火龙果 · 整理
uml.org.cn

Automotive Functional Safety
汽车功能安全

TÜV Rheinland
Industrial Services

汽车功能安全3大业务范围

火龙果 · 整理
uml.org.cn

- 培训及人员资质认证
- 流程及项目技术咨询
- 功能安全评估及认证

- 培训
- 项目支持
- 产品认证

- 审核员培训
- 指导流程搭建
- 指导流程改进
- ASPICE认证

火龙果 · 整理
uml.org.cn

Software architecture design

Objective of this Lecture

火龙果 · 整理
uml.org.cn

1. Reasonable classification of function groups
2. Reasonable distribution of function blocks
3. Figure out different ASIL level function blocks
4. Consider ISO26262 requirements at software architecture level
5. Work out safety mechanism at software architecture level

Lecture Agenda

火龙果 · 整理
uml.org.cn

- Software Architecture Introduction
- Software Architecture requirements from ISO26262
- Data flow analysis and Control flow analysis
- Safety Critical Analysis(SCA)
- ASIL Decomposition
- Coexistence of Software Components
- Interference between Software Components
- Dependent Failure Analysis
- Software Safety Analysis
- Mechanism for error detection and handling at Software architecture Level
- Verification of the software architectural design

What’s Software architecture?

火龙果 · 整理
uml.org.cn

Static aspect

Dynamic aspect

Current Problems in Automotive Domain

火龙果 · 整理
uml.org.cn

No SW architecture

- No base for software safety analysis
- No base for software change impact analysis

Incomplete Interface

- Wrong ASIL decomposition, e.g. ASIL-SWC receive the safety related input data from QM-SWC
- Wrong freedom of interference

Rough Architecture

- All SWC are ASIL x, over-design, over engineering.
- Tips: balance your efforts on SW architecture phase and SW unit phase

Notations for Software architecture

- ISO 26262-8, table 1

Methods		ASIL A	ASIL B	ASIL C	ASIL D
1a	**Informal notations**	++	++	+	+

Figure

1b	**Semi-formal notations**	+	++	++	++

- 该方法有完整语法定义，但是语义定义不完整

UML class diagram

Principles for software architecture design

火龙果 · 整理
uml.org.cn

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.

ASIL Decomposition

火龙果 · 整理
uml.org.cn

Decomposition of ASIL D

Decomposition of ASIL C

Decomposition of ASIL B

Precondition of ASIL Decomposition:

- **Redundant**
- **Independent**

Wrong Decomposition Implementation

- For example the SWC-2 receives the safety related input signals from the QM Software Unit.

  - **Corruption of the input signals by the QM (D) unit could lead to a common cause fault of SWC-1 and SCW-2.**
  - **Possible impact on safety!**

Criticality Analysis

火龙果 · 整理
uml.org.cn

1. Implicit safety goal and safety requirement
2. Clear software architecture
3. Identify the software components related to safety goal
4. Classify the software components relevance to the safety path

Motivation for interference safety and non-safe software

火龙果 · 整理
uml.org.cn

If safe and non-safe software have access to the same resources, it must be guaranteed that there is no negative impact on the safe program, data or operations by non-safe software.

Freedom from interference

Dependent Failure Analysis

火龙果 · 整理
uml.org.cn

- Independence is threatened by common cause failures and cascading failures, while freedom from interference is only threatened by cascading failures

- Both systematic failures and random hardware failures have the potential to be dependent failures

火龙果 · 整理
uml.org.cn

Safety Analysis

Safety Analysis can:

火龙果 · 整理
uml.org.cn

1. Identify or confirm the safety-related parts of software

2. Support the specification and verify the efficiency of the safety mechanisms

3. Support the analysis of Interference and dependent failure between software components

Process of SWFMEA

火龙果 · 整理
uml.org.cn

1. Have a clear idea of the function of SW
2. Classify the SW components according to the function of SW
3. Figure out the failure modes, corresponding effect and cause
4. Check whether the cause is covered by current test action or safety mechanism

Mechanisms for error detection at SW architectu

火龙果 · 整理
uml.org.cn

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.

Mechanisms for error handling at SW architectur

火龙果 · 整理
uml.org.cn

Table 5 — Mechanisms for error handling at the software architectural level

Methods		ASIL			
		A	B	C	D
1a	Static recovery mechanism[a]	+	+	+	+
1b	Graceful degradation[b]	+	+	++	++
1c	Independent parallel redundancy[c]	o	o	+	++
1d	Correcting codes for data	+	+	+	+

[a] Static recovery mechanisms can include the use of recovery blocks, backward recovery, forward recovery and recovery through repetition.

[b] Graceful degradation at the software level refers to prioritizing functions to minimize the adverse effects of potential failures on functional safety.

[c] Independent parallel redundancy can be realized as dissimilar software in each parallel path.

Verification of the software architectural design

火龙果 · 整理
uml.org.cn

Table 6 — Methods for the verification of the software architectural design

Methods		ASIL			
		A	B	C	D
1a	Walk-through of the design[a]	++	+	o	o
1b	Inspection of the design[a]	+	++	++	++
1c	Simulation of dynamic parts of the design[b]	+	+	+	++
1d	Prototype generation	o	o	+	++
1e	Formal verification	o	o	+	+
1f	Control flow analysis[c]	+	+	++	++
1g	Data flow analysis[c]	+	+	++	++

[a] In the case of model-based development these methods can be applied to the model.

[b] Method 1c requires the usage of executable models for the dynamic parts of the software architecture.

[c] Control and data flow analysis may be limited to safety-related components and their interfaces.