Fortentwicklung einer automotiven Anwendung mit Hilfe eines integrativen Entwicklungsprozesses

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

Zusammenfassung. Infotainment ist in der Automobilindustrie zu einem entscheidenden Wettbewerbsfaktor geworden. In diesem Zusammenhang entsteht eine Dienstelandschaft rund um das Fahrzeug, die dem zunehmend auf Kommunikation ausgerichteten Kundenbedürfnis gerecht werden soll. Dies geht einher mit steigenden Anforderungen an die Systemarchitektur im Fahrzeug, welche die hohe Entwicklungsdynamik in der Informationstechnologie berücksichtigen muss.

Dieser Aufsatz beschreibt das methodische Vorgehen bei der Entwicklung eines Forschungsprototypen, der als intelligentes Gateway die zentrale Position einer flexiblen Systemarchitektur darstellt. Beginnend bei der textuellen Erfassung der Nutzeranforderungen, deren Verfeinerung und Darstellung als Systemmodell bis hin zur tatsächlichen Umsetzung wird der Werdegang des Prototypen skizziert. Der dabei verwendete integrative Entwicklungsprozess führt die verschiedenen Fahrzeug-Entwicklungsprozesse von Elektronik und Software mit dem Dienst-Entwicklungsprozess zusammen.

Unterstützt wird dieser Prozess durch einen durchgängigen Werkzeugverbund. Dieser ermöglicht die Nachverfolgung und die Aktualisierung der spezifizierten Anforderungen während der gesamten Entwicklungsphase. Die Entwicklungsumgebung konzentriert sich dabei um einen Software-Demonstrator. Damit kann der Auftraggeber bereits in der frühen Projektphase die Funktionalität überprüfen. Darüber hinaus können Funktionen, die noch nicht real im Hardware-Demonstrator umsetzbar sind, mit dem Software-Demonstrator simulativ dargestellt werden.

1 Einleitung

Information, Kommunikation und Unterhaltung, kurz Infotainment, ist zu einem entscheidenden Wettbewerbsfaktor geworden, denn die Nutzergruppe erstreckt sich mittlerweile über den gesamten privaten und dienstlichen Anwenderbereich. Liegen die Interessen der privaten Nutzer deutlicher bei einer angenehmen Gestaltung der Fahrt, so steht für den Dienstwagennutzer Effizienz im Vordergrund.

Um diesen stetig wachsenden Bedürfnissen gerecht zu werden, entwickelt sich eine Dienstelandschaft, deren Dynamik den technologischen Entwicklungen der Informationstechnologie entspricht. Für die Automobilindustrie bedeutet dies, dass in den frühen Entwicklungsphasen eines Fahrzeugs bereits Technologietrends berücksichtigt werden müssen, für die erst in einigen Jahren Produkte verfügbar sind. Die zusätzliche Forderung nach Wartbarkeit und Aktualisierbarkeit von integrierten Infotainmentsystemen setzt eine flexible Systemarchitektur voraus [1].

Die Volkswagen Elektronik-Forschung hat in Zusammenarbeit mit den Partnern Extessy AG, Nordsys OHG und Trajet GmbH eine Fahrzeugplattform entwickelt, die Informationen aus den Fahrzeugbussen und externen Netzen sammeln und zur weiteren Verarbeitung zur Verfügung stellen kann. Diese Verarbeitung kann durch fahrzeuginterne Anwendungen geschehen, die Informationen können aber auch von einem Webserver zur Verarbeitung durch externe Dienste bereitgestellt werden [2].

Für die Entwicklung dieses Prototypen galt es einen Prozess zu definieren, der es ermöglicht, durch klar definierte Projektarbeit zu *schnellen Ergebnissen* in Form von *realitätsnahen Prototypen* mit *hoher Wiederverwendbarkeit* durch *transparente Dokumentation* zu kommen. Der Transfer der erzielten Forschungsergebnisse in die Abteilungen der Entwicklung sollte hiermit optimiert werden. Das durch diesen Prozess erreichte Endergebnis kann daher als *wesentlicher Teil* einer Anforderungsspezifikation für ein zukünftiges Produkt in den darauf aufbauenden Entwicklungsprozess einfließen.

Dieser Aufsatz gibt zunächst einen Überblick über die Projektarbeit und stellt die Hardware- und Software-Architektur des Prototypen dar. Der Prozess, der bei der Entwicklung des Prototypen verwendet wurde, ist im darauf folgenden Kapitel dargestellt. Hier wird auch die Tool-Kette erläutert, die diesen Prozess unterstützt. Im anschließenden Kapitel wird kurz auf die Projektorganisation eingegangen. Zum Schluss werden die Projektergebnisse zusammengefasst und ein kurzer Ausblick auf die Umsetzungsmöglichkeiten der offen gebliebenen Punkte gegeben.

2 Ein intelligentes Fahrzeug-Gateway

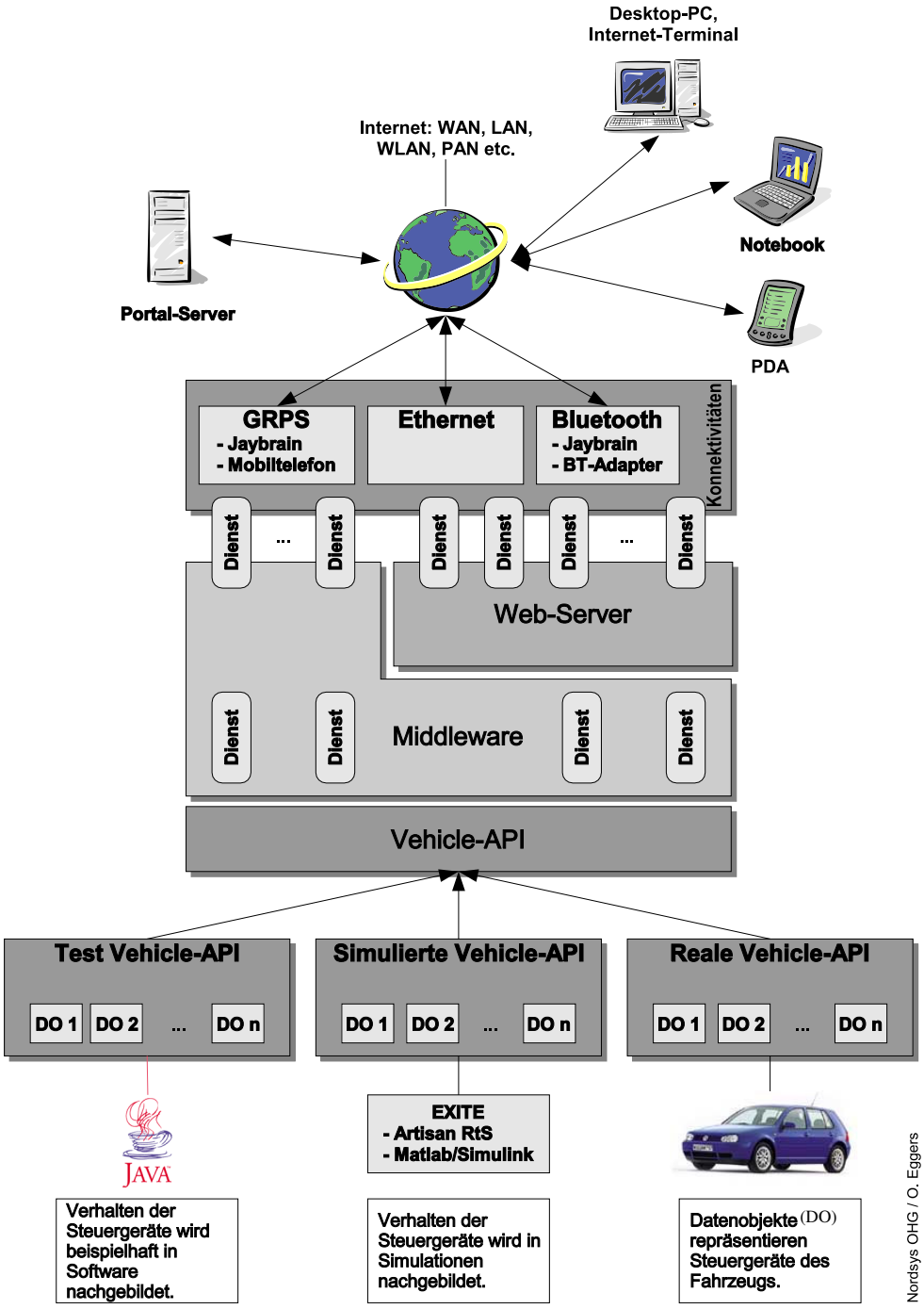

Abb. 2.1: iGate-Demonstrator — Systemübersicht

Gemeinsam mit den oben genannten Partnern wurde der Prototyp für ein intelligentes Gateway (iGate) entwickelt, welches einen kontrollierten Zugriff auf die Fahrzeugnetze und

die Kommunikation des Fahrzeugs mit der Außenwelt ermöglicht. Gleichzeitig bietet es eine Ablaufumgebung für Anwendungen.

Bei diesen Anwendung kann es sich sowohl um fahrzeugspezifische Anwendungen wie zum Beispiel Diagnose als auch um Infotainment-Anwendungen handeln. Ist beabsichtigt, auch so genannte "Third-Party-Applications" zuzulassen, müssen diese aus Sicherheitsgründen von Diagnose-Anwendungen vollständig getrennt ablaufen. Auf die architekturelle Partitionierung nach sicherheitskritischen und -unkritischen Anwendungen wird in diesem Aufsatz allerdings nicht weiter eingegangen sondern auf [2] verwiesen.

Abbildung 2.1 zeigt den Prototypen "iGate" in der Systemübersicht. Als Hardware-Plattform dient "jaybrain" der Trajet GmbH. Sie bietet ein GPS-Modul, das über eine GPS-API ansteuerbar ist, sowie die Konnektivitäten GPRS, Bluetooth und Ethernet. Die neuere Version hat zudem WLAN integriert. Das Fahrzeug kann somit auf das Internet zugreifen, ist seinerseits aber auch durch einen Web-Server für definierte Nutzergruppen (zum Beispiel der Fahrzeugbesitzer, -nutzer oder die Werkstatt) im Internet sichtbar. Der gesicherte Zugriff auf das Fahrzeug ist möglich von einem Portal-Server, wie er zum Beispiel beim Fahrzeughersteller stehen könnte, vom Home-PC des Fahrzeugbesitzers sowie von mobilen Endgeräten des aktuellen Fahrzeugnutzers. Die Kommunikations-Middleware bietet zudem eine Ablaufumgebung für Dienste, wie eingangs bereits erwähnt.

Die Fahrzeugbusse werden mittels einer Vehicle-API abstrahiert. Diese generische API ermöglicht das Entwickeln von Diensten ohne detaillierte Kenntnisse der Fahrzeug-Netzwerke. Dabei bildet die simulierte Spezialisierung der Vehicle-API die Schnittstelle zu einem Software-Demonstrator, welcher einerseits während des gesamten Projektverlaufs dem Auftraggeber zur Validierung der geforderten Funktionen dient, andererseits die Software-Entwicklung unterstützt und als Spezifikation für die Implementierung fungiert. Für die Test-Vehicle-API ist das Verhalten der Steuergeräte beispielhaft in Java nachgebildet. Sie ist Teil der Entwicklungsumgebung für die Implementierung. Die reale Vehicle-API hingegen steuert über die realen Fahrzeugbusse die Steuergeräte im Versuchsträger an. Diese drei Spezialisierungen ermöglichen die Substitution realer Hardware-Komponenten des Fahrzeugs durch simulierte Komponenten und umgekehrt. Dies hat entscheidende Vorteile für die Entwicklung, da somit die Entwicklungsumgebung flexibler, skalierbarer und kostengünstiger gestaltet werden kann. So ist es zum Beispiel nicht notwendig, dass eine reale Target-Hardware oder ein Fahrzeug für jeden Entwickler zur Verfügung steht.

3 Vorgehensweise im Projekt

Für die Entwicklung des im vorangehenden Kapitels beschriebenen Prototypen galt es einen Entwicklungsprozess zu finden, der zu den speziellen Rahmenbedingungen eines solchen Forschungsprojektes paßt. Nun weisen Entwicklungsprozesse mindestens drei Charakteristika auf, die sich bei der Entwicklung komplexer Systeme als problematisch erweisen können:

- Sie sind nicht "top-down" implementierbar (vom Management zu den Entwicklern) sondern müssen "gelebt" werden.
- Prozesse können nicht losgelöst von den Entwicklungswerkzeugen betrachtet werden, was nicht zuletzt auch den zuvor genannten Punkt betrifft.
- Prozesse interagieren mit ihrer Umgebung, zum Beispiel mit anderen, parallel laufenden Prozessen.

Ein Entwicklungsprozeß, der mehrere Werkzeuge mit einbeziehen soll, muss schon vor seiner Implementierung darauf ausgelegt sein, die Schnittstellen zwischen den benötigten Werkzeugen berücksichtigen zu können. Solche Schnittstellen zu anderen Werkzeugen werden in der Regel nicht bereitgestellt. Für komplexe Systeme, wie zum Beispiel ein Kraftfahrzeug, sind jedoch Werkzeuge erforderlich, die sich gut miteinander koppeln lassen.

Vor diesem Hintergrund ist diesem Projekt so vorgegangen worden, dass die zu verwendenden Tools den Prozess gestalteten und nicht umgekehrt: Angelehnt an den Step-X-Prozess [3] wurde hier der "bottom-up"-Ansatz verwendet und zuerst die *Auswahl* der Werkzeuge vorgenommen. Hierbei handelt es sich um Telelogic DOORS, Artisan RtS und Mathworks Matlab/Simulink, welche eine Teilmenge der im Step-X-Prozess referenzierten Werkzeuge darstellen. Diese Auswahl an Werkzeugen wurde projektspezifisch unter anderem auf ihre Verwendbarkeit und ihre Kopplungsmöglichkeiten hin überprüft, so dass zunächst die integrierte Entwicklungslandschaft aufgesetzt wurde, wie in [4] beschrieben.

Allgemeine Phasenmodelle für die Entwicklung von der Idee zum Produkt lassen sich nach [5] schwerpunktartig meistens in fünf bis sieben Phasen einteilen: Problemanalyse/Anforderungsdefinition, Systementwurf/Systemspezifikation, Komponentenentwurf/Komponentenspezifikation, Implementierung/Installation, Betrieb/Instandhaltung. Die Spezifikation ist danach definiert als eine detaillierte Beschreibung der Teile eines Ganzen und ihrer Eigenschaften bezüglich Größe, Qualität, Performance usw. sowie ihrer Beziehungen untereinander. Folgende Fragen müssen nach [5] zu einem System beantwortet werden:

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

Zu Beginn eines Forschungsprojektes kann dies in der Regel nicht zufriedenstellend geklärt werden. Der zu definierende Entwicklungsprozess muss also mit zum Teil vagen Anforderungen und wechselnden Randbedingungen zurechtkommen. Dennoch soll das *Ergebnis* eines Forschungsprojektes von so hoher Qualität sein, dass sie direkt in die erste Phase (Problemanalyse/Anforderungsdefinition) des Produktentstehungsprozesses der Entwicklungsabteilungen einfließen können. Die hier skizzierte Vorgehensweise zeigt einen Weg, wie dies zu erreichen ist (siehe hierzu auch [4]).

Eine der größten Herausforderungen der Automobilindustrie besteht in der Auflösung des Konflikts, der aus den unterschiedlichen Produktlebenszyklen von Fahrzeug und Informationstechnologie resultiert. Die im vorangegangenen Kapitel beschriebenen Spezialisierungen der Vehicle-API eröffnen neben den bereits erwähnten Vorteilen die Möglichkeit, genau diese Herausforderung deutlich besser zu bewältigen. Das dynamische Zusammenspiel und die modulare Austauschbarkeit von realen Hardware-Komponenten und simulierten Komponenten ermöglicht das Schritthalten mit den fortschreitenden Entwicklungen der Informationstechnologie während des Entwicklungsverlaufs eines Fahrzeugs. Die Konstellation der Projektbeteiligten und ihre Rollen in der Projektstruktur ergab sich aus genau diesen Anforderungen; sie ist in Abb. 3.1 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 Management (RM)-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 vorangegangenen Akquisephase vertraglich zwischen AG und Auftragnehmer (AN) vereinbart wurde.

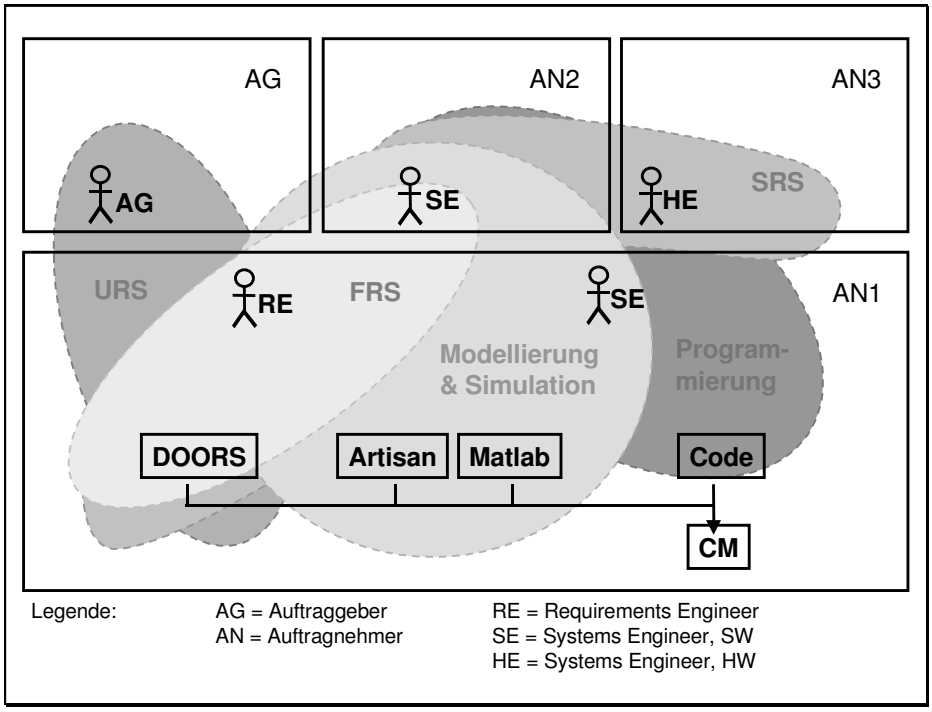

Abb. 3.1: Darstellung der Projektstruktur

Während des nachfolgenden 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. Die Modellierung dieser Anwendungsfälle wurde in UML mit dem CASE-Tool ARTiSAN RtS 4.2 durchgeführt. 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. Für die Modellbildung kontinuierlicher Signalflüsse wurde das Werkzeug Mathworks Matlab/Simulink R13 herangezogen, dessen Modelle sich über das inter-tool-engineering-Werkzeug „Extessy ExITE“ leicht mit denen aus der diskreten Modellwelt verbinden lassen.

Der Systems Engineer für Hardware (HE) evaluiert die Kombination an Bauteilen und gegebenenfalls zusätzlicher Software für die geforderte Funktionalität und stellt so 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.

Um die Versionierung, Konfigurierung und einen einheitlichen zentralen Datenbestand kümmerte sich das Configuration Management (CM) Werkzeug GNU WinCVS 1.3, welches gleichzeitig parallele Zugriffe auf gleiche Daten verwalten kann. Über CVS wurden auch die Zugriffsrechte auf Modelle und Modellteile verwaltet. Auch der vom HE eventuell erzeugte Programmcode wurde hier abgelegt.

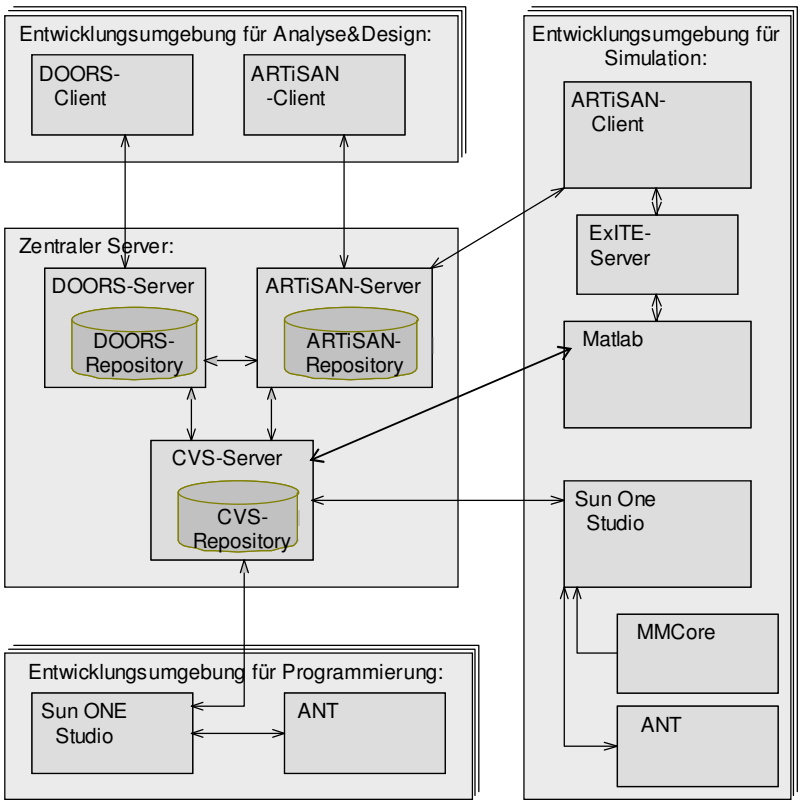

Abb. 3.2: System Architektur Diagramm

Der für diesen Entwicklungsprozess verwendete Verbund von Entwicklungswerkzeugen lässt sich schematisch anhand eines System Architektur Diagramms (Abb. 3.2) veranschaulichen. SE und HE arbeiteten mit der Entwicklungsumgebung für Programmierer (bestehend aus SunONE Studio und dem make-Tool ANT), 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 (im Bild oben). Zum Entwickeln der ausführbaren Modelle wurde die Entwicklungsumgebung, wie rechts im Bild schematisch angedeutet, bereitgestellt. Als zentrale Informationsverwaltung diente allen beteiligten Personen der im CVS abgelegte Datenbestand als verbindliche Arbeitsgrundlage. 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. Während des Entwicklungsprozesses können so zeitnah kontinuierlich veränderbare Anforderungen an die Systemkomponenten der Informationstechnologie aufgenommen und umgesetzt werden. Somit existiert zu jedem Zeitpunkt *genau eine zentrale, verbindliche* Datenbasis.

Zur Visualisierung und Steuerung der Produkt-Funktionen dient ein in JAVA programmiertes HMI, was mit dem Nordsys "MMCore" hergestellt wurde. Verhalten sich die entwickelten Funktionen auch wirklich so, wie es laut der Requirement Specification gefordert ist, entspricht dies einem vollständigen Systemtest. Dieser muss vom AG ausgeführt werden und dient damit als Basis für die Produktabnahme.

Den Gesamtüberblick über die Vorgehensweise sowie die Einordnung der verwendeten Projektdokumente und Werkzeuge in das V-Modell gibt Abb. 3.3. Konkret wurden so die Anpassung der Basis-Hardware „jaybrain“, die Entwicklung der Software-Architektur sowie die Entwicklung eines prototypischen Dienstes miteinander koordiniert. Von den vier Klassen der in [4] identifizierten Entwicklungsprozesse eines Fahrzeugs Antrieb/Karosserie/Elektrik, Elektronik, Software und Dienste wurden daher zumindest die drei letztgenannten zu einem integrativen Entwicklungsprozess zusammengeführt. Dabei wird deutlich, dass es nicht *den* integrativen Entwicklungsprozess geben kann, sondern dass dieser nach der hier beschriebenen Vorgehensweise von Fall zu Fall spezifiziert werden muss.

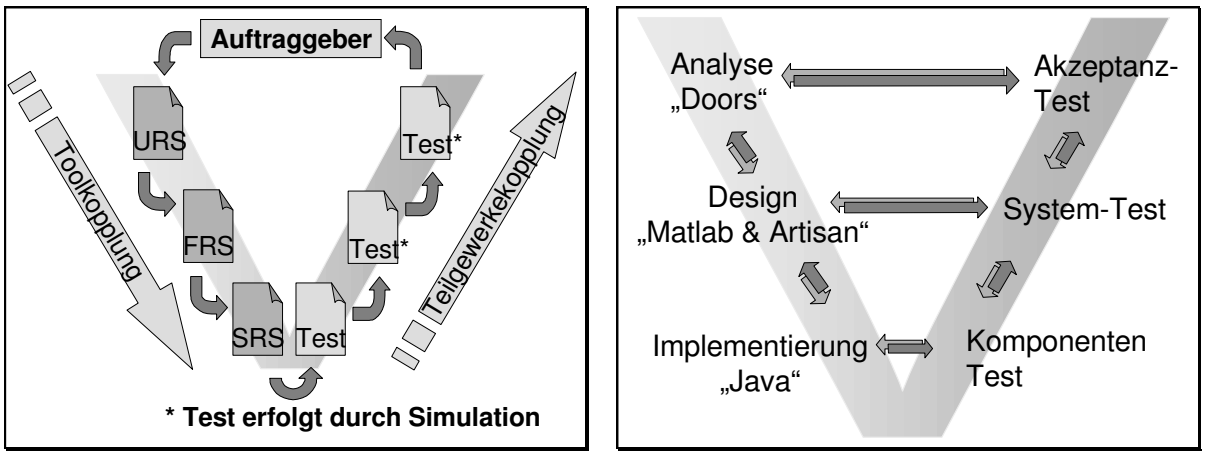

Abb. 3.3: Projektdokumente und Werkzeugkette im V-Modell.

Der iGate-Prototyp selber ist ein Power PC mit Embedded Linux und den Abmaßen 30 mm x 112 mm x 124 mm. Er ist integriert in einen serienmäßigen Golf A4. Zur Zeit wird lediglich ein CAN angesteuert, die Erweiterung auf vier CAN-Anschlüsse kommt jedoch in Kürze. Darüber hinaus bietet das iGate die Konnektivitäten Bluetooth und GPRS. Diese werden für den prototypischen Dienst genutzt, der den Fahrzeuginsassen nicht nur den Zugriff auf das Internet ermöglicht, sondern den Besitzer oder die Service-Werkstatt in die Lage versetzt, Daten drahtlos vom Fahrzeug zu bekommen. Der Ethernet-Anschluss wird zur fahrzeuginternen Kommunikation mit weiteren möglichen Infotainment-Komponenten verwendet. Die neuen Versionen des iGate können zudem über WLAN Verbindung nach außen aufnehmen, was die Nutzung so genannter "Hot Spots" möglich macht.

Das Ergebnis hat aufgrund Zusammenspiels von Software- und Hardware-Demonstrator eine hohe Transparenz bezüglich seiner Funktionalität und seiner Architektur. URS, FRS und SRS bieten eine Dokumentation und Spezifikation auf hohem Niveau. Der Wissenstransfer in die Vorentwicklung ist dadurch bestmöglich vorbereitet.

4 Projektorganisation

Da sämtliche für die Werkzeugkette verwendeten Tools auf Betriebssystemen von Microsoft laufen, wurde beschlossen, für die Projektorganisation MS-Word zu verwenden. Es ist am weitesten verbreitet und daher den meisten Nutzern vertraut. Hiermit lassen sich die elementarsten Aufgaben zur Projektorganisation erledigen. Drei Elemente dienen dabei als Basis:

- ein zentrales Navigationsdokument
- eine geordnete Verzeichnisstruktur
- Dokumentvorlagen

Das zentrale Navigationsdokument enthält Informationen über

- Organisation (z.B. über Aufbau und Ablauf des Projektmanagements)
- Inhalt (z.B. wie sie durch Analyse und Design in CASE-Tools wie DOORS® explizit dargestellt sind)
- Personen (wie sie durch Abbildung persönlicher Arbeitspakete im Gesamtbezug entstehen)

Es ist als aktives Dokument aufgesetzt so dass jedes Projektdokument mit dem zentralen Navigationsdokument verlinkt ist. Es enthält zudem (passive) Verweise auf die URS, FRS und SRS.

Für die Projektdokumete wurde eine Ordnerstruktur mit folgender obersten Hierarchieebene aufgesetzt:

- "0_ProjektORGA" enthält dabei die zur Verfügung gestellten Vorlagen.
- "1_ProjektAkquise" beinhaltet alle Dokumente, die bereits *vor* dem Zustandekommen des Projektvertrages anfallen.
- "2_ProjektArbeit" enthält alle zum Projekt zugehörigen Entwicklungsdokumente.
- "3_ProjektAbschluß" beinhaltet Übergabedokumente, Abnahmeprotokolle, etc.

Zudem wurden Dokumentvorlagen und Nomenklaturrichtlinien verwendet, wie sie in [6] genauer beschrieben sind.

Die verschiedenen Informationen und Daten für unterschiedliche Rollen/Personen wurden an einem Ort *zentral* bekannt gegeben und größtenteils direkt verlinkt. Damit war die Informationssicherheit und -verfügbarkeit projektweit jederzeit gewährleistet. Die Kontrolle des Projektverlaufes und der zu erwartenden Ergebnisse sowie die Koordination der beteiligten Projektpartner lief problemlos. Eine wirkliche aktive Kopplung an den für die Entwicklung notwendigen Werkzeugverbund ist allerdings nicht implementiert.

5 Ist das Problem damit wirklich gelöst?

Der hier genutzte Prozess zur Entwicklung von Forschungsprototypen hat sich während des gesamten Projektverlaufs als vorteilhaft erwiesen. Was zu Beginn eine vage Idee war, hat sich kontinuierlich zu einem Prototypen entwickelt, dessen reale Hardware-Komponenten durch simulierte Module ersetzbar sind und umgekehrt. Somit kann die Entwicklungsumgebung

flexibler, skalierbarer und kostengünstiger gestaltet werden. Die entstandenen Dokumente sind aufgrund ihrer Struktur und der Nachvollziehbarkeit hinsichtlich der Umsetzung jeder einzelnen Anforderung über die Software-Modelle bis hin zum Source Code trotz ihrer Komplexität klar zu überblicken und zu verstehen. Die Übergabe der Forschungsergebnisse in die Entwicklungsabteilungen sind also optimal vorbereitet.

Grundsätzlich muß beim Aufsetzen eines Projektes gewährleistet sein, dass alle Projektbeteiligten die gleiche Arbeitsumgebung (zum Beispiel Software-Versionen der Entwicklungs-Tools) haben, damit alle Personen auf die für sie relevanten Informationen zugreifen können. Um eine Durchgängigkeit der Werkzeugkette mit bestmöglicher Minimierung der Technologiebrüche zu erreichen, sind Laufzeit-Kopplungen oder zumindest direkte strukturelle Kopplungen erforderlich, was in diesem Projekt nicht in jedem Fall erreicht werden konnte. Hier ist noch Anpassungsarbeit erforderlich, die zum Teil mit erheblichem Aufwand verbunden sein kann.

Die Integration der Entwicklungsprozesse von Elektronik, Software und Diensten wurde zwar umgesetzt (siehe hierzu auch [4]), eine durchgängig echte Integration hätte jedoch auch die Entwicklungsprozesse von Karosserie, Motor, Getriebe, Elektrik gefordert, was allerdings nicht Ziel dieses Projekts war. Darüber hinaus gibt es viele begleitende Prozesse, die, sicherlich auch auf Werkzeugbasis, zu integrieren bleiben [4]. Bisher konnte eine Integration dort erfolgen, wo durchgängige Informationswege ohne Technologiebrüche gefunden respektive geschaffen werden konnten.

Dennoch schafft das hier abgebildete Gesamtsystem aus Prozess und Werkzeugen eine tragfähige Basis für die Integration projektrelevanter Informationen in *eine digitale* Datenbasis, wie sie von verschiedenen Rollen/Personen in verschiedenster Form benötigt werden - vom Projekt-Akquisiteur über den Projekt-Mitarbeiter bis zurück zum Auftraggeber.

6 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] STEP-X. "Strukturierter Entwicklungs-Prozess für Automobil-Anwendungen". http://www.step-x.de/

[4] Bernd van Vugt, Stefan Gläser. „Integrative Entwicklungsprozesse am Beispiel einer automotiven Anwendung“. In: „2. Requirements Engineering Tagung“, Beiträge zur gleichnamigen Tagung am 10. und 11. März 2003 in München. Tagungsband. 2003

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

[6] Ramona Völker. "Analyse und Design der Anforderungen an Projektmanagement im kooperativen Managementtool-Verbund". Diplomarbeit, vorgelegt an der Fachhochschule Würzburg-Schweinfurt im Fachbereich Informatik und Wirtschaftsinformatik bei Prof. Dr. J. Spielmann, 2003.