

## Einbindung bestehender Steuergerätesoftware in die Autosar-Architektur

„Cooperate on standards – compete on implementation“, so lautet der Grundsatz der Standardisierungsinitiative Autosar. Ein lange gehegter Wunsch der Automobilindustrie ist auf dem besten Weg, Realität zu werden. Die Branche ist dabei, den dafür notwendigen Paradigmenwechsel zu vollziehen. Vector Informatik beschreibt in diesem Beitrag Methoden und Werkzeuge, mit denen das Unternehmen die Autosar-Idee umsetzt.

## 1 Einführung

Steuergeräte waren in der Vergangenheit proprietäre Lösungen – höchst individuelle Komponenten, die hersteller- und fahrzeug-spezifisch entwickelt wurden. Doch die Vorgehensweise der steuergeräteezentrierten Entwicklung hat Nachteile. Eine einzige Änderung in den Anforderungen kann eine komplette Neuentwicklung des Steuergerätes notwendig machen. Mit zunehmender Komplexität der Steuergeräte-funktionalität ist dies mit einem enormen Aufwand verbunden. Neue Randbedingungen wie die Abbildbarkeit einer noch höheren Varianten- und Ausstattungsvielfalt, die Upgrade-Fähigkeit von Architekturen und nichtfunktionale Anforderungen wie Steuergeräte-Diagnose oder Verfügbarkeit verstärken die Frage nach einer konzeptionellen Neuorientierung.

Viele Hersteller und Zulieferer machten sich deshalb schon vor Jahren Gedanken, wie diese steuergeräteezentrierte Entwicklung durch einen funktionsorientierten Ansatz zu ersetzen ist. Die Standardisierungsinitiative Autosar verfolgt diesen Weg konsequent weiter. Ziel der Initiative ist die Trennung der kundenspezifischen Applikation von Basisfunktionen, die sich durch standardisierte Schnittstellen verbinden lassen. Dazu wurde die Software in ein Schichtenmodell unterteilt, und es wurden einheitliche Schnittstellen definiert. In Verbindung mit einer ebenfalls standardisierten Konfigurationsbeschreibung lassen sich damit in Zukunft Funktionen unabhängig von der darunter liegenden Hardware-Umgebung entwickeln, testen und optimieren.

Der Wechsel von kundenspezifischer Software hin zu standardisierter Basissoftware und Applikationssoftware mit einheitlichen Schnittstellen verspricht allen am Entwicklungsprozess beteiligten Partnern enorme Kosten- und Qualitätsvorteile. Die treibenden Faktoren sind eine leichtere Integrierbarkeit und Transferierbarkeit von Funktionen über Fahrzeug- und OEM-Grenzen hinweg, sowie mehr Flexibilität bei Wartung, Skalierbarkeit des Funktionsumfangs und die Hardware-Unabhängigkeit der Software. Als angenehmer Nebeneffekt ergibt sich eine steigende Zuverlässigkeit des Elektroniksystems.

Die Validität erhöht sich auch durch den von Autosar vorgegebenen Entwicklungsprozess. Er beginnt mit einer modellbasierten Funktionsbeschreibung, wird durchgängig toolbasiert weitergeführt und beinhaltet schließlich auch automatisch generierte und reproduzierbare Tests.

Im Mai 2005 wurde das Release 1.0 veröffentlicht, in dem bereits 31 Basissoftware-Module (BSW) spezifiziert waren. Die Funktionsfähigkeit der Module wurde in einer anschließenden Implementierungs- und Validierungsphase (Validator 1) überprüft und bestätigt. Im Mai 2006 waren bereits 42 BSW-Spezifikationen veröffentlicht, darunter auch die RTE (Runtime Environment). Alle Spezifikationen haben erfolgreich eine Testphase durchlaufen (Validator 2). Ende 2006 wurde das erste vollständige Release (R 2.1) vorgestellt, das auch die Methodikspezifikation beinhaltet.

Vector beteiligt sich als einer der ersten Premium-Partner der Autosar-Initiative aktiv an der Standardisierung und bietet sowohl Basissoftware als auch Werkzeuge für Autosar an.

## 2 Autosar-Architektur

Um die Ziele von Autosar – Trennung der Applikationssoftware von Basismodulen und -funktionen – zu verwirklichen, wird die Fahrzeugelektronik abstrahiert und in mehrere Schichten (Layer) unterteilt, **Bild 1**. Die Verbindung zum tatsächlichen Mikrocontroller und damit die physikalische Basis stellt der „Microcontroller Abstraction Layer“ dar, der die Funktionen und Peripherie des Mikrocontrollers abbildet. Er definiert Schnittstellen für Speicher, I/O-Treiber und dessen Kommunikationsverbindungen. In dieser Schicht können auch die Features softwaremäßig nachgebildet werden, die der Mikrocontroller nicht bietet. Als zweite Schicht liegt die ECU-Abstraktion darüber. Sie bestimmt die ECU-eigene Hardware-Auslegung und stellt beispielsweise Treiber zu ECU-externer Peri-

pherie zur Verfügung. In einer weiteren Schicht, dem Services Layer, werden verschiedene Basisdienste wie Netzwerkdienste, Speicherverwaltung, Buskommunikationsdienste und das Betriebssystem abgebildet. Diese Schicht ist von der Hardware bereits weitgehend unabhängig. Die RTE stellt den vierten Layer dar. Hier findet die eigentliche Trennung von Anwendungs- und Basissoftware (BSW) statt. Die RTE sorgt für die Integration der Anwendungssoftware und regelt den Datenaustausch zwischen Applikation und BSW. Erst durch diese Schicht wird die Wiederverwendung einmal geschriebener Applikationssoftware

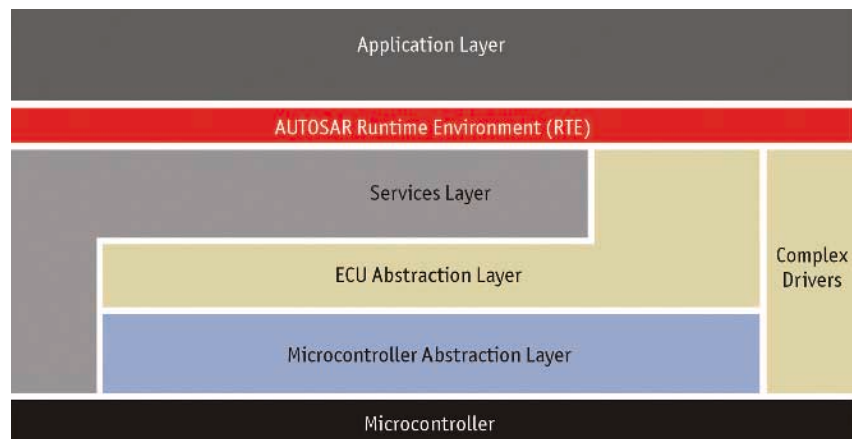
## Die Autoren



Dipl.-Ing. (FH)  
**Matthias Wernicke**  
ist für das Produktmanagement der DaVinci Tool Suite verantwortlich und ist auch in der Autosar-Standardisierungsarbeit aktiv.

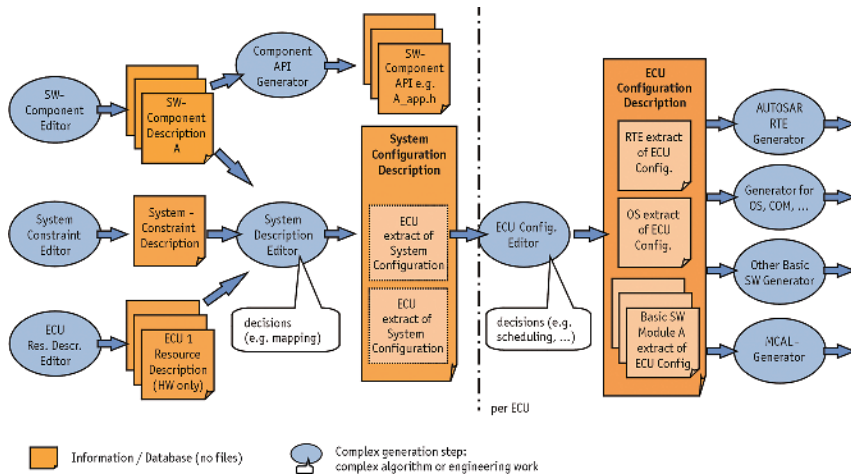


Dipl.-Ing. (FH)  
**Jochen Rein**  
ist Teamleiter in der Abteilung „Embedded Software Components“ und verantwortlich für den Bereich Produktmanagement Autosar BSW und CANbedded.



**Bild 1:** Autosar-Schichtenmodell der Steuergerätesoftware

**Figure 1:** Autosar layer model of the ECU software



**Bild 2:** Strukturbeschreibung der Softwarekomponenten. Die Erstellung von Autosar-konformen Softwarekomponenten ist in klar vorgegebene Entwicklungsschritte unterteilt

**Figure 2:** Structure description of software components. Programming Autosar-compliant software components is divided into clearly defined development steps

möglich, denn die definierten Schnittstellen ermöglichen es, dass Applikationssoftware ohne besondere Kenntnis der später verwendeten Hardware entwickelt und beliebig auf andere Autosar-konforme Steuergeräte transferiert werden kann.

Die Basis für die Konfiguration der Schichten bildet der so genannte Virtual Functional Bus (VFB). Über ihn kommunizieren in einer abstrahierten Sicht alle Komponenten der Fahrzeugelektronik. Die Softwarekomponenten verwenden dafür Ports, deren Schnittstelle über das Port Interface definiert ist. Konnektoren verbinden die Ports. Ob es sich dabei um eine Verbindung innerhalb eines Steuergeräts oder um eine Verbindung über einen externen Bus handelt, spielt für den VFB keine Rolle. Dies entscheidet sich erst, wenn die endgültige Systemauslegung erfolgt und Funktionen einem bestimmten Steuergerät zugewiesen werden.

Die Softwarekomponente selbst benötigt kein Wissen über diese spätere Aufteilung und kann so unabhängig entwickelt werden. Sie ist in Ausführungseinheiten aufgeteilt, so genannte Runnable Entities, die wie Prozeduren beim Eintreten eines bestimmten Ereignisses ausgeführt werden. Ein solches Ereignis ist beispielsweise das Eintreffen eines neuen Sensorwerts oder eine periodische Aktivierung. Durch die aus Sicht des VFB formulierte Beschreibung des Elektroniksystems sind die Schnittstellen der jeweiligen Softwarekomponenten definiert. Die Applikationskomponenten können damit unabhängig vom konkreten Steuergerät entwickelt werden.

Die „Gegenstelle“ auf dem Steuergerät ist die RTE. Sie gewährleistet den Zugriff

auf I/Os, Speicher und andere Basisdienste. Aufgrund der modellbasierten Beschreibung kann die RTE steuergerätespezifisch generiert werden und lässt sich damit ressourcenschonend an unterschiedliche Anforderungen anpassen.

### 3 Methodik

Nachdem Phase 1 abgeschlossen ist, hat Autosar mit dem Start der Phase 2 wie geplant begonnen, auch die Methodik des Entwicklungsprozesses in einem Standard zu formulieren. Die Einhaltung eines strukturierten Entstehungsprozesses ist anerkanntermaßen eine wichtige Voraussetzung für das Erstellen fehlerfreier Software. Mängel in der Anforderungsliste werden frühzeitig erkannt, die Wiederverwendung und Portierung von Softwarekomponenten vereinfacht und das System insgesamt zuverlässiger. Die Methodik gestattet dennoch Freiheiten: So bleibt es beispielsweise den Anwendern überlassen, ob ein Top-Down-Ansatz oder eine Bottom-Up-Entwicklung gewählt wird.

Die Autosar-Initiative sieht für den Software-Entwicklungsprozess eine durchgängige Unterstützung durch Werkzeuge vor. Mittels ausgereifter Tools werden Anforderungen strukturiert umgesetzt und konsistent weitergeführt, Konfigurationen systematisch erstellt und die komplexen Abhängigkeiten automatisch berücksichtigt.

Zunächst erfolgt die formale Beschreibung der drei Hauptbestandteile Software (SW Component), ECUs (ECU Resource) und System-Randbedingungen (System Constraints).

Durch geeignete Editoren entsteht daraus eine Konfigurationsbeschreibung des kompletten Systems, **Bild 2**. Diese System Configuration dient als Basis zur Erstellung der Steuergeräte-Konfigurationen (ECU Configuration), die vom Anwender mithilfe der Konfigurationswerkzeuge für die einzelnen Basissoftwaremodule erstellt werden. Mehrere Generatoren liefern am Ende des Prozesses die steuergerätespezifische Implementierung von RTE und Basissoftware. Alle im Entwicklungsprozess entstandenen Entwurfs- und Konfigurationsdaten werden in einem einheitlichen Format beschrieben. Autosar hat dafür ein Format auf XML-Basis definiert. Es garantiert einerseits die Durchgängigkeit des Entwicklungsprozesses und erleichtert andererseits auch das nahtlose Einfügen notwendiger oder zusätzlicher Tools.

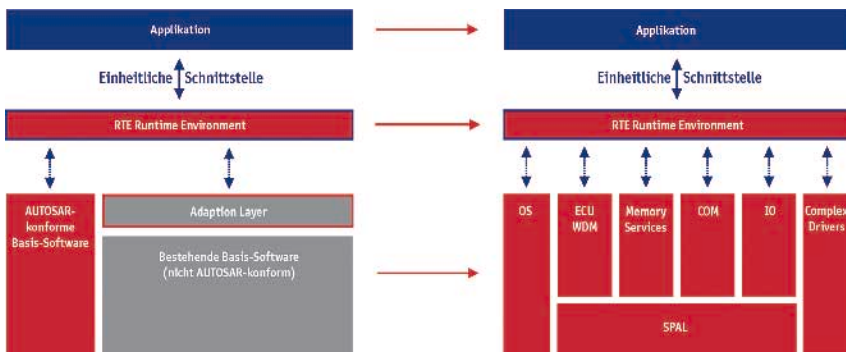
Bereits 2003 stellte Vector Informatik eine Werkzeugfamilie vor, die einen strukturierten und modularen Entwicklungsansatz zugrunde legt und die Anwender beim Entwurf verteilter Funktionen, bei der Zuordnung von Softwarekomponenten und bei der Code-Integration der vernetzten Steuergeräte unterstützt. Heute verwirklicht die DaVinci Tool Suite die Autosar-Idee und arbeitet konform mit der Spezifikation. Vector unterstützt damit als einer der ersten Tool-Anbieter überhaupt den nun vorgegebenen durchgängigen und konsistenten Autosar-Entwicklungsprozess.

Mithilfe der DaVinci Tool Suite werden alle Designschritte systematisch abgearbeitet und dabei die komplexen Zusammenhänge kontrolliert. Die Tool-Unterstützung beginnt beim Entwurf der Softwarearchitektur und reicht von der Netzwerkkonfiguration bis zur Codegenerierung. Ein gut funktionierendes Daten- und Konfigurationsmanagement ist für den Entwicklungsprozess eines komplexen Systems unerlässlich [1]. Die Werkzeugumgebung eASEE wacht darüber im Hintergrund ebenso wie über die Datenintegration oder die Benutzerverwaltung.

Der Entwicklungsprozess beginnt nach dem Festlegen der Requirements mit der formalen Beschreibung der Systemtopologie und der Softwarekomponenten. Dies geschieht mit dem DaVinci System Architect. Er weist Softwarekomponenten den einzelnen ECUs zu und legt damit fest, welche Kommunikation lokal und welche über ein Bussystem erfolgt. Anschließend werden die Schnittstellendaten der Softwarekomponenten den Bussignalen zugeordnet.

Der DaVinci Network Designer definiert das Layout und das Zeitverhalten (Scheduling) der Netzwerkbotschaften. Hierbei





**Bild 5:** Schnittstellen zur Anwendung – wer frühzeitig eine einheitliche Schnittstelle zur RTE schafft, ist von zukünftigen Migrationsschritten befreit

**Figure 5:** Interfaces to the application – creating standard interfaces to the RTE early, prevents from future migration steps

Schon heute können OEMs und Zulieferer mit einer Migration von bereits entwickelten Softwarefunktionen zu Autosar beginnen. Mit der Freigabe des Release 2.0 sind die Weichen für eine solche Umwandlung gestellt. Verschiedene Herangehensweisen sind hierbei denkbar. Eine modularisierte Basissoftware ist heute bereits Grundlage von Vectors CANbedded-Softwarekomponenten. Vector Kunden können also mit geringem Aufwand einzelne Module gegen Autosar-Komponenten tauschen. Dies gilt zum Beispiel für das Transportprotokoll, die Busanbindung und die Diagnose.

Das in der Automobilindustrie weit verbreitete Netzwerkmanagement (NM) von OSEK kann durch das Autosar-NM abgelöst werden. Allerdings ist zu beachten, dass bei einem gemischten Verbund von Steuergeräten mit Autosar-NM und OSEK-NM keine automatische Synchronisation erfolgt. Ein Steuergerät, das beide Netzwerkmanage-

ment-Varianten beinhaltet, könnte diese Aufgabe übernehmen.

Neu geschriebene Anwendungssoftware wird in einer Übergangsphase sowohl bisherige als auch neue Standardkomponenten unterstützen. Ein gewisser Overhead muss daher in Kauf genommen werden. Der große Vorteil ist aber die universelle Wiederverwendbarkeit in verschiedensten Projekten, denn die OEMs werden nicht notwendigerweise ähnliche Migrationsstrategien anstreben. Mischformen sind denkbar: Während ein OEM lediglich auf dem Bus eine Kompatibilität zu Autosar gewährleisten möchte, ansonsten aber möglichst wenig Änderung akzeptiert, könnte ein anderer OEM Komponenten mit sehr viel Interaktion auf dem Bus im ersten Schritt beibehalten, um die Kompatibilität zu bestehenden Steuergeräten zu erhalten.

Für Zulieferer bietet sich in jedem Fall ein frühzeitiger Umstieg auf die RTE an. Die Anwendung wird dadurch unabhängig

von den darunter liegenden Schichten und kann losgelöst von Hardware-Anforderungen entwickelt werden, **Bild 5**. Es soll aber nicht unterschlagen werden, dass dies mit einem Mehraufwand für die RTE verbunden ist. Diese muss nämlich jetzt die verschiedenen darunter liegenden Varianten unterstützen, sofern die Varianten selbst nicht bereits ein Autosar-konformes API zur RTE zur Verfügung stellen. Außerdem müssen die heute verwendeten Standardformate für Konfigurationsdaten wie DBC, LDF oder FIBEX in einheitliche Autosar-XML-Formate überführt werden.

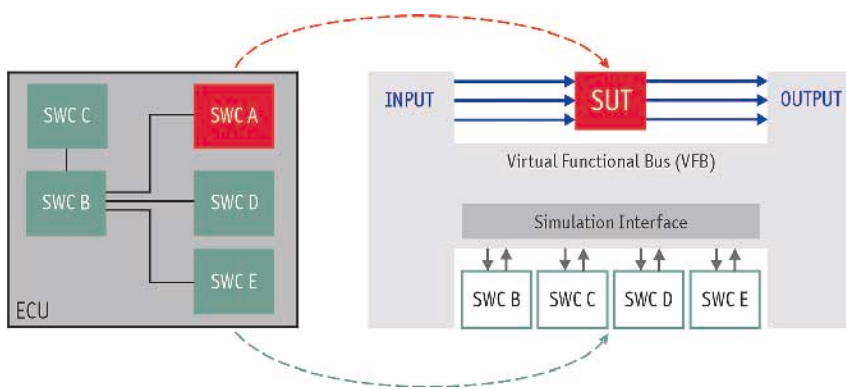
Der Umstieg bietet sich aber an, denn Zulieferer können sich unabhängig vom Fahrzeughersteller dafür entscheiden und auf ein standardisiertes API aufbauen, noch bevor der OEM dies fordert. Damit sind sie frühzeitig von weiteren Migrationsschritten befreit.

### 5 Nächste Schritte: IDE als Basis

Die Vernetzung der Komponenten in der Automobilelektronik muss mit einer engen Verzahnung der Tools einhergehen. Nur durch nahtlos ineinander greifende Werkzeuge ist der Entwicklungsprozess konsistent und die Komplexität beherrschbar. Mit der Integration der Toolkette in eine branchenweit akzeptierte Entwicklungsumgebung IDE (Integrated Development Environment) wie Eclipse oder Visual Studio kommen zusätzliche Vorteile zum Tragen. Erweiterungen ermöglichen die Codierung, das Debugging und den Test von Komponenten innerhalb der gleichen Umgebung, die gleichzeitig als Ausführungs- und als Simulationsplattform fungiert. Dadurch können ausführbare Softwarefunktionen real getestet werden, während zusätzliche Funktionen einfach als Modelle über ein Simulationsinterface ergänzt werden, **Bild 6**.

So ist kein zusätzlicher Aufwand für den Entwurf weiterer Autosar-Komponenten erforderlich, denn eine detaillierte Implementierung ist für den Funktionstest nicht notwendig. Einfache Testszenarien können schnell erstellt werden. Durch unterschiedliche Ausführungszeiten werden verschiedene Testschwerpunkte gelegt. Die schnelle Ausführung von Testfällen sorgt für hohe Wiederholraten und ermöglicht eine Vielzahl automatisierter Tests, eine langsame Ausführung dient hingegen dem komfortablen Debuggen der Software.

Mögliche Ergänzungen einer solchen IDE sind außerdem Testfallgeneratoren sowie statische Code- oder Coverage-Analysen.



**Bild 6:** Test einer Softwarekomponente in der Ausführungs- und Simulationsplattform – während eine ausführbare Softwarefunktion real getestet wird, können zusätzliche Funktionen einfach als Modelle über das Simulationsinterface ergänzt werden

**Figure 6:** Test of the software component in the execution and simulation platform – while testing an executable software function in real, additional functions can easily be completed as models via the simulation interface

Je besser solche Add-Ons an die Entwicklungsplattform gekoppelt werden können, desto einfacher und kostengünstiger sind Tests und umso zuverlässiger arbeitet später das Produkt.

## 6 Zusammenfassung

Die Komplexität der Fahrzeugelektronik nimmt weiter zu. Die Definition von Schnittstellen durch Autosar und die Beschreibung des Systems durch Abstraktionsebenen reduziert jedoch keinesfalls die Menge der Anforderungen und Abhängigkeiten der Funktionen untereinander. Sie zu beherrschen, erfordert leistungsfähige Auto-Entwicklungs-Tools.

Die Trennung von Basissoftware und Applikation schafft die Voraussetzung für eine Wiederverwendung von einmal geschriebenem Funktionscode auf verschiedenen Hardwareplattformen. Dies führt nicht nur zu einer qualitativen Steigerung des Systems durch vielfach erprobte Produkte, sondern ermöglicht auch die Konzentration der Entwicklungskraft auf Innovationen und neue Herausforderungen. Der durch die Spezifikation vorgegebene Entwicklungsprozess unterstützt eine strukturierte Herangehensweise und zeigt frühzeitig mögliche Fehlerquellen auf.

Vector Informatik hat die Idee von Autosar konsequent umgesetzt und unterstützt den Entwicklungsprozess in allen Phasen. Dies beginnt beim strukturierten Entwurf der Autosar-Softwarekomponenten und deren Verteilung auf Steuergeräte, über die Definition der Kommunikation bis hin zur Konfiguration der Basissoftware. OEMs wie Zulieferer können sich diesen Vorteil bereits heute in Autosar-Projekten zu Nutze machen. Auch bei den anfangs vorherrschenden Mischformen von herkömmlichen Komponenten und neuer Autosar-Technologie innerhalb eines Systems ist Vector der geeignete Umsetzungspartner. Mit Know-how zur gesamten Autosar-Software, Integrationserfahrung und geeigneten Tools zur Einbindung von Drittkomponenten hilft Vector bei allen Schritten in Richtung Autosar-Steuergerät.

For an English  
version of this  
article, see

**ATZ**  
**elektronik**  
WORLDWIDE



For information on subscriptions, just call us or send an E-Mail or fax:  
Vieweg Verlag Postfach 1546 D-65173 Wiesbaden  
Tel. +49 5241 80-1968 | E-mail: vieweg@abo-service.info