Linux as a Hypervisor for the automotive Industry.

Andreas Platschek <andi@opentech.at>
Nicholas McGuire <der.herr@hofr.at>

OVERSEE

An open and secure application and communication platform

OpenTech

oversee

Open Vehicular Secure Platform

SEVENTH FRAMEWORK PROGRAMME

Some Facts

- www.oversee-project.com
- OVERSEE is the acronym for Open VEhiculaR SecurE platform
- European research project funded within the 7th
- Framework Programme of the European Commission.
- Duration: 30 Month
- Start: January 1st, 2010

Author: change in footnote

The Consortium

Industrial Partners:

Academic Partners:

Goals of OVERSEE

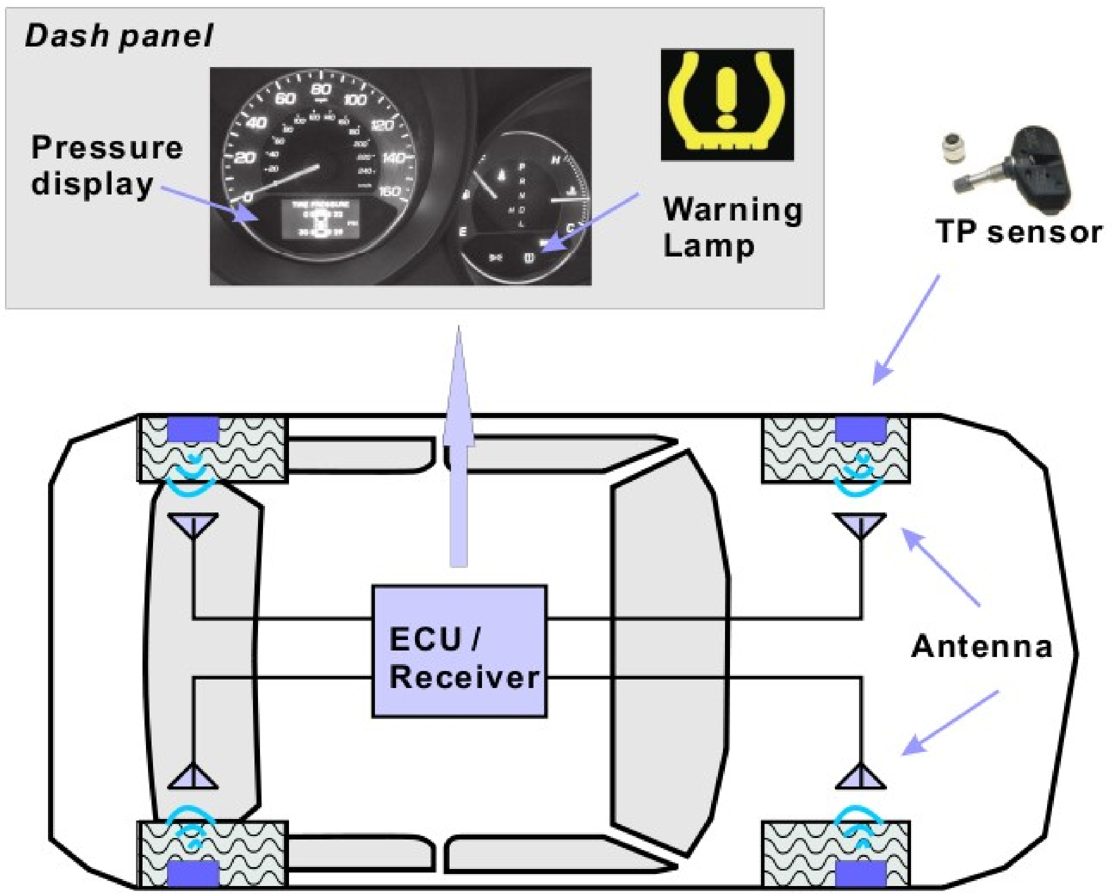
The idea of OVERSEE can be split in three main parts:

- The open platform for the execution of OEM and non OEM applications,
- the secure single point of access to
- internal and external communication channels.

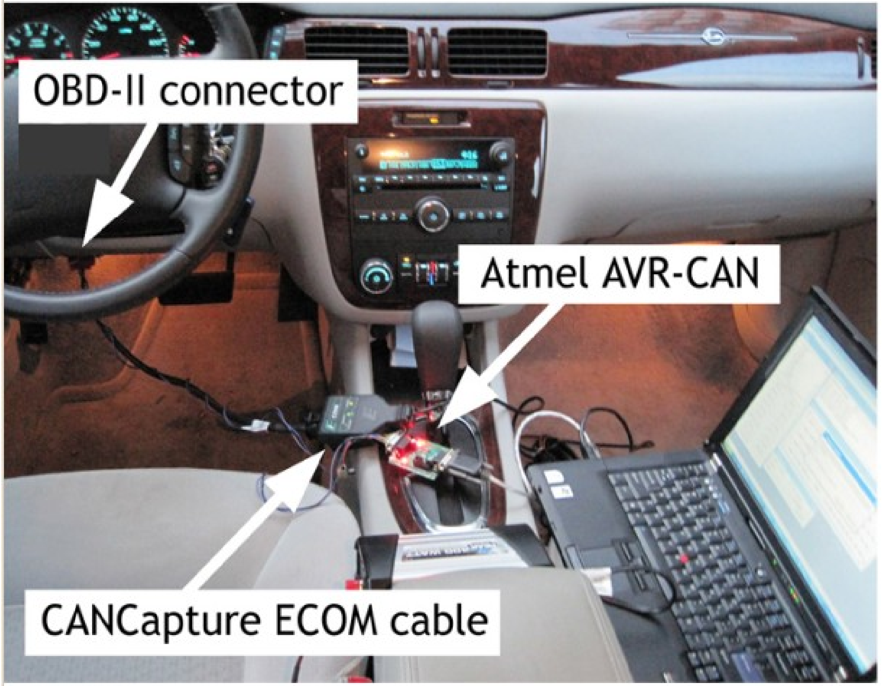
However, only the combination of these three objectives will offer the potential for a wide range of new automotive application.

Open architectures enable an industry-wide participation and portable software modules, and allow even small companies to enter this sector.

TPMS - Tire Pressure Monitoring System[1]



Experimental Security Analysis of a Modern Automobile[2]



Common Criteria

- “Common Criteria for Information Technology Security Evaluation” - ISO/IEC 15408
- 7 EAL's => The higher the number the more secure the system is supposed to be. (Not necessarily true)
- RHEL5 certified at EAL4+

Evaluation Assurance Levels

- EAL1 - functionally tested
- EAL2 - structurally tested
- EAL3 - methodically tested and checked
- EAL4 - methodically designed, tested and reviewed
- EAL5 - semiformally designed and tested
- EAL6 - semiformally verified design and tested
- EAL7 - formally verified design and tested

Evaluation Assurance Levels

Assurance class	Assurance Family	Assurance Components by Evaluation Assurance Level						
		EAL1	EAL2	EAL3	EAL4	EAL5	EAL6	EAL7
Development	ADV_ARC		**1**	1	1	1	1	1
	ADV_FSP	**1**	**2**	**3**	**4**	**5**	5	**6**
	ADV_IMP				**1**	1	**2**	2
	ADV_INT					**2**	**3**	3
	ADV_SPM						**1**	1
	ADV_TDS		**1**	**2**	**3**	**4**	**5**	**6**
Guidance documents	AGD_OPE	**1**	1	1	1	1	1	1
	AGD_PRE	**1**	1	1	1	1	1	1
Life-cycle support	ALC_CMC	**1**	**2**	**3**	**4**	4	**5**	5
	ALC_CMS	**1**	**2**	**3**	**4**	**5**	5	5
	ALC_DEL		**1**	1	1	1	1	1
	ALC_DVS			**1**	1	1	**2**	2
	ALC_FLR							
	ALC_LCD			**1**	1	1	1	**2**
	ALC_TAT				**1**	**2**	**3**	3
Security Target evaluation	ASE_CCL	**1**	1	1	1	1	1	1
	ASE_ECD	**1**	1	1	1	1	1	1
	ASE_INT	**1**	1	1	1	1	1	1
	ASE_OBJ	**1**	**2**	2	2	2	2	2
	ASE_REQ	**1**	**2**	2	2	2	2	2
	ASE_SPD		**1**	1	1	1	1	1
	ASE_TSS	**1**	1	1	1	1	1	1
Tests	ATE_COV		**1**	**2**	2	2	**3**	3
	ATE_DPT			**1**	**2**	**3**	3	**4**
	ATE_FUN		**1**	1	1	1	**2**	2
	ATE_IND	**1**	**2**	2	2	2	2	**3**
Vulnerability assessment	AVA_VAN	**1**	**2**	2	**3**	**4**	**5**	5

Table 1 - Evaluation assurance level summary

ISO/IEC 15408-3, clause 7

Evaluation Assurance Levels

- ADV_FSP.1 Basic functional specification
- ADV_FSP.2 Security-enforcing functional specification
- ADV_FSP.3 Functional specification with complete summary
- ADV_FSP.4 Complete functional specification
- ADV_FSP.5 Complete semi-formal functional specification with additional error information
- ADV_FSP.6 Complete semi-formal functional specification with additional formal specification

ADV_FSP.2 – Functional Specification

The ADV_FSP criteria are used to specify the level of formalism in the functional specification of the TOE.

Dependencies:

ADV_RCR.1 Informal correspondence demonstration.

Developer action elements:

ADV_FSP.2.1D The developer shall provide a functional specification.

The DO-178B requires the development of high-level requirements in section 5.1 to produce the documentation specified in section 11.9, “Software Requirements Data”.

[Towards Common Criteria Certification for DO-178B Compliant Airborne Software Systems - Jim Alves-Foss, Bob Rinker and Carol Taylor]

Mapping 15408 - DO-178B

Content and presentation of evidence elements:

ADV_FSP.2.1C The functional specification shall describe the TSF and its external interfaces using an informal style.

The style of specification is unspecified in DO-178B; but is assumed to be informal. However, section 11.6 “Software Requirements Standards” provides for defining this type of requirement. Understand that to be compliant with the CC, you must explicitly specify the security functions; this is similar to the explicit specification of safety requirements as specified in 5.1.2d. Again, these restrictions can be documented in 11.6

ADV_FSP.2.2C The functional specification shall be internally consistent.

This corresponds to DO-178B section 5.1.2a and to section 6.3.1b and is documented in sections 11.9 and 11.14 “Software Verification Results”.

[Towards Common Criteria Certification for DO-178B Compliant Airborne Software Systems - Jim Alves-Foss, Bob Rinker and Carol Taylor]

Using Linux Security Mechanisms

MILS Architecture based on the following mechanisms:

- Hypervisor checks for validity of channels
- IPtables Firewall
- LSM – Linux Security Modules
- FUSE – Userspace Filesystem

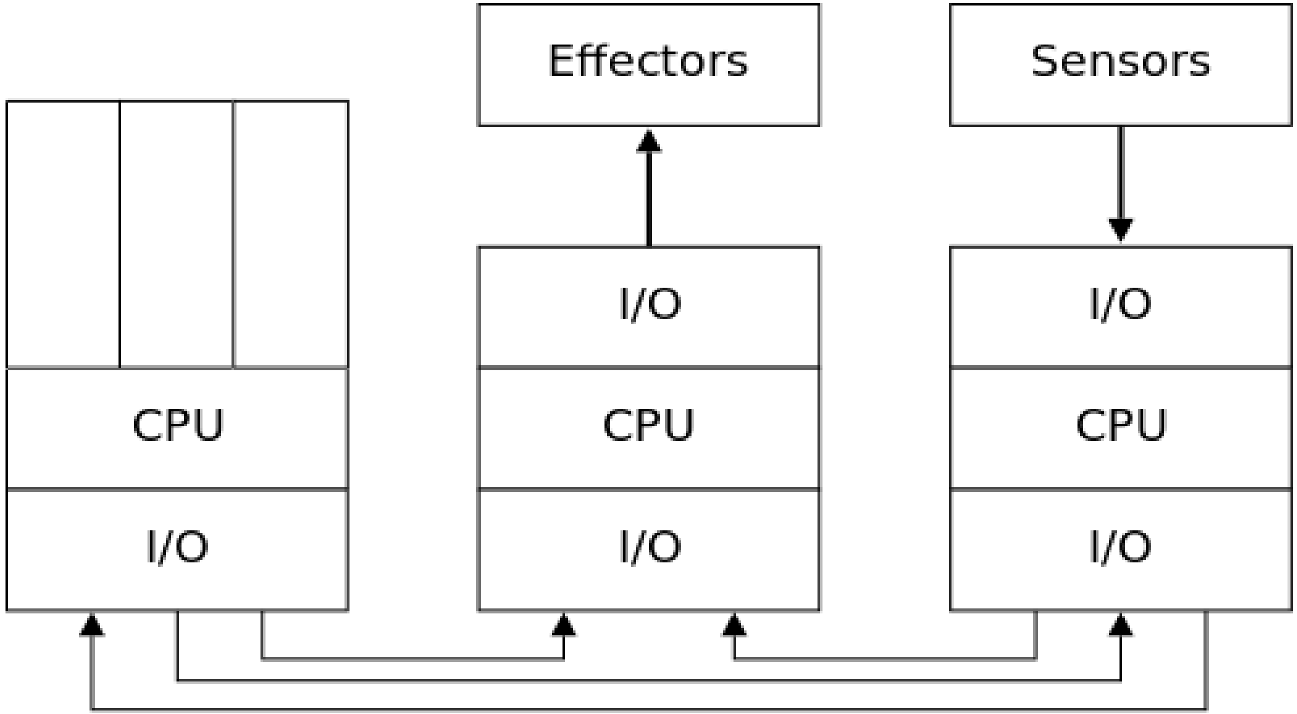
Federated vs. Integrated Architecture

Reasons for an Integrated Architecture

- reduction of hardware nodes
- better utilization of modern CPUs
- savings in power consumption, weight, cooling and costs
- better scalability
- higher flexibility
- reuse of (legacy) software modules

Federated vs. Integrated Architecture

Example: Federated Approach



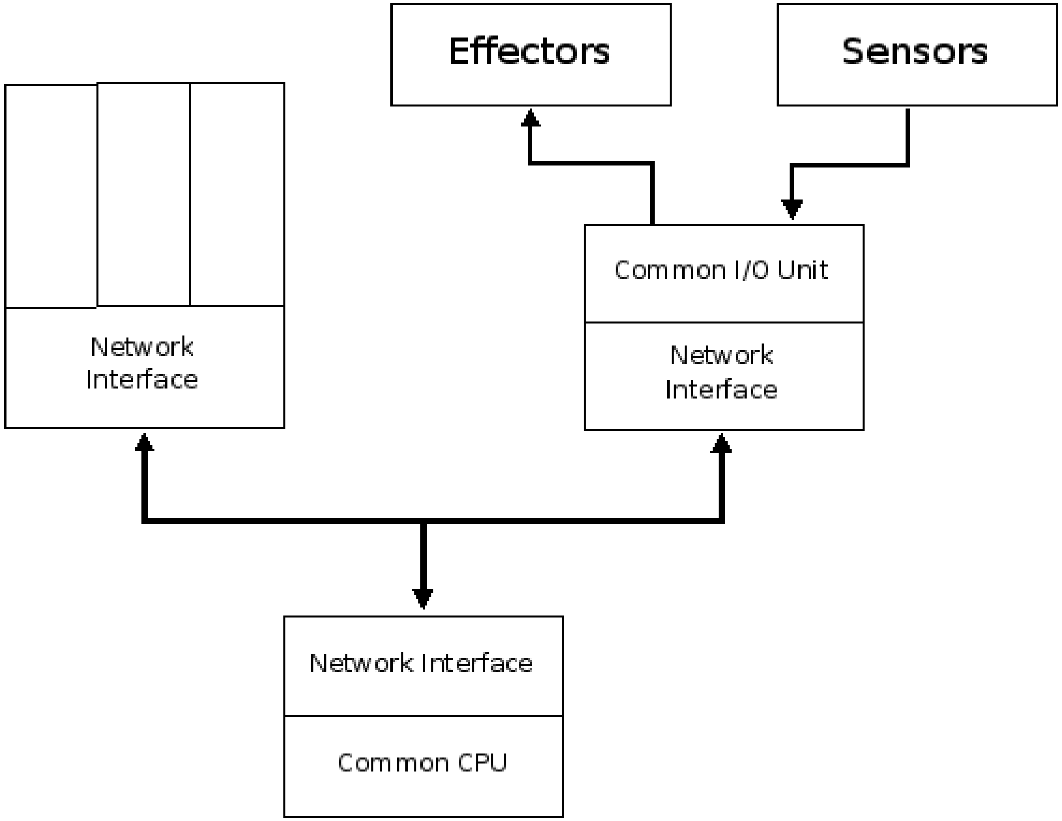
Federated Architecture

CPUs: 3
I/O Modules + Network Interface Modules: 5
Physical Communication Channels: 4

Federated vs. Integrated Architecture

oversee

Example: Integrated Approach



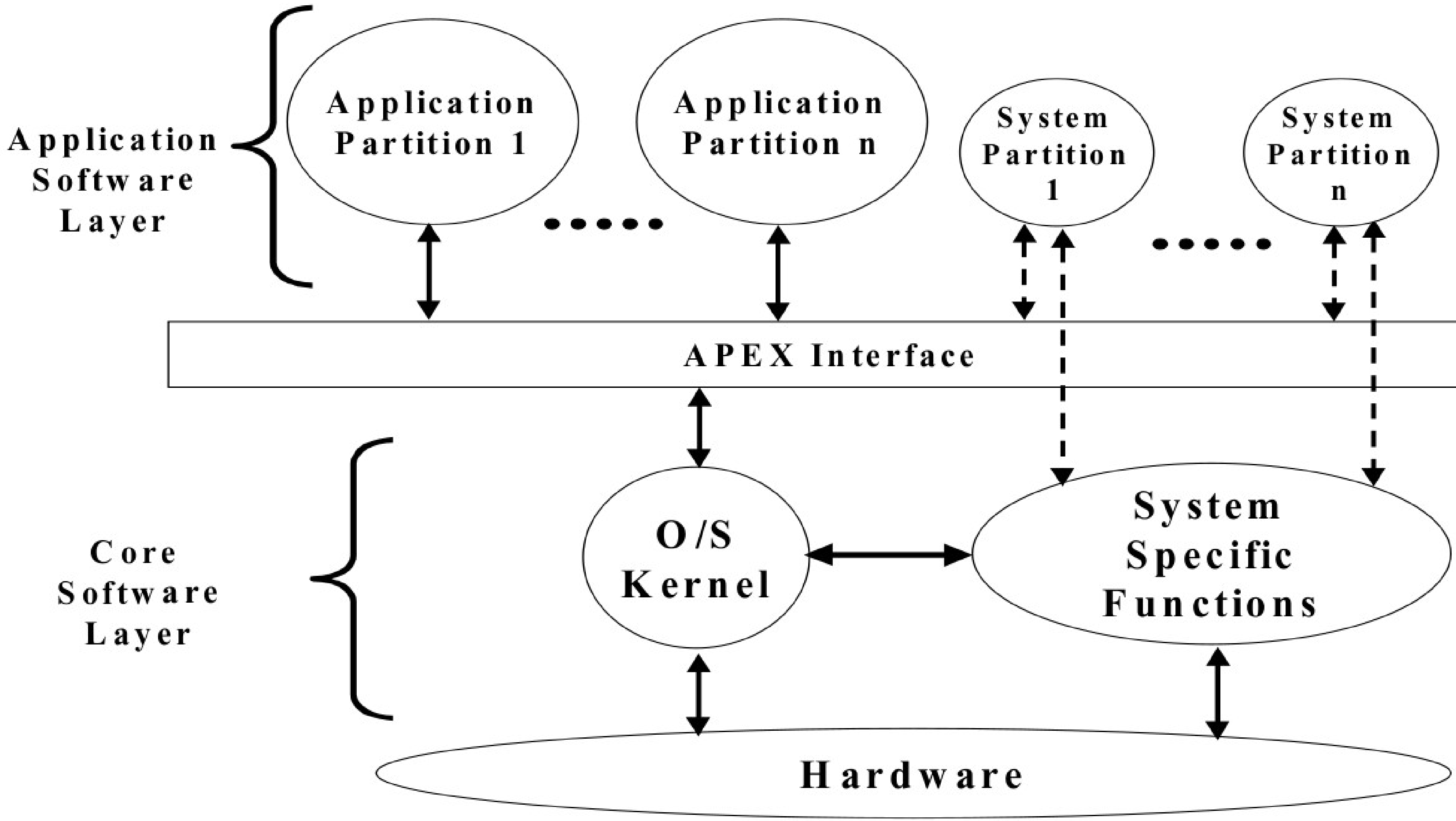
IMA Architecture

CPUs: 1

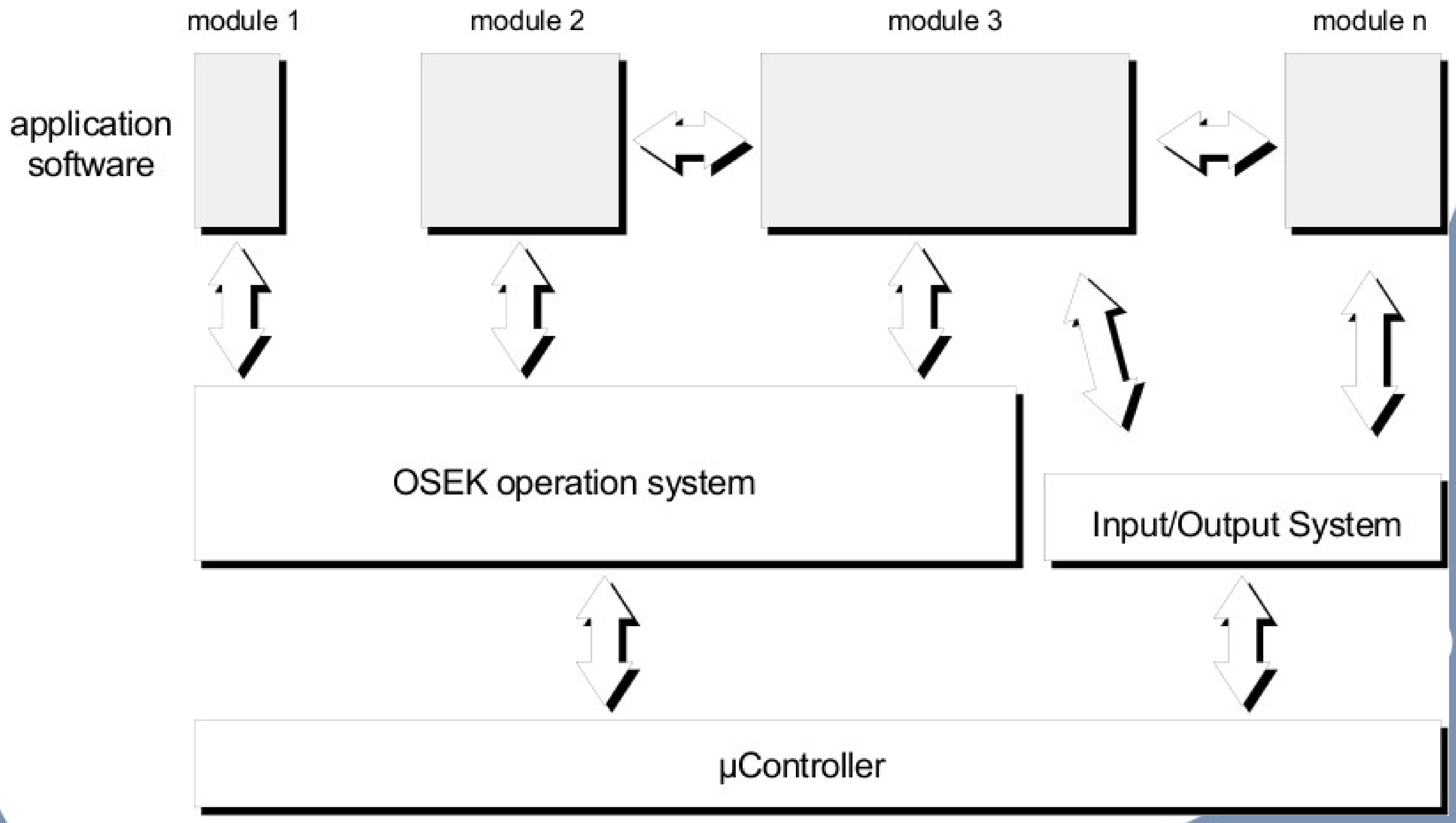
I/O Modules + Network Interface Modules: 4

Physical Communication Channels: 1

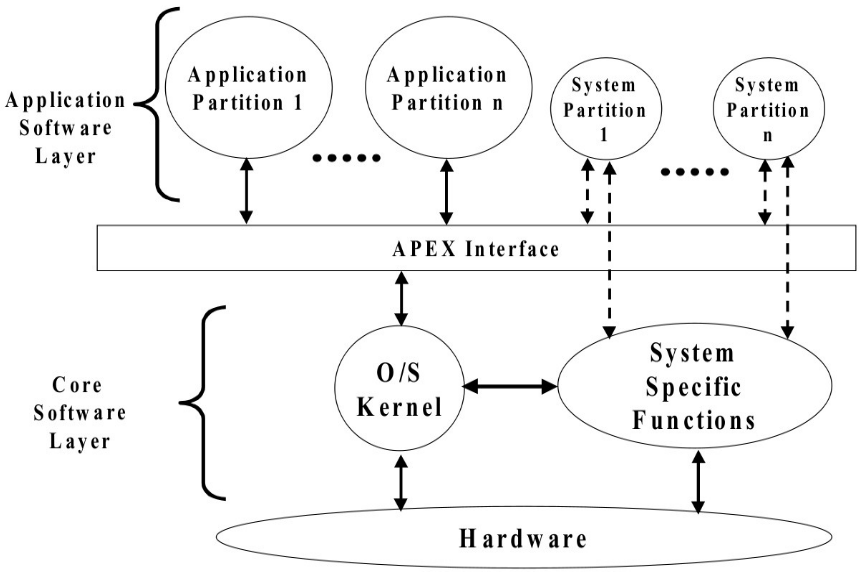
ARINC System Context

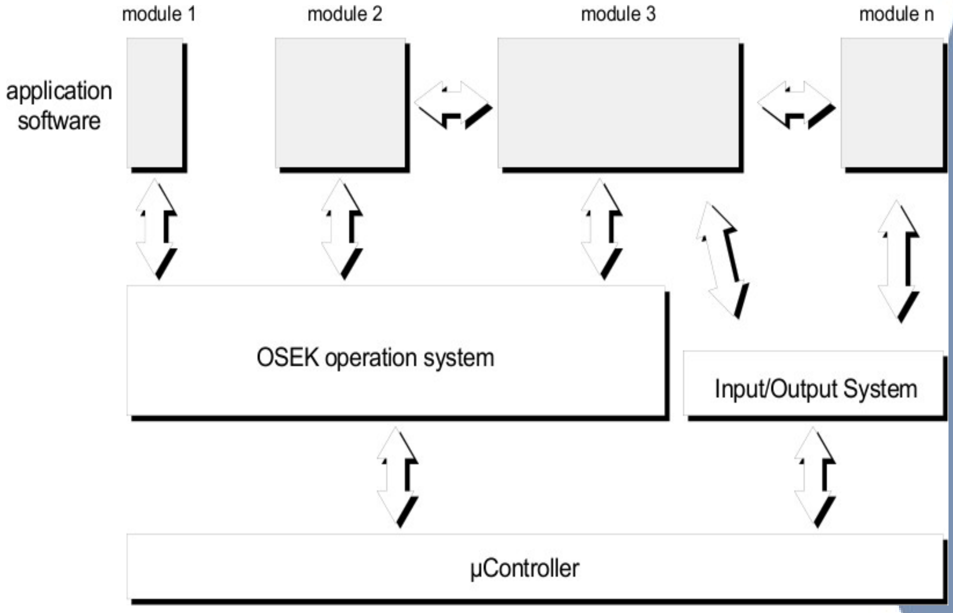


OSEK System Context



ARINC vs. OSEK



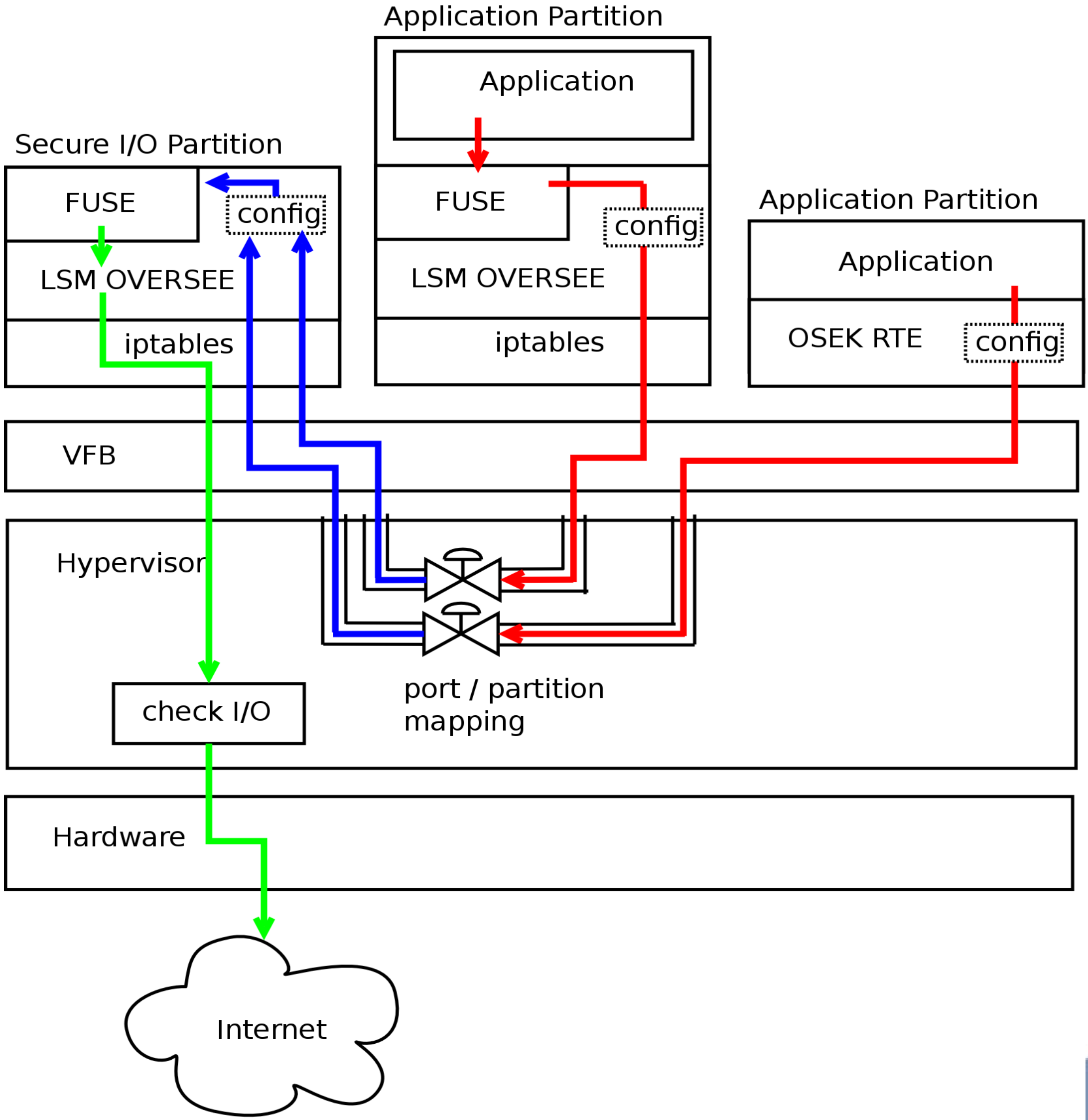


Integrated Approach in Functional Safty Standards

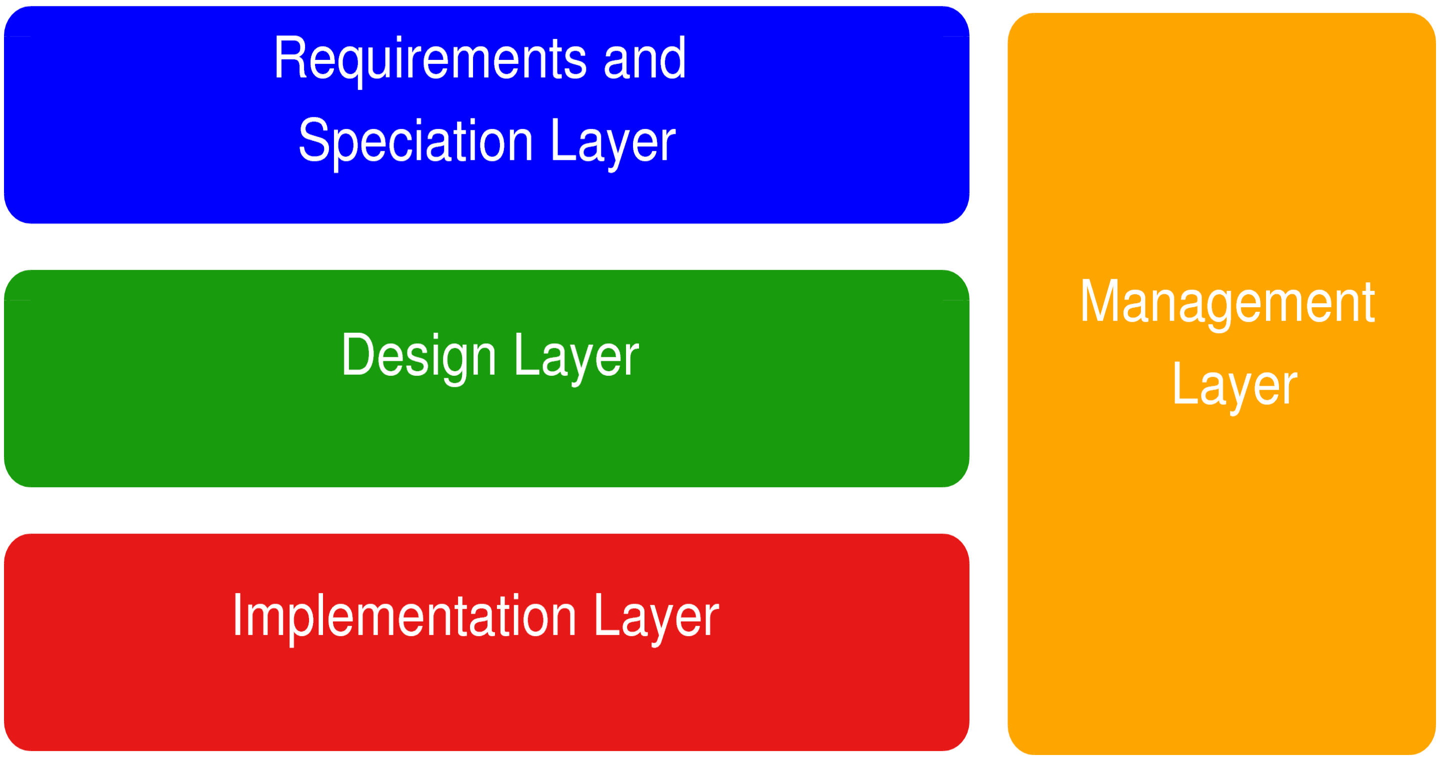
*"Where the application software is to implement safety instrumented functions of different safety integrity levels or Non-safety functions, then all of the software shall be treated as belonging to the highest safety integrity level, **unless independence between the safety instrumented functions of the different safety integrity levels can be shown** in the design. The justification for independence shall be documented.*
Whether independence is claimed or not, the intended SIL of each SIF shall be identified."

[ISO/IEC 61511-1, 12.4.2.5]
[ISO/IEC 61508-3, clause 7.2.4.7 / 7.2.4.8]

OVERSEE Design



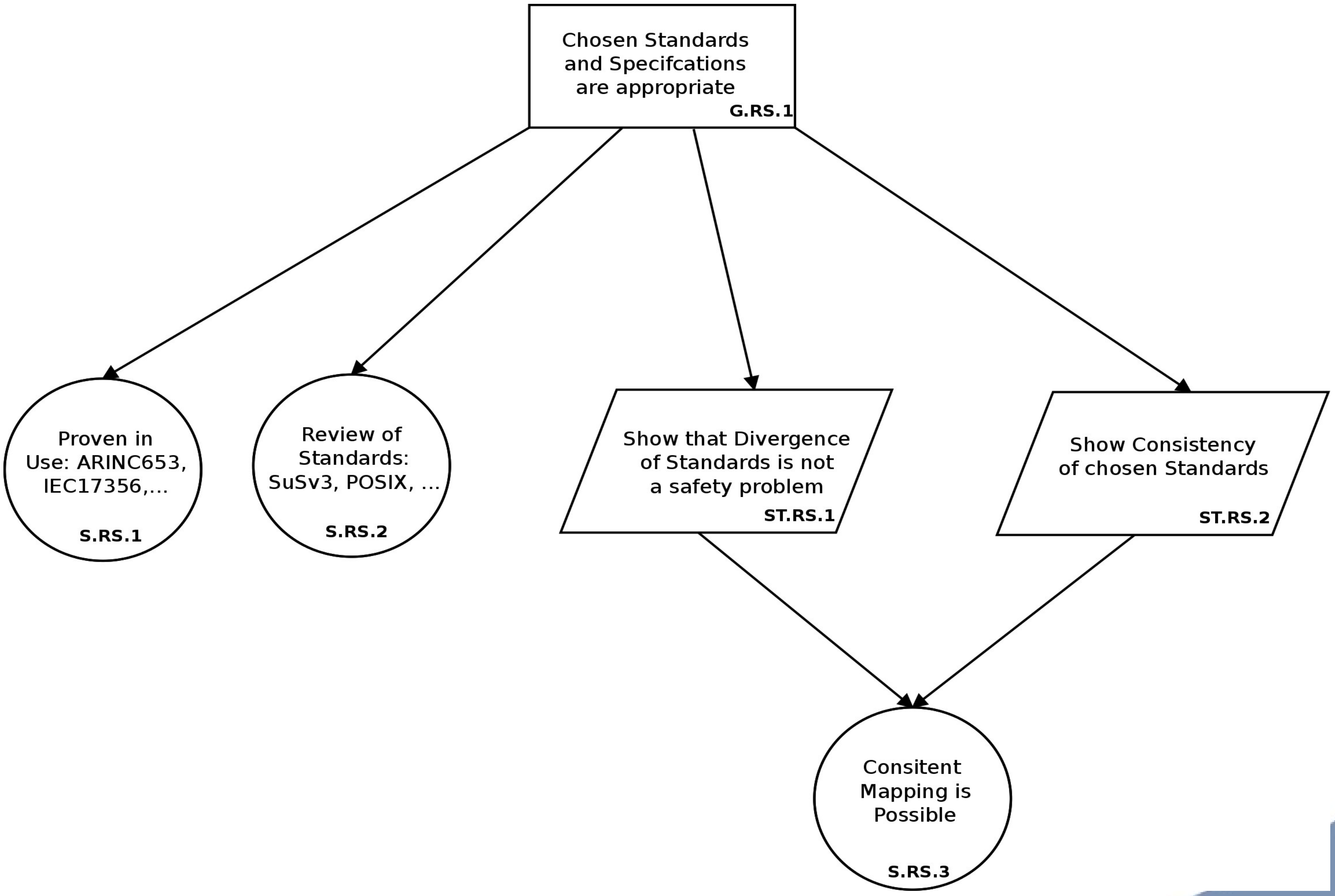
Layered Safety Case



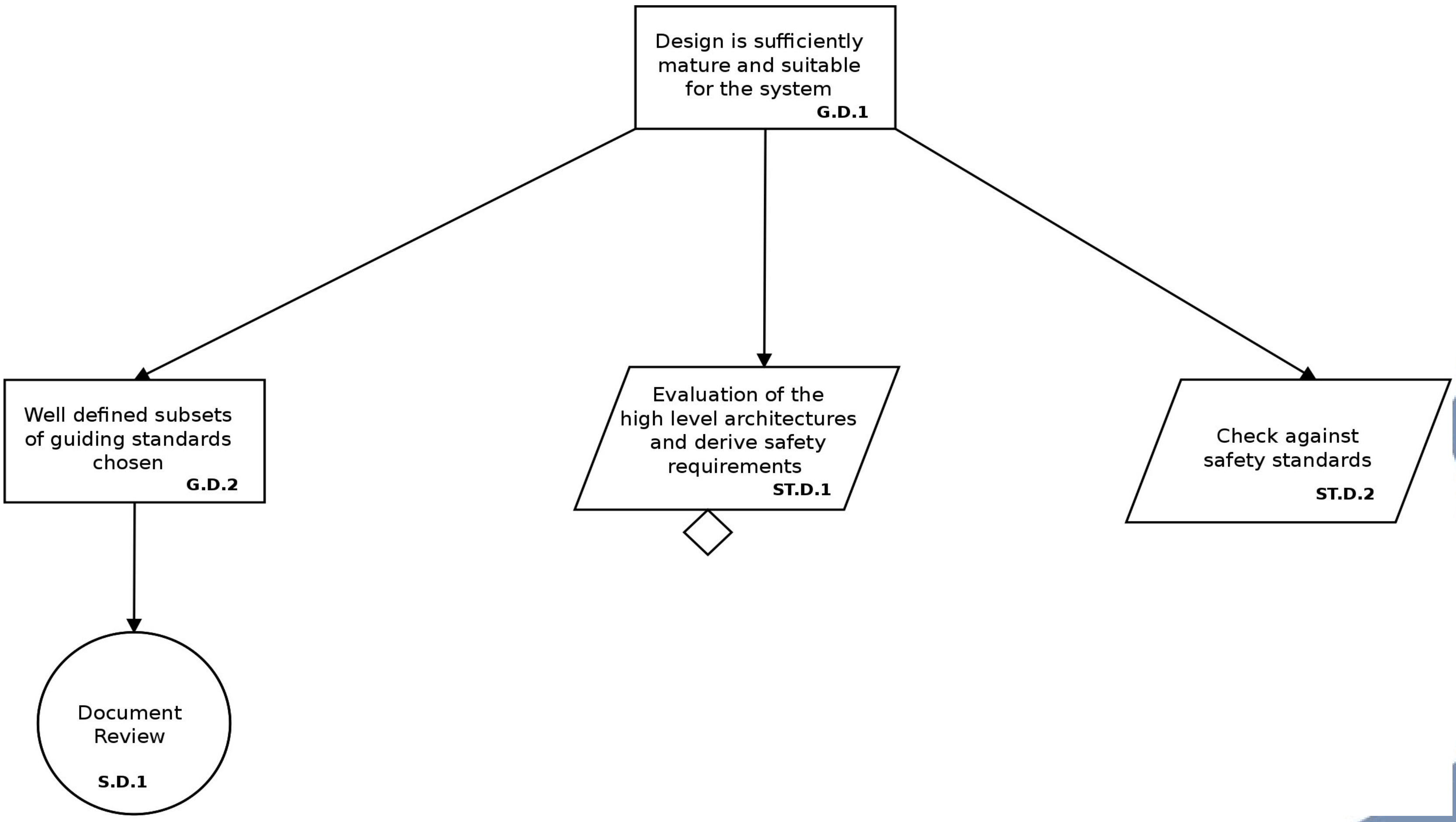
Why Layering the Safety Case?

- easier to handle Complexity
- analytical evidence could be pulled into the upper layers, leaving it untouched in case of a change in the lower layers
- Better Maintenance of the Safety Case
- “Divide and Conquer” - Strategy

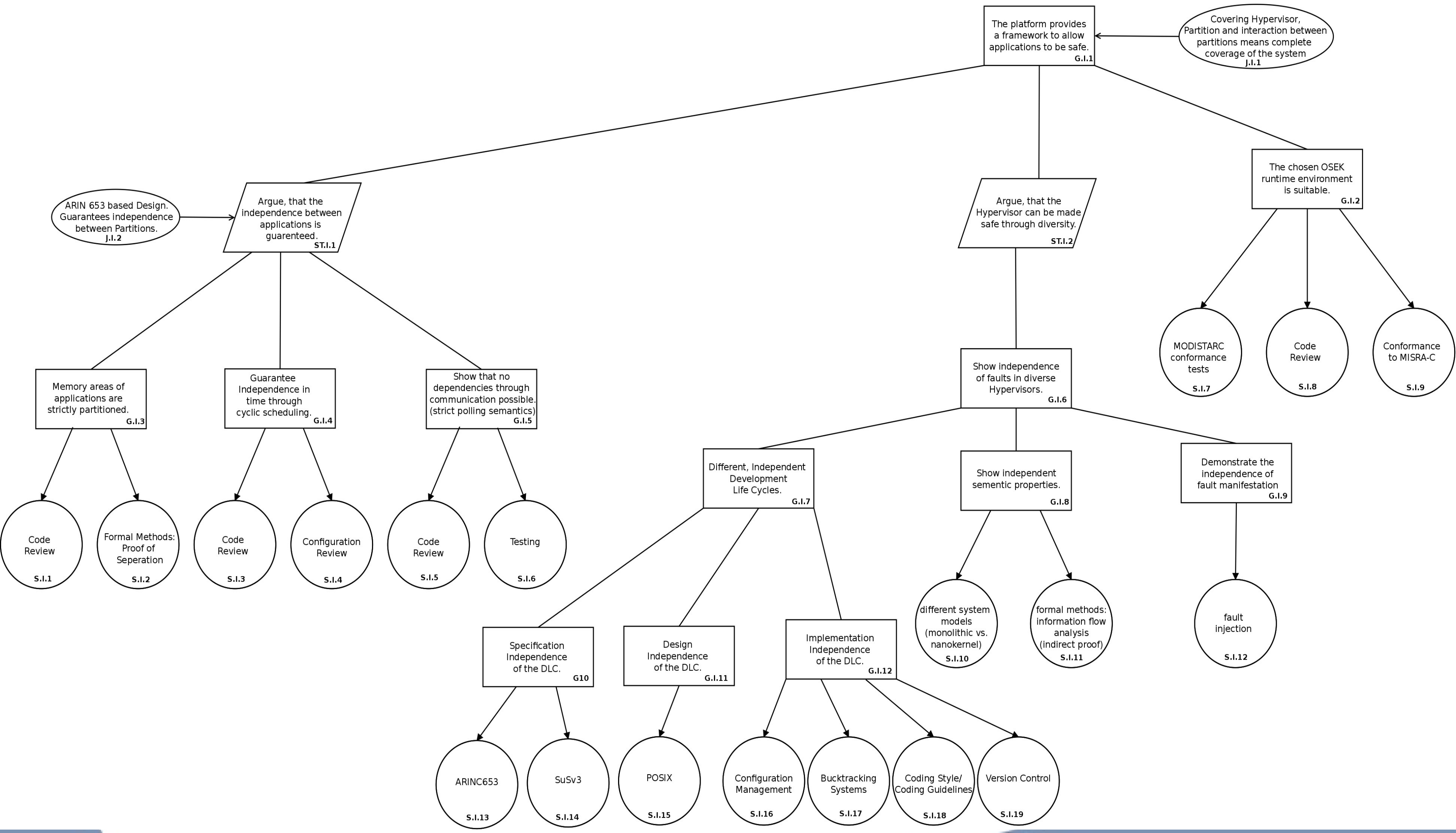
Requirements and Specification Layer



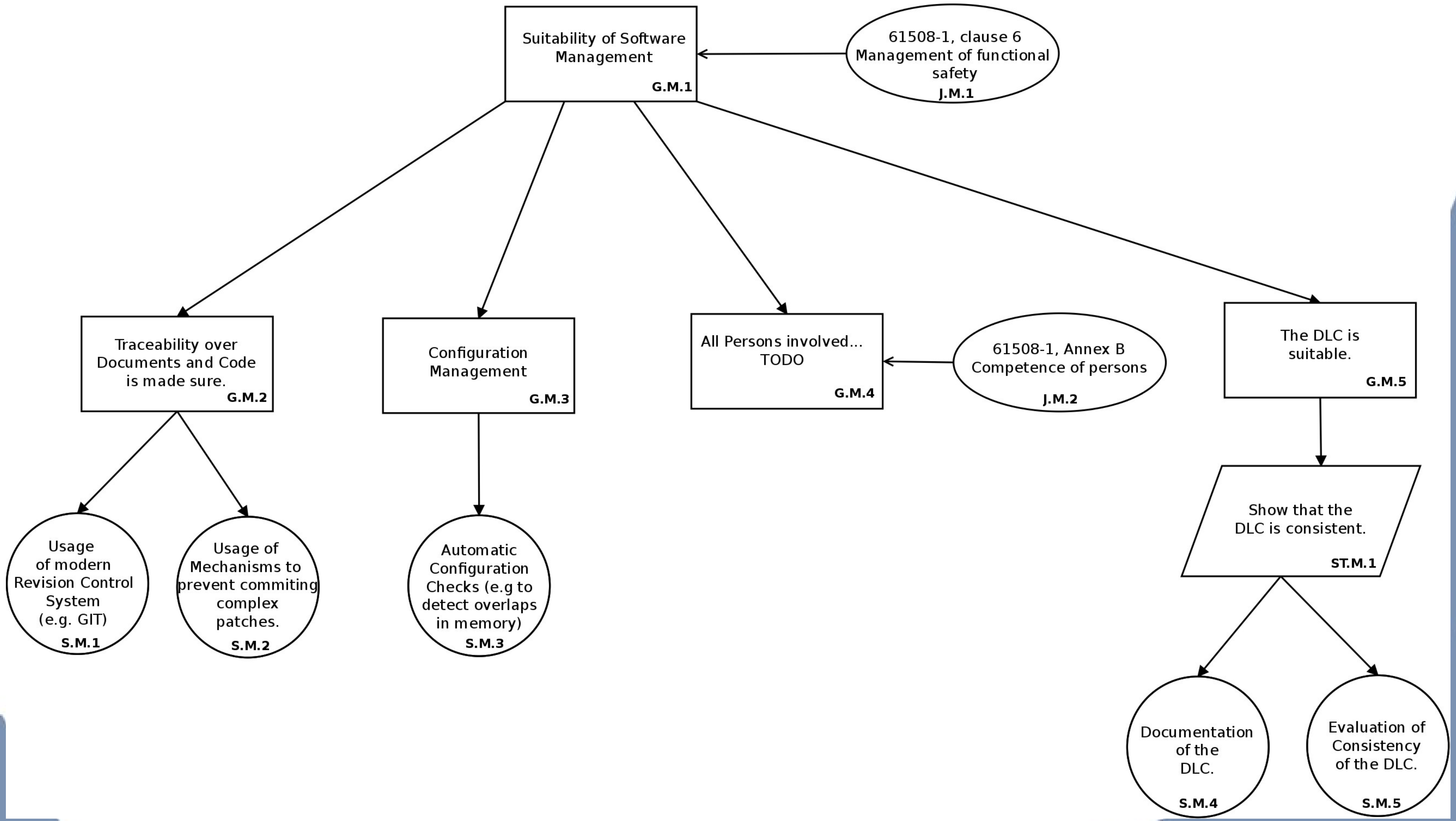
Design Layer



Implementation Layer



Management Layer



References

- [1]Security and Privacy Vulnerabilities of In-Car Wireless Networks: A Tire Pressure Monitoring System Case Study, Ishtiaq Rouf, Rob Miller, et al, 2010

- [2]Experimental Security Analysis of a Modern Automobile, Karl Koscher, Alexei Czeskis et. al, 2010