Contract-Based Safety Certification for the Dynamic Adaptation Behavior of Networked Embedded Systems

Vom Fachbereich Informatik der

Technischen Universität Kaiserslautern

zur Verleihung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

M. Sc. Sebastian Müller

Datum der wissenschaftlichen Aussprache: 09. September 2019

Dekan:	Prof. Dr. Stefan Deßloch
Erster Berichterstatter:	Prof. Dr. Peter Liggesmeyer
Zweiter Berichterstatter:	Prof. Dr. Karsten Berns

D 386

Abstract

Under the notion of Cyber-Physical Systems an increasingly important research area has evolved with the aim of improving the connectivity and interoperability of previously separate system functions. Today, the advanced networking and processing capabilities of embedded systems make it possible to establish strongly distributed, heterogeneous systems of systems. In such configurations, the system boundary does not necessarily end with the hardware, but can also take into account the wider context such as people and environmental factors. In addition to being open and adaptive to other networked systems at integration time, such systems need to be able to adapt themselves in accordance with dynamic changes in their application environments. Considering that many of the potential application domains are inherently safety-critical, it has to be ensured that the necessary modifications in the individual system behavior are safe. However, currently available state-of-the-practice and state-of-the-art approaches for safety assurance and certification are not applicable to this context.

To provide a feasible solution approach, this thesis introduces a framework that allows "just-in-time" safety certification for the dynamic adaptation behavior of networked systems. Dynamic safety contracts (DSCs) are presented as the core solution concept for monitoring and synthesis of decentralized safety knowledge. Ultimately, this opens up a path towards standardized service provision concepts as a set of safety-related runtime evidences. DSCs enable the modular specification of relevant safety features in networked applications as a series of formalized demand-guarantee dependencies. The specified safety features can be hierarchically integrated and linked to an interpretation level for accessing the scope of possible safe behavioral adaptations. In this way, the networked adaptation behavior can be conditionally certified with respect to the fulfilled DSC safety features during operation. As long as the continuous evaluation process provides safe adaptation behavior for a networked application context, safety can be guaranteed for a networked system mode at runtime. Significant safety-related changes in the application context, however, can lead to situations in which no safe adaptation behavior is available for the current system state. In such cases, the remaining DSC guarantees can be utilized to determine optimal degradation concepts for the dynamic applications.

For the operationalization of the DSCs approach, suitable specification elements and mechanisms have been defined. Based on a dedicated GUI-engineering framework it is shown how DSCs can be systematically developed and transformed into appropriate runtime representations. Furthermore, a safety engineering backbone is outlined to support the DSC modeling process in concrete application scenarios. The conducted validation activities show the feasibility and adequacy of the proposed DSCs approach. In parallel, limitations and areas of future improvement are pointed out.

Zusammenfassung

Unter dem Begriff der Cyber-Physischen Systeme hat sich ein zunehmend wichtiges Forschungsgebiet entwickelt, mit dem Ziel die Konnektivität und Interoperabilität von bisher getrennten Systemfunktionen zu verbessern. Heute ermöglichen die fortschrittlichen Vernetzungs- und Verarbeitungsfähigkeiten von eingebetteten Systemen den Aufbau stark verteilter, heterogener Systeme von Systemen. In solchen Konfigurationen endet die Systemgrenze nicht unbedingt mit der Hardware, sondern kann auch den weiteren Kontext wie Menschen und Umweltfaktoren einbeziehen. Da viele der potenziellen Anwendungsbereiche inhärent sicherheitskritisch sind, muss sichergestellt sein, dass die notwendigen Änderungen im individuellen Systemverhalten sicher sind. Allerdings sind die derzeit verfügbaren state-of-the-practice und state-of-the-art Ansätze zur Sicherheitsgewährleistung und Zertifizierung in diesem Kontext nicht anwendbar.

Um einen praktikablen Lösungsansatz anzubieten, stellt diese Arbeit einen Framework vor, der eine "Just-in-time"-Sicherheitszertifizierung für das dynamische Anpassungsverhalten von vernetzten Systemen ermöglicht. Als zentrales Lösungskonzept zur Überwachung und Synthese von dezentralem Sicherheitswissen werden dynamische Sicherheitsverträge (DSCs) vorgestellt. Letztlich eröffnet sich damit ein Weg hin zu standardisierten Dienstebereitstellungskonzepten als eine Menge von sicherheitsrelevanten Laufzeitnachweisen. DSCs ermöglichen die modulare Spezifikation relevanter Sicherheitsmerkmale in vernetzten Anwendungen als eine Reihe formalisierter Abhängigkeiten von Anforderungen und Garantien. Die spezifizierten Sicherheitsmerkmale können hierarchisch integriert und mit einer Interpretationsebene für den Zugriff auf den Umfang möglicher sicherer Verhaltensanpassungen verknüpft werden. Auf diese Weise kann das vernetzte Anpassungsverhalten bezüglich der erfüllten DSC-Sicherheitsmerkmale während des Betriebs bedingt zertifiziert werden. Solange der kontinuierliche Evaluierungsprozess ein sicheres Anpassungsverhalten für einen vernetzten Anwendungskontext bereitstellt, kann die Sicherheit für einen vernetzten Systemmodus zur Laufzeit gewährleistet werden. Signifikante sicherheitsrelevante Änderungen im Anwendungskontext können jedoch zu Situationen führen, in denen kein sicheres Anpassungsverhalten für den aktuellen Systemzustand vorliegt. In solchen Fällen können die verbleibenden DSC-Garantien genutzt werden, um optimale Konzepte für die Einschränkung der dynamischen Anwendungen zu ermitteln.

Für die Umsetzung des DSCs-Ansatzes wurden geeignete Spezifikationselemente und -mechanismen festgelegt. Basierend auf einem dedizierten GUI-Engineering-Framework wird gezeigt, wie DSCs systematisch entwickelt und in geeignete Laufzeitdarstellungen umgewandelt werden können. Darüber hinaus wird ein Leitfaden für das Safety Engineering vorgestellt mit dem Ziel den DSC-Modellierungsprozess in konkreten Anwendungsszenarien zu unterstützen. Die durchgeführten Validierungsaktivitäten zeigen die

Umsetzbarkeit und Eignung des vorgeschlagenen DSCs Ansatzes. Parallel dazu werden Grenzen und Bereiche zukünftiger Verbesserungspotenziale aufgezeigt.

Acknowledgements

Without the support of numerous persons and institutions, the work presented in this thesis could not have been carried out. I would like to take this opportunity to express my sincere gratitude for the help I have received in so many different ways.

First of all I would like to thank my doctoral supervisor Prof. Dr. Peter Liggesmeyer for his scientific and methodical support during the entire research phase. My thanks also go to Prof. Dr. Karsten Berns, the second reviewer of this thesis, for his constructive suggestions.

Special thanks also to the whole SEDA working group for the friendly working atmosphere, many valuable discussions and constant helpfulness, which contributed substantially to the success of this work.

I would especially like to thank Patrick Wolf for the excellent interdisciplinary teamwork.

Finally, my warmest thanks go to my parents Beatrix and Andreas and my wife Zai for their continued support during the preparation of this thesis.

Table of Contents

List of Figures

1 Introduction

1.1 Motivation

In recent years an increasing number of technical devices with so-called “smart” capabilities can be witnessed in everyone’s life. At present, for example, wearable devices like smart watches or smart bands are very popular, where “cloud-based” functions upgrade the classic time measurement function with fitness analysis or messaging functions. This development was driven by the evolution of embedded systems, which became more powerful and, at the same time, less expensive. Embedded computing devices enable a better connectivity of diverse technologies implemented on spatially distributed system platforms. Once the systems are interconnected, previously separate system functions can be combined to higher-level emergent functionality beyond the capabilities of single system devices. Due to their ability to integrate shared services, they provide additional user benefit and are referred to as smart devices.

In the industrial sector, the integrated management of physical assets in manufacturing processes together with the cyber computational space of networked systems has become a key research issue. High expectations are linked to related research topics such as Cyber-Physical Systems (CPS) [1]. The deployment of CPS capabilities in today’s production facilities enables the realization of innovative Industry 4.0 applications with significant economic potential [2]. This opens up new possibilities for meeting customer requirements, such as “same-day delivery” or mass individualized products with small production lot sizes [3].

In principle, the technical implementation is made possible by loosening the rigid system coupling of the various technical devices in a production line and, in parallel, by introducing more self-management capabilities for autonomously carrying out local system integration. This can be achieved by removing static system configurations that are usually designed to provide optimal system integration for only one specific task. Examples of static configurations are predefined interaction patterns such as motion sequences of smart manipulators or even fixed placements of manufacturing systems in the overall production line. Implementing self-management features, on the other hand, requires a powerful middleware that ensures the proper coordination of the diverse applications. As a result, the various devices belonging to a production process should become capable of reconfiguring themselves to dynamically changing task requirements and thus enable more flexible manufacturing plants. Ultimately, in future CPS operating environments it should thus be possible to solve arising tasks automatically with the help of emergent system of systems (SoS) configurations.

At the same time, we are currently witnessing a strong shift towards autonomous or highly automated driving in the automotive domain [4]. This development trend becomes possible by integrating a complete software backbone with powerful embedded sensors and processors into the vehicles. In parallel, new communication technologies with low time latencies such as 5G are evolving [5]. These technologies are applicable for vehicle-to-vehicle (V2V) as well as vehicle-to-infrastructure (V2I) communication [6]. The combination of these technical trends forms the basis for advanced collaboration concepts of spatially distributed vehicle platforms and infrastructure equipment. While in the industrial CPS domain the focus is on optimizing manufacturing processes, the networked systems here aim at better coordinated driving processes taking into account the respective driving goals.

Today, several vehicle manufacturers are already introducing real-time warning services to their on-board driver assistance systems in order to improve road traffic safety. For hazard prevention, these technologies enable the exchange of identified threats and further traffic information between the individual road users and infrastructure devices [7]. A disadvantage, however, is that these technologies are generally limited to optimizing safety by restricting the respective autonomous driving functions, e.g. by limiting speed or recommending alternative routes. For future applications, it is planned instead to enable extended autonomous driving functions based on networked autonomous vehicles to further improve safety and performance of road traffic [8]. Ideally, comparable to the objectives of the CPS domain, many flexible networking options can be realized for a multitude of applications. In order to foster this research direction, this thesis therefore has a special focus on the automotive sector, but is not limited to it.

In such applications of networked driving, forwarded external data can be integrated into the existing individual vehicle architecture in a variety of beneficial ways. For example, new spatial perspectives can be made available for a better environmental perception of the individual vehicle platforms. The precise coordination of particular driving processes also seems to be reasonable. This can be flexible groups of collaborating vehicles, which enable platooning processes on highways to safe fuel, coordinated lane changes or coordinated crossing at intersections. Other scenarios, for instance, are car parks, where the car park infrastructure is networked to the vehicle control architecture to selectively influence the autonomous driving behavior. Beyond such road traffic applications, also sound collaboration concepts for commercial vehicles, such as vehicle fleets in the off-road sector, are promising application domains. Here, a temporary virtual coupling of vehicles like harvesters and tractors in the field (or even larger fleets) could facilitate master-slave configurations that can reduce or even eliminate the need for human drivers. In such configurations, it becomes possible for the master vehicle to remotely supervise and control unmanned slave vehicles for a variety of applications.

Naturally, not all specified collaboration capabilities are suitable for all automotive system platforms or application environments. When considering the many different application scenarios, it becomes apparent that introducing a standardized networking technique can be beneficial for the overall interconnection and collaboration of the various systems. Ultimately, this opens up a path towards standardized service provision concepts for the flexible networking of distributed systems. This thesis therefore aims at introducing a harmonized networking approach that can be applied very efficiently across a broad spectrum of networked applications, not only in the automotive domain, but also in the CPS domain. An essential prerequisite for such a networking approach is a consistent specification and evaluation of the networked system features. In order to implement this, the various individual system capabilities for participating in a collaboration process as well as the new possible collaborative system capabilities need to be defined in a uniform manner.

At development time, especially the ability of a straightforward extension or reuse of already specified collaboration capabilities of systems for new application scenarios is important. Choosing a suitable modeling level and granularity could enable a systematic assignment of individual system capabilities to an integrated overall collaboration mode. If necessary dependencies of a networked functionality are only specified informally on an abstract level, it becomes difficult to trace certain collaboration capabilities back to the respective systems involved. The definition of an overall driving behavior for a networked vehicle fleet illustrates this. If the collaboration approach is limited to checking the correct operation of a networked functionality, e.g. predefined interaction sequences for platooning, it becomes difficult to differentiate between the individual collaboration capabilities afterwards and make parts of it reusable. Hence, an explicit specification of the individual contributions of networked systems to the respective collaboration capabilities seems necessary to achieve this goal.

Beyond that, modularization could support the reusability of parts of already specified collaboration processes for slightly different collaboration scenarios. This would allow to systematically consider already specified collaboration capabilities for new potential collaboration partner or application environments. Ultimately, an entire ecosystem of specified collaboration capabilities related to a particular application context could be formed on this basis.

During operation, a crucial factor is that the specified collaboration processes are sufficiently adaptive to meet the demands of the collaboration context. They should provide a suitable networked system behavior for the manifold collaborative situations that may occur in a particular networked application context. Especially in collaboration scenarios in road traffic, it becomes apparent that future networked vehicles have to deal with ever-changing situational goals for optimizing driving behavior during operation. These varying collaboration goals depend on the current driving situation, the available collaboration partners in the surrounding area as well as the individual objectives for current driving processes. If the specified collaboration processes in such a versatile application context of networked driving are not sufficiently adaptive, the respective dynamic driving context cannot be adequately addressed. In consequence, a limited variability of a networked mode caused by strongly preconfigured collaboration processes (e.g. predefined driving sequences) would lead to a very limited application context. It would thus severely restrict the overall potential to benefit from collaborative knowledge. It becomes visible that a suitable networking approach for enabling powerful collaboration processes needs to be largely dynamically adaptable during collaboration to ensure an efficient use in the diverse collaborative situations.

In order to provide a suitable solution approach, it seems promising in the automotive sector, similar to the presented industry research strategy for CPS, to meet the manifold and rapidly changing requirements for collaboration processes with an approach of loosely coupled systems along with more self-management capabilities. In this way, collaboration processes of networked systems could become more flexible and dynamically adaptable to a wide variety of collaboration scenarios and thus more powerful and efficient during operation. Consequently, this thesis focuses on the individual self-adaptive system platforms involved in collaboration processes. The respective automotive systems should optimize their driving behavior with external runtime knowledge in a reliable way during operation to enable the described networked applications.

In contrast, strongly preconfigured networking concepts, such as networked vehicle fleets with pre-defined motion sequences, limit the autonomy of the individual involved systems. These integration concepts correspond more to traditional concepts, such as conventional industrial manufacturing processes using static system configurations. Their high degree of customization for modeling the overall dynamic interdependencies of involved systems, makes a lightweight extension to other collaboration scenarios difficult to achieve. For the envisaged networking approach they can only be considered as individual, non-accessible networking options.

Since most of the required hardware resources are available in future autonomous vehicles, a vital precondition of connecting new system platforms and functions to existing collaboration structures is, as outlined before, a harmonized networking approach for autonomously carrying out collaboration processes. A broadly applicable networking technique especially requires a consistent specification and evaluation approach of the overall collaboration capabilities with respect to the individual system capabilities. In addition, the specified collaborative functionality has to be sufficiently adaptive in the respective changeable application context to enable dynamic adaptation processes during operation. Once accepted by certification bodies and industry, such an approach could accelerate the widespread use of the described promising networked applications. Otherwise, if no adequate standardization efforts are carried out, the opportunities for collaboration scenarios of networked systems could, even in the long term, be limited to a few proprietary solutions with a limited number of participating systems.

1.2 Problem Statement

All of the described use cases have in common that autonomous vehicles need to be sufficiently adaptive to change their driving behavior in networked driving modes. A major challenge for the practical application of runtime adaptations of the respective driving behavior results from the safety-critical context in which these systems typically operate. Inappropriate changes in driving behavior can lead to physical or material damage in the dynamic application context. Hence, behavioral adaptation processes based on the integration of external services must always be performed in a correct and predictable manner. Without an appropriate proof of safe behavior generation within an autonomous system, it will not be possible to justify the use of such promising networking techniques in public space. In particular, the respective certification bodies will insist on a proof of the system's safety, taking into account all conceivable hazardous situations. A decisive factor for applying such flexible collaboration approaches of distributed systems in public space is hence the certifiability for safety.

Ensuring safety for a modified driving behavior of networked vehicles in an open application context such as public road traffic, however, is a complex task due to the inherent uncertainty of operating conditions. The intended collaboration of vehicles and infrastructure can be affected by a variety of inevitable external influences from the surrounding environment. This can be unfavorable weather conditions, inappropriate driving behavior of surrounding vehicles and other reasons that are sometimes difficult to foresee. Unfortunately, the vehicle platform design cannot be modified in such a way that hazards arising from external influences in networked driving processes can be completely excluded. Accordingly, the likelihood of recurring system restrictions for such networked system applications is probably higher and more difficult to predict than in well-defined application scenarios without external influences (e.g. conventional manufacturing processes). It becomes apparent that the open and dynamically changing context makes optimal system integration (vehicles and infrastructure) for collaboration processes more difficult compared to a closed system context, which is not affected by external environmental challenges.

As a result, the absence of such external influences while driving in a networked mode has to be continuously monitored in order to ensure safety. Furthermore, if certain safety-critical collaboration capabilities are suddenly degraded in a networked driving process, a timely and situation-adapted reaction behavior has to be carried out in order to prevent collisions. In such a case, especially the availability of suitable emergency driving maneuvers decides whether the respective situational requirements for hazard prevention can be met. In other words, whether such external effects become hazardous for a networked vehicle depends primarily on the vehicle's safety concept for autonomous driving.

In an increasingly hazardous situation in a networked driving mode, however, the existing safety concept of the individual vehicles, specified for non-networked autonomous driving, cannot guarantee safety for a modified driving behavior. This is due to the fact that a modified driving behavior results in new collaborative driving situations. For example, when crossing an intersection in a coordinated manner without braking, new hazardous driving situations (considering the oncoming traffic) become possible. These new hazardous situations within a networked driving mode must be prevented for safety reasons by introducing new safe driving maneuvers. These driving maneuvers have to be adapted to the specific networked driving states. In consequence, the existing vehicle safety concept for individual autonomous driving must be adapted to the new application context when systems are being networked. An important prerequisite for achieving this is that the modified safety concept in a networked driving mode can be appropriately integrated into the existing vehicle safety concept and that they do not interfere with each other. This especially requires a continuous re-approval of the currently valid safety concept in a networked driving mode as well as a reliable dynamic adaptation concept of the individual driving behavior.

For the application of today's state-of-the-practice safety assurance and certification techniques, this is a major difficulty, as they rely on safety measures that are carried out completely at development time. Approaches for safety assurance and certification at runtime for dynamic adaptation processes of system behavior are so far not applied yet. Today's safety assurance techniques are aimed at identifying and correcting weaknesses in the system architecture before the system is put into operation. In this way, potential risks during operation should be reliably reduced to an acceptable level. The proposed collaboration processes, however, should be established in a flexible and self-managed manner at runtime whenever potential collaboration partner are within the communication range. When the envisaged flexible networking approach in combination with the highly versatile application context is analyzed for conceivable driving scenarios at development time, this leads to a multitude of possible collaborative driving situations. The attempt to guarantee safety before operation only with development time means would lead to a state space explosion of possible situations to be considered. In addition, it would be difficult to ensure that the safety concept for emergency driving maneuvers of networked vehicles is sufficiently adapted to the specific driving situations. As a result, the currently available safety assurance and certification techniques would thus, for safety reasons, severely restrict the potential networked driving modes.

It therefore seems inevitable to introduce new safety assurance and certification approaches, where suitable safety measures can be shifted to runtime. In this way, decentralized safety knowledge could be combined to optimally cope with emerging hazards in networked driving processes, when they occur, at runtime. Ultimately, a viable solution approach should enable safety certified and, at the same time, dynamically adaptable networked driving processes of vehicles in the versatile application context. Hence, the reliable runtime integration of external safety-related data into the individual vehicles for the derivation of a suitable networked driving behavior becomes a key issue. Safety relevant features of a networked mode could be identified at development time and correspondingly taken into account for certification issues and the derivation of an appropriate driving behavior at runtime.

1.3 Thesis Goals

It becomes apparent that today's safety engineering activities, which focus only on development time safety measures, are inappropriate for the considered emergent collaboration processes of distributed systems. The envisaged networking approach should provide flexible networking options considering different collaboration partners and objectives in versatile application environments. In the considered scenarios of the automotive domain, the existing safety concepts for non-networked autonomous driving cannot ensure safety for networked driving applications. Accordingly, as described in the previous section, adapted safe driving maneuvers are required for the respective networked driving modes. This requires new solution approaches that enable safety-certified dynamic adaptations of the driving behavior of networked autonomous vehicles in the open and dynamically changing application context.

If parts of the safety assurance activities could be shifted to runtime, the additional runtime knowledge could help to better cope with safety-critical networked driving states. Hence, a promising solution approach would be to monitor safety relevant networked features of involved systems to ensure that the overall collaboration of systems is safe. Furthermore, a reliable assessment of the current application context could be the basis for identifying the permitted degrees of freedom during operation of a networked mode. Ultimately, it could thus become possible to conditionally certify already at development time the dynamic adaptation behavior of involved systems with respect to certain networked driving states. By checking the respective validity of networked driving states during operation, safety for the active operating mode could then be certified at runtime.

Using runtime safety mechanism, however, it has to be specified, which parts of the safety assurance activities are shifted to runtime or remain at development time. Finally, the respective partitioning decides about safety and efficiency of the proposed networking technique. To establish system networks that are highly adaptive during operation, it is beneficial from performance perspective to shift as much safety assurance activities as possible to runtime. Thus, the flexibility of system adaptations can be adjusted in an arbitrary way. It has to be noticed, however, that the necessary safety engineering steps and the subsequent certification of systems and applications are typically challenging and also creative tasks, even for safety experts. In such certification processes, a comprehensive decision taking is required, where automated decisions using runtime evaluations are strongly limited. Hence, it is generally favorable from a safety perspective to keep as much safety assurance means as possible at development time and to minimize the amount of runtime safety assurance responsibility.

Due to the dynamic changes in the application context, however, it turns out that a key objective to enable safe collaboration processes of systems is to merge the distributed safety knowledge within the collaboration group when it is available, at runtime. In this way, it would also be possible to take into account the general availability of the surrounding systems for collaboration processes as well as their individual system capabilities when establishing certain networked driving modes. An essential prerequisite for merging distributed runtime knowledge is, of course, that it is determined in advance how this is carried out. In principle, this could be facilitated by lightweight runtime checks. It is therefore particularly important to provide a suitable linking logic for the available safety related runtime data from the various sources.

The following research challenges have to be addressed in order to derive a suitable solution approach for the safe networking of systems in the described application context:

1. With the aim of optimizing driving behavior of autonomous vehicles by considering external data in a safety critical context, behavioral adaptation processes must be made possible in a certifiable way. This requires a conclusive safety argumentation for a reliable integration of external runtime knowledge into a dedicated system platform along with a predictable and correct derivation of a networked system behavior.

2. From a certification point of view, the derivation of the modified overall safe system behavior should be traceable and verifiable as far as possible. Hence, for safety engineering and certification, the integration of external services should preferably be carried out in an incremental manner in combination with a transparent cause-effect relationship. A modular *Divide and Conquer* approach for a stepwise system integration could support this goal. It could provide the framework for testing and certifying particular integration steps of external services regarding the envisaged networked system modes.

3. The integration of the various external services to a specific system platform should always be considered with regard to the overall system safety as well as performance and the derivation of a resulting safe system behavior. As this is usually a creative development process for safety engineers, this cannot be automated in an arbitrary manner. In addition, it can be assumed that the dependency modeling between the various available services leads to a high degree of complexity. For this reason, a systematic modeling approach for the specification of optimal networked system modes seems necessary. In this way, it should be ensured that the available information is taken into account as optimally as possible in order to improve the safety and performance of the networked systems.

4. A “Graceful Degradation” concept should also be part of the specification of networked system modes. As already outlined, due to external influences in the open application context of networked driving processes, recurring restrictions of networked system capabilities cannot be completely avoided. Analyzing the recurring system restrictions in the design phase could be the basis to specify still safe, optimal degradation levels of system behavior for performance issues.

5. Customized emergency driving maneuvers for the modified vehicle's safety concept for autonomous driving in a networked mode should also be addressed in such a networking approach. In networked driving processes, it must be ensured that the system is able to optimally cope with emerging hazards that occur in the dynamic application context at runtime. Even if safety certification for a networked mode is no longer possible, a strategy should exist to reach a safe state. In particular, spatial and temporal constraints resulting from the dynamic application context should be considered to mitigate such hazards through appropriate reactive system behavior.
6. To increase the overall applicability and the operational effectiveness of the envisaged networking approach, many flexible networking options should be made available. To achieve this, an efficient specification technique for the divers system networks is required. Specifying networked system modes in a unified way seems necessary for this purpose. This could enable the reusability of parts of already specified networking features for different system platforms and application scenarios. Moreover, a standardized service provision concept for networked systems could be established in this way. Thus many potentially participating systems and application environments could be addressed. Ultimately, this can lead to a higher utilization rate of a particular networking technique.

The research focus is therefore on an adequate integration approach for external services on a dedicated system platform with the described features. In this way, safety-certified networked driving processes for optimizing safety and performance of autonomous vehicles should become possible. As a first solution approach, this thesis presents in the following Chapters the conceptual framework of *Dynamic Safety Contracts*.

1.4 Scientific Contribution

This thesis introduces a new contract-based safety assurance and certification approach called *Dynamic Safety Contracts (DSCs)*. With the DSCs approach, it becomes possible to reliably merge distributed or decentralized safety-related runtime knowledge in order to dynamically adapt the behavior of networked systems. The DSCs approach provides a systematic composition framework for safety-related services from distributed systems (e.g. vehicles and infrastructure) to specify safe networked system modes before operation at development time. And, associated to this, it provides a lightweight runtime evaluation mechanism that enables the specified networked system behavior to be performed in an autonomous manner.

Introducing a formalized service provision concept according to their contained safety quality characteristics makes it possible to treat shared safety-relevant data from different sources in a unified way for safety assessment. This becomes possible since the spectrum of potential output data from networked systems is known before operation and can be classified accordingly. This can be, for instance, networked safety-certified sensors as well as their measurement data in a specific application context. In this way, the shared safety-related services can be considered as safety-related runtime evidences. This in turn opens up the opportunity to introduce comprehensive safety assurance and certification concepts based on flexible system networks. Safety engineers can thus define at development time which collaborative system functionalities can be safely executed with respect to available safety evidences at runtime. As a result, networked system modes can be systematically developed according to networked system platforms and application environments.

DSCs are established in a modular way to explicitly merge formalized services containing safety quality characteristics that are related to each other in a specific networked application context. The encapsulated merging logic supports thus reuse and co-development processes. This allows safety experts from different domains to contribute their particular expertise either independently of one another or in parallel. In this way, an entire ecosystem of specified networking capabilities could be derived to provide many flexible networking options.

The developed DSC-Modules and their internal merging logic can be rigidly connected to each other to consider dynamic changes in the service availability during operation of a networked mode. The rigid allocation of DSC-Modules supports a fast and reliable propagation of merged services to the control level of the networked system behavior. The result are reproducible evaluation mechanisms based on pre-configured variability which can guarantee safety for dynamic adaptation processes of networked systems. Which configuration is currently valid is proofed at runtime based on the available top-level safety guarantees.

In this way, the merged services using DSCs enable a joint safety assessment of networked systems in an open and dynamically changing application context at runtime. They represent valid safety contracts for a networked system mode and provide suitable safety assurance and certification means for the execution of networked system functions. The specification is achieved by applying the DSC merging logic for integrating external services while evolving a safety assurance concept. The available composed services during operation of a collaborative mode are strictly related to the allowed degrees of freedom for networked system functions. The evaluated degrees of freedom can be compared to the current networked system state. This enables to introduce a holistic safety assurance approach, which also contains an active influencing of the networked system behavior according to safety aspects. The derived behavior adaptation concept is part of the DSC service composition concept and can be individually assigned to a particular system in a specific networked mode.

In a regular networked system mode, where no hazardous event occurs, the composed services are facilitated to provide safety certified enhanced system capabilities. To facilitate this, a continuous evaluation of allowed degrees of freedom of networked systems is conducted. In the ideal case (no degradation of collaboration capabilities) the collaborative capabilities with DSCs exceed the individual system capabilities with additional degrees of freedom. The available merged services using DSCs thus enable an optimized networked system behavior. For instance in a vehicle platoon, the networked driving mode provides optimized set points for reduced driving distances or higher average velocities.

In case of suddenly severely degraded collaboration capabilities, recognized based on degraded, missing or inconsistent shared services, the allowed degrees of freedom are reduced and the existing system coupling is loosened. Depending on the criticality of such missing collaboration capabilities, a rapid reaction behavior adapted to the collaborative mode can be part of the safety specification and can be initialized accordingly. It should ensure a return to a safe state for a system during operation of a networked mode as optimally as possible. In a platooning scenario, for instance, adequate breaking and evasive maneuvers with respect to the surrounding vehicles, e.g. to enable a standstill on the emergency lane, could be specified to reach a safe state in the networked driving mode.

Openness for previously unknown collaboration partner as required in road traffic applications, is also addressed by the DSCs approach. To facilitate this, the DSC approach facilitates a two step approach for safety assured collaboration processes. In the first step at integration time, before the actual collaboration is carried out, it can be determined in a reliable way which collaboration mode optimally fits for the collaboration of the particular systems. The DSCs approach therefore facilitates a negotiation process based on the respective hardware capabilities of involved systems. To the identified collaborative mode is the DSC merging logic assigned, which assures in the second step at runtime of the collaborative mode a safe operation as outlined before. In this way, it is tried to keep as much safety knowledge as possible at development time and keep the runtime model lightweight. In other scenarios, where collaboration partners and their collaboration context are already known before operation, a specified DSCs collaboration approach can be directly applied without facilitating a negotiation process in advance. Typical scenarios for this case are fixed configurations of networked vehicle fleets such as master-slave driving in the commercial vehicle domain.

1.5 Validation of the Presented Approach

At the moment, there is to the best of our knowledge no comparable approach available that enables the described flexible collaboration processes of distributed systems based on a standardized networking approach for available in- and external safety-related services. Thus a comparison to similar approaches cannot be conducted at this point. In this light a main task of this thesis is to show the applicability and feasibility of the proposed DSCs approach. For the specification and implementation of DSCs, a comprehensive insight and tool support is provided in this thesis. The respective steps were analyzed and validated with adequate test techniques.

To ensure optimal engineering processes for the DSC modeling, a systematic safety engineering strategy for DSCs has been established. It was validated with several use cases from the automotive domain. Furthermore, a dedicated GUI framework was developed to ensure an efficient DSC modeling. The GUI provided in this thesis was validated for different modeling scenarios. A key feature of the GUI is the automated code generation from the graphical representations of DSC-Modules. The suitability of the generated runtime representation of DSCs was validated considering safety and performance issues. The runtime mechanism of the DSC evaluation, as a key element for contract building, was analyzed according to its computation times for the overall DSC-Module propagation times as well as the evaluation times for the overall feedback behavior. The analysis findings showed that sufficiently short computation times can be achieved using the DSCs approach. Thus adequate evaluation times for the feedback behavior could be ensured.

To show the feasibility of the runtime safety assurance mechanism based on DSCs, an implementation of networked vehicles into a holistic 3D simulation environment based on the Unreal Engine was realized. The RRLAB department at TU Kaiserslautern developed the realtime robot control framework Finroc [9], which was facilitated for the DSC approach in combination with an integrated behavior-based control (iB2C) [10]. The simulated driving processes are particularly realistic, since the changing output guarantees directly influence the input parameters for DSC evaluation with a closed loop control architecture. The iB2C approach enabled the autonomous driving mechanisms for two vehicles in a platoon, while the DSCs approach provided the corresponding safety assurance and certification means for the networked driving processes. In the simulation environment, it was possible to create hazardous driving situations in a targeted manner. In this way, the suitability of the networked driving behavior could be checked based on the overall vehicle interaction sequences. Moreover, the correct reaction behavior in unsafe driving states could be observed in order to put the networked systems back into a safe operation mode.

Beyond that, the negotiation mechanism for a suitable networked DSC driving mode to address openness was validated with a dedicated simulation framework based on a platooning use case. Previously unknown vehicle systems and infrastructural devices could be inserted into a simulation environment with a simplified one-dimensional road via drag and drop. Subsequently, the optimal networked driving mode is automatically identified with a negotiation process based on the available collaboration capabilities and is put into operation with DSCs. During operation of a networked driving mode, the safety-relevant driving parameters could then be selectively modified. A visualization of the internal DSC signal propagation showed the effect on the resulting overall driving behavior. In this way, both the ongoing DSC runtime evaluation as well as the initialization and resolution of the DSC networking modes could be validated.

1.6 Thesis Structure

This thesis is structured into seven Chapters. After the introductory Chapter, the fundamentals of the given research subject are explained in more detail in Chapter 2. Moreover, a comprehensive insight into the related work is given. A special focus is on the current state of the art from the most contributing research communities as they are naturally the safety community and the open adaptive system community. Chapter 3 provides a brief overview of the solution approach. The envisaged application context and derived key features for a suitable solution approach are presented. On this basis, the conceptual framework of DSCs is introduced as the core solution approach. In the following, the specific characteristics of DSCs with regard to specification and implementation are described in Chapter 4. To ensure efficient modeling processes, this Chapter also presents a dedicated GUI-based modeling framework for automated code generation and for efficient runtime evaluations using BDDs. Chapter 5 then shows how a systematic development process for the specification of collaboration modes based on DSCs could be established. A special focus is on the joint consideration of the domain-level and the system-level engineering in the development process. In Chapter 6 the validation activities for the proposed DSC approach focus on the general applicability of the development time specification approach and the suitability of the runtime evaluation mechanisms. To demonstrate the general applicability of the DSC modeling approach as well as an efficient reuse of parts of the DSCs specification, the various scenarios in which the approach was utilized are summarized in this Chapter. For the validation of runtime mechanisms, the conducted simulation activities are presented in this Chapter as well. The main focus is on validating the intended dynamic behavior of networked vehicles as well as the evaluation times and efficiency of the runtime mechanism. The ability to be open to previously unknown participants based on runtime negotiation processes is also demonstrated. Finally, Chapter 7 summarizes and discusses the results achieved as well as potential future research aspects.

2 Related Work and Fundamentals

Overall, this Chapter provides a comprehensive insight into today's safety engineering and certification techniques to enable a reliable networking of distributed systems. As described in the introductory Chapter, an adjusted and holistic safety concept is crucial for the integration of external services into self-managing systems such as autonomous vehicles. In this context, the fundamental principles of collaboration and self-optimization of networked systems with the aim of facilitating dynamic behavioral adaptation processes are outlined at first. This Chapter thus forms a good starting point for understanding the solution approach that is derived in the next two Chapters.

Section 2.1 introduces the fundamental prerequisites for achieving benefits from the networking of distributed autonomous systems. Initially, the various levels of vehicle automation as well as a terminology for distinguishing the different possible integration levels of networked systems are presented. Then, in order to illustrate the incorporation of forwarded external data in individual self-managed systems, the working principle of self-optimizing systems with continuously recurring evaluation processes is explained in more detail. The underlying mechanisms for the technical implementation of external service integration in autonomous systems are outlined as well. The following Section 2.2 provides an overview of the current state-of-the-art and state-of-the-practice of safety assurance approaches for dynamic behavior adaptation. Here a distinction is made between the two particularly affected research groups, which are the adaptive systems community and the safety community. The adaptive systems community especially takes into account safety assurance approaches that allow a high dynamic variability by trying to solve safety issues at runtime. In parallel, the safety community is focusing on more conservative approaches with the goal of ensuring safety completely through development time measures. Beyond that, in order to derive a suitable solution approach later on, safety engineering approaches for modeling data flow structures as well as runtime evaluation approaches are presented here. Hence, Section 2.2 is also well suited to retrace the design decisions presented in Chapter 3 and 4 in relation to previous work.

2.1 Fundamentals of Networked Autonomous Systems

A key objective of current digitization trends towards Cyber-Physical Systems [1] is to improve the interconnection of previously separated system functions and thus enable new joint capabilities for the systems involved. In the industrial sector, such networked applications in combination with self-reconfigurable manufacturing systems enable Industry 4.0 [2] applications. In the transport domain, there are also strong research efforts aiming at a better networking of vehicles and infrastructure in order to increase road traffic safety and efficiency [11]. Modern, highly automated vehicles are getting more and more similar to complex networked mechatronic systems. A multitude of research initiatives can be summarized here under the term Intelligent Transportation Systems (ITS) [12]. They deal with a broad range of technologies and applications for networked road transport. These are for example wireless communication technologies as well as networked applications for automatic road enforcement or variable speed limits. In the dynamic and safety-critical application context of public road traffic, the reliable and safe generation of emergent driving behavior while being involved in a group of networked systems is of vital importance. For the realization of these promising networking concepts, it is therefore essential that the individual autonomous vehicle platforms can interpret the forwarded services correctly and are sufficiently self-adaptive to change their driving behavior in a suitable fine-grained and targeted manner.

The envisaged networked system applications in this thesis can only be achieved with a sufficiently high degree of autonomy for the involved systems in combination with a tight system coupling. To illustrate this, the different automation levels in the automotive sector are presented in a first step. The focus here is on automotive applications as this is the target application domain of this thesis, but is not necessarily limited to it. Subsequently, the various integration levels of networked applications are discussed. In particular, this should clarify the use of a uniform terminology as so-called *collaborative networks* throughout this thesis. In the following, the fundamental steps of behavior generation processes are outlined in order to enable a clear differentiation of internal mechanisms for system adaptations. Since the focus is on optimizing system capabilities through self-adaptation processes, this is explained with reference to the theory of self-optimizing systems. The related system architecture is well suited for establishing emergent and highly adaptive networks of collaborating systems. Afterwards, the different perspectives of behavioral adaptation processes are examined, as they are addressed by the intelligent system community and the adaptive system community. Furthermore, additional insight for the implementation of interdisciplinary aspects are outlined. This Section also discusses already implemented application scenarios as well as their operating principles.

2.1.1 Automation levels in the automotive sector

In order to dynamically adapt system functions for networked applications during operation, the respective systems need sufficient self-management capabilities. In applications in which the driving behavior of autonomous vehicles should be optimized, as described in the introductory Chapter, the respective driving function must be at least partially self-configurable. In Table 2.1 the automation levels for automotive platforms as specified by the SAE [13] are outlined. Whenever an automotive system, as shown in the table, can execute a certain driving task without human assistance, it is basically able to adapt itself autonomously to new tasks in the given application context. Hence, self-managed system functions in a certain application context are basically operational starting from automation level 1 as shown in Table 2.1.

This thesis, however, focuses primarily on enabling better coordinated driving processes by modifying the individual driving behavior of autonomous vehicles. Here, particularly a modified safety concept based on adapted driving maneuvers for autonomous driving is required. The autonomous driving functionality should be capable of performing all of the dynamic driving tasks also in unsafe networked driving situations. This is necessary because a human driver is not always able to take over control fast enough and react correctly. Dealing with the multitude of information in a hazardous driving situation is particularly difficult for the driver when he is distracted [14].

Up to level 2, safe driving maneuvers can only be ensured by a human driver. Also in level 3, the human driver is responsible for safety although some basic elements of safety assurance can be shifted to the autonomous driving behavior. Starting from level 4, the autonomous vehicle manages safety assurance for specific driving maneuvers completely on its own. Here, where no human assistance is required for transferring the system to a safe state, the safety concept for automated driving tasks can be readjusted accordingly. Consequentially, this thesis focuses on level 4 and 5 of automated driving systems (ADS). Ideally, since the intended networking approach for flexible collaboration should be as widely applicable as possible, the level 5 automation type is applied, as it can be activated everywhere and is thus most effective.

Level	Automation Type	Where Operational	If Automation Stops Working
0	No driving automation	Not applicable (no automation)	Not applicable (no automation)
1	Driver assistance	Limited roads or modes	Driver resumes performing all of the dynamic driving task
2	Partial driving automation	Limited roads or modes	Driver resumes performing all of the dynamic driving task
3	Automated driving (conditional)	Limited area, roads, and/or modes	Driver takes over after warning
4	Automated driving (high)	Limited area, roads, and/or modes	ADS brings vehicle to safe stop
5	Automated driving (full)	Everywhere on-road	ADS brings vehicle to safe stop

Table 2.1: Levels of Vehicle Automation (adapted from [13])

2.1.2 Integration levels of networked systems

In order to identify a suitable approach for sharing of safety-relevant knowledge among networked systems, it is important to know which features characterize networks at different integration levels. It became evident during the literature research for suitable definitions and delimitations of the terms *collaboration* and *cooperation* that they are often used indistinguishably. In this thesis the definitions from Camarinha-Matos et al. [15] are used due to the good overall thematic classification. The research group ranked the coalition types of networks according to their level of integration into four different categories, as shown in Fig. 2.1. As a rule of thumb, it can be noted that in this approach a higher integration level always inherits the capabilities of the lower levels.

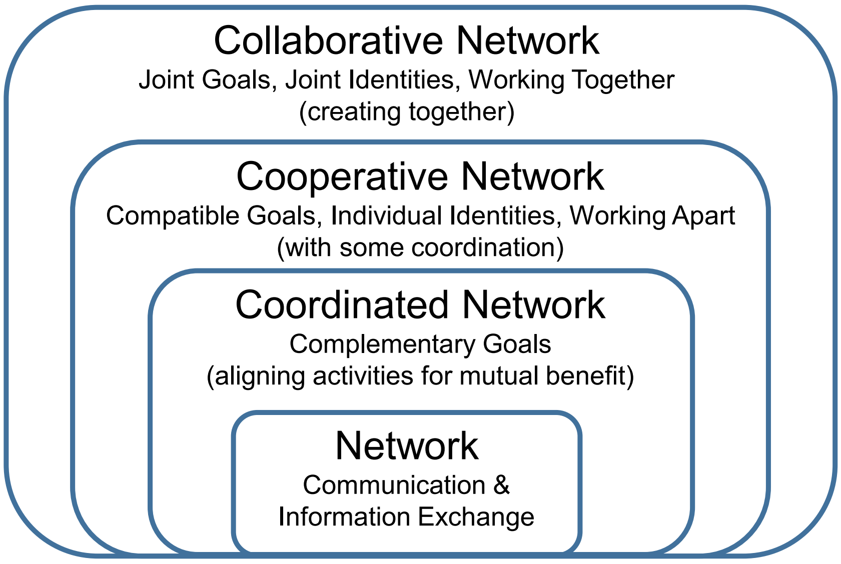


Figure 2.1: Integration Levels of Networks (adapted from [15])

At the first integration level, called *Networking*, communication and information exchange already takes place, which is the basis for mutual benefit of involved systems. However, neither a common goal nor a structure for individual contributions of systems has been defined. At the next level *Coordination* an aligning of activities takes place, so that information exchange becomes considerably more efficient. The resulting advantage, however, usually remains at the individual level since the objectives of the systems involved are typically very heterogeneous. The next integration level *Cooperation* focuses on sharing of resources of the involved parties in order to achieve compatible system goals. Although involved participants perform their work separately, a common plan exists to aggregate generated value of the various system to a value chain to reach the respective system goals. At the highest integration level *Collaboration* all activities are done jointly by the involved partners to reach a common goal. Hence, when collaborating, involved systems work together (co-labor) on a single shared goal. It represents a process, where a collaboration group enhances the capabilities of each other.

With regard to the initial statement mentioned above, it becomes apparent that the assignment of the two terms *collaboration* and *cooperation* to applications strongly depends on the respective perspective. Saekia [16] for example pointed out that software development activities are cooperative work but collaboratively performed by a team with various roles of persons and tools. Denise [17] analyzed these terms with regard to the resulting workflows of human employees and noticed that the three Cs (Communication, Coordination, Cooperation) support controlling tendencies that have a centralizing effect. In contrast to that *Collaboration* is about creating new value.

A typical use case in this thesis is the exchange of safety-related data to optimize driving processes of individual or entire groups of autonomous vehicles. Such applications can be preconfigured vehicle fleets such as master-slave vehicle configurations, which are presented later in the validation Chapter. In such scenarios, the master vehicle determines the respective system goals for the slave vehicles. In parallel, the autonomously driving slave vehicles need to find the best possible way to contribute to the group goals with respect to the other systems. Such vehicle groups thus clearly form a collaboration network. In other applications, such as typical road traffic situations, however, the term cooperation also seems sensible since individual goals for driving processes are pursued.

Generally, it is however true that with a higher level of integration from *Networking* to *Collaboration* the commitment for responsibility of involved participants for each other increases. The fact that this work is specifically focused on safety-assured networking concepts in safety-critical applications and that there is a high degree of mutual dependencies should be highlighted throughout this thesis. Similar to preconfigured vehicle fleets in off-road applications, ensuring safety and efficiency in public road traffic can be identified as a joint goal of available autonomous vehicles. Furthermore, referring to Denise, flexibility of emerging and decentralized networks could only be maintained in collaborative networks in contrast to centralizing and hierarchical communication structures from other less integrated networks. That´s why the term *collaboration* is used consistently in this thesis instead of *cooperation*.

2.1.3 Enhanced situation awareness and self-optimization

In the intended application context of collaboration networks it should be possible to share safety-relevant services between the networked systems in order to improve the individual situation awareness of autonomous vehicles. In this way, by specifying modified driving modes taking into account the improved situational awareness, existing driving processes should be optimized. If joint system knowledge is continuously considered for behavioral adjustment processes of existing driving behavior in autonomous systems, this is referred to in this thesis as self-optimization.

An important prerequisite for improving individual situation awareness is the correct decision-making taking into account external data in addition to the available internal data in the various dynamic driving situations. According to Endsley [18], situation awareness in a dynamic context consists essentially of three steps. The first level is the perception of the considered elements in the environment. An autonomous vehicle needs to know, next to its own status and dynamics, where other vehicles and obstacles are located as well as their status and dynamics. The second level is the comprehension of the current situation, where the disjointed elements are linked to form a holistic picture of the environmental situation. The third level is the projection of the current holistic picture to a future status. Possible future collisions can be detected and avoided by suitable adjustments of the current driving behavior.

Along with the implementation of an interpretation layer for situation awareness, the feasibility of self-optimization by external services strongly depends on the respective self-management capacities of systems in the dynamic context. In the following, the fundamental steps to enable self-optimization processes for ever-changing driving situations are described. The enhanced situation awareness together with the analysis of the fulfillment of the pursued goals represents here the "Analysis of the Current Situation" as a first step. The additional knowledge regarding the current situation is thus the basis for further adapted cyclical optimization steps. According to Frank et al. [19], self-optimization takes place in a system when the three actions and their descriptions[1] listed below are continuously repeated in a system:

1. Analysis of the Current Situation: The actual situation considered in the first action includes the state of the system and all possible observations about its environment. Observations can also be gained indirectly through communication with other systems. The status of a system also includes any past observations that may have been saved. An essential aspect of the analysis is the examination of the degree of fulfillment of the goals pursued.
2. Determination of the System Goals: In the second action, the new goals of the system can be determined by selecting, adjusting or generating them. Here, selection means the choosing of an alternative from a fixed, discrete, finite set of possible objectives. An adjustment of targets, on the other hand, describes the gradual change of existing targets. The term generation of targets is used if they are regenerated independently of the previously known ones.
3. Adaptation of the System Behavior: The adaptation of system behavior is determined by the three aspects parameters, structure and behavior. The third action describes the final feedback effect of the self-optimization cycle by adapting the system behavior. The individual cases of adaptation can vary greatly depending on the implementation level of a mechatronic system. The domain in which the adaptation is implemented also plays an important role.

Kephart and Chess [20] introduced the so-called MAPE-K loop (Monitor, Analyze, Plan, Execute and a Knowledge Base) as a widely accepted reference architecture to specify self-management capabilities for systems such as self-optimization processes. It can

[1]Translated from the german description [19]

be applied to assure safety for each individual steps of system adaptation. Originally, they stayed on a very abstract level with respect to autonomous computing elements in complex software. Later, their reference model was further refined from the adaptive systems community to the MAPE-MART (MAPE-Models@Runtime) loop as outlined in Section 2.3.

In the following, in Fig. 2.2 a similar reference architecture with a stronger focus on system interactions with the environment is shown, as introduced by Meystel and Albus [21]. In contrast to the MAPE-K loop this reference architecture highlights the interaction of the autonomous system with its environment using a separate sensor/actuator interface layer. As shown in Fig. 2.2, here the monitoring level of Mape-K is subdivided into two levels as *Sensors* and *Sensor Data Processing*. This makes the consideration of sensor data from different sources for a particular system in the addressed target application context more illustrative. Meystel and Albus refer to such systems as “Intelligent System”, since the complexity of decision making for autonomous interaction with an open environment is much higher than self-adaptation processes to a well-defined set of elements.

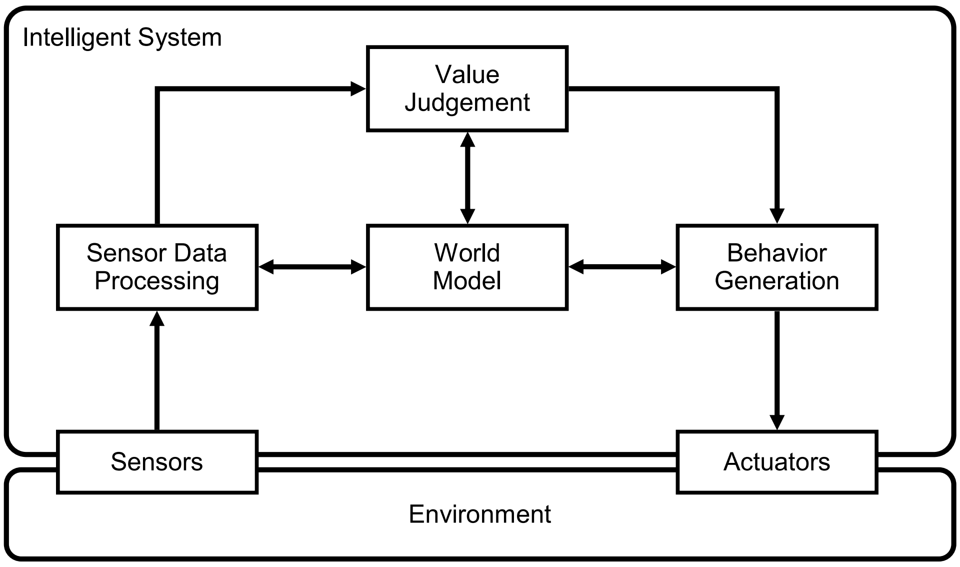


Figure 2.2: Reference Architecture of Intelligent Systems (adapted from [21])

This thesis deals with all three internal steps *Sensor Data Processing*, *Value Judgement* and *Behavior Generation*. If shared knowledge should be considered for the analysis of the current situation, a reliable merging concept has to be established for the additional consideration of external knowledge in addition to internal knowledge. The decision-making for adaptation processes, from the analysis of the current situation to the value judgement and the behavior generation, should have a clear cause-and-effect relationship to ensure that the behavior optimization of systems is carried out in a reliable manner. A prerequisite to enable this is the harmonization of shared sensor data. Consequently, the formalization of sensor output data is also part of this thesis.

2.2 Background on Safety Assurance for Dynamic Behavior Adaptation Using Shared Services

Although there are many research projects focusing on communication concepts for autonomous vehicles, the question of safety assurance and certification for establishing broadly applicable networked driving modes based on standardized shared services has been neglected to a large extent. It can be assumed that one reason for this is the necessary high degree of autonomy for automated driving as a prerequisite for the intended networked driving modes. As described in more detail in the next paragraph, most of the associated application scenarios for highly automated driving are not yet mature enough for practical use. Thus, for practical application, safety and certification issues for networked driving do not necessarily have to be clarified at this stage of technical development. Nevertheless, this development status can change rapidly with ongoing technical improvements.

At present, the supervision of autonomous vehicles in public space by human drivers is typically mandatory [22]. Here, the human driver must be able to take over control at least after warning. Exceptional cases such as extremely slow driving autonomous buses are not considered here, since the required driving tasks are not sufficiently dynamic or safety critical. Concluding from this it becomes visible that the current state of the practice of autonomous driving corresponds most likely to automation types up to level 3 as outlined in Table 2.1. Nonetheless, the current restrictions may not only have technical, but also legal reasons, since the respective national or even regional legal regulations are quite different. In parallel, however, it becomes more and more evident that it is a challenging task to guarantee safety for full automated driving behavior on level 4 and 5 [23]. This requires a comprehensive proof that all dynamic driving tasks can be reliably performed by the autonomous vehicle itself. In order to simplify these tests, in a first step fully automatic driving can be limited to certain driving situations on level 4. Overall, however, it becomes apparent that communication concepts for coordinated autonomous driving via the addressed collaboration networks are still difficult to implement at present.

The question of safety assurance and certification, on the other hand, becomes increasingly important in advanced projects which should be released to the market. Moreover, the question of standardization for networked collaboration concepts becomes a key topic when several different projects, which should be compatible, are about to enter the market. In recent years, nonetheless, some promising ideas and concepts have emerged in the safety community to enable modifications in system functions during operation. An inspiring research direction, for example, is on certifying safety at runtime. Such approaches aim at shifting parts of the safety assurance mechanisms to runtime. In contrast to conventional development time methods, these approaches also address the operating time of a system. Thus, safety can be guaranteed not only for "plug and play" applications to incorporate new components in safety-critical domains, but also for flexible networking concepts as in the target application scenarios. By integrating previously separate system features, new overall functionalities could be created in this way. This shows that suitable preliminary work is already available in the safety community.

As outlined in the introductory Chapter, the focus of this thesis is on specifying a broadly applicable networking approach for an enhanced situation awareness. It should address as many collaboration partners and scenarios as possible to optimize the individual driving behavior in the various dynamic driving situations. An important objective is the reduction of the high engineering effort resulting from individual and highly customized solutions for networked applications taking into account the multitude of vehicle platforms and characteristics. Standardization of collaboration capabilities and the reusability of already specified aspects is a key prerequisite for this. For this reason, it was concluded that it is not appropriate in this context to focus on fully preconfigured interaction patterns for specific combinations of platform partners at development time.

However, the handling of situations in networked driving modes that are classified as unsafe due to safety considerations remains a crucial issue. In the dynamic and versatile application context of networked driving processes, a simple deactivation of the collaborative functions can be insufficient if the previous certification basis is suddenly missing. The reasons for a missing certification basis can be manifold, e.g. communication problems, inadequate driving styles of surrounding vehicles or suddenly changing environmental conditions. It becomes apparent that if no close system coupling can be specified at development time, the safety assessment during operation becomes decisive, taking into account the external system conditions.

Generally, as it was outlined in the introductory Chapter, a modified driving behavior to enable networked driving processes also requires an adjusted safety concept. Hence, for the self-adaptive vehicle platforms, more fine-grained runtime checks seem to be indispensable to ensure safety during operation of a networked driving mode, as the vehicle constantly interacts with its dynamically changing environment. The consideration of individual external safety-relevant services in a more formalized way as runtime evidences could be the key to achieving this. In combination with a runtime safety assessment this could enable a precise determination of permitted system functions such as the allowed safe driving maneuvers. In order to specify such an evaluation model, an extensive analysis of related work has been conducted. The most important contributions to this are summarized in the following.

Section 2.2.1 first outlines condition monitoring strategies using additional external data for analytical fault detection in networked driving modes. If a monitored safety property is suddenly degraded, this can be utilized as a trigger for a predefined adaptation of a current driving behavior. Furthermore, the illustrative idea of health-signals is outlined, which allows the individual health states of a multitude of subcomponents to be linked and transferred to the behavior generation layer. Section 2.2.2 presents a classification of possible behavioral adaptation concepts with respect to identified anomalies and faults in a networked driving mode. In addition, this Section presents an ideal behavior generation architecture for integrating the intended dynamic adaptation processes into the existing non-networked overall system behavior. Utilizing such an architecture could support certification activities of the overall system safety with a systematic testing approach.

Then in Section 2.3 self-adaptive systems as addressed by the adaptive systems community are outlined. Here, the models@runtime research group provides valuable insights in this regard. Especially the engineering approaches using pre-configured variability are a useful contribution to understand how reliable dynamic system reconfigurations could work during operation. However, unlike the safety community in Section 2.4, safety in the adaptive systems community is considered only as one assurance aspect among many others.

Afterwards in Section 2.4 openness and self-adaptivity for external services as addressed by the safety community is outlined. Current standards for certification as well as the state-of-the-art approaches of the safety community are discussed, taking into account particularly important research areas such as modular certification in Section 2.4.1 and runtime certification in Section 2.4.2. Modularization allows to differentiate between various safety quality attributes and could thus be particularly helpful to specify the intended entire ecosystem of networked system capabilities. The addressed runtime certification approaches could be the key to link previously separate safety-related runtime services of distributed systems. Thus, a holistic runtime safety assessment for networked safety-critical applications, such as networked driving modes in the vehicle domain, could be enabled. Then in Section 2.4.3 available safety modeling techniques using fault tree architectures for deductive safety analyses and lightweight BDD evaluations are described. In particular, modularized approaches such as Component Fault Trees (CFTs) are outlined in this context. They seem to be promising for the identification of an structured integration approach of in- and external services that have specific common characteristics affecting the behavioral layer. It turns out, as concluded in Section 2.4.4, that a reliable runtime integration mechanism for external services, such as a contract-based merging logic, in combination with an overall lightweight runtime safety assessment using preconfigured variabilitiy seems to be most promising. In this way reliable and fast dynamic reconfiguration mechanisms of system behavior according to available in- and external available services could be established. However, such an approach is not yet available. A suitable solution approach is therefore proposed in Chapters 3 and 4.

2.2.1 Condition monitoring and signal-based fault-detection

Condition monitoring (CM) is particularly widespread in industry to increase safety and efficiency of machines. In this way, faults or anomalies can be detected and corrected at an early stage and the overall system availability can be increased. As described in the introductory Chapter, also for dynamic driving tasks of autonomous vehicles, a timely fault detection as well as a localization of error sources is essential. In this way, suitable reconfigurations of driving behavior can be triggered at an early stage in response to emerging hazards, thus preventing their actual occurrence. An illustrative example in

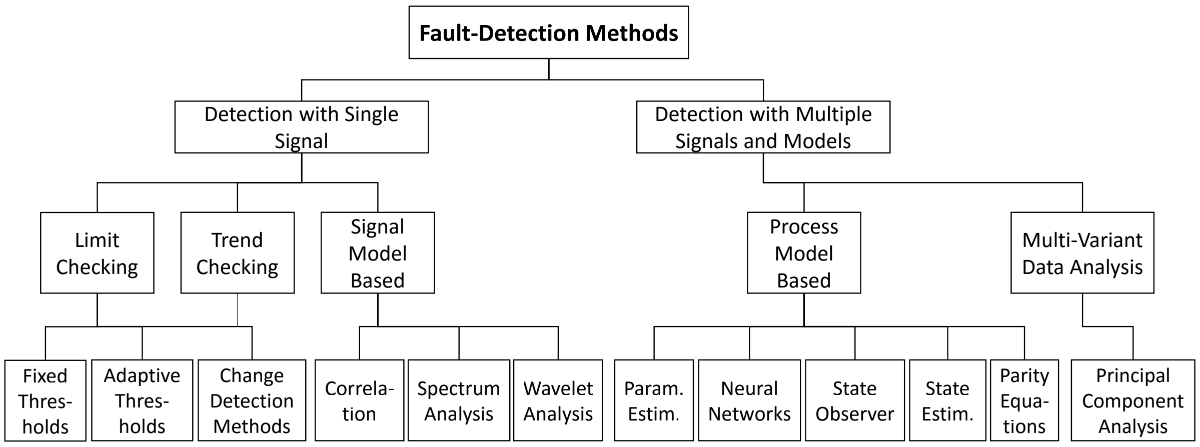


Figure 2.3: Classification of Signal-based Analytical Fault-detection Methods (adapted from [24])

the automotive sector are tire pressure control systems. In the event of a pressure drop, the driver is warned and can safely stop the vehicle without suffering a dangerous loss of steering ability while driving.

In the addressed networked driving modes of autonomous vehicles, the safety-relevant data are distributed among the various involved vehicles and infrastructure devices. As already outlined, in the dynamic driving context not only hardware or software failure of involved systems pose threats to the addressed collaboration modes. Also external factors, such as the unsuitable driving behavior of other vehicles or suddenly changing dynamic driving situations are difficult to predict and are particularly hazardous. Nevertheless, when trying to assess safety of the current driving condition with an enhanced situation awareness, the individual driving characteristics are of special importance. The still safe driving status of a particular autonomous vehicle depends mainly on individual driving characteristics such as vehicle weight or deceleration capabilities in combination with the current driving parameters. In the absence of detailed platform knowledge or the current driving parameters, it can thus be difficult to determine a safe driving status from an external perspective. Moreover, it would contradict the aspired solution approach of full autonomy for self-optimizing systems, which also includes situation analysis and determination of system goals for maximum flexibility as described in Section 2.1.3. Hence, when trying to assess safety at runtime by combining distributed safety knowledge, it makes sense to carry out the assessment for each vehicle individually.

For the respective networked autonomous system, it must therefore be determined in which way external safety-relevant data can be reliably taken into account in order to guarantee a safe networked driving mode in all conceivable driving situations. Efficient methods are needed to continuously monitor interlinked system safety properties resulting from safety critical dependencies to other systems and environment. Beyond that, a sudden deterioration of networked system conditions should reliably trigger an adequate behavior adaptation as a countermeasure. Thus the occurrence of hazards while driving in a networked mode should be prevented or at least mitigated as outlined before.

When using shared services for condition monitoring of networked system applications as well as triggering behavior adaptations, it has to be clarified which methods of safety-related runtime checks are most suitable. Isermannn [24] introduced a classification for fault-detection methods with respect to signal-based analysis techniques as shown in Fig. 2.3. In this thesis, a signal corresponds to a service used for sharing specific information among networked systems. Basically, the illustrated fault-detection methods can be subdivided according to their analysis techniques into methods that use single signals as well as several signals and modes.

Referring to the classification in Fig. 2.3, the focus of this thesis is on straightforward checks of limits, trends as well as correlations based on single forwarded signals. This limitation was made for two reasons. On the one hand, from a safety perspective the safety-related runtime analysis should be as lightweight and predictable as possible. This will be further clarified in Section 2.4. On the other hand, the shared safety-related data should be specified in such a way that they can be forwarded and interpreted in a standardized way as runtime evidences. This basically requires a harmonized basic abstraction level of the raw sensor measurement data and their various sensor fusion concepts. Hence, it does not seem sensible to take into account evaluation methods such as spectrum analysis and wavelet analysis. They represent low level analyses of raw measurement data. As a result, the analysis methods with fixed or adaptive thresholds, change detection methods or correlations seem to be most promising. They will be examined later in more detail for the derived solution approach.

2.2.2 Merging safety quality attributes and dynamic adaptation of system behavior

Highly automated systems such as driverless vehicles should be capable of adapting themselves to new dynamic driving tasks without human assistance in a safety-critical context. It is intended that ideally, instead of actively influencing driving behavior, people can turn to other tasks and use their time more effectively. Basically, depending on their passive or active role in adaptation processes, systems can be subdivided into adaptable and adaptive systems. In adaptable systems, users have to manually adapt the relevant system features in order to achieve the desired objectives identified by the user. This can be for example the manual setting of personal preferences in information systems as outlined by Oppermann [25]. In contrast to that, adaptive or (self-)adaptive systems change the passive role of an adaptable system to an active one. These systems can actively readjust themselves within a given range of dynamic variability in accordance to the respective active system goals.

Autonomous vehicles with the outlined self-optimization capabilities, as presented in Section 2.1.3, rely on the “Analysis of the Current Situation” as the initial action to determine system goals and to adapt their driving behavior in a reliable way. When anomalies or faults are detected in a networked driving mode, the effect on overall system safety should ideally be clarified in advance in order to derive an optimal driving behavior.

Kleinlützum et al. [26] proposed an illustrative approach for propagating and merging internal subcomponent conditions in a formalized manner in a hierarchical system architecture. Applying so-called *Health Signals* in modularized robot control frameworks should enable an anomaly detection in the implemented subcomponents and derive an adequate behavior adaptation in a predefined way. A *Health Signal* is generated based on a predefined semantic and represents the health condition of a subcomponent. Thereby the signal is propagated and modified based on a normalized scalar value $h \in [0,1]$. At the highest propagation level of *Health Signals*, the integrated component conditions should influence the generation of system behavior. This approach shows that the assignment of individual component conditions to the behavioral level can be carried out appropriately using continuous runtime checks. Such an ongoing propagation and merging mechanism for safety quality attributes also seems to be reasonable for the addressed solution approach. However, to assess safety in a networked driving mode various different safety-related runtime services from external sources have to be considered. This requires a more detailed semantic for specifying networked system conditions as well as a reliable integration approach for external services.

Moreover, for the optimization of driving behavior in networked driving modes, adequate dynamic adaptation mechanisms are required. They should be capable of adapting the existing overall driving behavior to available external services in a reliable way at runtime. In principle, adaptations can be triggered by the system itself or by an external source as outlined in Fig. 2.4. As explained before, this thesis focuses on the individual self-adaptive vehicle platform. In the intended use case the external data is always considered as additional input data to derive the individual driving behavior autonomously. Consequently, in order to take the available external services into account, it is necessary to define a sufficient dynamic variability for the driving behavior of networked vehicles.

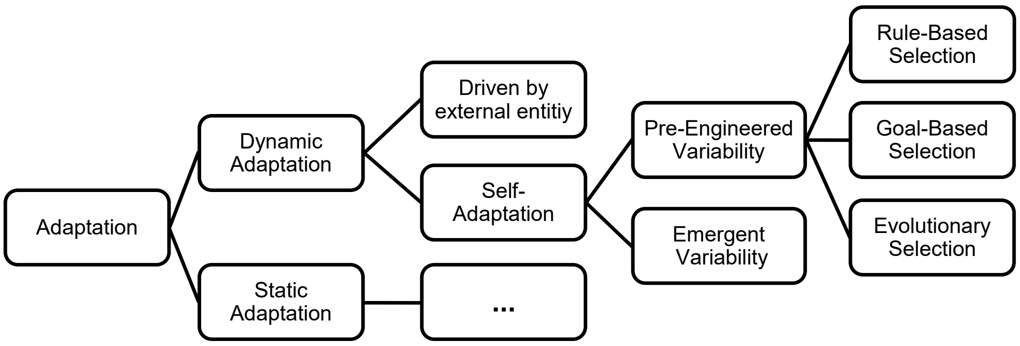


Figure 2.4: Classification of Adaptation Concepts (adapted from [27])

Within the specified dynamic variability, self-adaptation mechanisms can be implemented with respect to the classification in Fig. 2.4 that are either based on pre-engineered variability or on emergent variability. Considering the safety-critical application context, this thesis focuses on pre-engineered variability to restrict the necessary runtime safety responsibility. While a goal-based selection is most promising for establishing a networked driving mode, a more rigid evaluation mechanism such as a rule-based selection seems sensible for the ongoing collaboration. The rule-based adaptation could be suitable to react to emerging hazards in the dynamic driving context in a predefined and thus predictable manner.

The correct overall driving behavior according to the relevant dynamic driving tasks must then be validated with reference to the newly added adaptation mechanisms before practical use. In complex mechatronic systems such as autonomous vehicles, a holistic understanding of the internal structures of data processing is required in order to reliably modify the existing system behavior. Reports on necessary test activities of individual autonomous vehicles show that these have to be very extensive and are therefore time-consuming [28]. If no suitable strategy for self-optimization of autonomous systems with external knowledge is established, extensive testing activities might be necessary even for minor changes, since the entire system behavior is possibly changed. To avoid this, it can be tried to limit the modifications of the overall system behavior to certain subfunctions and in this way also to restrict the test activities to the affected parts of the system behavior. For this reason, control architectures for the derivation of system behavior ideally have traceable and clearly defined data processing structures in order to be able to adapt the relevant parts. This also requires the avoidance of feedback loops within the modified behavior generation structure to ensure a predictable system behavior and to avoid unpredictable side effects from modified elements.

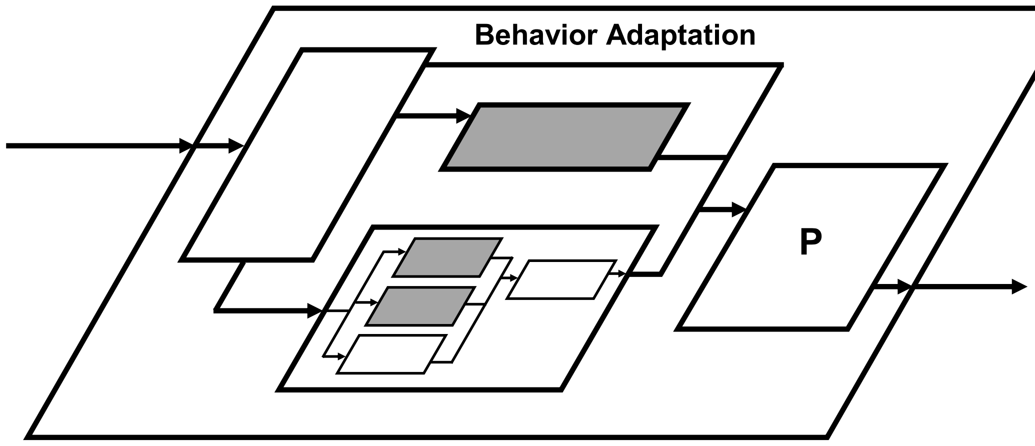


Figure 2.5: Structure, Behavior & Parameter Adaptation of a Self-optimizing System (adapted from [19])

Frank et al. [19] illustrated a control architecture design which is appropriate for self-optimizing systems as outlined in Fig. 2.5. A modular and hierarchically linked control architecture as shown in Fig. 2.5 is beneficial to implement the intended behavioral adjustments in a predictable and targeted manner. It shows a clear cause-effect structure that allows the reconfiguration of modularized parts of the specified behavior generation layer. When possible effects of modifications are well understood, the testing activities for certification can be limited to the affected parts of dynamic driving tasks. For example, the illustrated gray modules can be added, replaced or partially modified within their well-defined substructures to enable selective behavioral modifications or even major structural changes.

In principle, since a modified module in Fig. 2.5 influences the subsequent modules, it seems sensible to limit behavioral influencing mechanisms to the highest possible level of behavior generation (such as the P-module in Fig. 2.5). However, such a modification approach can be error-prone. When modifying parameter values or certain behavioral aspects, it is important to ensure that they are compatible with the other currently valid parameters and behaviors. For example, it is not possible to increase the speed of a vehicle while at the same time maintaining a safe distance to the vehicle in front. It turns out that the additional influences of modified submodules on other modules of the control architecture are indispensable for a consistent overall system behavior. For this reason, it is more effective to first analyze the existing behavior generation architecture and then to specifically adapt relevant parts of behavior generation on the respective integration level. In this way, it can be ensured that the various controlled driving parameters form a valid overall system behavior. A suitable integration approach for external services should be able to reflect this hierarchical structure of behavior generation.

2.3 Safety Assurance and Certification Approaches as Addressed by the Adaptive Systems Community

The partners of the adaptive systems community basically aim at integrating more variability for automated decision-making into complex software systems. For the user, a multitude of system characteristics can be improved, such as maintainability or usability in general. Since manual settings should be replaced or at least supported by automated adjustments, suitable assurance methods are required in the various steps of system adaptation. Safety assurance in this context is considered only as one aspect among many others. Blair et. al. [29] set an important starting point in the adaptive systems community, when they introduced the concept of models@runtime (M@RT) for complex software models. In particular, they pointed out how automated reasoning can support human decision taking and substitute it in the long term. Cheng et. al. [30] developed for M@RT the MAPE-MART loop, which is adapted from the MAPE-K loop. It provides a reference architecture for self-adaptation with a focus on individual assurance means for all necessary adaptation steps. Fig. 2.2 in Section 2.1.3 shows a comparable reference loop, but with an emphasis on the overall system's self-adaptation process in the context of safe environmental interactions. Instead, M@RT addresses the more abstract management of system elements and characteristics in complex software models. Generally, managing complex software models requires extensive autonomy for decision making which is hard to balance with strict safety requirements. For this reason, only assurance approaches for adaptation processes are outlined below that have a special focus on safety assurance. These approaches typically utilize validation and verification as a means for safety assurance. These techniques are categorized according to their application at development time or runtime and are analyzed for their suitability in the target application domain.

2.3.1 Approaches using validation and verification as a means for assurances

A fundamental concept for efficient validation and verification (V&V) means is "Design by Contract" as introduced by Meyer [31]. It enables the unambiguous assignment of system dependencies by means of a well-defined demand/guarantee interface description. Due to its lightweight evaluation mechanisms, it seems to be appropriate not only for development time V&V, but also for runtime V&V. However, since it is not a full-fledged assurance approach on its own, it is analyzed in more detail with respect to runtime certification in Section 2.4.2. The approach presented in this thesis is mainly based on such a kind of contract-based modeling.

Approaches which utilize development time V&V means for assurances are especially useful for component-based modeling to support safety engineers in the design phase. Giese et. al [32] outlined a component-based hazard analysis approach for checking all possible configurations in the design phase and determining the most optimal one. For the addressed networking approach, however, it does not seem sensible to analyze hazards considering all conceivable system configurations only using development time knowledge. There is no predefined overall interaction process specification available that could enable a pure development time analysis. Instead, the specified individual networked driving modes are always conditional, with reference to available runtime data. If runtime data cannot be taken into account in such scenarios, a development time hazard analysis would probably be too pessimistic regarding the huge runtime variability of networked driving modes. In consequence, the practical application of the intended solution approach would be severely restricted. Thus, safety for suitable dynamic driving behavior can only be guaranteed by shifting parts of the safety evaluation mechanisms to runtime where safety-relevant runtime data can be shared and interpreted.

There are also approaches which utilize runtime V&V as a means for assurances. Priesterjahn introduced an approach for ensuring safety at runtime [33]. For each potential reconfiguration it is evaluated at runtime whether it adds additional hazards to the overall system safety. Then the optimal set-up is automatically configured during operation. A drawback is however that the runtime safety evaluation depends on development time safety knowledge. Comparable to the development time V&V, the considered hazards from a development time perspective for dynamic driving tasks are probably too pessimistic. Furthermore, since only a conditional interaction process specification is available at development time, no suitable uniform hazard metrics can be introduced. Hence, such a type of automated evaluations cannot be used to check whether additional hazards to system safety arise in a certain networked driving mode.

2.3.2 Engineering approaches for dynamic reconfiguration

Next to assurance approaches that utilize V&V means for a formal safety assessment to derive in the next step an optimal configuration either at development time or runtime, there are engineering approaches which directly link current system states to an optimal configuration based on preconfigured variability. Here the "Analysis of Current Situation" and the "Determination of System Goals", as described in Section 2.1.3, for all conceivable operating states are already carried out at development time and are associated to a predefined set of evaluation results. In this way, the predictability of the evaluation results can be increased and the runtime variability can be limited. The utilization of such a degradation model can also take safety issues into account.

Rawashdeh et. al. [34] introduced a framework for specifying dynamically reconfigurable embedded systems. They described how graceful degradation concepts could be implemented in a reliable way to embedded systems using dependency graph modeling and a runtime evaluation mechanism. Also Trapp [35] specified a holistic modeling framework for pre-engineered variability applying a rule-based selection to enable reliable dynamic reconfiguration mechanisms. Adler et. al. [36] introduced a component-based modeling and verification technique for dynamic adaptation in safety-critical embedded systems.

Basically, these approaches try to ensure safety by establishing preconfigured variability for a particular system and verify its control algorithm at development time to ensure a safe operation at runtime. Preconfigured variability to ensure reliable dynamic reconfigurations is also applied in the solution approach of this thesis. Nevertheless, the aforementioned engineering approaches are focused on single system control architectures using internal services for dynamic reconfigurations. Instead, for the intended application context of networked system applications, the focus is on reliable integration mechanisms for external services as well as a runtime safety evaluation of networked system properties. To establish a widely applicable networking concept, this requires a more abstract and formal description of shared safety-related services as runtime evidences. In addition, openness for external services from previously unknown systems must be enabled in order to reliably optimize system behavior for all networked dynamic driving tasks.

2.4 Openness and Self-Adaptivity for External Services as Addressed by the Safety Community

With an ever-increasing proportion of software, modern technical systems can adapt flexibly to new tasks, such as in the application context of networked autonomous driving. The safety community naturally has a very conservative attitude towards enabling dynamic variability for system reconfigurations in safety-critical applications. For today's state-of-the-practice safety certification, all conceivable hazards and risks arising from the operation of a technical system should already be taken into account in the development phase with a comprehensive overall safety concept. On the basis of a conclusive safety argumentation, it can thus be demonstrated before commissioning that all possible application risks are sufficiently low.

Since the development of a holistic safety concept is a complex and sometimes also a creative task even for safety engineers, automated decision-making is in most applications strictly prohibited or at least largely restricted. Applications that utilize dynamic reconfiguration mechanisms in a safety-critical context are typically based on preconfigured variability to limit the amount of runtime responsibility for safety assurance. This can be, for example, the selection of an optimal software configuration from several predefined alternatives according to simple and robust runtime criteria. In this way, the number of possible runtime variants of safety assurance means can be restricted to an acceptable level and in parallel a minimum of runtime adaptability can be provided.

Nevertheless, when pre-configured variability is considered for today‘s safety assurance and certification approaches, it is primarily focused on the reconfigurability of component functions at integration time. For the addressed application context of networked autonomous systems, however, a single adaptation step at integration time would not be sufficient to adequately cover the manifold hazards in the application scenario without severely restricting the networked system functions. As already outlined in the introductory Chapter, there is an increasing demand for technical system, that can adapt flexibly in highly dynamic and variable application environments. It could thus become possible to continuously optimize the existing individual functional scope in the respective application context during operation.

Since current safety regulations do not sufficiently consider such promising approaches utilizing continuous self-adaptation, it seems sensible to develop an approach that closes this gap for future applications. This basically requires suitable safety assurance and certification techniques where necessary parts of the assurance means can be shifted to runtime in order to reliably assess the dynamic application context. To derive a suitable solution approach, the current state-of-the-art and state-of-the-practice of standards and certification techniques are outlined in more detail in the following.

2.4.1 Standards for safety certification and state-of-the-art approaches

Today, the IEC 61508 [37] is a widely accepted standard for specification, design and operation of Safety Instrumented Systems (SIS) [38]. It is a very generic standard that addresses different industrial sectors. However, these universal standards are often too imprecise for concrete application in the various domains such as aviation, railway systems or automotive systems. Consequently, domain-specific guidelines for implementation have been developed. In the aviation domain, the DO-178B standard [39] provides guidance on certification issues. For railway systems there are still many national regulations as in Germany issued by the Eisenbahn-Bundesamt. In the automotive domain, the ISO 26262 is the most important standard for functional safety [40].

If the planned promising networked applications should be enabled in future automotive systems, dynamic adaptation capabilities of driving modes become crucial. Today, for safety and economic reasons, technical components and functions of automotive systems are usually configured and certified once at design time. The ISO 26262 specifies that the safety-related system functions are not allowed to be configurable during operation. This is also caused by the enormous hazard potential of hacker attacks [41] [42], as has been prominently demonstrated in the past. Nevertheless, along with the development of communication technology, the demand for powerful connected services for autonomous vehicles is constantly growing as described in the introductory Chapter. The ISO 2626262 already addresses the so-called “Safety Element out of Context” (SEooC). The SEooC can be developed independently of the later application platform. This is done based on assumptions regarding the possible constraints in its operating environment. Furthermore, the AUTOSAR adaptive community develops a so-called heartbeat signal, which could be facilitated for automated reconfigurations [43].

Modular certification

Applying modular certification is especially helpful for system components, which should be subsequently integrated to an overall system. This can be carried out either manually at development time or in an automated integration phase at runtime. Rushby provided some basic ideas about modular certification [44]. In the aviation domain, the DO-279 [45] standard provides certification guidance for “Integrated Modular Avionics” (IMA). The FAA Document AC 20-148 [46] defines guidelines for specifying reusable software components. With DECOS (Dependable Embedded Components and Systems) [47] generic safety cases were introduced for incremental certification processes. For modeling complex safety dependencies in a modular way, the Goal Structuring Notation (GSN) by Kelly [48] [49] can be applied. It enables a compositional safety case construction.

Considering the envisaged application context of these modular certification approaches, it becomes apparent that they focus on the certification of safety at development time or at integration time to enable the establishment of integrated system functions. In the addressed open and highly dynamic application context, however, it is not possible to ensure safety completely at integration time. Instead parts of the safety assurance mechanisms have to be shifted to runtime to enable continuous runtime checks.

2.4.2 Runtime certification

Safety Kernels

Rushby [50] introduced in his early work the idea of “safety kernels”, which should continuously ensure safety for the operation of complex software systems. He pointed out that safety kernels should be able to enforce safety on the entire system without needing any other parts of the system to cooperate. To establish a kernelized system structure in an appropriate and efficient way, they should be limited to crucial system functions. As a result, however, safety kernels cannot evaluate the overall system behavior and thus enforce “good behavior”. Instead, according to Rushby, they are better suited for checking which properties the kernel does not provide and thus for guaranteeing negative properties. In this way, safety kernels can be facilitated for the prevention of “bad behavior”. This was an important starting point for the later derivation of runtime certification approaches in the safety community. It laid the foundation for enabling reliable dynamic reconfiguration mechanisms in complex software systems. Also in the target application context of networked autonomous driving, such an approach seems promising. Rushby also gives some additional insights on how such a kernel architecture should be structured. The implementation of such a structure is also considered for the presented solution approach.

Regarding runtime certification, Rushby made some interesting considerations regarding the possible benefits, as published in [51] [52]. Analyzing systems at runtime he came to the conclusion that certification of system parts would help to assess safety properties in an uniform way. He especially addressed the key parts of a software system and suggested to certify the system kernel and the running programs in a separate way. However, Rushby stayed on a very abstract level with his ideas and provided no concrete solutions at first. In his later publications, he then analyzed newly developed approaches in the safety community that dealt with his concepts [53].

Design by Contract

When implementing suitable evaluation mechanisms, the design principles introduced by Meyer [31] can be helpful. Meyer particularly focused on improving the reliability of software systems. He especially pointed out that today’s software engineering is focused on object-oriented programming to enable reuse. If such reusable components are used in a variety of applications, incorrect system behavior is particularly critical because the consequences are difficult to assess compared to application-specific developments. Consequently, he concluded that reliability is a central element of any meaningful definition of software quality. In order to enforce reliability in object-oriented approaches, he proposes methodological principles for the development of correct and robust software under the term “Design by Contract”. This is done based on the Eiffel analysis, design and programming language [54]. By applying such a systematic development approach, simple and powerful mechanisms for handling exceptions can be established. For conditional safety certification, as described below, this provides an important basis for dealing with abnormal cases as in the envisaged application scenario of networked vehicles.

Conditional Safety Certification

In the following, the concept of conditional safety certification is outlined, which enables to shift parts of the certification activities to runtime. Similar to conventional certification procedures, the specification of an entire safety concept for an examined system is still conducted in the development phase. A major difference is, however, that several variations of subsystem configurations are already part of the specification. When the respective system components are connected to an overall system at integration time, it can then be checked, either manually or in an automated way, which system configuration is currently active and the corresponding functional scope can be determined and guaranteed on the basis of predefined conditional safety specifications. Since integrated functionalities are not necessarily stable over time, certain conditions, such as the ongoing communication between subsystems, have to be continuously monitored during operation.

Conditional safety certification is particularly valuable in the context of the Open Adaptive Systems (OAS). If the specified safety certification variants are appropriately structured, a negotiation process can be utilized to automatically check whether there is an integrated overall functionality available to interconnect currently separate systems. In this way, by guaranteeing safety-certified, higher-level overall functionality at integration time, openness for previously unknown systems can be achieved.

Recently Schneider introduced a promising solution approach called “Conditional Safety Certificates” (ConSerts) [56] to achieve conditional safety certification in the context of OAS. In Fig. 2.6 a ConSert specification for a round baler is shown, which can be attached to a tractor [55]. The round baler is communicating to the tractor in a predefined way and is influencing the tractor control. Each ConSert provides a modularized demand-guarantee description and considers several slightly different variants of system configurations and conditions. This is done using so-called Conditional Safety Trees (CSTs) as illustrated in Fig. 2.6. Each CST is part of a conditional safety concept that has already been specified at development time. It represents a valid integrated functionality with respect to necessary features of involved systems. A set of CSTs can be utilized to establish several degraded modes of integrated system functions. This makes

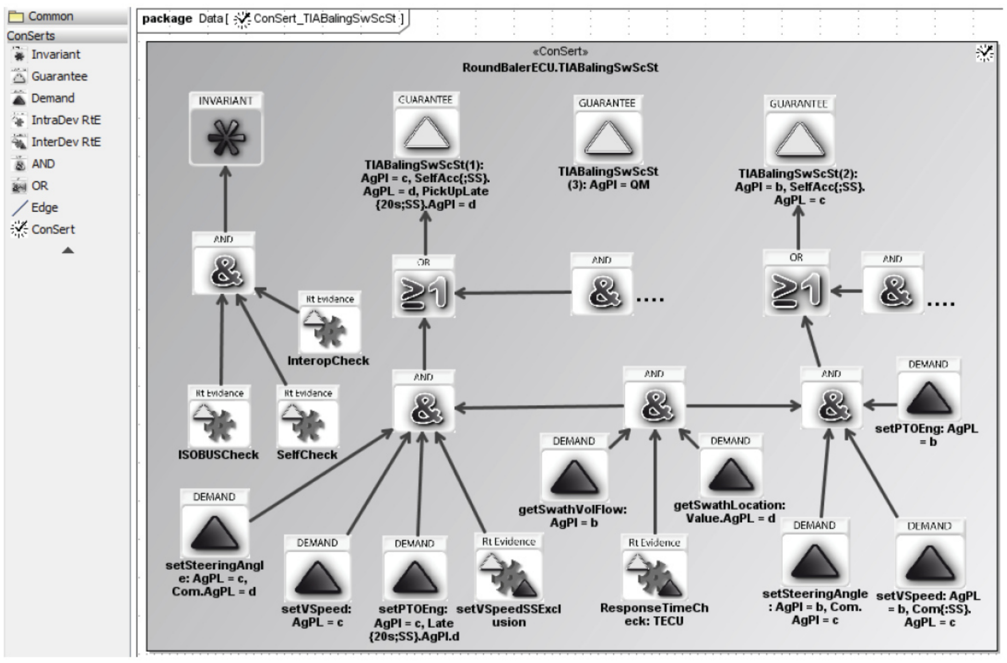

Common
ConSerts
Invariant
Guarantee
Demand
IntraDev RtE
InterDev RtE
AND
OR
Edge
ConSert
package Data [ConSert_TIABalingSwScSt]
«ConSert»
RoundBalerECU.TIABalingSwScSt
INVARIANT
GUARANTEE
TIABalingSwScSt(1): AgPl = c, SelfAcc(:SS). AgPL = d, PickUpLate (20s;SS).AgPl = d
GUARANTEE
TIABalingSwScSt (3): AgPl = QM
GUARANTEE
TIABalingSwScSt(2): AgPl = b, SelfAcc(:SS). AgPL = c
AND
OR
AND
OR
AND
Rt Evidence
InteropCheck
Rt Evidence
ISOBUSCheck
Rt Evidence
SelfCheck
AND
AND
AND
DEMAND
setPTOEng: AgPL = b
DEMAND
getSwathVolFlow: AgPl = b
DEMAND
getSwathLocation: Value.AgPL = d
DEMAND
setSteeringAngle: AgPL = c, Com.AgPL = d
DEMAND
setVSpeed: AgPL = c
DEMAND
setPTOEng: AgPl = c, Late (20s;SS).AgPl.d
Rt Evidence
setVSpeedSSExclusion
Rt Evidence
ResponseTimeCheck: TECU
DEMAND
setSteeringAngle : AgPl = b, Com. AgPl = c
DEMAND
setVSpeed: AgPL = b, Com(:SS). AgPL = c

Figure 2.6: ConSert Specification [55]

it possible not to lose certification for integrated functions in the event of minor runtime changes, but only to restrict system functionality.

At integration time, a negotiation process is utilized to examine whether certain input requirements of ConSerts can be met by existing basic guarantees of systems in order to grant a higher level guarantee. In this way openess for previously unknown systems can be enabled. In doing so, the certification process is not limited to one integration level, but can be carried out across an arbitrary number of hierarchical integration levels.

For safety engineering in general, the integration time can also be understood as the initial phase of the ongoing operation, the runtime. Nevertheless, to clarify the ConSerts operating principles, it seems sensible in the following to distinguish the actual operating time, which is considered as runtime, from the time, where functions or components are combined, which is considered as integration time.

A ConSert specification not only contains demands that are constant at runtime, such as necessary hardware or software capabilities of involved systems, but also requirements that can be variable at runtime. This can be the current power supply, communication aspects or the availability of certain system resources at runtime. Consequently, in addition to the static certificate descriptions, which only have to be addressed at integration time and then would no longer change, there are variable networked system characteristics that have to be continuously monitored at runtime. This is done using so-called Runtime Evidences (RtEs). Only the consideration of both features can guarantee safety on the basis of conditional safety specifications.

Nevertheless, the ConSerts approach primarily aims at adaptation processes that are triggered when new system functions or components are added at integration time. This is done by providing adequate negotiation and runtime certification mechanisms. Although runtime aspects such as the RtEs mentioned above are utilized, this is basically done to ensure that the integration time conditions are still valid. For minor changes during operation of a CST functionality, several alternative or degraded modes are specified in relation to a selected ConSert. They can be activated if required. If, however, a crucial RtE for an active integrated system function is missing during operation, the certification is withdrawn and the integrated functionality is deactivated. Then a new integration time starts where a new ConSerts negotiation process is initiated to establish an alternative integration of system functions.

The ConSerts approach thus consists of a static development time safety concept as well as a dynamic runtime safety concept. If the RtEs do not change starting from the integration time, safety for an integrated functionality can be fully ensured by relying on a development time safety concept. If, however, a RtE is suddenly missing after establishing an integrated functionality at integration time, also a kind of runtime hazard prevention takes place based on the resolution of the common functionality or the switching to an alternative, slightly different ConSert mode. Nevertheless, each CST can only guarantee safety for a complete ConSert mode, which makes it difficult to specifically adapt individual parameters with runtime mechanisms.

Overall, it becomes visible that the options for runtime reconfigurability of ConSerts, considered as actual operating time, is very limited. From a safety point of view, this is advantageous because the runtime responsibility for safety should generally be as small as possible. In the ConSerts application context of flexible "plug and play" approaches for attaching new components, most of the safety assurance aspects can already be clarified at development time. Hence, no further runtime safety mechanisms are required in the target application domain of ConSerts.

The addressed target application context in this thesis, however, is much more dynamic and variable, since flexible self-optimization processes with regard to available external runtime data and further environmental issues must be made possible. This requires a different certification approach based on more runtime variability in an established networked mode. In an established networked driving mode, services from different system sources as well as environmental aspects have to be continuously monitored in order to assess if the system is still in a safety certified operation mode. A safety assurance concept can no longer be defined almost completely at development time according to static safety dependencies of involved systems, but only conditional according to variable safety dependencies at runtime.

Hence, it is no longer sufficient at integration time to select and execute a predefined safety concept for safety assurance. Instead, a more analytical runtime safety assessment based on degraded system capabilities and the networked driving situation is required. In this way, a more targeted and fine-granular runtime adaptation of certain system parameters can be enabled. Next to slightly degradations of networked driving modes, this can also be utilized for preventing hazards by triggering adequate countermeasures in the dynamic context. Ultimately, this requires ongoing runtime adaptations for safety assurance instead of focusing on a single adaptation step at integration time which already takes almost all situational requirements into account.

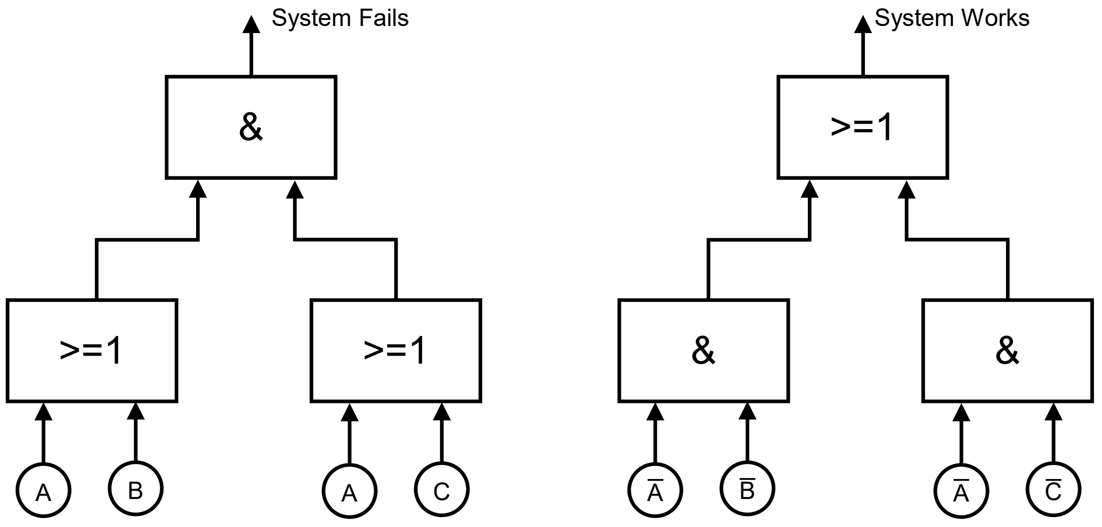
Generally, for the conditional safety certification of networked driving modes with respect to the ConSerts approach, it seems promising to introduce a more analytical safety assessment based on RtEs. They can be derived from internal and shared services. Since the consideration of external services should be enabled, independent of their transmitter platforms, it seems beneficial to consider each forwarded service with a separate RtE check. The focus for conditional safety certification in the target application context is therefore on specifying a holistic runtime evaluation model to check whether the current operational situation is still safety certified. Moreover, the evaluation model should provide a sufficient runtime variability for possible reconfigurations of system behavior in order to be able to deal with emerging hazards. In this way, by monitoring the absence of anomalies and the provision of an adequate degradation concept based on targeted countermeasures for detected anomalies, it should become possible to establish a conclusive safety argumentation for conditional safety certification.

2.4.3 Modeling runtime safety monitors

In safety critical applications, such as networked autonomous driving, certain types of failures for networked applications cannot be tolerated, since they can lead to serious accidents. Rushby [50] pointed out that failures are the result of faults and the prevention of failure therefore depends on eliminating faults, or on tolerating them safely. When specifying networked driving modes, various causes of faults and their influence on driving safety must be taken into account in order to derive adequate countermeasures. As already outlined, certain faults in networked driving modes, such as inadequate driving behavior of surrounding vehicles or communication problems, cannot be eliminated, but tolerated in a safe way.

A continuous monitoring of networked safety properties should make it possible to automatically trigger a suitable reaction behavior in the dynamic driving context if required. In order to identify a suitable reaction behavior while driving, an evaluation model for automated runtime checks is needed that specifies the impact of degraded networked system capabilities to the overall system safety. Ideally, comparable to the outlined contract-based merging logic in Section 2.4.2, the relevant safety properties in a networked driving mode can be aggregated in a traceable way to the overall safety of a networked driving mode. To achieve this, complex system safety dependencies in a networked driving mode must first be decomposed into manageable parts according to their hierarchy of failure influences.

Basically, the decomposition of complex safety dependencies into manageable parts can be achieved with classic Fault Tree Analysis (FTA) and its subsequent advancements. FTA was originally introduced in the 1960s [57] and offers a breakdown hierarchy according to the hierarchy of failure influences. FTA is typically applied at development time to assess the overall safety of a system and to identify weaknesses in the system architecture. For the envisaged runtime safety assessment, a composition model for RtEs has to be specified. It should also make the identification of weaknesses possible, but during operation with the goal of automatically deriving a suitable reaction behavior. The use of a modeling approach such as fault trees to propagate degraded safety parameters has several advantages. The way of integrating safety properties is still traceable and understandable, which enables an efficient modeling. In parallel, Fault Trees can be determined at development time and evaluated in an automated and lightweight way, for example using BDDs.



(a) System Fault Tree (b) System Success Tree

Figure 2.7: Comparison of a System Fault Tree and a Success Tree

Fig. 2.7 (a) shows a typical Fault Tree architecture for reliability analysis of technical systems. Here the occurrence of the Top Level Event (TLE) *System Fails* can be evaluated based on influences of basic events using statistical data and a merging logic. Fig. 2.7 (b) shows the Fault Tree Architecture from (a) transformed into a Success Tree for availability analysis of technical systems. For development time analyses using Success Trees, however, the identification of weaknesses in the system architecture would become more difficult, since failure rates of system components cannot be directly related to higher-level system failures. Here a corresponding mean time to repair (MTTR) for certain failures would have to be defined at development time to assess the trouble-free time for normal operation. Since the focus for development time safety analyses is typically on reliability, the Success Tree modeling technique is not very widespread in comparison to Fault Trees. In some cases, however, it also makes sense to integrate Success Tree architectures into Fault Tree architectures to obtain a more realistic system model. This is the case if there are dependencies between the branch point events. For this purpose, Andrews et. al. [58] introduced the Event Tree Analysis using BDD transformations.

For the planned application context, such a modeling approach using Success Trees appears promising. It could be used for modeling the composition of runtime evidences (RtEs) and for runtime analyses of networked system functions. While input A, B, C could be provided by different sources, their aggregation could ensure safety for an integrated overall functionality. In this way, the modeled Success Tree could represent a traceable and systematic merging concept for a continuous integration of distributed runtime evidences.

A drawback, however, is that when using the classic Fault Tree or Success Tree structure, only one TLE can be considered for safety assessment. In this way, it can only be evaluated whether the entire system or a certain sub-function is working correctly during operation or not. It would therefore not be possible to enable fine-grained adaptation processes with a uniform model. Further relevant aspects for assessing the overall system safety would have to be analyzed separately. Such an approach would thus not be suitable for modeling complex system safety dependencies. Consequently, regarding the addressed application context, a more fine-grained distinction according to certain collaboration capabilities has to be established.

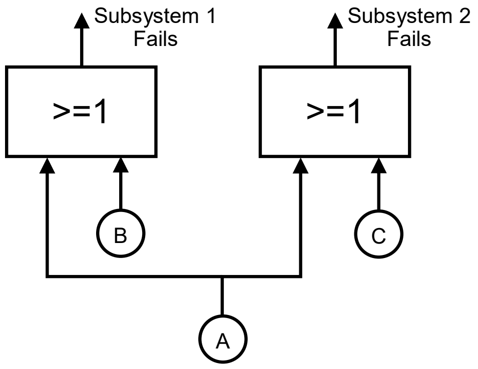


Figure 2.8: DAG with two Top-Events

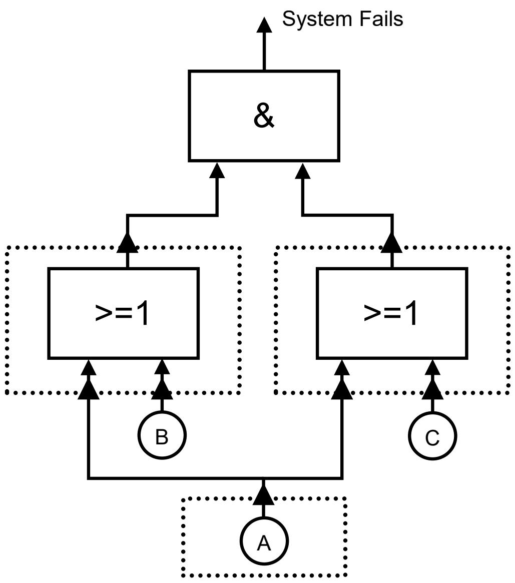


Figure 2.9: Component Fault Tree

The consideration of several TLEs can be implemented using so-called Directed Acyclic Graphs (DAGs) as shown in Fig. 2.8. The different output events in a DAG could be facilitated to show various different output effects on the system behavior. Kaiser facilitated DAGs in combination with a modularization concept and introduced the Component Fault Trees (CFTs)[59] as shown in Fig. 2.9. Applying CFTs certain failure causes as well as an internal merging logic could be related to technical components using a modular representation with an interface specification. Reusability as well as information hiding becomes possible in this way. Thus also repeated basic events could be related to a single source in a certain module.

These features also appear helpful for specifying runtime evaluation models. Comparable to CFTs also modularized Success Trees for runtime evidences could be established. For the addressed solution approach the intended modules would not refer to technical hardware components, but to certain networking capabilities of networked systems. Möhrle [60] introduced a comprehensive interface semantic for automatically arranging specified CFT modules in complex systems. Using a component-based architecture, Domis [61] focused on the integration of safety analyses and component-oriented development processes, which could provide valuable insights in this regard.

There are already many other safety assessment approaches available that focus on the propagation of safety quality features in order to enable automated analyses at development time. However, these approaches cannot be adapted without further ado to assess the overall condition of networked systems during operation. Since these approaches depend on development time knowledge they focus on the propagation of failure states and not on failure-free conditions for normal operation. Moreover, they inherit no lightweight runtime mechanism to integrate external runtime evidences comparable to the ConSerts approach as outlined in the previous Section 2.4.1.

Prominent approaches are the Failure Propagation and Transformation Notation (FPTN) from Fenelon [62] and the Hierachically Performed Hazard Origin and Propagation Studies (HiPHOPS) as introduced by Papadopoulos [63] [64]. Both approaches aim at safety assessment in complex systems by specifying holistic propagation models for failure influences using dependency graphs. Lisagor introduced the Failure Logic Modeling (FLM) [65] approach which inherits FPTN as well as HiPHOPS features to integrate failure information data in a systematic way. However, these approaches focus on holistic development time analyses to consider as many aspects of system behavior as possible. In the target application context, the focus is on lightweight runtime evaluations of external service integration to adapt system behavior in a predefined way. Adler introduced a hybrid CFT approach [66], where a component-based Fault Tree architecture is combined with priority mapping mechanisms in each module to determine the possibility that an adaptive system assumes a certain configuration considering failure influences.

Other approaches are not focused on failure propagation in a system, but on modeling the entire dynamic system behavior and its possible failure states. This can be for instance Dynamic Fault Trees [67], which enable the modeling of system behavior and interactions of subsystems using markov chains. Also State Event Fault Trees (SEFTs) [68] as introduced by Kaiser enable this. A benefit of these methods is that cyclic system dependencies can be modeled and thus the evaluation result is probably more realistic. However, these approaches are also designed for safety assessment at development time using automated evaluations of complex system characteristics. Additionally, a major drawback of these approaches is that the modeling of complex system dependencies can lead to a state space explosion. Considering the vehicle domain as the target application domain, however, it would probably become difficult to model all possible driving situations in convenient system states, as outlined in Section 2.1.2.

When modeling the integration of external services using runtime safety monitors, not only safety but also security issues need to be considered. Schoitsch et. al. [69] expressed the need for a safety and security co-engineering, co-certification and standardization. Attack Trees [70], as outlined by Schneier, enable the modeling of internal system faults and external hacker attacks in a unified fault tree model. Since the success rate of hacker attacks on a particular system are sometimes difficult to quantify, a combined qualitative and quantitative safety analysis was introduced by Steiner. The so-called Security enhanced Component Fault Trees (SeCFTs)[71] enable an enhanced system safety analysis regarding security events. These approaches could be helpful to identify vulnerabilities for the provision or consideration of external services in the planned system architecture at development time. Besides optimizing the system structure at development time, a runtime monitoring of security events could also support a safe networking of systems. This could be achieved by not providing or considering shared services in a network in case of identified threats.

2.4.4 Runtime anomaly detection and dynamic reconfiguration

As outlined in the previous Section, Fault Tree Analysis is still a standard technique for assessing safety of complex systems and has been further refined in many aspects. An essential benefit of this technique is that in the design phase the propagation of faults can be vividly modeled with the help of dependency graphs and merging logics. Thus, safety experts can quickly understand and modify already specified fault tree models and, based on this, optimize the existing or planned system architecture. Moreover, the evaluation of the overall system safety using FTA can be carried out reliably in an automated manner. This is made possible by its lightweight evaluation mechanisms, such as BDDs. These features also appear to be advantageous for modeling runtime safety monitors in combination with associated dynamic reconfiguration mechanisms.

With regard to the intended solution approach, however, it became apparent in the previous Section that these techniques focus only on development time knowledge. Even if dynamic aspects of system behavior are modeled, this is only meant to describe the entire system behavior as completely as possible from the development time perspective. These approaches do not aim at mapping the current system condition during operation to a state modeled at development time in order to better assess the current runtime situation. This is because they are only utilized to optimize the system architecture at development time. Accordingly there is no need to consider runtime aspects afterwards. However, as already described, in the target application domain of networked vehicles, not all conceivable hazards can be excluded already in the design phase. Consequently, condition monitoring is of central importance for assessing safety at runtime. In this way, warning states can be detected at an early stage and adequate countermeasures can be initialized. Ideally, the occurrence of hazards can be prevented in this way.

There are a few safety assurance approaches using Fault Trees that take runtime aspects into account in order to continuously monitor system conditions and reliably evaluate their effects on overall system safety. Shalev et. al. [72] introduced the so-called Condition-based Fault Tree Analysis (CBFTA), which makes it possible to update predefined fault trees with runtime data. They combined condition monitoring techniques and statistical data as input data for FTA to better predict the occurrence of certain hazardous events. By using runtime checks, the entire system status can be determined more realistically during operation. In this way, the analysis possibilities with fault trees can be expanded from pure design time analyses to recurring life cycle analyses for the entire period of use. This makes it possible to deal with identified anomalies in a running system in a timely and targeted manner.

Since the derivation of necessary safety assurance concepts is a complex and sometimes also a creative task, it cannot be automated in an arbitrary manner. For the development time FTA techniques as well as for the CBFTA, safety engineers interpret the analysis results and implement suitable solution concepts either in the design phase or during a temporary shutdown. This can involve significant decisions such as a system redesign or the replacement of components that require a completely new safety assessment.

Other approaches using Fault Trees such as Li [73][74] do not consider statistical data, but only runtime data for a continuous safety assessment during operation. The continuous evaluation of the TLE with runtime data is performed here to supervise the safe flight of an unmanned air vehicle (UAV). Since no statistical data is used, the quality of the analytical outcomes also changes. It is no longer a probability rating for the occurrence of the TLE in a certain time interval, but an assessment of the current overall system status based on the conditions of subsystems. Thus the existence of an actual fault which influences the overall system safety can be proven and suitable countermeasures can be initiated to avoid a critical system failure.

For the intended scenario of networked driving as well as the UAV scenario mentioned above, a shutdown of the system and a manual interpretation of the evaluation results is not possible due to the dynamic context. Hence, all modifications must be carried out in an automated manner during operation. Since the system modifications stay at the behavior generation level, the respective correct reaction behavior can be determined beforehand. To enable this, all possible evaluation results have to be analyzed already at development time and a conclusive overall safety concept has to be specified.

The possibilities for automated fault tree evaluation using the standard FTA are, however, clearly limited, regardless of whether only runtime data or also statistical data is taken into account. Certain system properties cannot be clearly differentiated from one another, as is the case with the component orientation of CFTs. Consequently, an approach using DAGs for the continuous evaluation of several critical factors in parallel seems suitable. In addition, similar to CFTs, a modular structure could support grouping of certain recurring elements for efficient modeling as well as evaluation.

As outlined before, however, the analysis results are not facilitated for development time analyses, but for safety critical runtime behavior adaptation. There must also be a mechanism to link the various monitored system conditions appropriately to the behavior generation layer, as described in Section 2.2.2. The formalized runtime safety evaluation for integrating individual external safety-related services must be associated with reliable dynamic reconfiguration mechanisms. To enable safe dynamic system reconfigurations using an approach based on pre-configured variability could be the key, as outlined in Section 2.3.2. However, there is currently no suitable solution approach available that addresses these two aspects with a consistent solution model.

2.5 Summary and Conclusion

In the following, the main findings of the state-of-the-art analysis of the adaptive system community as well as the safety community are briefly summarized. Furthermore, the need for such a solution approach is emphasized from the perspective of both research communities.

2.5.1 State-of-the-art from the adaptive systems community and the safety community

Basically, the Adaptive Systems Community provides a variety of assurance means for dynamic reconfiguration, but safety is typically only one aspect among many others. There are manifold approaches, which focus on validation and verification means for system adaptation processes. Basically, this helps to reduce the number of variants, which have to be analyzed at development time, but it provides no complete runtime safety assurance means for the context of networked driving. In the second research roadmap for self-adaptive systems, De Lemos et. al. [75] expressed the need for stricter safety assurance approaches. There are also assurance approaches, which explicitly focus on runtime hazard analysis. But typically they only consider development time knowledge for a safety evaluation at integration time. In this way, the runtime variability for safety assurance can be limited, but no further dynamic safety parameters can be taken into account. In the context of networked driving, however, dynamic safety aspects are particularly important to trigger adequate reconfiguration mechanisms.

The safety community naturally has a very conservative attitude regarding dynamic reconfiguration mechanisms and tries to avoid dynamic variability as far as possible. Most of the safety analysis and assurance approaches are developed to support engineers during their development time activities. They lack an adequate degree of formalization that would be required for automated runtime adaptations of system behavior. However, it became evident that it is unavoidable to shift parts of the safety assurance means to runtime. If these promising networking concepts should be established, certain variable degrees of freedom for involved systems are indispensable. Hence, when bringing together the viewpoints of the adaptive systems community as well as the safety community, it is most promising to focus on approaches with the smallest amount of runtime responsibility. In the safety community, Rushby introduced the idea of safety kernels. He suggested to shift parts of the safety assurance means to runtime in order to enable dynamic reconfiguration mechanism of systems in safety critical domains. Later Schneider introduced ConSerts as the first feasible implementation approach for runtime certification. It provides valuable insights for establishing higher level functionality using conditional safety certification. However, since the approach is aligned to safety assurance for flexible plug and play approaches, its dynamic variability is largely limited to the integration time. In the highly dynamic and variable application context of networked driving, more runtime variability for dynamic reconfigurations is required.

2.5.2 The need for a safe optimization approach for networked systems using external services

There are already many researchers dealing with the safe networking of systems, especially in the vehicle sector. Oestberg dealt with the question how the networking of vehicles for the exchange of safety-relevant data can be realized with standards such as AUTOSAR [76][77]. For this purpose, he suggested to introduce an individual database to each vehicle where safety-related vehicle data such as sensor data can be stored. In the next step, the data bases should be synchronized between the collaborating vehicles to optimize their individual safety assessment. Based on that, Oestberg concluded that a safety contract concept is necessary for the dynamic safety assessment, but he did not provide a concrete one. Priesterjahn introduced a runtime safety analysis for networked vehicles based on failure propagation models [33]. However, this safety analysis approach did not take into account the reaction phase.

As a conclusion for this Chapter, it can be stated that currently no suitable safety assurance approach is available for the envisaged application context. A flexible runtime certification approach is required to dynamically optimize system behavior through individual external services. This requires a fine-grained conditional safety certification approach to ensure safety based on lightweight runtime checks in combination with predefined reconfiguration mechanisms according to the permitted degrees of freedom.

3 Solution Overview

This Chapter provides a conceptual overview of the Dynamic Safety Contracts (DSCs) solution approach, which is the main contribution of this thesis. For this purpose, this Chapter explains the key aspects of specification and implementation as well as the addressed application context in a descriptive way. It represents a good starting point to delve deeper into specific characteristics of DSCs. This can be done by following the corresponding references to Chapters 4 and 5.

Initially, the first Section presents a brief solution overview of the developed contract-based safety assurance approach with regard to current state of the art and the specified key characteristics of DSCs. In the following Section, the addressed target application domain is outlined as well. In this context, especially the working principle of collaboration processes of networked vehicles and infrastructure in the automotive domain is described. This particularly helps to clearly understand the derived modeling approach for dynamic behavior adaptation in the following Sections. Section 3.3 then discusses fundamental safety assurance strategies for dynamic self-optimization of individual networked systems by integrating shared safety knowledge. The following Section 3.4 summarizes the required key features for specifying reliable behavior adaptation concepts in the outlined target application context. Please note that further aspects of the derived features of the solution approach as well as a detailed description of the lack of current state-of-the-art safety assurance approaches can be looked up in Chapter 2. In the next Section 3.5, a basic modeling architecture for DSCs is derived from the identified key characteristics. Subsequently, Section 3.6 describes how the pre-configured DSC networking modes can be utilized to enable openness for previously unknown systems at integration time. Section 3.7 summarizes the conducted operationalization steps for the DSCs approach. This comprises the conducted design decisions for the GUI-based DSC modeling as well as necessary engineering steps to provide computable representations of DSCs. Finally, the last Section 3.8 outlines how DSC networking concepts can be established by applying standardized safety engineering activities to ensure operational safety and performance in a certifiable manner.

3.1 Solution Approach: Dynamic Safety Contracts

The most important finding from the conclusion in the state of the art of Chapter 2 is that there are at the moment no directly applicable solution concepts available, which fulfill the demands of safety assurance and certification for specifying dynamic behavior adaptation concepts in the addressed application context of networked embedded systems. Since safety is a key prerequisite for the practical use of such flexible networking concepts, there is an inescapable demand for the development of new suitable safety assurance techniques where parts of the safety assurance means can be shifted to runtime. Nevertheless, it became apparent that the safety research community has recognized the increasing demand for flexible runtime safety assurance means and strengthens its research efforts in this area. This can be observed by the recently proposed just-in-time certification approaches as described in Chapter 2 as well as a variety of research projects related to this application domain, such as the CrESt (Collaborative Embedded Systems) project [78] or the ARAMiS II project [79]. In parallel, today already manifold research has been done in the adaptive systems community and has influenced this work in numerous ways. Important preliminary results, which contributed to this solution approach, will be mentioned wherever required in this thesis.

As it was outlined in the initial description of the solution approach in Section 1.4, the DSCs approach aims at shifting necessary parts of safety assurance and certification activities to runtime. A reliable runtime safety assessment of the current operating situation of networked systems, taking into account the partners involved and the environment, should provide additional assurance means to support safety assurance and certification measures. The core solution technique of this thesis, to facilitate this, are *Dynamic Safety Contracts*. With the help of DSCs, dynamic adaptation concepts for networked embedded systems can be specified in a systematic way at development time depending on the availability of safety-relevant data from decentralized sources at runtime. During operation, a key feature of DSCs is therefor to integrate and interpret decentralized safety-related runtime data from separate sources in a predefined and reliable way.

To enable this, a safety-assured integration concept for available external safety-related data can be specified using DSCs with respect to certain safety-critical features of a networked mode at development time. This is done using so-called DSC-Modules. They contain a modularized set of predefined DSCs to specify a degradation architecture for critical safety features in networked applications. A specified DSC-Module can be reused in other similar networked applications. It can be continuously evaluated during operation based on the currently available shared safety guarantees. By this means, the impact of dynamically varying services on the networked system applications can be systematically specified at development time. On this basis, a safety assurance concept for a networked mode can be completely predefined before commissioning. Each DSC-Module thus represents a runtime safety monitor for a specific safety feature during operation. By supervising a set of interconnected DSC-Modules, a holistic virtual safety cage for adaptation processes of the networked system behavior can be realized in this way. It thus becomes possible to limit the runtime variability of networked system applications to well defined degrees of freedom and to provide a dynamic adaptation concept in a predictable and reliable manner. Beyond that, the DSC approach is a viable solution for the runtime safety certification of networked system applications, as it largely limits the runtime safety responsibility of systems in collaboration scenarios.

3.2 Networking Autonomous Vehicles to Optimize Driving Behavior

Before explaining the DSCs approach in more detail, however, the target application domain for specifying dynamic adaptation behavior is described in the following. The DSCs approach aims in particular at standardized networking approaches in the automotive domain in order to link the various automotive systems and infrastructure devices with each other. Since this application domain is also very illustrative, this Section helps to better understand the resulting dynamic safety dependencies of involved systems in an established networked mode.

A central characteristic of the addressed collaboration scenarios is that the focus is always on a single networked vehicle, which tries to collaborate with surrounding vehicles and infrastructural devices in a previously unknown environment. External data is incorporated into a particular vehicle platform to optimize its driving behavior in a predefined way based on a networked driving mode specification. It benefits from the collaboration, by especially taking external safety-related knowledge into account in order to improve the individual situation awareness or to enable coordinated driving processes. A suitable safety assurance and certification concept, such as the proposed DSC approach, is a prerequisite for this. DSCs are designed to enable both by evaluating the runtime availability of in- and external safety guarantees in a networked operation mode and by dynamically adapting the existing system behavior accordingly.

Naturally, it must be taken into account in this context that not every external source is trustworthy. Forwarded data can be manipulated or just corrupted, which could cause additional hazards. Ideally, this can be clarified before initializing a networked application. For that reason, appropriate internal safety and security measures have to be implemented on the involved system platforms to ensure that the potential collaboration partners and their transmitted data are approved for a particular networked mode. However, since this thesis concentrates primarily on runtime safety assurance and certification means for the ongoing networked application, these conventional safety measures were not in the focus of this thesis, but are partially addressed in Chapter 7. For dynamically establishing new collaboration processes with previously unknown systems, a suitable negotiation process is outlined in Chapter 3.6.

Basically, the need for the integration of external safety-related data arises due to the spatial restrictions of sensors on a particular system platform. Although powerful sensors might be implemented on a certain system platform, physical restrictions like “looking around the corner” cannot be enabled by an isolated single system. Also understanding the behavior or even forecasting the future behavior of surrounding vehicles might be

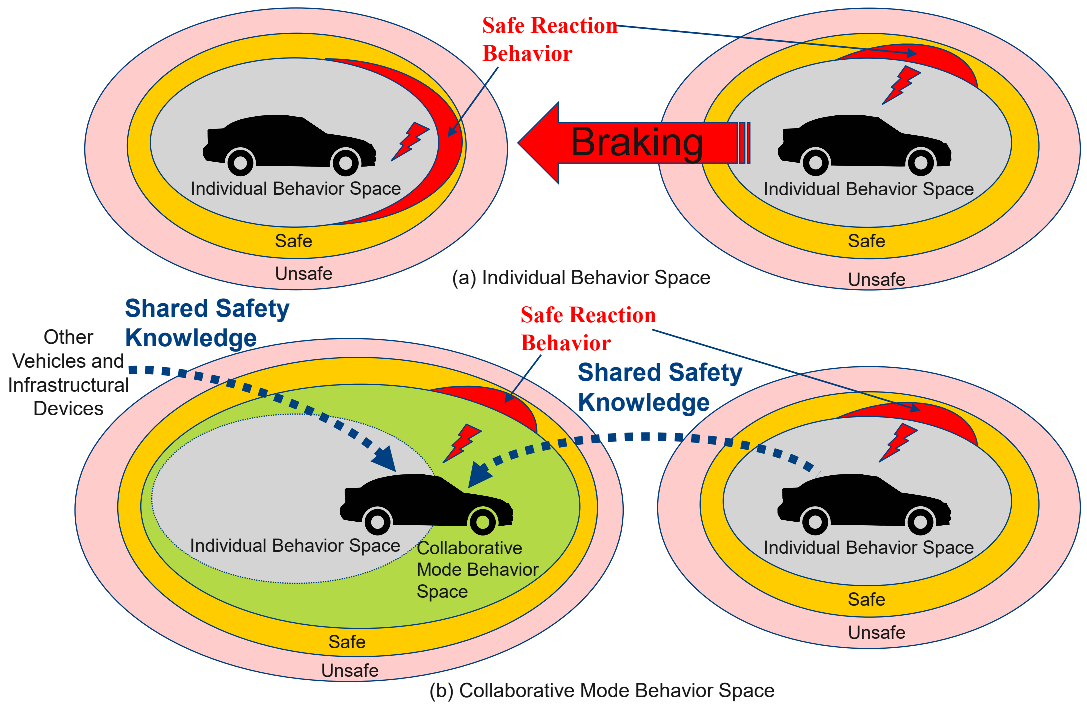

Safe Reaction Behavior
Braking
Individual Behavior Space
Safe
Unsafe
Individual Behavior Space
Safe
Unsafe
(a) Individual Behavior Space
Shared Safety Knowledge
Other Vehicles and Infrastructural Devices
Safe Reaction Behavior
Shared Safety Knowledge
Individual Behavior Space
Collaborative Mode Behavior Space
Safe
Unsafe
Individual Behavior Space
Safe
Unsafe
(b) Collaborative Mode Behavior Space

Figure 3.1: Comparison of Individual and Collaborative Behavior Space

impossible just by monitoring them from an external point of view. Hence, an appropriate solution would be to network these systems and their sensors to gain an enhanced situation awareness. For a system involved in such a collaboration group, this could provide additional spatial perspectives for a better environmental perception as well as new insights into surrounding systems for a better risk assessment. Also for enabling coordinated driving processes, networked driving modes can be specified in such a way that autonomously driving vehicles are dependent on the forwarding of external services and can thus be better controlled from an external point of view.

Figure 3.1 shows in a simplified manner a typical collaboration scenario with a platooning use case. The focus in this illustration is on the operating principle of the addressed collaboration processes as well as the associated safety assurance concept. The reason for utilizing the platooning use case is, as it was already described in the previous Chapters, that safety assurance for platooning processes in different facets was a central research topic during the development of the DSCs approach. To highlight the benefit of the addressed system collaboration, the graphical depiction in Fig. 3.1 compares the vehicle platoon in the upper layer, where no collaboration takes place, with the vehicle platoon in the lower layer, where a predefined collaborative mode is initialized for the following vehicle. Both platoons are driving in an autonomous manner and have a safety assured individual behavior space based on the individual non-networked driving capabilities.

If the autonomous vehicles are driving in an individual autonomous driving mode as shown in the upper layer, safety relevant runtime data is not shared within the platooning group. The consequence is that interaction processes have to be carried out in a more conservative way to ensure safety. The most obvious disadvantage for the platoon is the larger average distance between vehicles while driving. The result is a lower utilization of the transport infrastructure and a higher fuel consumption for the following vehicle since a restricted slipstream driving leads to a higher air resistance.

In such a scenario, where no collaboration takes place, various other drawbacks can be identified for the following vehicle. A general disadvantage of such a driving situation is that the front vehicle covers a crucial part of the sensor perception area of the following vehicle. Due to the absence of a networking concept for the integration of shared safety knowledge, it is not possible to adapt the driving behavior of the following vehicle preventively to hazards recognized by the leading vehicle. In fact, the reaction behavior to detected hazards from the leading vehicle, such as braking or evasive maneuvers, could even cause additional collision risks for the following vehicle. Hence, the only possible option for the following vehicle is to closely observe and try to interpret the driving behaviour of the front vehicle. In case of detected behavior changes of the front vehicle, however, it has to assume for safety reasons a possible worst case situation. Ultimately, this leads to a severely restricted ability of the vehicles to interact in such a non-networked platoon.

In contrast to that, by introducing collaboration processes based on shared safety knowledge, the follower vehicle can benefit from having a vehicle in front as shown in the lower layer. It can extend its individual degrees of freedom for an optimized driving behavior according to safety and performance criteria. In Fig. 3.1 the optimized behavior space is summarized as *Collaborative Mode Behavior Space*. The forwarded safety relevant runtime data helps to get a more complete picture of the current driving situation considering the environment and surrounding systems. Furthermore, in case of occurring hazards the collaborative safety knowledge can be facilitated to enable a more situation-adapted reaction behavior to bring the system back to a safe state. In this way, the system collaboration allows better aligned driving processes with smaller average distances between the considered vehicles.

While in a typical road traffic scenario the scope of vehicle networking is probably limited to an enhanced situational awareness, it is also conceivable to specify networked driving modes for coordinated driving processes. This was implemented for the presented use case for master-slave driving in the off-road sector. As part of a research project, master-slave vehicle fleets should be developed to enable driverless slave vehicles in order to reduce the number of human drivers. A detailed description will be provided in Section 4.1. Later in this thesis, a further use case from an industry research project for coordinated driving processes is presented. This use case addresses connected driving in a parking garage and is referred to as intelligent automated parking (IAP).

Considering the vehicle platoon in Fig. 3.1 for the master-slave driving use case, the following vehicle in the platoon was typically considered as a driverless slave vehicle. The allowed movements of the slave vehicle in the illustrated *Collaborative Mode Behavior*

Space in Fig. 3.1 are strongly dependent on the provided services of the master vehicle. In parallel, the individual behavior space for autonomous non-networked driving is strongly reduced. The master vehicle thus represents an additional supervisor level and provides its required safety knowledge to the slave vehicle. In this way, safety for the autonomous driving processes of the slave vehicle should be increased. Nevertheless, the slave vehicle still has to ensure safety on its own and the master vehicle only imposes additional constraints to support the safe autonomous driving.

At this point, it should be noted once again that the safety assurance concept for the illustrated collaboration process in Fig. 3.1 is limited to sharing safety knowledge within a collaboration group and to optimize and extend its individual behavior space accordingly. A tight system coupling of collaboration partners, maybe by using predefined interaction sequences of the overall networked vehicle fleet is not part of the collaboration scenario. In this way, although a combined situation awareness for the platooning group is utilized, the involved collaboration partners can still be distinguished as separate legal entities. Safety responsibility can thus be limited to a single collaboration partner. Ultimately, this enables the introduction of lightweight and customized safety assurance and certification solutions for the individual networked systems.

3.3 Specifying Flexible Networking Concepts for Loosely Coupled Systems

The dynamically changing environment with surrounding vehicles and infrastructural devices in a typical road traffic scenario, leads to a huge number of potential collaboration partner in a limited period of time. In principle, additional beneficial collaboration processes with other systems should be established, whenever they come into the communication range. For a particular networking approach, the total number of systems, which are capable of participating in collaboration processes, is probably the most important factor for performance reasons. In the described platooning scenario, for example, the collaboration group does not necessarily has to be limited to the two illustrated networked vehicles. Additional safe collaboration processes with other surrounding systems could be established subsequently or even in parallel. Hence, in a suitable collaboration process specification, the integration as well as the separation of diverse additional collaboration partners while operating a certain networked mode have to be considered. These capabilities decide on its general applicability and thus its operational effectiveness in the target application context.

This holds especially true for collaboration scenarios in public road traffic, where a large variety of different systems has to be considered for potential collaboration processes. The diversity of systems, thereby, can also be increased by further aspects like modifications in hardware or software. They could affect the particular system capabilities for participating in collaboration processes. Furthermore, as mentioned before, collaboration processes with several different systems in parallel should also become possible. In the platooning scenario, for instance, a collaboration network with infrastructural devices based on vehicle-to-infrastructure (V2I) communication should be established parallel to the vehicle-to-vehicle (V2V) communication. As a result, even more possible system combinations have to be addressed for networked groups of systems to establish meaningful collaboration processes.

Considering the resulting manifold collaboration scenarios, a safety assurance concept, which focuses on highly customized solutions for the various systems and their specific interaction processes in an open and highly dynamic application context, could become a very complex engineering task. In this case, the high engineering effort for each networked mode would probably lead to a low distribution of such a networking approach. An important objective is therefore to reduce the respective engineering effort for new collaboration processes to an acceptable level. This could be achieved by standardizing certain collaboration capabilities that are frequently used to make them reusable for other networked modes. In addition, in order to increase the reuse rate for the development of networked modes, certain rarely used collaboration capabilities can be excluded, provided that the resulting networked application is not too severely restricted. This could be the case, for example, if the considered collaboration capabilities only refer to particular systems type combinations with a very specific application context, such as a group driving behavior for particular vehicle types in a certain road environment.

To take this into account, the DSCs approach tries to limit most of the safety assurance means to the individual involved system platforms. Comparing such a segregated safety assurance approach to an overall approach for the whole group of systems, which maybe includes a specified group behavior, this should enable less complex safety assurance measures for a specific collaboration scenario. This enables, while considering the impact of shared safety knowledge on the individual system safety, to largely ignore the respective external platform features, provided that no direct influences exist. In this way, safety engineers can focus on the impact of various dynamically changing services on a particular system platform. Ultimately, it should thus enable more flexible collaboration options for the various systems. Surrounding vehicles, for instance, typically pose a risk of collision. Hence, their intended driving behavior, their internal system condition as well as their spatial position could be beneficially considered for a runtime safety assessment of a collaborative mode. Other aspects, however, which have no direct influence on the collaborative mode safety, may be excluded. In other collaboration scenarios, where external data is forwarded by infrastructural devices only with regard to environmental aspects, it seems even possible to completely omit these external platform features. This becomes possible because the collaborating systems themselves do not pose any threat in this case. Hence, limiting the safety assurance approach to individual systems in a networked mode has the potential to simplify the analysis of the existing system interdependencies and to reduce the resulting system complexity for safety assurance measures.

A drawback of such collaboration scenarios of loosely coupled systems, however, is that the performance of an application in a networked mode tends to be slightly lower. During operation of the addressed DSC collaboration processes, the illustrated *Safe Collaborative Mode Behavior Space* in Fig. 3.1 limits the individual behavior space for specifying a collaborative mode behavior. Assuming a certain group driving behavior, the individual behavior spaces of systems could possibly be merged to some extent. Thus, for example, more complex mutual interaction processes could be established.

In comparison, however, next to the resulting more complex safety engineering activities, a specified group behavior has further crucial disadvantages. That is why we decided not to further evaluate this approach. Basically, it has to be clarified who is responsible for safety in case of damages to the involved participants. It is also necessary to clarify according to which rules certain automotive systems in hazardous driving situations are prioritized. In addition, the option to remotely control driving commands as required for a group behavior can be considered as a critical system vulnerability for hacker attacks. Moreover, it can be assumed that the specification of a group behavior with more than two vehicles is very complex and difficult to implement.

Considering the before mentioned, it can be summarized that the most important objective for a suitable solution approach is to introduce more flexible collaboration concepts for frequently changing collaboration partners and scenarios. With the described individual safety assurance approach the allowed participants for a collaborative mode specification doesn´t depend on a certain combination of systems and their combined behavior, but on certain forwarded safety knowledge of systems. In this way, it becomes possible, that systems involved in a collaborative mode share safety-related data without knowing the effect on the respective collaboration partners. Such a decentralized approach could foster the standardization of collaboration capabilities and service provision concepts for the different available system platforms in a specific domain. It could thus become possible to specify new collaboration processes with a reduced engineering effort due to a high reusability rate from other similar collaborative modes.

3.4 Ensuring Safety for a Modified Driving Behavior

By trying to utilize potential state-of-the-art approaches for safety assurance of the addressed collaboration processes, certain drawbacks became visible. A detailed summary of these approaches can be looked up in the previous Chapter. A major difference from today´s typical safety assurance domains to the addressed collaboration processes is that safety has not only to be assured for a single system in a closed context. Instead, it has to be assured for an integrated collaborative mode established by distributed systems that interact with an open and unknown environment. A reasonable approach to tackle the diverse collaborative situations would be to ensure safety by introducing a holistic state-based model for collaboration processes. Each state could be individually assessed with development time safety measures. Consequentially, the result would be a very complex adaptation space and reaching a sufficient level of completeness would become a major challenge. In the conclusion of the state-of-the-art Chapter it was already outlined that the combination of an open context and the performed system interaction processes leads to a high-degree of complexity.

Another crucial safety aspect, which was already outlined, is the dynamic application context of the planned collaboration processes in the automotive domain. Emerging hazards have to be tackled rapidly and in a manner that is appropriate for the specific driving situation. In such an application context, if a safety assurance approach would only rely on development time knowledge, substantial worst case assumption would have to be taken into account. Ultimately, this could significantly reduce the overall performance of collaboration processes due to many additional protective safety measures. Moreover, occurring hazardous situations in a networked mode might not be tackled in an optimal way.

In contrast to that, by shifting necessary parts of safety assurance means to runtime, many of these problems could be solved. Dynamically changing safety parameters from involved collaboration partners and environment could be assessed and facilitated for safety certification issues. The decentralized safety knowledge within a collaboration group could thus be merged and analyzed in a real time manner. With the objective to determine safety for an active collaborative mode, the current state of an ongoing system collaboration established by spatially distributed systems could be assessed in this way. Basically, such an approach could help to reduce the adaptation space to a manageable size and to introduce adequate countermeasures for detected threats in the dynamic context. It has to be clarified, however, in which way safety-related data from involved system can be aggregated in a uniform manner to ensure safety for the overall integrated collaborative functionality. In addition, a runtime representation for safety-related data has to be specified that supports safety engineering activities as well as efficient runtime evaluation techniques.

A beneficial strategy to describe complex safety dependencies is modularization as it was already outlined by Schneider [55]. In the safety community there is a general trend towards modular safety engineering and certification as it is described in Chapter 2. In fact, modular approaches for safety certification are already widely accepted and used in safety-critical applications. Nevertheless, today´s state-of-the-practice modular safety engineering and certification approaches does not inherit the required key characteristics. These approaches typically focus only on development time safety certification activities. Many of these techniques try to pre-certify separate submodules in order to reduce the complexity for overall system certification. Ultimately, this should allow incremental certification steps to facilitate the integration of new submodules.

For the planned solution approach, the advantages of using a modular approach for safety engineering and certification are manifold. Next to reducing the complexity, safety engineers can focus on specific aspects related to their expert knowledge and differentiate it from other related domains. In this way a holistic safety assurance concept can be developed step by step in a systematic way. For safety engineering, existing modularized

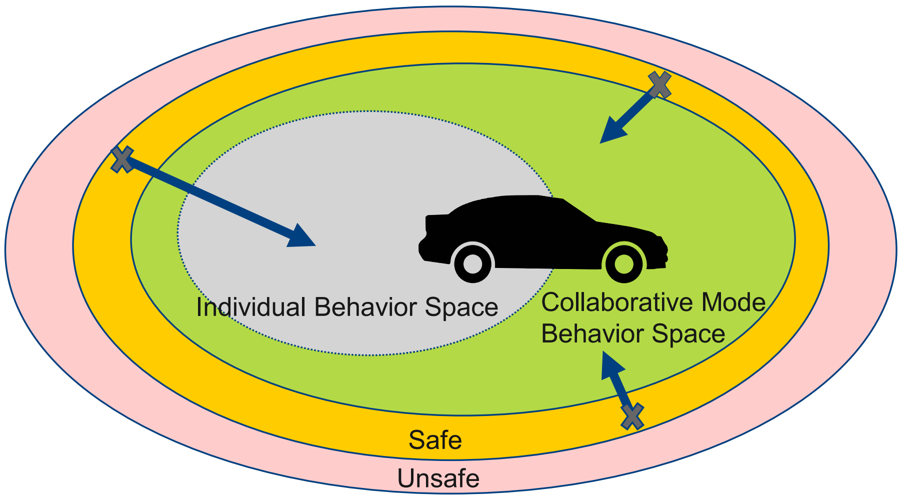


Figure 3.2: Specifying Safe Return in Networked Applications

knowledge from other collaborative modes can ideally be reused. As a precondition, of course, the safety-related content has to be decomposable to the required level of granularity. For automated runtime evaluation, the modularized safety content has to be available in a formalized way to enable an adequate data exchange between the different modules. Furthermore, the forwarding of services between modules has to be done in a fast and reliable way to ensure an efficient runtime evaluation.

An efficient runtime evaluation technique in turn can be facilitated to specify a situation-adapted virtual safety cage for networked driving. Although complex system function might be performed during collaboration of systems, the allowed degrees of freedom could be specified in a simplified way based on continuously evaluated threshold values. Such a safety cage specification could be specially developed for a dedicated system involved in a collaborative mode. Whether a situation assessment during operation of a collaborative mode is evaluated to be safe or unsafe as well as the level of criticality, can be part of the safety cage specification. Depending on the respective evaluation result, the allowed functional scope could be reduced or enlarged and a suitable safe adaptation process could be triggered. Accordingly, safety goals can be systematically derived for diverse situations that could occur during operation of a collaborative mode. All necessary safety engineering activities for the safety cage description can be performed at development time. Hence, it can be facilitated as a key element of safety argumentation for the certification of a collaborative mode description.

In case that a system is evaluated to be in an unsafe operation mode based on the virtual safety cage description, it has to be transferred back to a safe mode as optimally as possible. The addressed dynamic and open application context requires a timely and situation adapted reaction behavior. Hence, the reaction behavior as part of the virtual safety cage description is crucial for safety assurance and risk mitigation of a collaborative mode description. Considering the illustrated *Collaborative Mode Behavior Space* in Fig.

3.2, an optimal reaction behavior for all potential hazardous situations, which could occur during operation of the collaborative mode, has to be specified. In this context the term "optimal" means that the safety as well as the performance perspective should be considered. However, it should also be noted that a reaction behavior in a dynamic context can itself create a risk. For instance a breaking maneuver could cause a rear end collision with a following vehicle during platooning. Hence, the correct initialization of a certain behavior has to be guaranteed by safety means. As mentioned before, this could be enabled with a formalized situation assessment based on a runtime safety evaluation. In this way, a specification of a safe reaction behavior for a collaborative mode would become possible. This would enable to clearly specify and justify with a valid safety argumentation the execution of a certain reaction behavior with respect to the available runtime safety evidences. This, however, requires that the effect of each forwarded safety-related service with degraded or even contradictory safety evidences on the respective collaborative mode is part of the collaborative mode safety specification.

The following key features, which have been identified, can thus be summarized as follows:

1. Runtime Evaluation Mechanism: Considering rapid changes of service availability as well as environmental changes, an adequate representation for dynamic variability is needed, which evaluates the impact on the overall system safety at runtime. For an efficient runtime composition mechanism of the spatially distributed safety knowledge a lightweight runtime check has to be introduced. Finally, the separation between safety assurance measures at development time and at runtime must be carried out in a way that the dynamic collaboration context is optimally addressed.

2. Modularization of Networked Safety Features: For specifying the integration of external services in a systematic way, a structured modeling approach has to be defined. Partitioning complex safety dependencies into manageable safety arguments such as a modularized demand-guarantee modeling can be part of the solution. A modularized approach can associate different types of merged services with certain system safety features.

3. Consistent Service Composition Concept: With respect to requirement 1 the evaluation concept must be implemented in a lightweight way to support continuous and fast evaluation processes in the dynamic collaboration context. This requires a standardized communication concept with a suitable formalization of safety-related parameters. In addition, a representation for specifying the available safety dependencies on a particular system platform has to be introduced. This is ideally done in a compositional form based on demand-guarantee relationships.

4. Virtual Safety Cage for Modified System Behavior: For certifying safety for a networked system application, the allowed degrees of freedom for the intended system platform must be clearly specified for all potential driving situations during collaboration. Considering the before mentioned requirements, the situation assessment

has to be implemented in a formalized way based on the modularized safety features. Regarding certification aspects, it should be possible to completely specify a resulting virtual safety cage description at development time based on safety considerations.

5. Fast and Reliable Reconfiguration Mechanisms: If the current collaborative situation is evaluated to be unsafe based on the safety cage specification, the dynamic context requires appropriate safe reaction behavior. A reaction behavior has to be specified in a way that the involved system is transferred back to a safe mode as optimally as possible. Also the start and the end of a regular optimized collaborative mode behavior has to be specified with respect to the dynamic context. In the vehicle platooning scenario, for instance, coupling and decoupling processes of vehicles in a platoon have to be considered.

While the fundamental approach for research challenges 2, 3 and 4 as the main contribution of this thesis are explained in Section 3.5, the research challenge 5 is explained specifically with regard to its implementation in Section 4.5, but is also already outlined throughout this Chapter. This is done since the resulting collaborative mode behavior represents a straightforward implementation based on the DSCs approach to show its feasibility. For research task 1, different solution concepts have been developed depending on the application context and are explained in Section 6.2.1.

3.5 A Modeling Approach for DSCs

If there are hazardous situations during collaboration of systems, maybe with a resulting physical or material loss, responsibility and liability issues have to be clarified. Hence, next to the derived research challenges, legal issues are an important aspect considering the vehicle domain as the target application domain. This becomes particularly visible for the described platooning of vehicles in public space, where potential collaboration partners are previously unknown to each other. Although they establish a collaboration group of systems, they still represent separate legal entities such as the various vehicles with different owners. But also for other scenarios like master-slave driving, where involved vehicles eventually belong to the same owner, the identification of error sources with respect to the responsible system manufacturer is a necessary analytical capability. Since the collaborative mode behavior, as shown in the platooning scenario in Fig. 3.1, is established by several parties with spatially distributed systems, the system interfaces are therefore particularly important for the integration of external services.

As it was outlined before, the DSCs approach aims at vehicle collaboration scenarios where an individual system behavior is optimized with collaborative safety knowledge. To ensure a safe collaboration of the various systems, it was outlined in Section 3.3 that an individual safety assurance approach for the involved participants is utilized since it has various benefits in this application context. This requires that safety for

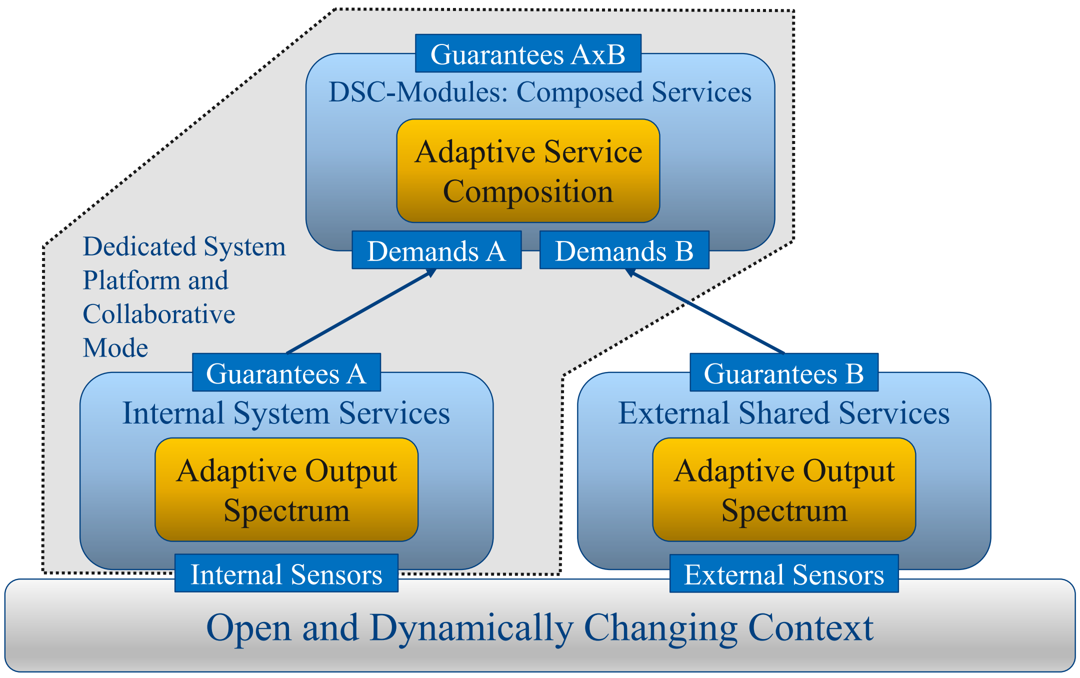


Figure 3.3: Service Composition with DSCs

the resulting optimized system behavior is guaranteed by each collaboration partner individually. Hence, the impact of a varying service availability during collaboration of systems on the individual system safety has to be clearly specified for a collaborative mode description. Ideally, this has to be done in a quasi-contractual description model to guarantee safe interaction processes and to trace back faulty signals to involved parties. A service integration concept becomes necessary for each system platform that enables the reliable integration of external data for safety assurance and certification means. To emphasize this commitment for the joint safety responsibility of various legal parties for a limited period of time, the solution approach is called "Dynamic Safety Contracts" (DSCs).

In Fig. 3.3 the fundamental approach of a service composition with DSCs is shown. It should illustrate that for the DSCs approach internal and external safety-related sensor data can be jointly considered in a formalized way for a single system safety specification. The lower gray bar *Open and Dynamically Changing Context* illustrates the application context in which a collaboration process of spatially distributed systems is performed. With respect to the presented platooning use case in Fig. 3.1, this could represent the public road traffic environment in which a vehicle platoon is driving. The module *Internal System Services*, shown on the lower left side in Fig. 3.3, is linked to the illustrated environmental context bar via the *Internal Sensors* and is associated to a dedicated system. The sensors, which are attached to a single system platform, generate internal safety-related runtime guarantees from monitoring the unknown environment. This is done by transforming the continuously monitored safety-related environmental data to formalized internal system guarantees based on the assigned *Adaptive Output Spectrum* specification. The illustrated adaptive output spectrum thus depicts an abstraction level of measured sensor data for further internal evaluation processes. It represents a typical step for measurement value processing. For a single non-networked vehicle, the possible spectrum of sensor values can be facilitated to develop an overall safety concept for autonomous driving based on the available formalized safety guarantees at runtime. This is done by assigning permitted degrees of freedom like a virtual safety cage to certain measured operating states.

Since the generated internal system guarantees are only valid for a short period of time and are part of the runtime safety assurance concept, they are referred to in this thesis as “Runtime Evidences” (RtEs). RtEs are generated by surrounding systems as well. The RtE generation from external sources is summarized in the *External Shared Services* module on the lower right side in Fig. 3.3. The external RtE generation takes place in parallel. By also introducing this external safety knowledge to a particular system, the individual system capabilities, as already outlined, can be optimized from safety as well as performance perspective. This, however, requires a sound integration concept of the distributed safety knowledge on a dedicated system platform for a collaborative mode specification.

Exactly this is done in the higher-level module in Fig. 3.3 shown as *DSC-Modules: Composed Services*. The higher-level module summarizes a holistic service composition concept for internal and external system guarantees. It represents a set of linked DSC-Modules as shown in Fig. 3.4. With a runtime evaluation of the illustrated linked DSC-Modules the allowed degrees of freedom as well as the resulting collaborative mode behavior can be determined for a specific system in a collaboration scenario.

Generally, the specification of DSC-Modules becomes possible since the collaborative mode is specified for a dedicated system platform and collaboration scenario. Hence, the available shared safety guarantees from the involved systems, as illustrated in Fig. 3.3, are completely known before operation. For the DSCs approach, internal services, as shown in Fig. 3.3, are considered by analyzing the predefined adaptive output spectrum of each source that generates internal RtE guarantees. Sources from external systems are treated in the same way as it is outlined in Fig. 3.3. Basically, the subordinate sensor concept as the origin of the generated service guarantees has no influence on the modeling approach, since the RtEs can be defined very specifically according to the concrete system configuration. The guarantee services could be generated either by an individual sensor or a holistic sensor fusion network. Further details about the specification of an adaptive output spectrum and its working principle will be provided later in this thesis, in Section 4.3.

In this way, the impact of the various different services with missing or degraded RtEs on a dedicated system safety can already be defined at development time. To ensure a safe operation in the diverse situations, which could occur during collaboration, the permitted degrees of freedom for a system can be specified with an adaptation concept

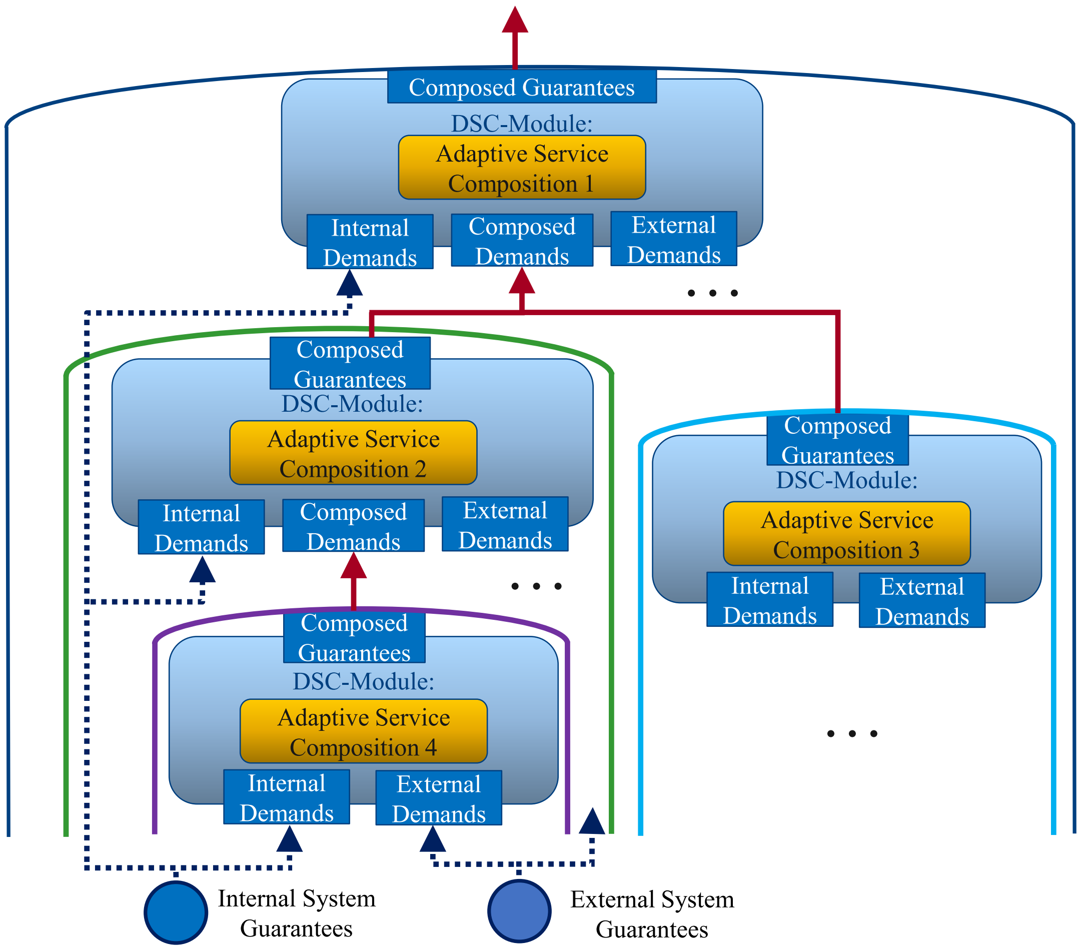


Figure 3.4: Hierarchical Arrangement of DSC-Modules

based on the available composed guarantees. For the DSCs approach this is done by specifying a virtual safety cage description based on contract-based service composition concept. In order to address the dynamic application context in the vehicle domain, a safe reaction behavior is part of the safety cage description. In the described platooning scenario in Fig. 3.1, for instance, the virtual safety cage is represented by the illustrated collaborative mode behavior space. At runtime, it is then checked based on the pre-defined service composition concept which services are available in order to derive the currently permitted scope of functionality.

In accordance to the derived key features in Section 3.4 it is possible to completely specify the service composition concept with DSC-Modules at development time. As it was outlined for the derivation of requirement 2, applying a modularized service composition concept simplifies necessary safety engineering activities. The safety engineers can thus focus on specific aspects of safety assurance. This enables a reduced adaptation space for safety assessment and thus more efficient modeling processes. Typically, several DSC-Modules are rigidly interconnected in a predefined way at development time as shown in Fig. 3.4. Based on the hierarchical composition model of DSC-Modules, the top-level system safety features for a collaboration process can be continuously evaluated at runtime.

Each DSC-Module typically focuses on a particular safety feature, which is relevant for the safe operation of a collaborative mode. To facilitate this, a DSC-Module specifies possible degradation steps for a certain safety feature based on a pre-configured variability of safety contracts. Due to the modularized approach for DSCs, the contract logic is implemented as a prioritized sequence of demand-guarantee relationships. At runtime it is then checked which safety contract configuration is active based on the available safety-related output guarantees. Hence, *Dynamic Safety Contracts* specify the correct merging of internal and shared (external) safety related runtime data for a dedicated system safety in a certain collaborative mode. An evaluated output of a safety contract module represents a valid runtime safety guarantee based on collaborative knowledge. It is part of the decision whether the current operational situation of a collaborative mode is safe or not.

From performance perspective ideally a fine-grained degradation concept is specified to ensure that slightly degradations of safety properties does not restrict the collaboration capabilities too much. To enable a service composition concept as complete as possible, the monitored safety features, specified for a collaborative mode, have to address different hierarchical integration levels of composed services. For example, there can be safety-related features on a lower level, such as a basic collaborative perception capability. In addition, there can be superordinate safety features, such as the abstract capability of being part of a vehicle platoon based on several underlying subdecisions. Consequently, the DSC-Modules are intended to be arranged in a hierarchical way as shown in Fig. 3.4. This requires that next to the integration of basic services, also higher-level composed services could be considered as input demands on the various hierarchical levels. Independently of the hierarchical level, the illustrated DSC-Modules can be specified individually. Naturally, the evaluation concept with interlinked DSC-Modules requires a suitable granularity of service types. The granularity level of the virtual safety cage description, for instance, is limited by the available top-level guarantee services. This is the case, since the addressed virtual safety cage representation is evaluated based on the top-level DSC safety features. It specifies a safe collaborative mode behavior considering all possible degradation scenarios of composed services in the dynamic context.

In Fig. 3.5 a virtual safety cage specification based on the DSC service composition is shown. In a typical networking situation the *Optimized Collaborative Mode Behavior* can be derived. In this mode, based on the DSC service composition, the networked application can be continuously adapted to minor changes in service availability. Thus, the system behavior can adapt itself in a smooth way to certain changes in the dynamic collaboration context before a hazardous or a warning state situation occurs. In this context, the warning state is considered to be a still safe situation in which a future hazardous situation can be predicted, although it has not yet occurred at that point in

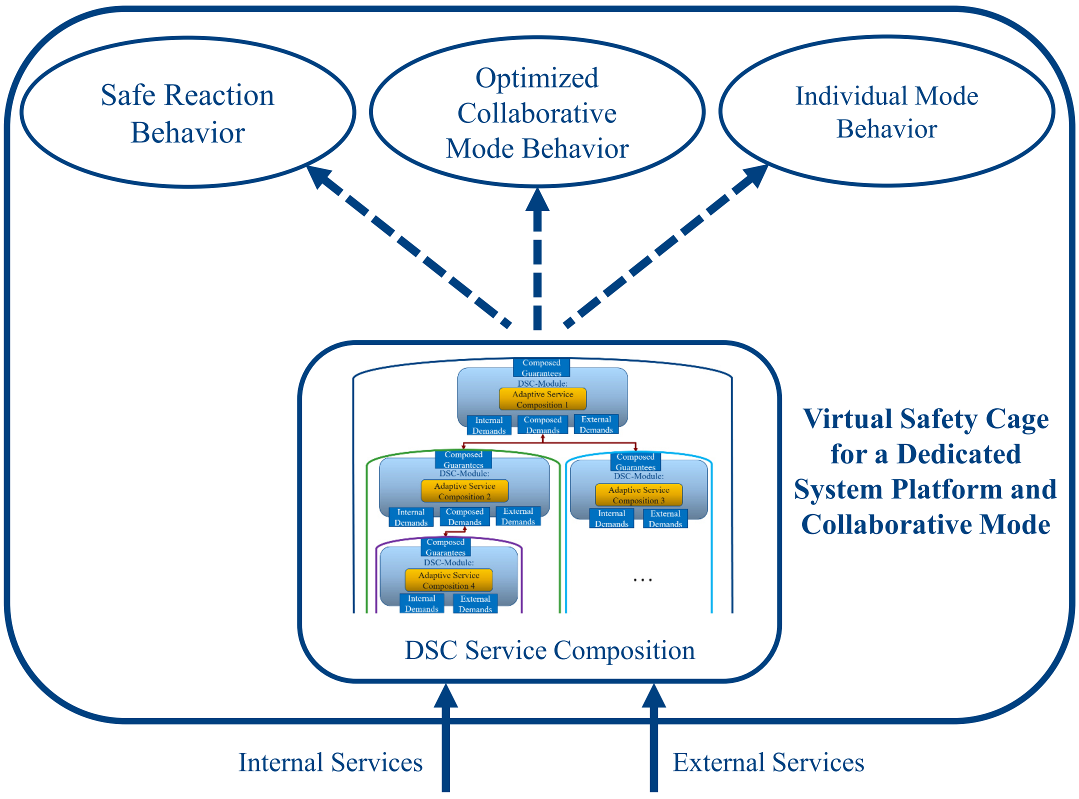

Safe Reaction Behavior
Optimized Collaborative Mode Behavior
Individual Mode Behavior
Composed Guarantees
DSC-Module:
Adaptive Service Composition 1
Internal Demands
Composed Demands
External Demands
Adaptive Service Composition 2
Adaptive Service Composition 3
Adaptive Service Composition 4
...
DSC Service Composition
Virtual Safety Cage for a Dedicated System Platform and Collaborative Mode
Internal Services
External Services

Figure 3.5: Virtual Safety Cage Specification for a Collaborative Mode

time. The hazardous situation describes a situation in which the system itself can no longer avoid an accident, but can only try to leave this state. Basically, the described normal behavior should be implemented in a smooth way, since an abrupt behavior change, as already outlined, could itself lead to a hazardous situation. In the automotive domain, for example, this can be the risk of collision with the following vehicle due to unexpectedly strong braking maneuvers.

If the system is already in a warning state, a *Safe Reaction Behavior* has to be triggered based on the situation assessment of the composed services in order to avoid a hazardous state. However, it is also possible that the networked system is already accidentally in a hazardous state. While the regular networked behavior (*Optimized Collaborative Mode Behavior*) is defined for a specific, rather ideal system state, the safe reaction behavior has to be defined for various warning and hazardous states.

Fig. 3.5 also outlines the *Individual Mode Behavior* which specifies the dissolution of the collaborative mode behavior to the non-networked individual behavior in the dynamic context. This is done by introducing a dedicated service guarantee that indicates the agreement of others to participate in a collaboration mode. If there is no agreement to be part of a collaboration network, the collaboration mode behavior is resolved smoothly. It can thus be avoided that suddenly missing external services in a collaboration situation are interpreted as a hazardous situation.

With respect to the modeling process of a networked DSC mode based on linked DSC-Modules, Chapter 5 presents a systematic engineering strategy for this approach. Similar to conventional safety engineering approaches, most safety assurance and certification activities can already be conducted at development time to limit the runtime parts to a minimum. Based on a hazard analysis of the collaboration situations that may occur in the intended collaboration scenarios, the potentially hazardous situations have to be identified. They form the basis for deriving the safe operating conditions which are monitored at runtime based on a set of linked DSC-Modules.

Generally, for the specification of DSC-based collaboration scenarios, it has to be clarified whether openness for previously unknown collaboration partners is required or not. For the master-slave driving domain this is typically not the case. In other scenarios like the presented platooning scenario in public road traffic this is a necessary requirement. In order to make this possible, the following Section presents a solution concept for identifying optimal DSC-based networking modes for new emerging collaboration partners.

3.6 Establishing Collaboration Processes with DSCs

For collaboration scenarios such as the already introduced use case for master-slave driving, a suitable collaborative mode description can already be defined at development time due to the predefined collaboration partners. This is possible since the involved systems as well as the intended application context are completely known at development time. On this basis, a concept for the available system services as well as the dedicated application context can be completely evolved during the development phase of a collaborative mode for an optimal trade off between safety and performance. Hence, the question whether other collaborative modes are more suitable with respect to a certain application context doesn´t has to be clarified at runtime. Furthermore, the restricted number of collaboration partner in such a collaboration scenario helps to reduce the risk of security threats such as hacker attacks on vulnerable systems.

In contrast to the aforementioned, however, in collaboration scenarios such as the public road traffic, openness for collaboration networks with previously unknown collaboration partner can be considered as as an essential feature. Nevertheless, for systems, which should generally be open for collaborations processes with other systems, various drawbacks can arise. For example, unified collaboration capabilities with an imprecise application task can probably lead to inflexible and inefficient development processes for safety assurance concepts. The various manufacturers from all possibly involved systems would have to be considered for such a unified collaboration approach. Furthermore, by considering that the applied safety engineering strategies of manufacturers are sometimes not compatible, this could even be impossible.

Another possibility to enable openness would be that a certain group of potential col-

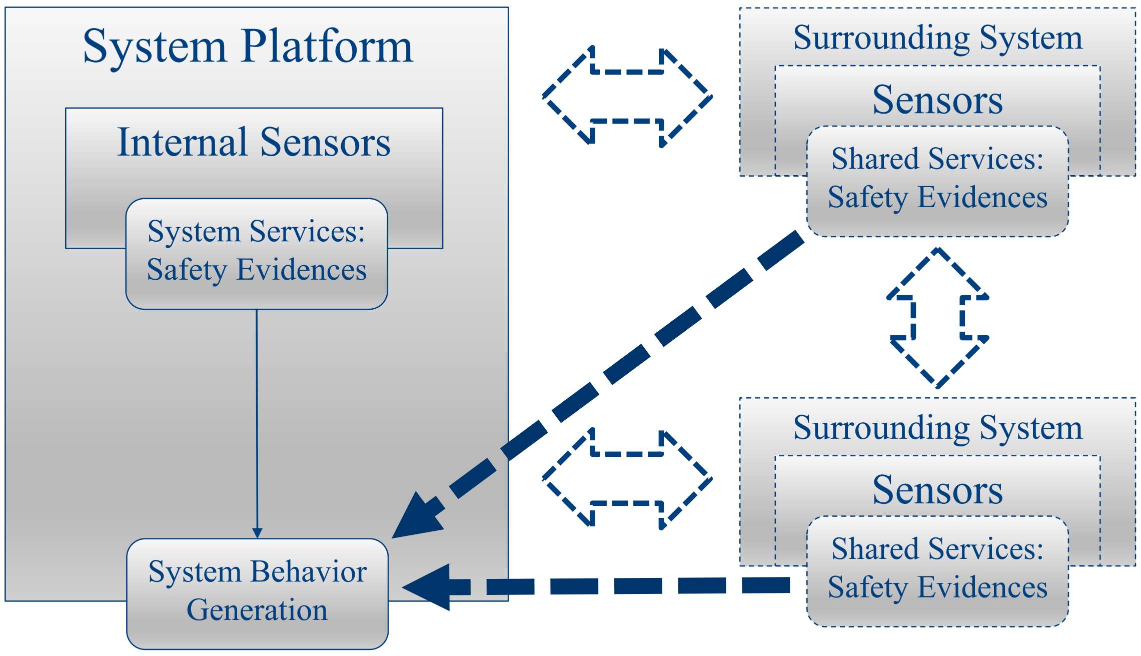

System Platform
Internal Sensors
System Services: Safety Evidences
System Behavior Generation
Surrounding System
Sensors
Shared Services: Safety Evidences
Surrounding System
Sensors
Shared Services: Safety Evidences

Figure 3.6: Considering External Services to Optimize Individual Behavior

laboration partners negotiates a suitable collaborative mode at runtime. This can be facilitated based on harmonized description of individual system capabilities. A key advantage is that collaborative modes can be specified with respect to individual system capabilities. In this way, the addressed adaptation space for required safety engineering activities can be reduced to a manageable size. In addition, by assessing the individual system capabilities at runtime, establishing collaboration processes could become more flexible, since slightly modified system functions could possibly still be considered for degraded collaboration modes.

In the following, the fundamental procedure of establishing a collaboration of systems for the DSCs approach is explained in an illustrative way. The system platform in Fig. 3.6 on the left side represents a system, which tries to collaborate with an arbitrary number of surrounding systems. In the presented platooning scenario, the system platform would represent the following vehicle while the surrounding systems would represent the front vehicle and the communication infrastructure. The overall objective of this collaboration group is to make shared safety-related sensor data available for the system behavior generation of the following vehicle. This means, as already outlined for the DSCs approach in Section 3.3, that the considered system platform only collects data from surrounding systems. It does not execute tightly integrated functionalities, in the sense of mutual influences on driving behavior. No system actively influences other systems like a remote controller or has the role of a system integrator. As already outlined, in such a case the resulting system complexity for safety assurance as well as liability issues regarding responsibility for physical or material damage would be much more difficult to assess.

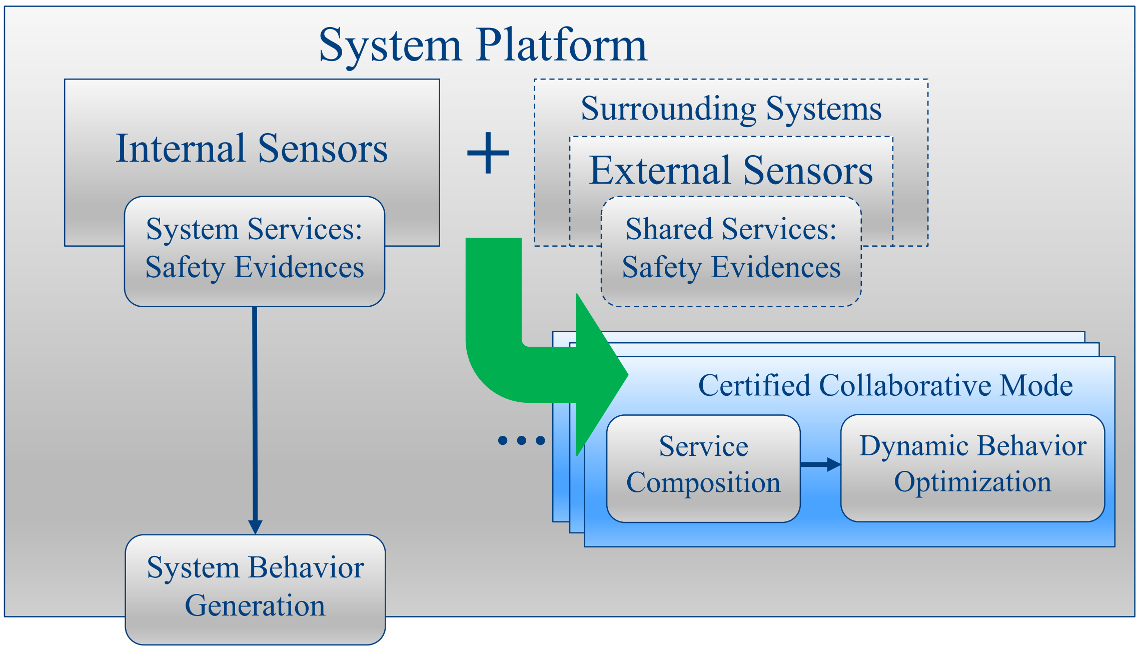

System Platform
Internal Sensors
System Services: Safety Evidences
+
Surrounding Systems
External Sensors
Shared Services: Safety Evidences
Certified Collaborative Mode
Service Composition
Dynamic Behavior Optimization
...
System Behavior Generation

Figure 3.7: Safe Determination of a Collaborative Mode

For establishing a DSCs collaboration process a two-step approach is facilitated. The first step determines an optimal collaborative mode in a reliable way before operation as shown in Fig. 3.7. In this thesis this is considered as “integration time”. The second step ensures safety during operation of a collaborative mode as shown in Fig. 3.8. This is referred to as “runtime” in this thesis. While the first step is not time-critical, the second step has to be executed close to real-time. This holds true since the dynamic application context of the vehicle domain requires a timely determination of an adequate system behavior based on the DSCs virtual safety cage description. If the collaboration situation during operation of the collaborative mode is evaluated to be unsafe, a suitable reaction behavior has to be initiated immediately.

The first step for initializing a collaboration process is to identify an optimal collaborative mode for the integration of external data from currently available collaboration partner. Naturally, this has to be done in accordance to the safety regulations in a safety-critical domain. In Fig. 3.7 a set of potential collaborative modes is shown as *Certified Collaborative Mode*. For the safe determination of an optimal collaboration mode (Fig. 3.7) a negotiation process is applied. For this purpose, the system capabilities such as safety classifications of sensors are introduced to the negotiation process as static hardware guarantees. The implementation of a concrete negotiation process is provided in Section 6.2.2.

When a certified collaboration mode is identified, the second step of ongoing collaboration is initialized (Fig. 3.8). This is the active integration of external sensor data, which is continuously forwarded from surrounding systems according to the demands of

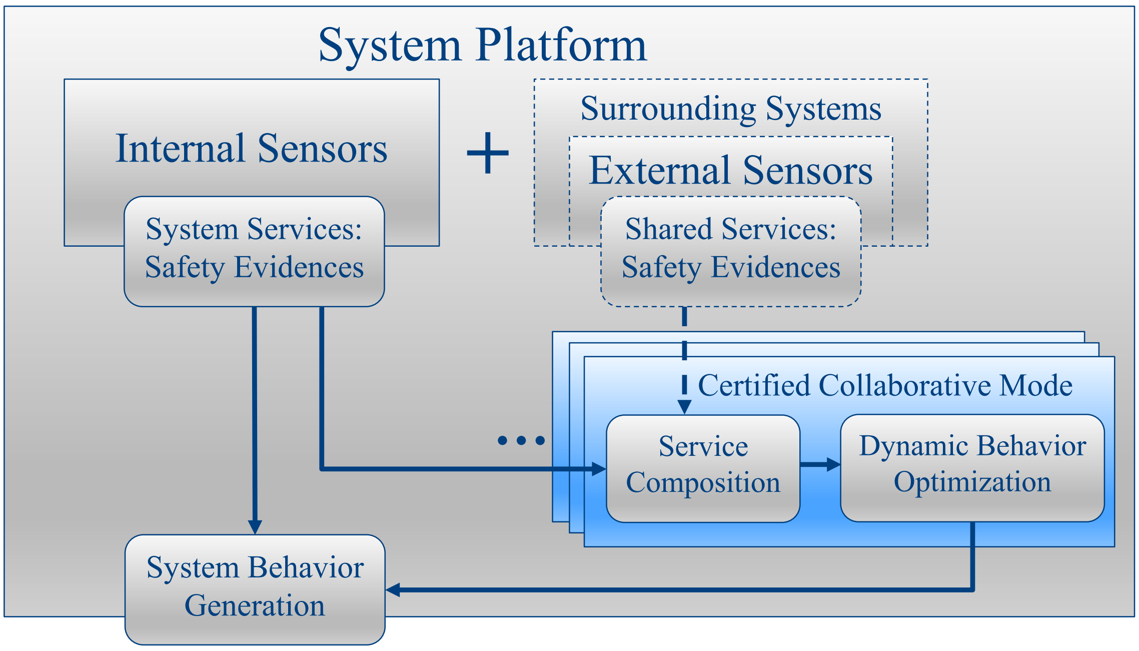

System Platform
Internal Sensors
+
Surrounding Systems
External Sensors
System Services: Safety Evidences
Shared Services: Safety Evidences
Certified Collaborative Mode
...
Service Composition
Dynamic Behavior Optimization
System Behavior Generation

Figure 3.8: Safe Composition of Services to Optimize Individual Behavior

the collaboration mode. Internal and external services are continuously composed based on the DSC evaluation and the individual system behavior is optimized accordingly. A detailed description of the process of identifying a suitable collaborative mode, its initialization and the on-going operation is provided in Section 6.2.

3.7 Operationalization of DSCs

To close the gap between a theoretical approach to practical use, the different facets of operationalization of the DSCs approach will be outlined in the following. It should be noted, however, that the selected operationalization approach strongly depends on design decisions and should not be understood as mandatory. Also other modeling approaches are basically conceivable, but they have to be in accordance to the identified requirements in Section 3.4.

For the introduction of formalized description models for networked applications, a suitable definition of potential service types is needed that can be applied for a modular demand-guarantee representations. In Section 4.2 and 4.3 the concept of qualitative and quantitative RtE services is provided. However, suitability of service types strongly depends on the respective application context.

For system collaboration processes in a dynamic environment such as the provided vehicle platooning scenario, for instance, it became apparent that especially quantitative parameters like speed and distance are relevant for safety assurance. To take this into account, also DSC-Modules with quantitative guarantees were introduced, next to modules with qualitative guarantees, as described in Section 4.4. For the mapping of fulfilled demand ports to guarantee ports of DSC-Modules, a merging logic is provided in Section 4.4.2. Depending on the implementation scenario, it may be useful to introduce additional merging elements. To enable a safe and efficient modeling of DSC-Modules, a dedicated MagicDraw Plug-In was developed as described in Section 4.4.4. A suitable linking concept for hierarchical evaluation processes of DSCs, as shown in Fig. 3.4, is described in Section 4.4.5.

In order to evaluate available RtEs for a networked DSC mode at runtime, DSCs have to be available in a computable representation. The evaluation of DSC-Modules has to be performed in a fast and reliable manner to achieve an optimal reaction behavior to changing safety features. To make this possible, the MagicDraw Plug-In also provides automated code generation from the graphical DSC-Module representation. Moreover, Chapter 6 compares different evaluation approaches for their runtime feasibility.

For highly dynamic collaboration processes such as the platooning scenario, a suitable approach for dynamic behavior adaptation is needed. For a networked system in a DSC mode, an optimal transition to a safe state has to be ensured in case of degraded safety features. For the platooning use case, a dedicated DSC-Module was introduced to specify an adequate dynamic adaptation behavior as explained in Chapter 4.5. The feasibility of the proposed overall adaption concept as well as the concrete dynamic adaptation behavior was validated with a comprehensive simulation environment as described in Chapter 6.2.3 as a proof of concept.

Furthermore, many potential DSC application scenarios require openness to previously unknown systems for efficient collaboration processes. In other scenarios, however, collaboration partners can already be predefined at development time such as the presented use case for master-slave platooning. The implementation of negotiation processes for identifying the best possible networked DSC applications is described in Section 6.2.2.

In addition, further aspects such as security concerns or legal aspects are relevant for practical use. These issues are specifically addressed in Section 7.2 as part of the future research agenda for practical implementation.

3.8 Safety Engineering Backbone for Contract-based Safety Certification

Since today's state-of-the-art safety assurance and certification techniques doesn't consider runtime evaluations, new safety engineering strategies have to be established. Thereby the derived safety engineering activities should be as close as possible to the state-of-the-practice methodologies to built upon existing experience of safety engineers. For the presented DSC approach, a Safety Domain Model (SDM) was derived which subdivides the safety engineering activities into domain-level and system-level engineering. This seems particularly useful due to the many possible, only slightly different networked DSC applications in a certain domain. The corresponding reuse strategy of parts of existing DSC collaboration modes is explained in more detail in Chapter 5.

In Section 5.1, the overall engineering methodology on the domain-level is explained. Here, the considered collaboration scenarios of systems have to be analyzed for potential hazardous situations that may occur during operation. The necessary operating conditions to ensure safe collaboration processes must then be derived from these findings. In Section 5.2, the system-level engineering for the realization of concrete networked DSC applications is summarized. Based on the derived safe collaboration conditions from the domain-level, dedicated RtEs and DSC-Modules can be specified for the continuous monitoring of the relevant safety features. The ultimate goal on the system-level is the safety certification of the networked DSC mode by the certificate issuer on the target platform. As part of a holistic safety concept, it should thus become possible to certify networked DSC applications using development time and runtime means. In Section 5.3 it is also shown in which way parts of already specified collaboration processes can be systematically reused.

Before a certain networked mode is tested in practice, it is possible to systematically optimize the entire safety concept based on findings from simulations. In particular, the SDM model supports iterative optimization on the basis of the experience gained. To demonstrate that a particular DSC mode meets the necessary safety and performance criteria, simulations can be performed based on clearly defined test cases. The simulation activities as part of the validation for the networked DSC modes are described in Section 6.2.3.

4 Specification and Implementation of DSCs

This Chapter deals with the operationalization of the DSCs solution approach based on the underlying concepts as described in Chapter 3.5. For this purpose, the various constituents of the DSCs approach are explained based on a dedicated DSC engineering framework in combination with mathematical representations. In order to provide a comprehensive insight into this approach, an use case for vehicle platooning is derived in parallel to the various specification steps. Starting from the description of the application scenario, a suitable service provision concept for the DSC-based networking of systems is identified. In the following, for monitoring relevant safety features in networked DSC applications, an evaluation logic with so-called DSC-Modules is introduced. Moreover, a straightforward adaptation behavior is defined on the basis of the scenario specification. Together with Chapters 3 and 5, this Chapter therefore forms the main contribution for the operationalization of the DSCs approach. Chapter 3 focuses on the fundamentals of establishing networked DSC applications. Chapter 5 presents a suitable DSC engineering strategy for the domain-level as well as for the system-level to support a systematic development of DSC networking modes. In addition, Chapter 5 explains the DSC concept for reusing parts of already specified networking modes. Basically, many of the specific DSC characteristics described here have already been presented in various publications [80] [81] [82] [83] [84].

4.1 Running Example: Platooning

As already introduced in the previous Chapters, the focus for the DSCs approach in this thesis is on the automotive domain, but is not limited to it. In this context, the presented "Platooning" use case aims at networked vehicle fleets which are driving in a so-called platoon. In a networked platooning driving mode, the driving behavior of vehicles can be optimized with respect to performance parameters such as speed and driving distance. The use of the DSCs solution approach in such scenarios should enable safety assurance and certification for the modified driving behavior of the individual vehicles involved.

Such platooning applications can be used not only in typical road traffic environments, but also in the off-road domain with specific vehicle fleet configurations such as "Master-

Slave" driving. For the DSCs approach, both scenario variants were considered. In the considered off-road scenario it is planned to enable unmanned slave vehicles to support the master vehicle. For this purpose, all possible driving maneuvers of the slave vehicle must be specified in such a way that they are dependent on the authorization of the master vehicle.

In order to ensure safety for the platooning driving mode, safety-relevant data has to be shared between the involved vehicles using wireless communication. To access the current operational situation of a following vehicle in a networked platoon, the internal operating status of the following vehicle as well as the external status of the front vehicles and maybe other external sources has to be merged in a reliable way. As will be explained in more detail in the following Chapter, this especially requires an adequate runtime representation of the networked safety features as well as a suitable runtime evaluation mechanism. For this purpose, this Chapter especially addresses the necessary service provision concept for establishing networked driving modes based on decentralized safety knowledge.

The basic operating principle for deriving an adapted driving behavior in a networked driving mode can be looked up in Section 3.2. The derivation of particular safety goals and the overall safety concept for a collaboration scenario is described in Chapter 5.2. The concrete implementation as well as the conducted simulation activities for the platooning use case can be looked up in Chapter 6. In principle, the addressed vehicles can be equipped with different types of sensors and sensor fusion concepts as long as these are suitable for achieving the defined safety goals.

4.2 Generation of Safety Evidences at Runtime

In order to reliably integrate shared safety-relevant data on certain system platforms as an additional assurance means for safety, adequate service provision concepts need to be defined. In particular, the contained safety information has to be formalized in such a way that it can be interpreted on other system platforms in an unambiguous and lightweight way. In order to achieve this, the concept of "Runtime Evidences" (RtEs) is utilized for the DSCs approach. A RtE represents an abstract safety evidence for a specific dynamically variable and safety-relevant aspect of a collaboration scenario. Depending on the alteration speed of the monitored characteristics, RtEs are only valid for a short period of time.

RtEs represent a rather generic concept and are not limited to specific types of safety-related aspects that are to be monitored at runtime. Consequently, the checking criteria for RtEs can be specified in manifold ways at development time. Safety classifications schemes such as the Safety Integrity Level (SIL) [85], the Automotive Safety Integrity Level (ASIL) [40], the Software Safety Classes for medical devices [86] and the Design Assurance Level (DAL) [39] can provide guidance for the implementation of adequate

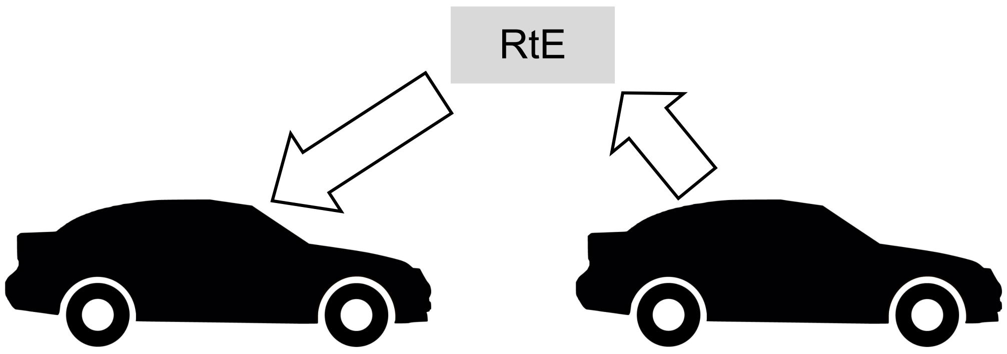


Figure 4.1: Generation and Forwarding of a Single RtE

hardware/software solutions. In the next step, the reliability of the RtE generation must be thoroughly tested. Here, the focus should be on the reliability of the RtE generation in the concrete application scenario based on the scenario context and the platform-related implementation approach.

Naturally, the focus for networking autonomous vehicles is on the recognition of complex environmental issues. This can be, for example, the reliable detection of surrounding vehicles in a challenging application context. If the requirements for the generation of a single RtE are fulfilled at runtime, as illustrated in Figure 4.1, a RtE can be forwarded to other networked system platforms as part of a set of requested RtEs. The proposed solution approach based on DSCs focuses especially on the reliable integration of the forwarded RtEs on other system platforms. Based on the available additional safety knowledge on the networked system platforms, an adapted driving mode with an adapted safety concept can be initialized.

Generally, a suitable standardization of RtEs and their checking criteria in the respective domain is important. The contained safety knowledge in RtEs determines the overall scope of possible networked applications. On the one hand, no raw measurement data should be forwarded, as these are difficult to interpret by other networked systems. On the other hand, the shared data should not be too abstract to allow a fundamental analytical fault detection in a networked mode. Hence, an appropriate abstraction level of safety-related knowledge is crucial for the efficient sharing of RtEs.

When specifying networked driving modes such as the platooning scenario presented below, it became apparent that it is sensible to introduce two different RtE service types. Ideally, RtEs that are closely related to each other can be grouped and mapped to a particular scale type. By using standardized scale types, the merging logic can also be harmonized later on. To enable this, a qualitative and a quantitative RtE representation for safety-related data was utilized. Whereas qualitative RtEs are basically defined as Boolean data types, different numerical data types can be used to define the value ranges for quantitative RtEs. While each qualitative RtE describes compliance with the requirements of one particular safety-relevant state, a quantitative RtE can cover several different states. To achieve this, a set of qualitative RtEs is mapped to a dedicated scale, as illustrated in Figure 4.2. In this way, sender and receiver platform can exchange a number of related RtEs without explicitly specifying several different qualitative RtE service types for a networked mode. In addition, as will be explained later, it becomes possible to directly address a whole set of RtEs with a single merging logic.

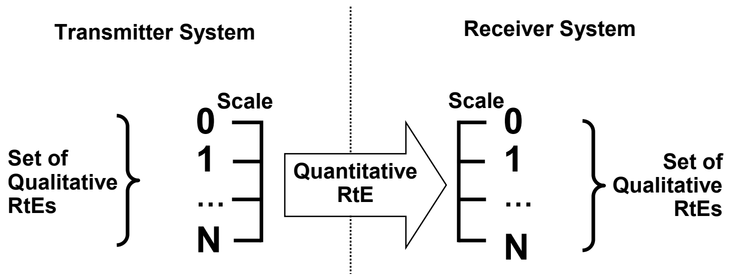


Figure 4.2: Transformation Scale for Quantitative RtEs

In a typical engineering process of a networked DSC application, as described in Chapter 5, the focus for specifying RtEs is on the concrete system platforms to be networked and the identified safe operating conditions for the targeted collaboration scenario. In the next step, appropriate RtEs have to be introduced on the basis of these identified safety features. Subsequently, still at development time, a conditional behavior adaptation concept can be defined for the networked system platforms on the basis of the RtE specification and its validity at runtime.

Running Example: Determining an Appropriate Service Concept

To illustrate feasible RtE services and their usage, the RtEs used for the platooning scenario are summarized in Table 4.1. The considered RtE services for communication between vehicles and infrastructure are typically generated from on-board sensors of the participating systems. They monitor internal system conditions or environmental aspects. Ideally, the RtE services can be specified in such a way that the system hardware doesn't need to be modified or upgraded. The procedure for deriving suitable RtEs for a networked DSC application is explained in Chapter 5.

As explained before, a fundamental distinction was made between qualitative and quantitative RtEs. For the platooning application, for example, the reliable mutual recognition was identified as an important precondition for ensuring a safe driving distance. This could be enabled by using the available rear sensors of the front vehicle and the front sensors of the following vehicle. Adequate RtE services were needed for this purpose. Since for the reliable detection only an unambiguous safe detection is permitted, no further intermediate states are required. The reliable detection can thus be sufficiently represented by a single qualitative RtE. In other cases, such as the current speed limit, it seems sensible to introduce a quantitative RtE with which different states can be specified. Later on, these various considered states can be used for comparative runtime checks. The meaning of the illustrated RtE services and their concrete usage in the platooning mode is explained in more detail in Chapter 4.4.4.

RtE Service	**Data Type**	**Description**	**Providers**
Platooning OK	Qualitative	Front vehicle has detected no threats for platooning.	Front Vehicle
Traction OK	Qualitative	Front vehicle has no traction difficulties.	Front Vehicle
Safe Detection of V1	Qualitative	Front vehicle can reliably detect the following vehicle with onboard sensors.	Front Vehicle
Distance to V1	Quantitative	Distance measurement of the front vehicle to the following vehicle.	Front Vehicle
Valid Speed Limit	Qualitative	Front vehicle or infrastructure confirms the provision of a valid speed limit.	Front Vehicle / Infrastructure
Speed Limit	Quantitative	Front vehicle or infrastructure provides a speed limit value.	Front Vehicle / Infrastructure
Vehicle 1 OK	Qualitative	Front vehicle accepts the networked driving mode.	Front Vehicle
Speed	Quantitative	Internal speed measurement of the following vehicle.	Following Vehicle
Safe Detection of V2	Qualitative	Following vehicle can reliably detect the front vehicle with onboard sensors.	Following Vehicle
Distance to V2	Quantitative	Distance measurement of the following vehicle to the front vehicle.	Following Vehicle
Vehicle 2 OK	Qualitative	Following vehicle accepts the networked driving mode.	Following Vehicle

Table 4.1: Specification of RtE Service Types

4.3 Dynamic Safety Guarantees, Demands and Contracts

The dynamic integration of external safety-related data requires a concept in which the observed characteristics can be formalized and assigned to each other. To identify the relevant safety features of a networked mode that need to be assessed during operation, a comprehensive hazard analysis has to be performed for the planned collaboration scenario, as described in Chapter 5. Based on the identified safety features, demand-guarantee relationships of internal and external RtEs can be specified using a dedicated DSC contract logic. Fig. 4.3 illustrates the relationship of shared RtE services in a networked DSC mode with respect to the specified safety features using demand-guarantee dependencies.

By merging distributed RtEs, higher-level networked safety guarantees can be derived for the networked DSC applications. The fundamental idea of DSC service composition is illustrated in Fig. 3.3 in Section 3.5. Before operation, an optimal networked DSC mode can be identified based on the general availability of certain RtE services. During operation, depending on the availability of specific internal and external RtEs on the respective system platform, safety can be guaranteed for the networked adaptation behavior.

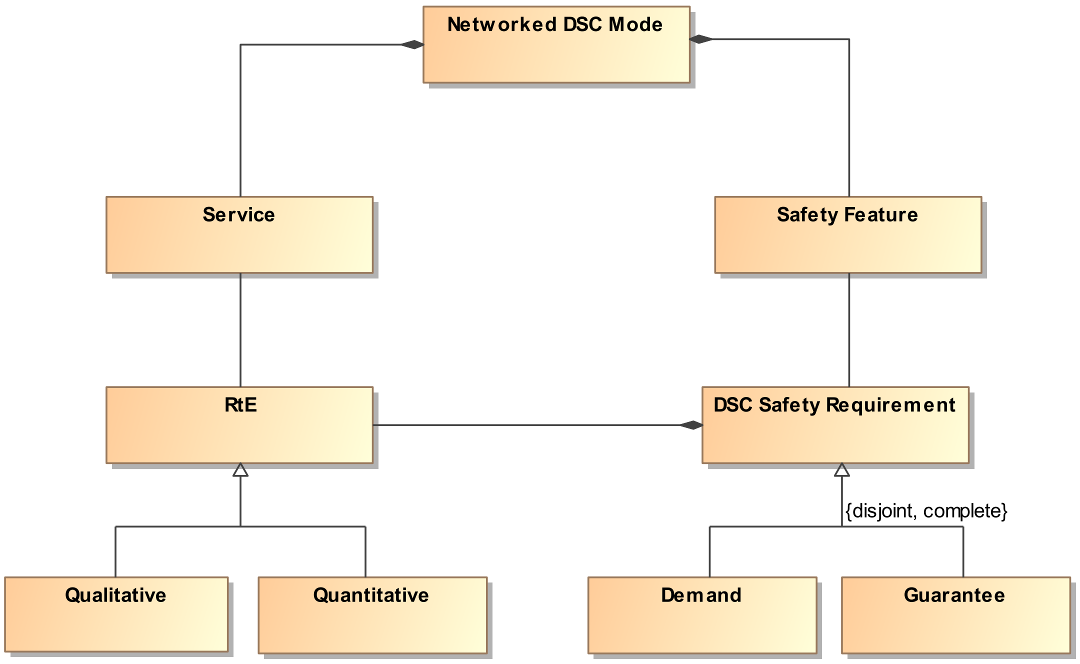


Figure 4.3: Safety Modeling of a Networked DSC Mode

The possibility of specifying a suitable degradation architecture for an appropriate networked adaptation behavior depends strongly on the granularity of the defined RtE service concept. To enable this, the contract logic as well as the overall RtE composition concept is ideally structured in such a way that an adequate networked system behavior can be systematically derived from it. Moreover, it becomes apparent that a RtE always refers to a specific application-related safety feature. Consequently, it appears reasonable to merge them according to the networked safety features. To achieve this, a modular integration approach with so-called DSC-Modules was utilized, as explained in Section 4.4.

4.4 Definition of DSC-Modules

A modular approach is of great advantage for efficient development processes, as it enables a traceable subdivision into affected features of a networked mode and provides a reusable description of the existing system interdependencies. Each DSC-Module represents a certain safety-relevant operating condition which is identified from a HARA for a particular networked collaboration scenario. Further details regarding the derivation of DSC safety features as the basis for specifying DSC-Modules can be found in Chapter 5. In order to associate these modularised safety features with each other, a consistent safety property representation as well as a formal interface description has to be specified.

4.4.1 Qualitative and quantitative degradation architectures

Taking into account constantly changing collaboration situations in a networked mode, a contract-based safety modeling approach requires a suitable, fine-grained degradation architecture for the identified safety features. As it was outlined in Section 4.2, RtEs are available either in a qualitative or quantitative form. To prevent a loss of information, it therefore seems reasonable to specify a uniform approach for contract modeling that takes both types of RtEs into account. Thus, in addition to a qualitative RtE safety contract modeling, an extended contract logic for the quantified description of safety-relevant data is introduced. Moreover, due to the modular approach, the evaluated output guarantees of safety contracts should be compatible with all other available qualitative and quantitative RtEs as input for higher-level evaluations. Consequently, also two slightly different types of DSC-Modules are introduced with the distinction of either qualitative or quantitative output guarantees.

4.4.2 Definition of DSC-Modules based on mapping functions

The qualitative and quantitative RtEs are described first since they represent the basic input guarantees for DSC-Modules. Generally, a valid RtE can be described with the following elements:

$$RtE = (RtE\_TypeId, DataType, Value)$$

The first element represents the identifier for the associated service type. It should be defined based on the type system in the respective application domain. The second element defines the RtE data type. As already mentioned, qualitative RtEs are basically defined as Boolean data types, while different numeric data types can be used to define the value ranges for quantitative RtEs. The last element specifies the current value based on the data type. In principle, it is possible to convert qualitative and quantitative RTE into their respective other form using the following equation:

$$quant.RtE(Scale(0,...,N)) = \begin{cases} qual.RtE_0 & \Big(\Sigma SR_0 = true\Big) \\ ... & \\ qual.RtE_N & \Big(\Sigma SR_N = true\Big) \end{cases}$$

The criteria for fulfilling a particular safety requirement (SR) of a DSC-Module can be summarized with the following equation. Here the "." is used as an operator to refer to elements of a set. For example, the *g.Value* denotes *Value* $\in$ *g*.

$$sat\_SR(g,d) = \begin{cases} \text{true if} & \begin{pmatrix} (g.RtE\_TypeId = d.RtE\_TypeId) \\ \wedge satDT(g.DT, d.DT) \\ \wedge satValue(g.Value, d.Value) \end{pmatrix} \\ \text{false} & \text{otherwise} \end{cases}$$

Generally, DSC-Modules are specified based on RtE safety demands that are mapped to associated RtE safety guarantees. This is done based on configurations as outlined in the following equation:

$$Conf_{DSC} = (RtE^{Prov}_{Conf}, RtE^{Req}_{Conf}, \Lambda_{Conf})$$

In a DSC-Module, a monitored safety feature is specified based on at least one provided RtE^{Pro}_{Confx}. Each RtE^{Pro}_{Confx} is associated to at least one required RtE^{Req}_{Confx}. Thus, when degraded states are considered for a monitored safety feature, a set of configurations arises in a particular DSC-Module.

4.4.3 Graphical modeling elements for DSC-Modules

The dependencies in a DSC-Module are mapped by directed acyclic graphs of *Boolean* functions as well as *If* and *Transformation* functions. The leaves represent the requirements of the associated mapping functions as a number of input ports. They specify the necessary RtEs which have to be available at runtime. In the flow direction, the root of the safety tree represents the output port, which determines potential higher-value RTEs.

A DSC-Module can be defined as:

- A set of input variables representing demands on particular qualitative or quantitative RtE types
- A set of Boolean gates, If gates and Transf. Fct. gates
- A number of directed edges connecting the elements

- A set of output variables representing guarantees on particular qualitative or quantitative RtE types

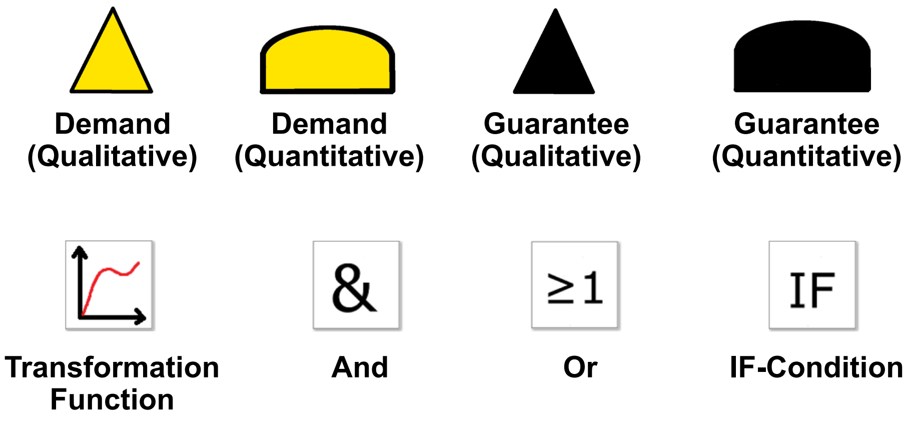


Figure 4.4: Graphical Elements for DSC Merging Logic

Propagation and fusion of boolean and numerical values

In the considered use cases, the four fusion gates illustrated in Fig. 4.4 were applied. In the following the respective operations for boolean and numerical operators are summarized. As described in the previous Sections, the numerical operators typically refer to a particular scale.

The *And* and *Or* gates only consider boolean values as in- and output parameters.

$$AND - Gate = (y = (x_1 * x_2 * ... * x_n))$$

$$OR - Gate = (y = (x_1 + x_2 + ... + x_n))$$

The *IF* gate considers two numerical values as input parameters and a boolean value as a output parameter. In addition to the operator *greater than* used in the equation represented by >*, the following operators can also applied: <, =, ≥, ≤, ==, !=

$$If - Gate = (y = (x_1 >^* x_2))$$

The *Transf. Function* gate considers numerical values as in- and output parameters. The following operators can be applied: -, +, *, /, ^

$$Transf.Function - Gate = (y = Function(x))$$

4.4.4 Graphical specification of DSC-Modules

This Section deals with the graphical specification of DSC-Modules. The engineering of DSC-Modules is done with a dedicated GUI, which is presented at the end of this Section. In the following, first the qualitative and then the quantitative DSC-Module specification is introduced.

Basic architecture for qualitative DSC-Modules

A qualitative DSC-Module for monitoring a certain safety feature consists at least out of two qualitative outgoing ports as shown in Fig. 4.5 and an undefined number of qualitative as well as quantitative demand input ports. In this way, a basic runtime decision taking for the monitored safety feature based on at least two different states becomes possible. These outgoing ports contain a state where no restrictions can be identified based on the specified input demands as well as at least one degraded state. As a modeling convention for DSC-Modules, these monitored states are arranged from left to right according to the evaluation order and the decreasing output guarantees.

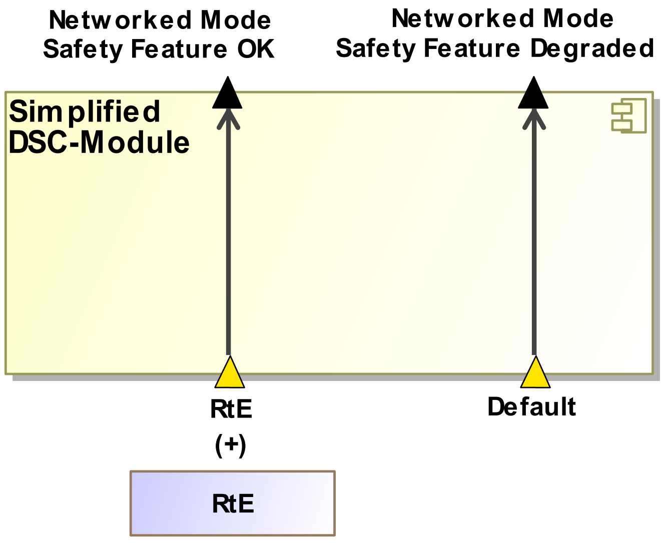


Figure 4.5: Simplified Qualitative DSC-Module

Basically, for the most degraded state no input demand needs to be specified as shown in Fig. 4.5. It is then only a proof of the successful evaluation of the respective module. This is due to the evaluation sequence of the output guarantees from left to right, where if no higher level output guarantee is valid, automatically the remaining state is valid. For the presented DSC-Modules, the design decision was made to explicitly model the empty input demands. Thus it can always be ensured that one state of a DSC-Module is valid. Moreover, by explicitly modeling the empty input demands, it can be recognized from the external module perspective that no input guarantee is required to activate the most degraded output guarantee.

In the illustrated DSC-Module in Fig. 4.5, two qualitative input ports are specified, which are directly connected to the output ports. For the qualitative input ports it is checked whether the specified RtE demands are fulfilled based on a binary decision taking. The evaluation result is then forwarded to the assigned output port.

Nevertheless, the applicability of such a module type without internal merging logic, as shown in Fig. 4.5, is very limited. Hence, it should be considered rather as a simplified special case for explanation. Generally, the specification of modularized DSC safety monitors aims at supervising networked safety features that cannot be assessed by any participating system alone. Typically, if a single involved system would have all the required input data, the available RtE could be linked to a higher level DSC-Module with a customized demand port specification. Alternatively, a modified RtE definition could be introduced to directly address a higher level networked feature. However, in some cases it may be helpful to explicitly model the abstraction step from a more specific to a more generic networked safety feature. In the following, DSC-Modules with an internal merging logic are introduced, which were utilized for the running example.

Qualitative DSC-Module *Platooning Mode Acceptance Check*

The qualitative DSC-Module in Fig. 4.6 has two qualitative input ports from different sources, which are connected with an *And* gate to the output port. Generally, if for a qualitative DSC-Module only qualitative input ports are defined, only the boolean merging logic based on *And* and *Or* gates can be applied.

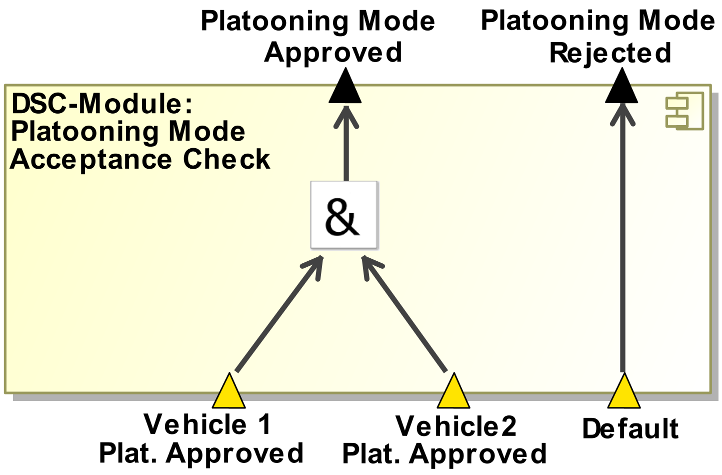


Figure 4.6: Qualitative DSC-Module *Platooning Mode Acceptance Check*

Running Example: In the running example the *Platooning Mode Acceptance Check* module monitors if both participating systems still agree to the networked mode. In the absence of the approval of one or both involved systems, the internal approval for the networked mode is lost. In the implementation scenario, the networked mode is then resolved in a predefined way. The utilization of an *Or* gate in this DSC-Module would mean that one or the other system would suffice as an approval for the networked mode.

Qualitative DSC-Module *Speed Check*

The DSC-Module in Fig. 4.7 shows a DSC-Module with two qualitative and two quantitative input ports as well as three qualitative output ports. Due to the qualitative output ports, it represents a qualitative DSC-Module. The dotted edges represent the propagation of numerical values and the solid edges represent the propagation of boolean values. To evaluate the forwarded numerical values at the quantitative input ports, a binary decision taking with an adaptive threshold using the *IF* gate is conducted. This is done using an associated scale as presented in Fig. 4.2. Generally, the consideration of external quantitative RtEs such as the *Speed Limit* is only possible in combination with an associated qualitative RtE, here the *Valid Speed Limit*. It is provided by the evaluating system platform itself. With the additional qualitative RtE, the remaining numeric value at the quantitative input port can be classified as valid or invalid. This is particularly helpful if the quantitative RtE is no longer available. In this case, the remaining value at the quantitative input port is no longer defined. The *Speed* value is considered to be part of the evaluating system platform in this scenario and requires no additional qual. RtE check.

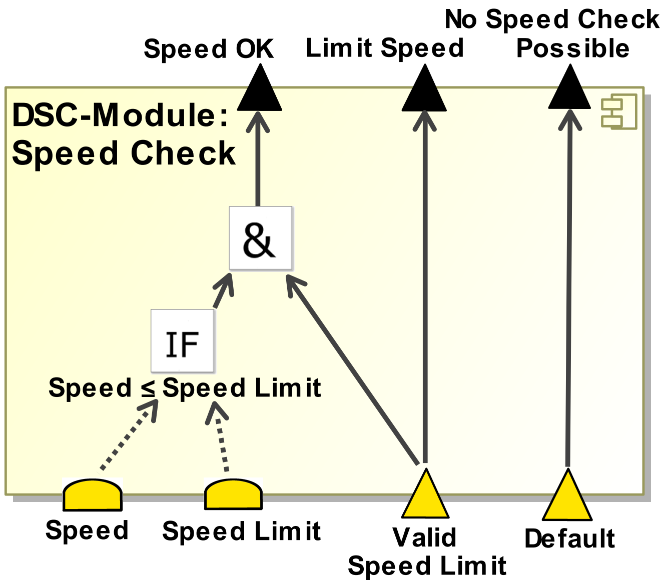


Figure 4.7: Qualitative DSC-Module *Speed Check*

Running Example: In the implementation scenario, the *Speed Check* module compares the internal current speed measurement with the external speed limit information. The quantitative *Speed Limit* RtE is shared by the master vehicle in the platooning group. For platooning on public roads, it is provided by the infrastructure. The *Speed Limit* value can vary over time, therefor the up-to-dateness of the received speed limit value has to be guaranteed. In addition, the validity of the *Speed Limit* RtE must also be checked with reference to the sender, as is the case with other RtEs. Based on the availability of the qual. *Valid Speed Limit* RtE during operation and the accordance of the current speed value with the speed limit the internal safety guarantee *Speed OK* can be provided. If the qual. RtE for a valid speed limit is available, but the current speed value is outside its specification, at least the internal safety guarantee *Limit Speed* can be provided. The contained valuable information is that although the current speed is outside its boundaries, the system is able to recover the system to the speed limit specification due to braking in an automated way. A missing *Valid Speed Limit* RtE in the vehicular infrastructure would activate the *No Speed Check Possible* output guarantee. It indicates that the evaluating system platform is not able to check this safety relevant aspects on its own. In this case, a more safety-oriented reaction behavior can be triggered. To identify a suitable reaction behavior, the *No Speed Check Possible* output guarantee as a fault description can be valuable information.

Qualitative DSC-Module *Detection Quality*

The qualitative DSC-Module in Fig. 4.8 consists out of three qualitative and two quantitative input ports as well as three qualitative output ports. To check whether the forwarded numerical values at the two quantitative input ports correlate in a certain value range, a binary decision taking based on the *IF* gate is conducted. In the *Detection Quality* module, two *IF* gates are used to check whether the two numerical values correlate in two separate value ranges. For a valid output guarantee, two additional qualitative input demands, which are linked with an *And* gate, need to be fulfilled for both cases. In this way, the adequacy of the forwarded numerical value based on the data type specification as well as the up-to-dateness of the value can be guaranteed. If the two forwarded numerical values do not correlate within the two specified value ranges or if the two associated RtE demands are not met, the most degraded output guarantee is activated.

Running Example: In the platooning scenario, the *Detection Quality* module is applied to ensure that both vehicles have reliably detected each other with respect to their relative position. Three evaluation results are considered based on a safe mutual detection in combination with a distance measurement based on the located position. The output guarantee *Safe Redundant Detection of V1 & V2* means that both vehicles confirm the mutual relative position to each other in a correct way. The evaluation result *Sensor Inconsistency* is active, if both vehicles are confident that they safely detected each other, but are mislocated. The output guarantee *No Match for Detection Quality* represents the gray zone where doubts exist because the measurement inaccuracy hinders a definitive statement.

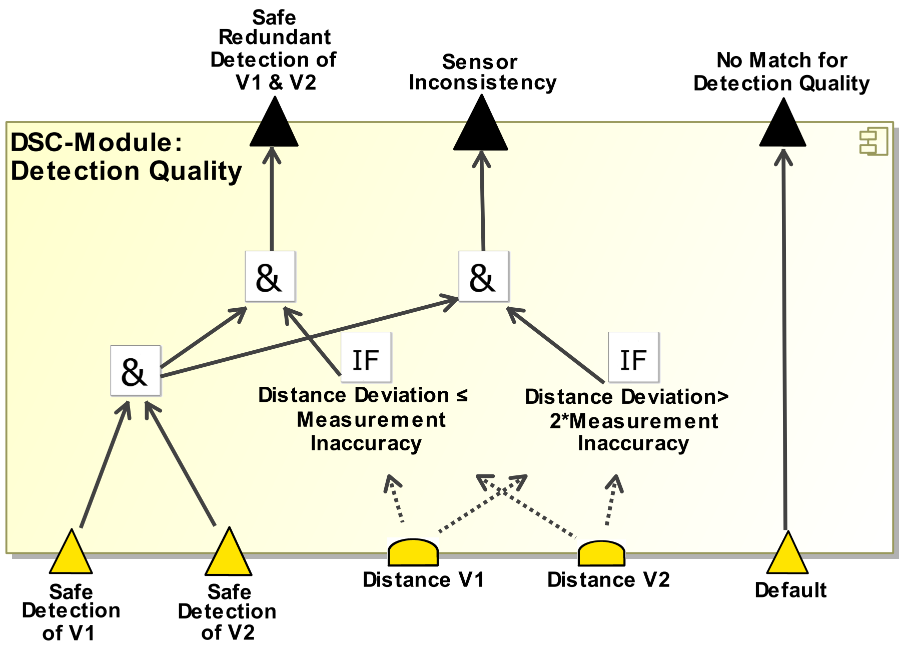

Safe Redundant Detection of V1 & V2
Sensor Inconsistency
No Match for Detection Quality
DSC-Module: Detection Quality
&
&
&
IF
Distance Deviation ≤ Measurement Inaccuracy
IF
Distance Deviation> 2*Measurement Inaccuracy
Safe Detection of V1
Safe Detection of V2
Distance V1
Distance V2
Default

Figure 4.8: Qualitative DSC-Module *Detection Quality*

Basic architecture for quantitative DSC-Modules

A quantitative DSC-Module for monitoring a certain safety property consists out of exactly one quantitative safety guarantee output port as shown in Fig. 4.9 and an undefined number of qualitative as well as quantitative demand input ports. In contrast to the qualitative DSC-Modules, where only a basic stepwise degradation concept can be specified, the quantitative DSC-Modules can be used to define a rather continuous degradation concept. Basically, this depends on the applied scale type and its increments. Thus, a more fine-grained degradation concept can be achieved with the quantitative DSC-Modules in order to obtain as much system performance as possible in a networked mode. In contrast to qualitative DSC-Modules, however, the specified quantitative DSC-Modules cannot provide different RtE output guarantees, but a single quantitative RtE output guarantee. For specifying a fine-grained degradation architecture for the output value, the *Transformation Function* gate can be utilized in combination with qualitative and/or quantitative input demands.

Quantitative DSC-Module *Platooning Driving Distance*

The quantitative DSC-Module in Fig. 4.9 consists out of four qualitative and one quantitative input ports as well as one quantitative output port. For the quantitative output port, three different degradation concepts are specified in the DSC-Module. Based on the fulfilled RtE demands one of these degradation concepts is selected. The optimal guarantee value is only forwarded to the output port if all specified input demands are fulfilled at runtime. If a particular qualitative input RtE is missing, another concept for the output value generation is applied. If no RtE demand is fulfilled, the output value is set to the maximum degraded value.

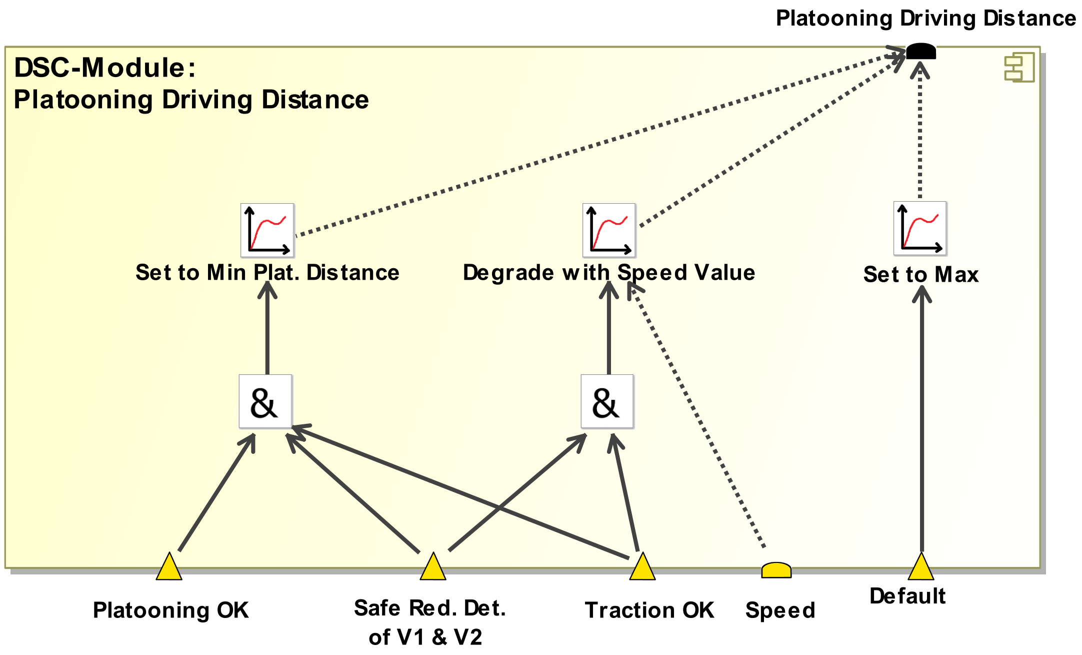

Platooning Driving Distance
DSC-Module:
Platooning Driving Distance
Set to Min Plat. Distance
Degrade with Speed Value
Set to Max
&
&
Platooning OK
Safe Red. Det. of V1 & V2
Traction OK
Speed
Default

Figure 4.9: Quantitative DSC-Module *Platooning Driving Distance*

Running Example: The *Platooning Driving Distance* module evaluates the smallest possible driving distance to the front vehicle in a platoon as a quantitative RtE output guarantee. In the platooning use case, the still safe driving distance in a platoon depends on the individual detection and reaction capability of a vehicle as well as the respective capabilities of the front vehicle. In the *Platooning Driving Distance* module, the three considered RtE demands are the external *Platooning OK* RtE, the internal *Traction OK* RtE and the *Red.Detection* RtE, which is composed from in- and external data in the *Detection Quality* module.

In case the three input RTE demands are met, a minimum possible driving distance in a platoon can be guaranteed as safe regardless of the current speed. The internal *Traction OK* RtE ensures that there are no traction problems on the considered vehicle platform. In this way a sufficient reaction capability of the considered vehicle can be guaranteed. The composed *Safe Red. Det. of V1 & V2* RtE from the *Detection Quality* module ensures that there are no difficulties with the mutual detection of vehicles in a platoon and their relative position to each other. Thus the detection capability of vehicles in a platoon is ensured. The external *Platooning OK* RtE is forwarded by the front vehicle in a platoon. It guarantees that there are no hazards detected in the vehicle ahead and confirms that warnings are directly propagated to the follower vehicle. These hazards can be internal traction problems of the front vehicle as well as detected environmental threats of the front vehicle. With the help of timely warnings, the following vehicle can react to detected hazards of the leading vehicle in a preventive manner. Thus, in case of an emergency braking of the leading vehicle, the follower vehicle can brake accordingly without substantial time delays. As a result, safety can be ensured for a constant minimal driving distance for all possible driving situations in a platoon.

If the *Platooning OK* RtE is missing, there is no direct warning for the follower vehicle. It was assumed in this case that an additional reaction time to detect sudden speed changes of the front vehicle is required. Depending on the current speed, the traveled distance varies until the emergency braking is triggered. Utilizing the current speed as an input parameter, this allows to dynamically determine the minimum necessary driving distance to the front vehicle based on the *Transformation Function* gate.

A missing *Traction OK* RtE or a missing *Safe Red. Det. of V1 & V2* RtE leads always to a maximum degradation of the output guarantee. As already described, the reasons for this are serious, such as traction problems due to a lack of grip or environmental perception problems. The DSC-Module can no longer guarantee driving distances which are suitable for platooning. It can only guarantee very large driving distances to the front vehicle where collision avoidance can still be guaranteed.

4.4.5 Composition of DSC-Modules and evaluation procedure

To ensure an efficient evaluation process of the individual DSC-Modules, the module dependencies are specified with direct assignments of in- and output ports as shown in Fig. 4.10. Thus, the qualitative and quantitative safety-relevant data can be forwarded between the modules in a fast and reliable manner. In this modeling process, the DSC-Modules have to be arranged hierarchically according to the required evaluation sequence of RtEs for higher-level safety features. Therefore, already merged RtEs which are derived from certain DSC-Modules can only be forwarded to higher-level DSC-Modules. The available basic internal and external RtEs that have not been merged in a DSC-Module can be integrated to all hierarchical evaluation levels of the composed DSC-Modules. Based on the available top-level RtEs, a suitable adaptation behavior can be specified for a networked mode. In Fig. 4.10 such an adaptation behavior is specified in the *Networked Platooning Driving Behavior* module. A detailed description of the behavior adaptation concept is provided in the following Section 4.5. Fig. 4.11 shows another composition of DSC-Modules for the *Networked IAP Mode Driving Behavior*. The IAP use case is discussed in more detail in Chapter 6 as part of the validation activities.

A predictable reaction behavior to unsafe situations can only be guaranteed based on continuously accessible safety features. This requires an ongoing check of the hierarchically integrated DSC-Modules to ensure that the current operational situation is still safe. For the specification of a DSC networking concept without static module dependencies, a negotiation process would be necessary to link the respective DSC-Module

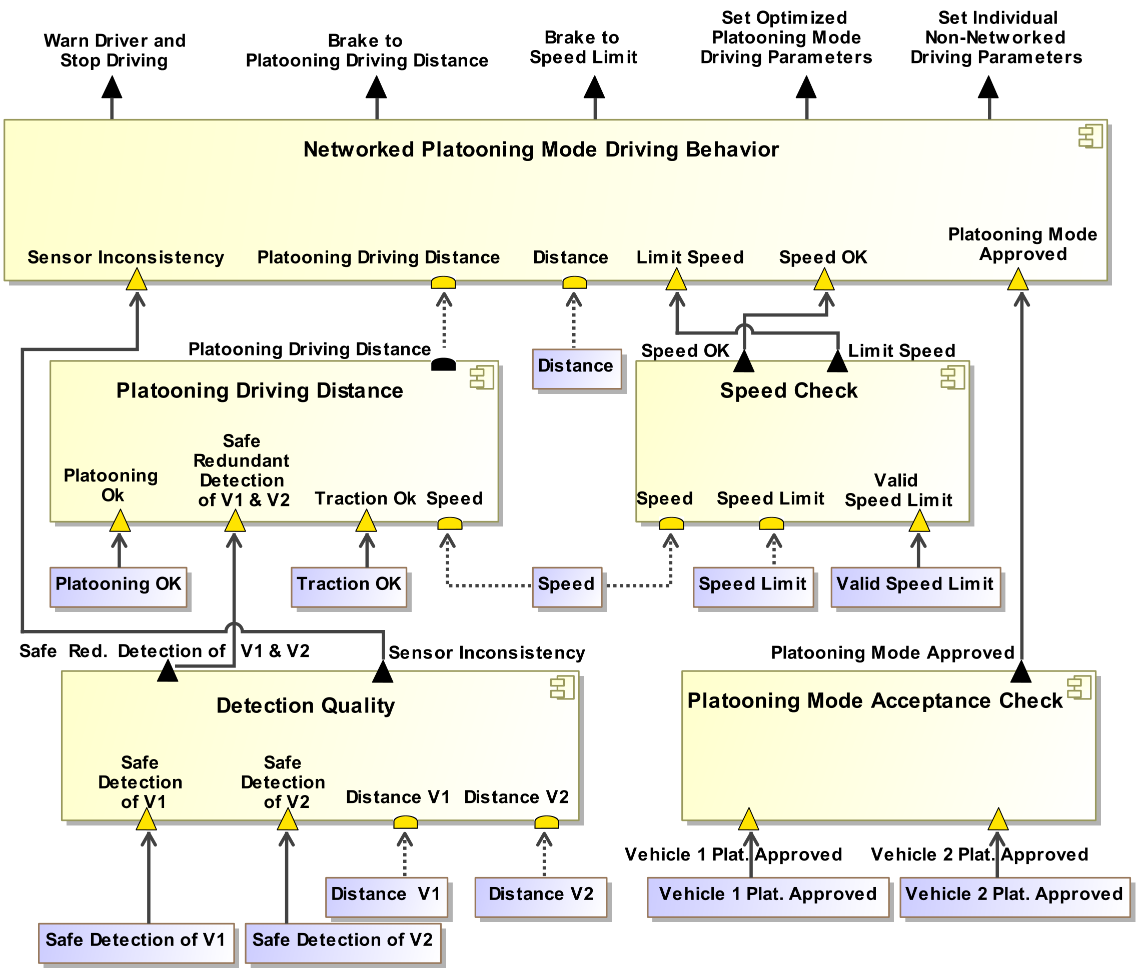


Figure 4.10: Hierarchical Integration of DSC-Modules

demands and available RtE guarantees. In the time-sensitive application context of networked driving, however, a negotiation process would be disadvantageous since it would require additional time for the cyclic evaluation mechanism. Moreover, the flexible propagation mechanism between DSC-Modules, would make the evaluation process more complex and less predictable due to ongoing rearrangement processes. Ultimately, evaluation errors or delays could trigger undesired system reactions in the networked mode.

In contrast, direct assignments of DSC-Modules enable a much more traceable and predictable evaluation concept for safety considerations. In fact, here the evaluation step of RtE allocation has already been done manually at development time. When specifying DSC-based networking concepts, however, it should be noted that the advantages of static dependency modeling are achieved with a reduction of openness and flexibility during operation of a collaborative mode. At runtime, the evaluated DSC-Modules represent the currently valid networked safety features of a system in a networked mode. Previously unknown runtime safety guarantees cannot be integrated into the ongoing evaluation process without further ado. To still maintain openness for static DSC dependency modeling, Chapter 3.6 presents a two-step approach. This approach covers the identification of optimal networking concepts at integration time and the subsequent ongoing operation at runtime.

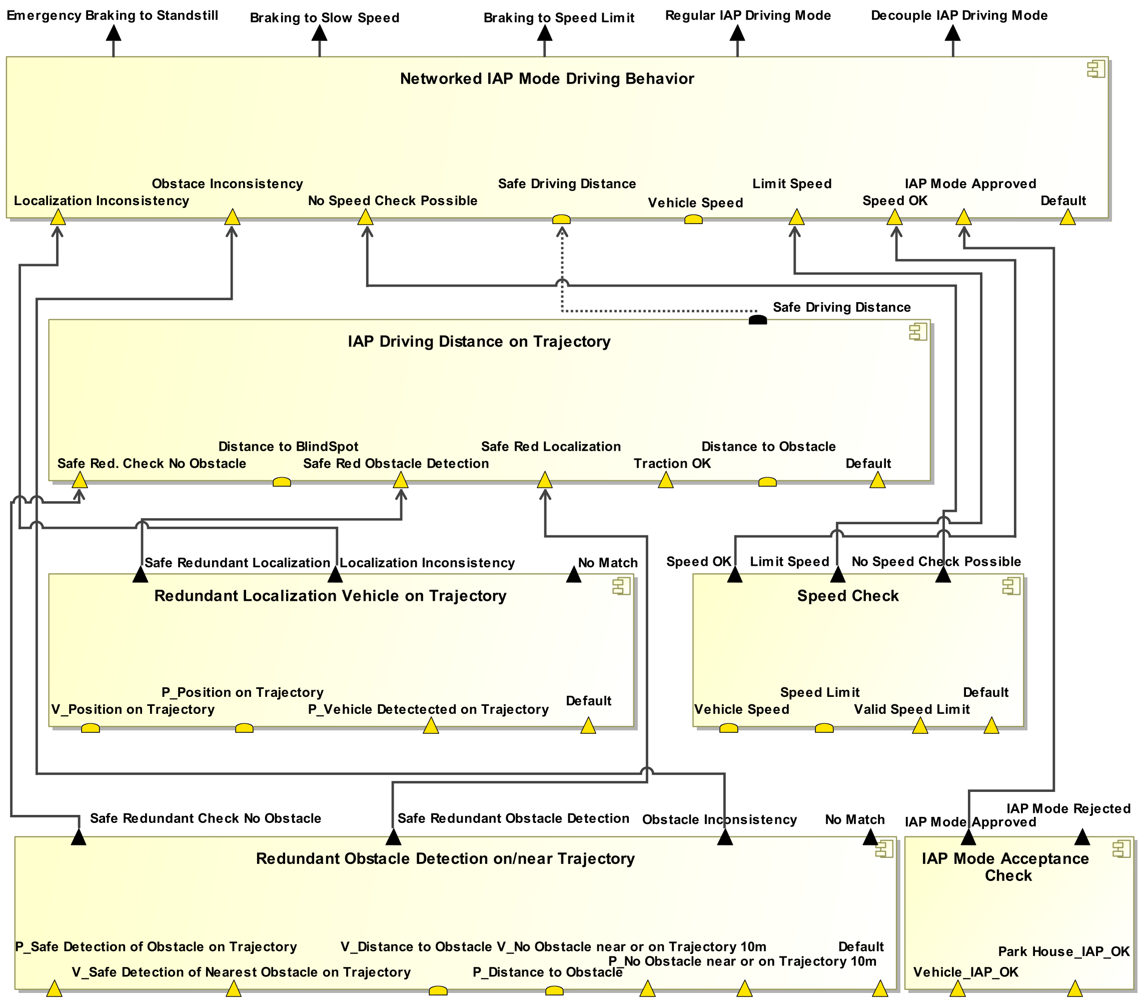

Emergency Braking to Standstill
Braking to Slow Speed
Braking to Speed Limit
Regular IAP Driving Mode
Decouple IAP Driving Mode
Networked IAP Mode Driving Behavior
Obstace Inconsistency
Localization Inconsistency
No Speed Check Possible
Safe Driving Distance
Vehicle Speed
Limit Speed
IAP Mode Approved
Speed OK
Default
Safe Driving Distance
IAP Driving Distance on Trajectory
Distance to BlindSpot
Safe Red. Check No Obstacle
Safe Red Obstacle Detection
Safe Red Localization
Distance to Obstacle
Traction OK
Default
Safe Redundant Localization
Localization Inconsistency
No Match
Redundant Localization Vehicle on Trajectory
P_Position on Trajectory
V_Position on Trajectory
P_Vehicle Detectected on Trajectory
Default
Speed OK
Limit Speed
No Speed Check Possible
Speed Check
Speed Limit
Default
Vehicle Speed
Valid Speed Limit
Safe Redundant Check No Obstacle
Safe Redundant Obstacle Detection
Obstacle Inconsistency
No Match
IAP Mode Rejected
IAP Mode Approved
Redundant Obstacle Detection on/near Trajectory
P_Safe Detection of Obstacle on Trajectory
V_Distance to Obstacle
V_No Obstacle near or on Trajectory 10m
Default
P_No Obstacle near or on Trajectory 10m
V_Safe Detection of Nearest Obstacle on Trajectory
P_Distance to Obstacle
IAP Mode Acceptance Check
Park House_IAP_OK
Vehicle_IAP_OK

Figure 4.11: Composition of DSC-Modules for IAP Mode

4.5 Top-Level Safety Quality Attributes and Dynamic Adaptation Behavior

To specify a dynamic reaction behavior for degraded safety features of a networked mode at runtime, a qualitative DSC-Module based on qualitative and quantitative RtE input demands as shown in Fig. 4.12 was utilized. The considered demand input ports represent the most relevant safety features for deriving a suitable networked system behavior. The guarantee output ports represent different networked behavior that can be triggered during operation of a networked mode.

The evaluation order for the validity of the networked adaptation behavior is from left to right, the same as for the DSC-Modules described before. As a design decision, the outputs for the networked adaptation behavior are always arranged in such a way that particularly safety-critical networked situations are checked first according to their criticality. Since the related adaption behavior are considered as safety-relevant emergency behavior, they should be triggered very rarely in a regular networked mode. If this is not the case, the safety concept must be revised. Typically, most of the time the safety-certified approval for enhanced system functions should be active in a networked mode. Here, the involved systems should be able to maintain the permitted degrees of freedom without constant safety-related interventions. In order to leave the networked system mode without safety-related interventions, the safe decoupling of systems is also specified for networked modes. The resulting overall virtual safety cage for a networked system is illustrated in Fig. 3.5.

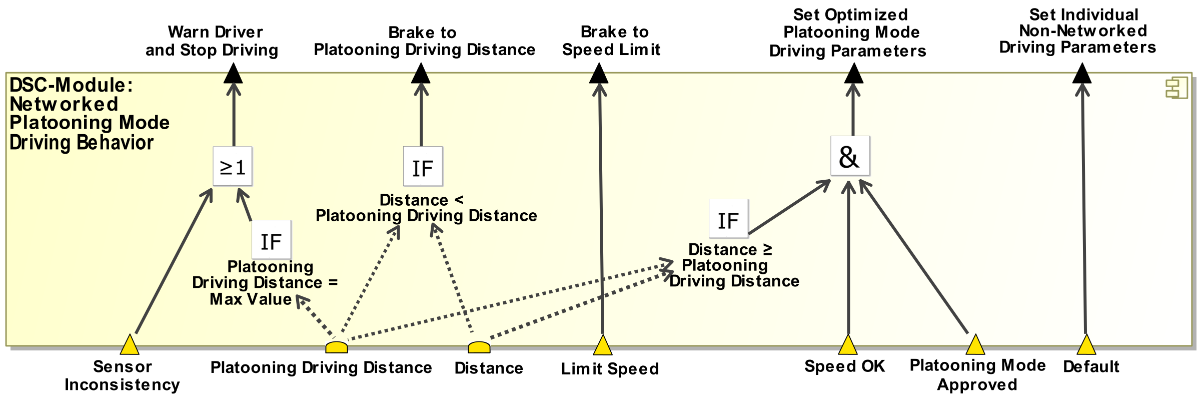

Warn Driver and Stop Driving
Brake to Platooning Driving Distance
Brake to Speed Limit
Set Optimized Platooning Mode Driving Parameters
Set Individual Non-Networked Driving Parameters
DSC-Module: Networked Platooning Mode Driving Behavior
≥1
IF
Distance < Platooning Driving Distance
IF
Platooning Driving Distance = Max Value
&
IF
Distance ≥ Platooning Driving Distance
Sensor Inconsistency
Platooning Driving Distance
Distance
Limit Speed
Speed OK
Platooning Mode Approved
Default

Figure 4.12: Qualitative DSC-Module *Networked Platooning Mode Driving Behavior*

Running Example: In the platooning scenario, the *Networked Platooning Mode Driving Behavior* Module in Fig. 4.12 specifies how the system should react in the various collaboration situations of a networked mode based on the available runtime safety guarantees.

The most hazardous situation occurs when the networked system can no longer guarantee safety for the networked autonomous driving behavior. This can be caused by a total loss of the vehicles reaction capability due to traction problems. In such a situation it is not possible to guarantee a safe driving distance anymore. Alternatively this can be caused by a total loss of the vehicles environmental perception capability due to sensor problems where the vehicle is not able to detect the front vehicle anymore. In this case the *Warn Driver and Stop Driving* output behavior is activated. In such a networked driving situation, the human driver should be warned and asked to take over control. In parallel, preventive safety measures such as emergency braking should be initialized.

In case the following vehicle is not complying with the evaluated safe driving distance to the front vehicle, the networked system behavior initializes a brake command until a safe operational state is recovered. The reason for this is not necessarily an inappropriate driving style of the following vehicle. The safe driving distance can also be changed by the front vehicle if it suddenly stops providing the *Platooning OK* RtE. The reason for the missing forwarding of the RtE can be a detected hazard for the front vehicle such as traction problems.

In case the driving distance to the front vehicle is sufficient, but the vehicle is driving too fast, another brake command with a smaller deceleration force is triggered. At that moment when the vehicle has sufficiently reduced the speed considering the speed limit, the *Speed Check* module switches from *Limit Speed* to *Speed OK*. This means that the *Limit Speed* RtE is no longer available and the validity of the next possible adaptation behavior can be checked.

Considering a too short driving distance and a too high speed at the same time, a prioritized reaction behavior for recovering the safe driving distance is performed. This order was chosen because it was assumed that a too short driving distance to the front vehicle is probably more hazardous than exceeding the speed limit.

In case the *Speed OK* RtE is available and the current driving distance to the front vehicle is larger or equal to the evaluated minimum safe driving distance, the *Set Optimized Platooning Mode Driving Parameters* can be provided in principle. However, in addition a *Platooning Mode Approved* RtE as an input demand was introduced to enable the networked system mode to be deactivated without triggering a safety-related emergency behavior. This would otherwise be the case if the shared RtEs would suddenly no longer be available. As a result, if neither hazardous situations can be detected nor the extended networked mode is valid, the individual mode *Set Individual Non-Networked Driving Parameters* is automatically activated. Specific states of DSC-Modules that are not explicitly modeled, such as *No Speed Check Possible* or *Platooning Mode Rejected*, automatically lead to the dissolution of the networked mode.

4.6 GUI-based Engineering and Transformation into a Computable Representation

For the specification of DSC-based networking modes a dedicated GUI was developed by Niklas Klipphahn. This was done based on a MagicDraw plug-in [87] as shown in Fig. 4.13. The different gate types can be inserted into the DSC-Modules via drag and drop. The correct interconnection of ports and gates is ensured by smart manipulators. The solid arrows indicate the forwarding of qualitative data and the dotted arrows indicate quantitative data. DSC-Modules and RtEs can be imported and composed in a comprehensive way as shown in Fig. 4.10 and Fig. 4.11. Based on the GUI model a automated code generation was implemented.

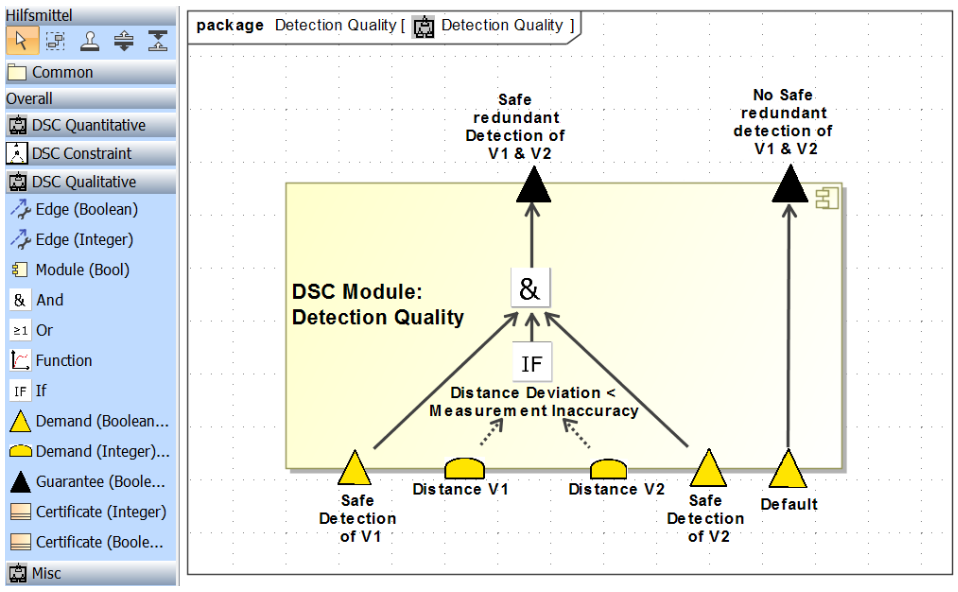


Figure 4.13: MagicDraw GUI for DSC Specification

Basically, the overall computation time of networked DSC-Modules is time-critical. It can affect the performance of networked systems in many of the potential DSC application domains. In particular, the worst case evaluation time is important as it sets the minimum possible time delay of the networked mode reaction behavior in critical situations. For the networked DSC-Modules as shown in Fig. 4.10 and Fig. 4.11 various evaluation methods were analyzed. Generally, the most appropriate way to reduce evaluation times is to convert the existing fault tree structures from DSCs into BDDs. Yi et al. [88] showed that fault tree structures can be systematically converted into BDDs. To enable this, they have to be available in a structured data format such as XML. These aspects can be looked up in Section 6.2.1 as part of the validation activities. In particular, it became apparent that, depending on the concrete networked mode, different types of evaluations such as module-oriented or behavior-oriented evaluations can be advantageous.

5 Engineering of DSC Networking Modes

This Chapter provides guidance for the systematic derivation of networked DSC applications. A suitable basic structure for DSC-Modules as well as a transformation strategy into appropriate runtime representations was already introduced in the previous Chapter. The focus here is therefore on establishing adequate safety engineering activities. These activities should be carried out completely at development time with the ultimate goal of enabling safety certification by authorities for a particular networked DSC mode.

It should be noted, however, that it cannot be the intention of this Chapter to provide a full-fledged solution concept for each concrete use case. The DSC approach is, due to its generic specification concept, basically applicable in a broad range of possible application domains and scenarios. The necessary safety engineering steps, however, depend strongly on the specific system configuration and the respective application context. Moreover, the development of a holistic safety concept for practical application is often rather unsystematic. It relies heavily on domain specific expert knowledge and the experience of safety engineers. For this reason, this Chapter focuses on the overall DSC safety engineering strategy for networking systems to support engineering processes in the respective application domains and scenarios.

Basically, for the specification of networked DSC applications, it seems to be particularly useful to differentiate between the necessary engineering activities on the domain-level as well as on the system-level. The subdivision into domain-level and system-level helps engineers to focus either on common scenario-related aspects or on specific implementation considerations. Moreover, certain identified scenario-related issues of networked DSC applications can be relevant for a variety of concrete implementations in the respective context. Hence, after their identification, they should be taken into account in all related DSC applications of that domain in order to continuously improve the system-level engineering based on it. The domain-level engineering can thus support the systematic capturing of all relevant DSC networking features and the derivation of a conclusive safety argumentation in the respective domain.

Naturally, an important prerequisite for this is a suitable framework for the management of the gathered knowledge. It should help to associate the relevant findings in a domain with each other and to make it accessible for the system-level engineering. The concept of a *Safety Domain Model* (SDM) [55], as illustrated in Figure 5.2, seems to be particular useful for this purpose. Based on the domain-level engineering, the utilization of a SDM can support the efficient derivation of concrete safety goals as well as a holistic safety concept on the system-level for the realization of concrete application scenarios. Beyond that, utilizing a SDM for the DSCs approach can foster the systematic reuse of parts of already specified DSC modes for other networked applications. To this end, Section 5.3 proposes a strategy to link the already specified DSC-Modules on the system-level with the related SDM findings according to their networking objectives.

In the following, Section 5.1 describes the objectives and procedures for safety analysis and engineering of networked DSC applications on the domain-level. Subsequently, Section 5.2 explains the corresponding system-level parts to derive concrete DSC networking modes. Finally, Section 5.3 shows how parts of already specified DSC networking modes can be systematically reused in other similar applications.

5.1 Domain-Level Safety Analysis and Engineering

The overall objective of safety analysis and engineering on the domain-level is to identify potential networked applications and their associated hazards in a given context. In addition, as will be explained later, it is necessary to derive suitable conditions for ensuring the safe operation of particular networked applications already in this phase. The corresponding findings can then be aggregated in the SDM as illustrated in Fig. 5.2. Subsequently, on the system-level, concrete application scenarios can then be developed based on these findings. The use of a SDM is, however, not mandatory since it only provides additional support for the systematic domain-level engineering.

As a first step on the domain level, potential collaboration scenarios of the addressed system platforms in a specific context have to be identified. For the participating system platforms, this requires the analysis of the respective system characteristics as well as the available service types in order to derive possible networked applications from it. In the automotive domain, for example, the driving dynamics of particular vehicles as well as their on-board sensors and communication technology are crucial. In the next step, the possible networked situations that need to be addressed for a particular application scenario have to be specified. For the presented platooning use case, for example, this can refer solely to off-road applications or also to specific highway settings. Accordingly, different environmental perception concepts and driving maneuvers have to be considered for different networked driving situations.

Once the overall operating conditions for a collaboration scenario are clarified, a comprehensive safety analysis can be started to identify potential hazards. In the automotive sector, for example, the ISO 26262 standard is typically used for hazard and risk analysis (HARA) of the resulting system configuration. Here, the focus is on faults in the service provision concept, which can ultimately lead to a hazard. This can be, for instance, the unintended omission or commission of specific service types. In principle, it could be tried to carry out such a HARA also for collaboration scenarios based on the resulting combination of the analyzed systems, the application scenarios and the possible situations in the planned networked mode. Later on, at the system level, concrete safety measures have to be taken to prevent or mitigate these identified hazardous states in practical applications.

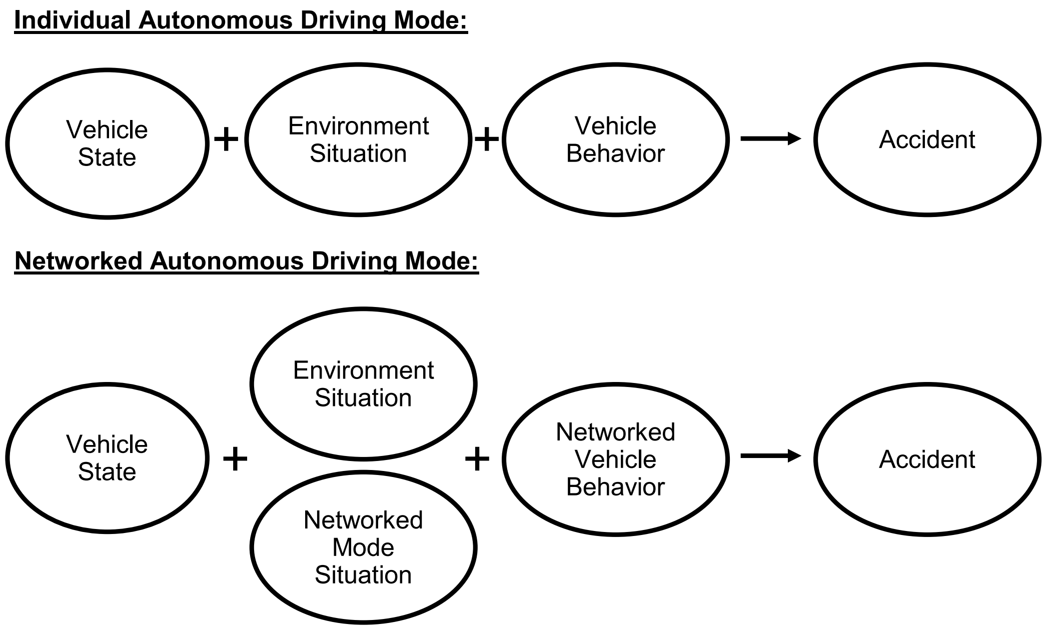


Figure 5.1: HARA for Individual and Networked Autonomous Driving Modes

However, when trying to transfer such conventional HARAs to networked driving applications such as platooning, it becomes apparent that this approach is not suitable for identifying hazardous states in collaboration scenarios. In the following, this is explained using Fig. 5.1. Here, the relevant factors that can lead to an accident for individual as well as networked autonomous driving are illustrated. The *Networked Autonomous Driving Mode* extends and modifies the existing *Individual Autonomous Driving Mode* of vehicles according to the respective networking characteristics. In this context, the *Networked Mode Situation* represents new driving situations. For the presented platooning use case this can be, for example, shorter driving distances to the front vehicle. Here, the conventional HARA aims at faults in the service provision concept for the intended behavior. Part of the intended behavior in a platoon can be, for example, braking or evasive maneuvers. When analyzing the omission of an associated trigger signal of the networked follower vehicle in a platoon, however, it becomes apparent that it does not necessarily lead to a collision with the front vehicle. This can be the case when the front vehicle accelerates or changes lanes at the same time. Hence, it can be concluded that focusing on faulty services is not suitable for reliably identifying hazards for collaboration scenarios.

In consequence, other safety engineering strategies have to be utilized to ensure safety for collaboration scenarios such as the networked driving modes in the automotive domain. A possible solution is to concentrate on the safe operating conditions that must be met in order to exclude the occurrence of the potential hazards during operation. For the derivation of safe collaboration conditions, however, no straightforward assessment approach is available. One option can be to decompose the functional dependencies of a networked application. Nevertheless, the addressed collaboration scenarios are strongly dependent on the environmental application context and are therefore difficult to analyze in a unified holistic model. As a result, the existing dependencies of networked systems and their application environment in a collaboration scenario can be difficult to assess.

To some extent, this issue can be solved by defining a SDM in which once identified safe collaboration conditions can be stored in a structured way and reused whenever required. In this way, it is also possible to conduct additional iteration routines later on to modify or integrate new safe collaboration conditions. Also the outlined simulation activities in this thesis can support the derivation of hazardous situations and the related safe operating conditions. In a very mature state of the collaboration scenario, it also seems reasonable to consider newly identified hazardous states from real world test applications for the optimization of the SDM.

For the presented platooning use case, especially the driving distance for the networked follower vehicle to the front vehicle was identified as a safety-critical condition. Depending on the current driving speed and further runtime parameters, the required safe driving distance fluctuates and was therefore specified in a variable way for the platooning mode. As long as the necessary driving distance can be maintained in the networked platooning mode, the risk of a collision is sufficiently low.

In addition to the necessary safe driving distance, also other factors were identified to be relevant for safety in a platooning mode. Especially the actual perception capability of networked systems to localize each other, as a precondition for a reliable situation assessment, was considered to be crucial. Another important factor that was identified are the internal system conditions of the networked vehicles in a platoon. They can also be safety-critical for the surrounding vehicles. Other external aspects that can be relevant for safe platooning, such as the current speed limit, were also taken into account.

5.2 System-Level Safety Analysis and Engineering

For implementing the proposed DSC approach in real applications, the main objective on the system-level is the concrete specification of RtEs, DSC-Modules and the overall networked DSC mode according to the specific application environment of the collaboration scenario. The ultimate goal is to establish a conclusive safety concept for the real application that enables safety certification by the certificate issuer. To support this process, this Section particularly highlights differences to conventional safety engineering concepts and describes corresponding safety engineering strategies.

On the basis of the domain-level engineering for the applications considered in the SDM, it becomes possible to identify networked applications that appear particularly promising for the system-level engineering. In a first step on the system-level, as illustrated in Fig. 5.2, a collaboration scenario as well as the addressed system platforms have to be selected from the SDM. In the next step, suitable safety goals and a holistic safety concept for the concrete implementation have to be derived for the selected scenario. This is done on the one hand by conventional internal safety measures of the involved systems and on the other hand on the basis of the derived safe collaboration conditions.

For internal safety measures of involved systems in collaboration scenarios, today's state-of-the-practice safety assurance techniques can be applied. Possible causes for the violation of intended functions in scenario applications have to be analyzed for their probability of failure and the consequential severity of harm. Broadly applied safety analysis techniques, such as FTA or FMEA, are well suited to assess these malfunctions in particular system configurations. The addressed systems can thus be checked for safety weaknesses to ensure that they meet the defined safety goals.

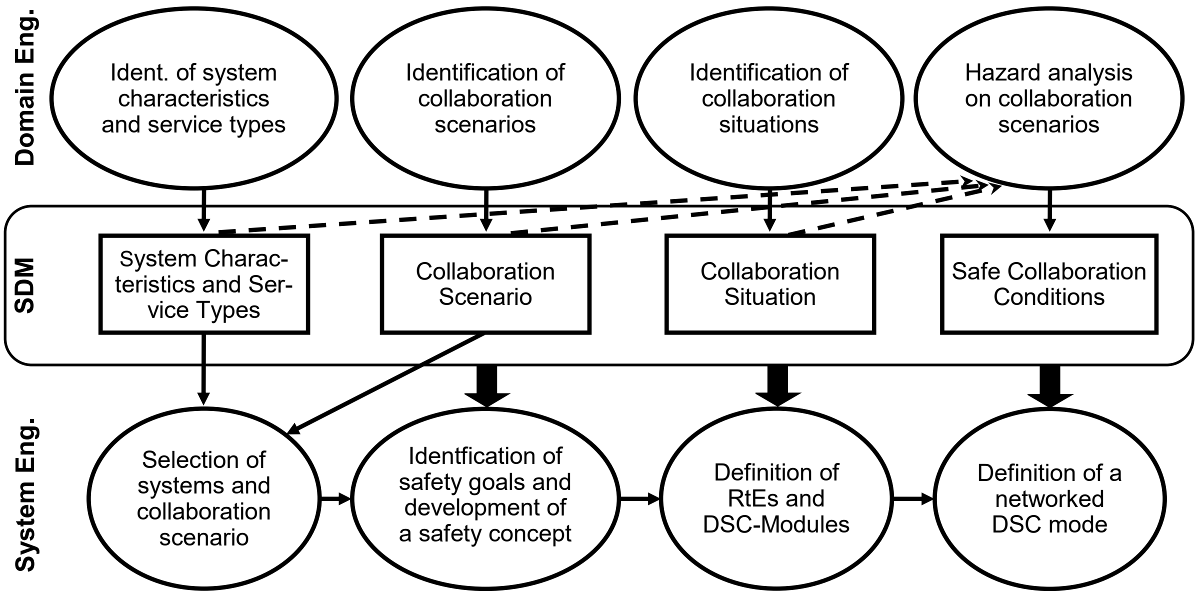

Domain Eng.
Ident. of system characteristics and service types
Identification of collaboration scenarios
Identification of collaboration situations
Hazard analysis on collaboration scenarios
SDM
System Charac-teristics and Ser-vice Types
Collaboration Scenario
Collaboration Situation
Safe Collaboration Conditions
System Eng.
Selection of systems and collaboration scenario
Identfication of safety goals and development of a safety concept
Definition of RtEs and DSC-Modules
Definition of a networked DSC mode

Figure 5.2: Safety Engineering Strategy and the SDM

In order to guarantee safety for the specified collaboration scenarios based on the identified *Safe Collaboration Conditions*, the decentralized safety knowledge of the involved systems has to be merged and evaluated in a predefined way. Then, on this basis, appropriate adaptations of the system behavior on the target platform have to be carried out in the dynamic collaboration context. For this purpose, suitable DSC-Modules representing the *Safe Collaboration Conditions* have to be defined for the DSC approach. For the presented platooning use case, for example, the reliable mutual detection of vehicles in a platoon was identified to be a crucial condition for safe platooning. Since environmental factors such as fog can influence this safe collaboration condition, it has to be continuously monitored during operation. If this condition is degraded while driving in a networked vehicle group, appropriate emergency maneuvers have to be initiated. For the specification of DSC-Modules with respect to the derived safe collaboration conditions, it has to be clarified whether a qualitative or quantitative module representation is more appropriate. In general, the quantitative description form is more suitable for networked modes for performance reasons, but its use depends on the availability of suitable input values.

In addition, the concrete RtEs representing the formalized input data for DSC-Modules need to be defined. If basic RtEs for DSC-Modules are specified this has to be made in accordance to the SDM dependencies and to the already established DSC-Modules. For the definition of basic RtEs also the potential utilization in other collaboration scenarios has to be considered. Defining a new RtE is also a question of the availability of the underlying service types on the addressed system platforms. In addition, higher level RtEs based on established DSCs have to be analyzed for a correct integration of subordinated guarantees. Because of the modular concept, however, safety engineers only have to focus on a particular demand-guarantee relationship of DSCs.

Basically, safety for a networked DSC mode is thus checked based on the availability of particular RtEs in a specific networked situation of a collaboration scenario. In the platooning use case, for example, the safe detection of the front vehicle represents such a RtE. Consequently, the focus on the system-level should be on the reliable generation of the RtEs based on the actual system configuration in a realistic collaboration scenario environment. As a prerequisite for this, the implemented sensor concept needs a suitable safety classification such as the SIL or ASIL classification as an internal safety measure. It should correspond to the high endangerment for collaboration scenarios such as in the automotive domain for networked driving.

In the presented platooning use case, a modified driving behavior for a networked vehicle is derived from the DSC evaluation. The modified behavior is specified based on the degraded DSC safety features on the target system platform caused by missing RtEs as input demands. In a networked DSC mode, degraded safety features force a reaction of the system to return to a safe state. Since the DSC evaluation based on BDDs is very lightweight, the achievable reaction time to changing DSC safety features in the networked system behavior depends primarily on the concrete implementation. When the networked DSC mode is specified, extensive integration tests have to be conducted on the target application platform as a proof for sufficiently short response times. Hence, safety certification for a networked DSC application is only possible with respect to the concrete system setting. If either the computing capacity, the communication technology or other platform-related aspects are modified, new integration tests for the specified DSC mode are required.

In principle, a certificate can only be issued if the intended modified DSC behavior for all conceivable situations of a collaboration scenario can be demonstrated by practical tests. Therefore, more complex collaboration concepts tend to require a much higher effort for testing in order to check all possible scenario situations. Nevertheless, it can be observed today in the certification of autonomous driving functions that this generally poses a significant challenge due to the open application context and regulatory issues. Due to the modular approach, it can be tried to reuse already certified parts of a collaboration scenario such as the use of certain DSC-Modules and RtEs on specific target platforms. In this way, the scope of the necessary certification activities could be reduced, at least for the platform integration of the DSC evaluation.

5.3 Systematic Reuse of DSC-Modules

This Section focuses on the systematic reuse of parts of already specified DSC networking modes for other networked applications. A particular DSC mode, as shown in Fig. 4.10, consists of several DSC-Modules that are directly connected to each other to enable a fast and reliable DSC evaluation on the respective system platforms. Basically, however, each collaboration scenario requires an individual DSC networking mode. This is because it consists either of different collaboration partners with other system characteristics and available service types, or of different collaboration goals with different identified conditions for a safe collaboration. As a result, as outlined in the previous Section, also a new conclusive safety concept for safety certification is required for each new DSC networking mode. Due to the complex decision-making process, this cannot be automated in an arbitrary manner. For the proposed DSC approach, however, it is possible to significantly reduce the development effort by standardizing shared RtEs and systematically reusing DSC-Modules or individual DSCs.

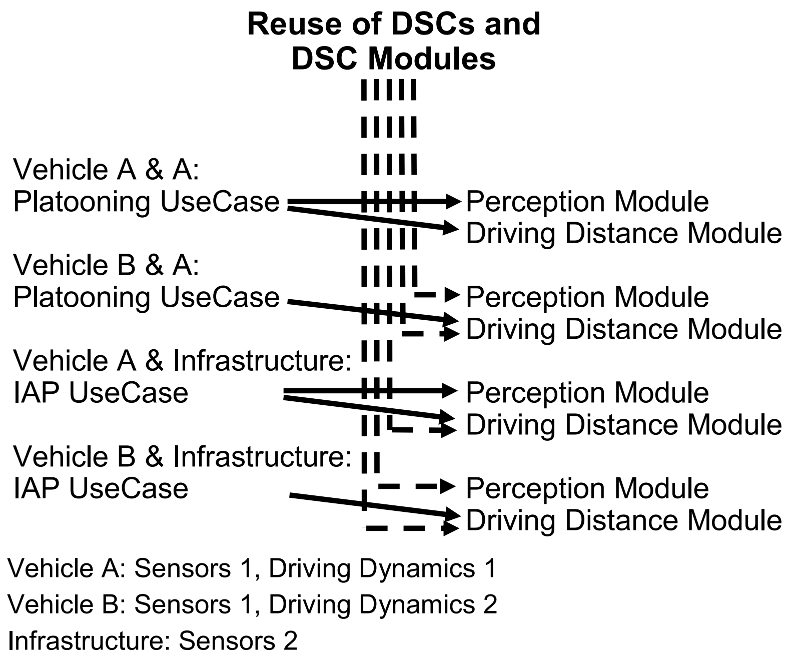

Reuse of DSCs and DSC Modules
Vehicle A & A: Platooning UseCase
Perception Module
Driving Distance Module
Vehicle B & A: Platooning UseCase
Perception Module
Driving Distance Module
Vehicle A & Infrastructure: IAP UseCase
Perception Module
Driving Distance Module
Vehicle B & Infrastructure: IAP UseCase
Perception Module
Driving Distance Module
Vehicle A: Sensors 1, Driving Dynamics 1
Vehicle B: Sensors 1, Driving Dynamics 2
Infrastructure: Sensors 2

Figure 5.3: Systematic Reuse of DSCs and DSC-Modules

In the following, the intended reuse strategy for the DSC approach as shown in Fig. 5.3

is explained based on a successive development process of several networked DSC applications. In the initial phase, for example, the presented platooning use case should be developed with two vehicles of type A without already having previous knowledge from similar applications. This requires the specification of the utilized DSC-Modules and the shared RtEs. The utilized DSC-Modules are, for example, the already described DSC *Perception Module* for mutual detection as well as the DSC-Module for deriving the safe driving distance. Subsequently, the platooning use case should be extended to another following vehicle type B equipped with the same on-board sensors. Such situations may typically occur with different vehicles from the same manufacturer. Here, the already specified RtEs as well as the overall *Perception Module* can be reused. Only the *Driving Distance Module* has to be adapted to the different driving dynamics of the new following vehicle. In the next step in the successive development of networked applications, the IAP scenario as a new scenario for networked driving should be specified for vehicles A and B. For the first vehicle under consideration, a new environmental perception concept based on a new *Perception Module* for the parking garage and vehicle A needs to be defined. From the platooning use case, however, the already specified RtEs for reliable vehicle recognition can possibly be reused. Also the existing *Driving Distance Module* based on the vehicle A driving dynamics can be partially reused. Afterwards, for integrating vehicle B into the IAP scenario, the *Perception Module* from vehicle A in the IAP scenario can be completely reused. For the necessary *Driving Distance Module* for vehicle B, the scenario-related DSC-Module specification from vehicle A and the platform-related knowledge from the platooning use case for vehicle B can be reused.

With a larger number of collaboration scenarios under consideration, the advantages of domain-level engineering in combination with the SDM for standardization also become more apparent. Already well-developed collaboration scenarios can be extended to other systems with regard to the considered situations and the identified safe collaboration conditions. In addition, for new collaboration scenarios, similar conditions for safe collaborations can be derived from existing ones. It can be assumed, therefore, that the application of a SDM is particularly useful for systems that are generally open to collaboration processes as required in public road traffic. Besides that, the utilization of a SDM can also be beneficial for standardization of shared RtEs according to most common safe collaboration conditions.

6 Validation

In this Chapter, the validation activities of the presented DSCs approach as the core solution concept are summarized. They were carried out in parallel with the evolving approach. The conducted validation activities can be divided into the applicability of the development time parts of the approach as provided in Chapter 4 and 5 as well as the operationalization of the runtime parts of the approach as given in Chapter 4.

The validation of the development time parts in Section 6.1 focuses on the suitability of the conceptual modeling framework as well as the design guidelines to systematically derive networked applications using DSCs. The experience gained from several applications in the automotive domain was utilized for this purpose.

Section 6.2 deals with the validation of the technical feasibility of the required runtime mechanisms. First, the findings regarding feasibility and adequacy of various DSC runtime representations with respect to their computation times are presented. Then, the focus is on the feasibility of the planned negotiation mechanism for the flexible initialization of DSC networks. This is clarified based on the experience gained with a dedicated simulation framework. Lastly, the validation activities regarding robustness and correctness of the specified dynamic adaptation behavior are explained. For this purpose, a comprehensive 3D simulation environment was set up in the context of networked vehicle fleets. It is designed in such a way that it depicts complex application environments as realistically as possible.

6.1 Applicability of the Core Concepts and the Methodology

For the development time parts of the approach, the focus is on validating the conceptual framework of DSCs as well as the derived engineering steps. For this purpose, insights are given into the experience gained with the provided engineering backbone in several use cases. As described in Chapter 5, for the specification of networked applications using DSCs, the engineering process is subdivided into the domain-level and system-level engineering. The findings from the domain-level are stored in the Safety Domain Model (SDM), which forms the basis for the definition of RtEs, DSC-Modules and the entire dynamic adaptation behavior at system-level. Especially the reusability of parts of the DSC specification is an important feature to support efficient development and certification processes. Specified overall DSC-Modules as well as individual DSCs as part of the DSC-Module description for monitoring networked safety features can be stored in the SDM to make them reusable for other similar applications. In the following, these synergy effects are also addressed by the selection of two applications from the same domain, here the automotive sector.

To validate the specified development time engineering activities for the DSCs approach, the platooning scenario presented in Section 3.2 was selected as a standard test scenario. The platooning scenario is particularly beneficial as it could be applied with relatively small modifications to a wide range of similar applications of networked vehicle fleets as presented in Chapter 4.1. Moreover, this scenario allows to demonstrate the correct functioning and the resulting benefits of networked vehicle fleets using DSCs in a comprehensible manner. Besides the platooning scenario, an industry use case for Intelligent Automated Parking (IAP) was developed based on the presented solution approach. It represents a highly customized DSC networking mode in accordance to the detailed specification for networked driving in an parking garage. Here, a suitable networking concept had to be systematically derived from a multitude of requirements. The IAP scenario also demonstrates the reusability of already specified individual DSCs as well as complete DSC-Modules. Networked safety features that have already been specified in the platooning scenario were utilized for this purpose.

6.1.1 Use Case: Safe platooning in networked vehicle fleets

The considered platooning use case as shown in Fig. 6.1 was part of a research project for the safe cooperation and navigation of networked vehicle fleets in the commercial vehicle domain. To obtain a realistic prototype application for the networking of vehicle fleets, so-called Gator vehicles from the Robot Research Lab at TU Kaiserslautern were utilized as a demonstrator platform. They are specially designed for networked driving applications, but due to legal reasons they were not ready for practical use at the time of validation. However, since the sensor hardware was already integrated into the vehicle platform and a sensor fusion concept was specified, suitable service types for establishing networked driving modes could be identified.

As a first step in the SDM engineering, the possible beneficial networked driving scenarios in combination with their associated driving situations that could occur during operation were derived and added to the SDM. In the next step, for the further engineering of the SDM, a HAZOP analysis was performed to identify the relevant networked safety features for a networked driving mode as outlined in Fig. 5.2. The necessary safety-relevant services from the involved systems could be derived in this way. Based on this, the relevant service types for establishing master-slave configurations for platooning could be identified at the system-level. When the SDM engineering was considered to be complete with the derived safety features, the system-level engineering was initialized for specifying suitable RtE services, DSC-Modules and their evaluation concept. In Table

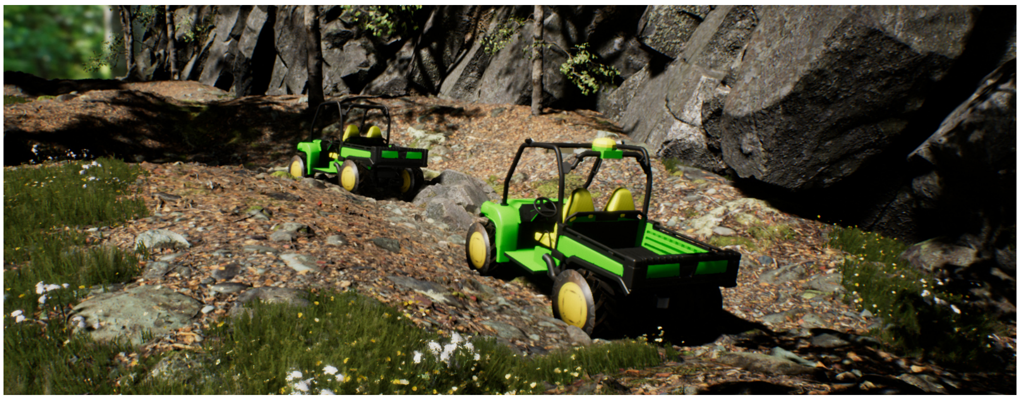


Figure 6.1: Simulated Gator Platoon in a Master-Slave Vehicle Configuration [84]

4.1 in Chapter 4 the derived RtE services are shown. The specified DSC-Modules as well as the overall adaptation concept for this scenario are described in Section 4.4 and 4.5. For the specification of the DSC networking concept, a dedicated GUI, as illustrated in Fig. 4.13, was utilized.

As it became apparent, specifying and interpreting the individual safety contracts as well as the entire DSC-Modules is still intuitive for developers. This is particularly important for the division of work according to expert knowledge in the development of safety-critical applications. Project members who were not familiar with the DSC modeling approach were able to understand the included content without misunderstandings. Also the defined interface semantic for the conceptual modeling framework seems to be well suited. In particular, the quantitative interface semantic, besides the qualitative ones, could be applied beneficially for the definition of a graceful degradation concept in a traceable manner. Thus a gradual and not abrupt degradation of networked system features could be defined.

Moreover, it became evident that the modularized specification approach helps to divide complex safety dependencies into manageable parts. Thus, project members were able to optimize individual DSC-Modules, such as the presented module for enhanced spatial perception, without having the entire evaluation model in mind. It could thus help safety engineers to collaborate by providing fine-grained task description for involved expert groups. Vehicle manufacturers, for example, have detailed knowledge regarding the driving dynamics of their vehicle platforms. However, since they typically use third-party sensors for environmental perception, they are not necessarily specialists in this field. The division of work for modeling the networked platooning mode from Chapter 4 could look like this. The DSC-Module for an enhanced environmental perception of networked vehicles could be developed and supplied by experts from the sensor manufacturer. The vehicle manufacturer who implements the physical sensors only has to integrate this module into the DSC networking concept. In this way, vehicle manufacturers can focus on their expertise for driving dynamics in order to specify adapted, networked driving behavior.

Next to the correct specification of individual DSC-Modules, the feasibility of a proper linking of modules to specify a suitable dynamic adaptation behavior is essential. For the platooning scenario, an overall reaction behavior has been successfully developed based on the introduced DSC-Modules as described in Section 4.5. The derived contract-based reaction behavior in the platooning scenario is, however, only one way for a possible implementation. The advantage here is that a lightweight pre-configured variability can be determined in a reliable manner only on the basis of the DSCs evaluation mechanism. This is particularly helpful for understanding the DSC operating mechanism, since the adaptation behavior is specified in an explicit manner. Due to the very systematic adaptation concept, it may also be possible to reuse parts of it. However, corresponding interfaces that influence the regular control must be specified in parallel. The implementation for the platooning scenario is explained in more detail in Section 6.2.

Another feasible solution approach is to forward the evaluation results of the individual DSC-Modules to the regular control in order to trigger behavior adaptations there. As additional input data for the individual behavior generation, this could enable more sophisticated autonomous driving behavior on the one hand. On the other hand, the reliability of the resulting control layer could be decreased due to more complex runtime safety considerations. The main difference would be that the separation of the DSC adaptation level to the regular control level would be dissolved. The changes in the safety concept as well as the driving behavior derivation would thus become more complex, since the DSC evaluation originally relies on the targeted adaptation of certain driving parameters in a networked mode and not on complete driving behavior. Only for emergency driving maneuvers, if required, the original DSCs networking approach considers significant changes in the existing driving behavior. Beyond that, while the standardized definition of modules could be maintained for the DSC engineering process, the unified DSC behavior adaptation level would no longer be applicable. As a result, more customized solution concepts for the behavior generation of individual networked systems would be required, which was not in the focus of this thesis. However, the modular DSC evaluation approach without the DSC adaptation level could be part of the future research.

6.1.2 Use Case: Intelligent automated parking

By specifying the networked driving mode for the IAP use case based on DSCs, the suitability of the conceptual modeling framework as well as the design guidelines could be demonstrated. While the DSC modeling conventions were defined in parallel to the evolving platooning use case, they were applied in this use case for the first time without further adjustments. Starting from the overall safety goal to avoid collision in a parking garage, the relevant networked safety features could be derived in a systematic manner. The networked driving functionality in the parking garage could thus be improved based on the presented solution approach according to safety and performance criteria.

In this scenario, it was also possible to show the reusability of parts of the DSC specification for other system platforms and scenarios. Due to the modular specification, certain DSC-Modules, such as the *Speed Check* module from the platooning use case, could be reused. One of the outcomes for reusing parts of the DSC specification is that it makes sense to differentiate DSC-Modules for the engineering process into those that are specifically tailored for a certain system platform or an application scenario and those that are rather generic. The *Speed Check* module, for example, will probably use a similar service concept on the various system platforms and is suitable for many networked driving applications. It can thus be reused in many ways for different system platforms and scenarios. In contrast, the DSC-Module for the remaining permitted driving distance in a parking garage, for example, is highly customized for the IAP use case. Basically, it can be reused to consider other automotive system platforms for the IAP use case, but not for other application scenarios. Other DSC-Modules, such as the module for networked environmental perception or the top-level module for triggering specific driving maneuvers, depend strongly on the individual capabilities of the respective system platforms. They can potentially be reused for the same type of automotive system platforms in other networked applications, but not for other system platforms in the same application scenario. In principle, reusability is not limited to the module level, but can also take place on a subordinate level of the individual DSCs as well as on a superordinate level with several interconnected DSC-Modules. A vivid example for this are the specified DSCs for counterwise vehicle detection in the presented platooning use case and the joint obstacle detection in the IAP use case. While the overall DSC-Modules differ, some central DSC runtime proofs are identical. Here the modularization helps to identify the relevant DSCs from the overall context. As presented in Chapter 5, for a suitable reuse of parts of the DSC specification in other networked applications, the DSC-Modules have to be categorized accordingly.

6.2 Technical Feasibility of the Runtime Mechanisms

In this Section, the validation activities and their outcomes for the specified DSC runtime mechanisms from Chapter 3 and 4 are presented. The corresponding validation activities are subdivided into three main sections. In a first step, the findings from the analysis of the various DSC evaluation strategies for a cyclic overall safety assessment are explained. This is done with respect to their processing times and evaluation efficiency. Then, the operationalization of the DSC negotiation mechanism for identifying suitable networked modes is clarified based on a prototype implementation. Finally, the extensive simulations to ensure the adequacy of the networked system behavior using the DSC approach are described and the gained experiences are summarized.

6.2.1 Adequate DSC runtime evaluation mechanisms

The feasibility of the runtime mechanisms of the DSC networking technique is a key prerequisite for the implementation of the proposed solution approach. The reliability of the DSC runtime evaluation should be ensured by the choice of a simple and robust decision-making mechanism as outlined in Section 2.2.1. Furthermore, a special focus is on the total computation time of the interconnected DSC-Modules that limits the number of possible runtime checks in a specific time interval. It can thus influence the minimum possible reaction time to emerging hazards. Consequently, for the technical feasibility of the DSC approach, it is particularly important that the cyclic DSC evaluation times to interpret the available RtEs input parameters are sufficiently low. Basically, this also requires a high sampling rate to update the availability of RtEs at runtime. However, the generation times of RTEs cannot be influenced at this point, since they depend primarily on the implementation.

The specified DSC evaluation model from the GUI in Fig. 4.13 in Chapter 4 can be transformed into a computable representation in an automated manner. Basically, the DSC-Modules can be transferred into different computable representations such as the Boolean or the BDD logic. In a comparative analysis for the ConSerts approach [55], in which a similar qualitative specification for a module evaluation is used, it was shown that the BDD analysis can be calculated more efficiently. The reason for this is that BDDs represent a compressed representation of Boolean functions. The quantitative parts of the DSC specification such as adaptive thresholds or correlations for analysis can also be transformed into simple and robust boolean threshold checks as described in Section 4.4. If decisions are to be combined across several modules, however, the quantitative module interfaces must be considered differently. The specified quantitative DSC-Modules are designed to forward numerical values at their outgoing ports. In consequence, it is not possible to transform the overall DSC evaluation model into a compressed BDD representations. Only decisions up to quantitative value propagation interfaces (the outputs of quantitative DSC-Modules) can be aggregated. The reason for this is that these relevant parameters can only be evaluated in a hierarchical manner at runtime and can vary continuously during operation.

It becomes apparent that the modular character of the DSCs approach can be resolved for some types of runtime evaluations. Hence, the main focus, in addition to introducing a fundamental runtime representation of the modular DSCs, is on the overall evaluation strategy for the interconnected DSC-Modules with respect to the behavior adaptation mechanisms. For the overall DSC networking approach, two main evaluation strategies were identified. This is the module-oriented bottom-up DSC evaluation and the adaptation behavior-oriented DSC evaluation. While the module-oriented evaluation focuses on the consistent hierarchical propagation of safety quality attributes among networked safety features, the behavior-oriented evaluation concentrates on the validity of output behavior and removes the module orientation. Both evaluation methods have advantages and disadvantages, which are discussed in the following.

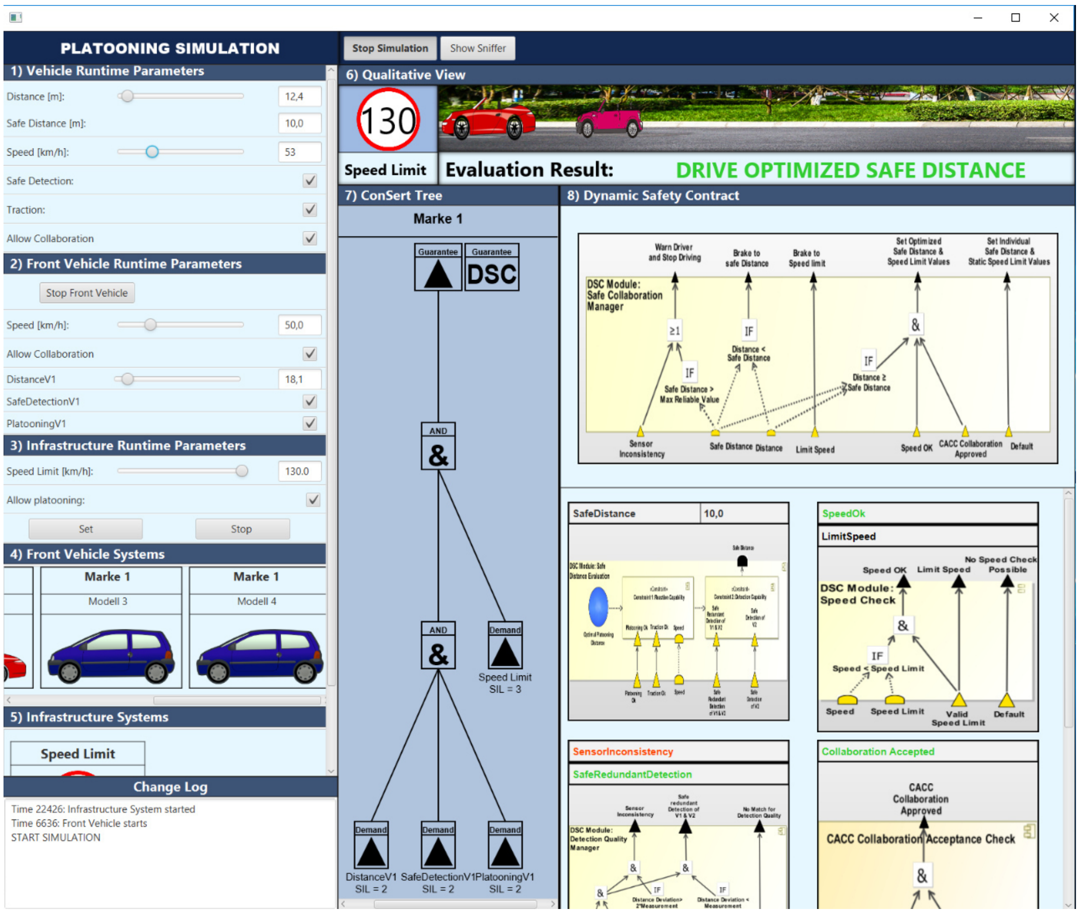
For the module-oriented evaluation approach, generally all DSC-Modules have to be evaluated before an adaptation behavior is triggered. However, especially for triggering a particularly time-sensitive adaptation behavior, it is sometimes not necessary to take all accessible information into account and to evaluate all DSC-Modules. Particularly critical and targeted adaptation behavior are typically focused on a specific unsafe driving state and the corresponding safety-critical features. By evaluating the additional DSC-Modules that are not required in this context, valuable time can pass before the intended adaptation behavior is triggered.

By applying the adaptation behavior-oriented DSC evaluation, this time delay can be prevented for the respective time-sensitive adaptation behavior. After the DSC networking mode is specified at development time, the respective conditions for triggering a particular adaptation behavior are traced back and then aggregated for a targeted runtime check. In this way, only the relevant DSC decisions are considered for a prioritized evaluation of a certain adaptation behavior. Applying this evaluation approach, the modular orientation is only available for development time engineering activities and is resolved for the runtime evaluation. The resulting drawback is that the average evaluation time for the whole set of specified adaptation behavior is higher. The reason for this is that certain input conditions, due to the missing module orientation, have to evaluated repeatedly. This is the case when specific input demands are important for several different adaptation behavior at runtime. Here, in contrast, the module-oriented evaluation has the advantage that no conditions have to be checked repeatedly. The top-level DSC guarantees are evaluated uniformly once and can be accessed several times in a single evaluation cycle. In particular, if many adaptation behaviors are specified, the avoidance of repeated evaluations can significantly reduce the average evaluation times.

It becomes apparent that in some scenarios a hybrid evaluation approach might be beneficial. Particularly time-sensitive adaptation behavior can be evaluated with the targeted adaptation behavior-oriented DSC evaluation, while the remaining ones can be evaluated with the more efficient and systematic module-oriented DSC evaluation. The described effects were analyzed with a simplified simulation environment as part of a student research project. However, since the differences in the overall evaluation times are very small and also depend heavily on the structure of the specified DSC networks, it does not seem sensible at this point to provide analysis results in the form of statistical measurement data. To further validate the described runtime mechanisms in future, standardized DSC evaluation networks have to be simulated on real ECUs.

6.2.2 DSC negotiation mechanism for networking systems at integration time

Whenever systems come within the communication range of each other, they should establish flexible networks for collaboration. To enable this, the respective systems must be open to previously unknown systems, as outlined in Section 3.6. For this purpose,



Distance [m]: 12,4
Safe Distance [m]: 10,0
Speed [km/h]: 53
Safe Detection: ✓
Traction: ✓
Allow Collaboration ✓

Speed [km/h]: 50,0
Allow Collaboration ✓
DistanceV1 18,1
SafeDetectionV1 ✓
PlatooningV1 ✓

Speed Limit [km/h]: 130.0
Allow platooning: ✓

Evaluation Result: DRIVE OPTIMIZED SAFE DISTANCE

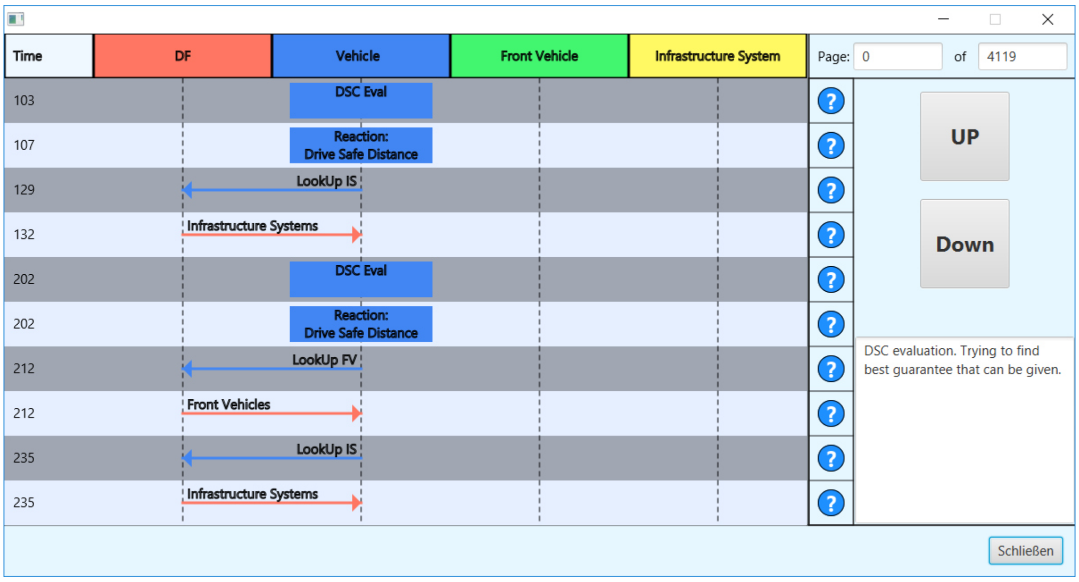
```
Time 22426: Infrastructure System started
Time 6636: Front Vehicle starts
START SIMULATION
```

Figure 6.2: Platooning Simulation Framework

a negotiation process seems appropriate to mediate between the respective system capabilities. It could determine an optimal networking of the involved systems and their functionalities. As outlined in Section 3.6, the negotiation process is performed at integration time of systems. When a networked system mode is identified and activated, the actual operating time begins. Only when the active networked system mode is deactivated, which is only possible in a safe operating state, a new integration time for the networking of systems can be started. Since the required mechanism at integration time of systems is not time-sensitive, the focus for the validation was on the feasibility of the negotiation mechanism in the DSCs application context.

Based on the Jade simulation framework, an illustrative negotiation process was implemented by Daniel Hillen for the platooning use case. In this platooning simulation, the initialization as well as the operation of the networked driving behavior is analyzed for the following vehicle according to the platooning use case description. For the application scenario it was assumed that two previously unknown vehicles on a highway get in contact to each other and establish a wireless communication network. For this purpose, vehicles with different networking capabilities can be flexibly added to the road simulation enviroment via drag and drop. In addition, the infrastructure can be added as a third networking partner that can impose a variable speed limit.

As soon as two vehicles are inserted to the road simulation environment, the described negotiation process starts. In a first step, based on the respective networking capabilities of the involved systems, an optimally integrated networked driving mode is evaluated. During this integration phase, the two vehicles are still in an individual driving mode. When an optimal networking mode has been identified, it can be activated then. During operation of a networked driving mode, it is possible to selectively manipulate specific driving or environmental parameters and to observe the correctness of the resulting networked driving behavior. The respective output guarantees of the DSC-Module ports are made visible with animated graphics. The adequacy of the resulting networked driving behavior can thus be checked in a qualitative manner. Moreover, this simulation framework makes it possible to graphically display the communication between the participants during the negotiation process at integration time as well as at operation time as shown in Fig. 6.3. To validate the negotiation mechanism, various vehicle combinations with divers networking capabilities were simulated. The feasibility of the planned communication concept in the integration phase of systems could thus be demonstrated. Hence, openness for networked DSC applications can basically be achieved wherever required.



DSC evaluation. Trying to find best guarantee that can be given.

Figure 6.3: Graphical Visualization of Networked DSC Mode Communication

6.2.3 Dynamic adaptation behavior during operation of a networked DSC mode

While the previous Section showed the feasibility of openness to previously unknown systems for the DSCs approach, this Section focuses on the resulting networked driving behavior during operation of a networked DSC mode. In order to validate the feasibility and adequacy of the specified dynamic adaptation behavior based on DSCs, a comprehensive virtual simulation framework was established. This test environment makes it possible to simulate the driving behavior of networked vehicles in a very realistic manner.

To set up a suitable test bed, Patrick Wolf from the RRLAB at TU Kaiserslautern implemented a detailed physical model of so-called *Gator* test vehicles, including realistic driving dynamics, first in the V-REP (Virtual Robot Experimentation Platform) simulation environment and later in the Unreal Engine simulation environment as shown in Fig. 6.1. Since more realistic environmental scenarios could be created with the Unreal Engine and thus more meaningful results could be achieved, the later validation activities were only carried out with the Unreal Engine. In parallel, the vehicle control for the regular autonomous driving behavior as well as the networked DSC driving behavior was implemented with the real-time robot control framework Finroc. This framework was developed at the RRLAB and supports the handling of large control networks. Finroc can be connected to the simulation environment via predefined in- and output ports. In this way, by combining the simulation environment and the robot control framework, a closed loop control for real-time sensor/actor data processing could be realized.

For the simulation of the specified networked driving behavior using the DSCs approach, a master-slave vehicle platoon based on two *Gator* vehicles was implemented in this test bed. The basic working principle of the networked driving mode in the platooning scenario can be looked up in Fig. 3.1 of Section 3.2. In contrast to the platooning scenario for public road traffic presented there, however, the simulated master-slave platooning basically aims at a closer system coupling. This is particularly useful for networking fleets in the off-road sector, where the focus is on common goals for coordinated driving processes. Here, an additional safety layer can be defined, such as driving authorizations for the slave vehicles that are dependent on the master vehicle. Thus, autonomously driving slave vehicles can be guided and supervised by a master vehicle with a human driver.

In the simulated scenario, the master vehicle is controlled with a virtual control panel. A single slave vehicle follows the master vehicle in an autonomous way based on the so-called iB2C (integrated behavior based control) control logic in a highly unstructured environment. In this way, it was possible to systematically cause critical driving situations for the networked DSC driving mode and to observe the resulting reaction behavior.

In the networked driving mode, the iB2C and the DSC control logic are evaluated in parallel for the slave vehicle as illustrated in Fig. 6.4. While the iB2C control provides the regular robust autonomous driving behavior, the DSC control logic ensures the adherence of the currently permitted degrees of freedom in the networked driving mode and triggers adaptation behavior if necessary. The DSCs are evaluated inside the Finroc framework based on the available RtEs. The availability of certain networked safety features based on the DSC-Modules can then be checked with a runtime visualization. The DSC adaptation behavior are connected to the iB2C control logic in the Finroc control framework as shown in Fig. 6.5. The correctness of the specified networked driving behavior can then be analyzed based on the simulated driving sequences in the test framework and based on the signal processing in the Finroc control framework.

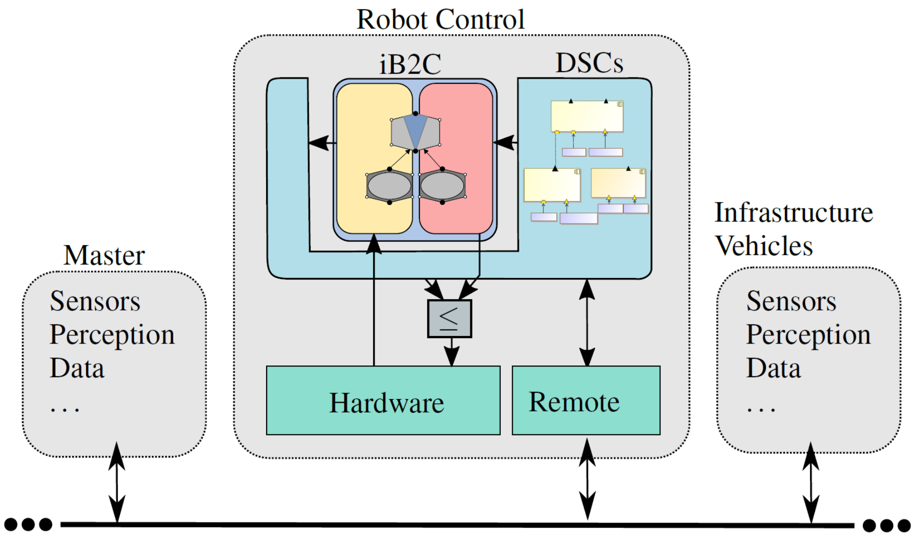


Figure 6.4: Service Types for Gator Cooperation [84]

In order to validate the feasibility and adequacy of the DSCs solution approach, a central objective was to simulate the identified hazardous situations that are stored in the SDM for the networked driving mode. These hazardous networked driving situations should be prevented by the specified DSC adaptation behavior. In the test environment, the correct handling of this driving situations in accordance to the specified dynamic DSC adaptation behavior could be shown. Also the DSC compatibility with a real-time control framework could be demonstrated in this way. Due to the many variants of possible driving states for a hazardous situation, however, the focus was preferably on the threshold conditions for triggering a certain adaptation behavior. To assess even more driving states for checking the reliability of the networked driving behavior, automated tests based on randomly generated driving sequences can also be performed in future.

Naturally, it should be noted that the simulation environment only depicts a simplified model of the real world and thus does not fully reflect the real driving behavior. To ensure safety for networked DSC driving processes, the reliability of the specified networked driving behavior also has to be validated with real world applications. A vivid example for this is the tire grip and its influence on the deceleration capabilities,

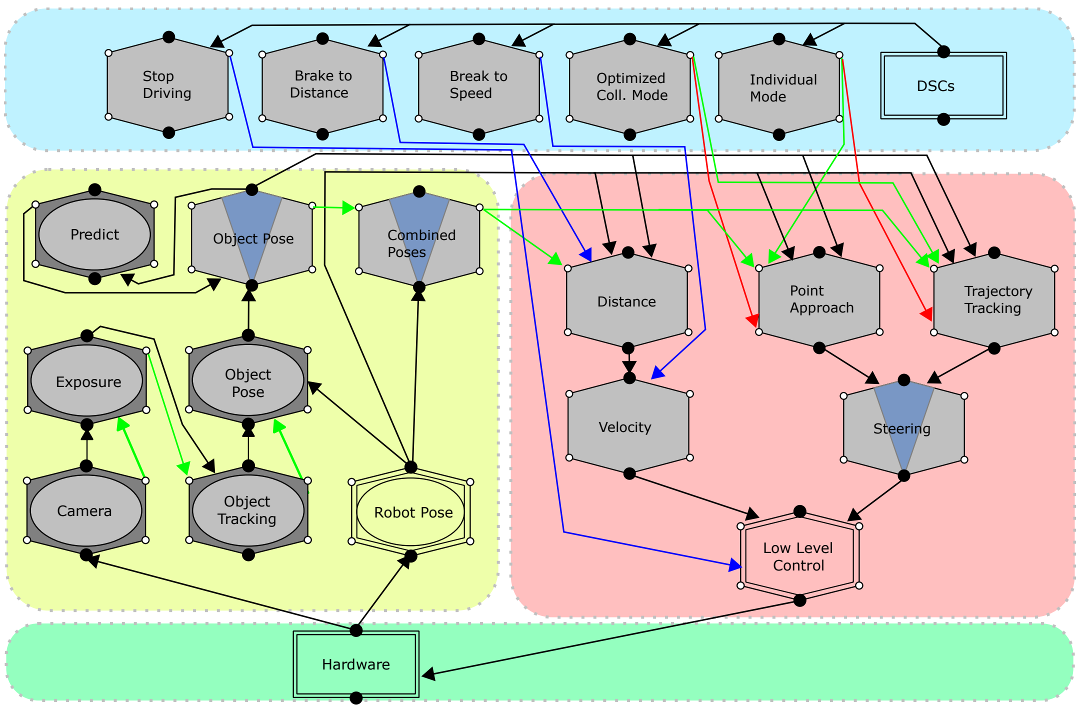


Figure 6.5: Integrating the DSC Adaptation Behavior to the iB2C Control [84]

which cannot be simulated in accordance to the real driving dynamics. However, with the simulation framework the correct relative interaction between the simulated vehicles according to the specified networking mode could be demonstrated. In principle, there are no restrictions for the DSCs approach to take more complex environmental factors into account.

Beyond that, the presented simulation framework also provides a good basis for the optimization of already specified DSC applications according to safety and performance criteria. The modularized DSCs specification is particularly well suited for iterative optimization. Identified hazardous situations in the simulation of the specified networked DSC driving mode can be considered in the SDM and the DSC specification can be optimized accordingly. Chapter 5 provides guidance for this purpose. In this way, the DSC specification can be better prepared for the subsequent step of real world testing.

6.3 Validation Summary

In summary, it can be concluded that the validation activities in this Chapter did not encounter any difficulties that could prevent a practical application, neither for the required development time parts nor for the runtime parts of the DSCs approach. With the help of the IAP use case it could be shown that the original driving functionality based on a simple “Go” signal could be systematically extended in a variety of beneficial ways.

Accordingly, the conceptual framework of the DSCs approach as well as the derived engineering steps seem to be well suited for the specification of the dynamic adaptation behavior of embedded systems in networked applications. The main focus of the validation activities for the DSC runtime parts was on demonstrating practical solutions to the existing challenges, such as the networking of a variety of heterogeneous systems in the Platooning and in the IAP use case. In addition, using simulation environments that are as realistic as possible, such as in the platooning scenario based on the Unreal Engine and Finroc, should provide close-to-reality findings for the feasibility of the networked driving behavior. The generation of RtEs based on environmental information and the initiation of targeted behavior adaptations could be thoroughly observed and validated.

In the next step, the practical implementation on a real ECU basis should be a priority for the further elaboration of this approach. In addition, the implementation within a wireless network with a low latency such as the new 5G standard should be investigated. Ultimately, the effective worst case execution time for a DSC evaluation cycle can be determined in this way. It determines the minimal possible reaction time to emerging hazards for the DSCs approach and thus the achievable dynamics for networked behavior adaptations. In addition, since the presented validation activities are limited to the vehicle domain, the applicability in other feasible domains should be analyzed as well. As described in the introductory Chapter, this can be Industry 4.0 applications such as networked co-working robot arms or modular networked manufacturing systems with decentralized sensors. In principle, it also seems promising, although this is not part of the target application domain, to validate the applicability for future data processing systems with cloud or multicore based solutions using concurrent task processing in safety-critical applications.

7 Conclusions and Future Research Agenda

In this Chapter, first the contributions of the proposed DSC approach to safety assurance and certification are summarized. For this purpose, the fulfilled research goals that were derived in the introductory Chapter are explained and limitations are pointed out. Afterwards, Section 7.2 highlights future research topics for the presented solution approach and Section 7.3 gives an outlook for its practical application.

7.1 Scientific Contributions and Limitations

DSCs constitute a new approach for merging decentralized safety-related runtime knowledge in order to derive safety-certified networked system behavior from it. It is expected that future embedded systems are capable of establishing emergent collaboration processes in flexible SoS configurations. As outlined in Chapter 2, there are at the moment from best of our knowledge no comparable approaches available that tackle the specific demands of runtime safety certification for the dynamic adaptation behavior of networked embedded systems.

As it became apparent in this thesis, a paradigm change is necessary in the safety community to enable such promising applications. Today's state-of-the-practice safety assurance and certification techniques are limited to development time safety means. Instead, it was stated in Chapter 2 that suitable runtime safety means are required for this purpose. With the ConSerts approach, Schneider recently presented a promising strategy for introducing runtime certification means aimed at more flexible systems configurations. Here, the runtime safety responsibility of systems is kept at a minimum by shifting only the necessary parts of the assurance means to runtime. In this way, a sufficient adaptability of systems during their integration could be enabled to establish safety-certified configurations. In comparison, the proposed DSC approach aims primarily at dynamic changes in the application context when the system configurations are already established. Based on fine-grained runtime checks, DSCs provide adequate means for enabling targeted dynamic adaptation processes in safety-critical application environments. The DSC approach thus represents a further step in the development of the state-of-the-art in this research area.

For the evolving DSC approach, the research challenges identified in the introductory

Chapter were all taken into account accordingly. The key driver for the presented solution approach was to optimize system behavior in safety-critical applications by integrating external, shared services. Here, the focus was on networked driving processes in the automotive domain. But, as described throughout this thesis, other domains such as the industrial CPS domain seem promising as well.

For practical implementation, it was pointed out that safety certifiability is the most important prerequisite to provide a sufficient runtime variability of systems during operation. Hence, safety and certification issues were decisive for the definition of the proposed approach. It was tried to keep most of the required safety assurance and certifications means at development time. Consequently, DSCs focus on lightweight run-time checks to reliably assess the validity of a precertified system behavior in a networked mode. The actual safety certification of the entire networked system behavior for the DSC approach can already take place at development time. This can be done conditionally with respect to the available runtime safety guarantees during operation. In this way, DSCs could provide adequate means for runtime safety certification of the networked system behavior.

Naturally, it is difficult to estimate the future acceptance of the proposed runtime certification approach by the safety community and certification authorities. Besides specific implementation issues that are addressed in Chapter 7.3, it also depends on many other technical and social trends. For instance, it can be assumed that the trend towards autonomous driving, as described in Chapter 2.1.1, has an important influence on the proposed approach. Here, the safety certification of complex autonomous driving functions is carried out on the basis of environmental recognition algorithms. On the one hand, this would provide the foundation for introducing networked driving processes based on fully autonomous vehicles. And, on the other hand, it would also contribute to the described paradigm shift in the safety community towards more runtime variability of systems in open and safety-critical application environments. Such a development trend could thus pave the way for the envisaged networked driving processes of autonomous vehicles. As in the DSCs approach, it appears feasible to implement this on the basis of a rigorous monitoring of networked safety features.

Another central research issue was the provision of a systematic integration concept for shared services on a particular system platform according to safety and performance criteria. This thesis provides a holistic modeling framework for specifying the networked system behavior based on preconfigured variability. As already described, the DSC approach focuses on the influence of specific external services on a particular system platform rather than on the networked system group as a whole. It enables the modular specification of networked safety features in a systematic way based on available safety-related services. Here, modularization enables to divide complex safety dependencies into manageable ones. Utilizing a demand-guarantee dependency modeling it was possible to specify a reliable runtime merging concept for the shared services on a particular system platform.

In addition, it was concluded in the introductory Chapter that a specification concept for networked system modes requires suitable degradation steps in order to achieve the best possible system limitations according to safety and performance criteria. To make this possible, a major distinction for modeling networked safety features was done into a qualitative and quantitative degradation concepts. For this purpose different merging logics as well as different module and port types were specified. In this way, a more fine-grained downgrading of specific behavioral aspects of the networked systems could be achieved.

Beyond that, it was recognized that the specified dynamic behavior in a networked system mode requires appropriate reaction behavior in the event of sudden hazards. In networked driving processes of vehicles, for instance, adequate emergency driving maneuvers are required. Here, the more fine-grained monitoring of networked features using qualitative as well as quantitative representations also enables a more precise situation assessment of networked systems at runtime. This in turn was utilized for the DSC approach to provide a more targeted triggering of reaction behavior such as emergency driving maneuvers.

In addition to these requirements, a further aspect was the reusability of parts of the existing specifications for other networked applications as well as the standardization of shared services among networked systems. Applying the modular DSC specification, the reusability of specified networked safety features could be demonstrated in several use cases. The modular DSCs can be categorized, stored and reused according to their specified safety features as well target application scenarios. In order to support efficient engineering processes of networked applications, extensive databases with already specified modular DSCs could be set up for this purpose. In addition, it could be shown that the contract-based safety modeling based on consistent service types also supports the harmonization of service provision concepts.

To show the feasibility of deriving networked system behavior with the DSCs approach, this thesis also presents a dedicated DSC-Module for specifying the resulting system behavior based on the available runtime guarantees. Nevertheless, the core concept of the DSC approach is limited to merging decentralized safety knowledge of networked applications in the modularized DSC safety features. Using a straightforward reaction-behavior like the presented one with a dominant influence to the regular control is a good starting point for observing the correct operation of the DSC evaluation mechanism and the reaction behavior in practical tests. Moreover, it enables the definition of a traceable preconfigured variability and is thus particularly suitable for certifying safety of networked system behavior.

At the same time, however, in addition to safety, also performance criteria are central to achieve an optimal integration of the adapted DSC behavior into the regular non-networked system behavior. Depending on the application, a closer integration of the networked DSC system behavior into the regular can be beneficial from a performance point of view. Generally, however, the autonomous system must be able to properly associate the specified networked safety features with other individual environmental factors for behavior generation. Here, the focus would no longer be on the individual RtE for adapting the networked system behavior, but on the resulting overall configuration. The impact of individual RTE's on the system behavior would then become more difficult to track in the various application scenarios. Ultimately, this would also make certification activities more complex and thus more difficult to achieve.

A major limitation of the DSC approach, as has become apparent in this thesis, is that the benefits of DSC networks up to this state of research are limited to single systems. Individual systems gain advantages through networked DSC applications with other surrounding systems to solve upcoming tasks more effectively. Here the monitored DSC safety features are utilized to optimize safety and performance of networked systems without restricting the individual autonomous driving features too much. Optimization of the overall group behavior of networked systems can thus only be achieved indirectly. For example, the entire road traffic can benefit from improved driving maneuvers of the individual vehicles. However, it is not possible to control the vehicles in a centralized manner with precise group driving maneuvers. Considering larger groups of networked systems, a direct coordination of the networked applications from a superordinate control level can be beneficial. In such cases the monitored networked safety features could be the basis for deriving the overall coordination of the networked group of systems. It has to be clarified then which system platform evaluates the networked safety features, how the evaluation results are interpreted and shared among the involved systems, and who is responsible in case of damages. However, the definition of a group behavior of networked systems would also require more customized solutions for specifying the interaction processes of involved systems in specific application scenarios. It was therefore not in the scope of this thesis.

A rather general limitation for the definition of networked DSC modes is that it can be difficult in the addressed open context to optimally specify DSCs only with design time knowledge according to safety and performance criteria. The open application context requires extensive testing in simulations as well as in real-world applications. With increasing experience based on tests, different variants can then be examined and the networked DSC application as well as the SDM model can be optimized accordingly.

7.2 Future Research Agenda

With the aim of putting the presented DSC approach into practice, important future research topics are outlined in the following. A special emphasis is on the vehicle domain as the target application domain. First, the focus for future research is on methodical aspects of the approach, then on specific implementation aspects and then on the resulting practical tests of real networked applications.

Especially, the optimal integration of the networked DSC-based system behavior into the individual system platforms should be part of future research. This thesis presents a solution concept to dominantly influence the non-networked control architecture. However, although this is specified with a clearly separable two-layer architecture, interfaces for influencing the existing control architecture must still be defined in order to reliably incorporate the DSC evaluation results. An alternative possibility, as already mentioned in the previous Section, would be to forward only the merged DSC guarantees to the behavior generation level. Here, they could be evaluated together with other individual findings of environmental perception to derive an optimal autonomous driving behavior. While the presented variant appears to be favorable for certification issues due to its clear decision-making structures, the second variant can be better suited from a performance point of view, but is more unsystematic in its specification. Ultimately, in both variants the correct interpretation of DSC runtime guarantees in combination with other individual environmental perception factors has to be validated in order to derive a safe autonomous driving behavior. This requires extensive practical testing and should be part of future research.

Similar to the design decision for the optimal integration of the DSC evaluation into the existing control architecture, the question arises which granularity for the shared RTEs is appropriate for networking systems based on DSCs. In the presented platooning scenario and the IAP scenario, the vehicle communication is focused on RtEs using basic sensor or sensor fusion data such as distance measurements and acknowledgments for sensor recognition. In addition to these, however, also more complex RtEs such as *Platooning OK* were utilized, which consist of several sub-decisions and were specially introduced for this networked application. They are hardly applicable in other networked DSC applications. Given an open application context such as public road transport, it seems preferable to focus on less complex RtEs to ensure that this data can be interpreted by other systems. Generally, thus more types of RtEs are available that can potentially be shared and linked within DSC-Modules. In addition, forwarding RTEs at a lower level would also allow a higher re-use rate of networked DSC-Modules for other similar networked applications. On the other hand, it seems sensible to limit the DSC runtime responsibility for merging RtEs and to condense the contained safety-relevant data of RtEs as far as possible before forwarding them. Each additionally required RTE in a particular networked mode can probably increase the risk of a forwarding or evaluation error. This can ultimately cause additional hazards for the networked application in case of its unavailability. It becomes apparent, that the optimal granularity of shared RtEs depends on many aspects such as harmonization of service provision concepts as well as safety consideration and should be part of future research.

Besides that, for networked DSC applications only a maximum of five interconnected DSC-Modules were considered in this thesis. For the introduction of more complex DSC evaluations with more DSC-Modules, additional research is required to analyze the resulting modeling complexity and evaluation performance. However, since the DSC evaluation is time-sensitive and varies depending on the number of DSC-Modules, it becomes already visible that a holistic integration test on the running system platform is necessary.

In case several networked systems run DSC applications in parallel and influence each other, it will be necessary to coordinate them appropriately. In the automotive domain, this can be the case when a suitable group behavior of networked vehicles in a specific driving context is required. Concrete examples are the presented Platooning use case or the IAP use case when networked driving processes of several vehicles have to be coordinated at once. Here, a more centralized coordination should be part of the individual DSC networks that can centrally supervise and manage the networked systems by the provision of dedicated RtEs. For this purpose appropriate means have to be introduced, which enable an overall decision making for a group of networked systems, a coordination of feasible DSC networks in a particular context and a surveillance of the individual networked systems for a correct group interaction.

For practical implementation, it also seems reasonable that the continuous runtime evaluation of DSCs is ensured by an additional monitoring functionality. In case of an interrupted DSC evaluation, the system could thus automatically switch to a safe mode. An appropriate solution would be to introduce an additional RtE which guarantees the safe operation according to predefined timing constraints. It could monitor the correct operation with a watch dog function and in case of an error suspend the current networked mode in a predefined way. As part of integration tests on the respective system platforms, the implementation of such monitoring functionalities should be part of future research to ensure the reliability of the specified DSC evaluation mechanisms in real applications.

In addition, the reliable generation of RtEs in the respective application domain is an important future research topic as it is the groundwork for generating the networked system behavior based on DSCs. Besides the generation of RtEs based on internal system states, especially the RtE generation based on external environment data can be error-prone. With DSCs themselves, it is only possible to compare similar RtEs for logical conformance and thus achieve a redundant assessment for a higher level of safety integrity. The validity of a single shared RtE based on its generation, however, cannot be subsequently validated. Comparable to the environmental perception capabilities of autonomous vehicles, similar evaluation criteria must be established for the generation of RtEs. As part of the practical implementation, the reliable RtE generation should also be part of the future research.

Furthermore, security issues are a crucial factor for the safe networking of systems in safety-critical applications. For instance, hackers could try to manipulate the wireless communication link between vehicles. Since security was not within the scope of this thesis, it should be part of future research. Basically, however, the security concepts available today, such as the protection of communication through encryption and the authentication of communication partners, appear to be suitable for this purpose. A key prerequisite is that the respective networked applications are thoroughly examined for potential security weaknesses. This includes the generation of shared RtEs which could be manipulated as well as the overall trustworthiness of potential partners in the surrounding environment. As a first solution idea, hacker attacks could be tackled with a continuous system self-check on each platform. Thus, when forwarding RtEs, it could be ensured that the networked systems are in a good working condition and have detected no security threats. The trustworthiness of the surrounding systems could in turn be verified by a cloud-based system rating derived from past experience in networked applications.

The adherence to time constraints is also crucial for the practical implementation of networked DSC applications, as the focus is on highly dynamic tasks. Here, in such safety-critical applications, especially the worst-case execution time is decisive. It specifies the time interval in a networked DSC application from a missing RtE to the response of the system. As it became apparent in Chapter 6 the pure DSC evaluation times are sufficiently low. Hence, it strongly depends on the concrete implementation. It is mainly dependent on three factors, which are the update time of the RtEs, the transmission time between the systems and the evaluation time using DSCs. Especially with additional computing power, sensor measurements can be analyzed in shorter update intervals and data can be sent and evaluated faster. Hence, these three factors can be largely customized by selecting the appropriate computing power and wireless communication technology for a specific system configuration. It can be assumed that future autonomous vehicles will be sufficiently equipped to enable both. It is therefore particularly important for future research that these factors are properly aligned in order to enable a suitable real-time behavior for networked applications. Naturally, the refresh rate of the RtEs should therefore be as short as possible for further evaluation. Moreover, in practical tests the reliability of continuous data transfer has to be validated.

Beyond that, a central future research topic will be the implementation of the DSC networking technique in real world applications. Generally, first practical tests should be carried out with additional safety measures. The networked driving behavior could be limited to a slow driving mode in the initial test phase and the potential cooperation partners could be fixed at development time. The presented Master-Slave vehicle configuration seems to be especially suited for this purpose since additional safety constraints can be introduced for the autonomously driving slave vehicle. As a test site, a restricted area in the off-road area with a small number of exposed participants is suitable, e.g. fields in the agricultural domain. Since modeling and simulation of the networked driving processes was mainly carried out with the physically available *Gator* vehicles at the RRLAB, these vehicles would be perfectly suited for first practical tests. Nevertheless, such practical tests of the networked DSC driving applications require adequate test sites and the clarification of legal issues in advance.

Later on such test scenarios could be extended to practical application scenarios in the off-road domain such as harvester scenarios or milling machines with less complex driving tasks. For practical testing, road traffic applications are basically the last step in which the overall safety concept of networked driving must already be very mature. Here, the focus for networked autonomous driving will be on certification issues for compliance with regulatory issues.

7.3 Forward-looking Statement

In the following, an outlook is given on the practical application of the proposed DSC approach using runtime certification means. For this purpose, the current development trends in research and industry as well as specific implementation aspects were taken into account. A special focus is again on the automotive domain as the target application domain. Although it is difficult to provide any definitive statements for its future application, research on this topic has shown that the demand for such flexible and at the same time safety-certifiable networking options is constantly growing.

Especially, the increasing use of embedded computing devices, which have become increasingly powerful and cost-efficient in recent years, will be an important catalyst for the broad networking of previously separate system functions. Today, corresponding research efforts are particularly visible in the industrial CPS domain with innovative Industry 4.0 applications. Many of these networked applications will be performed in a safety-critical context. Considering complex manufacturing steps such as human-machine interaction processes, appropriate safety assurance and certification techniques are required for this purpose. In principle, DSCs offer a suitable solution concept for this by reliably accessing and interpreting safety-relevant data from the networked sensors and production systems. However, concrete DSC use cases still have to be developed for the industrial CPS domain.

In parallel, new communication standards such as 5G with low latency are evolving. The related communication technologies are especially suitable for networking systems in mobile applications such as in the automotive sector. Particularly in the automotive sector, as described in the Chapter 2, there is a great potential to make road traffic more safe and efficient by networking vehicles and infrastructure. It can be assumed that the combination of future autonomous vehicles with powerful on-board ECUs and the next-generation communication technology will enable data exchange according to the necessary real-time requirements for DSC evaluation. Consequently, from a technical point of view, the practical implementation of DSC-based networked driving processes depends on the introduction of the aforementioned technologies.

Furthermore, the interoperability between all potential networking partners in the various application domains will be a central concern to avoid a separation between different technical platforms that will later become incompatible. Hence, the specification of suitable standardized service provision concept will be an essential preliminary step for the broad networking of systems. In order to establish networked safety-critical applications, it will also be necessary to provide an appropriate formalization of the shared safety-relevant data to make it accessible for other networked systems. This is particularly relevant for safety-critical networked applications such as those in the automotive sector. With an increasing number of spatially distributed systems that can flexibly establish wireless networks, also the need for standardized concepts for sharing safety-relevant date will get more attention by research and industry. The presented DSC approach provides a first solution concept for this purpose.

Besides technical implementation issues, it is difficult to predict the acceptance of the presented networking approach by authorities in the various application domains. For instance, in the automotive sector, the future regulations for autonomous driving are still being developed. One reason for this is that today's autonomous vehicles are not yet mature enough for fully autonomous driving and the future capabilities of autonomous vehicles are difficult to predict. Besides that also the existing national regulations for using autopilot functions with a lower automation level are very different. While in some countries such techniques are regulated in a very restrictive way, in others there are fewer rules to encourage the development of such innovative techniques. In addition, ethical questions as well as liability questions are not yet sufficiently addressed by society and public authorities.

With regard to liability issues, it seems possible to treat DSC-based networked driving processes in a similar way to today's non-networked piloted driving functions. Here, the respective vehicles using autopilot function are responsible for their individual driving maneuvers [89]. Whether this responsibility is ultimately transferred to the driver or the vehicle manufacturer is not considered here. Also the DSC approach is specified in such a way that the respective networked vehicle remains responsible for the correct interpretation and derivation of a safe driving behavior. A major difference to non-networked autonomous driving is of course that also external data is considered. However, the other networked systems are only responsible for the correctness of the shared safety-relevant data. They are not responsible for the resulting driving behavior of other networked vehicles. Here, the fine-grained DSC runtime proofs using RtEs also appear to be well suited to clarify liability issues of the parties involved. In case of a damage, the continuously merged safety-related data can be utilized for tracing a wrong decision-making back to its source. For this purpose, the DSC-based decisions would have to be stored for a certain period of time. It becomes apparent that the contract-based safety specification is generally beneficial for the future implementation in this respect.

A typical challenge for the introduction of new technologies, such as the networking of systems on a large scale, is that a lot of engineering effort is required without having a tangible benefit right from the beginning. This can be the case for centralized networking approaches of many heterogeneous systems. In the automotive domain, for example, defining a group driving behavior for an intersection control scenario would require that all approaching vehicles are capable of participating in this networked driving application. Here, if only one participant could not be part of the networked group or would not act according to the driving commands, the overall coordination of the networked vehicle group would probably be terminated. A real benefit would thus only arise at a later stage when this technology is supported by all systems. The introduction of such promising technologies can be delayed as a result. For the proposed DSC approach, this is not a major issue as it primarily focuses on decentralized and local networked applications. Thus, not all surrounding system platforms or all capabilities of a particular platform, such as in the automotive sector for networked driving processes, need to be part of a networked application. With regard to the aforementioned intersection control scenario, driving processes of certain vehicles could be optimized and others not. Ultimately, the proposed DSC networking approach seems to be well suited even in the early deployment phase when only a few systems support such a networking technology.

References

[1] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3:18–23, jan 2015.

[2] *Deutschlands Zukunft als Produktionsstandort sichern - Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0 – Abschlussbericht des Arbeitskreises Industrie 4.0.* acatech, 2013.

[3] Giovani Da Silveira, Denis Borenstein, and Flávio S Fogliatto. Mass customization: Literature review and research directions. *International Journal of Production Economics*, 72(1):1–13, jun 2001.

[4] Keshav Bimbraw. Autonomous cars: Past, present and future - a review of the developments in the last century, the present scenario and the expected future of autonomous vehicle technology. In *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics.* SCITEPRESS - Science and and Technology Publications, 2015.

[5] Mamta Agiwal, Abhishek Roy, and Navrati Saxena. Next generation 5g wireless networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 18(3):1617–1655, 2016.

[6] Ning Lu, Nan Cheng, Ning Zhang, Xuemin Shen, and Jon W. Mark. Connected vehicles: Solutions and challenges. *IEEE Internet of Things Journal*, 1(4):289–299, aug 2014.

[7] Saif Al-Sultan, Moath M. Al-Doori, Ali H. Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of Network and Computer Applications*, 37:380–392, jan 2014.

[8] Mario Gerla, Eun-Kyu Lee, Giovanni Pau, and Uichin Lee. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*. IEEE, mar 2014.

[9] Max Reichardt, Tobias Föhst, and Karsten Berns. Introducing finroc: A convenient real-time framework for robotics based on a systematic design approach. *Robotics Research Lab, Department of Computer Science, University of Kaiserslautern, Kaiserslautern, Germany, Technical Report*, 2012.

[10] Martin Proetzsch, Tobias Luksch, and Karsten Berns. The behaviour-based con-

trol architecture iB2c for complex robotic systems. In *Lecture Notes in Computer Science*, pages 494–497. Springer Berlin Heidelberg.

[11] Katrin Sjoberg, Peter Andres, Teodor Buburuzan, and Achim Brakemeier. Cooperative intelligent transport systems in europe: Current deployment status and outlook. *IEEE Vehicular Technology Magazine*, 12(2):89–97, jun 2017.

[12] Panagiotis Papadimitratos, A La Fortelle, Knut Evenssen, Roberto Brignolo, and Stefano Cosenza. Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE communications magazine*, 47(11):84–95, 2009.

[13] SAE Taxonomy. Definitions for terms related to on-road motor vehicle automated driving systems. Technical report, SAE International, 2014.

[14] Kathrin Zeeb, Axel Buchner, and Michael Schrauf. Is take-over time all that matters? the impact of visual-cognitive load on driver take-over quality after conditionally automated driving. *Accident Analysis & Prevention*, 92:230–239, jul 2016.

[15] Luis M. Camarinha-Matos and Hamideh Afsarmanesh. Collaborative networks. In *IFIP International Federation for Information Processing*, pages 26–40. Springer US, 2006.

[16] M. Saeki. Communication, collaboration and cooperation in software development-how should we support group work in software development? In *Proceedings 1995 Asia Pacific Software Engineering Conference*. IEEE Comput. Soc. Press, 1995.

[17] Leo Denise. Collaboration vs. c-three (cooperation, coordination, and communication). *Innovating*, 7(3):1–6, 1999.

[18] Mica R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, mar 1995.

[19] Ursula Frank, Holger Giese, Florian Klein, Oliver Oberschelp, Andreas Schmidt, Bernd Schulz, Henner Vöcking, Kathrin Witting, and Jürgen Gausemeier. Selbstoptimierende systeme des maschinenbaus. *Definitionen und Konzepte. Sonderforschungsbereich*, 614, 2004.

[20] J.O. Kephart and D.M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, jan 2003.

[21] Alexander M Meystel and James S Albus. Intelligent systems: Architecture. *Design, Control-Wiley-Interscience*, 2002.

[22] Markus Maurer, J Christian Gerdes, Barbara Lenz, Hermann Winner, et al. Autonomous driving. *Berlin, Germany: Springer Berlin Heidelberg*, 10:978–3, 2016.

[23] Todd Litman. *Autonomous vehicle implementation predictions.* Victoria Transport

Policy Institute Victoria, Canada, 2017.

[24] Rolf Isermann. *Fault-Diagnosis Applications: Model-Based Condition Monitoring Actuators, Drives, Machinery, Plants, Sensors, and Fault-tolerant Systems.* Springer Publishing Company, Incorporated, 2014.

[25] Reinhard Oppermann. From user-adaptive to context-adaptive information systems (von benutzeradaptiven zu kontextadaptiven informationssystemen). *I-com*, 4(3/2005):4–14, 2005.

[26] Kalle Kleinlützum, Werner Brockmann, and Nils Rosemann. Modellierung von anomalien in einer modularen roboter-steuerung. In *Autonome Mobile Systeme 2007*, pages 89–95. Springer Berlin Heidelberg, 2007.

[27] Mario Trapp. *Modeling the adaptation behavior of adaptive embedded systems.* PhD thesis, 2005.

[28] Nidhi Kalra and Susan M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94:182–193, dec 2016.

[29] Gordon Blair, Nelly Bencomo, and Robert B France. Models@ run. time. *Computer*, 42(10), 2009.

[30] Betty H. C. Cheng, Kerstin I. Eder, Martin Gogolla, Lars Grunske, Marin Litoiu, Hausi A. Müller, Patrizio Pelliccione, Anna Perini, Nauman A. Qureshi, Bernhard Rumpe, Daniel Schneider, Frank Trollmann, and Norha M. Villegas. Using models at runtime to address assurance for self-adaptive systems. In *Models@run.time*, pages 101–136. Springer International Publishing, 2014.

[31] B. Meyer. Applying design by contract. *Computer*, 25(10):40–51, oct 1992.

[32] Holger Giese and Matthias Tichy. Component-based hazard analysis: Optimal designs, product lines, and online-reconfiguration. In *Lecture Notes in Computer Science*, pages 156–169. Springer Berlin Heidelberg, 2006.

[33] Claudia Priesterjahn, Christian Heinzemann, Wilhelm Schafer, and Matthias Tichy. Runtime safety analysis for safe reconfiguration. In *IEEE 10th International Conference on Industrial Informatics.* IEEE, jul 2012.

[34] O.A. Rawashdeh and J.E. Lumpp. A technique for specifying dynamically reconfigurable embedded systems. In *2005 IEEE Aerospace Conference.* IEEE, 2005.

[35] Marc Förster Mario Trapp, Rasmus Adler. Runtime adaptation in safety-critical automotive systems. 2007.

[36] Rasmus Adler, Ina Schaefer, Mario Trapp, and Arnd Poetzsch-Heffter. Component-based modeling and verification of dynamic adaptation in safety-critical embedded systems. *ACM Transactions on Embedded Computing Systems*, 10(2):1–39, dec

2010.

[37] IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems, Part 1-7, 2011.

[38] Ron Bell. Introduction to iec 61508. In *Proceedings of the 10th Australian workshop on Safety critical systems and software-Volume 55*, pages 3–12. Australian Computer Society, Inc., 2006.

[39] DO-178B: Software Consideration in Airborne Systems and Equipment Certification, 1992.

[40] ISO/CD26262: Road vehicles, Functional Safety Part 1-10, 2011.

[41] Vrizlynn L.L. Thing and Jiaxi Wu. Autonomous vehicle security: A taxonomy of attacks and defences. In *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, dec 2016.

[42] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, Tadayoshi Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*, volume 4. San Francisco, 2011.

[43] Simon Furst and Markus Bechter. AUTOSAR for connected and autonomous vehicles: The AUTOSAR adaptive platform. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*. IEEE, jun 2016.

[44] John Rushby and Paul S Miner. *NASA Contractor Report CR- 2002-212130, NASA Langley Research Center*, 2002.

[45] DO-297: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations, Radio Technical Commision for Aeronautics (RTCA) SC-200, 2005.

[46] AC 20-148: Reusable Software Components, 2004.

[47] Egbert Althammer, Erwin Schoitsch, Gerald Sonneck, Henrik Eriksson, and Jonny Vinter. Modular certification support—the decos concept of generic safety cases. In *Industrial Informatics, 2008. INDIN 2008. 6th IEEE International Conference on*, pages 258–263. IEEE, 2008.

[48] Tim Kelly. Concepts and principles of compositional safety case construction. *Contract Research Report for QinetiQ COMSA/2001/1/1*, 34, 2001.

[49] Tim Kelly and Rob Weaver. The goal structuring notation–a safety argument notation. In *Proceedings of the dependable systems and networks 2004 workshop*

on assurance cases. Citeseer, 2004.

[50] John Rushby. Kernels for safety. *Safe and Secure Computing Systems*, pages 210–220, 1989.

[51] John Rushby. Just-in-time certification. In *12th IEEE International Conference on Engineering Complex Computer Systems (ICECCS 2007)*. IEEE, 2007.

[52] John Rushby. Runtime certification. In *Runtime Verification*, pages 21–35. Springer Berlin Heidelberg, 2008.

[53] John Rushby. Trustworthy self-integrating systems. In *Distributed Computing and Internet Technology*, pages 19–29. Springer International Publishing, dec 2015.

[54] B. Meyer. Eiffel: The language. *Information and Software Technology*, 34(12):827, dec 1992.

[55] Daniel Schneider. *Conditional Safety Certification for Open Adaptive Systems.* Phd thesis, Technical University of Kaiserslautern, 2014.

[56] Daniel Schneider and Mario Trapp. Conditional safety certification of open adaptive systems. *ACM Transactions on Autonomous and Adaptive Systems*, 8(2):1–20, jul 2013.

[57] W. S. Lee, D. L. Grosh, F. A. Tillman, and C. H. Lie. Fault tree analysis, methods, and applications - a review. *IEEE Transactions on Reliability*, R-34(3):194–203, aug 1985.

[58] J.D. Andrews and S.J. Dunnett. Event-tree analysis using binary decision diagrams. *IEEE Transactions on Reliability*, 49(2):230–238, jun 2000.

[59] Bernhard Kaiser, Peter Liggesmeyer, and Oliver Mäckel. A new component concept for fault trees. In *Proceedings of the 8th Australian workshop on Safety critical systems and software-Volume 33*, pages 37–46. Australian Computer Society, Inc., 2003.

[60] Felix Möhrle, Kai Bizik, Marc Zeller, Kai Höfig, Martin Rothfelder, and Peter Liggesmeyer. A formal approach for automating compositional safety analysis using flow type annotations in component fault trees. In *Safety and Reliability – Theory and Applications.* CRC Press, jun 2017.

[61] Dominik Domis and Mario Trapp. Integrating safety analyses and component-based design. In *Lecture Notes in Computer Science*, pages 58–71. Springer Berlin Heidelberg, 2008.

[62] P. Fenelon, J. A. McDermid, M. Nicolson, and D. J. Pumfrey. Towards integrated safety analysis and design. *ACM SIGAPP Applied Computing Review*, 2(1):21–32, mar 1994.

[63] Yiannis Papadopoulos and John A. McDermid. Hierarchically performed hazard

origin and propagation studies. In *Computer Safety, Reliability and Security*, pages 139–152. Springer Berlin Heidelberg, 1999.

[64] Y. Papadopoulos, J. McDermid, R. Sasse, and G. Heiner. Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure. *Reliability Engineering & System Safety*, 71(3):229–247, mar 2001.

[65] O Lisagor, JA McDermid, and DJ Pumfrey. Towards a practicable process for automated safety analysis. In *24th International system safety conference*, volume 596, page 607, 2006.

[66] Rasmus Adler, Marc Forster, and Mario Trapp. Determining configuration probabilities of safety-critical adaptive systems. In *21st International Conference on Advanced Information Networking and Applications Workshops*. IEEE, 2007.

[67] Joanne Bechta Dugan, Salvatore J Bavuso, and Mark A Boyd. Dynamic fault-tree models for fault-tolerant computer systems. *IEEE Transactions on reliability*, 41(3):363–377, 1992.

[68] Bernhard Kaiser, Catharina Gramlich, and Marc Förster. State/event fault trees—a safety analysis model for software-controlled systems. *Reliability Engineering & System Safety*, 92(11):1521–1537, nov 2007.

[69] Erwin Schoitsch, Christoph Schmittner, Zhendong Ma, and Thomas Gruber. The need for safety and cyber-security co-engineering and standardization for highly automated automotive vehicles. In *Advanced Microsystems for Automotive Applications 2015*, pages 251–261. Springer International Publishing, jun 2015.

[70] Bruce Schneier. Attack trees. In *Secrets and Lies*, pages 318–333. Wiley Publishing, Inc., oct 2015.

[71] Max Steiner. Integrating security concerns into safety analysis of embedded systems using component fault trees. 2016.

[72] Dan M. Shalev and Joseph Tiran. Condition-based fault tree analysis (CBFTA): A new method for improved fault tree analysis (FTA), reliability and safety calculations. *Reliability Engineering & System Safety*, 92(9):1231–1241, sep 2007.

[73] Guoqi LI. A modified fta approach to achieve runtime safety analysis of critical systems. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E97.A(10):2083–2085, 2014.

[74] Guoqi Li. FTA based runtime evaluation of system reliability for autonomous unmanned vehicle. In *2011 Seventh International Conference on Computational Intelligence and Security*. IEEE, dec 2011.

[75] Rogério de Lemos, Holger Giese, Hausi A. Müller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M. Villegas, Thomas Vogel, Danny Weyns, Luciano Baresi, Basil Becker, Nelly Bencomo, Yuriy Brun, Bojan

Cukic, Ron Desmarais, Schahram Dustdar, Gregor Engels, Kurt Geihs, Karl M. Göschka, Alessandra Gorla, Vincenzo Grassi, Paola Inverardi, Gabor Karsai, Jeff Kramer, Antónia Lopes, Jeff Magee, Sam Malek, Serge Mankovskii, Raffaela Mirandola, John Mylopoulos, Oscar Nierstrasz, Mauro Pezzè, Christian Prehofer, Wilhelm Schäfer, Rick Schlichting, Dennis B. Smith, João Pedro Sousa, Ladan Tahvildari, Kenny Wong, and Jochen Wuttke. Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II*, pages 1–32. Springer Berlin Heidelberg, 2013.

[76] Kenneth Östberg and Rolf Johansson. Use of quality metrics for functional safety in systems of cooperative vehicles. In *Lecture Notes in Computer Science*, pages 174–179. Springer Berlin Heidelberg, 2012.

[77] Kenneth Östberg and Magnus Bengtsson. Run time safety analysis for automotive systems in an open and adaptive environment. In *SAFECOMP 2013-Workshop ASCoMS (Architecting Safety in Collaborative Mobile Systems) of the 32nd International Conference on Computer Safety, Reliability and Security*, page NA, 2013.

[78] Collaborative Embedded Systems (CrESt). `https://crest.in.tum.de/`. Last visited: 2018-10-09.

[79] ARAMiS II. `https://www.aramis2.org/`. Last visited: 2018-10-09.

[80] Sebastian Müller and Peter Liggesmeyer. A motion certification concept to evaluate operational safety and optimizing operating parameters at runtime. In *International Conference on Computer Safety, Reliability, and Security*, pages 156–166. Springer International Publishing, Cham, 2015.

[81] Sebastian Müller and Peter Liggesmeyer. Dynamic safety contracts for functional cooperation of automotive systems. In *International Conference on Computer Safety, Reliability, and Security*, pages 171–182. Springer International Publishing, Cham, 2016.

[82] Sebastian Müller and Peter Liggesmeyer. Safety assurance for emergent collaboration of open distributed systems. In *2016 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 249–256. IEEE, 2016.

[83] Sebastian Müller and Peter Liggesmeyer. Systematic composition of services from distributed systems for highly dynamic collaboration processes. In *International Conference on Computer Safety, Reliability, and Security*, pages 225–236. Springer International Publishing, Cham, 2017.

[84] Sebastian Müller, Patrick Wolf, Karsten Berns, and Peter Liggesmeyer. Combining behavior-based and contract-based control architectures for behavior optimization of autonomous vehicles in unstructured environments. In *Commercial Vehicle Technology 2018*, pages 324–335. Springer Vieweg, Berlin, 2018.

[85] International Electrotechnical Commission et al. Functional safety of electri-

cal/electronic/programmable electronic safety related systems. *IEC 61508*, 2000.

[86] IEC 62304: Medical device software Software life cycle processes, 2006.

[87] MagicDraw. https://www.nomagic.com/products/magicdraw. Last visited: 2018-12-07.

[88] Ren Yi, Liu Linlin, and Zeng Shengkui. Fault tree data structure based on XML and the conversion method to BDD. In *2009 WRI World Congress on Computer Science and Information Engineering*. IEEE, 2009.

[89] Gary E Marchant and Rachel A Lindor. The coming collision between autonomous vehicles and the liability system. *Santa Clara L. Rev.*, 52:1321, 2012.

Sebastian Müller

Lebenslauf

Ausbildung

2012 – 2014	**Masterstudium EIT - Automatisierungstechnik**, *TU Kaiserslautern.*
2008 – 2012	**Kooperatives Ingenieurstudium (KOI)**, *Hochschule Kaiserslautern und BASF SE in Ludwigshafen am Rhein.* ○ Bachelorstudium Elektrotechnik-Automatisierungstechnik ○ Ausbildung zum Elektroniker für Automatisierungstechnik

Berufliche Erfahrungen

2015 – 2018	**Wissenschaftlicher Mitarbeiter**, *TU Kaiserslautern, Fachbereich Informatik, Lehrstuhl Software Engineering: Dependability.*
2012 – 2014	**PLT-Ingenieur**, *BASF SE.*