Absicherung von komplexen Software-Komponenten vernetzter Fahrzeuge

Gereon Weiß, Christian Drabek
{gereon.weiss,christian.drabek}@esk.fraunhofer.de
Fraunhofer ESK, Hansastr. 32, 80686 München

Zusammenfassung. Die Menge an Software in Fahrzeugen nimmt nicht nur stetig zu, sondern übernimmt auch immer komplexere und kritischere Funktionen. Hiermit steigt auch die Komplexität der Schnittstellen und Funktionsinteraktionen stark an. Gute Beispiele hierfür sind die Hochintegration oder auch Fahrzeugumweltvernetzung. Damit diese Funktionen zukünftig in hoher Qualität realisiert und abgesichert werden können, sind neue Entwicklungs- und Absicherungskonzepte sowie Methoden unerlässlich. Im Beitrag werden die Herausforderungen zukünftiger vernetzter Fahrzeugsysteme diskutiert. Darüber hinaus wird eine modellbasierte Methodik vorgestellt, die eine Absicherung von komplexen Software-Schnittstellen ermöglicht, wie sie für Ethernet- oder V2X-basierte Funktionen notwendig sind.

1 Motivation

Durch die steigende Zahl an Diensten und die zunehmende Vernetzung wächst die Komplexität von Fahrzeugsoftware [1].

Abbildung 1: Beispielszenario zukünftig vernetzter Fahrzeuge

Zusätzlich stellt die interaktive Vernetzung zukünftiger Fahrfunktionen neue Anforderungen an die Absicherung von Fahrzeugfunktionen (vgl. Abbildung 1). Um diese Anforderungen zu erfüllen, werden neue Test- und Verifikationsmethoden benötigt. Insbesondere spielt hierbei die automatisierte, werkzeuggestützte Handhabung der Absicherung während des Entwicklungsprozesses eine bedeutende Rolle.
Hierbei sind für die Fahrzeugdomäne neue Interaktionsmechanismen (z.B. Service Discovery [2]), komplexere Schnittstellen (z.B. Connected Car- / App-APIs [3]) und Kommunikationstechnologien (z.B. CAN, Ethernet, SOME/IP, V2X [4]) zu berücksichtigen. Um diese Herausforderungen handhaben zu können, ist eine durchgängige Berücksichtigung der Kommunikationsschnittstellen von der Spezifikation bis zur Absicherung entscheidend.

2 Kommunikationsschnittstellen

Grundsätzlich besteht die Definition von Kommunikationsschnittstellen aus einer statischen Struktur (bspw. Funktionssignaturen mit Parametern) und dem dynamischen Verhalten (Ablauf der Kommunikation). In der Regel wird bei der Spezifikation Hauptaugenmerk auf statische Schnittstellenbeschreibung gelegt. Jedoch kommt dem Schnittstellenverhalten bei heutigen komplexen Interaktionen eine große Bedeutung zu, so dass auch diese in der Spezifikation berücksichtigt werden müssen. Häufig wird dies für einfache Fälle in Form von Sequenzabläufen spezifiziert. Dies stößt jedoch für zustandsbehaftete Systeme an Grenzen, da hier Abhängigkeiten zu vorangegangener Kommunikation nur schwer berücksichtigt werden können. Dies können Zustandsautomaten wiederum gut modellieren. Somit können Spezifikationen vernetzter Fahrzeugfunktionen gut durch statische Beschreibungen mit Zustandsautomaten bereitgestellt werden. Da bei der Realisierung heutiger Fahrzeugfunktionen sehr häufig eine Vielzahl von Zulieferern beteiligt ist, kommt hier der Schnittstellenkonformität eine besondere Rolle zu. Denkt man sogar an zukünftig fahrzeugübergreifende Funktionen, so wird deutlich, wie zunehmend wichtig ein konformes Kommunikationsverhalten wird.

Die Absicherung von Kommunikationsschnittstellen wird natürlich bereits langjährig adressiert (z.B. bei CAN). Durch komplexer werdende Interaktionen und neu adaptierte Technologien erhalten aber zukünftig neue Kommunikationsparadigmen Einzug in den Automotive-Bereich, wie etwa geswitchte Ethernet-Netzwerke im Fahrzeug oder dienstbasierte Kommunikation zu einem Backend. Daher ist eine Abstraktion von der Kommunikationstechnologie und der Anwendungslogik sinnvoll, um eine hohe Wartbarkeit und Wiederverwendbarkeit zu erreichen (vgl. Abbildung 2). Zur Beschreibung der Schnittstellen können Interface Description Languages (IDLs) verwendet werden, die sich wiederum meist teilautomatisiert auf konkrete Kommunika-

tionsprotokolle abbilden lassen. Aufbauend auf dieser Beschreibung können Events definiert werden, welche in der Anwendungslogik verwendet werden. Dies erlaubt eine Entkopplung der Anwendung von der Kommunikation. So können Anwendungen über verschiedene Kommunikationsprotokolle kommunizieren, ohne dass solche Beschreibungen angepasst werden müssen.

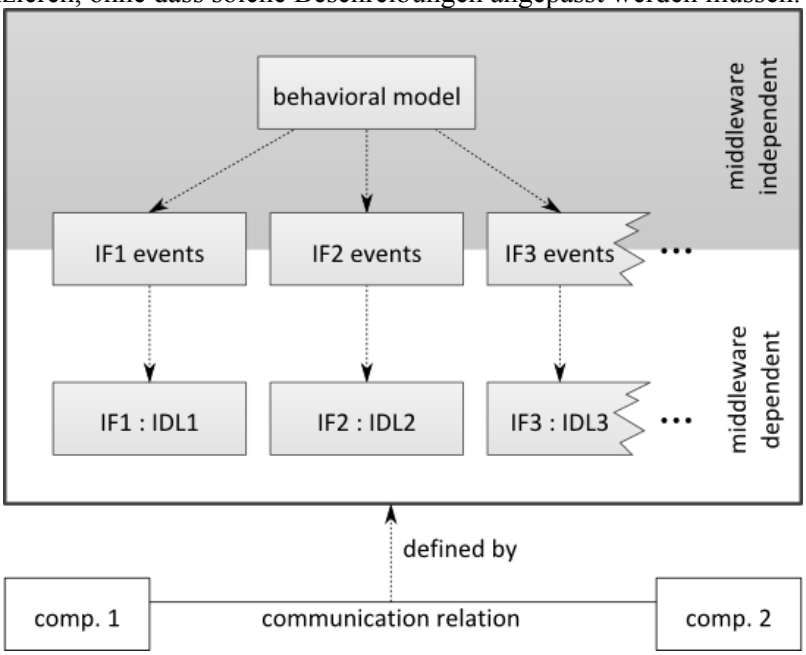

Abbildung 2: Abstraktion der Kommunikation

Wichtig bei einer solchen Absicherung von Komponentenschnittstellen sind der Grad der Automatisierung und die Integration in Software-Werkzeuge. Mit DANA [5] wurde eine Eclipse-basierte Werkzeugkette geschaffen, welche die Spezifikation, Simulation und Überprüfung von vernetzten Funktionen ermöglicht. Aufbauend auf einer IDL [6] können mittels einer Domain-Specific Language (DSL) kommunikationsunabhängige Events definiert werden. Diese können dann wiederum in den Modellen verwendet werden, welche das Schnittstellenverhalten beschreiben. Da DANA als offene Plattform im Eclipse-Framework [7] konzipiert ist, können einfach Erweiterungen als Plug-Ins realisiert und neue Anwendungsfelder adressiert werden. Im Folgenden werden beispielhaft ausgewählte Anwendungsbeispiele und Best Practices vorgestellt, die dies verdeutlichen.

3 Ausgewählte Beispiele

Steuergeräte, die viele Software-Komponenten beinhalten, können schnell von dem spezifizierten Sollverhalten abweichen. Dies muss nicht immer zu einem sichtbaren Fehlverhalten führen, sorgt jedoch in der Integrationsphase oder bei Zuliefererwechsel für Probleme. Hier haben sich insbesondere sich wiederholende oder fehlende Nachrichten als häufig auftretende Fehler gezeigt. In solchen Fällen kann die semi-formale Spezifikation von kritischen Anwendungsteilen besonders sinnvoll eingesetzt werden.

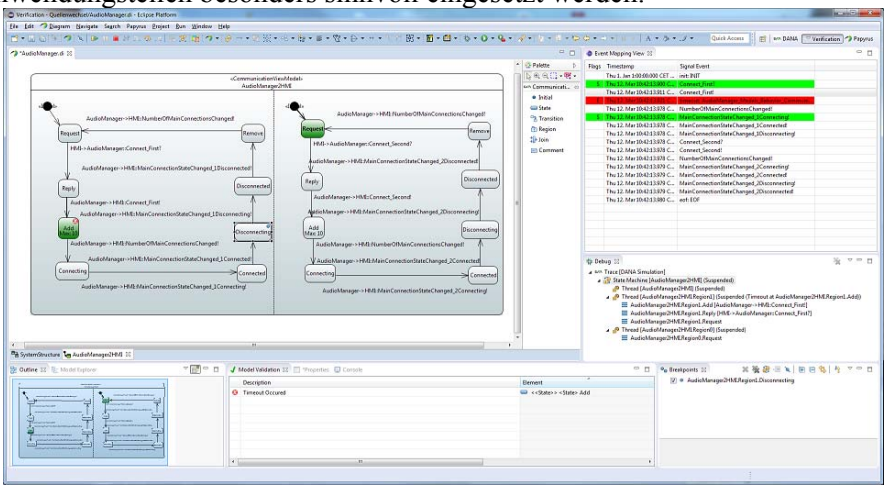

Abbildung 3: Beispielmodell eines Infotainment AudioManagers

Komplexe Applikationen zeigen ein unterschiedliches Schnittstellenverhalten bzgl. ihres aktuellen Zustands abhängig von kommunizierten Daten. Daher ist es wichtig, diese ausreichend im Modell berücksichtigen zu können. Besonders schwierig bei der Absicherung vernetzter Funktionen sind sogenannte spontane Fehler, die nur in sehr speziellen Fällen auftreten. Diese sind in der Praxis sowohl schwer nachzustellen als auch zu identifizieren. Um in langen Datentraces solche Fehler zu finden, ist der automatisierte Abgleich des Sollerhaltens mit dem implementierten Verhalten hilfreich. Dabei ist auch eine automatische Wiederaufnahme der Absicherung nach einem Fehler entscheidend, um Einzelfehler in langen Datenaufzeichnungen zu finden. Dazu kann eine Synchronisierung des Modells mit der Kommunikation genutzt werden. Die Abstraktion der Kommunikation erlaubt eine hohe Wiederverwendbarkeit und erleichtert die Fehlersuche durch einen Teile-und-Herrsche Ansatz. Beispielsweise können im Infotainmentbereich die gleichen Anwendungsmodelle wiederverwendet werden, egal ob eine DBUS oder SOME/IP-Kommunikation verwendet wird (wie z. B. bei einem AudioManager, s. in Abbildung 3).

4 Ausblick auf vernetzte Fahrzeuge

Die Kommunikationsschnittstellen von Fahrzeugen sind im rapiden Wandel. Neben dem Aufkommen neuer Kommunikationsarten und -technologien werden Autos übergreifend vernetzt. Dies führt unter anderem auch dazu, dass nicht mehr alle Daten an einem zentralen Punkt wie bei einem BUS zur Verfügung stehen und eine Absicherung darauf aufbauender Fahrfunktionen erschwert. Ein Beispiel hierfür ist eine Gefahrenwarner Assistenzfunktion, die über eine Vernetzung mit dem Backend oder anderen Fahrzeugen das Warnen vor Gefahrensituationen erlaubt. Die Auswirkungen solcher vernetzter Funktionen auf die Absicherungsmethoden [8] und Integration in Tools werden aktuell untersucht.

Literaturverzeichnis

[1] D. Juergens, et al., "Implementing Mixed Criticality Software Integration on Multicore - A Cost Model and the Lessons Learned," in *SAE International, Technical Paper 2015-01-0266*, 2015.

[2] L. Völker. (2014,) SOME/IP Service Discovery, Vector Automotive Ethernet Symposium. [Online]. https://vector.com/portal/medien/cmc/events/commercial_events/Automotive_Ethernet_Symposium_AES14/AES14_04_Voelker_BMW_Lecture.pdf

[3] HERE. (2015) Vehicle Sensor Data Cloud Ingestion Interface Specification. [Online]. https://company.here.com/automotive/new-innovations/sensor-ingestion/

[4] European Telecommunications Standards Institute (ETSI). (2016) Cooperative ITS. [Online]. http://www.etsi.org/index.php/technologies-clusters/technologies/intelligent-transport/cooperative-its

[5] Fraunhofer ESK. (2016) DANA Framework. [Online]. http://www.esk.fraunhofer.de/de/forschung/projekte/DANA.html

[6] (2016) Eclipse Franca. [Online]. http://projects.eclipse.org/projects/modeling.franca

[7] Eclipse Foundation. (2016) Eclipse. [Online]. https://www.eclipse.org

[8] C. Drabek, A. Paulic, and G. Weiss, "Reducing the verification effort for interfaces of automotive infotainment software," *Society of Automotive Engineers (SAE World Congress)*, 2015.