Virtualisierung – Das zentrale Werkzeug für kürzere Entwicklungs- und Releasezyklen von software-basierten Kundenfunktionen in der Automobilindustrie

Strenge, Alexander

Carmeq GmbH
Carnotstr. 4
10587 Berlin
E-Mail:
alexander.strenge@carmeq.com

ABSTRACT

Die digitale Vernetzung des Automobils (Connected Car) verstärkt die bereits eingesetzte Verschiebung der Wertschöpfung in der Automobilindustrie. Weg von der Fokussierung der Hardware-Entwicklung und der Fahrzeugproduktion bis End-of-Production hin zu Software bzw. software-basierten Kundenfunktionen über den gesamten Lebenszyklus des Fahrzeugs. Die Fahrzeughersteller (OEMs) haben auf die Digitalisierung des Fahrzeugs mit einer neuen End-to-End Elektrik-/Elektronik-Architektur (E/E-Architektur) reagiert, welche das Fahrzeug insbesondere für Online Remote Updates und -Upgrades befähigt. Neben der Veränderung der E/E-Architektur müssen für die Ermöglichung von kürzeren Release- und somit Entwicklungszyklen von software-basierten Funktionen neue Entwicklungsmethoden und -prozesse eingeführt werden.

Die Virtualisierung der Entwicklung und Absicherung von software-basierten Kundenfunktionen ist das zentrale Werkzeug, um kürzere Entwicklungszyklen zu realisieren. Die frühzeitige Funktionserlebbarkeit durch virtuelle Funktionsintegration und schneller abgesicherte Funktionshübe durch Software-in-the-Loop-Verfahren können erreicht werden. Weiterhin braucht es ein zyklisches Entwicklungsvorgehen zur Erreichung einer reduzierten Time-to-Market und, um auf auftretende Fehler im Feld, z.B. bei sicherheitskritischen automatisierten Fahrfunktionen, kurzfristig und flexibel reagieren zu können.

Für die virtuelle Entwicklung und Absicherung software-basierter Funktionen soll ein Überblick an Methoden und Prozessen gegeben werden. Die Virtualisierung soll weiterhin in Einklang mit einem zyklischen Entwicklungsvorgehen für kurze Releasezyklen gebracht werden.

SCHLÜSSELWÖRTER

Connected Car, Digitalisierung, Simulation, Virtuelle Entwicklung und Absicherung, Software-basierte Funktionen, Zyklische Entwicklung, E/E-Architektur, Software-in-the-Loop, Digital Twin

AKTUELLE HERAUSFORDERUNGEN: ANNÄHERUNG DER KUNDENBEFÜRFNISSE AN MOBILE DEVICES

Die digitale Vernetzung des Automobils (Connected Car) verstärkt die bereits eingesetzte Verschiebung der Wertschöpfung in der Automobilindustrie. Weg von der Fokussierung der Hardware-Entwicklung und der Fahrzeugproduktion bis End-of-Production hin zu Software bzw. software-basierten Kundenfunktionen über den gesamten Lebenszyklus des Fahrzeugs. Der Megatrend der Digitalisierung wird begleitet vom Eindringen der New Economy aus der Consumer Electronics Branche in die klassische Automobilindustrie. Die neuen Player (z.B. Google (Waymo)) bringen das Kundenbedürfnis nach kürzeren Releasezyklen von Software-Updates und -Upgrades over-the-Air mit. So zeigt sich, dass die Maßstäbe der Welt der Mobile Devices zunehmend auch ihre Gültigkeit für den Automobil-Sektor beanspruchen.

VERÄNDERUNG DER E/E-ARCHITEKTUR FÜR UPDATE- UND UPGRADEFÄHIGKEIT UND DIE NEUE ROLLE DES OEM

Die OEMs stehen vor der großen Herausforderung, hochvernetzte software-basierte Kundenfunktionen sicher auf die Straße zu bringen, eine große Breite an Modellreihen und Ausstattungsvarianten über den gesamten Lebenszyklus der Fahrzeuge effizient darstellen zu können. Dies impliziert auch software-basierte Funktionen durch Software-Updates bzw. -Upgrades über den Lebenszyklus eines Fahrzeugs in kurzen Releasezyklen nachbessern bzw. nachladen zu können.

In technischer Hinsicht haben einige OEMs auf diese Erwartung mit einer neuen E/E-Architektur – in Zusammenarbeit mit AUTOSAR – reagiert. Die bisher üblichen bauteilorientierten Einzelsysteme in der E/E-Architektur (AUTOSAR Classic) werden somit zunehmend durch eine Hochintegration verteilter software-basierter Funktionen auf Hochleistungsrechnern abgelöst (AUTOSAR Adaptive).

Bei AUTOSAR Classic sind die einzelnen Steuergeräte im Fahrzeug statisch in das Gesamtsystem eingebunden. Die Verbindungsbeziehungen wurden also bereits zum Zeitpunkt der Konfiguration festgelegt. Zur Laufzeit können nur bedingt Änderungen vorgenommen werden. Die auf AUTOSAR Adaptive basierenden Anwendungen haben hier einen entscheidenden Vorteil: Sie können zur Laufzeit in das System eingebunden werden und die neue service-orientierte Kommunikation ermöglicht es zudem, Anwendungen zu jedem beliebigen Zeitpunkt in das Gesamtsystem zu integrieren. Dadurch können die Softwarekomponenten unabhängig voneinander entwickelt, getestet und verteilt bzw. aktualisiert werden.

Die neue E/E-Architektur trennt damit einerseits das Fahrzeug funktional noch deutlicher zwischen Hardware und Software, andererseits befähigt es das Fahrzeug bzw. den Kunden zu Online Remote Updates und Upgrades durch die End-to-End-Wirkkette zwischen Fahrzeug und Backend. Weiterhin können dadurch rechenintensive Funktionen aus der Sensor-Aktor-Ebene ins Backend verlagert werden.

Neben der technischen E/E-Architektur ändert sich damit jedoch auch das Zusammenarbeitsmodell zwischen Zulieferer und OEM. So werden Zulieferer in Zukunft nicht zwingend vollständig Hardware-/Software-integrierte Bauteile liefern. Stattdessen wird die Lieferung von Hardware (HW) und Software (SW) getrennt von unterschiedlichen Lieferanten erfolgen. Die Software kann dabei aus OEM-Eigenentwicklung stammen oder auch von Drittanbietern, die ihre Inhalte in die OEM-Ökosysteme einspeisen. In Zukunft müssen die OEMs die E/E-Integration und -Absicherung von SW-/SW- und SW-/HW-Verbünden auf E/E-Systemverbund- und E/E-Gesamtverbundebene bedienen und beherrschen.

VIRTUALISIERUNG DER ENTWICKLUNG UND ABSICHERUNG SOFTWARE-BASIERTER FUNKTIONEN

Neben der neuen technischen E/E-Architektur bedarf es einer erheblichen Veränderung der Entwicklungsmethoden und -prozesse für software-basierte Kundenfunktionen. Eine lineare Produkt- bzw. Funktionsentwicklung ohne konsistente Trennung von Hardware- und Software-Entwicklung ist nicht mehr hinreichend. Während früher Bauteile separat entwickelt und zunächst einzeln auf ihre Funktionalität überprüft und anschließend im Wirkketten- und Fahrzeugverbund getestet wurden, muss heute die Vernetzung der Fahrzeugfunktionen bereits in frühen Entwicklungsphasen vor Hardwareverfügbarkeit als Entwicklungsaufgabe verstanden werden.

Systems Engineering und Funktionsorientierte Entwicklung als Grundlage

Unter „Systems Engineering“ wird kurz umschrieben, die Gesamtheit der Entwicklungsaktivitäten, die notwendig sind, um ein System zu entwickeln, verstanden. Die zentralen Herausforderungen bestehen darin, eine eindeutige und in sich stimmige Spezifikation zu erstellen, alle relevanten Informationen für alle Beteiligten bereitzustellen, die Rückverfolgbarkeit zwischen allen Aktivitäten zu gewährleisten und das System im geeigneten Umfang zu verifizieren und zu validieren. Die Herausforderung muss dabei konsistent vom Gesamtsystem bis zur kleinsten technischen Einheit und über mehrere Technikdomänen (Hardware bzw. Mechanik, E/E, Software) gemeistert werden (siehe Abbildung 1). Weiterhin rückt neben der Produktentstehung die Betrachtung des Systems Engineering über den gesamten Lebenszyklus durch die digitale Vernetzung des Fahrzeugs in den Fokus.

Die „Funktionsorientierte Entwicklung“ kann als Entwicklungsparadigma für vernetzte software-basierte Funktionen innerhalb des Systems Engineering verstanden werden. Es bezeichnet einen – im Gegensatz zur bauteilorientierten Entwicklung stehenden – domänenübergreifenden und durchgängigen Entwicklungsprozess über mehrere technische Ebenen hinweg. Dabei werden von einem Kundenwunsch ausgehend sogenannte kundenerlebbare Funktionen abgeleitet und schrittweise auf die Ebene der technischen Realisierung heruntergebrochen (siehe Abbildung 1).

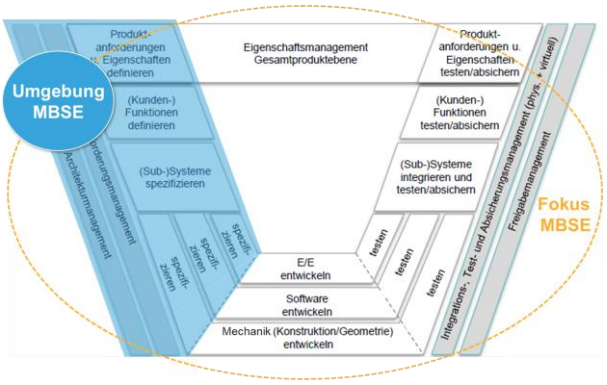

Abbildung 1: V-Modell zum Systems Engineering (mit Fokus Produktentstehung) und Verortung des MBSE

Zur inhaltlichen Darstellung des Entwicklungsvorgehens wird das bekannte V-Modell aus der Software-Entwicklung für die funktionsorientierte Entwicklung eingesetzt und erweitert (siehe Abbildung 1). Der Entwicklungsprozess muss dabei zusätzliche Ebenen durchlaufen: eine Ebene der logischen Funktionsdefinition bzw. Funktionsarchitektur und der technischen Systemdefinition bzw. Systemarchitektur. Damit soll sichergestellt wer-

den, dass die Entwicklung aller mechatronischen Komponenten aufeinander abgestimmt ist und diese als Gesamtsystem funktionieren. Im Fokus der Darstellung stehen hierbei häufig die E/E-Komponenten bzw. die Steuergeräte, auf denen die Software eingebettet ist.

Modellbasiertes Systems Engineering (MBSE)

Modellbasiertes Systems Engineering (MBSE) kann als Virtualisierung der Entwicklung bzw. Spezifikation von software-basierten Kundenfunktionen verstanden werden. Die Entwicklungsumgebung von MBSE lässt sich dementsprechend auf der linken Seite des V-Modells abbilden (siehe Abbildung 1). MBSE kommt die wichtige Aufgabe zu, die unnötige Duplizierung von Informationen und die asynchrone parallele Weiterentwicklung von Daten zwischen den Technikdomänen (Mechanik, E/E, Software) zu vermeiden. Die Entwicklung von Systemen und die Entwicklung von Software findet auf unterschiedlichen Abstraktionsebenen und mit unterschiedlichen Sichtweisen statt. Während sich die Entwicklung auf Systemebene auf die Beschreibung des „Was“ konzentriert, geht es bei der Entwicklung ausführbarer Software um das „Wie“ in Bezug auf dieses „Was“. Beide Aktivitäten können modellbasiert geschehen.

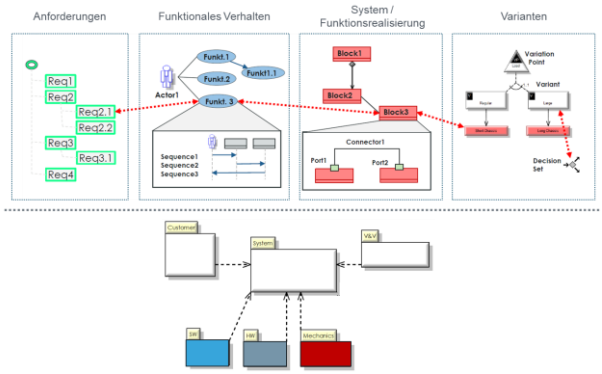

Abbildung 2: Durchgängige (oben) und domänenübergreifende (unten) Modellierung als Ziel des MBSE

Entscheidend ist dabei die Durchgängigkeit der Tools bzw. der Entwicklungsumgebung und eine einheitliche Datenbasis für alle Ebenen (vom Gesamtsystem über Sub-Systeme bis zur kleinsten technischen Einheit) und Technikdomänen (siehe Abbildung 2). Nur so kann die Modellierung bzw. Spezifikation von Funktion, System und Software ineinandergreifen und somit konsistent sein. Hierbei steht insbesondere die Traceability im Fokus. Deswegen ist eine informationsgerechte bzw. spezifische Aufbereitung für verschieden Rollen innerhalb des MBSE essentiell, um eine handhabbare und zielgerichtete Anwendung zu gewährleisten (siehe Abbildung 3).

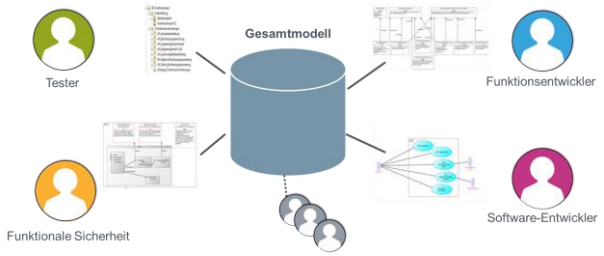

Abbildung 3: Rollen-spezifische Aufbereitung von Informationen innerhalb des MBSE

Bei einer durchgängigen Methodik – und somit auch Toolkette – beim MBSE können bereits in der frühen Entwicklungsphase entwicklungsbegleitend modellbasierte Tests durchgeführt werden. Eine gemeinsame Datenbasis ist essentiell für eine verteilte Entwicklung durch parallele Teams über Unternehmensgrenzen und Technikdomänen hinweg sowie einer agileren Vorgehensweise. Das MBSE – inklusive einer simulationsfähigen Systemspezifikation bzw. -architektur – stellt eine essentielle Eingangsgröße für die virtuelle E/E-Integration und -Absicherung dar.

Virtuelle E/E-Integration und -Absicherung

Die bisherige Abhängigkeit von Erprobungsträgern bzw. HW-Prototypen stellt eine restriktive Beeinflussung der E/E-Integration und -Absicherung von software-basierten Funktionen während des Produktentstehungsprozesses dar. Diese Restriktion wird umso kritischer für eine valide E/E-Integration und -Absicherung von neuen Software-Updates und -Upgrades über den gesamten Lebenszyklus eines Fahrzeugs. Neben der Abhängigkeit zur Hardwareverfügbarkeit verhindert speziell eine geringe Automatisierbarkeit und Skalierbarkeit der bisherigen Vorgehensweise für die E/E-Integration und -Absicherung kurze Entwicklungszyklen. Den genannten Restriktionen kann durch virtuelle Methoden für die E/E-Integration und -Absicherung begegnet werden.

Virtuelle E/E-Integration

Die virtuelle E/E-Integration beschreibt eine Integration vor bzw. ohne Hardwareverfügbarkeit. Hierbei ist zwischen vertikaler (Steuergeräte-Sicht) und horizontaler (Funktionssicht) Integration zu unterscheiden (siehe Abbildung 4). Insbesondere die horizontale Funktionsintegration und dessen Virtualisierung rücken in den Fokus des OEM.

Die vertikale E/E-Integration beschreibt eine Integration von Softwarekomponenten (SWC) auf das HW-Target (Steuergerät bzw. Hochleistungsrechner). Damit wird die SWC mit den Softwarebestandteilen der HW-näheren Schichten (z.B. Laufzeitumgebung (RTE), Basissoftware, Betriebssystem, Micro-Controller Abstraction Layer) integriert. Bei der virtuellen vertikalen Integration werden diese verschiedenen Schichten bzw. Bestandteile simuliert. Die Virtualisierung kann in verschiedenen Detailierungsgraden ablaufen und schließt selbst eine Virtualisierung bzw. Emulation der Hardware nicht aus.

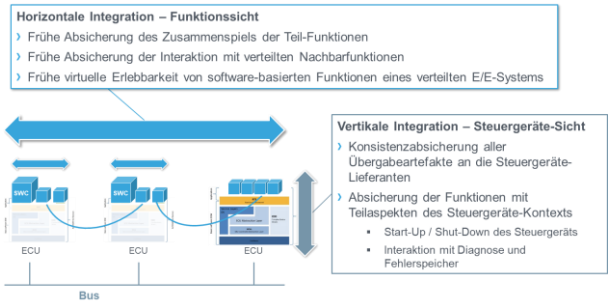

Abbildung 4: Virtuelle E/E-Integration in die Horizontale (Funktionssicht) und Vertikale (Steuergeräte-Sicht)

Die horizontale E/E-Integration stellt eine Funktionsintegration innerhalb eines Steuergeräts oder über verteilte Systemkomponenten, d.h. mehrere Steuergeräte, dar (siehe Abbildung 4). Die virtuelle horizontale Integration beschreibt somit eine rein funktionsorientierte Integration von SWCs mit SWCs durch Abstraktion von den HW-näheren Schichten. Die SW-/SW-Integration auf E/E-Systemebene ist durch Virtualisierung somit der SW-/HW-Integration vorgelagert. Für die Virtualisierung der horizontalen Integration kann es verschiedene Ansätze geben, die maßgeblich durch die Generationen der E/E-Architektur und somit auch durch AUTOSAR geprägt sind. Für frühere E/E-Architekturen mit zahlreichen, verteilten Steuergeräten (Architektur basierend auf AUTOSAR Classic) ist eine virtuelle Integration durch sogenannte virtuelle Steuergeräte (V-ECUs) zu ermöglichen, die durch den Zulieferer dem OEM bereitgestellt werden. Für die neue E/E-Architektur mit Hochleistungsrechnern (Architektur basierend auf der AUTOSAR Adaptive Plattform) kann die virtuelle Integration unabhängig vom HW-Target mittels Bereitstellung der SWCs und einer virtuellen Integrationsplattform bzw. einem Software-Stack bewältigt werden. Grundlage hierfür ist jedoch die Einhaltung von einheitlichen Funktionsarchitekturen und einer strikten Dekomposition von funktionalen Zusammenhängen entsprechend AUTOSAR Adaptive.

Neben der Bereitstellung von E/E-Komponenten als Software- bzw. Simulationskomponenten wird – wie erwähnt – eine virtuelle Integrationsumgebung bzw. -plattform zur Integration der verschiedenen Komponenten benötigt. Diese virtuelle Integrationsplattform muss weiterhin eine Architektur bereitstellen, die eine Umschaltbarkeit zwischen virtueller Integration (SW-/SW-Verbund) und Integration von realen Steuergeräten (SW-/HW-Verbund) für einen E/E-System- bzw. E/E-Gesamtverbund ermöglicht. Dies ist die Basis für die Durchgängigkeit der SW-/SW-Integration zur SW-/HW-Integration und somit für eine effiziente, gemischte Absicherungsstrategie an SiL- und HiL-Prüfständen für die E/E-Absicherung.

Virtuelle E/E-Absicherung

Die virtuelle E/E-Integration ist Eingangsgröße der virtuellen E/E-Absicherung an Software-in-the-Loop (SiL) und Hardware-in-the-Loop (HiL) Prüfständen (siehe Abbildung 5). Zur virtuellen Absicherung werden zusätzliche Simulationskomponenten benötigt, die teilweise nicht dem eigentlichen Wertschöpfungsprozess dienen, aber essentielle Hilfsmittel für die virtuelle E/E-Absicherung darstellen. Die benötigten Simulationskomponenten lassen sich in die drei Kategorien Virtuelles Fahrzeug, Virtuelle Umwelt und Virtueller Fahrer aufteilen:

- Virtuelles Fahrzeug: Neben der Virtualisierung der E/E-Komponenten (Steuergeräte oder Hochleistungsrechner) als Simulationskomponenten für SiL-Tests müssen auch vereinfachte physikalische Modelle der mechatronischen (Sub-)Systeme (z.B. Lenkung, Bremse) zur virtuellen Absicherung der funktionalen Anforderungen bereitgestellt werden.
- Virtuelle Umwelt: Die Simulationskomponenten dieser Kategorie bilden die physikalische und digitale Umwelt nach. Zur Virtualisierung der Umwelt gehören z.B. Sensormodelle, virtuelle Fahrstrecken und die Simulation von Ladesäulen.
- Virtueller Fahrer: Die Simulationskomponenten dieser Kategorie bilden das Fahrverhalten eines realen Fahrers nach. Die Fahrermodelle bzw. Verkehrssimulationen sind notwendig für eine intelligente, realitätsnahe Closed-Loop-Simulation von virtuellen Fahrzeugen in einer virtuellen Umwelt.

Mithilfe einer funktions- bzw. systemspezifischen Auswahl von Simulationskomponenten der drei Kategorien können anhand von szenarien-basierten Tests virtuell die funktionalen Anforderungen abgesichert werden.

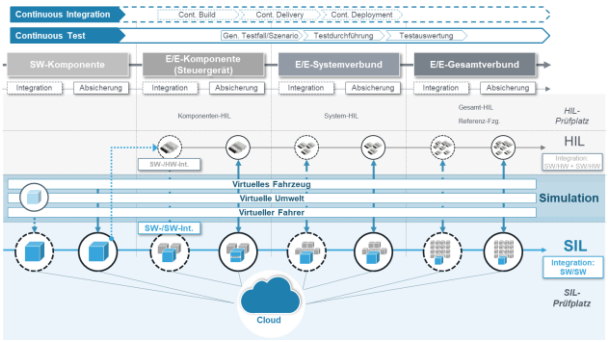

Abbildung 5: Logistischer Prozess für eine virtuelle E/E-Integration und -Absicherung

Die oben beschriebenen Ansätze der Virtualisierung der E/E-Integration und -Absicherung ermöglichen schneller integrierte Funktionshübe und die Sicherstellung von frühzeitiger Freigabefähigkeit durch Effizienzpotentiale. Dies stellt einen wesentlichen Beitrag zur Erreichung einer reduzierten Time-to-Market dar. Dieser Beitrag durch die Virtualisierung sollte durch eine hohe Automatisierung und Skalierbarkeit unterstützt werden. Enorme Effizienzhebel sind hierbei Ansätze wie Continuous Integration, Testautomatisierung und die Verlagerung der

Rechenaktivitäten bzw. Simulation in eine Cloud-Computing Umgebung für einen hohen Skalierungsfaktor.

DIGITAL TWIN: ZYKLISCHES VORGEHEN FÜR UPDATES UND UPGRADES VON SOFTWARE-BASIERTEN KUNDEN-FUNKTIONEN IN KURZEN ZYKLEN

Ein linearer Produktentstehungsprozess ohne Trennung der Hardware- und Software-Entwicklung ist nicht mehr hinreichend. Der Kunde erwartet kurze Releasezyklen von Software-Updates und -Upgrades und sicherheitskritische automatisierte Fahrfunktionen erfordern eine kurzfristige, flexible Reaktionsfähigkeit auf auftretende Fehler im Feld.

Neben der Entwicklung und Absicherung von software-basierten Kundenfunktionen ist der Einsatz digitaler Modelle in der Automobilindustrie bereits weit verbreitet. Die Modelle werden bei der Produktentwicklung (Konstruktion, z.B. Computer-Aided-Design), bei der Produktabsicherung (Virtueller Crashtest, z.B. Finite-Elemente-Methode), bei der Produktentstehung (Virtueller Versuchsbau, z.B. Digital-Mock-Up) und zur Visualisierung (z.B. hochwertige Renderings) eingesetzt. Allen hier dargestellten Anwendungsfällen ist gemein, dass sie im Vorfeld der konkreten Nutzung des Produkts zum Einsatz kommen. Das Konzept des Digital Twin geht über dieses Verständnis hinaus und betrachtet insbesondere die Betriebsphase eines Produkts. Aufgenommene Daten aus dem laufenden Betrieb fließen in das Modell zurück und können zur permanenten Optimierung genutzt werden.

Durch die Digitalisierung des Fahrzeugs wird das Fahrzeug immer mehr zu einem Software-Produkt. Produkte bzw. Systeme werden beim Digital Twin als Software-Repräsentation auf digitaler Ebene gespiegelt – und können damit folglich genauso wie eine Software über ihren gesamten Lebenszyklus behandelt werden. Der Betrieb und somit die Betrachtung des Fahrzeugs über dessen gesamten Lebenszyklus steht beim Digital Twin Ansatz für software-basierte Kundenfunktionen im Fokus. Die Datenaufnahme von Sensoren aus dem laufenden Betrieb des Kundenfahrzeugs bzw. beim Betrieb eines Erprobungsträgers im Feld fließen in die digitale Repräsentation zurück und können zur permanenten Optimierung der software-basierten Funktionen und somit für Software-Updates und -Upgrades genutzt werden. Eine Annäherung an den Digital Twin Ansatz mit einem zyklischen Entwicklungsvorgehen für Software-Updates und -Upgrades ist in Abbildung 6 dargestellt. Die vorgestellten Ansätze der Virtualisierung der Entwicklung und Absicherung software-basierter Funktionen sind hier eingeordnet (siehe Schritte 2-5).

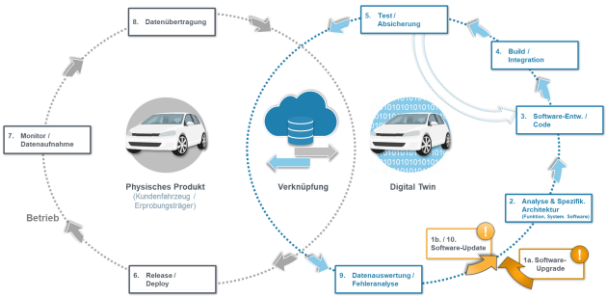

Abbildung 6: Zyklisches Vorgehen für Software-Updates und -Upgrades für software-basierte Funktionen (Digital Twin)

Das zyklische Vorgehen lässt sich vereinfacht in 10 Schritten beschreiben:

1a./1b. **Software-Upgrade / Software-Update**: Der zyklische Prozess beginnt mit zwei alternativen Prämissen, dem Software-Upgrade (1a.) oder Software-Update (1b. / 11). Für ein Software-Upgrade beginnt der Zyklus neu, folglich ist die rechte Kreishälfte mit deutlich höherem Aufwand verbunden. Vor Beginn eines neuen Zyklus für ein Software-Update wurde dieser bereits einmal durchlaufen und es wurde ein software-seitiger Fehler bzw. Mangel festgestellt. Dieser soll anschließend innerhalb des erneuten Durchlaufs behoben werden. In den Ausgangspunkten des Prozesses werden folglich die Anforderungen an das Software-Update bzw. Software-Upgrade spezifiziert.

2. **Analyse&Spezifikation Architektur:** Es wird jeweils die Funktions-, System- und Software-Architektur untersucht, ob ein Software-Update bzw. Upgrade realisierbar ist mit der vorhandenen Architektur. Weiterhin wird die Architektur daraufhin analysiert, ob ausreichend Ressourcen-Vorhalt über den gesteckten aktiven Wartungszeitraum des Fahrzeugs bereitsteht. Neue Anforderungen bzw. realisierbare Änderungen können durch den modellbasierten Ansatz des Systems Engineering frühzeitig und konsistent angepasst, simuliert und nachverfolgt werden.
3. **Software-Entwicklung/Code:** Zu verändernde bzw. neue Softwarekomponenten werden entwickelt. Durch die modellbasierte Vorgehensweise für die Entwicklung der Softwarekomponente kann der Software-Code bzw. das Kompilat mit Hilfe von Generatoren aus den Modellen erzeugt werden.
4. **Build/Integration:** Die Funktionsintegration mit den veränderten bzw. neuen Softwarekomponenten findet vorrangig per SW/-SW-Integration statt. Continuous Integration sorgt für eine effiziente virtuelle Integration und kann in einer Cloud-Computing Umgebung hoch skaliert werden.
5. **Test/Absicherung:** Nach der Funktionsintegration kann die software-basierte Funktion virtuell

abgesichert werden. Dafür werden die notwendigen Simulationskomponenten am SiL-Prüfstand bereitgestellt. Die Testautomatisierung dient zur effizienten virtuellen Absicherung und die Simulationsinstanzen können in einer Cloud-Computing Umgebung hoch skaliert werden.
Die Schritte 3 bis 5 werden so lange wiederholt, bis die spezifizierte Anforderung bzw. gewünschte Optimierung erfüllt ist.

6. **Release/Deploy:** Der Funktionshub bzw. die Aktualisierung der Kundenfunktionen findet durch Online Remote Update bzw. Upgrade statt.
7. **Monitor/Datenaufnahme:** Das Fahrzeug bzw. die software-basierte Funktion wird während des Betriebs in Kundenfahrzeugen oder Erprobungsträgern überwacht. Mit Hilfe von Corner-Case-Detektoren, zur Erkennung von kritischen Szenerien, werden relevante Daten aufgenommen, um anschließend Fehler aufzudecken (siehe Schritt 9). In frühen Entwicklungsphasen kann die untersuchte Funktion im Shadow Mode semi-aktiv sein, d.h. die Funktion leitet keine Befehle an die Aktuatoren weiter.
8. **Datenübertragung:** Die aufgenommenen Daten werden per Backend-Anbindung in die Cloud übertragen.
9. **Datenauswertung/Fehleranalyse:** In der Cloud werden per Big Data Analysemethoden die großen Datenmengen mit einem Label versehen und auf Fehler untersucht. Hierfür wird die Integrations- und Testumgebung aus den Schritten 4 und 5 genutzt.
10. **Software-Update:** Nach Aufdecken eines funktionalen software-seitigen Fehlers beginnt der zyklische Prozess erneut (siehe 1a./1b.).

In dem oben dargestellten zyklischen Vorgehen stellt insbesondere die Absicherung der Interaktion von Software und Hardware in SW-/HW-Verbünden für alle möglichen Fahrzeugkonfigurationen eine der zentralen Herausforderungen der Virtualisierung dar. Deshalb wird zeitnah eine gemischte Strategie an SiL- und HiL-Prüfständen für die E/E-Integration und -Absicherung unumgänglich. Weiterhin wird das durchgängige und konsistente Daten-, Konfigurations- und Änderungsmanagement in einem bzw. mehreren Software-Repositories ein wichtiges, aber komplexes Themenfeld bilden.

ZUSAMMENFASSUNG

Die Update- und Upgradefähigkeit von Fahrzeugen per Online Remote Update ist eine Reaktion auf neue Kundenbedürfnisse bzw. -funktionen und stellt eine Herausforderung für die Automobilindustrie dar. Die technische Grundlage wurde durch eine neue End-to-End E/E-Architektur (AUTOSAR Adaptive) geschaffen. Es sind zusätzlich erhebliche Veränderungen des Entwicklungsvorgehens erforderlich. Die Virtualisierung wird ein zentrales Werkzeug zur Beherrschung von Software-Updates und -Upgrades in kurzen Releasezyklen sein. Dabei kann das modellbasierte Systems Engineering als Methode für eine verteilte, domänenübergreifende und durchgängige Entwicklung dienen. Die Virtualisierung der E/E-Integration und -Absicherung ist essentieller Enabler und Effizienzhebel im Entwicklungsprozess. Das Konzept zur ganzheitlichen Virtualisierung mittels Digital Twin sollte für ein zyklisches Vorgehen bei der Entwicklung von software-basierten Kundenfunktionen in der Automobilindustrie eingebracht werden.

LITERATUR

Getos, “Virtuelle Absicherung in der Praxis – Funktions-SIL-Stationen bei BMW“, 8. dSPACE Anwenderkonferenz, München, 2016

Haasis, “Systems Engineering for future mobility”, REConf 2016, München, 2016

Holzmann, Hahn, et. al. “Simulation-Based ESC Homologation for Passenger Cars”, in: ATZ worldwide, 114(9), pp. 40-43, 2012

Krause, Timpner, et al. “Methoden zum entwicklungsbegleitenden Testen für automatisierte und vernetzte Fahrfunktionen,” in: AAET – Automatisiertes und vernetztes Fahren, pp. 88-109, Braunschweig, 2017

Markl, “The Adaptive Platform for Future Use Cases”, 8. Vector Congress, Stuttgart, 2016

Martinus, Deicke, Folie, “Virtueller Fahrversuch – Hardwareunabhängige Integration von Seriensoftware“, ATZ elektronik 8, pp. 344-349, 2013

Mikelsons, Samlaus, “Towards Virtual Validation of ECU Software using FMI”, in: Proceedings of the 12th International Modelica Conference, Prag, 2017

Oel, Pohl, et. al. “digital readiness - Virtual Integration and Validation in Volkswagen Development’s SimLAB”, in: VDI-Bericht 2299, ELIV 2017

Parrott, Warshaw, “Industry 4.0 and the digital twin – Manufactury meets its match”, Deloitte University Press, 2017

Ringler, “Virtuelle Integration zur frühen Absicherung von AUTOSAR-Applikationen“, 6. Vector Congress, Stuttgart, 2012

Schneider, Schick, Palm, “Virtualisation, Integration, and Simulation in the Context of Vehicle Systems Engineering”, embedded world 2012, Nürnberg, 2012

Stadler, Gruber, “Functional Engineering Platform – A Continuous Approach Towards Functional Development“, in: 7th Conference on Simulation and Testing for Vehicle Technology, Berlin, 2016

Vöst, “Die Industrielle Revolution des Testens”, 8. IBS–Workshop Automotive SW Engineering – Virtuelle Absicherung, Chemnitz, 2017

Waymo, “On the road to fully self-driving. Waymo Safety Report”, available at: https://storage.googleapis.com/sdc-prod/v1/safety-report/waymo-safety-report-2017-10.pdf (accessed 28 February 2018)

Wille, Krieger, “Ethernet & Adaptive AUTOSAR – Schlüsselelemente der neuen E/E-Architektur bei Volkswagen“, 3. Vector Automotive Ethernet Symposium, Stuttgart, 2017

KONTAKT

Alexander Strenge (Carmeq GmbH)
alexander.strenge@carmeq.com