

Zur Implementierung sollten Methoden und Tools aus WP3 dienen. Die anderen deutschen Partner (Siemens VDO – Integrierter Starter-Generator, Opel Powertrain – Motor) setzten, jeweils bezogen auf ihre Produkte, vergleichbare Schwerpunkte.

Von den Projekt-Verantwortlichen des WP4 wurde im Rahmen des internationalen WP4-Kick-Off-Meetings von allen Beteiligten gefordert, in allen Arbeitspaketen von den jeweils beteiligten Partnern ein gemeinsames Konzept für die Validierung vorzusehen.

WT4.2 Powertrain

Im Rahmen des WT4.2-internen, internationalen Kick-Off-Meetings bei ZF legte man im Einvernehmen mit allen Partnern (Opel, Siemens VDO, PSA) ein gemeinsames Konzept für den Validator in diesem Arbeitspaket fest. Damit ergab sich im internationalen Projekt durch die Harmonisierung ein etwas anderer Rahmen im Vergleich zu der ursprünglichen nationalen Planung.

Aus unserer Sicht sollten in diesem Arbeitspaket, basierend auf der aus WP1 und WP2 resultierenden Architektur, für den Powertrain relevante Applikations-Software-Komponenten und deren Schnittstellen identifiziert und dann bezüglich ihrer Verteilbarkeit im Powertrain-Netzwerk analysiert und klassifiziert werden. Darauf aufbauend wollte man eine prototypische Umsetzung realisieren und den Test der Architektur, wie oben beschrieben, durchführen.

Mit dem Gesamt-Kick-Off-Meeting im Arbeitspaket WP4 begann ca. ein Jahr nach Projektstart die abgestimmte Arbeit zur Realisierung der Validatoren in den verschiedenen Domänen.

Im Rahmen des internen Kick-Off-Meetings hatte man sich auch schnell auf ein Konzept mit klar definierten Partner-Beiträgen geeinigt, das im folgenden Bild dargestellt ist.

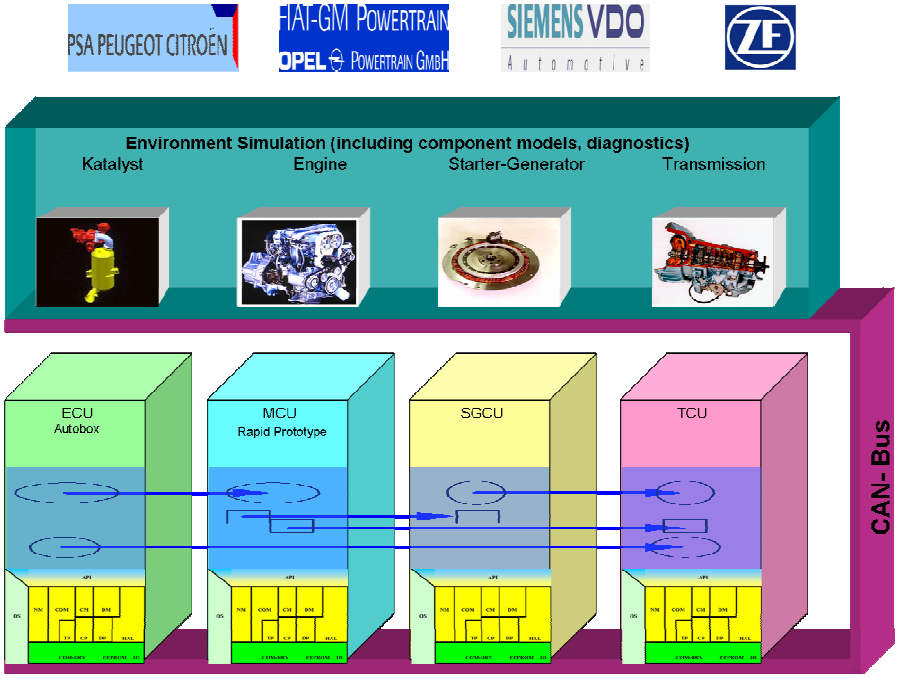

Abbildung 13: Abgestimmter Aufbau des Powertrain Validator

Die vier Projektpartner Opel Powertrain, PSA, Siemens VDO und ZF beschlossen, die folgenden Powertrain-Komponenten im Rahmen des Projekts einzubringen:

- Benzin-Verbrennungsmotor (Opel Powertrain)
- Katalysator / Wegfahrsperre / Wärme-Management (PSA)
- Integrierter Starter Generator (Siemens VDO)
- Automatik-Getriebe (ZF)
- Simulations- und Integrationsumgebung (ZF)

Das gemeinsam definierte Szenario sah vor, durch Verschiebung und Verteilung von typischen Applikations-Software-Komponenten, die in einer Powertrain-Architektur (siehe WP2) realisiert werden sollten, das Steuergerät für die PSA-Funktionen einzusparen.

Mit der Aufnahme der Entwicklungsarbeit wurde ein Entwicklungsprozess festgelegt und gemeinsam ein Projektplan erstellt. Die Realisierung implizierte einen verteilten, Unternehmensgrenzen überschreitenden Entwicklungsprozess mit besonderem Fokus auf Schnittstellen. Die Entwicklung wurde in 2 Phasen eingeteilt.

Die erste Phase zielte ab auf eine Simulationsumgebung, die bei jedem der Partner auf dem PC lauffähig sein sollte. Man einigte sich darauf, Matlab/Simulink® und Stateflow® dafür einzusetzen. Die Phase selbst wurde weiter unterteilt in mehrere Entwicklungsschritte.

In der zweiten Phase sollte der Validator, der auch seriennahe Steuergeräte beinhalten und über CAN mit einem Simulationsrechner vernetzt sein sollte, entwickelt werden. Das komplett integrierte Powertrain-Steuerungssystem sollte aus Kosten- und Aufwandsgründen nur bei ZF vorhanden sein.

Die Beiträge der Partner wurden folgendermaßen festgelegt:

- Opel Powertrain implementiert ein Motormodell sowie eine verteilbare Momentenmanagement-Funktion, eine verteilbare Fahrstrategie und die Motorsteuerung.

- PSA implementiert das Fahrermodell sowie verschiebbare Funktionen für Wegfahrsperre, Katalysator und für eine verteilbare Wärmemanagement-Funktion

- Siemens VDO implementiert Modelle des integrierten Starter-Generator (ISG) und der Batterie, sowie eine verschiebbare Batteriemanagement-Funktion, eine verteilbare Momenten-management-Funktion und die ISG-Steuerung.

- ZF implementiert Modelle für Getriebe, Fahrzeug und für die notwendige Umgebung, sowie eine verteilbare Schaltstrategie und die Getriebesteuerung. Zusätzlich stellt ZF die Integrations-umgebung für die Realisierungen in Simulation und im Validator und führt alle dafür erforderlichen Aufgaben aus.

Entsprechend den definierten Zuständigkeiten der Beteiligten (Opel, Siemens VDO, PSA und ZF) wurde die Arbeit an einem virtuellen Prototypen mit Hilfe der Entwicklungs-Tools Matlab/Simulink® und Stateflow® gestartet.

Erstes Zwischenergebnis war eine Simulation des Zusammenwirkens der Powertrain-Basiskomponenten, bestehend aus Motormodell, Starter-Generator-Modell und Getriebemodell, stimuliert mit einem Umgebungsmodell für Fahrer, Fahrzeug und dessen notwendiger Umgebung. Das Modell hatte noch folgende Einschränkungen:

- Die Struktur hätte eine direkte Übertragung in die Hardware-in-the-Loop-Umgebung nicht zugelassen, da die entsprechenden Komponenten nicht isoliert waren.
- Die Signalnamen waren zwischen den Komponenten nicht abgestimmt, dementsprechend ergaben sich in Simulink® Verbindungen zwischen Ports mit unterschiedlichen Namen. Z. B. wurde die Motordrehzahl von Port A (n_mot) nach Port B (eng_speed) geliefert.
- Weitere Signal- und Software-Komponenten-Eigenschaften waren noch nicht abgestimmt.

Daraus ergab sich die Notwendigkeit, eine weitergehende Strukturierung mit Interface-Standardisierung vorzunehmen.

In etwa zeitgleich mit dem Abschluss der ersten Simulationsphase war die Umstrukturierung in WP2 erforderlich geworden (siehe WP2-Beschreibung). Diese hatte insofern Auswirkungen, als beteiligte Partner aufgrund des in WP2 entstehenden Mehraufwands durch die Spaltung in Domänen, ihre Ressourcen neu zuordneten. Das betraf unser Arbeitspaket besonders, da PSA als Hauptverantwortlicher für Teilprojekt WP2 seine Ressourcen nun dort konzentrierte. Dadurch entstand die Notwendigkeit, das Arbeitspaket neu aufzustellen.

Die Lösung sah so aus, dass man die bereits erarbeiteten Strukturen beibehielt und ZF die noch ausstehenden PSA-Arbeiten für Simulation und Validator in reduziertem Umfang mit übernahm. Ziel war dabei, die Szenarien hinsichtlich Verschiebung, mit dem vorgesehenen Effekt eine Elektronik einzusparen, umsetzen zu können. Verzichtet wurde auf die Katalysator-Funktion und die Wärmemanagement-Funktion.

Eine Aufteilung in verschiebbare (Funktionalitäten wie Schaltstrategie, Fast-Start, Fast-Stopp, Batteriemanagement oder Wegfahrsperre) und nicht verschiebbare (Aggregat-Steuerungsfunktionen wie Motor-, ISG- und Getriebesteuerung) war zu diesem Zeitpunkt bereits einvernehmlich diskutiert worden. Hintergrund ist die enge Bindung der Aggregat-Steuerfunktionen zur sehr aufwändigen Sensorik / Aktuatorik des jeweiligen Aggregats. Damit ergab sich die im folgenden Bild dargestellte, entsprechend modifizierte Zuordnung von Komponenten-Verantwortlichkeiten für die Umsetzung von Simulation und Validator.

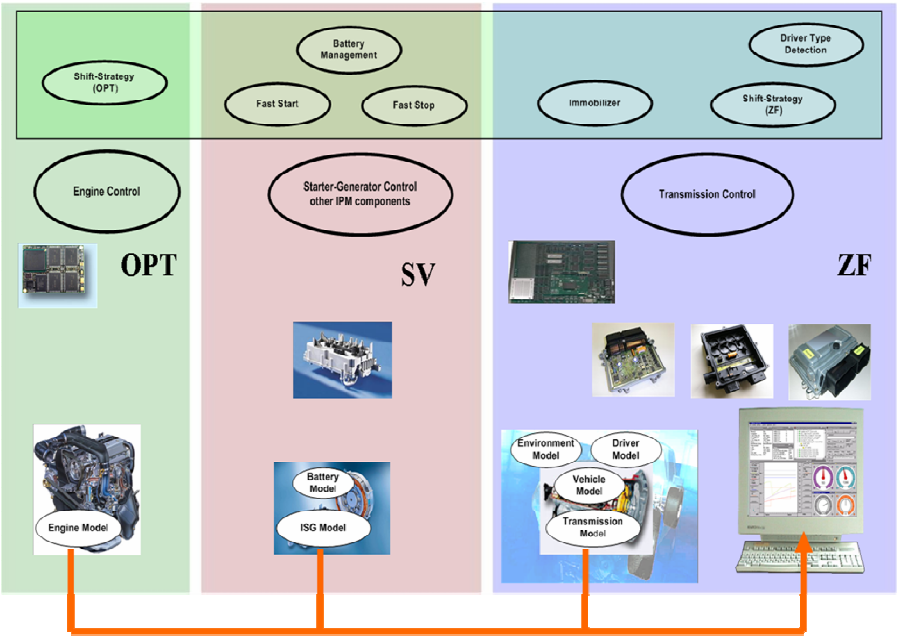

Abbildung 14: Abgestimmte Komponenten-Verantwortlichkeiten

Oben im gerahmten Teil sind die verschiebbaren Komponenten Partner-spezifisch gruppiert, darunter die nicht verschiebbaren Aggregat-Steuerungsfunktionen, sowie die zu integrierenden Steuergeräte. Die Aggregate selbst und deren Umgebungen waren durch Simulations-Komponenten realisiert und mussten auf dem Modellrechner (rechts unten im Bild) integriert werden (gelber Pfeil). Die seriennahen Integrations-Steuergeräte wurden von Siemens VDO (ISG Steuergerät) und von ZF bereitgestellt.

Die Mapping-Szenarien, die die Zuordnung von Target-Softwarekomponenten zu Steuergeräten beschreiben, wurden ebenfalls entsprechend angepasst.

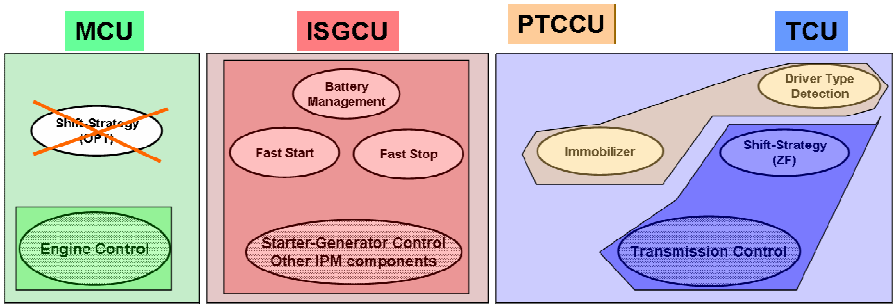

Abbildung 15: Szenario 1 als Basisszenario

Die Target-Softwarekomponenten wurden im Szenario 1 entsprechend den Partnerbeiträgen den Steuergeräten zugeordnet. Dieses Szenario sollte als Referenz-Szenario für die folgenden Szenarien dienen.

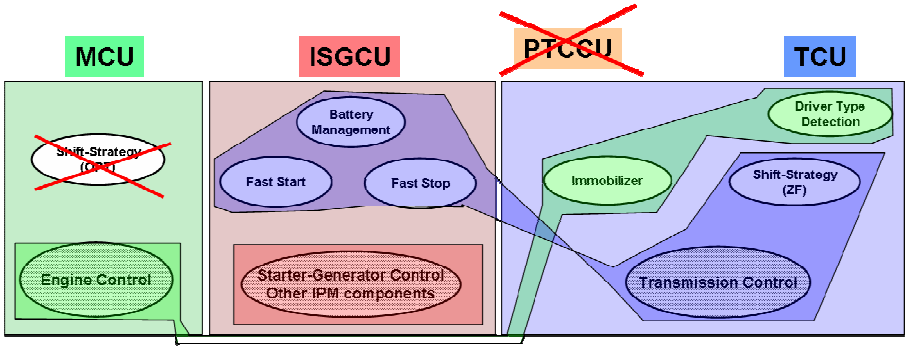

Abbildung 16: Szenario 2 als Verschiebeszenario

Dabei stellte ZF ein Steuergerät, das nur Management-Funktionen enthält und durch die Verschiebung von Komponenten entsprechend Szenario 2 entfiel, die Idee hierfür wurde ursprünglich mit PSA gemeinsam entwickelt.

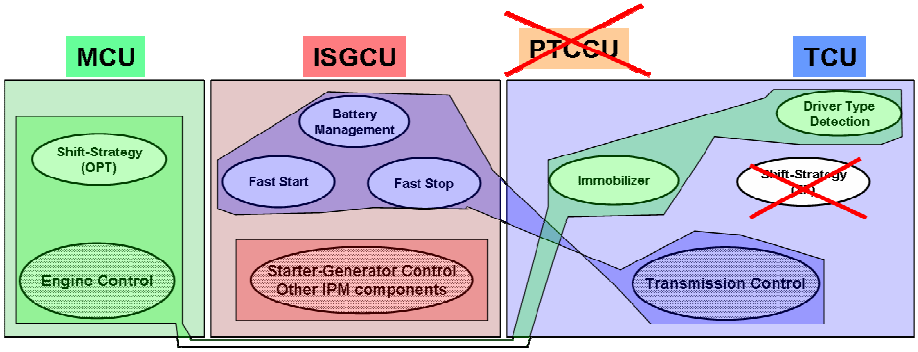

Abbildung 17: Szenario 3 als Austauschszenario

In Szenario 3 wurde die Schaltstrategie als Komponente ausgetauscht. Die 3 Szenarien wurden vorgeschlagen, um damit die Ergebnisse aus WP2 in Verbindung mit der in WP4 vorgenommenen Funktions-Zerlegung zu verifizieren.

Aus rechtlichen Gründen (IPR) wurde vereinbart, nur Bibliothekskomponenten auszutauschen, also alle Software-Komponenten als „Black Boxes“ zu realisieren.

Während wir in der ersten Phase nur mit Simulations-Software-Komponenten zu tun hatten, konnten die Software-Komponenten für die zweite Phase in verschiebbare oder nichtverschiebbare Target-Software-Komponenten und in Simulations-Software-Komponenten unterteilt werden.

Die Interaktionen aller Software-Komponenten sind in der logischen Architektur, dargestellt im folgenden Bild, zu sehen.

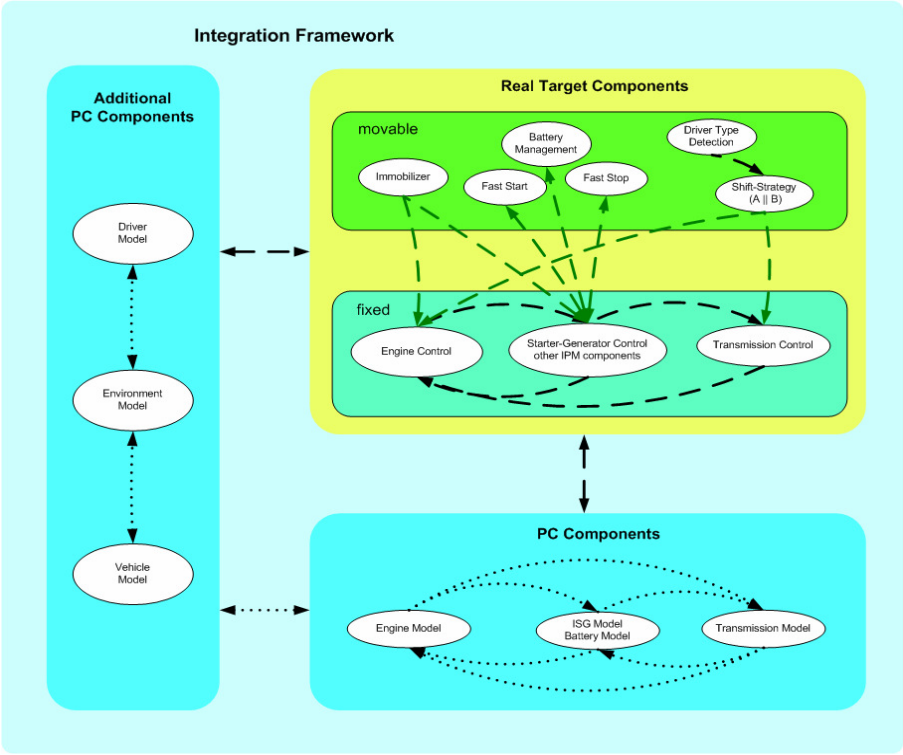

Abbildung 18: Logische Architektur des Powertrain Validator

Für die weitere Verfeinerung der Modellstruktur wurde diese Komponententeilung zugrunde gelegt.

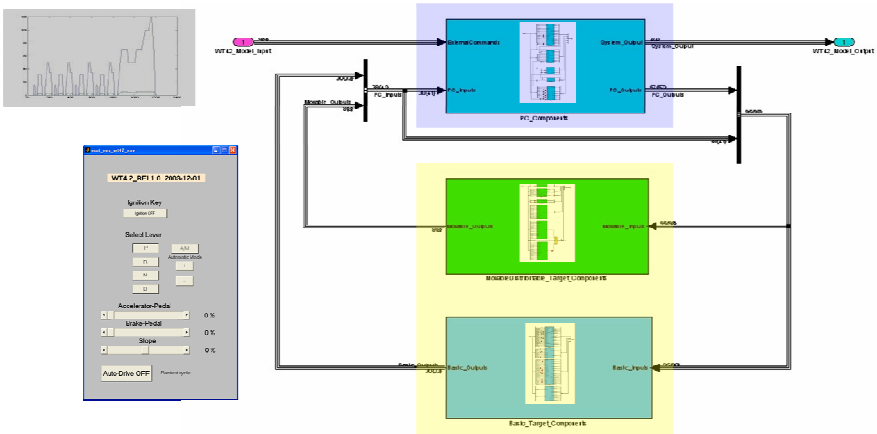

Abbildung 19: Strukturiertes Simulink® Modell

Das Ergebnis der Strukturierung ist in der abgebildeten Simulink®-Struktur dargestellt. Im Modellblock oben sind im Vorwärtszweig die Simulations-Software-Komponenten zusammengefasst. Darunter

befinden sich in der Rückführung die Target-Software-Komponenten, aufgeteilt in verschiebbare und nichtverschiebbare Komponenten.

Parallel zur weiteren Verfeinerung der Funktionsstruktur auf Modellebene in Phase zwei, wurde eine Einigung über Signalnamen erzielt, die ebenfalls bereits im Simulink®-Modell berücksichtigt wurde. Das Gesamtmodell wird von extern über ein Panel bedient oder durch einen ECE Fahrzyklus stimuliert. Es ergibt sich keine nennenswerte Abweichung zwischen Soll-Fahrgeschwindigkeit und Ist-Fahrgeschwindigkeit in der Gesamtmodell-Simulation.

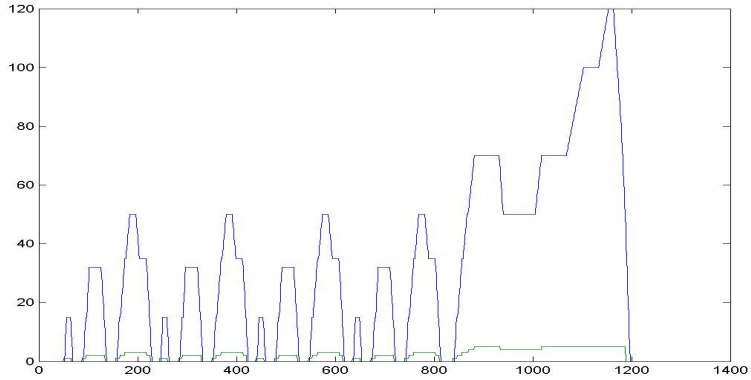

Abbildung 20: ECE Fahrzyklus als Referenz-Stimulation

Mit der Entwicklung der Simulation wurden die Anforderungen aus Sicht der Powertrain Validierung gesammelt und in das WP4-Anforderungsdokument Deliverable [D4.1] eingebracht, in dem die Anforderungen von allen WP4-Domänen zusammengefasst wurden.

Als entschieden werden musste, welches Tool zur Integration verwendet wird, gab es zwar bereits aus WP3 das GME2000-Tool, doch dieses eignete sich auf dem verfügbaren Stand nur für eine Dokumentation, nicht aber für eine echte System-Integration. Aus diesem Grund wurde die Entscheidung getroffen, die DaVinci Tool Suite der Fa. Vector GmbH als Integrations-Tool einzusetzen. Ermöglicht wurde diese Entscheidung durch die folgenden Randbedingungen:

- Das erste Release sollte im passenden zeitlichen Rahmen fertig gestellt werden.
- Das Tool bot so weit offene Schnittstellen, dass die Integration der für die Validierung erforderlichen Powertrain-Middleware Services als machbar erachtet wurde.
- Das Tool sollte auch Unterstützung für die seitens ZF eingesetzten Motorola PPC Target-Steuergeräte enthalten.
- Es bot eine Prozess-Schnittstelle zu außerhalb von DaVinci entwickelter Anwendungs-Software. Das war zum einen für die Entwicklungspartner und zum anderen für die Realisierung der echtzeitfähigen Simulation relevant.

Mit dieser Entscheidung waren dann auch die Randbedingungen für die Middleware-Implementierung geklärt. Es konnte hier nun ebenfalls eine konkrete Planung der Implementierung stattfinden.

Im weiteren Verlauf wurden die Simulationskomponenten auf eine echtzeitfähige PC-Plattform (wie unten abgebildet) übertragen.

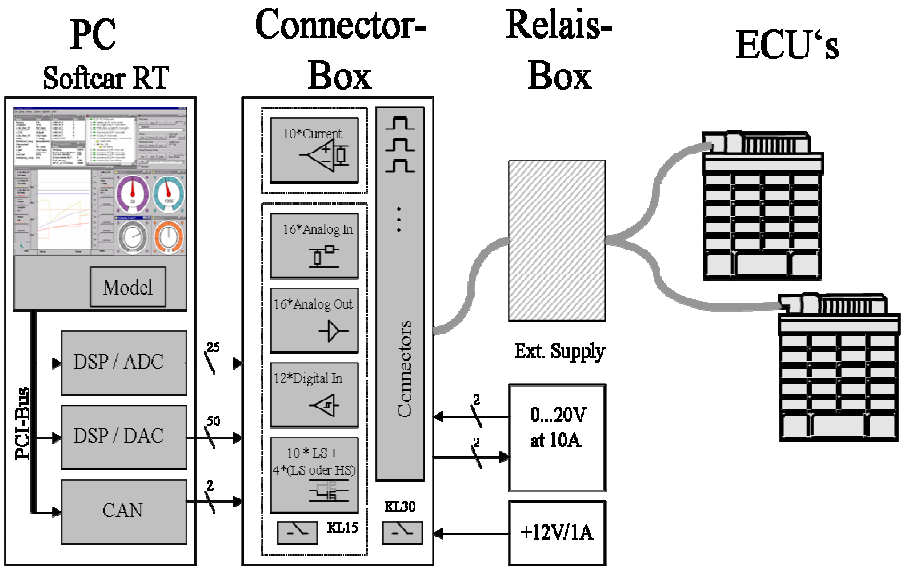

Abbildung 21: Modellrechner

Dieser Übergang betraf Fahrer-, Fahrzeug- und Umgebungsmodell, sowie Motor-, ISG-, Batterie-, und Getriebemodell und entspricht somit dem Übergang von „Model in the Loop“ zu „Software in the Loop“, wo im konkreten Fall das Modell durch Code und ein für den PC übersetztes Executable ersetzt wird (siehe Bild unten).

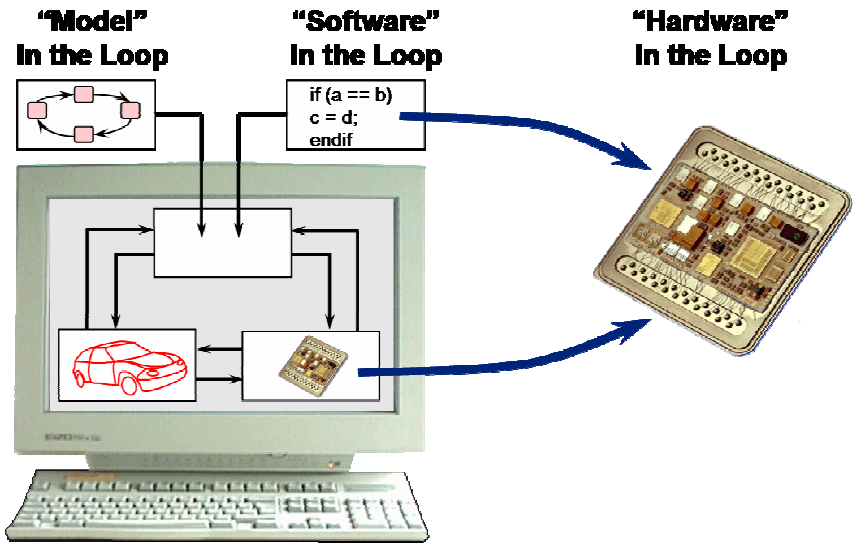

Abbildung 22: Testumgebungen

Um den Übergang mit den Partnern realisieren zu können, wurde seitens ZF ein auf Bibliotheken basierendes Integrationskonzept erarbeitet und mit den Partnern abgestimmt. Die echtzeitfähigen Modelle wurden durch Abgleich mit der Simulink®-Simulation verifiziert. Dabei sind Abweichungen aufgetreten, die von allen Beteiligten Nachbesserungen an den Realisierungen erforderlich machten.

In Abstimmung mit den Partnern wurde der Validator-Aufbau definiert wie im folgenden Bild dargestellt.

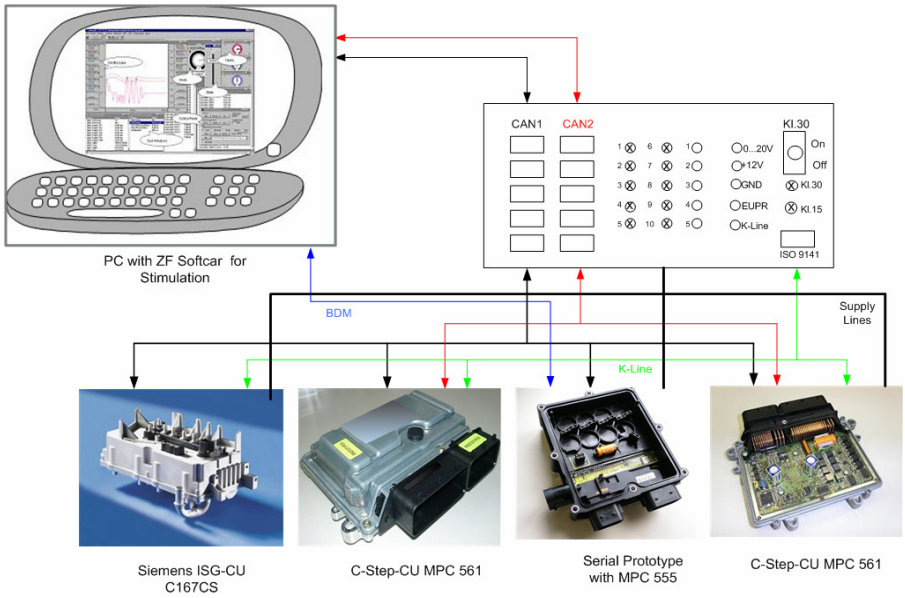

Abbildung 23: Validator-Konzept mit Steuergeräten

Der komplette Aufbau beinhaltet einen Simulationsrechner, vier Steuergeräte (drei Steuergeräte von ZF, eines von Siemens VDO, sowie die Anbindung eines Prototypensteuergeräts von Opel Powertrain, dessen Inhalt im Validator auf einem der ZF-Steuergeräte implementiert werden sollte. Dafür mussten für alle Steuergeräte Verbindungen für Kommunikation und Versorgung untereinander und zusätzlich mit dem Modellrechner und für die Steuergeräte für deren Programmierung vorgesehen werden.

Die folgenden Szenarien wurden abschließend vereinbart.

- Das Basis-Szenario wird realisiert durch eine geeignet strukturierte Anwendungsfunktion für den Powertrain, stimuliert durch den ECE Fahrzyklus. Es soll als Referenz für die nachfolgenden Szenarien dienen.
- Das Verschiebe-Szenario realisiert im Vergleich zum Basis-Szenario eine unterschiedliche Verteilung der Anwendungs-Software-Komponenten auf das Target-Netzwerk durch Re-Konfiguration. Durch das neue Mapping soll ein Steuergerät eingespart werden. Durch einen Referenztest soll die Funktion nachgewiesen werden. Das Szenario dient zusätzlich der Überprüfung von Mode Management und Parameter-Kalibrierung.
- Beim Austausch-Szenario soll, ausgehend vom Verschiebe-Szenario die Schaltstrategie durch Re-Konfiguration ausgetauscht werden. Durch einen Referenztest soll die Funktion nachgewiesen werden.

Die Planungen wurden in den Deliverables [D4.2-Architecture-Powertrain], [D4.2-Functions-Powertrain] und [D4.3-Interfaces-Powertrain] dokumentiert.

Im Verlaufe des Projekts stellte sich heraus, dass für den C167-Prozessor, den das ISG-Steuergerät einsetzte, keine Target-Software-Komponenten im Rahmen des Projekts verfügbar sein würden. Somit konnte dieses Steuergerät nicht direkt in die Integration mit DaVinci einbezogen werden, sondern musste als Extern-Entwicklung, wie im Übrigen auch der PC, behandelt werden. Beim PC lag der Grund darin, dass nicht das von DaVinci unterstützte CANoe-Tool wegen der hohen Echtzeitforderungen an die Modelle für Motor, Getriebe und ISG eingesetzt wurde.

Ein Import des Matlab®-Modells in den Software-Struktur-Editor der DaVinci Tool Suite lieferte die Schnittstellen aller Software-Komponenten, die Signal- und Komponentenspezifikationen mussten entsprechend der abgestimmten Spezifikation hinzugefügt werden.

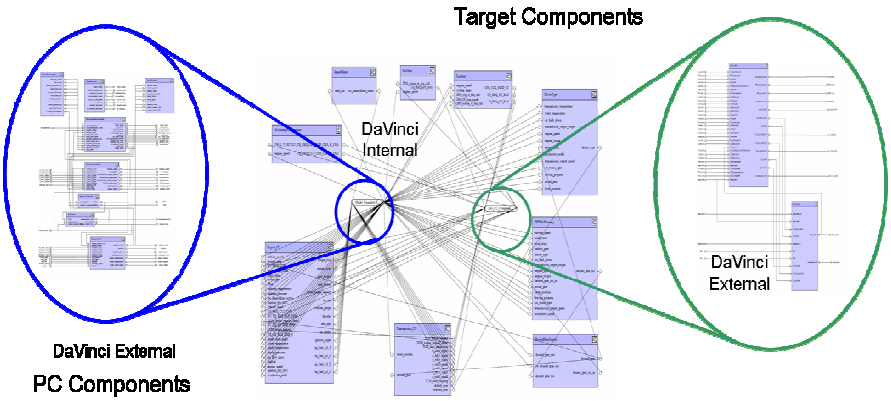

Abbildung 24: Software-Struktur in DaVinci

Dabei wurden externe Komponenten (Simulations-Software-Komponenten und Target-Software-Komponenten des ISG-Steuergeräts) zu Einzelkomponenten, soweit dies sinnvoll war, zusammengefasst. Insbesondere mussten die verschiebbaren Target-Software-Komponenten des ISG natürlich auch als Einzelkomponenten dargestellt werden.

Der Integrationsprozess ist beispielhaft für das ISG-Steuergerät dargestellt, Datenbasis ist eine generierte CAN-Datenbasis.

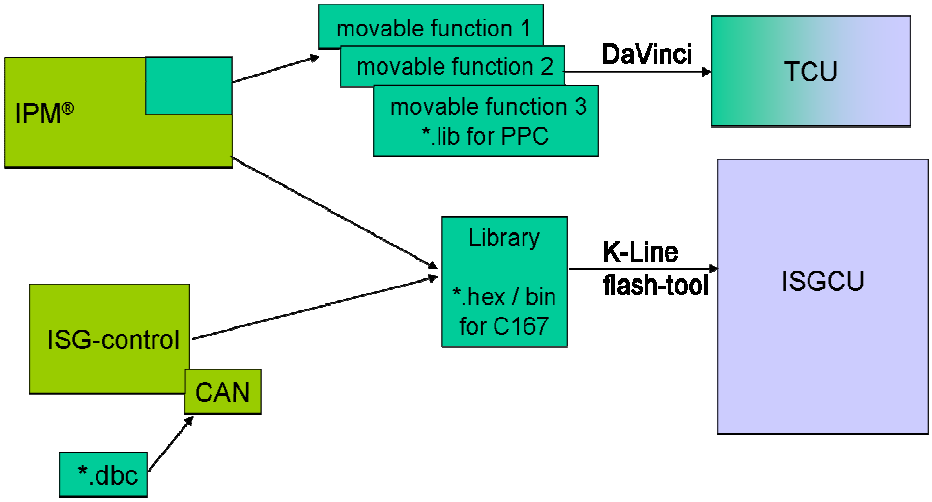

Abbildung 25: Konzept zur Integration des ISG-Steuergeräts

Für die Realisierung von Middleware- und Kommunikations-Layer wurde auf in DaVinci realisierte Target-Basis-Software-Komponenten zurückgegriffen, so weit dies für die Szenarien und die Middleware-Services möglich war. Insbesondere wurde als Kommunikations-Layer die DBKOM/DBKOMGenV2.48 verwendet. Eine Analyse der Features und Schnittstellen hatte im Rahmen von WT2.3 zu dem Ergebnis geführt, dass dieser Schritt sinnvoll zu machen ist. Den Prozess für die Integration der zusätzlichen Middleware-Teile zeigt das folgende Bild exemplarisch für die Services zur Kalibrierung:

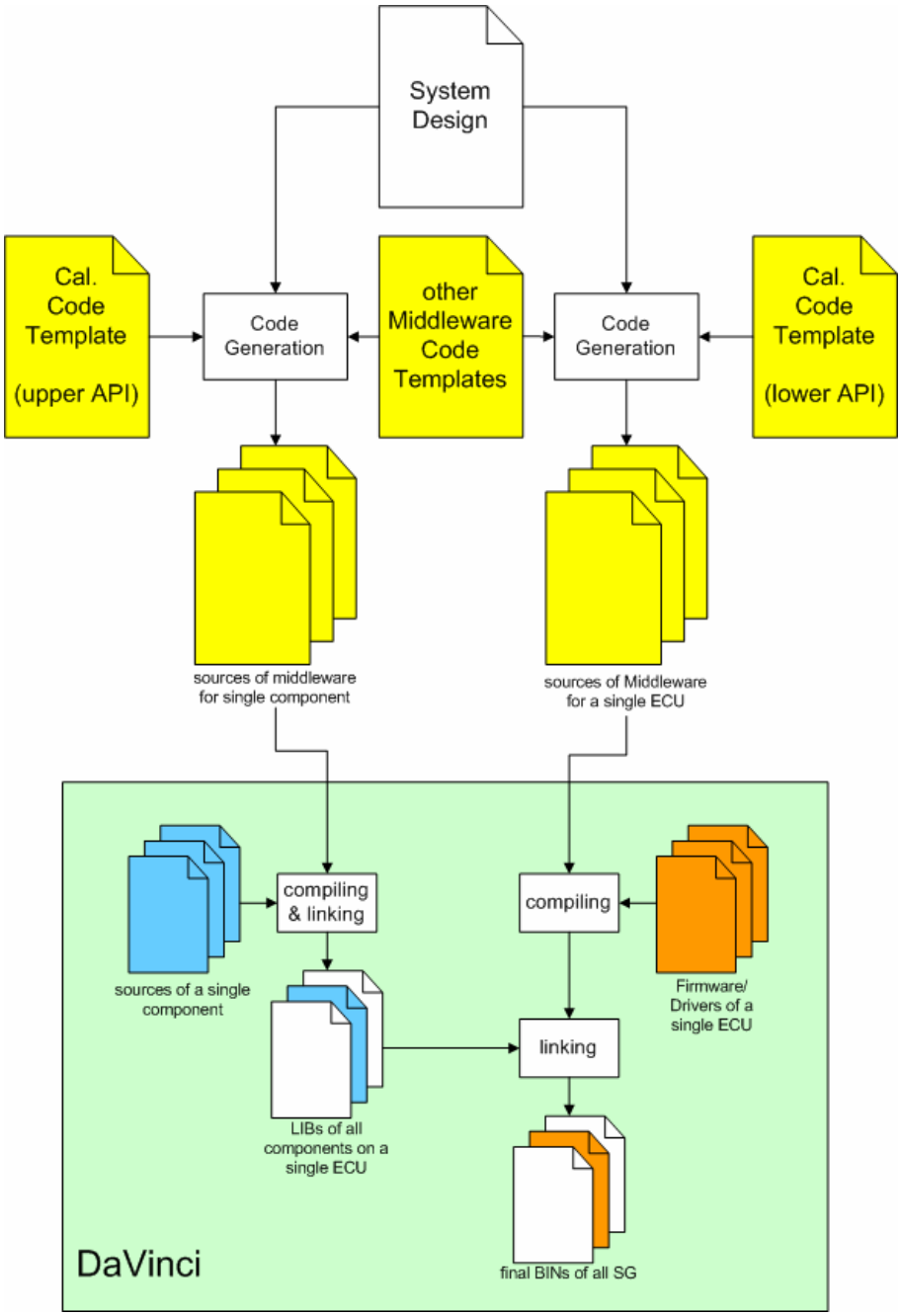

Abbildung 26: Middleware-Integration am Beispiel Kalibrierung

Im weiteren Verlauf wurde an Spezifikationen gearbeitet, die für die verschiedenen Integrationsschritte, auch für den der Midleware-Integration, erforderlich waren.

Die benötigten Middleware-Komponenten wurden entsprechend der geplanten Szenarien für die Umsetzung im Powertrain Validator entwickelt. Das folgende Bild veranschaulicht exemplarisch das Mode Management Szenario, bei dem eine Master Mode Manager Komponente auf dem Getriebesteuergerät (TCU) vorgesehen wurde, welche die Startup-Phase der Controller synchronisieren sollte.

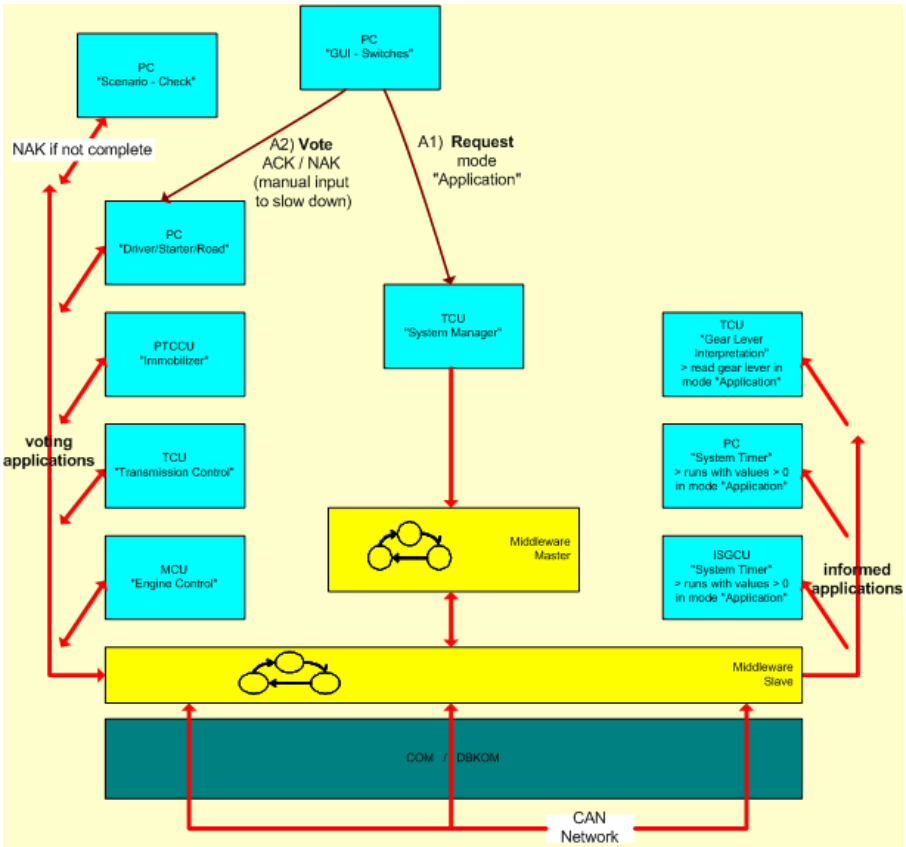

Abbildung 27: Implementierung am Beispiel Mode Management

Entsprechend wurden auch die anderen Middleware-Bestandteile anhand der Szenarien definiert und implementiert. Dies bezieht sich auf die transparente Kommunikation und auf die Parameter-Kalibrierung entsprechend [D2.3-Powertrain].

Nicht implementiert und entsprechend auch nicht validiert wurden die Konzepte zur Diagnose und zum transparenten Zugriff auf persistente Speicher. Diese Teile konnten aufgrund der Umstrukturierung in WP2 nicht im ursprünglich geplanten zeitlichen Rahmen mit der noch verfügbaren Kapazität bearbeitet werden, somit war die Implementierung der spezifizierten Konzepte nicht mehr möglich.

Um zu einsatzfähigen Steuergeräten zu kommen, war es notwendig, alle drei Steuergeräte für die Generierung und Integration einzurichten. Das Ergebnis der Integration ist im folgenden Bild dargestellt.

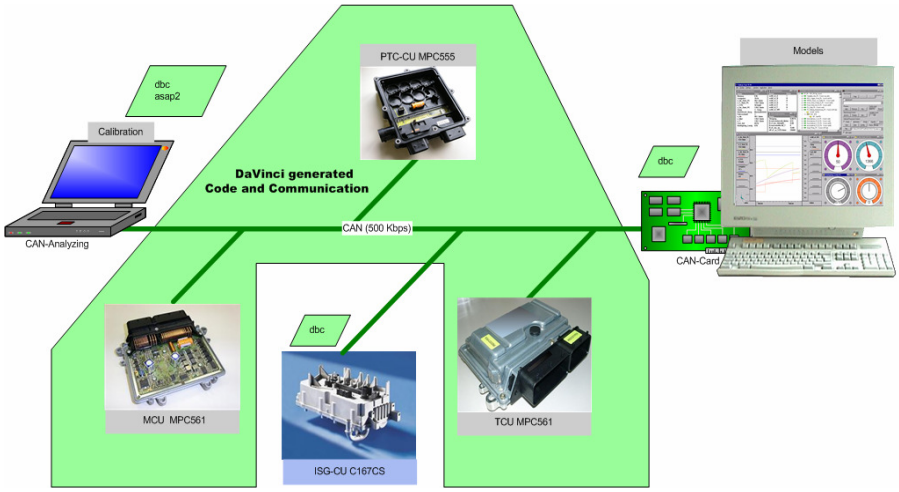

Abbildung 28: Integrierte Powertrain-Steuergeräte

Aufgrund der Anforderungen an die Middleware war teilweise die Anpassung der in DaVinci enthaltenen Basis-Software erforderlich, bevor diese für den Einsatz im jeweiligen Steuergerät konfiguriert und um fehlende Teile ergänzt werden konnte (Basis-Software-Komponenten für Download, Speicherorganisation, Controller-Konfiguration, usw.).

Für die Integration der Steuergeräte wurde seitens ZF ein auf Bibliotheken basierendes Integrationskonzept definiert und mit den Partnern abgestimmt. Die Ergebnisse wurden in den Deliverables [D4.3-Powertrain] und D4.4-Powertrain] dokumentiert.

Die gelieferten Komponenten wurden in DaVinci integriert und durch Abgleich mit den in Simulink® aufgezeichneten Simulationen verifiziert.

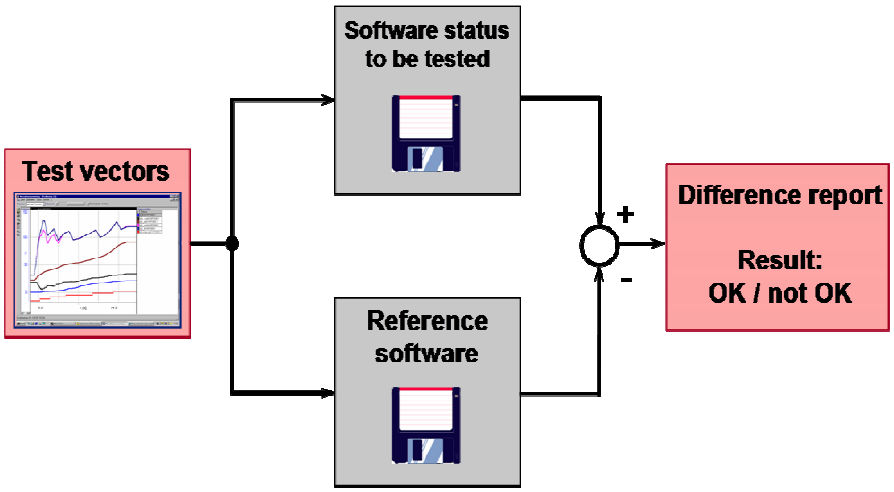

Abbildung 29: Referenztest

Abweichungen mussten von den jeweils Zuständigen nachgebessert werden, so dass eine Reihe von Iterationen erforderlich war, bis das Gesamtsystem lauffähig fertig gestellt und die verschiedenen Szenarien für die Verschiebung und den Austausch von Anwendungsfunktionen realisiert werden konnten. Die Referenztests erfolgten wie im Bild oben dargestellten.

Für die Validierung waren letztlich Aufzeichnungen von Systemgrößen wie Fahrgeschwindigkeit und deren Übereinstimmung mit den entsprechenden Referenzgrößen maßgeblich. Damit konnte die prinzipielle Tauglichkeit der Konzepte in Verbindung mit den Prämissen (Z. B. Nicht-Verschiebbarkeit von bestimmten Komponenten) nachgewiesen werden, siehe [B].

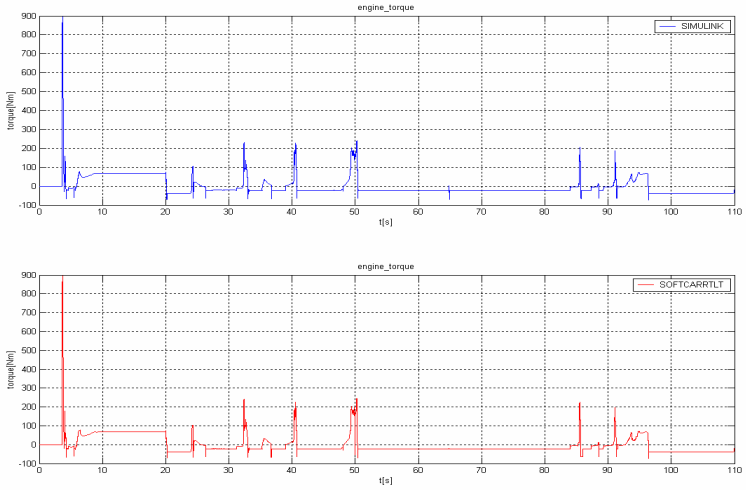

Abbildung 30: Referenz Simulation – Test PC-Modell

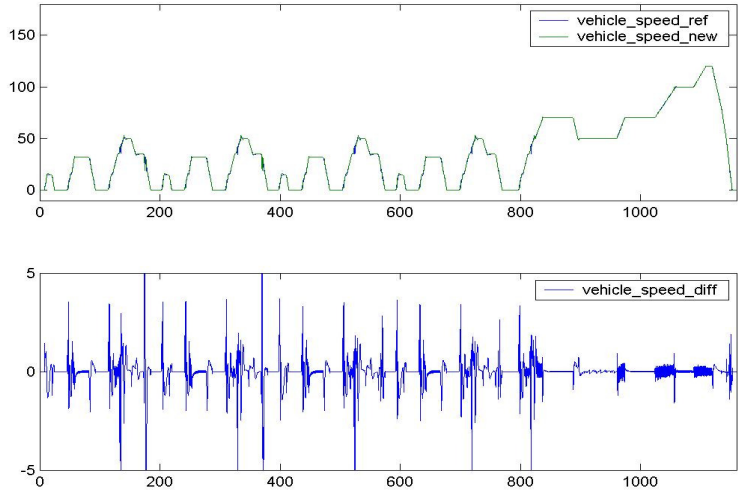

Abbildung 31: Referenz Szenario 1 – Test Szenario 2

Die Ergebnisse wurden in Deliverable [D4.5-Powertrain] dokumentiert.

Im Rahmen der Abschlussveranstaltung in München am 21./22. Juni 2004 konnte der Validator präsentiert werden.

Abbildung 32: Aufbau des Validators zur Abschlußpräsentation

In den folgenden Bildern ist der Steuergeräteverbund, bestehend aus Motor-Steuerung, ISG-Steuerung, Getriebesteuerung und dem Koordinations-Steuergerät, welches im Szenario 2 entfällt, zu sehen.

Abbildung 33: Powertrain Steuergeräte-Verbund mit PC

Auf der Veranstaltung wurden die Ergebnisse aller Teilprojekte von den Work Package Leadern dargestellt. Die Veranstaltung und insbesondere die Präsentation der Validatoren allgemein fand großes Interesse.

WT4.4 Human Machine Interface

In diesem Arbeitspaket sollten die Möglichkeiten untersucht werden, wie durch neue Technologien die Mensch-Maschine-Schnittstelle an die neuen Anforderungen der Fahrzeugvernetzung angepasst werden können. Die strikte Trennung der Bedienfunktionen von Lenkung, Motor, Getriebe, Bremse und Komfortelektronik gehört auf lange Sicht der Vergangenheit an. Die bisher getrennten Komponenten werden in engem Kontakt stehen und somit neue bzw. veränderte Aufgabenstellungen übernehmen.

Im Rahmen der Arbeiten sollte untersucht werden, welche Funktionen sinnvoll in gemeinsamen Interfaces gekoppelt werden können und welche Technologien für Ein- und Ausgabekomponenten dafür geeignet sind. Unser Beitrag sollte die für den Powertrain relevanten Ein- und Ausgabedaten beleuchten und deren Instrumentierung im Zusammenspiel mit dem Gesamtfahrzeug-konzept vornehmen.

Mit den sonst beteiligten Partnern und unter Leitung von CRF konnten verschiedene Szenarien abgestimmt werden, die aus der Sicht einer Powertrain-Anwendung in Zusammenhang mit der Fahrerschnittstelle (Display, Bedieneinheiten) interessant sind. Es handelt sich dabei um spezielle Situationen wie Einpark-, (Stopp & Go)-, oder Winterbetrieb, in denen der Fahrer per Bestätigung bestimmte Features frei schalten und dazu passende Anzeigen erhalten kann. Die Getriebesteuerung kann in diesen Situationen mit hoher Priorität auf die entsprechenden HMI Ressourcen zugreifen.

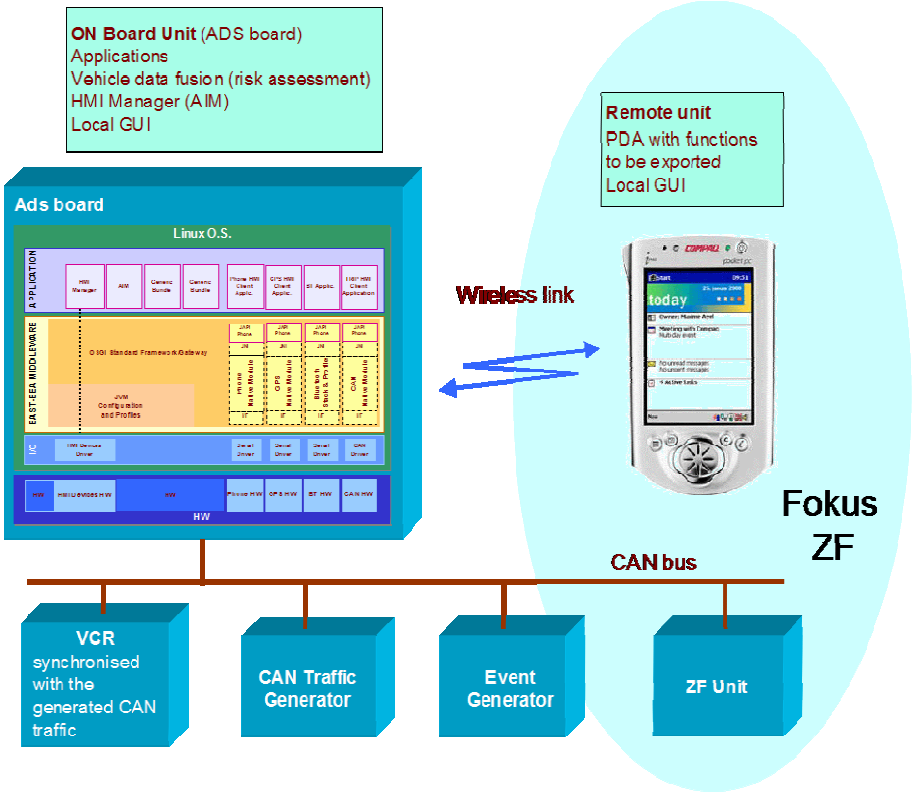

Abbildung 34: Beteiligung am HMI Validator

Die definierten Szenarien wurden in einem über CAN angebundenen Simulationsrechner (ZF Unit) realisiert.

Dazu bedurfte es einer XML basierten Definition und der Programmierung der Szenarien. Der nachfolgende Auszug enthält beispielhaft die HMI-Interface-Spezifikation für die Einpark-Hilfe.

```
XML-Page definition of "Easy Parking"-page:
<Page xmlns:xsi=http://www.w3.org/2001/XMLSchema-Instance
        xsi:noNamespaceSchemaLocation="TAMLPages.xsd">
    <Title>Driver Assistance</Title>
    <Autoaction Timer="5"><!– or timeout set by can –>
                <Action>0</Action>
    </Autoaction>
    <Message>
            <Message>Do you want Easy Parking?</Message>
            <AssociatedAction>
                <Label><Short>YES</Short><Long>YES</Long></Label>
                <Action>1</Action>
            </AssociatedAction>
            <AssociatedAction>
                <Label><Short>NO</Short><Long>NO</Long></Label>
                <Action>0</Action>
            </AssociatedAction>
    </Message>
</Page>
```

Abbildung 35: XML-Beschreibung für die Einparkhilfe

Durch die starke Beschränkung auf die einzelnen Domänen entstanden nur exemplarisch Domänen übergreifende Ergebnisse.

In der Domäne HMI hat CRF in Kooperation mit XFIN Magneti Marelli und C-Lab die entsprechende Spezifikation erstellt und umgesetzt. Anschließend hat ZF die zur Verfügung gestellten Schnittstellen für die Realisierung der definierten Anwendungsfälle verwendet.

Die Beiträge finden sich in den Deliverables der HMI-Domäne [D4.2-HMI], [D4.3-HMI], [D4.5-HMI].

Die implementierten Szenarien konnten als Teil des HMI Validator ebenfalls in München bei der Abschlussveranstaltung demonstriert werden.

2.2 Voraussichtlicher Nutzen

Software-Standardisierung, insbesondere im Bereich der Basis-Software, ist als Grundlage für einen weiterhin wachsenden Software-Anteil aus technischen, wirtschaftlichen und qualitativen Gründen zwingend erforderlich, Deutlich wird dies mit der folgenden Grafik, die ein enormes Wachstum des Wertschöpfungsanteils von Standardsoftware im Kraftfahrzeug im Zeitraum 2002 bis 2015 prognostiziert.

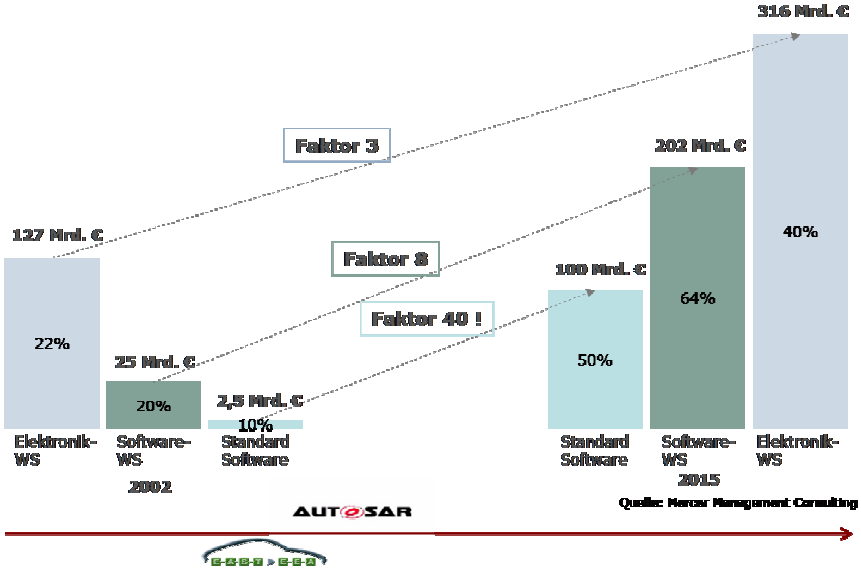

Abbildung 36: Standardisierung als Treiber

Verteilte Systeme müssen als Schlüsseltechnologie für zukünftige Fahrzeug-Generationen betrachtet werden.

Vor einer Serieneinführung müssen Anschlussarbeiten durchgeführt werden. Z. B. müssen die zunehmenden Sicherheitsforderungen einbezogen werden, eine Migration von heutigen Implementierungen auf die Architektur muss vorgesehen werden und die rechtlichen Rahmenbedingungen müssen geschaffen werden.

2.3 Fortschritt auf diesem Gebiet bei anderen Stellen

Fortschritte auf diesem Gebiet in Europa waren wegen der großen Beteiligung i. W. durch das Projekt bestimmt.

In 2003 hat sich die weltweite Industrie-Initiative AUTOSAR mit dem Ziel konstituiert, eine EAST-EEA entsprechende, fahrzeugweite Architektur gemeinsam zur Serienreife zu führen. Das sich bis Ende 2004 alle „Global Player“ der Automobilindustrie weltweit dieser Initiative angeschlossen und vertraglich verpflichtet haben, beweist die führende Rolle Europas auf diesem Gebiet, angestoßen auch durch das EAST-EEA Projekt.

2.3 Erfolgte und geplante Veröffentlichungen

[A] *T. Thurner, J. Eisenmann, U. Freund, R. Geiger, M. Haneberg, U. Virnich, S. Voget*. Das Projekt EAST-EEA – eine middlewarebasierte Softwarearchitektur für vernetzte Steuergeräte. VDI-Tagung Elektronik im Kraftfahrzeug, Baden-Baden, 2003.

[B] R. Geiger, S. Lämmermann. Übergang von Simulation zur Erprobung: Erfahrungen, Chancen, Risiken. VDI-Tagung Erprobung und Simulation in der Fahrzeugentwicklung, Würzburg, geplant für Juni 2005.

Literaturverzeichnis

[1] CAN-Bus Spezifikation,
URL: http://www.can.bosch.com/docu/can2spec.pdf

[2] MOST Cooperation, URL: http://www.mostnet.de

[3] FlexRay Consortium, URL: http://www.flexray.com

[4] OSEK/VDX-Konsortium, *Spezifikation eines Betriebssystems, eines Netzwerkmanagements und einer Kommunikationsschicht*, URL:http://www.osek-vdx.org, 2001.

[5] J. Eisenmann, M. Köhn, P. Lanchès, A. Müller. *Entwurf und Implementierung von Fahrzeugsteuerungsfunktionen auf Basis der Client/Server-Architektur*, VDI-Tagung „System-Engineering in der KFZ-Entwicklung“, 1997.

[6] P. Lanchès, J. Eisenmann, M. Köhn, J. Holland. *Client/Server Architecture - Managing New Technologies for Automotive Embedded Systems*, Convergence, 1998.

[7] A. Müller. *Client/Server-Architektur für Steuerungsfunktionen im KFZ*, it+ti – Informationstechnik und Technische Informatik, 1999.

[8] A. Müller. *Entwurfsmethodik und automatisierte Verteilung für Steuerungssoftware in einem verteilten Rechnersystem in der Automobilelektronik*. Dissertation Universität Tübingen, 1999.

[9] S. Boutin, *Architecture Implementation Language (AIL),* 1er Forum AEE, Guyancourt, März 2000,
URL: http://aee.inria.fr/forum/14032000/SB_Renault.pdf

[10] ITEA-Projekt VHE-Middleware,
URL: http://www.c-lab.de/vhe-middleware/

[11] Carnegie Mellon University, Software Engineering Institute, URL: http://www.sei.cmu.edu/plp/

[12] P. Clements, L. Northrop: *“A Framework for Software Product Line Practice”*, SEI Report, July 1999,
URL: http://www.sei.cmu.edu/plp/frameworkv2.7.pdf

[13] P. Donohoe, ed.*: “Software Product Lines; Experience and Research Directions”*; Proc. of the First Software Product Line Conference (SPLC 1), Denver, Aug. 2000, Kluwer Academic Press, 2000

[14] ITEA-Projekt Dess, URL: http://www.dess-itea.org/

[15] C. Scheidler, et. Al., *Systems Engineering of Time-Triggered Architectures – The SETTA Approach*, *DCCS 2000: 16th IFAC Workshop on Distributed Computer Control Systems,* Sydney, Australia, 29th November – 1st December, 2000.

[16] J. Ruh, U. Virnich, Hugo G. Chalé-Góngora, The SETTA automotive demonstrator - A design approach on the perspective of the automotive industry, ECEC2001, 8th European Concurrent Engineering Conference, Valencia, Spain 18-20 April, 2001.

[17] B. Aumann, W.-D. Gruhle, M. Sieger, O. Buchhold, F. König. Einsatz objektorientierter Methoden zur Entwicklung von Echtzeit Software - Fallbeispiel Getriebesteuerung. VDI Tagung Elektronik im Kraftfahrzeug, Baden-Baden, 1998.

[18] H. Krimmel, H. Deiss. Vom virtuellen Prototypen zur Seriensteuerung: Einsatz moderner Entwurfs- und Simulationshilfsmittel. 3.Stuttgarter Symposium Kraftfahrwesen und Verbrennungsmotoren, 1999.

[19] H. Deiss, H. Krimmel. Rapid Prototyping and Automatic Code Generation in Transmission Control. PROFES '99, Oulu/ Finnland, 1999.

[20] B. Aumann, H. Deiss, B. Müller, F. König. Konfigurierbarkeit, Schlüssel zur Wiederverwendung von Software. 10. Aachener Kolloquium, Aachen, 2001.

[21] H. Deiss, M. Dornseiff, H. Krimmel, E. Sax. Testing methods for complex control systems - Increased testing penetration through automated testing. FISITA, Helsinki, 2002.

[22] M. Wolff, U. Gillich, E. Bosch, W.-D. Gruhle. Softwareplattform und standardisierte Schnittstellen - Voraussetzungen für ReUse, Qualität und verteilte Systeme. VDI Tagung Elektronik im Kraftfahrzeug, Baden-Baden, 2003.

[23] P. Feulner, M. Sieger, F. König. Workflow-Management in der Software-Entwicklung - Effizienzsteigerung durch Wieder-verwendung von Prozessen. VDI Tagung Elektronik im Kraftfahrzeug, Baden-Baden, 2003.

[24] B. Hardung, M. Wernicke, A. Krüger, G. Wagner, F. Wohlgemuth. Entwicklungsprozess für vernetzte Elektroniksysteme. VDI Tagung Elektronik im Kraftfahrzeug, Baden-Baden, 2003.

Anlage 1: Erfolgskontrollbericht

Anlage 2: Kurzfassung