Integrative Entwicklungsprozesse am Beispiel einer automotiven Anwendung

Bernd van Vugt[1], Stefan Gläser[2]

[1]EXTESSY AG
Major-Hirst-Straße 11c
D-38442 Wolfsburg
b.vanvugt@extessy.com

[2]VOLKSWAGEN AG
Forschung Elektroniksysteme
Brieffach 1776
D-38436 Wolfsburg
stefan.glaeser@volkswagen.de

Abstract. Infotainment als immer wichtiger werdender Wettbewerbsfaktor für die Automobilindustrie erfordert branchenübergreifendes Denken und Planen, da zur Integration von multimedialen Funktionen ins Fahrzeug zwei höchst unterschiedliche Entwicklungsprozesse zusammengeführt werden müssen: Für ein neues Fahrzeugs kann man von einer Entwicklungszeit von ungefähr fünf Jahren ausgehen, wogegen der Entwicklungszyklus in der Informationstechnologie zwischen einem halben und zwei Jahren anzunehmen ist. Die Zusammenführung dieser Prozesse gestaltet sich nicht zuletzt deswegen problematisch, weil die Struktur der Automobilindustrie nahezu vollständig auf den bestehenden, konventionellen Entwicklungsprozess ausgerichtet ist und damit wenig Freiheitsgrade für eine adäquate Reaktion auf die dynamischen Anforderungen des Multimediamarktes läßt.

Hier kann ein durchgängig tool-unterstützter Entwicklungsprozess helfen, beide Prozesse zu einem einzigen zu verschmelzen. Eine besondere Bedeutung fällt dabei dem Requirements Engineering zu, welches Anforderungen für beide, Fahrzeug und Multimediasystem, als eine zentrale „Wissensbasis“ identifizieren und spezifizieren soll. Eine objektorientierte Vorgehensweise, welche die Schnittstelle zwischen einzelnen Modulen auf funktionaler Ebene definiert, ermöglicht die Entwicklung ausführbarer Modelle anhand eines digitales Lastenheftes. Die Komponenten dieser Modelle sind nun beliebig austauschbar. Es können dann nicht nur beliebig Module ersetzt, sondern auch Software-Module gegen echte Hardware ausgetauscht werden.

Auf diese Art erhält man einen Katalog von Anforderungen, welche, unabhängig von ihrer Granularität und Maturität, den Ausgangspunkt für eine konsistente Zusammenführung beider Entwicklungsprozesse darstellen.

1 Einleitung

Der Kundenwunsch nach Mobilität und Individualität prägt zur Zeit das Bild in der Automobilbranche. Dabei wird deutlich, dass die Bedürfnisse der Kunden sich nicht auf das Fahrzeug allein, sondern in verstärktem Maß auch auf die damit verknüpften Dienste (Services) gerichtet sind. Dies betrifft insbesondere Dienste zur Information und Unterhaltung der Fahrzeuginsassen.

Die Kraftfahrzeugindustrie hat bereits auf diese Bedürfnisse reagiert, dennoch besteht eine der großen Herausforderungen in der Auflösung des Konflikts, der sich aus den unterschiedlichen Lebenszyklen von Automobilen mit mehr als zehn Jahren und Produkten der Informationstechnologie mit Lebenszyklen kleiner als drei Jahren ergibt. Für die Automobilindustrie bedeutet dies, dass in den frühen Entwicklungsphasen eines Fahrzeugs Informationstechnologien und Standards berücksichtigt werden müssen, die erst in einigen Jahren verfügbar sein werden. Die zusätzliche Forderung nach Wartbarkeit und Aktualisierbarkeit von integrierten Infotainmentsystemen setzt eine flexible Systemarchitektur voraus, die nicht nur das Fahrzeug betrifft sondern auch auf die Umgebung bezieht [1].

Die Volkswagen Elektronik-Forschung befasst sich unter anderem mit Diensten, die insbesondere durch das Zusammenspiel zwischen Auto, Haus, Dienste-Anbietern im Internet und mobilen Endgeräten verfügbar sind oder voraussichtlich in absehbarer Zeit verfügbar sein werden. In Zusammenarbeit mit den Partnern Extessy AG, Nordsys OHG und Trajet GmbH wurde eine Plattform für das Fahrzeug entwickelt, die Informationen aus den Fahrzeugbussen und externen Netzen sammelt und zur weiteren Verarbeitung zur Verfügung stellt. Diese Verarbeitung kann fahrzeugintern durch interne Anwendungen geschehen, die Informationen können aber auch von einem Webserver zur Verarbeitung durch externe Dienste bereitgestellt werden [2].

Dieser Aufsatz beschreibt im nachfolgenden Abschnitt die zur Entwicklung dieser prototypischen Plattform verwendete Vorgehensweise. Die gewonnenen Erkenntnisse werden diskutiert und offene Punkte aufgezeigt. Der zweite Hauptteil wirft Fragen zu aktuellen Vorgehensmodellen auf und schlägt Alternativen dazu vor. Zum Schluss wird ein kurzer Ausblick auf die Umsetzungsmöglichkeiten dieser Ideen zur Entwicklung nachfolgender Prototypen gegeben.

2 Vorgehensweise im Projekt

[3], [4] definieren die Spezifikation als eine detaillierte Beschreibung der Teile eines Ganzen und ihrer Eigenschaften bezüglich Größe, Qualität, Performance usw. sowie ihrer Beziehungen untereinander. Folgende Fragen zu einem System müssen beantwortet werden:

- Was tut es?
- Worauf wirkt es?
- Unter welchen Bedingungen arbeitet es?
- Welchen Einschränkungen unterliegt es?

Der Begriff „Spezifikation“ kann aber nicht nur als Synonym zu einer Sammlung von Anforderungen gebraucht werden, sondern bezeichnet auch den Prozeß zur strukturierten

Entwicklung von Anforderungen [3]. Dieser Prozeß sei hier mit „Requirements Specification“ bezeichnet. Parallel dazu werden die in der Requirements Specification aufgestellten Anforderungen während des gesamten Verlaufs des Entwicklungsprozesses auf ihre Umsetzung und gegebenenfalls notwendige Änderung überprüft, was hier mit „Requirements Management“ bezeichnet werden soll. Beides, Requirements Specification und Requirements Management wird in diesem Aufsatz zu dem Begriff „Requirements Engineering“ zusammengefaßt [5].

Vorgehensmodelle zur Integration informationstechnisch geprägter Komponenten im Fahrzeug werden in der Automobilindustrie bereits seit einigen Jahren eingesetzt. Hieraus wachsen immer wieder neue Erkenntnisse insbesondere im Bereich der verteilten Entwicklung vernetzter Steuergeräte, den automotiven Anforderungen an Beschreibungsmittel, dem Einsatz von Werkzeugketten zur Unterstützung von Entwicklungsprozessen und auch zu den Entwicklungsprozessen selber. Nachfolgend soll dargestellt werden, welche Vorgehensweise zur Entwicklung der in [2] beschriebenen Plattform verwendet wurde und welche Erkenntnisse dabei gezogen wurden.

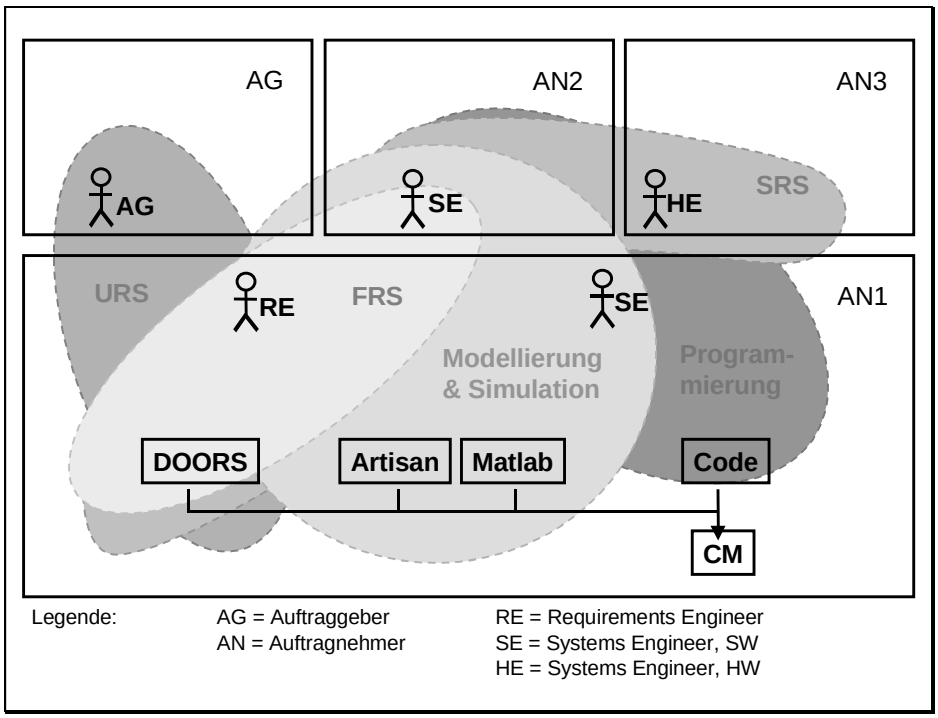

Abb. 2-1: Darstellung der Projektstruktur

Das methodische Vorgehen während der Projektdurchführung lässt sich am einfachsten anhand Abbildung 2-1 verdeutlichen. Die Einordnung der verwendeten Projektdokumente und Werkzeuge in das V-Modell ist in Abbildung 2-2 dargestellt.

Der Auftraggeber (AG), im Diagramm oben links, arbeitete mit dem Requirements Engineer zusammen eine erste Version der User Requirement Specification (URS) aus. Diese initiale URS wurde im Requirements Engineering (RE)-Tool „Telelogic DOORS 6.0“ erfasst, auf das beide Parteien einen definierten Zugriff erhielten. In der ersten Version enthielt die URS dabei nichts anderes als das, was in der vorangegengenen Akquisephase vertraglich zwischen AG und Auftragnehmer (AN) vereinbart wurde. Dies stellte sicher, dass zu Beginn des verschmolzenen Analyse- und Designprozesses eine grobe inhaltliche Abgrenzung (nicht

zuletzt juristisch) festgehalten werden konnte. Diese wurde nun iterativ „top-down“ weiter verfeinert. Mit zunehmender Spezifikationstiefe verschob sich dabei die Rolle des AG von der Requirements Specification zum Project Controling durch gezieltes Requirements Engineering.

Während des iterativen Verfeinerungsprozesses bestand die Aufgabe von Requirements Engineer und Systems Engineer für Software (SE), die Anforderungen des AG in einer Function Requirement Specification (FRS) auf Funktionen abzubilden. Diese erschlossen sich über Anwendungsfälle, die, zunächst rein textuell, ebenfalls in DOORS erfasst wurden.

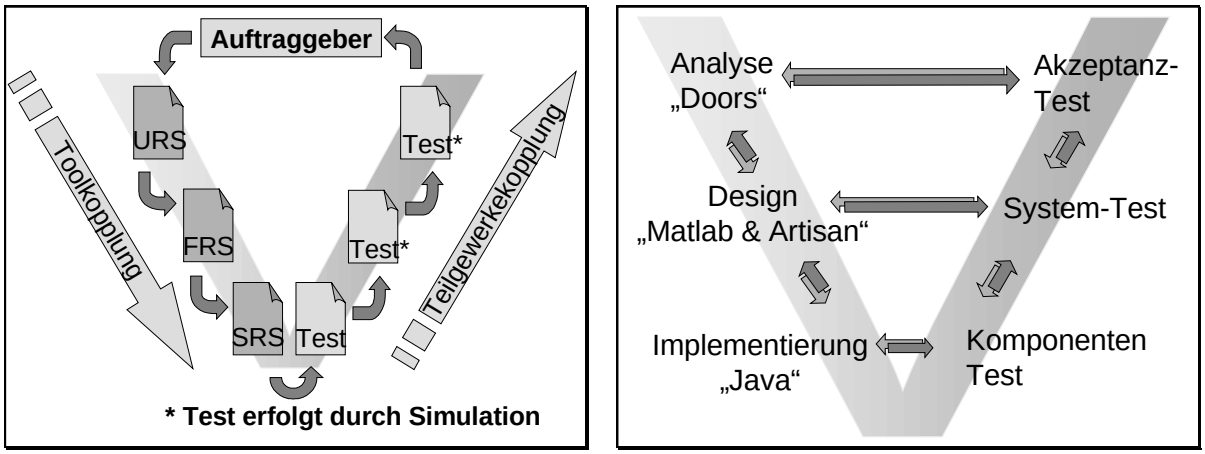

Abb. 2-2: Projektdokumente und Werkzeugkette im V-Modell.

Im Anschluß wurden die textbasierten Anwendungsfälle als UML-UseCase-Diagramme im Werkzeug „ARTiSAN RtS 4.2“ beschrieben, die dann ihrerseits beispielsweise über tiefer detailliertere Objekt Kollaborations Diagramme, Objekt Sequenz Diagramme (und andere) bis auf Klassendiagrammebene verfeinert werden konnten. Da das verwendete Werkzeug ebenso wie das zuvor beschriebene über eine eigene Datenbank nebst Änderungsverwaltung verfügt, kann so die Entstehungsgeschichte jedes einzelnen Modells bei Bedarf lückenlos nachvollzogen werden. Dies gilt auch für gebildete Verknüpfungen zwischen in DOORS erfassten Anforderungen und in ARTiSAN entwickelten Modellen, die über ein sogenanntes Surrogatmodul miteinander verknüpft werden können. So kann auch die Zuordnung zwischen den Requirements und den Modellen strukturiert verifiziert und validiert werden.

Im dem hier konkret beschriebenen Projekt reichte jedoch die an ausführbaren Modellen zur Verfügung stehende Vielfalt nicht aus: Diskrete Modelle, wie sie in ARTiSAN entwickelt und zur Ausführung gebracht werden können, sind oftmals ungeeignet um beispielsweise kontinuierliche Signalflüsse abzubilden. Dafür erschien das Werkzeug „Mathworks Matlab/Simulink R13“ geeignet, dessen Modelle sich über das inter-tool-engineering-Werkzeug „Extessy ExITE“ leicht mit denen aus der diskreten Modellwelt verbinden lassen.

Um die Versionierung, Konfigurierung und einen einheitlichen zentralen Datenbestand kümmerte sich dabei das Configuration Management (CM) Werkzeug „GNU WinCVS 1.3“, welches gleichzeitig parallele Zugriffe auf gleiche Daten verwalten kann. Hierüber wurden auch die Zugriffsrechte auf Modelle und Modellteile verwaltet, dabei erscheint allein der „exclusive edit“-mode empfehlenswert. Andere Versionierungstools wie beispielsweise das PVCS von Merant lassen sich sogar in ARTiSAN integrieren, so dass man sich in der Regel

nicht mit Fragen des An- und Abmeldens befassen muss. Leider ist dann aber auch der (einmalige) Konfigurationsaufwand relativ hoch.

Nach mehreren Iterationen der Analyse- und Design-Phase standen Aufbau und Umfang des zu erstellenden Produkts weitläufig fest. Nun konnten zwei Entwicklungsprozesse vorangetrieben werden: je einer für Dienste und Hardware. Letzterer benötigt Einblick in die zu entwickelnde Funktionalität (FRS) sowie das Produktumfeld (URS), um seinerseits *die richtige* Funktionalität bereitstellen zu können. Während der Evaluierungsphase des Systems Engineer für Hardware (HE), durch welche Kombination an Bauteilen und gegebenenfalls zusätzlicher Software die geforderte Funktionalität bereitgestellt werden kann, stellt er eine implizite Zuordnung zwischen FRS und der Systems Requirement Specification (SRS) her. Werden diese zeitgleich in DOORS erfasst, kann die Zuordnung zwischen FRS und SRS auch explizit erfolgen, was dann wieder den Vorteil der „traceable links“ mit sich bringt. So wurde zu jedem Zeitpunkt sichergestellt, dass eine klare Abgrenzung zwischen Projektinhalten und Nicht-Projektinhalten spezifiziert ist, welche sich dann auch eindeutig in der nachfolgenden Umsetzung wiederfindet. Der vom HE eventuell erzeugte Programmcode wurde im selben Projekt des verwendeten CM-Tools abgelegt, wie sämtliche anderen erzeugten (Modell-)Daten auch.

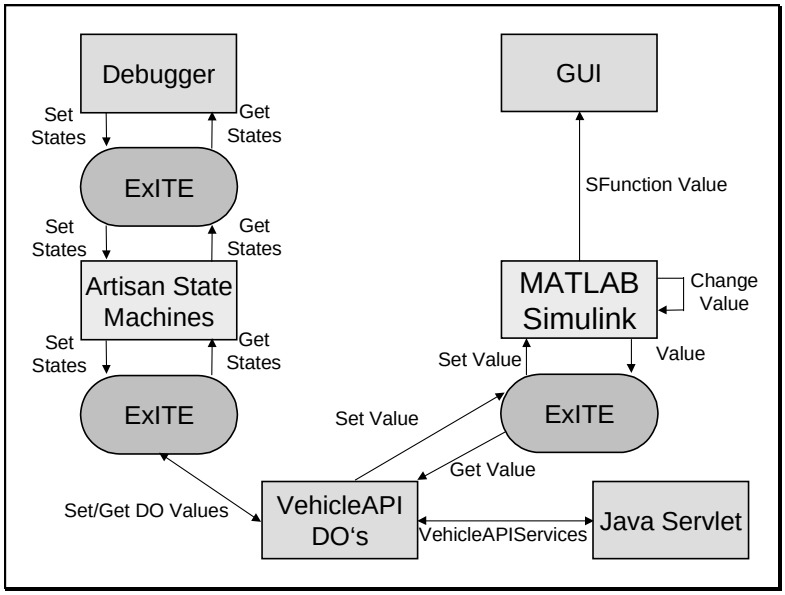

Abb. 2-3: Architektur des Entwicklungsprozesses aus Werkzeugsicht

Anhand der Darstellung der Architektur des Entwicklungsprozesse aus Werkzeugsicht (Abb. 2-3) wird der Vorteil modellbasierter Entwicklung deutlich. Der Modellanteil mit kontinuierlichem Signalfluss wurde in Matlab (rechts im Bild) entwickelt, der zustandsbasierte in ARTiSAN (links). Beide Modelle wurden an einen zentralen Datenspeicher („VehicleAPI DO‘s“, i.e DatenObjekte für Schnittstelle zu den Fahrzeugbussen) über das inter-tool-engineeing-Werkzeug ExITE angeschlossen. Anschliessend mussten nur noch die entsprechenden HMI in Java programmiert werden, die dem Zweck der Visualisierung und der Steuerung der Produkt-Funktionen dienten. Dies entspricht, wenn die entwickelten Funktionen auch wirklich das tun, was sie laut Requirement Engineering sollen, einem vollständigen Systemtest. Führt ihn der AG aus, kann auf dieser Basis die Produktabnahme erteilt werden.

Die für diesen Entwicklungsprozess verwendete Kette von Entwicklungswerkzeugen lässt sich schematisch anhand eines System Architektur Diagramms (Abb. 2-4) veranschaulichen.

SE und HE arbeiteten mit der Entwicklungsumgebung für Programmierer, wenn Software-Code manuell programmiert und nicht aus ARTiSAN-Modellen heraus generiert werden konnte respektive angepasst werden musste. Der Requirements Engineer bedient sich der DOORS- und ARTiSAN-Clients (Abb. 2-4, oben). Zum Entwickeln der ausführbaren Modelle wurde die Entwicklungsumgebung wie rechts im Bild schematisch angedeutet bereit gestellt. Als zentrale Informationsverwaltung diente allen beteiligten Personen der im CVS abgelegte Datenbestand als verbindliche Arbeitsgrundlage.

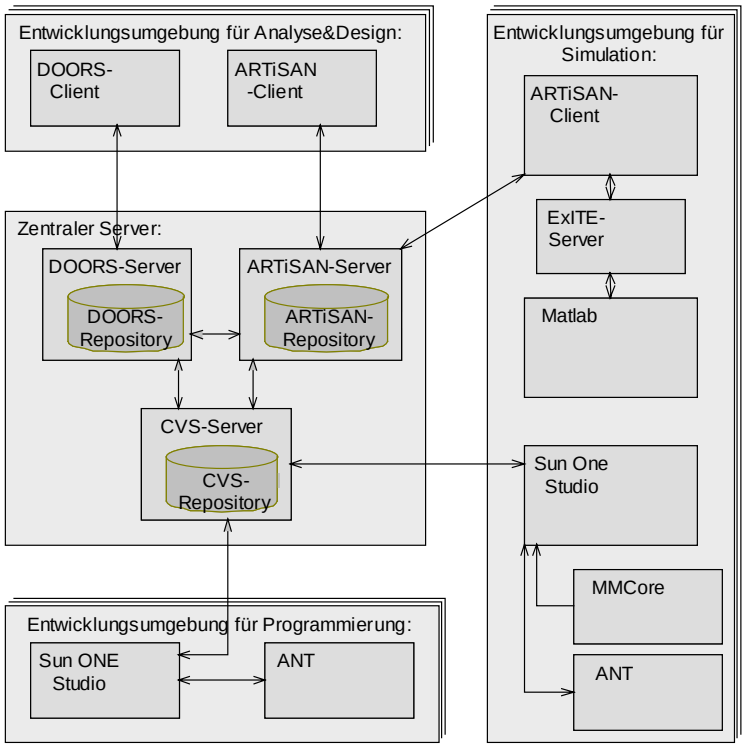

Abb. 2-4: System Architektur Diagramm

Die hier mittels Toolkopplung durchgängig gemachte Werkzeugkette ermöglichte eine strukturierte Vorgehensweise bei der Definition der Anforderungen, ihrer Verfeinerung und ihrer Verfolgung und Überprüfung an dem implementierten Prototypen. Der Forderung nach kurzen Entwicklungszeiten, die zur Integration von Informationstechnologie im Fahrzeug notwendig sind, kommt die enge Zusammenarbeit zwischen Auftraggebern und Auftragnehmern durch eine firmenübergreifende Engineering Platform entgegen. Während des Entwicklungsprozesses können so zeitnah kontinuierlich veränderbare Anforderungen an die Systemkomponenten der Informationstechnologie aufgenommen und umgesetzt werden.

Offen bleibt aber nach wie vor die Frage der Zusammenführung der Entwicklungsprozesse von Fahrzeug und Informationstechnologie. Hier versucht der nächste Abschnitt konstruktive Anregungen zu geben.

3 Neue Ansätze

Höhere Kundenanforderungen an Komfort und Sicherheit ermöglichten den Einzug von Elektronik ins Kraftfahrzeug. Da diese einer sehr viel schnelleren Evolution unterworfen ist, ist man dazu übergegangen, beide Entwicklungsprozesse voneinander zu trennen. Die Korrelation zwischen beiden Entwicklungsprozessen ist jedoch nach wie vor sehr hoch, was sich vor allem hinsichtlich querschnittlicher technischer Parameter (z.B. Gewichte, Einbaugrößen und Schnittstellen) als auch nicht-technischer (z.B. Finanzierung, Design, Logistik und Ressourcenplanung) nicht ändern wird.

Es scheint darüber hinaus durchaus sinnvoll, die Entwicklung von Software und die Elektronikentwicklung auseinanderzuhalten. Zwar bildet die Elektronik die Umgebung für die auf ihr implementierte Software und umgekehrt, aber Neuentwicklung, Wiederverwendbarkeit und Flexibilität stehen unter völlig anderen Voraussetzungen. Der Software-Entwicklungsprozess ist u.a. abhängig von der geforderten Funktion, der Komplexität des Produktes, der Zielplattform, der Wiederverwendbarkeit von Software-Modulen und der Anzahl der zur Entwicklung einsetzbaren Kapazitäten. Die Flexibilität in der Entwicklung ist aber in jedem Fall höher anzunehmen als bei den zuvor genannten Entwicklungsprozessen.

Zudem wird im Zeitalter von Kommunikation, Information, Unterhaltung und Mobilität mit dem Produkt „Auto“ nicht mehr das Fahrzeug allein, sondern in immer stärkeren Maß Dienste assoziiert, die diese Bedürfnisse dem Kunden bereitstellen. Speziell die große Gruppe der Informations- und Unterhaltungsdienste läßt die Anforderungen an Elektronik stetig steigen. Da Dienste sich weitestgehend fahrzeugunabhängig entwickeln, liegt die Idee nahe, auch diesen Entwicklungsprozess den bisher genannten Entwicklungsprozessen zu entkoppeln.

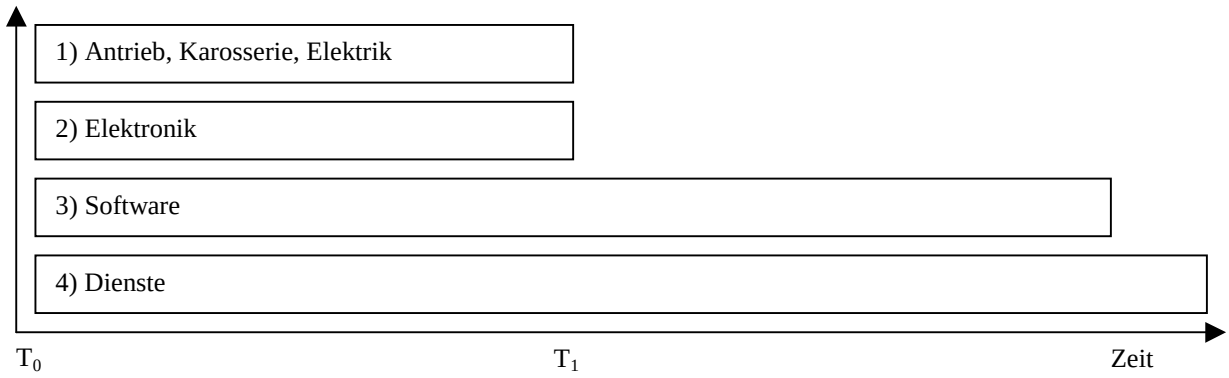

Abb. 3-1: Die vier Arten der Entwicklungsprozesse eines Fahrzeuges

Es kann also von vier korrelierten Entwicklungsprozessen ausgegangen werden (Abb.3-1), wobei maßgeblich nach wie vor der Entwicklungsprozess für Antrieb, Karosserie und Elektrik sein wird, an welchem sich die Entwicklungsprozess der Elektronik, der Software und mit Einschränkung die Entwicklungsprozesse der Dienste orientieren werden.

Allgemeine Phasenmodelle für die Entwicklung von der Idee zum Produkt lassen sich schwerpunktartig meistens in fünf bis sieben Phasen einteilen: Problemanalyse/Anforderungsdefinition, Systementwurf/Systemspezifikation, Komponenten-entwurf/Komponentenspezifikation, Implementierung/Installation, Betrieb/Instandhaltung [3]. Diese Phasen finden sich nicht nur in den hier dargestellten Entwicklungsprozessen selbst

wieder, sondern beschreiben gleichfalls die zu integrierenden Komponenten des angestrebten, integrativen Entwicklungsprozesses.

Unter einem integrativen Entwicklungsprozess sei in diesem Zusammenhang ein Entwicklungsprozess verstanden, der *sämtliche* zum Fahrzeug gehörenden Entwicklungsprozesse durchgängig und nahtlos miteinander verbindet („harte Integration“). Dieser Anspruch auf vollständige Integration erweist sich jedoch bei näherer Betrachtung als problematisch: Integrationsbemühungen, die beispielsweise auch die Integration der Fertigung (Implementierungsphase) anstreben, sind im Rahmen eines Projekts kaum durchführbar.

Bisher sind die vier Entwicklungsprozesse, wie in Abbildung 3-1 visualisiert, mehr oder weniger miteinander verwoben, jedoch nicht vollständig integrativ. Eine lose Kopplung der Prozesse existiert bereits. Eine vollständige Integration kann jedoch nur strukturiert und koordiniert erfolgen. In dem hier vorgestellten Projekt war die Vorgehensweise „bottom-up“, das heißt der Entwicklungsprozess für die Elektronik (i.e. HW-Komponenten) mit der Software (i.e. Steuerungsprogramme) und der Dienste (e. g. „Türen öffnen/schließen“) wurden durch Nutzung eines gemeinsamen Tools zur Anforderungserfassung (DOORS) und eines weiteren zur Modellbildung (ARTiSAN RtS) an einem zentralen Ort (GNU CVS) integrativ dargestellt. Eine durchgängig echte und somit „harte“ Integration hätte jedoch auch die Integration von Entwicklungsprozess 1 gefordert, was nicht Ziel dieses Projekts war. Auf dem Weg zu einem vollständig integrativen Entwicklungsprozess treten aber folgende Fragen auf:

- Der Entwicklungsprozess von Antrieb, Karosserie und Elektrik bleibt nach wie vor maßgebend für die Entwicklung des Fahrzeugs. ⇒ Können diese Anforderungen ebenfalls im verwendeten RE-Tool erfasst und ohne Technologiebrüche bis hin zur Produktion verknüpft werden?
- Mit der Möglichkeit, Software während der Laufzeit installieren, starten, stoppen, aktualisieren und deinstallieren zu können, endet der Entwicklungsprozess nicht zwangsläufig mit dem Produktionsstart T_1. ⇒ Es entstehen somit zwei Lebenszyklen: auf der einen Seite der des Fahrzeugs mit all seinen Komponenten und auf der anderen Seite der der Dienste. Können auch die Anforderungen an diese beiden Lebenszyklen im verwendeten RE-Tool erfasst und bis zum Zyklusende sinnvoll weiterbearbeitet werden?
- Dienste werden bis auf wenige Ausnahmen fahrzeugunabhängig und in der Mehrzahl sogar herstellerunabhängig sein. ⇒ Kann ein herstellerunabhängiges Sicherheits- und Nutzungskonzept sowie eine einheitliche Systemarchitektur definiert und weitreichend offengelegt werden?

Fest steht, dass der „harte“ integrative Entwicklungsprozess eine ganzheitliche Requirements Specification beinhaltet. Dieser berücksichtigt per Definition die Anforderungen aus allen vier Entwicklungsprozessen bereits bei der Ideenfindung zum Zeitpunkt T_0. Im Gegensatz dazu beinhaltet der hier realisierte Entwicklungsprozeß lediglich drei der vier identifizierten Prozesse: „Elektronik“, „Software“ und „Dienste“.

Auf dem Weg zu einer *vollständigen* Spezifikation im „harten“ integrativen Entwicklungsprozeß erscheint es deshalb sinnvoll, zunächst alle Anforderungen an die *Gesamtheit* der Anforderungen zu klassifizieren. Diese Gesamtheit der Anforderungen ist zu Projektbeginn höchstens implizit von AG und AN erfasst, jedoch in der Regel nicht explizitiert worden. Deshalb müssen auch sie neben den Produktanforderungen explizit erfasst und klassifiziert werden. Für eine vollständige Spezifikation empfehlen [3], [6] neben den funktionalen Anforderungen noch die Kategorien der störungsrelevanten, der sicherheitsrelevanten, der leistungs-, der implementierungstechnischen, der

umweltspezifischen, der wirtschaftlichen, der dynamischen und der gesetzlichen Anforderungen, wobei eine strikte Trennung in diese Anforderungskategorien als nicht immer möglich und notwendig angenommen wird. Wichtig ist jedoch neben einer solchen Strukturierung auch deren Zuordnung zu den einzelnen Entwicklungsphasen.

Die Granularität dieser Requirements Spezification wird zu diesem Zeitpunkt für die ins Fahrzeug integrierten Komponenten der Informationtechnologie deutlich unschärfer ausfallen als für die Komponenten des Fahrzeugs. Ein integrativer Entwicklungsprozess zeichnet sich daher durch eine Flexibilität aus, die einen Reifeprozess für die aus der Informationstechnologie generierten Anforderungen zuläßt.

4 Ist das Problem damit wirklich gelöst?

Bisher konnte eine Integration dort erfolgen, wo durchgängige Informationswege ohne Technologiebrüche gefunden respektive geschaffen werden konnten. Hier liegt, wie bereits beschrieben, das verwendete RE-Werkzeug aufgrund seiner Verknüpfung zum Design-Werkzeug hinsichtlich seiner Integrationsfähigkeit relativ weit vorn. Jedoch kann das RE, so wichtig es auch in seiner zentralen Rolle ist, nicht der Startpunkt für die Integration sein - zu viele Prozesse finden bereits, aus chronologischer Sicht, *davor* statt. Aus Sicht des RE erfolgte im durchgeführten Projekt eine „top-down“-Kopplung, mit dem RE an der top-Position. Da dem RE jedoch auch Phasen wie die Projektakquise vorausgehen, bleibt die Frage nach einer Kopplung in die vorangegangenen Phasen vom durchgeführten Projekt unberührt.

Darüber hinaus gibt es viele begleitende Prozesse, die sicherlich auch auf Werkzeugbasis zu integrieren blieben, was zu einer Integration zur Projektlaufzeit, also projektbegleitend, führen müsste. Ebenso ist auch die Frage nach der Integration des Entwicklungsprozesses bis in die Produktion bisher ungeklärt; zumindest auf Basis durchgängiger CASE-Tools.

Somit bleibt folgende unvollständige Liste integrationsbedürftiger Komponenten bestehen:

- Fehlermanagement
- Dokumentenmanagement
- Ideenmanagement
- Risikomanagement
- Qualitätsmanagement
- Sicherheitsmanagement

Hinter dem Begriff „Sicherheitsmanagement“ verbirgt sich die Forderung nach einem prozessbegleitenden Requirements Engineering für sämtliche Aspekte der Datensicherheit (Security), welche im schlimmsten Fall auch Einfluß auf die Sicherheit der Fahrzeuginsassen (Safety) haben. Speziell die Anforderungen an Datensicherheit wachsen in gleichem Maß wie die Entwicklungen auf dem Gebiet der Informationstechnologie voranschreiten.

Zu diesen gelisteten technischen Komponenten lassen sich sicherlich auch beliebig viele nicht-technische hinzuziehen wie beispielsweise das Projektmanagement, die Logistik und vieles mehr.

Welche dieser Komponenten in den Folgephasen integriert werden, ist bisher ungeklärt. Anzunehmen ist, dass eine allumfassende, projektgebundene Integration nicht möglich ist und

eine Gesamtintegration mit kaum vertretbaren Aufwand verbunden wäre, zumal die Frage nach der Integrationsfähigkeit der verwendeten und zu verwendenden Tools separater Überprüfung bedarf und möglicherweise ein KO-Kriterium darstellt.

5 Literatur

[1] Ricky Hudi, Michael Bartl, Robert Tappe. „Infotainment-Systeme im Fahrzeug“. Sonderheft Elektronik Automotive S. 42 bis 48, September 2001, Elektronik, Poing, Band 50 (2001)

[2] Andreas Lübke, Amer Aijaz, Stefan Gläser, Jens Krüger, Stefan Schmalwaßer, Urs Thürmann, Sebastian Wegrowski. „Eine Telekommunikationsanlage für den Einsatz im Kraftfahrzeug“. In: Zentrum für Verkehr der Technischen Universität Braunschweig (Hrsg.). „IMA 2002- Informationssysteme für mobile Anwendungen“. Beiträge zum gleichnamigen 1. Braunschweiger Symposium vom 16. und 17. Oktober 2002, S. 47-61. Technische Universität Braunschweig. Tagungsband. VDE Verlag GmbH, Berlin, Offenbach.

[3] Eckehard Schnieder. „Methoden zur Automatisierung – Beschreibungsmittel, Modellkonzete und Werkzeuge für Automatisierungssysteme“. Vieweg Verlag, Braunschweig/Wiesbaden, 1999

[4] H.-J. Schneider. „Lexikon Informatik und Datenverarbeitung Version 4.0“ Oldenbourg Verlag, München u.a., 1997

[5] Sebastian Kuschel. „Requirements Engineering“. Diplomarbeit. Lehrstuhl für Hochfrequenz- und Kommunikationstechnik, IESK, Otto-von-Guericke-Universität Magdeburg / Volkswagen, 2003

[6] A. Weinberg. „Entwicklung einer Methode für den Systementwurf mit Netzen“. Diplomarbeit, Institut für Datenverarbeitungsanlagen, TU Braunschweig, 1987