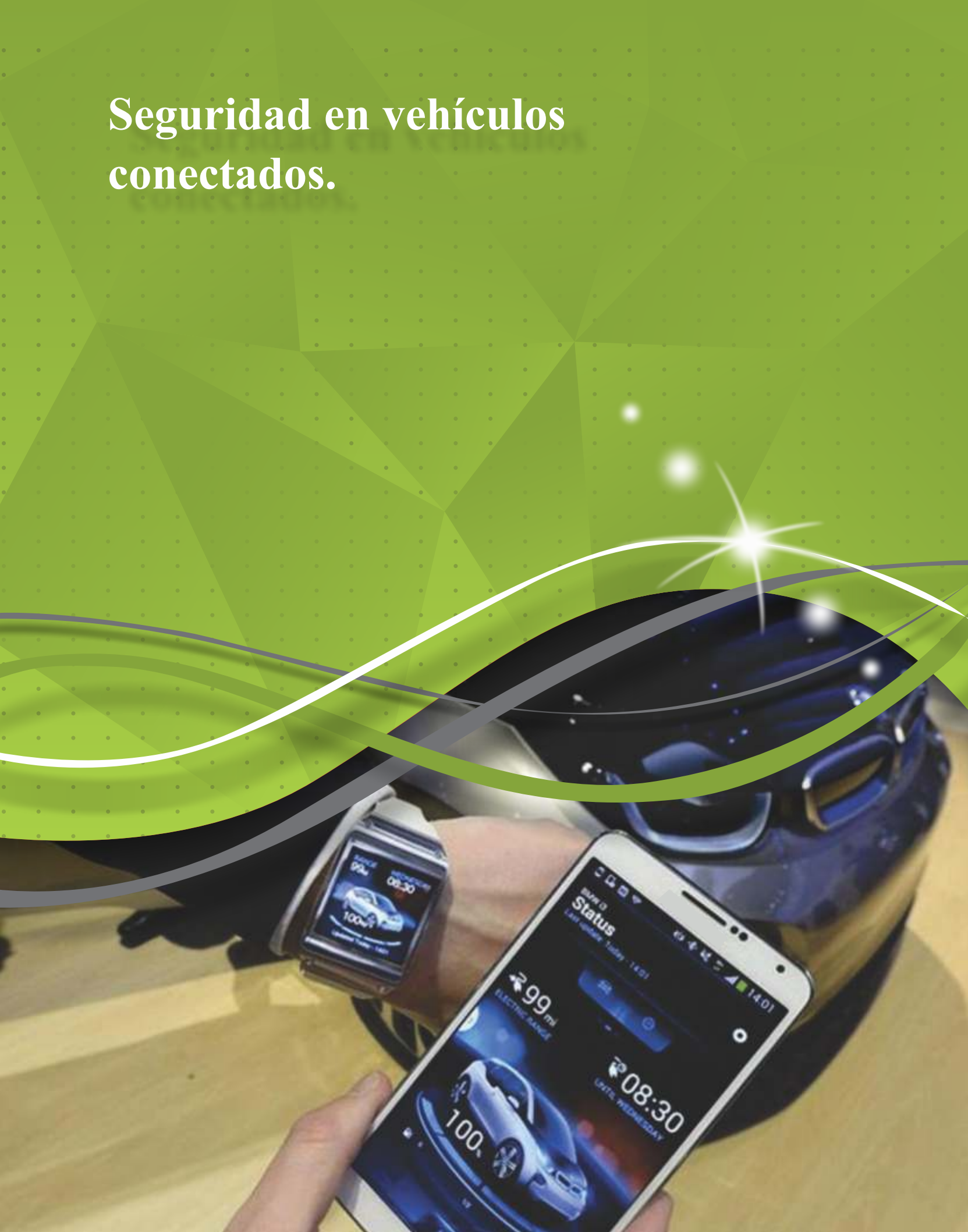
Seguridad en vehículos conectados.



Seguridad en vehículos conectados[1]

Safety and security in connected cars

André Hergenhan[2]
OpenSynergy GmbH Berlín, Alemania.

Artículo recibido en enero de 2015; artículo aceptado en febrero de 2015.

Citación del artículo: Hergenhan, A. (2015). Seguridad en vehículos conectados. *I+D Revista de Investigaciones*, 5(1), 70-81.

Resumen

El punto de partida de este artículo es la pregunta ¿sería posible integrar el vehículo en el Internet de las cosas sin perjudicar la seguridad de su red electrónica? Este interrogante conduce al término seguridad y sus diferentes aspectos en el contexto del desarrollo de software para dispositivos de vehículos. Después, se hará un resumen del estado actual de la técnica, especialmente de los esfuerzos en el área de la normalización, en el cual hubo avances notables en la última década hacia sistemas electrónicos seguros. A continuación se mostrarán las medidas constructivas para el mejoramiento de la seguridad del software en dispositivos electrónicos de vehículos. En concreto, se presentarán las posibilidades de las tecnologías de micronúcleo y virtualización y su utilización en el sistema operativo automotriz Coqos, un producto de la empresa alemana Open Synergy.

Para terminar, se describirá un sistema telemático para camiones que muestra las ventajas de Coqos en cuanto a seguridad.

***Palabras clave*:** Autosar, ISO 26262, sistema operativo, virtualización.

Abstract

The question, whether automotive ECUs (electronic control units) can easily be integrated into the Internet of Things (IoT) will serve as the starting point of this paper. It will lead us directly to the concepts of safety and security and their various aspects in the context of automotive software development. Thereafter, I will summarize the state of the art, especially the tremendous standardization efforts in the field over the past decade. Subsequently, I will show constructive measures which may improve safety and security of software of automotive ECUs. In

[1]Artículo de enfoque cualitativo, perteneciente al área de Ingeniería Industrial, subárea de seguridad. Fecha de inicio: 2014, fecha de terminación: 2015.

[2]Ingeniero electrónico (Dipl.-Ing.) de la Universidad Técnica de Dresde, Alemania, 1994. Doctorado en Informática (Dr. rer. nat.) de la Universidad de Tubinga, Alemania, 2001. Correo electrónico institucional: andre.hergenhan@opensynergy.com

particular, I will consider technologies like microkernel and virtualization and their application in COQOS, an operating system of the German company OpenSynergy that enables virtualization for automotive ECUs. At the end of the paper, I would like to emphasize the benefits of COQOS on the basis of a practical example.

Keywords: Autosar, ISO 26262, operating system, virtualization.

Introducción

La unidad principal (*head-unit*) es un dispositivo central en vehículos de última generación. No es solo una interfaz entre el hombre y el vehículo, sino también un punto esencial para integrar el vehículo en el Internet de las cosas. Por eso la *head-unit* implementa funcionalidad de productos electrónicos de consumo, de tecnologías de Internet, de manejo de confort y fuera de eso, sistemas de asistencia. Cuanto más conductores usan las aplicaciones en sus dispositivos móviles, tantos más fabricantes tienen que proveer los vehículos con estas y otras funcionalidades.

¿Entonces, implica esto que el auto será una nueva víctima de los *hackers* y la unidad principal es su portón abierto (Bederman, 2014)? ¡Nada de eso! No tiene que ser así. Ya existen soluciones para limitar o mitigar los riesgos. Al respecto en la última década, han pasado muchas cosas en el campo automotriz.

Seguridad en el ámbito automotriz

Cuando se habla sobre el tema de seguridad, ante todo se tiene que distinguir entre varios aspectos. No en vano en inglés hay dos palabras para eso: *safety* y *security.*

Safety

Safety representa la seguridad del sistema técnico en cuanto a confiabilidad, disponibilidad y tolerancia frente a defectos de hardware, por ejemplo por medio de redundancia, etc. Si se aplica este término a la unidad principal, significa que ésta, en primer lugar y a toda costa, necesita funcionar de manera definida en todo momento. Es decir, que el software en su totalidad debe comportarse de forma definida, pase lo que pase, con el hardware periférico, si ocurre un despropósito en los buses de datos o si parte del software se comporta de forma errática.

¿Parece obvio? Quizás, pero es muy difícil conseguirlo. Analice qué medidas deberían tomarse por ejemplo, para:

Integrar aplicaciones de proveedores distintos en un dispositivo, sin efectos secundarios (sin influencia mutua).

Asegurar que la radiación cósmica no tendrá efectos fatales.

Ignorar datos irregulares en las interfaces internas hacia los otros dispositivos («*babbling idiots*»).

Asegurar que la implementación corresponda exactamente a los requerimientos.

Poder justificar cada etapa de desarrollo y cada decisión de diseño, incluso unos años después.

Según la necesidad de reducción del riesgo, se distinguen cuatro niveles de integridad de seguridad (ASIL); por ejemplo, un dispositivo para controlar el *airbag* requerirá necesariamente un mayor nivel que un dispositivo para controlar la luz interior.

Security

A diferencia de *safety*, *security* trata la protección de sistemas contra perjuicios o manipulaciones de terceros, y comprende por lo tanto, la protección de: (a) informaciones, (b) infraestructura, (c) aplicaciones (programas) y (d) redes.

Con respecto a los dispositivos en el vehículo, significa entre otros, que se requiere:

Proteger los datos de las aplicaciones contra pérdida, mal uso, acceso no autorizado, modificación, publicación, robo, etc.

Proteger los datos de los buses internos (por ejemplo CAN, FlexRay, LIN o Ethernet) contra modificación, publicación, supresión o contra cambios en su planificación.

Evitar el espionaje de información, durante el trayecto entre la unidad principal en el vehículo y un servidor que esté ubicado en cualquier lugar fuera del vehículo.

Evitar ataques externos vía Internet, que intenten manipular la memoria de programación de las aplicaciones.

Medidas

Las fronteras entre *safety* y *security* se han desdibujado. A pesar de que los dos se enfocan en aspectos diferentes, hay cosas en común: las medidas relacionadas deberán reflejarse tanto en el diseño del sistema, de software y hardware, como en los métodos, procesos de desarrollo o medidas de organización.

No obstante, el propósito de este artículo será mostrar medidas constructivas para el diseño del software.

Tendencias en los sistemas electrónicos automotrices

Desarrollo hasta ahora

Desde los primeros dispositivos electrónicos, el sistema electrónico del vehículo ha evolucionado extraordinariamente. Hoy en día contiene más de cincuenta dispositivos, todos conectados por medio de buses.

Es interesante que esta arquitectura de sistema sea determinada por las relaciones comerciales entre los fabricantes de vehículos y sus proveedores y viceversa.

Un proveedor desarrolla y comercializa una «caja» de hardware y software, un nodo electrónico que tiene poca funcionalidad. Los costes por pieza de hardware son cruciales y por eso estos nodos se basan a menudo en microprocesadores de 8 o 16 bits, con unidades periféricas muy específicas.

El fabricante de vehículos define los requisitos del sistema, de interfaces (los datos de los buses), controla el desarrollo de sus proveedores y finalmente, tiene que integrar todos los dispositivos en la red electrónica del vehículo. Hasta hoy, hay pocos proveedores puros de hardware o software.

Crítica

Este sistema recuerda el principio de «divide y vencerás». La complejidad del sistema entero será fraccionada en varios sistemas menos complejos. El proveedor del sistema parcial solo tendrá que solucionar pequeños problemas, entre estos los que corresponden a *safety* y *security*. Las influencias mutuas de las funcionalidades de dispositivos distintos, prácticamente no existen, excepto los datos en sus interfaces. Por eso, el proveedor asumirá tanto la responsabilidad por la funcionalidad del dispositivo, como por su *safety* y su *security*.

Desafortunadamente hay también desventajas, cuya importancia aumentará con la mayor funcionalidad en el futuro.

El ramal de cables es ya hoy una de las piezas más pesadas del vehículo. El aumento del número de dispositivos en la red electrónica implicará el aumento de:

- Peso del ramal de cables y el aumento de consumo de combustible consecuente.
- Escasez de espacio en el vehículo para instalarlos.
- Fuentes de errores a causa de la multiplicidad de conexiones entre los dispositivos.

Además, hay otras tendencias que han cuestionado la arquitectura en la práctica:

Los fabricantes de vehículos desean diferenciarse de la competencia por su funcionalidad basada en software, particularmente en sistemas de asistencia. Por lo tanto, ellos serán también proveedores de software.

En el mundo del Internet de las cosas, el vehículo es solo un integrante entre muchos. No obstante, las aplicaciones en la nube están distribuidas entre varios integrantes. Es decir, habrá proveedores que no ofrecen hardware y mucho menos sistemas embebidos automotrices.

Desarrollo en el futuro

Dadas las desventajas de la arquitectura actual, para los futuros sistemas el sector automotriz está cambiando su modo de pensar desde hace tiempo: salir de la funcionalidad en «cajas» particulares, hacia software integrado a pocos ordenadores comunes del vehículo. Sin embargo, hay que superar una serie de retos: Uno de los más importantes consiste en establecer plataformas de software, es decir sistemas operativos, para facilitar la integración de las diferentes aplicaciones de manera segura en un solo hardware. Por supuesto, es más difícil prevenir los efectos secundarios si todas las aplicaciones usan los mismos recursos comunes, como la CPU o la memoria. Sin embargo, hay que alcanzar el mismo nivel de seguridad que con la arquitectura tradicional.

Solo entonces los proveedores de los sistemas electrónicos continuarán asumiendo la responsabilidad por sus productos, tanto el hardware como el software.

Estado actual de la técnica

A continuación se discute sobre algunas tecnologías que representan el estado actual de la técnica en el campo automotriz y que a la vez son importantes para la comprensión del análisis siguiente. La lista no tiene la pretensión de ser exhaustiva, solo dar a conocer las principales tecnologías influyentes en este artículo.

Esfuerzos de normalización

Las normalizaciones reflejan sobre todo el estado actual de la técnica de un sector entero: «*best practices*» son recolectadas y evaluadas para ser generalizadas. A partir de entonces, nada puede estar por debajo de estos criterios. Es decir, cada normalización en sí, es una contribución para establecer seguridad.

Autosar. Autosar es una asociación de fabricantes de vehículos proveedores de software y hardware, y desarrolladores de herramientas de software a nivel mundial, cuyo objetivo es desarrollar y establecer una arquitectura abierta para el sistema electrónico del vehículo (Autosar GbR, s.f.).

Dicho de manera simplificada, Autosar detalla una plataforma de integración para componentes de software a nivel de todo el sistema electrónico del vehículo. Por lo tanto, se está especificando: (a) una arquitectura de software para dispositivos compuesta por un software básico (incluido el sistema operativo), componentes del software y una middleware como adhesivo entre los componentes del software y el software básico; (b) métodos de desarrollo; (c) formatos de intercambio para datos y (d) herramientas para apoyar métodos de la ingeniería dirigidos por modelos.

Genivi. La Alianza Genivi es también un consorcio de fabricantes de vehículos y proveedores a nivel mundial. A diferencia de Autosar, Genivi solo está enfocado en establecer un sistema operativo global para dispositivos de *infotainment* (infoentretenimiento) basado en Linux (Genivi Alliance, s.f.).

ISO 26262. La norma ISO 26262 se deriva de la norma IEC 61508 y detalla normas para la seguridad (*safety*) de los sistemas eléctricos / electrónicos en los vehículos.

Aesas. Aesas es un consorcio de proveedores automotrices alemanes (AESAS, s.f.).

En lo referente a la seguridad, Aesas define cómo los proveedores de software a través de Autosar deberían aplicar la norma ISO 26262 (en particular el tema de SEooC).

Common Criteria. *Common Criteria* es una norma internacional (ISO/IEC 15408) que detalla la certificación de *security* de ordenadores.

Sistemas operativos

Osek-OS, Osek-Time. OSEK (Osek, s.f.) es un gremio de estandarización automotriz que se puede considerar como el precursor de Autosar.

OSEK-OS es una especificación para un sistema operativo en tiempo real, sobre todo por microcontroladores de 8 y 16 bits, usados por los dispositivos pequeños de arquitectura de sistema actual.

OSEK-Time es una norma de un sistema operativo en tiempo real específico para sistemas dirigidos a tiempo (FlexRay).

Autosar OS. Autosar OS (Autosar GbR, s.f.) se puede considerar como una elaboración de Osek-OS y Osek-Time. A diferencia de estos, la norma de Autosar OS detalla además funcionalidades protegidas como protección de memoria o el control de comportamiento temporal de programas.

Si se habla de hardware, Autosar OS soporta todo el rango de microprocesadores contemporáneos, comenzando por los microcontroladores hasta los de varios núcleos.

En este sentido, Autosar OS podría ser un módulo hacia una arquitectura electrónica con pocos nodos de computación.

Virtualización de sistemas operativos. La virtualización es una tecnología informática que permite ejecutar un sistema operativo dentro de otro. La capa de arquitectura que provee la virtualización, se llama normalmente «hypervisor».

Esa tecnología es conocida sobre todo en el sector de servidores web. Pero desde hace varios años también se intenta usar en el campo de sistemas embebidos. No obstante, requiere procesadores modernos que, por ejemplo, soporten la protección de memoria.

Debido a sus propiedades, esta tecnología es aplicable para migrar los sistemas de software enteros de «cajas» antiguas a nodos de computación más grandes del futuro.

Micronúcleo (microkernel). El término micronúcleo (*microkernel* en inglés) denomina una manera particular de construcción de los núcleos para sistemas operativos, diferente de los núcleos monolíticos.

El micronúcleo provee solamente un conjunto de servicios y métodos básicos, como servicios para la administración de memoria y procesos para la comunicación entre procesos o para sus planificaciones. Todos los otros servicios son ejecutados por fuera del núcleo.

En consecuencia, la implementación del micronúcleo puede ser muy pequeña. Cuanto menos complejo el código, menor es la posibilidad de que existan errores, sobre todo errores con consecuencias fatales. ¡Téngase en cuenta que el código del núcleo es ejecutado en el modo «supervisor»! Por eso, aún existen enfoques para verificar matemáticamente que el código sea correcto (Nicta, s.f.).

L4 se refiere a una especificación de un micronúcleo particular con varias implementaciones por sistemas embebidos. Sobre la base del micronúcleo L4 se han establecido partes numerosas: componentes para sistemas operativos en tiempo real, capas de middleware o hypervisores y herramientas.

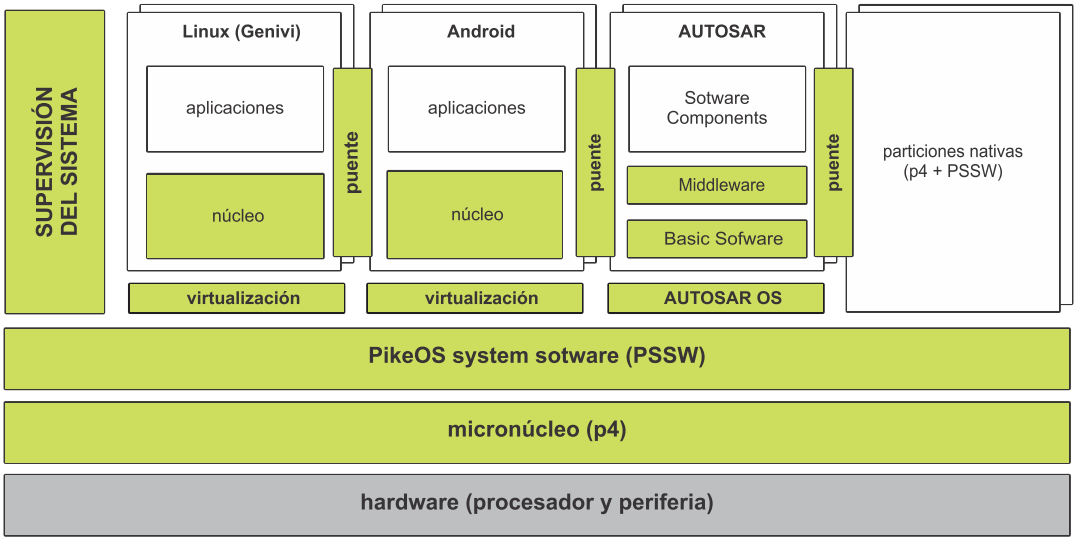
La tecnología L4 tiene suficiente madurez, lo que tiene gran relevancia en la industria desde hace algún tiempo. Por ejemplo, el celular seguro del gobierno alemán («Merkelphone») lleva un sistema operativo basado en micronúcleo L4 (Trust2Core GmbH, s.f.).

Entonces, sistemas operativos a base de micronúcleos pueden también contribuir a la seguridad (en particular *safety*) de los dispositivos del vehículo.

Coqos - un sistema operativo automotriz apoyado en virtualización

Coqos (Open Synergy GmbH, s.f.-a) es el sistema operativo automotriz de Open Synergy (Open Synergy GmbH, s.f.-b) que incorpora todos los aspectos ya mencionados.

Figura 1. La arquitectura de sistemas basados en COQOSCoqos



Fuente: El autor

Arquitectura

La Figura 1 muestra la arquitectura fundamental de sistemas basados en COQOS; los elementos propios de éste están marcados en verde.

Así, estos sistemas se componen de las siguientes partes:

El micronúcleo p4 que es una implementación de norma L4.

Un elemento llamado *Pike OS system software* (PSSW) que usa p4[5] e implementa servicios del sistema operativo.

Una o varias particiones que proveen una plataforma a través de la norma Autosar, la cual consiste en: (a) una capa que provee servicios de Autosar OS; (b) los módulos de *Autosar Basic Software*; (c) un Autosar middleware y (d) los *Autosar Software Components*.

- Una o varias particiones ejecutando Android, el cual consiste en: (a) una capa de para-virtualización, (b) un núcleo Android adaptado y (c) las aplicaciones en el espacio de usuario.
- Una o varias particiones ejecutando Linux (Genivi), el cual consiste en: (a) una capa de para-virtualización, (b) un núcleo Linux adaptado y (c) las aplicaciones en el espacio de usuario.
- Una o varias particiones que implementen aplicaciones en tiempo real basadas directamente en p4 o PSSW.
- Una partición de supervisión que controle el ciclo vital fundamental y supervise el comportamiento (el estado) de todas las demás particiones.
- Puentes entre las particiones distintas, que permiten una comunicación segura entre ellas.

[5] p4 y Pike OS system software (PSSW) son productos de la empresa alemana Sysgo (Sysgo GmbH, s.f.)

Núcleo p4 y PSSW – elementos claves para la seguridad

Micronúcleo p4

El micronúcleo p4 es un elemento clave del sistema seguro, porque su código se ejecuta en el modo supervisor y por eso utiliza todas las propiedades de los procesadores, para ofrecer servicios de seguridad.

Sin embargo, el núcleo no es responsable de generar objetos o gestionar el sistema de ningún modo. En este sentido, p4 solamente ofrece:

- Capas para adaptar el hardware.
- Servicios para generar objetos del núcleo (procesos - *tasks*, hilos - *threads*, particiones).
- Servicios para gestionar la memoria (alocación, *mapping*, administración de la caché).
- Servicios para permitir comunicación entre procesos (vía el núcleo) y administrar eventos asíncronos.
- Programación (*scheduling*).
- Servicios para controlar derechos (de comunicación, la capacidad de crear objetos del núcleo y la capacidad de acceder a partes de memoria).
- Capacidades de sincronización entre hilos en paralelo.

Ámbito de ejecución – PSSW

Al contrario del micronúcleo p4, la capa de *Pike OS system software* (PSSW) no solo provee servicios del sistema operativo clásico basado en núcleo p4, sino que controla el ciclo de vida del sistema en general, a través de una configuración estática. La configuración (*virtual machine initialization table* - VMIT) incluye el listado de:

- Todos los objetos del núcleo que necesiten ser creados: particiones, procesos, hilos.
- Todas las relaciones de comunicación entre las particiones o procesos.
- Todas las capacidades o derechos de los objetos del núcleo.
- Los presupuestos de tiempo disponible para las particiones.

Este mecanismo asegura que sea prácticamente imposible crear objetos adicionales del núcleo, durante el tiempo de ejecución del sistema. De esta forma se contribuirá una vez más a la seguridad (*safety*) del sistema, ya que así no puede ser comprometido fácilmente.

Adicionalmente, el PSSW implementa servicios para supervisar el estado del sistema, particularmente sus particiones, procesos e hilos, los cuales se pueden usar para implementar «*health monitoring*» de acuerdo con la norma de la industria aeronáutica Arinc 563.

Peculiaridades

El sistema Pike OS (p4 y PSSW) ha sido desarrollado originalmente para sistemas electrónicos basados en software de la industria aeronáutica. Así, un objetivo en su desarrollo ha sido cumplir con los estándares de esa industria y por eso Pike OS satisface los más altos requisitos respecto a *safety* y *security*.

Pike OS fue certificado por las normas DO-178B DAL B, IEC 61508 SIL-3 y EN50128 SIL-3/4.

Propiedades de Coqos

Con todos los aspectos ya mencionados, los sistemas con Coqos permiten:

- El desarrollo de subsistemas que cumplan con la norma Autosar.
- El desarrollo de subsistemas en tiempo real.
- La integración de aplicaciones y sistemas enteros basados en Linux, Genivi o Android, gracias a la utilización de tecnología de virtualización.
- El mejoramiento de la seguridad, es decir, *safety* y *security* mediante: (a) la utilización de un micronúcleo ya certificado; (b) la asignación y limitación de recursos a subsistemas (particiones, procesos) y su seguimiento como la limitación de presupuesto de tiempo, la limitación de acceso a entradas/salidas (I/O), la limitación de acceso a áreas de la memoria específicas (protección de memoria) o la limitación de canales de comunicación (cortafuegos); (c) «*health monitoring*» parecido a la norma de la industria aeronáutica; (d) la integración de subsistemas con propiedades distintas (de tiempo real, de niveles ASIL distintos o virtualizados) sin efectos secundarios; (e) la integración de aplicaciones de proveedores, distintas, sin efectos secundarios (software como producto) y (f) subsistemas que cumplan con la norma ISO 26262.
- La implementación de casos de uso típicos de forma sencilla, por ejemplo de: (a) subsistemas que los obliguen a arrancar muy rápido en el inicio (por ejemplo cámara de marcha atrás) y (b) subsistemas de diagnóstico.
- El aprovechamiento de hardware de la manera eficiente, escalable y flexible, ya que Coqos facilita la utilización de: (a) tanto procesadores pequeños como procesadores con multinúcleos y (b) hardware gráfico moderno.
- La reutilización de *frameworks* de la electrónica de consumo.

Gama de aplicaciones de Coqos

Así, Coqos es especialmente apto para diseñar:

- Sistemas de infoentretenimiento (unidades principales - *head-units*).
- Sistemas de conectividad para integrar el vehículo en el Internet de las cosas, incluidos sistemas telemáticos.
- Sistemas de asistencia.
- Sistemas Multi-Autosar para integrar productos de varios proveedores en un nodo de computación.

Ejemplo: El siguiente ejemplo se inspira en un proyecto de vehículo desarrollado realmente. En concreto, describe un sistema telemático para flotas de camiones.

Requerimientos funcionales

El dispositivo telemático debería implementar, entre otros, los siguientes requerimientos funcionales:

- Geolocalización para rastrear el camión vía GPS y GSM.
- Carga de datos de navegación de un portal del operador logístico, al dispositivo telemático del camión vía GSM.
- Acceso remoto vía GSM al tacógrafo digital por parte del operador logístico, para leer datos del conductor, como tiempo de conducción y reposo.

- Lectura de memorias de errores de dispositivos enteros del vehículo de forma remota vía GSM, por parte del operador logístico.

Contexto

La Figura 2 muestra el contexto del sistema telemático con dos partes interesadas: el operador logístico por un lado y el vehículo y su red electrónica por el otro.

El operador logístico maneja un portal telemático (una aplicación web) y puede comunicarse con el dispositivo telemático del camión vía una interfaz de aire (GSM).

Un subsistema de GPS de dispositivo telemático ofrece la posibilidad de localizar cada vehículo en particular.

Debido a que el dispositivo telemático está conectado con todos los demás, hay una ruta desde el portal telemático hacia la red electrónica interna. Ésta es necesaria para cumplir con los requerimientos, pero claramente también es un riesgo.

Requerimientos adicionales

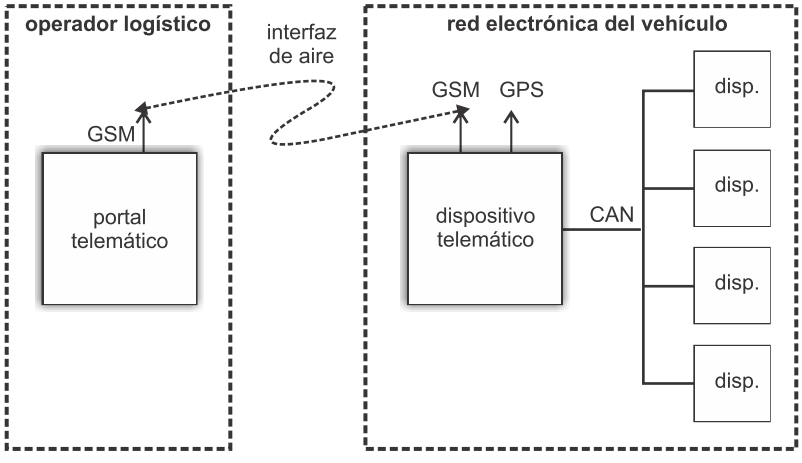
Además de requerimientos funcionales, hubo otras condiciones:

- El fabricante del vehículo deseaba suministrar software en forma de *Autosar software components*.
- El fabricante del vehículo requería una arquitectura que cumpliera con la norma Autosar.
- Hubo un consorcio de tres empresas que desarrollaron en conjunto el dispositivo telemático: (a) el proveedor de hardware, (b) Open Synergy como proveedor del sistema operativo Coqos y como integrador de todas las partes del software y (c) un proveedor de un *framework* telemático basado en Linux, el cual debía ser necesariamente reutilizado y de funcionalidad telemática.

Solución

La arquitectura fundamental se basó en Coqos (véase también Figura 1) con base en Pike OS y tres particiones: una partición Autosar, una partición Linux virtualizada y una partición que

Figura 2. El contexto de ejemplo



Fuente: El autor

implementa la supervisión de las otras dos. Un puente entre Autosar y Linux facilita la comunicación necesaria.

Esta arquitectura tiene unas ventajas:

- El subsistema Autosar requerido pudo implementarse.
- Un framework telemático disponible en Linux pudo reutilizarse.
- El fabricante del vehículo pudo suministrar Autosar software components que fueron integrados en el subsistema Autosar.
- Autosar y Linux pudieron integrarse a una plataforma sin efectos secundarios: (a) no es posible manipular la memoria de los demás y (b) cada uno tiene su propia programación (*schedule*).
- El puente entre ambas particiones actúa como un cortafuegos de manera controlable y segura, es decir, hay una ruta de afuera hacia adentro del vehículo, pero solamente para comunicaciones permitidas.

Conclusión

Este texto se ocupó de aspectos, problemas y soluciones que son importantes para el desarrollo de dispositivos seguros en vehículos, en especial cuando estos deben ser integrados en el Internet de las cosas.

Muchos de los aspectos discutidos aquí son válidos en general y pueden ser utilizados en otras áreas en donde también se requiera la integración de dispositivos electrónicos en el Internet de las cosas.

Referencias

Aesas. (s.f.). *La página de inicio.* Descargado 15/09/2014, de http://www.aesas.org/

Autosar GbR. (s.f.). *La página de inicio.* Descargado 15/09/2014, de http://www.autosar.org/

Bederman, U. (2014). *El auto, ¿una nueva víctima de los hackers?* Descargado 15/09/2014, de http://www.dattamagazine.com/el-auto-una-nueva-victima- de-los-hackers/

Genivi Alliance. (s.f.). *La página de inicio.* Descargado 15/09/2014, de http://www.genivi.org/

Nicta. (s.f.). *Secure Microkernel Project (seL4).* Descargado 15/09/2014, de http://ssrg.nicta.com.au/projects/seL4/

Open Synergy GmbH. (s.f.-a). *COQOS Operating System*. Descargado 15/09/2014, de http://www.opensynergy.com/en/products/coqos/

Open Synergy GmbH. (s.f-b). *La página de inicio*. Descargado 15/09/2014, de http://www.opensynergy.com/

OSEK. (s.f.). *La página de inicio.* Descargado 15/09/2014, de http://osek-vdx.org

Sysgo GmbH. (s.f.). *La página de inicio.* Descargado 15/09/2014, de http://www.sysgo.com

Trust2Core GmbH. (s.f.). *La página de inicio.* Descargado 15/09/2014, de http://www.trust2core.de/