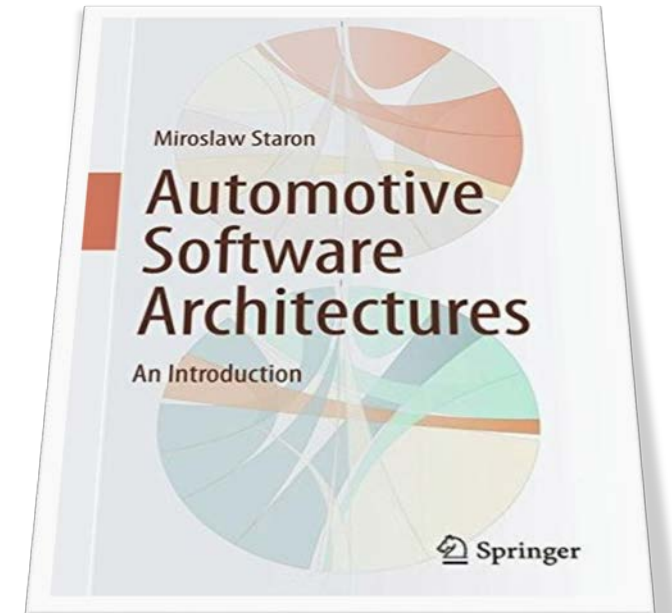




Bridging Functional Safety Analysis and Software Architecture Assessment

Safety scenarios in Architecture Trade-off Analysis Method (ATAM)



Miroslaw Staron

Software Engineering

Computer Science and Engineering

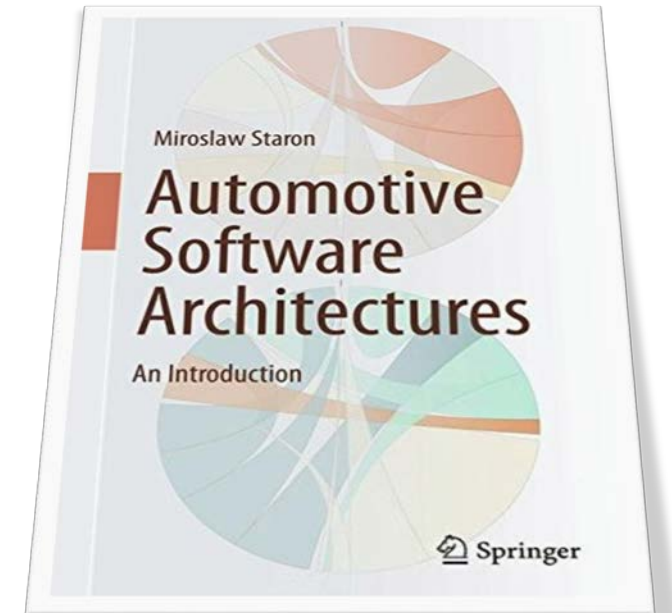
Chalmers | Göteborgs universitet





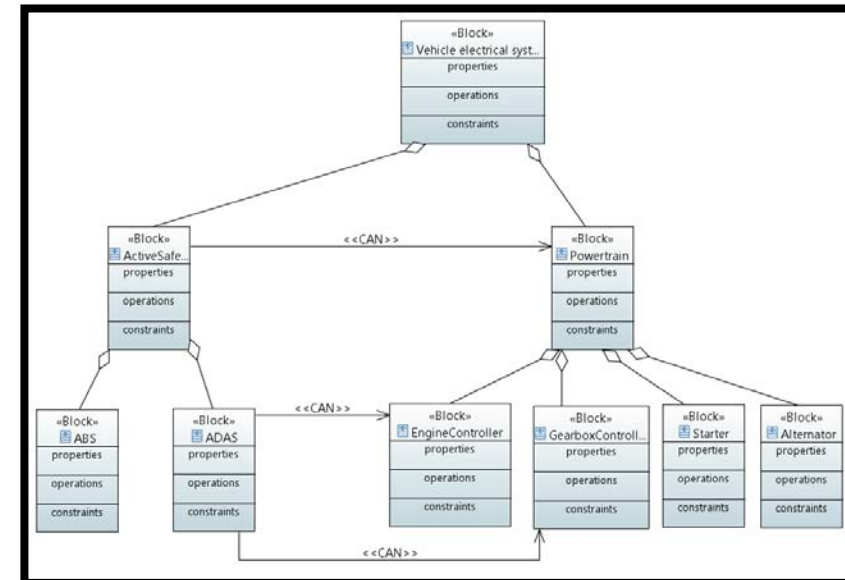
Outline of my talk

- Architecture Trade-off Analysis Method – ATAM
- Example analysis with adding a rearview camera
- Examples of common modifiability scenarios in architecture analysis
- ISO/IEC 26262 safety analysis and its impact on architecture analysis
- New scenarios for safety analysis
- Summary and research outlook



Software architecture and its viewpoints

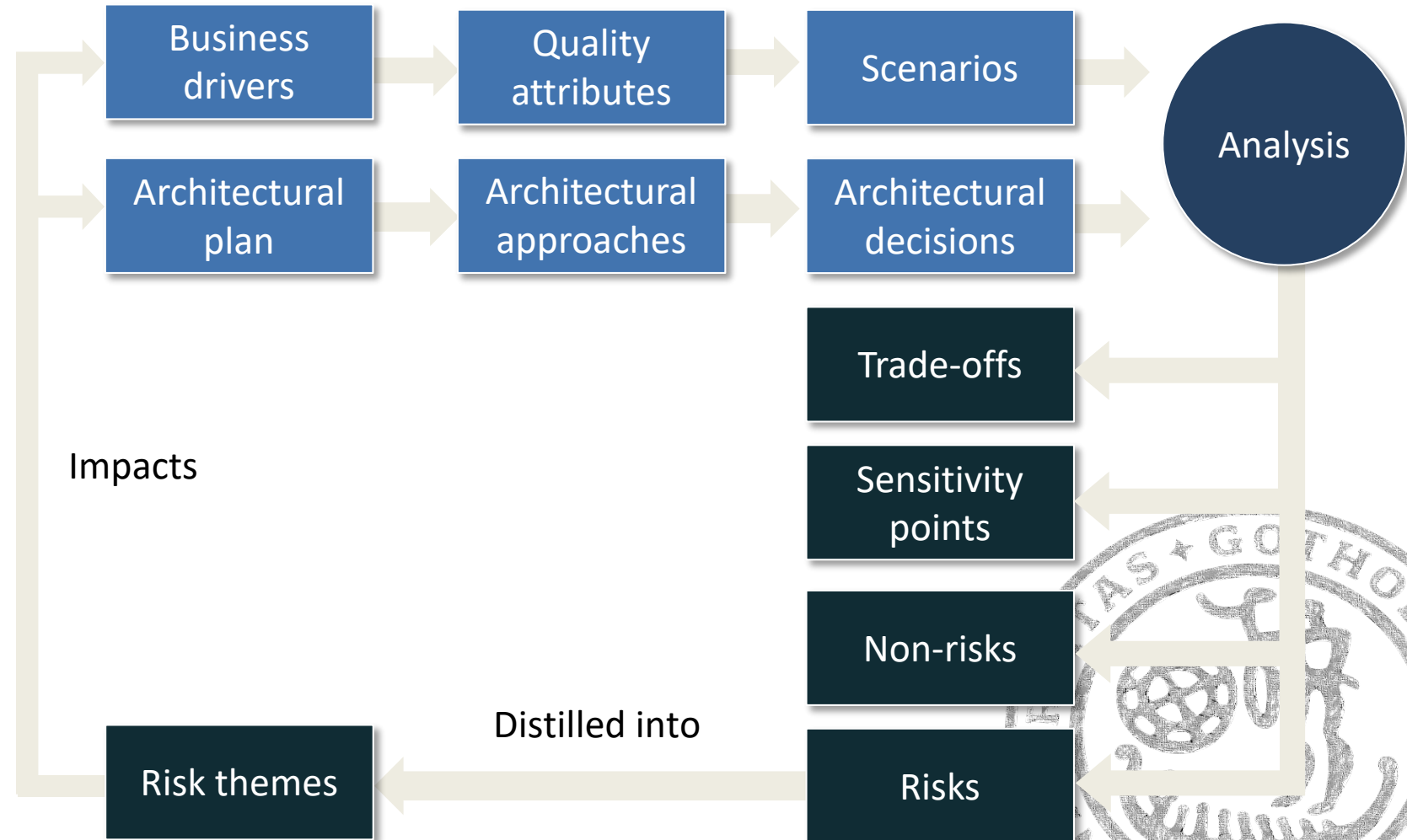
- Software architecture
 - *Software architecture refers to high-level structures of a software system, the discipline of creating such structures, and the documentation of these structures*
- The most common viewpoints
 - Logical viewpoint
 - Software classes, Simulink blocks, source code modules, etc.
 - Physical viewpoint
 - ECUs, buses
 - Deployment viewpoint
 - Execution processes deployed onto ECUs, signals on buses
 - Functional viewpoint
 - Features and functions



ATAM

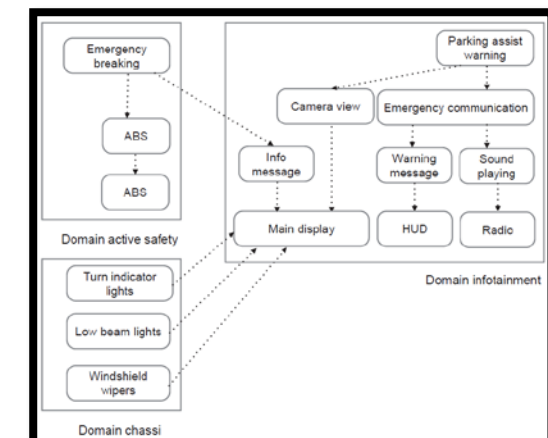
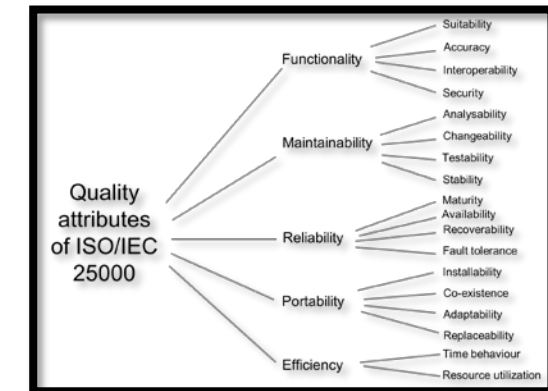
- Architecture Trade-off Analysis Method

- Addresses the question *How good is my architecture?*
- Evaluates the architecture from the perspective of **quality attributes** to identify **risks** and the related **sensitivity points**



ATAM process has eight steps

- Present ATAM
- Present business drivers
- Present architecture
- Identify architectural approaches
- Generate quality attribute utility tree
- Analyze architectural approaches
- Brainstorm and prioritize scenarios
- Present results





MOTIVATIONAL EXAMPLE

THE IMPACT OF ADDING A REAR CAMERA ON THE SAFETY OF THE ELECTRICAL SYSTEM

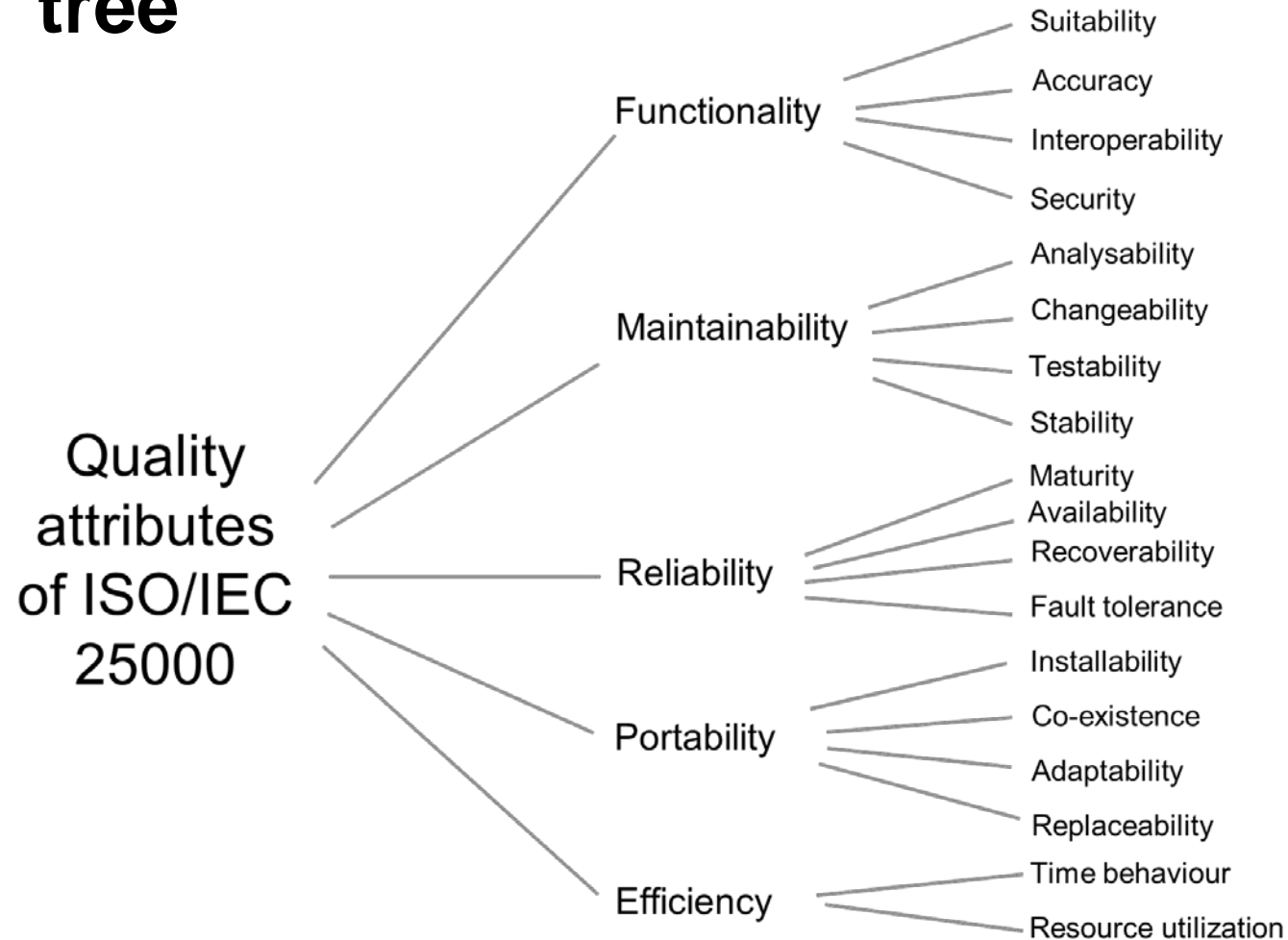


Business drivers

- The car's electrical system should support the advanced mechanisms of active safety (i.e. controlled by software) and should assure that none of the mechanisms interferes with another one, jeopardizing the safety.
- Main characters in this play
 - Electrical system
 - Active safety
 - Interference



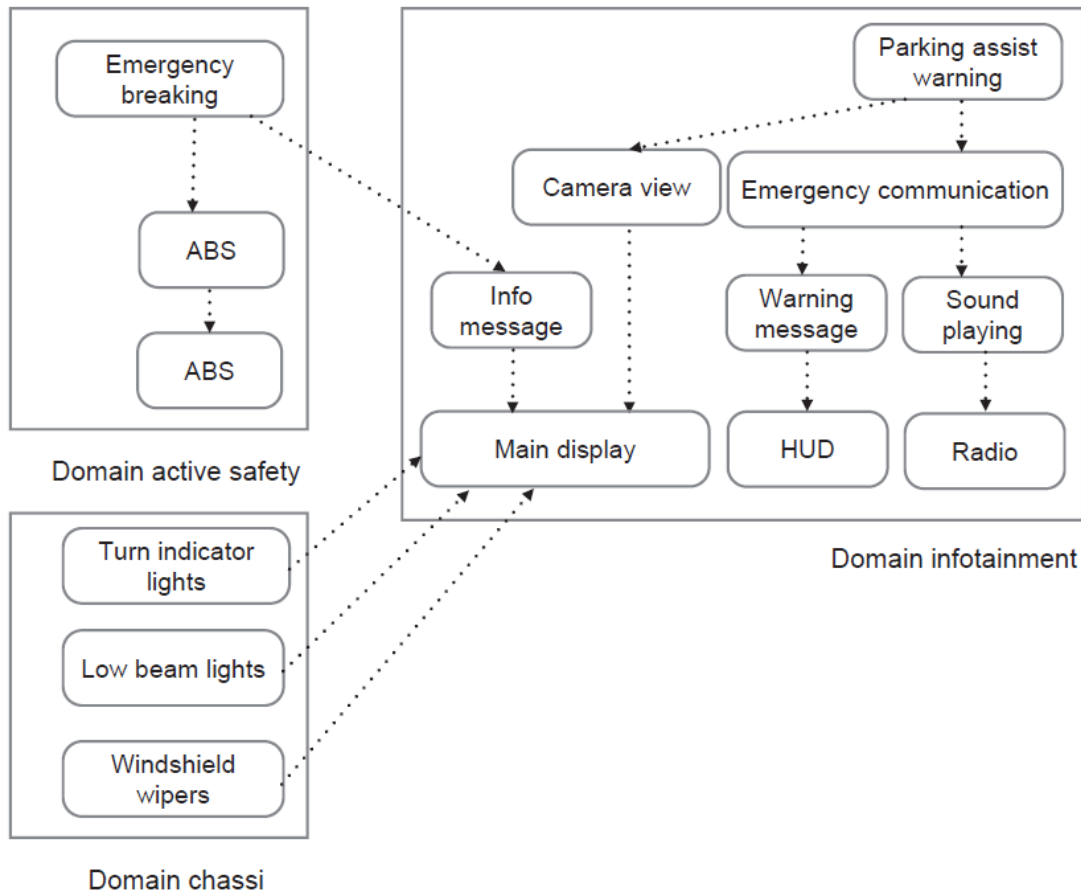
Where things can go wrong: relevant quality attributes tree



Focus on today's talk:

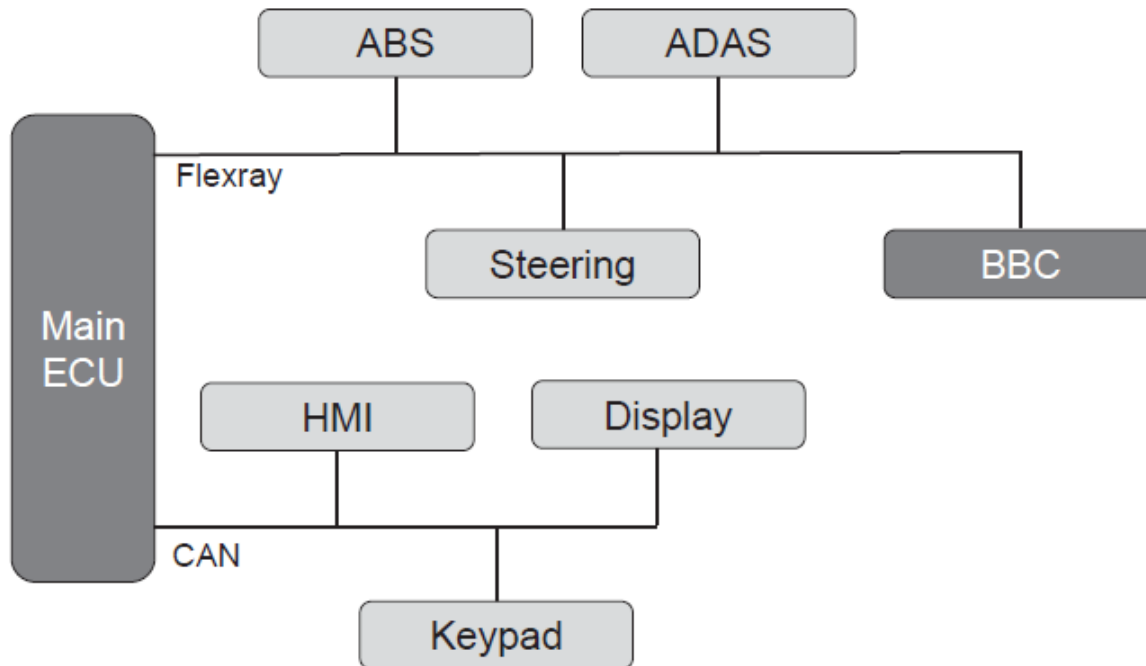
Adding safety as a quality attribute

Functional architecture – how functions depend on one another

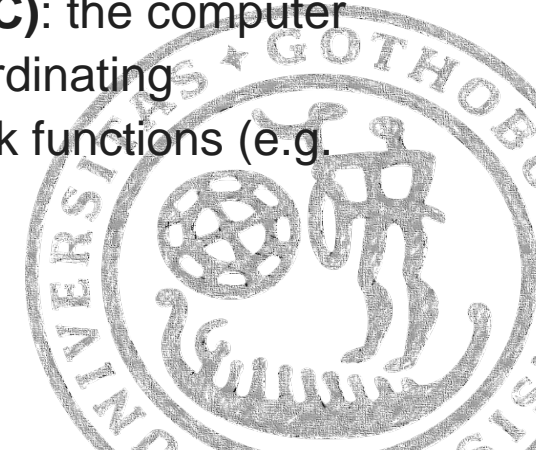


This view helps us to overview functions which are available in our product line

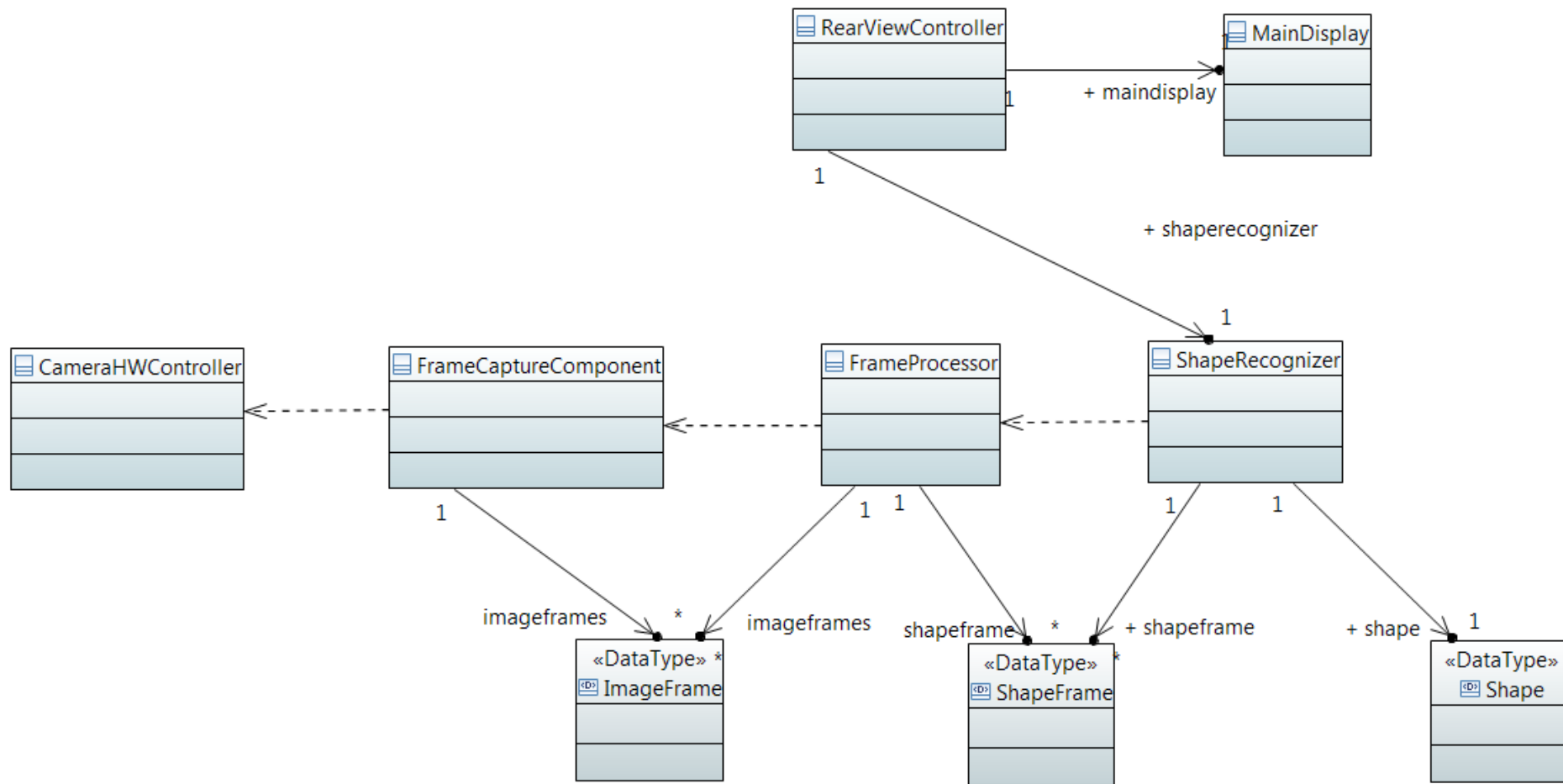
Physical architecture – which computers we can use



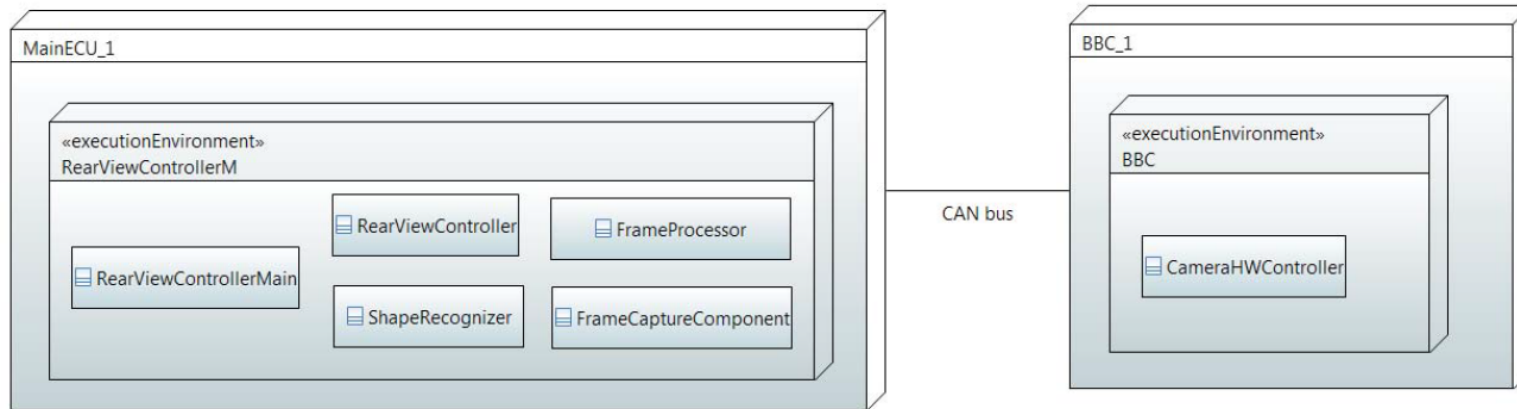
- **Main ECU:** the main computer of the car, controlling the configuration of the car, initialization of the electronics and diagnostics of the entire system. The main ECU has the most powerful computing unit in the car with the largest memory
- **Back Body Controller (BBC):** the computer which is responsible for coordinating functions controlling the back functions (e.g. stop lights)



Logical architecture – which software components are active

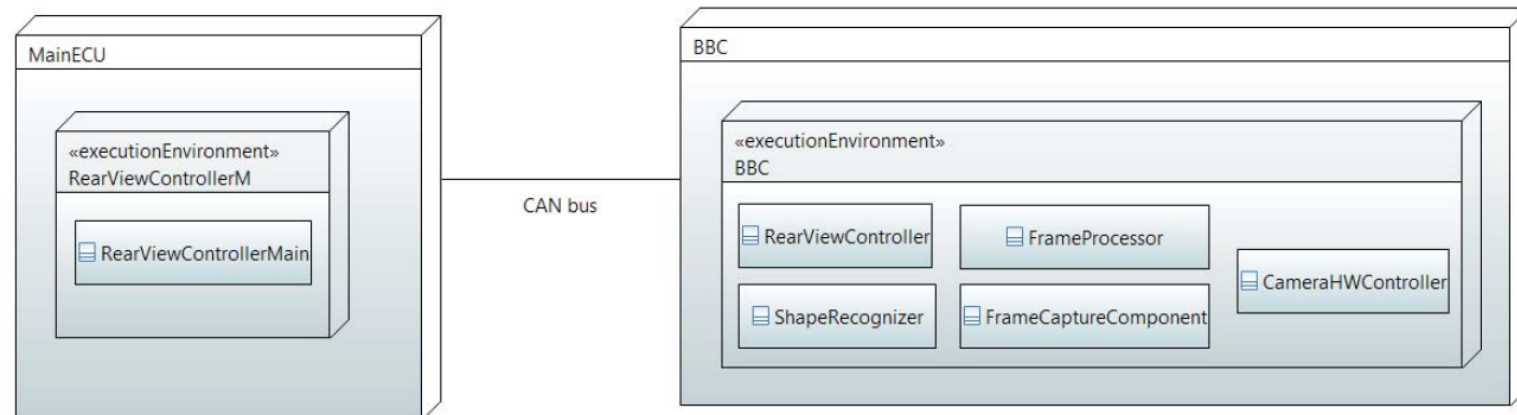


Two different architectural approaches for adding the rear camera



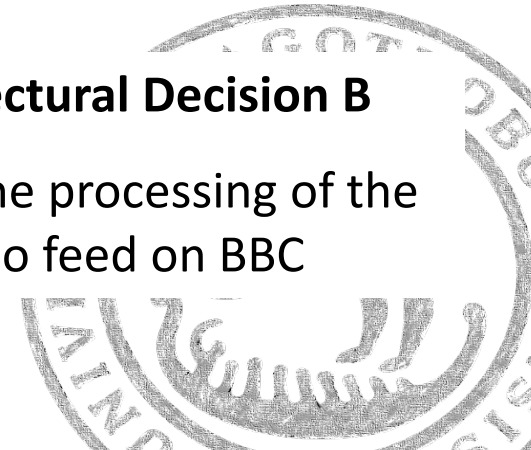
Architectural Decision A

Placing the processing of the video feed on the Main ECU

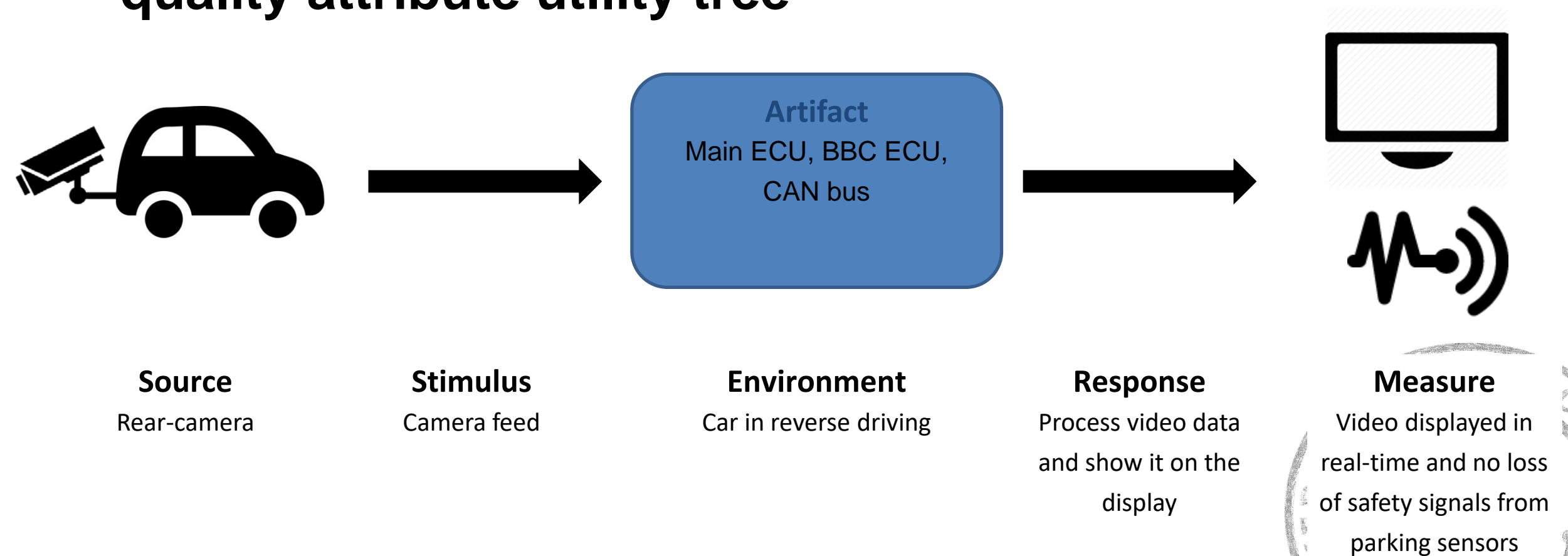


Architectural Decision B

Placing the processing of the video feed on BBC



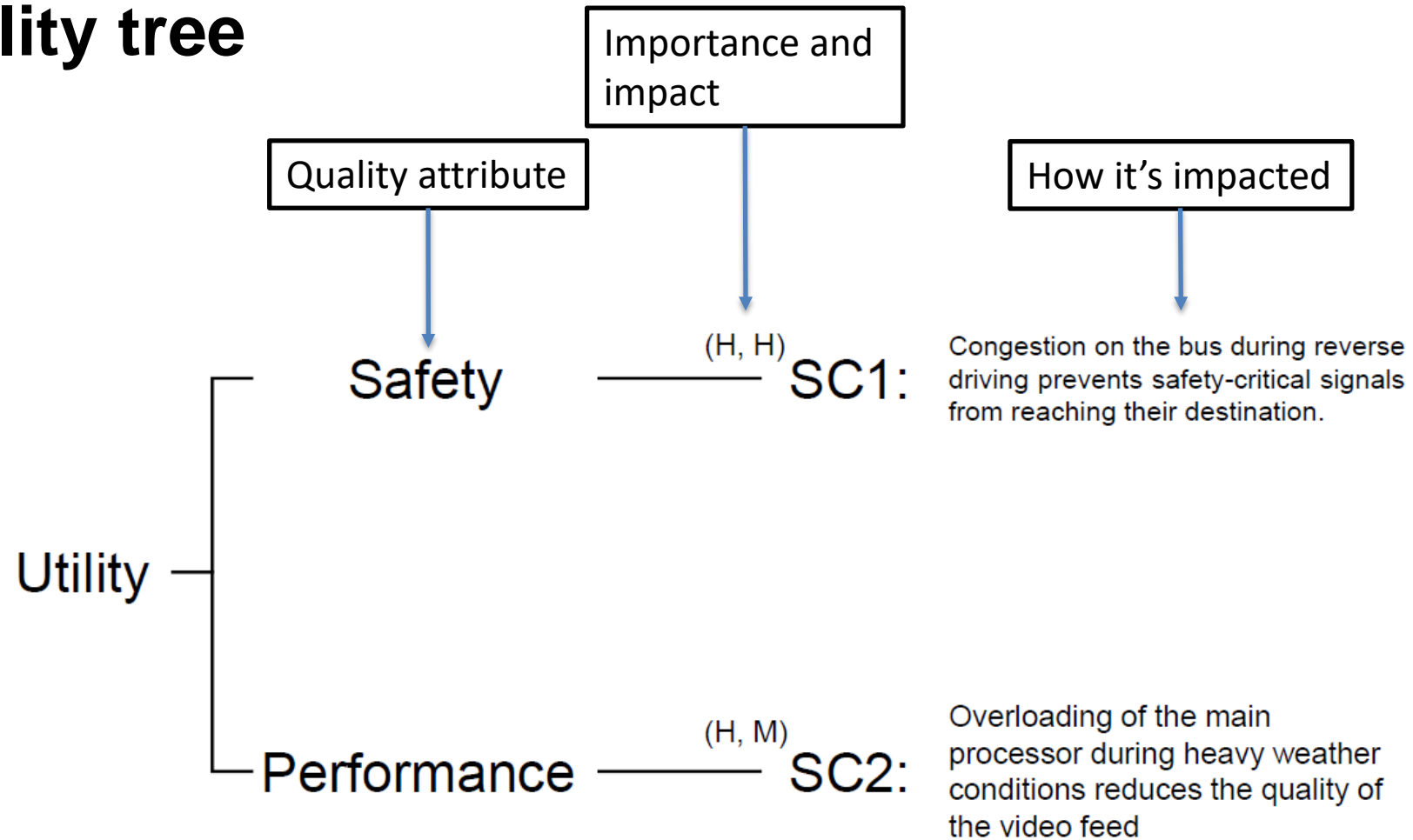
Identifying the relevant quality attributes – generating quality attribute utility tree





Quality attribute utility tree

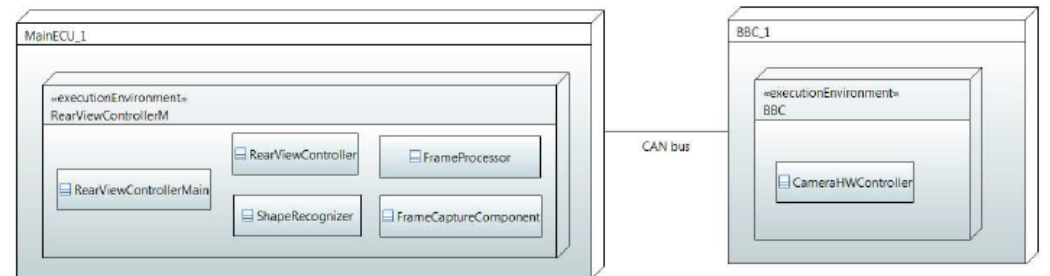
- Quality attributes take part in our trade-off
- Once we know that we can start brainstorming about their importance and impact
 - On business drivers
 - On quality attributes





The trade-off

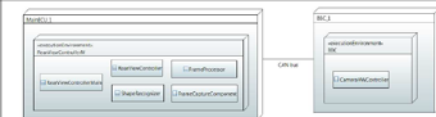
- Brainstorming and the second analysis lead to the identification of
 - Attributes
 - Stimulus
 - Trade-offs
 - Risks
 - Sensitivity points

Scenario 5	Capture video during the reverse driving (backing up) of the car from the rear-camera and show it on the main display.		
Attributes	Safety.		
Environment	Car in reverse driving.		
Stimulus	Camera feed to be shown on the display.		
Response	Process video data and show it on the display.		
Architectural decisions	Sensitivity	Trade-off	Risk
Placing the processing of the video feed on the Main ECU	S1	T1	R1
Placing the processing of the video feed on BBC		T2	R2
Reasoning	<p>The functioning of the main ECU is vital to the system (see sensitivity point S1)</p> <p>Safety versus lowered cost (see trade-off point T1)</p> <p>Safety requirement might be at risk due to heavy processing on Main ECU (see risk R1)</p>		
Architecture diagram			

Summarizing the ATAM example allows to introduce new scenarios

- In the example we focused on the modifiability
→ we could focus on reliability, security, ...
- Safety was implicit → could be explicit
- The summary shows a good way to put together an argument
 - Could be used in ISO/IEC 15939 argumentation if used correctly



Scenario 5	Capture video during the reverse driving (backing up) of the car from the rear-camera and show it on the main display.		
Attributes	Safety.		
Environment	Car in reverse driving.		
Stimulus	Camera feed to be shown on the display.		
Response	Process video data and show it on the display.		
Architectural decisions	Sensitivity	Trade-off	Risk
Placing the processing of the video feed on the Main ECU	S1	T1	R1
Placing the processing of the video feed on BBC		T2	R2
Reasoning	The functioning of the main ECU is vital to the system (see sensitivity point S1) Safety versus lowered cost (see trade-off point T1) Safety requirement might be at risk due to heavy processing on Main ECU (see risk R1)		
Architecture diagram			

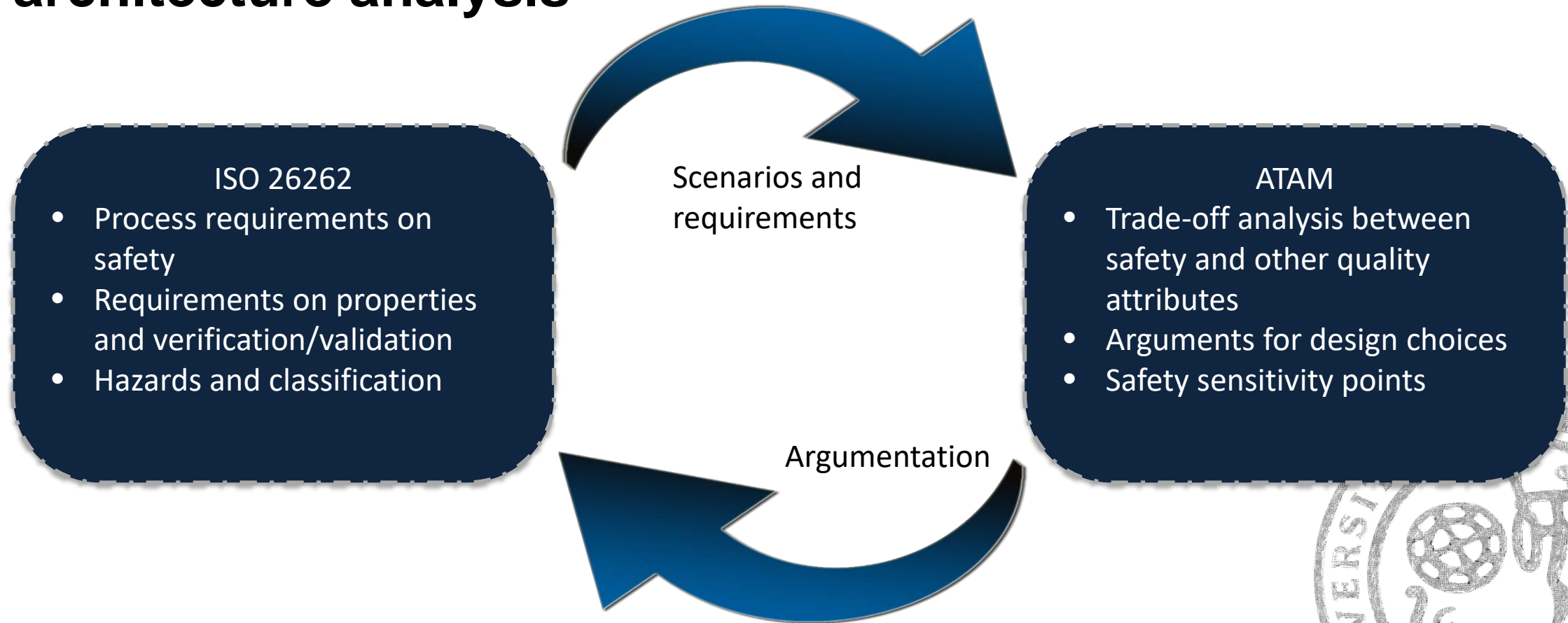


Modifiability scenarios used in ATAM

- Scenario 1: A request arrives to change the functionality of the system.
 - The change can be to add new functionality, to modify existing functionality, or to delete functionality
- Scenario 2: A request arrives to change one of the components (e.g. because of a technology shift)
 - The scenario needs to consider the change propagation to the other components.
- Scenario 3: Customer wants different systems with different capabilities but using the same software
 - Therefore advanced variability has to be built into the system.
- Scenario 4: New emission laws
 - The constantly changing environmental laws require adaptation of the system to decrease its environmental impact.
- Scenario 5: Simpler engine models
 - Replace the engine models in the software with simple heuristics for the low-cost market.



ISO/IEC 26262 safety analysis and its impact on architecture analysis





Software architecture in ISO 26262

- Notation
 - Formal – informal

- Principles
 - Hierarchical
 - Restricted size
 - ...

- Code/control flow complexity
 - Algorithms, state machines, block diagrams

Table 3 — Principles for software architectural design

Methods		ASIL			
		A	B	C	D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components ^a	++	++	++	++
1c	Restricted size of interfaces ^a	+	+	+	+
1d	High cohesion within each software component ^b	+	++	++	++
1e	Restricted coupling between software components ^{a, b, c}	+	++	++	++
1f	Appropriate scheduling properties	++	++	++	++
1g	Restricted use of interrupts ^{a, d}	+	+	+	++

^a In methods 1b, 1c, 1e and 1g "restricted" means to minimize in balance with other design considerations.

^b Methods 1d and 1e can, for example, be achieved by separation of concerns which refers to the ability to identify, encapsulate, and manipulate those parts of software that are relevant to a particular concept, goal, task, or purpose.

^c Method 1e addresses the limitation of the external coupling of software components.

^d Any interrupts used have to be priority-based.

Table 4 — Mechanisms for error detection at the software architectural level

Methods		ASIL			
		A	B	C	D
1a	Range checks of input and output data	++	++	++	++
1b	Plausibility check ^a	+	+	+	++
1c	Detection of data errors ^b	+	+	+	+
1d	External monitoring facility ^c	o	+	+	++
1e	Control flow monitoring	o	+	++	++
1f	Diverse software design	o	o	+	++

^a Plausibility checks can include using a reference model of the desired behaviour, assertion checks, or comparing signals from different sources.

^b Types of methods that may be used to detect data errors include error detecting codes and multiple data storage.

^c An external monitoring facility can be, for example, an ASIC or another software element performing a watchdog function.



Ways of bridging safety and ATAM

- Introduce safety scenarios to ATAM analysis
 - Use hazard analysis techniques to generate the scenarios
- Introduce ATAM trade-offs into the safety argumentation
 - Use the items from tables 3 and 4, Chapter 6, ISO 26262
 - Add these items to the ATAM templates, e.g. sensitivity point description
- Introduce safety properties explicitly into every quality attributes utility tree
 - Hierarchical structure of software components
 - ...





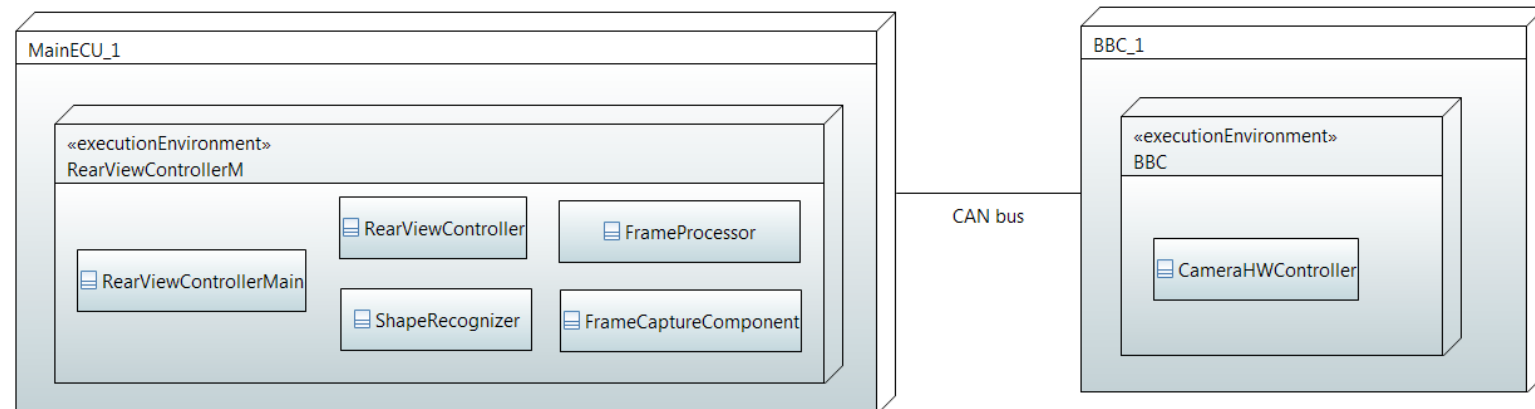
Examples of new scenarios for safety analysis

- Scenario 1:
 - A component's ASIL level is raised from ASIL C to ASIL D:
 - How will this affect the design of the system?
 - Which new checks have to be done?
- Scenario 2:
 - External monitoring facility needs to be added to a component
 - How will this affect the functionality?
- Scenario 3:
 - Increased autonomous driving level from 3 to 4 NHSTA:
 - Level 3: The driver can fully cede control of all safety-critical functions in certain conditions
 - The car senses when conditions require the driver to retake control and provides a "sufficiently comfortable transition time" for the driver to do so.
 - Level 4: The vehicle performs all safety-critical functions for the entire trip, with the driver not expected to control the vehicle at any time.



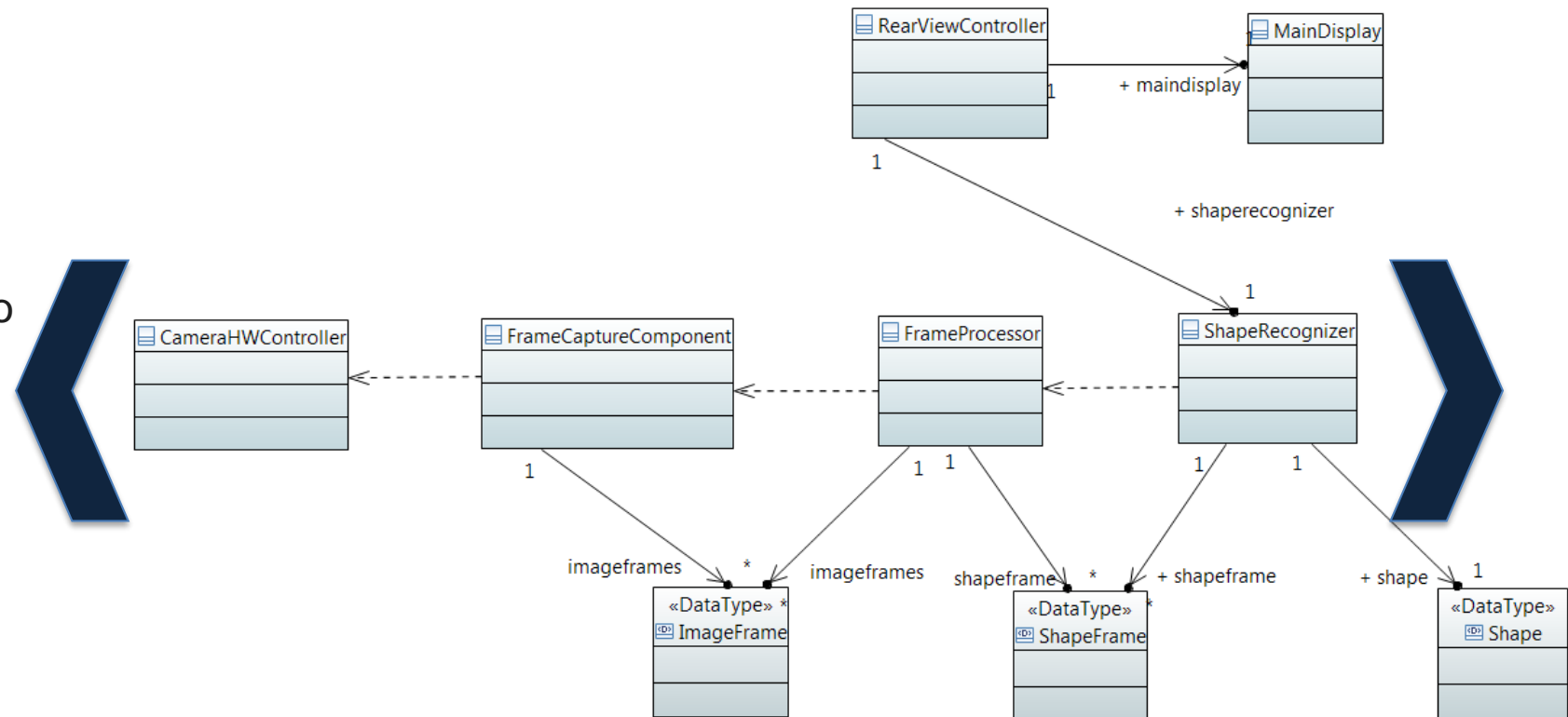
Scenario 1: example

- MainECU_1 – since the camera feed is safety critical with potentially high impact (ASIL D) we need to raise the ASIL level of MainECU_1 to ASIL D
 - New sub-scenarios:
 - restricted use of interrupts
 - plausability checks
 - Sensitivity point 1: execution environment
 - Risk 1: camera feed can take over all processing power (no interrupts)
 - Trade-off 1: place the camera feed processing on BBC_1



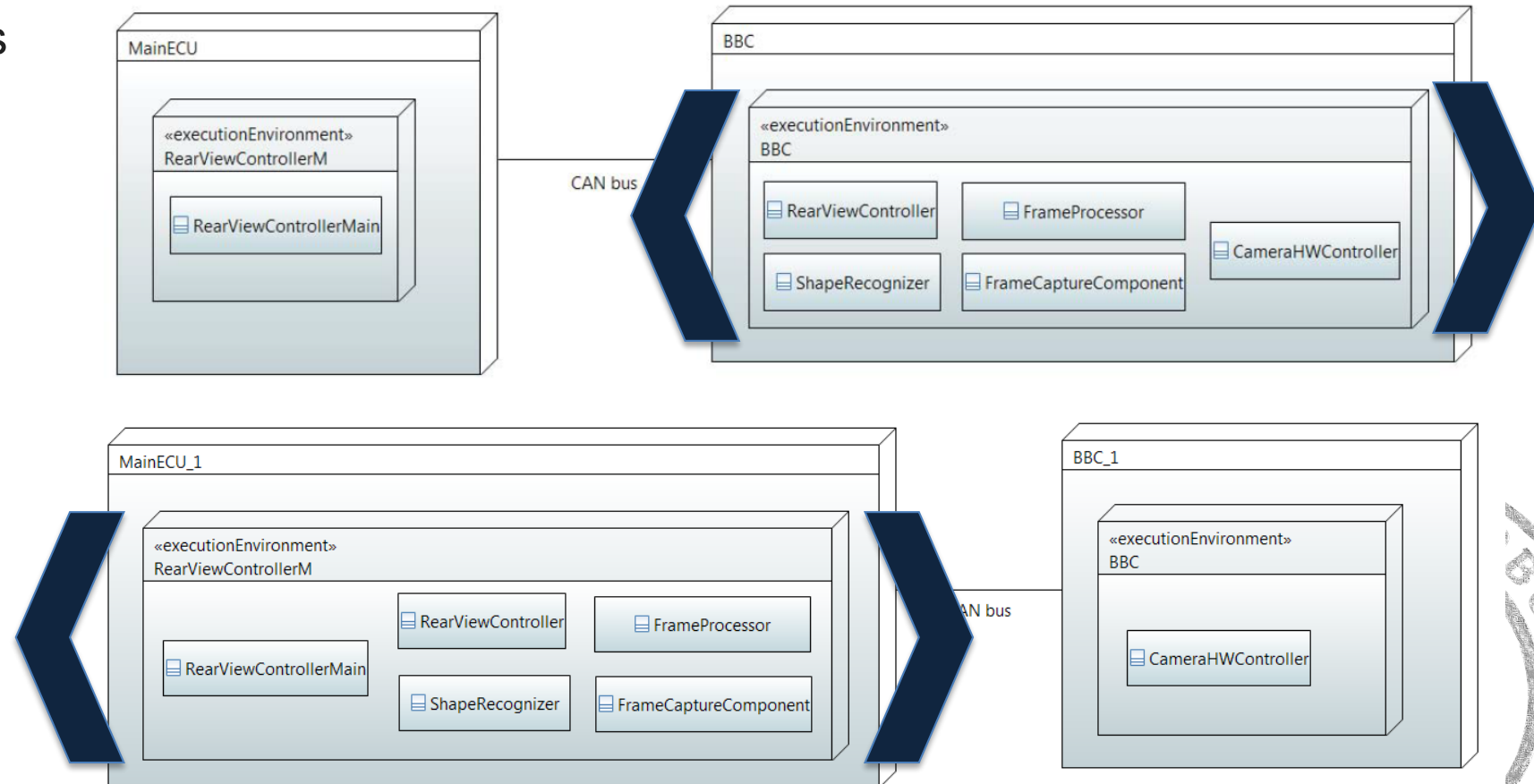
Sensitivity points – the most important outcome of scenario 1

- How should we V&V the components?
 - Which components can be complex?
 - When should we redesign to increase safety?



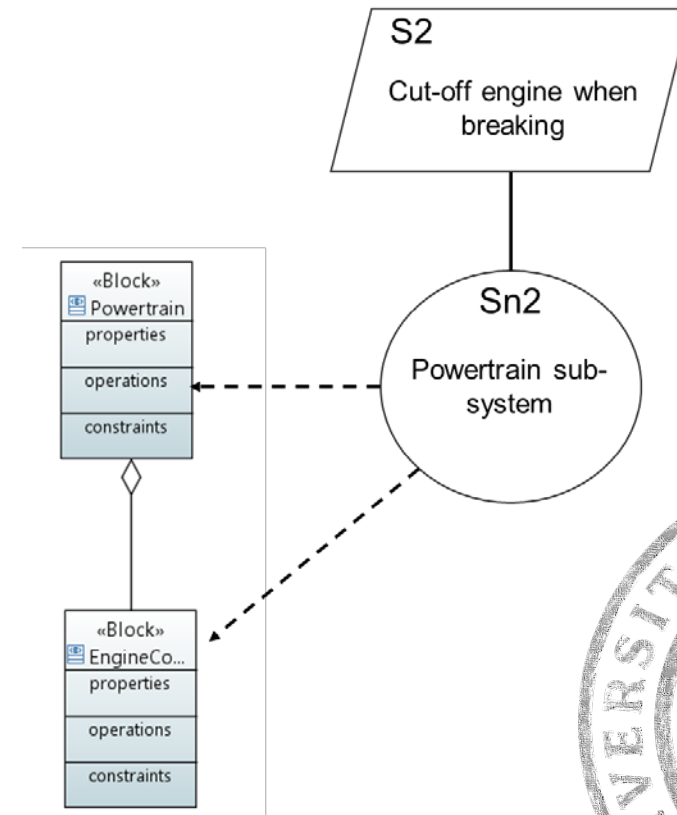
Sensitivity points – the most important outcome of scenario 1

- What kind of mechanisms should we use?
 - Is sandboxing needed?
 - Are interrupts allowed?



Next steps: improvement of safety analysis – link safety goal notation with architecture notations (e.g. SysML)

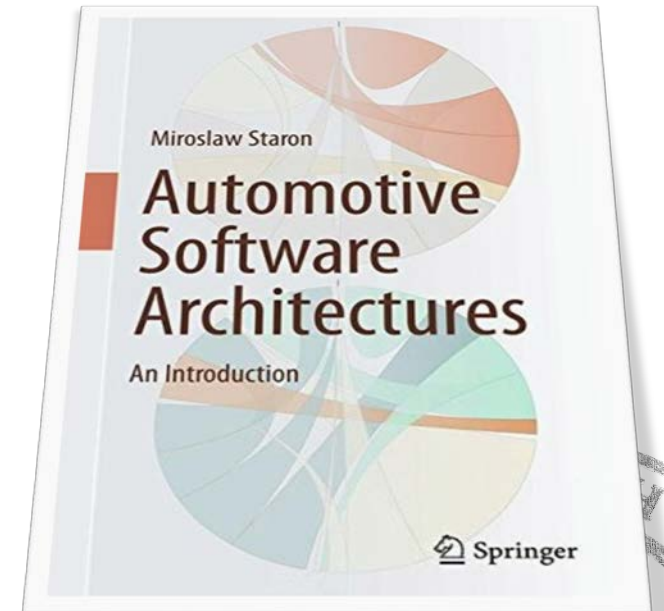
- Traceability between hazard analysis and software components
- Traceability of the design
- V&V methods aligned with Agile SW development





Summary and research outlook

- ATAM provides methods and tools to address the question: *How good is our architecture?*
- ISO 26262 provides the requirements for safety analysis and system construction (process)
- Bridging these leads to decreased workload for architecture analysis and safety argumentation
- In the end we can even address the question: *How safe is our architecture?*





Acknowledgements

- I would like to thank Dr. Imed Hammouda for letting me reuse his introductory slides about ATAM (slide 4 and 13)

