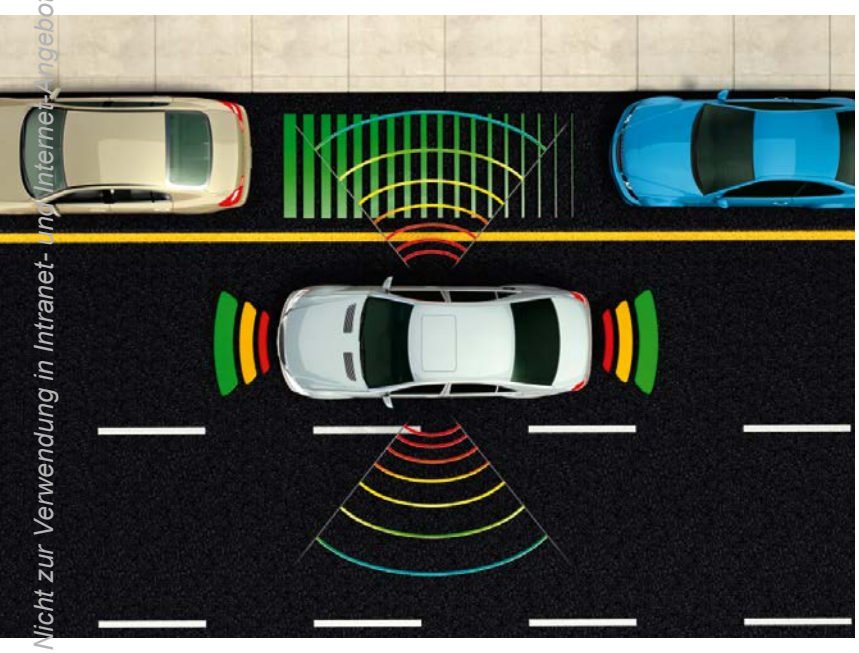
Software für autonome Kfz-Applikationen entwickeln



Das Konzept autonomer Fahrzeuge oder unbemannter Drohnen hat in der letzten Zeit ein beträchtliches öffentliches Interesse hervorgerufen. Aus kommerzieller Sicht scheint die Idee einleuchtend, doch birgt sie große Herausforderungen für Entwicklerteams, die dieser Idee Leben einhauchen sollen. Dieser Artikel beleuchtet die Standards, die Entwickler kennen sollten, und die Schritte, die es durchzuführen gilt, um autonome oder andere Automotive-Applikationen sicher zu machen.

Die meisten Entwicklungsmethoden für sicherheitsrelevante Systeme in Kraftfahrzeugen sind glücklicherweise schon im Rahmen der ISO 26262 formalisiert. Die Norm hat den Titel "Road Vehicles – Functional Safety" und ist Bestandteil der breiter gefassten IEC 61508 „Funktionale Sicherheit sicherheitsbezogener elektrischer und elektronischer Systeme für Kraftfahrzeuge". Die ISO 26262 wurde im November 2011 veröffentlicht und adressiert potenzielle Gefahr bringende Fehlfunktionen von elektrischen und elektronischen Fahrzeugsystemen. Sie betrifft die sicherheitskritischen Funktionen in allen gegenwärtigen Fahrzeugtypen. Das heißt, nicht nur in autonomen Fahrzeugen, auch in bemannten Fahrzeugen kommen immer häufiger Fahrerassistenzsysteme (Advanced Driver Assistance Systems ADAS, Bild 1) zum Einsatz, z. B. Spurwechselassistent/ Spurverlassenswarnung, welche die Kontrolle über wichtige Fahrfunktionen, wie Lenken und Bremsen, übernehmen können.

Entwicklungsprozess, Sicherheitsanalyse und Risikoklassifizierung

Wie also sollte ein Entwickler vorgehen, der mit einem Softwaredesignprojekt für ein autonomes Fahrzeug betraut ist? Die Entwicklung folgt im Allgemeinen dem in Bild 2 dargestellten Ablauf.

Eine Kernaufgabe im Entwicklungsprozess ist es, sämtliche Projektanforderungen zu identifizieren und diejenigen zu kennzeichnen, die das Potenzial einer Sicherheitsgefährdung bergen. Auf Basis einer solchen Sicherheitsanalyse wird ein Abgleich durchgeführt, der sowohl die Software- als auch die Hardwareplattform berücksichtigt und letztendlich eine Klassifizierung der Sicherheitsrisiken gemäß ASIL (Automotive Safety Integrity Levels) A, B, C oder D ermöglicht. ASIL-D ist die höchste Risikostufe und besagt, dass eine Fehlfunktion lebensbedrohliche oder tödliche Verletzungen zur Folge haben kann und daher ein Höchstmaß an Sicherheit zu gewährleisten ist. Entwickler ohne umfassende Erfahrung in Sachen funktionale Sicherheit sollten auf jeden Fall eine entsprechende Weiterbildung absolvieren, um auf dieser Basis alle Aspekte der Softwaresicherheit fachlich beurteilen zu können.

Software-Sicherheitsplan

Das formale Dokument, das all diese Informationen zusammenfasst, ist der so genannte Software-Sicherheitsplan. Er wird gewöhnlich in einer Konzeptbesprechung abgestimmt und deckt die Systemarchitektur, Module, Sicherheitsanforderungen und -funktionen, kritischen Pfade und Diagnosen ab. Die

Bilder: Wind River

Diagnose ist ein grundlegender Bestandteil jeder ISO-26262-Zertifizierung und soll die Ausfallrate senken. Der Software-Sicherheitsplan geht nicht im Detail auf die Funktionsweise der Software ein. Dies ist Aufgabe der Software-Anforderungsdokumentation; sie identifiziert die funktionalen Software-Units, die von den Entwicklern zu erstellen sind.

Der Software-Sicherheitsplan liefert Aussagen darüber, welche Software-Komponenten erforderlich sind. In den meisten Fällen wird ein Echtzeit-Betriebssystem (RTOS) als Basis-Software-Komponente spezifiziert. Die Auswahl eines RTOS erfordert weitreichende Überlegungen. Beispielsweise muss der Entwickler sicherstellen, dass das RTOS nach IEC 61508 zertifiziert ist. Durch die Verwendung entsprechend zertifizierter Software kann er sich auf die vom RTOS-Anbieter belegte Zertifizierung beziehen.

Partitionierung und Virtualisierung

Basis für das erforderliche Maß an Sicherheit ist die zeitliche und räumliche Trennung von Funktionen. Damit wird vermieden, dass sicherheitskritische Applikationen von nicht-sicherheitskritischen beeinflusst werden. Die räumliche Trennung verhindert, dass die Daten einer Partition die Daten oder den Programmcode in einer anderen Partition verändern. So lässt sich beispielsweise sicherstellen, dass Code, der auf Partition 1 abläuft, nicht auf Output-Devices zugreifen kann, die gerade von einer Applikation höherer Kritikalität verwendet werden, welche auf Partition 2 läuft. Die derzeit erhältlichen 32- und 64-bit-CPUs verfügen über eine Memory Management Unit (MMU), die diese Funktion softwareseitig unterstützt. Eine zeitliche oder temporäre Partitionierung stellt sicher, dass sicherheitskritische Applikationen die erforderliche Zeit für die Ausführung erhalten, z. B. für den Zugriff auf einen Prozessor, gemeinsam genutzte Ressourcen oder ein physikalisches Device, ohne dass eine Applikation niedrigerer Kritikalität zur selben Zeit darauf zugreifen kann.

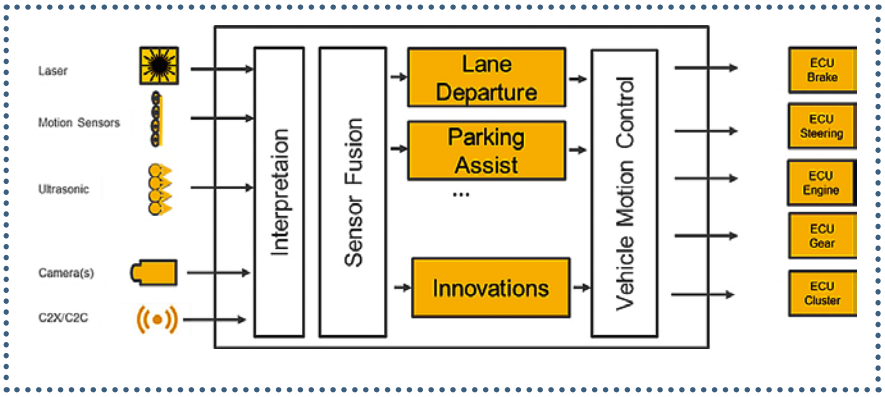


Bild 1: Entwurf eines ADAS-Systems.

In einem nächsten Schritt erfolgt ein Abgleich der Software-Sicherheitsanforderungen. Wie erwähnt ist die IEC 61508 eine Basisnorm und die ISO 26262 eine Applikationsnorm. Eine Zertifizierung nach IEC 61508 erfüllt zudem nicht automatisch die Anforderungen aus ISO 26262. Als Mindestanforderung ist eine Compliance-Matrix aus IEC 61508 nach ISO 26262 zu übertragen. Dieses Vorgehen stellt jedoch keine ideale Lösung dar, und das Ziel sollte die formale Zertifizierung nach ISO 26262 sein. Dies beinhaltet eine Betrachtung des Gesamtsystems über die Software hinaus sowie das Identifizieren aller entsprechenden sicherheitsrelevanten und nicht-sicherheitsrelevanten Funktionen.

Die Virtualisierungstechnologie ermöglicht es, eine zeitliche und räumliche Partitionierung in sicherheitskritischen Designs zu implementieren. Basierend auf Virtualisierung – auf einem Single- oder Multicore-Chip – können verschiedene Applikationen in geschützten Partitionen voneinander getrennt und über einen Hypervisor gesteuert ablaufen. Dies senkt den zeitlichen und finanziellen Aufwand »

Bild 2: Die wichtigsten Schritte im Entwicklungsprozess.

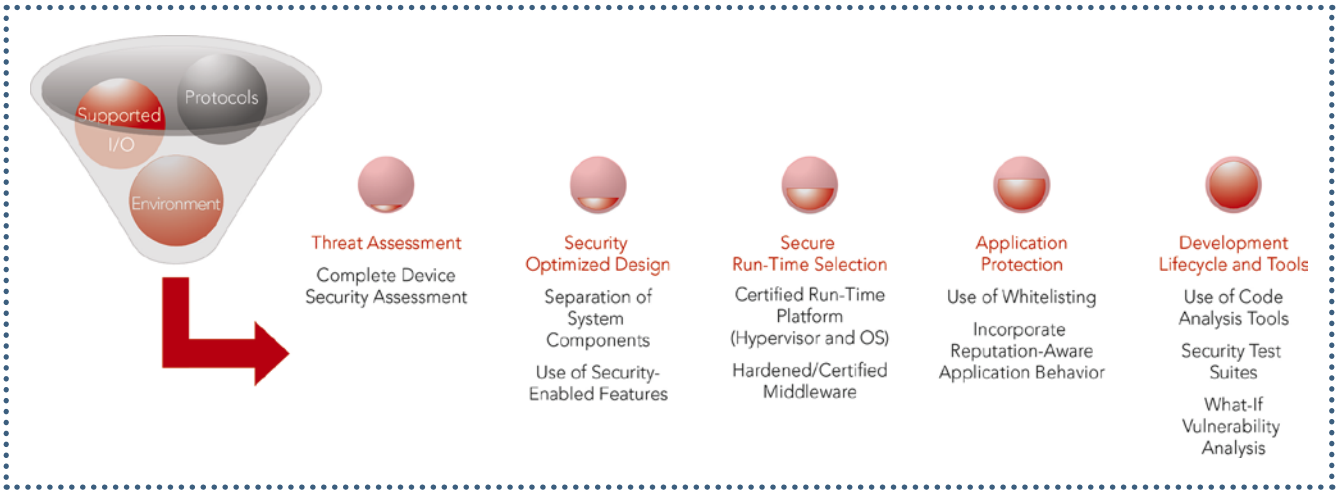


Bild 3: Schritte zu mehr Automotive-Embedded-Sicherheit.

sowie die Komplexität bei der Systementwicklung und beschleunigt gleichzeitig den Test- und Zertifizierungsprozess. Zudem erlaubt die Embedded-Virtualisierung dem Entwickler, elektronische Steuergeräte konsolidiert zu implementieren, was letztendlich wieder die Systemkosten und Footprints reduziert. Mit gängigen Embedded-Prozessoren ist es für Automotive-Softwareentwickler einfacher, mehrere Applikationen auf einem Prozessor zusammenzufassen, anstatt dafür mehrere separate Schaltungen zu verwenden. Wenn kundenspezifische Boards wegfallen, vereinfacht dies auch die Wartung erheblich.

Partitionierung führt zu höherer Systemsicherheit und schützt darüber hinaus die Software besser gegen Bedrohungen von außen, z. B. im Zusammenhang mit einem zeitlich ungünstigen Softwareupdate oder einem bösartigen Angriff (Bild 3). Jede Partition lässt sich über eine eigene Firewall schützen. Eine Lücke in einer Haupt-Firewall dagegen würde einem Eindringling den Zugriff auf das gesamte System ermöglichen. Ist eine Applikation beeinträchtigt, lässt sich ein Eindringen auf nur eine Partition eingrenzen und dort einfach erkennen und beseitigen. Dies spart Zeit, senkt Kosten und reduziert Sicherheitsrisiken. Darüber hinaus wird sichergestellt, dass Eindringlinge, besonders Schadprogramme, sich nicht in weiteren Systemkomponenten ausbreiten. Vor allem aber wird verhindert, dass Hacker auf den Netzwerkstack zugreifen und so weitere Angriffe starten oder aus der Distanz die Kontrolle über das Fahrzeug übernehmen können. Der Einsatz eines Echtzeit-Betriebssystems (RTOS), welches schon von Haus aus Partitionierung und Virtualisierung unterstützt, vereinfacht auch den Zertifizierungsprozess. Das RTOS VxWorks von Wind River beispielsweise unterstützt Virtualisierung sowie die räumliche und zeitliche Partitionierung; darüber hinaus ist es nach IEC 61508 zertifiziert.

Zertifizierung

Im Verlauf des Projekts wird die gesamte Applikation wie in jedem anderen Softwareprojekt verschiedenen Entwicklungsreviews, Tests, Verifikation und Validierung unterzogen. Sobald alles intern getestet wurde, ist eine unabhängige Zertifizierungsstelle, z. B. TÜV-Nord, Bureau Veritas oder Lloyds, für die Zertifizierung nach ISO 26262 und IEC 61508 hinzuzuziehen. Anhand der Testergebnisse und -berichte sowie weiterer Dokumente, die für die Zertifizierung nach ISO 26262 relevant sind, führt diese Stelle selbst eine Analyse der Applikation durch, anhand derer sich bestimmen lässt, ob der Entwicklungsprozess der Software unter Einhaltung der geltenden Vorgaben erfolgt ist und ob die Software entsprechend zertifiziert werden kann. Der gesamte Zertifizierungsprozess kann etliche Wochen in Anspruch nehmen und beinhaltet mehrere Reviews der zu testenden autonomen Fahrzeugplattform.

Fazit

Bis autonome Fahrzeuge auf unseren Straßen unterwegs sein werden, ist es noch ein langer Weg. Dennoch können Entwicklungsteams schon jetzt Strategien ausarbeiten und sich mit den wichtigsten Entwurfsbetrachtungen und künftigen Anforderungen in Sachen Zertifizierung auseinandersetzen. Angesichts der Komplexität autonomer Fahrzeuge und der kritischen Rolle, die die Software dabei spielt, wird es für Automotive-Unternehmen immer wichtiger, geeignete Partner mit entsprechenden Kompetenzen ausfindig zu machen, die ihnen dabei helfen, in der Spur zu bleiben. ■ (oe)

» **www.windriver.com**

Franz Walkembach ist Senior Product Manager Automotive Solutions bei Wind River in Ismaning.

Andreas Lindenthal ist Head of System Architects bei Wind River in Ismaning.