Workflow between ISO 26262 and ISO 21448 Standards for Autonomous Vehicles

by Kaushik Madala, Carlos Avalos Gonzalez and Gokul Krithivasan
Portland, Oregon

Assuring safety is important in autonomous vehicles. The safety related to autonomous vehicles can be primarily viewed from two perspectives: the functional safety (FuSa) perspective and the safety of the intended functionality (SOTIF) perspective. While FuSa ensures the system has an acceptable risk with respect to malfunctions of electrical and electronic components, SOTIF ensures the system has an acceptable risk with respect to functional insufficiencies and performance limitations.

ISO 26262 and ISO 21448 are the state-of-the-art international standards used to ensure compliance with FuSa and SOTIF for autonomous automotive systems, respectively. The ISO 21448 standard mentions the need for alignment of ISO 26262 activities with the ISO 21448 activities and describes the mapping at a very high level. However, given the iterative nature of SOTIF activities in ISO 21448, the workflow between the two standards is not a direct one-to-one mapping. Hence, we need a clear understanding how we can align ISO 26262 and ISO 21448 activities, and on how analysis done in one standard can impact the other.

To achieve this, in this paper we propose a detailed workflow between ISO 26262 and ISO 21448 standards. We discuss guidelines on how to find if a change to design due to SOTIF modification can affect FuSa analysis and vice versa. We also discuss the aspects we need to consider for agile development when we want to ensure the system being analyzed complies both with FuSa and SOTIF.

Introduction

Safety is one of the prominent aspects we need to consider while developing an automotive system. With many companies moving toward development of autonomous vehicles, the complexity of systems has been increasing, thereby increasing the complexity with respect to safety assurance. Safety standards such as ISO 26262 [Ref. 1], ISO 21448 [Ref. 2], and ANSI/UL 4600 [Ref. 3] offer guidance on how the engineers and analysts can assure safety of an automotive system.

ISO 26262 [Ref. 1] is a functional safety standard, which provides guidance on how to demonstrate the electric and electronic components of a system does not create unacceptable risk. ISO 21448 [Ref. 2], on the other hand, is a safety of the intended functionality (SOTIF) standard, which offers guidance on how to ensure the system has no unreasonable risks due to functional insufficiencies and performance limitations of the components in the system. SOTIF standards aims at reducing both known and unknown risks. Unlike ISO 26262 and ISO 21448 standards, UL 4600 [Ref. 3] is a standard that offers guidance on building a safety case for an autonomous system. All three standards aid in making an autonomous vehicle safer.

While UL 4600 is a stand-alone standard, ISO 21448 Clause 4.4 mentions the need to align its activities with ISO 26262 activities. While the ISO 21448 Annex A.2.3 provides high-level information on the alignment of ISO 21448 and ISO 26262 activities, we do not have sufficient information on how these activities interact. Moreover, ISO 26262 limits its analysis to the electrical and electronic components related to safety systems. However, ISO 21448 is applicable to the entire autonomous vehicle as well as any human-machine interfaces. As a result, it is necessary to understand what is the preferable workflow of activities at system, hardware and software levels, and how they might not only help in ensuring safety from the perspective of both ISO 26262 and ISO 21448, but also in making the process systematic for identifying unknowns.

To address the limitation, in this paper we detail a workflow that aligns the activities between ISO 26262 and ISO 21448. We also discuss the reasons behind the flows considered in the workflow, and we give a high-level illustrative

example to show the application of the workflow. We believe our workflow will help in creating a better process in organization as well as foster better communication among teams that work together.

The rest of the paper is organized as follows. In the next section, we discuss the background topics such as the ISO 26262 workflow, and ISO 21448 workflow, and related work. Then, we discuss the workflow and alignment of ISO 26262 and ISO 21448 activities. After the description of workflow, we provide a high-level illustrative example for the workflow and conclude.

Background and Related Work

ISO 26262

ISO 26262 [Ref. 1] is the functional safety standard for automotive vehicles. Functional safety refers to absence of unreasonable risk due to malfunctions of electrical and electronic systems. ISO 26262 follows a V-model for system-level development, hardware development and software development. The process model followed by ISO 26262 for automotive systems along with part numbers associated with the phase is as shown in Figure 1.

We start with the concept phase (Part 3 of ISO 26262) during which we define the item for which we are going to analyze functional safety. During this phase we create a functional block diagram and identify functionalities associated with each block. Once the item definition is completed, we identify malfunctions for the blocks within the item boundary. Based on these malfunctions we identify corresponding hazards and perform hazard analysis and risk assessment (HARA). During HARA we assign the severity (S), controllability (C), and exposure (E) ratings for each malfunction, and identify corresponding automotive safety integrity level (ASIL) level. The highest level identified is the ASIL level that is allocated to the entire system. We also define safety goals corresponding to each hazard during HARA. Once HARA is completed, we create functional safety requirements and aggregate all the findings into a functional safety concept (FSC).

Once the FSC is generated, we move to the next phase: system specification and design (Part 4 of ISO 26262). During this phase, we define the system specification, architecture, technical safety requirements, hardware-software interface requirements, and aggregate findings into a technical safety concept (TSC). After defining the TSC, we move to development at the hardware level (Part 5 of ISO 26262). During this phase, we create a

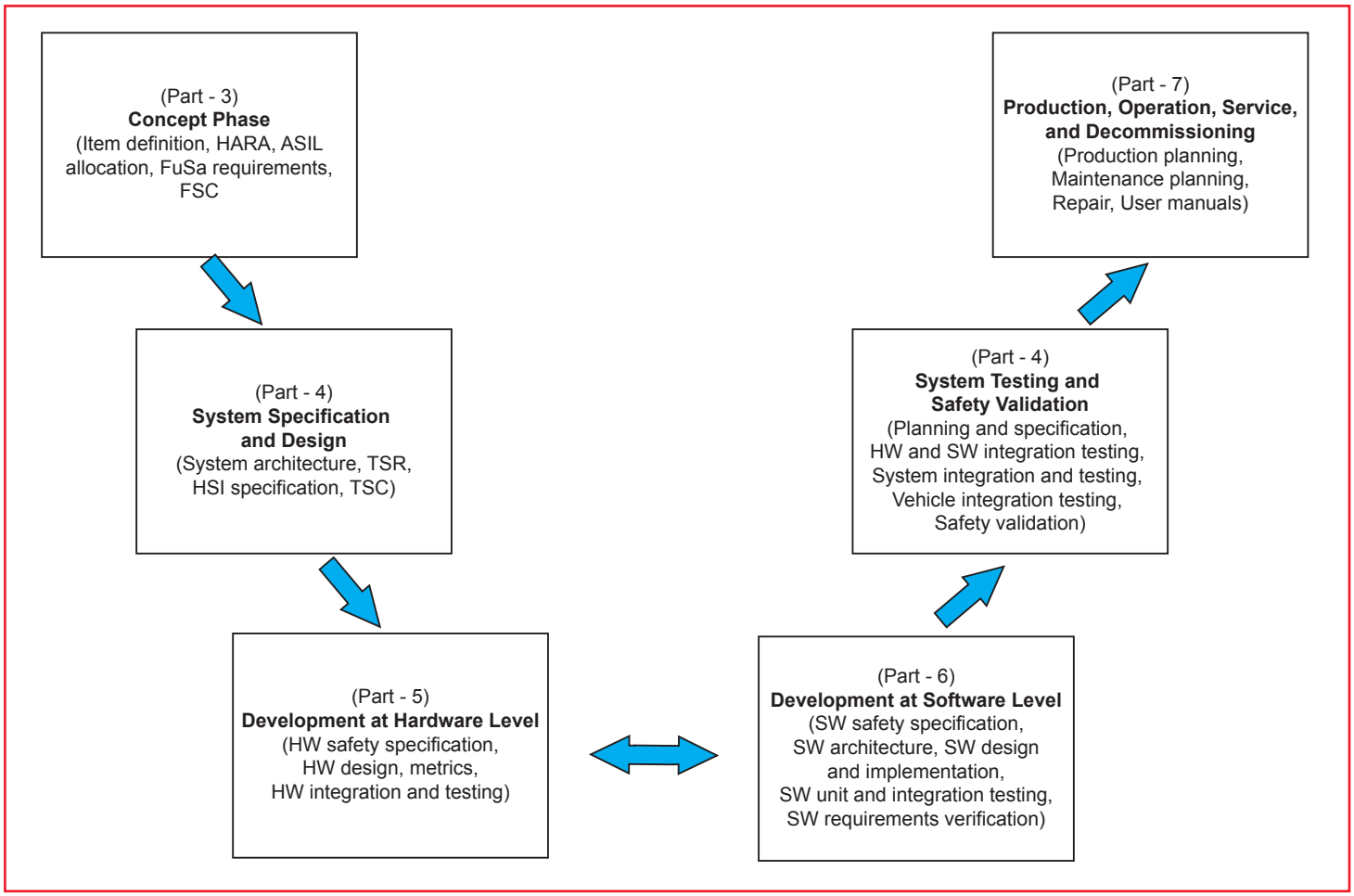


Figure 1— High-level Workflow of ISO 26262 (Functional Safety).

hardware (HW) safety specification and HW design, analyze and assess HW architectural metrics, calculate random hardware failure rates, and perform HW integration and testing.

We start the development at the software-level phase (Part 6 of ISO 26262) in parallel to the hardware development phase. In this phase, we start by defining the software (SW) safety specification, SW architecture, and design. We then implement the software, and perform unit, integration, and requirements-based testing.

After the completion of development at hardware and software levels we move to the system testing and safety validation phase (Part 4 of ISO 26262), during which we create a plan to perform integration testing for HW, SW, system, and the vehicle, as well as performing safety validation. Once the testing is completed, we move to the production, operation, service, and decommissioning phase (Part 7 of ISO 26262), during which we perform planning for production and maintenance, create instructions for repair and prepare user manuals. While following the process model of ISO 26262, we might use additional supporting processes detailed in Part 8 of ISO 26262 such as tool support qualification and configuration management.

ISO 21448

Unlike ISO 26262, ISO 21448 [Ref. 2] does not deal with malfunctions of electrical and electronic components in a vehicle, nor is it restricted to the safety system. ISO 21448 is the standard for safety of the intended functionality and is mainly applicable to systems with autonomy. It deals with safety issues that arise because of functional insufficiencies, performance limitations, and foreseeable misuses.

The process suggested by ISO 21448 is shown in Figure 2. The figure shows each phase in the process along with their corresponding clause number in the standard. As shown in the figure, we start with gathering specification and design (Clause 5 of ISO 21448) for the autonomy system. Based on the functionalities defined in the specification, we perform hazard identification and risk evaluation (HIRE) (Clause 6) by identifying potential hazardous behaviors.

During HIRE, we assess controllability (C) and severity (S) of each hazardous behavior and assign them to have reasonable and acceptable risk if C = 0 or S = 0. If both C and S are greater than zero, then the risk is considered as unacceptable.

It is also during the HIRE we can start the specification for acceptance criteria for hazards with unacceptable risks. To define acceptance criteria, we need to have a rationale such as GAMAB (globally at least as good, from the French *"globalement au moins aussi bon"*), ALARP (as low as reasonably practicable), and MEM (minimal endogenous mortality) [Ref. 4], and a data source that has information on crashes or fatalities. Using this information, we propose an acceptance criterion which ensures the number of potential accidents that can results are fewer than with a human driver or fewer or similar to the previous version of the vehicle.

After the HIRE is completed, we perform the analysis of triggering events and functional insufficiencies (Clause 7). In this phase, we identify sensor limitations, algorithm limitations, actuator limitations, and possible misuse cases. We then analyze the triggering conditions

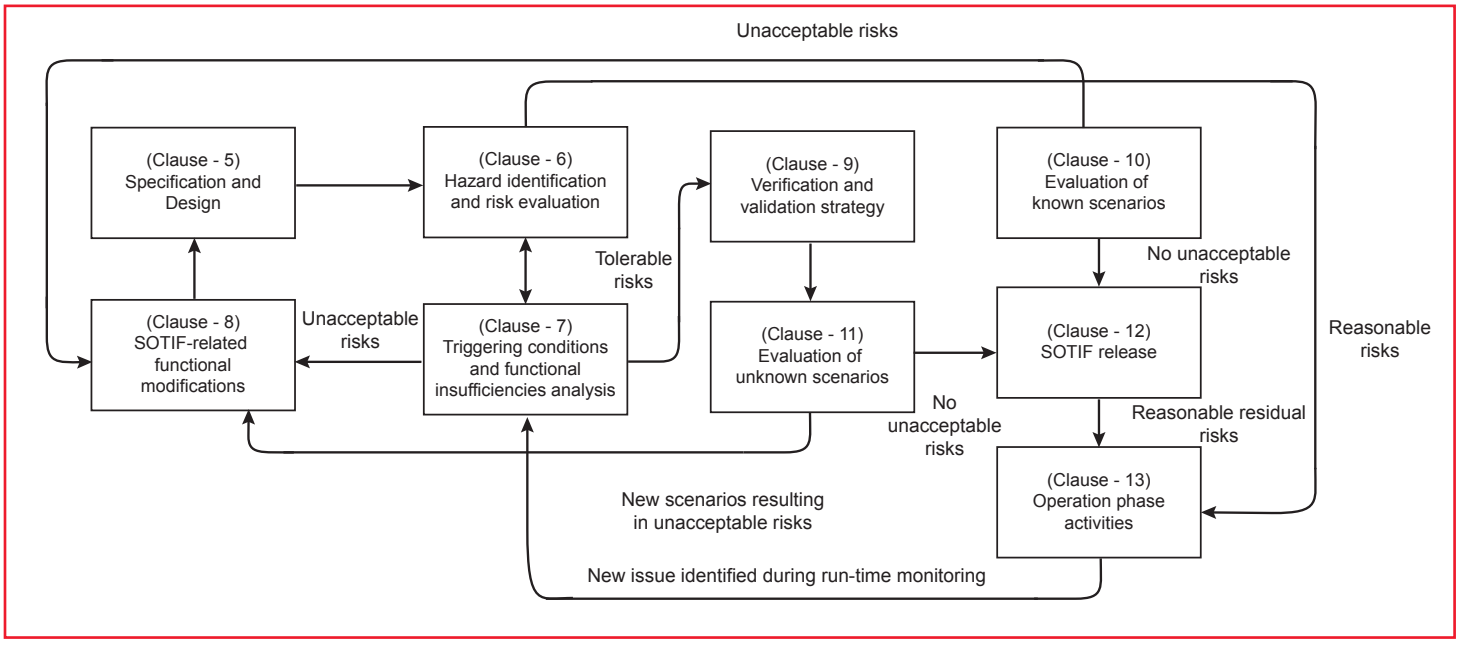


Figure 2 — ISO 21448 (SOTIF) Workflow.

that have unacceptable risks, i.e., residual risk is high and corresponding acceptance criteria might not be met. For triggering conditions with unacceptable risks, we define SOTIF-related functional modifications (Clause 8). If a triggering condition does have either C = 0 and S = 0, but its acceptance criteria is achievable, then we can consider it to have tolerable risk and move to the definition of the verification and validation strategy (Clause 9).

After the verification and validation strategy is defined, we perform the evaluation of known scenarios (Clause 10), during which we perform sensor verification, algorithm verification, actuator verification, and vehicle verification. Once we complete the evaluation of known scenarios, we move to the evaluation of unknown scenarios (Clause 11). We perform this by performing a safety validation in the real world and ensuring the residual risk is at an acceptable level.

After the completion of the evaluation of unknown scenarios, we move to SOTIF release (Clause 12), where we ascertain if the system is rigorously and sufficiently analyzed to ensure that the SOTIF is acceptable. If not, it is required to perform SOTIF functional modifications. If the SOTIF is deemed to be acceptable by assessors, then we can go to production and operations. During the operation phase (Clause 13), it is still required to have mechanisms such as run-time monitoring to identify any new event or condition that can result in SOTIF issues.

Related work

To date, some researchers [Refs. 5-8] have proposed workflows between ISO 26262 and ISO 21448 by focusing on a specific type of system such as a machine learning (ML) system or advanced driver assistance system (ADAS).

For example, Kirovskii and Gorelov [Ref. 6] proposed a workflow considering ISO 26262 and ISO 21448 for driver assistance systems by mostly focusing on system-level behavior. Radlak et al. [Ref. 8] proposed an approach for suggesting how to organize the machine learning based products in order to ensure compliance with both ISO 26262 and ISO 21448. Kirovsky and Byakov [Ref. 7] proposed an approach to specify requirements related to situations gathered from statistics taking into account both ISO 26262 and ISO 21448. Ishigooka et al. [Ref. 5] proposed a design process for degradation of automated driver systems considering both ISO 26262 and ISO 21448.

Although these approaches offer guidance on how to ensure adherence to ISO 26262 and ISO 21448, they only focus on a specific system or an aspect. Unlike these approaches, in our approach, we propose a complete generic workflow that covers the system, hardware and software development.

Workflow between ISO 26262 and ISO 21448

Figure 3 illustrates our proposed detailed workflow between ISO 26262 and ISO 21448. The rectangles indicate the ISO 26262 activity along with its associated part number. The rounded rectangle represents a SOTIF activity along with its respective clause number. The single directional arrow represents unidirectional association — it implies it is an anticipated sequential flow of activities. The bidirectional arrow represents a bidirectional association, which implies that, based on the updates in one of the activities, the other activity might need to be performed again or the corresponding work products need to be modified. The bidirectional dotted arrow represents bidirectional impact between an ISO 26262 activity and ISO 21448 activity. A bidirectional impact implies that information found in one standard might affect the activity or work product in the other standard. We shall now discuss the workflow.

As shown in Figure 3, ISO 26262 starts with item definition, whereas ISO 21448 starts with specification and design. Note that it may not be always possible to have a full specification and design at the beginning. Hence, SOTIF is a highly iterative process. From the specification and design, we can generate high-level functional architecture (similar to functional block diagram for FuSa, but oriented towards autonomy). The item being used in ISO 26262 need not match completely with the autonomy system we analyze in ISO 21448.

There are three different types of architectures we can consider between a FuSa system and autonomy system, which are shown in Figure 4. As shown in the figure, for type 1 architecture we have a separate protection system and autonomy system. For example, if we consider a vehicle to have a remote monitor to ensure safety of the system. Then the system related to the remote operator and controls are considered for functional safety, whereas for SOTIF we consider autonomy, as well as any of the human machine interactions. In these architectures, the SOTIF and functional safety aspects do not affect each other too much and performing a change impact analysis is easier.

A second type of architecture we consider is the type 2 architecture in which a functional safety system is a subset of autonomy. An example is a vehicle in which the emergency braking system is considered a FuSa system, but the vehicle has additional perception algorithms which are not included as a part of a FuSa system. In these systems, we need to map any updates related to the FuSa components and their interfaces to SOTIF and vice versa. It helps in making the design better as well as in considering both FuSa and SOTIF when we need to provide functional modifications in SOTIF.

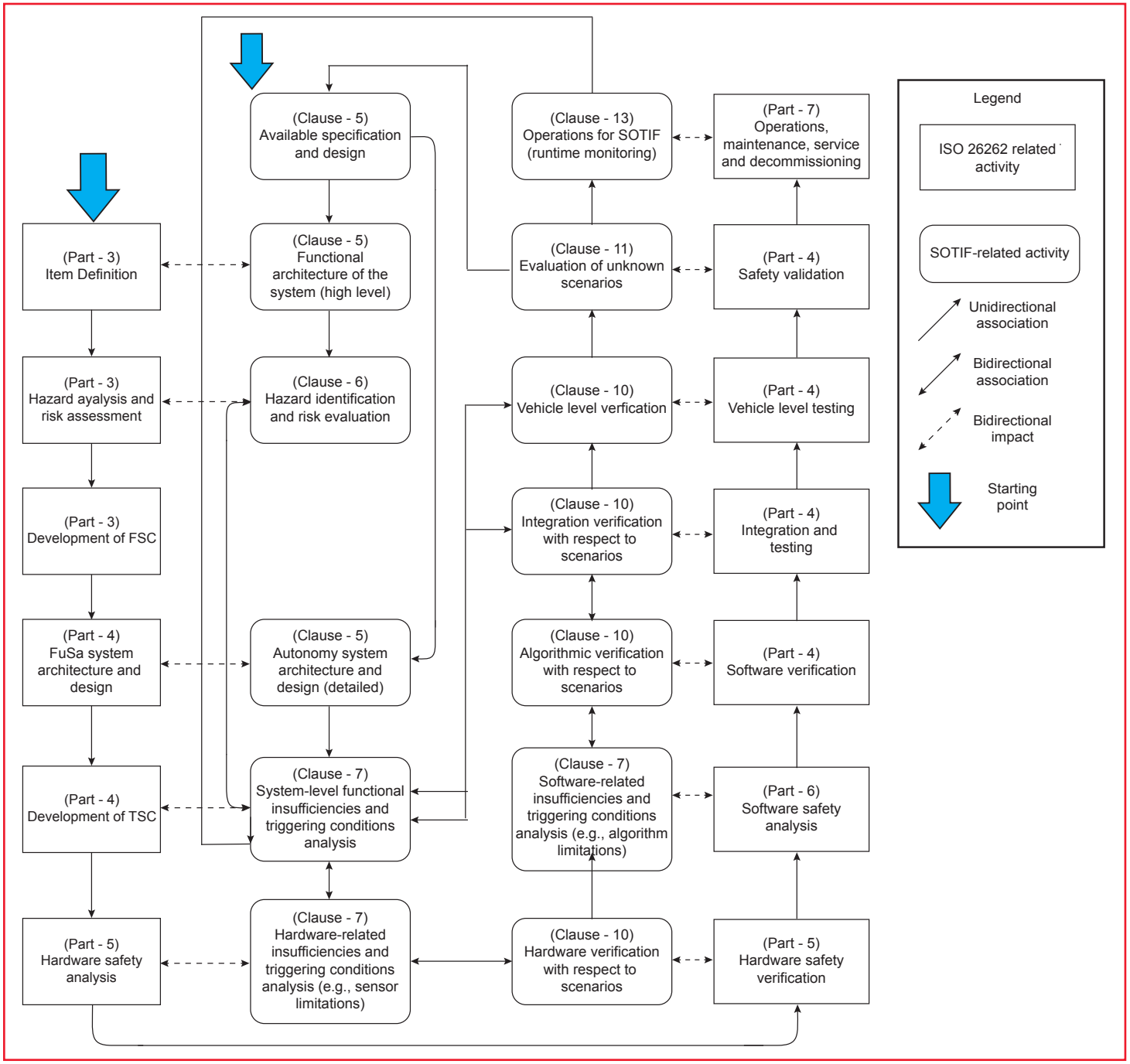
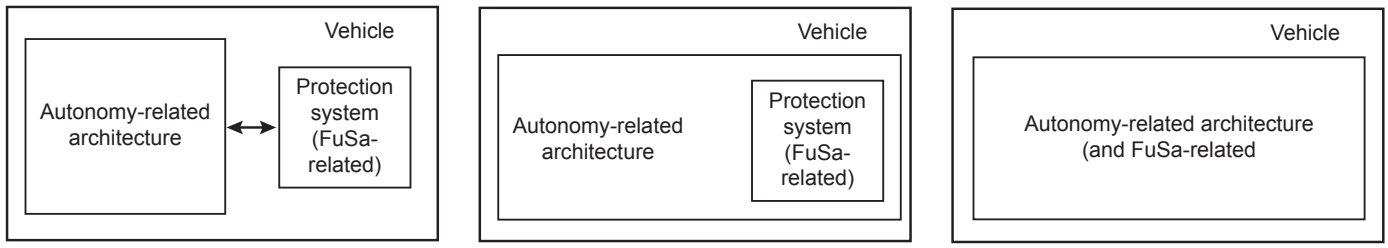


Figure 3 — Workflow Between ISO 26262 and ISO 21448.



Type 1. Vehicle in which autonomy and functional safety-related system operate separately and may communicate with each other

Type 2. Vehicle in which functional safety-related system is a sub-set of autonomy

Type 3. Vehicle in which autonomy system and functional safety system are one and the same

Figure 4 — Types of Architectures that are Possible in a Vehicle Under Analysis for Which We Aim to Comply with ISO 26262 and ISO 21448.

A third type of architecture, referred to as type 3 in the figure, is the one where the architecture that needs to comply with both FuSa and SOTIF is the same. An example is a vehicle that uses the end-to-end machine learning (ML) model [10], i.e., an ML model which takes sensor data as input and produces the wheel motor speeds as outputs.

After the item definition in ISO 26262, as mentioned in background and related work, we perform HARA. In the case of ISO 21448 we perform HIRE. Note that during HARA it might be possible to identify SOTIF-related issues and similarly during HIRE we might be able to identify malfunctions. Hence, we need to keep track of them and update the analyses accordingly.

After the HARA, in ISO 26262, we create a functional safety concept (FSC) that contains functional safety requirements, safety goals, ASIL information, and any findings from HARA. We do not have the equivalent of FSC in ISO 21448. After the FSC is created in ISO 26262, we move to the system development phase during which we first create system specification and design. While ISO 21448 does not have an equivalent phase, the architecture that was initially part of the specification and design could be updated (along with the FuSa system architecture, if it is part of autonomy).

After the system architecture and design phase in ISO 26262, we complete the technical safety concept (TSC) that contains technical safety requirements, hardware-software interface specification, and any other observations made as a part of the system safety analysis. An equivalent of TSC phase in ISO 21448 is the analysis of functional insufficiencies and triggering conditions at the system level. In this phase, similar to how we anticipate functional safety issues based on hardware and software details in ISO 26262, we identify potential insufficiencies and their causes based on the list of sensors, algorithms, and actuators. However, these insufficiencies cannot be finalized as we do not know exactly what are the insufficiencies of components and the triggering conditions causing those insufficiencies. A bottom-up analysis is necessary to understand if the potential insufficiency we assume can actually occur at the system level.

After the development of TSC in ISO 26262, we move to the hardware (HW) safety analysis phase. Note that although we specify hardware safety analysis phase, after the completion of TSC, hardware and software development can be started in parallel. During the HW analysis phase we generate hardware specification, design, assess the HW architectural metrics and perform FMEDA [11]. In case of ISO 21448, an equivalent analy-

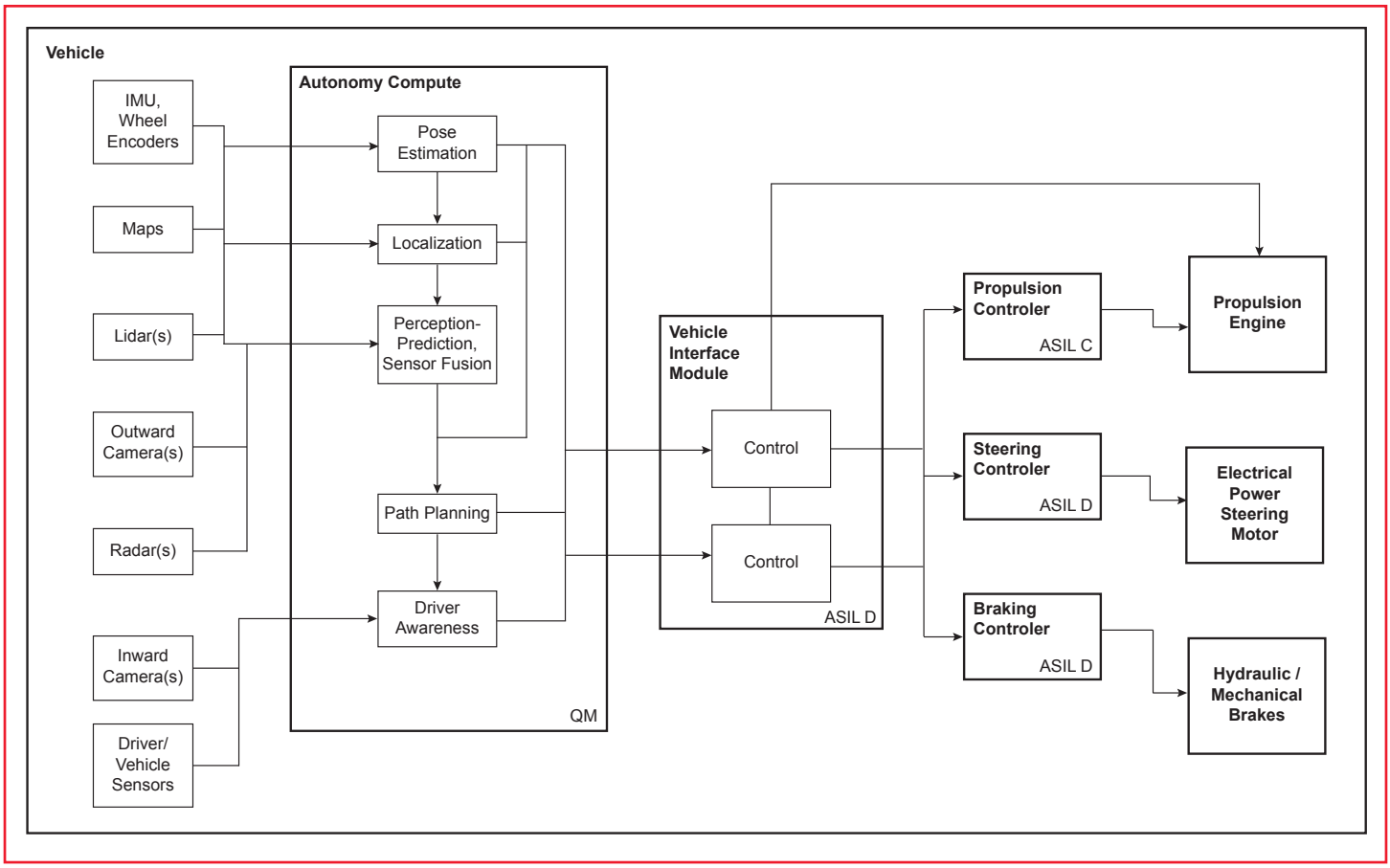


Figure 5 — Type 2 Architecture of Pose, Localization, Prediction, Planning and Control Functions.

sis we perform is the analysis of functional insufficiencies and triggering conditions specific to HW components. During this phase we estimate the sensor limitations, actuator limitations, and conditions under which ECU or any other hardware performance can be affected.

After the HW safety analysis in ISO 26262, we perform HW safety verification, during which we integrate the hardware and test it. An equivalent in ISO 21448 is verification of the hardware components with respect to scenarios that can occur in an operational design domain (ODD). By doing so, we can verify if the system is behaving safely as expected in the presence of triggering conditions.

Following the hardware (HW) safety verification in ISO 26262 is the software safety (SW) analysis, where we define SW specification, design and perform software FMEA to identify SW-level issues. An equivalent in ISO 21448 is the analysis of functional insufficiencies and triggering conditions for SW. During this phase we identify algorithmic limitations. Note that in case of machine learning (ML) algorithms we need to perform analysis for not only ML models but also the ML libraries that support the models.

After the SW safety analysis in ISO 26262, we perform SW verification via unit testing, integration testing, and requirements-based testing. An equivalent step in ISO 21448 is the verification of software with respect to the scenarios in the ODD. We might be able to uncover new triggering conditions for algorithms during this analysis. If the triggering conditions at HW or SW are updated, then a bottom-up analysis is performed to update the system-level insufficiencies as well as the vehicle-level hazards. Note that during the analysis or verification process, we might be able to uncover new FuSa or SOTIF issues and needs to be mapped across standards accordingly. If we proposed a modification to address SOTIF issues, we should also take into account if the newly added change can result in any new functional safety issues.

After the completion of SW verification in ISO 26262, we perform integration and testing, followed by vehicle-level testing. The corresponding equivalents in ISO 21448 are the scenario-based integration verification and vehicle verification. During this activity, if we find any new issues, we map them to the system-

Table 1 — Workflow Illustration for the Example Type 2 Architecture Considered in Figure 6.

P/C	FuSa	SOTIF
	Item Definition	**Functional Architecture of the System**
P3, C5	Describing diverse sensor modalities as inputs that provide the perceived information to an autonomy computation, which performs the high-level task of analyzing the data captured by the sensors.— information which eventually is transmitted in form of waypoints of trajectory to vehicle interface module (VIM). Upon correct reception, information received is cross checked and, according to the availability of downstream actuators' controllers, is converted into safe driving commands to properly activate actuators.	Function to be executed during the approved highways once automatic control is activated and driver is paying attention to the road with hand on the steering wheel. Function to be performed mainly by perception of sensors, information processed in autonomy compute and shared resourced over VIM. Medium speed vehicles, road signaling and no pedestrians are expected during the driving cycle. Functionality capable to detect limited field of view of camera, inclement weather conditions and degrade its operation and start progression to safe state once a deviation of the nominal specification is found.
	Hazard analysis and risk assessment (HARA)	**Hazard identification and risk evaluation**
P3, C6	Unintended activation over safe limit yaw rate of the steering wheel once the vehicle is on high-speed highway, is found as a critical hazardous event ranked as ASIL D due to high severity, high exposure due to very frequent access to highway and duration of it is considered long period of time. Top level safety requirement (safety Goal SG) is defined to mitigate hazardous event occurrence.	Sensor correctly sense environment, camera range to detect traffic sign has a limited visibility that impede ability to take the required exit off the highway, leading to unintended activation over safe limit yaw rate of the steering wheel once the vehicle is on high-speed highway; since sign is detected very close to the exit, the safe yaw rate is exceeded that which is expected.
	Functional safety concept (FSC)	
P3	Requirements related to steering controls are derived from defined safety goal in order to assure safety goal.	

	FuSa system architecture and design	Autonomy system architecture and design (detailed)
P4,C5	Allocation and critical timing intervals are defined to identified elements of the high- level architecture. Sensors are ASIL B, autonomy compute QM, VIM ASIL D, Propulsion controller ASIL C, Braking controller ASIL D, steering controller ASIL D.	Updated autonomy architecture based on latest requirements and functional safety architecture (if FuSa system is part of autonomy system). We will be considering all the components shown in Figure 6 along with their respective interfaces.
	Development of TSC	**System-level analysis of functional insufficiencies and triggering conditions**
P4, C7	Safety mechanisms on how to mitigate the faults related to steering controls are defined. Any hardware-software interfaces that needs to be considered between steering controller and VIM, as well as with steering motors, are considered and analyzed	For the hazardous behavior discussed in the hazard identification and risk evaluation above, if we consider the perception system, a functional insufficiency at a system level will be limited visibility.
	Hardware safety analysis	**Hardware-level analysis of functional insufficiencies and triggering conditions**
P5,C7	During hardware safety analysis, we consider which hardware components are used for steering controls and identify the anticipated failure rate for their random failures	The hardware components corresponding to the perception system can be the sensors that it gets information from. For the steering system, we consider the steering motors. If we consider the camera, a potential reason for low visibility can be fog.
	Hardware safety verification	**Hardware verification W.R.T scenarios**
P5, C10	During hardware verification, we perform fault injection and analyze if the hardware has expected failure rate and works as intended or not.	Based on our analysis in previous step, we analyze the performance of hardware for different possible triggering conditions and check if behavior is as expected or not.
	Software safety analysis	**Software-level analysis of functional insufficiencies and triggering conditions**
P6, C7	During software safety analysis, we analyze what faults might occur if there are problems with signals from and to steering controls.	The software components for perception system can be the algorithm used to detect traffic signs. One factor that can affect the algorithm is glare, which can delay the identification of the traffic sign.
	Software safety verification	**Software verification W.R.T. scenarios**
P6, C10	During this step, we test each software unit related to steering control and then integrate them to verify if signals are sent as intended. We also perform requirements-based testing.	Based on our analysis in previous step, we analyze the performance of algorithms for different possible triggering conditions and check if behavior is as expected or not using a large data set.
	Integration and testing	**Integration verification W.R.T. scenarios**
P4, C10	We integrate the hardware and software of the steering modules and test the system. We also integrate the steering modules with VIM and steering motors and test their integration.	We integrate hardware and software with respect to perception system and other systems, and check if they provide acceptable performance with respect to scenarios that can occur in ODD.
	Vehicle-level testing	**Vehicle-level verification**
P4, C10	We integrate all the systems and test them..	We evaluate the entire vehicle for all known scenarios.
	Safety validation	**Evaluation of unknown scenarios**
P4, C11	We check if safety goals are all met or not.	We check if the residual risk with respect to unknown scenarios is acceptable.
	Operations, maintenance, service, and decommissioning	**Operations of SOTIF**
P7, C13	We create production plan, user manuals, upgrade and repair instructions	We perform run-time monitoring to identify any new potential triggering conditions.

level analysis of triggering conditions and functional insufficiencies. Also, if we are able to identify any new triggering conditions for system-level events based on verification at the hardware and software levels, the test plan is updated. Note that the acceptance of the risks at vehicle-level verification is done based on acceptance criteria and corresponding validation targets we set.

After the vehicle-level testing in ISO 26262, we perform safety validation. An equivalent to this activity in ISO 21448 is the evaluation of unknown scenarios, where we deploy a vehicle into the road and calculate the residual risk. Any new issues identified during unknown scenarios are used to perform SOTIF functional modification and updating of the specification and design as required.

After the safety validation in ISO 26262, we move to production, operations, decommissioning and the service phase during which we create production and maintenance plans, procedures for upgrades and repairs, and development of user manuals. An equivalent in ISO 26262 is the operations phase during which we monitor the vehicle deployed in the real world to see if there might be an occurrence of potential unknown triggering conditions. If a new triggering condition is identified, we add it to the existing list and reiterate the SOTIF process.

Illustrative Example

In order to provide a clear guidance of the proposed detailed workflow shown on Figure 3 and the different type of architectures shown on Figure 4, consider the following type 2 architecture shown on Figure 5 to perform automatic control of the propulsion, brake and steering actuators.

The workflow for the example is provided in Table 1. The first column in the table refers to part numbers of ISO 26262 and clause numbers of ISO 21448, which we consider to have an alignment. The notation PX refers to part number X and notation CY refers to clause number Y.

Conclusion and Future Work

In this paper, we proposed a detailed workflow between ISO 26262 and ISO 21448 by showing which phases need to be aligned together. We also discussed the need to ensure that a design change to address a SOTIF issue does not result in a new FuSa issue and vice versa. We discussed the workflow with an example architecture that has an automatic control of the propulsion, brake and steering actuators. Although we discussed the alignment of phases, we did not talk about how quality management and change management must be considered when aligning both the standards. We intend to perform such analysis as a part of our future work.

References

1. International Organization for Standardization. *ISO 26262:2018, Road Vehicles – Functional Safety,* 2018.
2. International Organization for Standardization. *ISO/DIS 21448, Road Vehicles - Safety of the Intended Functionality,* 2021.
3. Underwriter Laboratories. *ANSI/UL 4600 - Standard for Evaluation of Autonomous Products,* 2020.
4. Kron, H. "On the evaluation of risk acceptance principles," *19th Dresden Conference on Traffic and Transportation Science,* 2003.
5. Ishigooka, T., S. Otsuka, K. Serizawa, R. Tsuchiya, & F. Narisawa. "Graceful Degradation Design Process for Autonomous Driving System," *International Conference on Computer Safety, Reliability, and Security*, 19-34, Springer, 2019.
6. Kirovskii, O. M., & V. A. Gorelov. "Driver Assistance Systems: Analysis, Tests and the Safety Case. ISO 26262 and ISO PAS 21448," *IOP Conference Series: Materials Science and Engineering,* IOP Publishing, 2019.
7. Kirovsky, O., & K. Byakov. "Scenario-based definition of technical safety requirements for autonomous road vehicles," I*OP Conference Series: Materials Science and Engineering,* 012- 016), IPO Publishing, 2020.
8. Radlak, K., M. Szczepankiewicz, T. Jones, & P. Serwa. "Organization of machine learning based product development as per ISO 26262 and ISO/PAS 21448," *2020 IEEE 25th Pacific Rim International Symposium on Dependable Computing (PRDC),* 110-119, IEEE, 2020.
9. Amini, A., I. Gilitschenski, J. Phillips, J. Moseyko, R. Banerjee, S. Karaman, & D. Rus. "Learning Robust Control Policies for End-to-end Autonomous Driving from Data-driven Simulation," *IEEE Robotics and Automation Letters,* 1143-1150, 2020.
10. Goble, W. M., & A. C. Brombacher. "Using a Failure Modes, Effects and Diagnostic Analysis (FMEDA) to Measure Diagnostic Coverage in Programmable Electronic Systems," *Reliability Engineering & System Safety,* 145-148, 1999.