

Neue Wege zur Steuergeräte-Software

Iterative Zusammenarbeit von OEM, Zulieferer und Software-Lieferant – Teil 1

Ein Hauptgrund für die Einführung von AUTOSAR ist neben der Standardisierung der Basis-Software auch eine höhere Wiederverwendbarkeit der Funktions-Software. Dies hat allerdings auch Einfluss auf die Zusammenarbeit von OEM, Zulieferer, Halbleiterhersteller und Software-Lieferant. Dieser erste Teil der zweiteiligen Serie erläutert die Grundlage für die erfolgreiche Zusammenarbeit: die AUTOSAR-spezifischen Austauschformate und Werkzeuge.

Von Pascale Morizur, Matthias Wernicke und Justus Maier

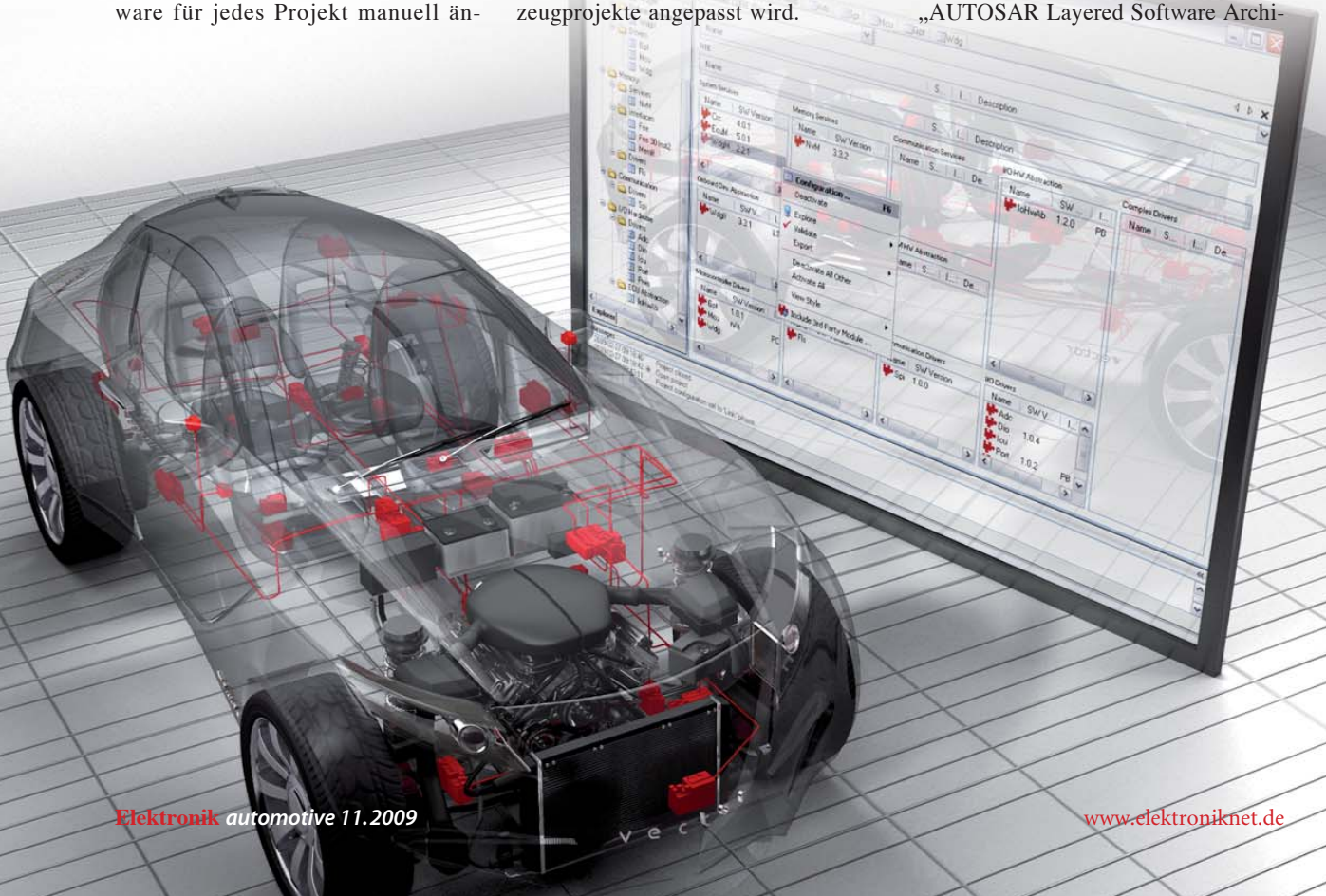
Jeder OEM hat für die Steuergeräte seiner Fahrzeuge eigene Anforderungen hinsichtlich Funktionen, Kommunikation und Diagnose. Diese Anforderungen sind in OEM-spezifischer Software umgesetzt. Bedient ein Zulieferer mehrere OEMs, muss er die Steuergeräte-Software für jedes Projekt manuell ändern.

Selbst wenn die Funktions-Software von der Software zur Anpassung an die OEM-spezifischen Anforderungen entkoppelt ist, ist dieser Vorgang doch arbeitsintensiv und fehleranfällig. Bild 1 zeigt, wie eine unveränderte Funktions-Software ohne AUTOSAR an verschiedene Fahrzeugprojekte angepasst wird.

Ein Ziel von AUTOSAR ist das Minimieren dieses Anpassungsaufwands bei der Software-Integration. Dafür sind in AUTOSAR die konsequente Abstraktion der Software von der Hardware sowie die Aufteilung der Software in Module mit definiertem Funktionsumfang und genauen Schnittstellen festgelegt. Diese Module können kombiniert und vor allem weitgehend konfiguriert werden, um die Anforderungen verschiedener OEMs abzudecken. Die manuelle Anpassung der Software entfällt und die Wiederverwendung wird erhöht. OEM-spezifische Software-Teile, z.B. für die Diagnose, sind durch die definierten Schnittstellen mit geringem Aufwand austauschbar.

Die AUTOSAR-Referenzarchitektur

Die Beschreibung der AUTOSAR-Referenzarchitektur ist in dem Dokument „AUTOSAR Layered Software Archi-



tektur“ [1] enthalten. Darin ist die Steuergeräte-Software in drei Teile gegliedert (Bild 2):

- ▶ Die Funktions-Software besteht aus Software-Komponenten (SWCs). Diese werden auf Basis eines Virtual Functional Bus (VFB) unabhängig von den Steuergeräten erstellt und können über Schnittstellen miteinander kommunizieren.
- ▶ Die Run-time Environment (RTE) dient als Laufzeitumgebung für die SWCs und bildet die technische Realisierung des VFB auf einem konkreten Steuergerät.
- ▶ Die Basis-Software-Module (BSW) übernehmen die Grundfunktionen des Steuergeräts. Diese enthalten auch höherwertige Standarddienste, z.B. die Verwaltung der ECU-Zustände oder Diagnosedienste.

Die RTE ist die Schicht zwischen der Funktions-Software und den Basis-Software-Modulen. Sie stellt alle von den SWCs benötigten Schnittstellen bereit, um auf Daten und Dienste der BSW-Module zugreifen zu können. Beispiele hierfür sind Signalwerte aus dem Kommunikationsnetzwerk (CAN, LIN, FlexRay), I/O-Signale oder Standarddienste der BSW-Module. Die Schnittstellen ergeben sich aus den SWC-Description-Dateien. Außerdem sorgt die RTE mithilfe des Betriebssystems für die Ausführung der SWCs und für die Kommunikation der SWCs untereinander.

Die BSW-Module gliedern sich gemäß der AUTOSAR-Architektur [1] in drei Schichten:

- ▶ Service Layer.
- ▶ ECU-Abstraction Layer.
- ▶ Microcontroller Abstraction Layer (MCAL).

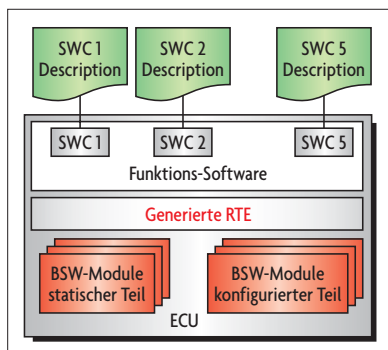


Bild 2. AUTOSAR-Architektur eines Steuergeräts.

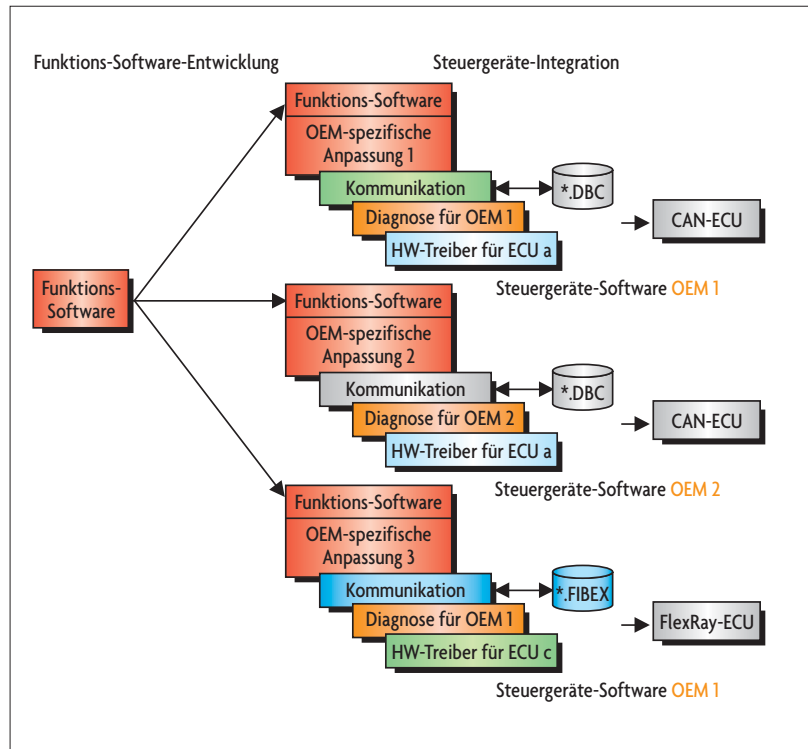


Bild 1. Entstehung von Steuergeräte-Software ohne AUTOSAR.

Dabei spielen die BSW-Module des Service Layers eine besondere Rolle, denn sie enthalten Standarddienste für die Funktions-Software, die über spezielle Schnittstellen innerhalb der RTE abgerufen werden können. Die Konfiguration dieser Dienste wird im zweiten Teil des Artikels näher beschrieben.

AUTOSAR Release 3.0 definiert etwa 50 unterschiedliche, konfigurierbare und teilweise sehr komplexe Basis-Software-Module. Die Mehrzahl von ihnen enthält Funktionen, die bereits in den bisherigen Software-Architekturen üblich waren, jetzt aber genauer gegeneinander abgegrenzt sind. Diese konsequente Aufteilung der Funktionen in einzelne Software-Module garantiert die gewünschte Hardware-Abstraktion und die Skalierbarkeit für unterschiedliche Arten von Steuergeräten. Solche Standard-Module erhöhen die Qualität der Steuergeräte-Software. Die Standardisierung umfasst in den meisten Fällen sowohl die Schnittstellen als auch die Funktion der BSW-Module. Eine Ausnahme stellen die BSW-Module für die Diagnose dar. Da der Diagnoseablauf sehr stark von den Fertigungs- und After-Sales-Prozessen des OEMs abhängt, sind in AUTOSAR nur die

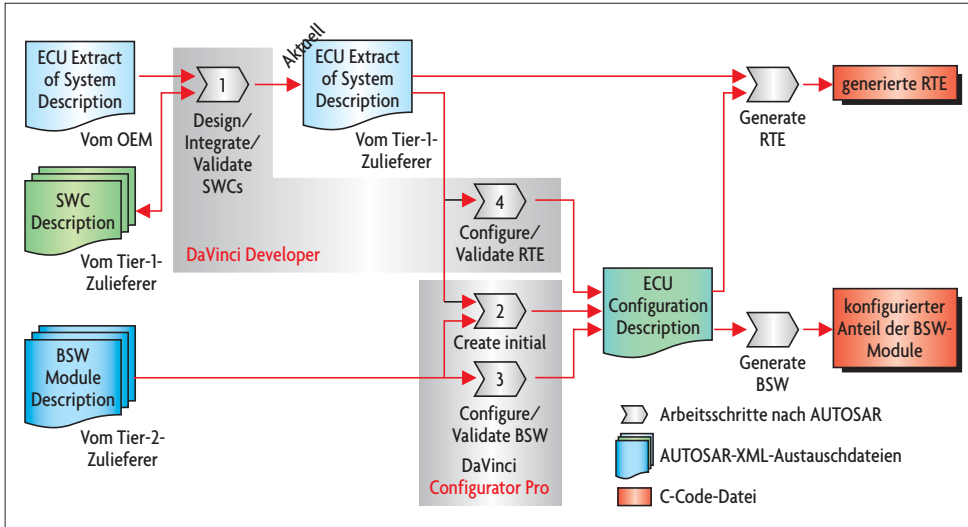


Bild 3. Werkzeuggestützte Integration der SWCs und Konfiguration der RTE sowie der BSW-Module nach der AUTOSAR-Methodik.

Schnittstellen der Diagnosemodule definiert. Als Folge gibt es OEM-spezifische Implementierungen der Diagnosemodule. Diese werden von Vector für viele OEMs bereitgestellt; es verbleibt für den Zulieferer die Aufgabe, die jeweilige Variante zu konfigurieren und zu integrieren.

Sowohl die BSW-Module als auch die RTE sind als Software-Produkte unterschiedlicher Software-Lieferanten (Tier-2) verfügbar, wie die Microsar-Produkte von Vector, die alle BSW-Module und die RTE gemäß AUTOSAR Release 3.0 abdecken. Obwohl es sich um Standard-Software-Produkte handelt, müssen die BSW-Module und die RTE an die projektspezifischen Randbedingungen angepasst werden, z.B. OEM, Fahrzeugreihe und Steuergerätevariante. Dies geschieht während der Konfiguration über geeignete PC-basierte Werkzeuge. So können die RTE mit dem DaVinci Developer und die BSW-Module mit dem DaVinci Configurator Pro von Vector konfiguriert werden.

AUTOSAR definiert Datenaustausch zwischen Entwicklungspartnern

Die Methode zur Entwicklung von Steuergeräte-Software wurde von der AUTOSAR-Organisation in der AUTOSAR-Methodik [2] definiert. Sie unterteilt den Entwicklungsprozess in drei Aktionen und standardisiert den Datenaustausch zwischen den Ent-

wicklungspartnern mit einem Satz von XML-Dateien:

SWC Description ▶ Komponenten-Implementierung: Der Zulieferer oder der OEM definiert die SWCs. Hierzu erstellt er für jede SWC eine XML-Datei, die SWC Description. Diese beschreibt die Schnittstellen und den Ressourcen-Bedarf der SWC. Im Anschluss erstellt er dazu passend die C-Dateien für die Implementierung.

System Description ▶ Systemkonfiguration: Der OEM definiert auf Basis der SWCs zuerst den Funktionsumfang für das gesamte Fahrzeug unabhängig von den Steuergeräten. Als nächstes entwirft er die Kommunikationsnetzwerke und verteilt die SWCs auf die vorhandenen Steuergeräte. Das Ergebnis wird in der System Description gespeichert.

ECU Extract of System Description ▶ Für jedes Steuergerät reduziert der OEM die System Description zu einem „ECU Extract of System Description“, das der OEM an die Zulieferer des entsprechenden Steuergeräts weitergeben kann. Diese Datei ersetzt zur Konfiguration der BSW-Module die bislang verwendeten .dbc-, FIBEX- oder .ldf-Dateien.

Aktuell ECU Extract of System Description ▶ ECU-Entwicklung und -Konfiguration: Ausgehend vom „ECU Extract of System Description“ integriert der Zulieferer seine eigenen SWCs. Das Ergebnis ist ein vollständiger und aktueller „ECU Ex-

tract of System Description“, der nun die Beschreibung aller SWCs –sowohl vom OEM als auch vom Zulieferer – eines Steuergeräts enthält.

BSW Module Description Eine weitere Voraussetzung für die Steuergerätekonfiguration sind die „BSW Module Description“-Dateien, die die Definition der Datenstrukturen und aller konfigurierbarer Parameter eines BSW-Moduls enthalten. Diese Dateien sind implementierungsspezifisch und neben den Generatoren und dem statischen Code Bestandteil der BSW-Module.

ECU Configuration Description Im Anschluss erstellt der Zulieferer die initiale „ECU Configuration Description“ (Aktion 2 in

Bild 3) basierend auf dem aktuellen „ECU Extract of System Description“ und den „BSW Module Description“-Dateien. Danach beginnt er mit der Konfiguration des Steuergeräts und dokumentiert sie in der „ECU Configuration Description“. Hierzu verwendet er geeignete Werkzeuge, um die Parameter der BSW-Module und der RTE einzustellen und zu prüfen (Aktionen 3 und 4 in Bild 3). Die „ECU Configuration Description“ ist die Grundlage für die steuergerätespezifische Generierung der RTE und der BSW-Module durch die zugehörigen Generatoren.

Die AUTOSAR-Methode ist flexibel und für die Praxisanforderungen unterschiedlicher Projekte oder unterschiedlicher OEMs geeignet. So ist z.B. die Verwendung von SWCs in der System Description optional. Bild 3 zeigt am Beispiel der Werkzeuge DaVinci Developer und DaVinci Configurator Pro, wie die ECU-Entwicklung und -Konfiguration werkzeuggestützt umgesetzt werden kann.

Konfiguration und Integration aller Software-Teile

Während des in AUTOSAR definierten Konfigurationsprozesses wählt der Zulieferer aus seiner Komponentensammlung die SWCs aus, die er für die Funktion des Steuergeräts benötigt. Er integriert sie, zusammen mit den BSW-Modulen und der RTE, in sein Steuergerät. Dadurch verlagert sich die

Hauptarbeit bei der Integration der Steuergeräte-Software von der manuellen Anpassung des Code auf die werkzeuggestützte Konfiguration der BSW-Module und der RTE.

Da der aktuelle Stand der AUTOSAR-Spezifikationen noch einige Interpretationsfreiräume enthält, ist es ratsam, entweder das gesamte BSW-Paket oder zumindest definierte BSW-Blöcke aus einer Hand zu beziehen. Der Vorteil ist, dass der Software-Lieferant diese Module bereits einem Integrationstest unterziehen kann. Möglich ist aber auch der Erwerb einzelner Module verschiedener Tier-2-Zulieferer sowie der Einsatz abgewandelter BSW-Module. Allerdings erhöht sich dabei der Integrationsaufwand, da sowohl die Gesamtfunktionalität sichergestellt als auch die Integration in die Konfigurationswerkzeuge durchgeführt werden müssen.

Grundsätzlich werden alle Microsar-BSW-Module systematischen Integrationstests unterzogen. Darüber hinaus kann der Integrationsumfang wahlweise auf Software-Module von Drittherstellern wie beispielsweise MCAL-Treiber erweitert werden.

Für die Konfiguration der BSW-Module benötigt der Zulieferer ein universelles Werkzeug, das ihn mit komfortablen Funktionen unterstützt. Aus diesem Grund hat Vector den DaVinci Configurator Pro neu entwickelt.

Er unterstützt drei Anwendungsfälle:

- ▶ Konfiguration der Microsar-BSW-Module von Vector.
- ▶ Konfiguration der AUTOSAR-BSW-Module von Drittherstellern.
- ▶ Konfiguration selbst erstellter Software-Module.

Die Konfiguration der Microsar-BSW-Module erfolgt über eine grafische Oberfläche, mit der die komplexen Zusammenhänge der Konfigurationsparameter vereinfacht dargestellt werden. Weiterhin ist die Auswahl der Parameter auf gültige Eingabewerte beschränkt und das Einstellen von unplausiblen Werten wird verhindert.

Für die Konfiguration der BSW-Module von Drittherstellern ist der in AUTOSAR definierte „Generic Configuration Editor“ (GCE) im DaVinci Configurator Pro enthalten. Alternativ kann der Zulieferer für diese Module auch eine komfortable und integrierte Konfigurationsoberfläche entwickeln. Das erfolgt mit dem DaVinci Configurator Kit. Damit werden für die Software-Module die „BSW Module Description“-Dateien erstellt, Bedienoberflächen definiert, Validierungsregeln festgelegt und Code-Generatoren für die Erzeugung von ausführbaren Code-Teilen erstellt. Dieses Verfahren kann der Zulieferer auch zur

Konfiguration von eigenen BSW-Modulen nutzen.

Als Ergänzung zur AUTOSAR-Methode enthalten sowohl der DaVinci Configurator Pro als auch der DaVinci Developer Validierungsregeln. Sie stellen sicher, dass einzelne Parameter sowie komplexe Parametergruppen und deren Abhängigkeiten untereinander validiert werden und dass die „ECU Configuration Description“ konsistent generiert wird. Diese Konsistenz ist zwingend notwendig für die anschließende Generierung der RTE sowie der BSW-Module.

Im zweiten Teil dieses Artikels wird anhand ausgewählter Anwendungsfälle gezeigt, wie die Austauschdateien und Konfigurationswerkzeuge in der Praxis eingesetzt werden. Das Erstellen einer kompletten AUTOSAR-konformen Steuergeräte-Software für einen OEM wird erläutert und dargestellt, wie sich die Software pflegen und an die Anforderungen anderer OEMs anpassen lässt. *sj*

Literatur

- [1] Layered Software Architecture.
www.autosar.org/download/specs_aktuell/AUTOSAR_LayeredSoftwareArchitecture.pdf
- [2] AUTOSAR Methodology.
www.autosar.org/download/specs_aktuell/AUTOSAR_Methodology.pdf



**Dipl.-Ing.
Pascale Morizur**

studierte Physik-Elektronik an der Grande Ecole ICPI in Lyon (F). Nach ihrem Abschluss 1986 arbeitete sie zehn Jahre bei MAN-Nutzfahrzeuge in der Vorentwicklung im Bereich CAN, J1939 und Diagnose. Seit 2005 ist sie bei Vector als Produkt-Manager im Bereich Embedded-Software-Komponenten tätig.
pascale.morizur@vector.com



**Dipl.-Ing.(FH)
Matthias Wernicke**

war nach Abschluss seines Studiums der Industrieelektronik an der FH Ulm zunächst vier Jahre im Daimler Forschungszentrum in Ulm tätig. Seit Anfang 2000 arbeitet er bei Vector Informatik in Stuttgart an der Entwicklung von Methoden und Werkzeugen für den Entwurf verteilter Elektronikfunktionen im Kfz. Er ist heute für das Produkt-Management der DaVinci-AUTOSAR-Werkzeuge verantwortlich.
matthias.wernicke@vector.com



**Dipl.-Inf. (FH)
Justus Maier**

studierte Informatik in Regensburg. Er begann seine berufliche Laufbahn als Entwickler für Standard-Software in der Versicherungsbranche. Seit acht Jahren beschäftigt er sich mit der Konzeption und Weiterentwicklung von Werkzeugen für die Steuergerätekonfiguration im AUTOSAR-Umfeld. Als technischer Produkt-Manager des DaVinci Configurator Pro ist er bei Vector seit 2006 tätig.
justus.maier@vector.com