Run time safety analysis for automotive systems in an open and adaptive environment

Kenneth Östberg, Magnus Bengtsson

Run time safety analysis for automotive systems in an open and adaptive environment

Kenneth Östberg and Magnus Bengtsson

Electronics / Software
SP Technical Research Institute of Sweden
Borås, Sweden
{kenneth.ostberg, magnus.bengtsson}@sp.se

Abstract. Cooperative vehicles are no longer fiction. A key factor is the ability for vehicles to exchange information with their environment. The shared information can be used to realize new functionalities, from virtual traffic lights to emergency braking, thus with potential to increase safety and efficiency of vehicle systems. However, external information has inherent uncertainties and this poses a threat to safety. In this paper we will discuss how to handle these uncertainties by use of dynamic safety contracts. We propose an extension to AUTomotive Open System Architecture (AUTOSAR) which consists of a safety manager which actively enforces the safety rules described in such safety contract. We also propose to integrate the architecture of an Intelligent Transport System (ITS) station tightly to AUTOSAR. It is our hypothesis that such architecture provides a viable platform for run time safety assessment. Future research work is to evaluate what kind of safety assessments our system can be able to handle.

Keywords: Safety, AUTOSAR, contract, cooperative system

1 Introduction

There are many reasons to allow vehicles to communicate and coordinate their actions. Shared information has the potential to increase safety and efficiency of vehicle systems. This can also have a direct impact on the cost for a car e.g. for safety reason it may be necessary to have two redundant radars operating between two cars when platooning. If the cars can share information and understand intrinsic uncertainties in the information, it may be sufficient that a car is equipped with only one radar in the front and one radar in the rear. The whole cooperative system will still have two radars operating in between two adjacent cars.

adfa, p. 1, 2011.

An open system is a system that interacts with its environment by exchanging information in one form or another. An adaptive system is a system that is able to handle a dynamic environment, new entities to interact with enter and old ones disappear. Such systems are very complex to analyze and understand from any perspective. Our concern is about safety and today's safety standard, ISO 26262, does not address the situation where vehicle systems are sharing information related to safety. It can be argued that a static design time analysis will have to take a worst case position which might turn out to be overly pessimistic in real situations. It therefore seems necessary to move part of the safety analysis and safety cases from design time into run time to be able to benefit from the advantages of a cooperative systems. Let's clarify with an example.

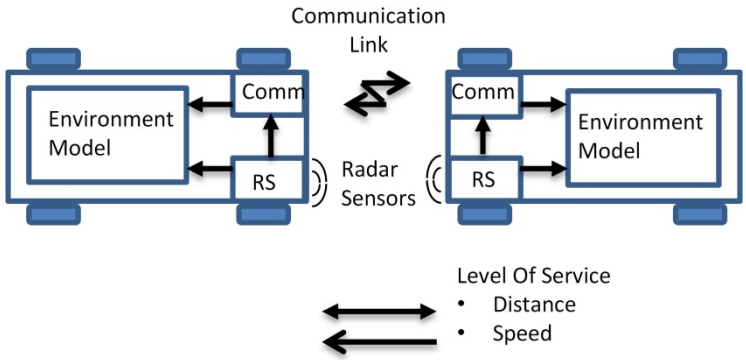

Fig. 1. Two vehicles sharing safety related information

In **Fig. 1** two vehicles are sharing radar information. The information from internal and external sensors is used to build a representation, a model, of the vehicle's environment. We will elaborate on that model more in following section but here we just name it an environment model. The environment model carries all relevant information from a run time safety perspective such as: if the vehicle is participating in a Vehicle Ad-hoc Network (VANET), if external sensors are used, do external sensors give new or redundant information etc. Important attributes in the model are accuracy and precision in the information. As long as the two radars sensors in **Fig. 1** correlate well enough in their distance measurement we are safe, see **Fig. 2**. The question of what constitutes well enough has no absolute answer; it is a relative question that depends on the real situation, our operational situation. Level of service can be defined as a measure of the effectiveness of elements of transportation infrastructure. In our scenario, level of service is determined by the spacing between the vehicles and their speed. Short spacing and high speed means a high traffic flow. The correlation between shared information is a key factor for determining the achievable level of service.

One vehicle may alone have several sensors that should correlate so one might think that there is no major difference in the safety analysis between an individual

vehicles and a cooperative system. We believe that there are differences that matters and will discuss this further in section 3. We can either analyze all our expected situations at run time to find the worst case scenario and from that derive safety integrity levels which will make us safe all the time or we may postpone the analyze to run time. In the later approach the operational situations with their safety integrity levels must be formalized and stored in an appropriate format in the vehicle to be used in run time. We define a safety contract as one or more properties that need to be fulfilled among two parties in order to maintain safe operation. As we allow contracts to be dynamic they must be attributed with information when they are supposed to be activated or deactivated. In our case we relate a safety contract to an operational situation and our properties are related to safety integrity levels. This is a very general definition and we will elaborate more on safety contracts in section 5.

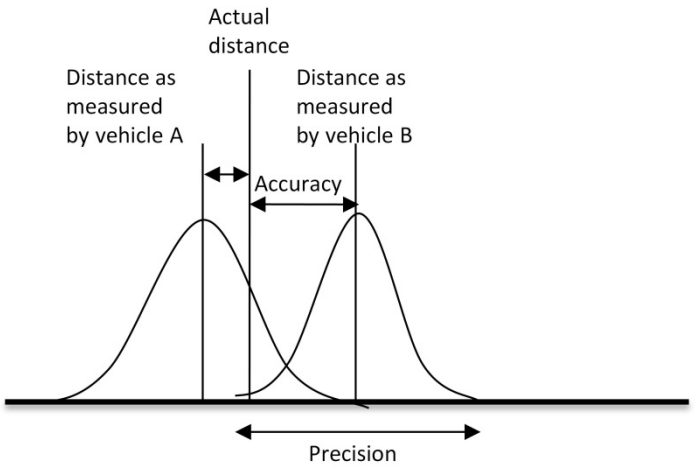

Fig. 2. Accuracy and precision

Another important quality metric that has to be considered is the operational status of the communication link. This can be measured by different metrics like Received Signal Strength Indication (RSSI), Link Quality Indication (LQI) or simply Bit Error Rate (BER). Properties like this should also be subject to the safety analysis and included in a safety contract. These safety contracts must be handled in run time by some form of safety mechanisms.

2 Safety kernels

There are no clear definitions of the related concepts of: safety kernel, safety manager and safety mechanisms. Safety kernels have been described in different forms since the early 1980's. The name kernel can be derived back to early concepts of operating system design and protection rings for fault tolerance and security aspects.

Having different levels of trust and authorities for different programs should make it easier to uphold security. These ideas have then evolved into the domain of safety. In [1] Rushby has taken a theoretical view of safety kernels and the properties they can enforce. The operation of a safety kernel can coarsely be divided in two classes, passive and active. A passive safety kernel provides mechanisms and solutions to assist in building a safe system. There is though no guarantee that an application is using these facilities or using them in a safe way. That has to be asserted by analyzing the system after it has been built to check for possible unreliable behavior. An active system is enforcing the system to be safe by monitoring the system and comparing its operation against a well-defined set of safety properties.

In the domain of operating systems there are ongoing discussions on the different merits of a monolithic kernel design versus a modular architecture. The modular approach strives to separate mechanisms from policies. Mechanisms are static and provide means for different policy components to achieve their given task. The advantages are that policy managers can be run in user privilege mode and thus keeping the kernel smaller, safer and more secure. The policies become easy to modify and change during run time i.e. the rules, not the components handling them. Another benefit of a modular system design is that the components have clear interfaces and responsibilities and may even be small enough to be subjected to formal verification [3]. The benefit from monolithic kernels is faster response time as much of the kernels internal communication can be handled by shared memory.

Let's look at AUTOSAR to understand its current safety mechanism. A memory protection unit (MPU) constantly monitors memory accesses and compares them to a table of allowed accesses for each software process. It is a low level device implemented in hardware and can therefore be classified as an active safety mechanism. AUTOSAR has support for classifying software modules as critical and will allocate them, during the software build process, to memory partitions to be monitored by a MPU. This protects data from spatial interferences. The watchdog manager in AUTOSAR provides a temporal monitoring capability for software components in the form of checkpoints that can be created, in run time, to form control flow graphs that are supervised. This checks data integrity from a temporal aspect. The temporal aspects can be either pure logical, just checking paths or physical by checking the real timing between checkpoints. The watchdog manager can be classified as a passive policy manager.

The need for run time safety contract has already been concluded by others. Schneider and Trapp have introduced a concept of conditional safety certificates (ConCerts) and how to operationalize them [4]. This is an important contribution but they do not discuss how to handle the uncertainties in the information coming from the communication with the environment at run time and relate these uncertainties to current operational situation to understand if the situation is safe or unsafe. How to perform the safety analysis for a system as described previously and standardize a safety contract that can be negotiated and monitored at run time are an open issues

and our future research work. We will elaborate on the design time safety analysis process and safety contract in the next sections. Safety and security aspects are clearly closely related and interdependent in an open environment. Information access is vital for run time safety analysis and safety components may have access rights which make them obvious targets for malicious security attacks. It is therefore necessary to monitor information both from safety as well as a security perspective at the same time. This paper will not discuss security further but both safety and security are system attributes and they both need attention during the whole life cycle process.

Neither of the safety concepts discussed above addresses the upcoming need of monitoring safety contracts that addresses uncertainties, or quality metrics, in data, i.e. data integrity. We therefore propose to add a safety manager to AUTOSAR and position it as an active policy manager that enforces safety rules.

3 Functional safety standard for road vehicles

Safety is always a system attribute and has to be supported by a life cycle process. We must therefore also look at the impacts on ISO26262 which is the safety standard for road vehicles. ISO 26262 takes as starting point of a safety analysis an item. An item is a pure logical concept and defined as a system or array of systems to implement a function at the vehicle level, see **Fig. 3**.

The safety analysis process has its starting point in scenarios and operation situation to understand if there exist potential sources of harm, i.e. hazards. Any hazard is then subjected to hazard and risk analysis and considered from three dimensions, its controllability, its severity and the probability of exposure. A top level safety requirement is derived and assigned an Automotive Safety Integrity Level (ASIL) based on these three parameters. From that point the operational situation does not serve any more purpose as the strongest safety requirement will eventually determine the ASIL.

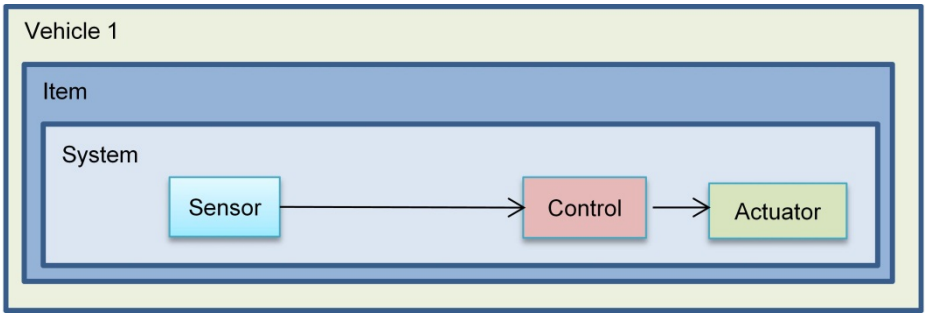

Fig. 3. An item

We introduce a concept of a wireless communication link between vehicles and define a linked vehicle as a vehicle participating as a node in a VANET for the purpose

of coordinated or cooperative control. The definition of a linked item and linked system follows directly according to **Fig. 4**. The difference between items at vehicle level and linked item at cooperative level is that at vehicle level the items are analyzed as independent. At cooperative level the linked item must be analyzed as dependent items. The integrity levels at cooperative level are relational and dynamic properties and dependent on actual run time operational situation, e.g. the spacing between the vehicles, correlation between shared information, the road condition, the weather condition etc.

As we introduce the concept of linked item, and therefore do not have a static view of the system during the life cycle process, we need to keep track of all information that was used for the safety requirements including the operational situations. This information has to be formalized in an appropriate format suitable for run time use.

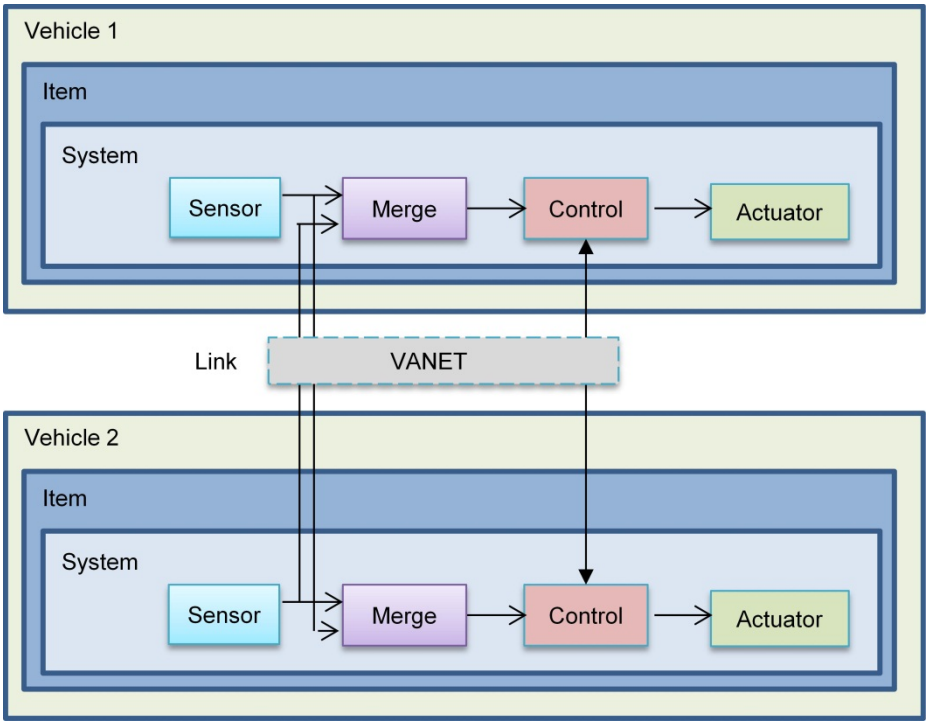

Fig. 4. Linked items

We have so far looked at safety related issues with open and adaptive system of vehicles. We have argued that a pure static analysis will be very restrictive due to the dynamic and uncertain behavior of such system. Our aim with this paper is to define the key concepts needed for achieving safety assessment in run time and furthermore to present a conceptual model of a safety architecture suitable for run time based safety assessment.

4 Modeling approach

In this section we aim to identify the key concepts needed in order to define an architecture that can handle safety assessment in runtime. We will suggest a system model, or system architecture, for our future research activities. The novelty with our approach to conduct safety assessment in runtime is that we define dynamic safety contracts that describes operational situations and relates them to safety properties that need to be monitored during run time. We introduce the concept of a Global Static Map (GSM) to hold the safety contracts. We introduce a safety manager in the context of AUTOSAR which monitors the actual operational situation and tries to match that situation to the predefined set of operational scenarios in the GSM. The safety contracts that are related to the matching scenarios are retrieved and subjected to safety analysis.

Our safety concept is general e.g. we may define link connection and link disconnection procedures as operational situations. It is thus possible to define safety contracts that must hold for the link to be formed initially e.g. number of sensors needed, their type and accuracy, the kind of failure semantics they have and what needs to be monitored during the links operational time etc. In the same way it is possible to specify under what conditions the link should be disconnected.

The concept of a linked item constitutes the boundaries from safety analysis perspective. A dynamic vehicle system implies some form of vehicular ad-hoc networks (VANET) technology. The European Telecommunications Standard Institute (ETSI) has a set of standards related to Intelligent Transport Systems (ITS). We assume that future vehicle system will adhere to such standards. These standards must be the base upon which we define a linked item and its life cycle, how it is created, maintained and dissolved.

In ITS, a Local Dynamic Map (LDM) [2] is a conceptual data storage containing information which is relevant to the safe operation of a ITS station. LDM is thus not defined for vehicles alone but for all type of systems that participate in an ITS. It is our hypothesis that LDM is suitable for retrieving actual operational situation for our safety assessments. It is convenient to have single source of information for all run time related safety information. We thus propose to augment the information in LDM with safety related information needed to support our approach i.e. attribute data with quality metrics describing uncertainties e.g. current link status, how well redundant information correlates etc. We plan to experiment with an LDM tightly integrated with AUTOSAR and fast enough to supply the software components with their information, see **Fig. 5**.

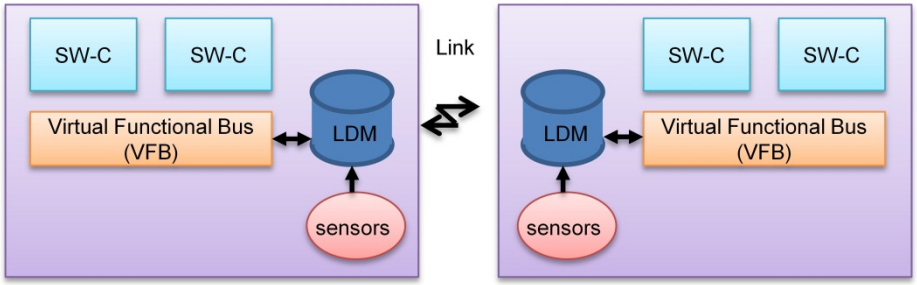

Fig. 5. The logical view of LDM in AUTOSAR

The LDM creates a new abstraction layer for the software components as they become unaware of actual sensors. Only information in the LDM becomes relevant and can come from internal sensors, external sensors or even merged sensor data. The integrity of the information in LDM is vital from both safety and security perspective. ITS standards address the security aspect which is natural as the LDM is a kind of database. To check for safety properties we add a safety manager which monitors the LDM from safety perspective.

We have introduced a novel concept, a GSM to match the LDM. It is our data storage for the design time safety analysis and contains operational scenarios and related safety contracts. The GSM is a vital component as it must be possible to quickly match and select the applicable safety contracts. In the future we envision that all types of ITS station can have both a LDM and a GSM together with a unified safety & security manager. A vehicle is an instantiation of a mobile ITS station. There is a hidden implication here, we must match the GSM with a design database to holds all relevant information. What we describe here are three databases, the LDM, the GSM and the design database. We envision an underlying database technology supporting our needs. Having the same foundation for the databases is important as they will have common semantic contents. As we stressed before, safety is a system attribute. It will break at the weakest chain of the link. Data integrity, which we will discuss more in next section, is important during the whole life cycle process. Understanding how to design a database that suits our needs to be secure, flexible, adaptable and fast is important part of our future research work.

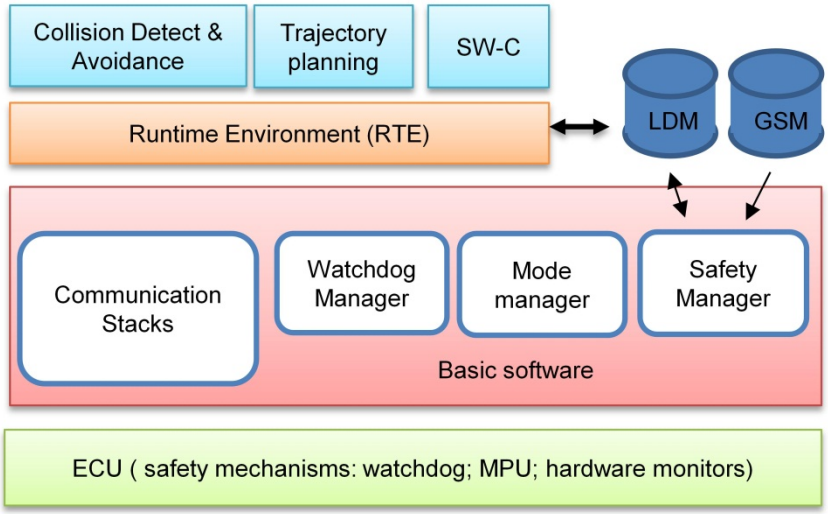

Fig. 6. AUTOSAR extended with safety components

The safety manager is an active monitoring entity that has a set of rules that it monitors. We will look further at safety contracts in next section. **Fig. 6** shows all the concepts from a vehicle perspective in relation to AUTOSAR. Our extensions to AUTOSAR are: the LDM, the GSM and the safety manager. We would like to stress that AUTOSAR is a very general architecture concept and would easily adapt to any ITS station. **Fig. 6** also shows some supporting software components that reads and updates the LDM, though important they are not considered as key components and hence out of the scope for this paper.

To achieve a safe cooperation in our system of linked vehicles we must also handle consensus i.e. that the information in each vehicles LDM is consistent with others LDMs and that decisions taken in individual vehicle are known and acceptable to all other nodes. AUTOSAR defines a mode manager that is responsible for the state of the vehicle's system, from the lowest level i.e. individual ECUs to the highest level, the whole vehicle. The mode manager consists of two parts, Mode Arbitration and Mode Control. It seems natural to extend its responsibilities up to the next level, our linked system of vehicles. Linking to another vehicle could be viewed as new dynamic ECUs appear in the existing system. Arbitration and control must then be extended to include other vehicles in order to reach consensus. Distributed arbitration and control in a dynamic environment is a research topic on its own and not part of our scope. For our purpose we assume that the LDM contains all the relevant information the safety manager needs and that the mode manager is responsible for updating the LDM. The process flow is as follows. When the safety manager discovers unsafe situations it notifies the mode manager. The mode managers must arbitrate and reach consensus between the vehicles. How well consensus is reached is something that the safety manager could monitor e.g. information like decision latency and link quality

are important metrics. If the vehicles cannot communicate or reach consensus the vehicle must operate alone. In this case we rely on traditional safety analysis and have the same situation as today with respect to ISO 26262 and AUTOSAR.

5 Safety contracts

A safety contract in its most general definition is a contract between two, or more, parties who exchange information and where the integrity of the information is important from a safety perspective. The contract specifies properties that must be satisfied for the contract to be valid. Our contracts should be formally described in a suitable way to be able to be monitored by our safety manager. In [5] integrity is defined as 'absence of improper system alterations'. Hence data integrity can be defined as 'absence of improper system data alterations'. The distinction is important as one can separate a software system in the program, and the data it processes. Both have potential to be a safety concern but program alteration can be hindered by storing the program in non-volatile memory. It should also be noticed that the program has some form of functional integrity, i.e. how well it has been verified to its specification. And the specification in turn may have some integrity attributes or quality metrics describing how much trust one should have on it. Another definition of data integrity is that "data should be consistent during its life time". This definition stresses that data may have a life time and that all users of the data should have a view of it that makes the whole system consistent.

Before discussing safety properties let's take a look at software properties in general. Software can be checked for properties in a variety of ways. The first step is the static checking that the compiler performs. These checks can be simple, like syntax checking or more complex, like type checking. Still, checking is a process to evaluate software against some predefined property. Compilers never checks for functionality though. For this purpose we can either rely on static property checking or dynamic property checking. Static techniques for property checking can broadly be classified as either model checking or theorem proving. These are considered difficult and costly today so we rely more on dynamic verification techniques like simulation. Simulation in turn can be static i.e. be performed with a known set of test data, or dynamic where the test set is randomly produced. In any case we always have a reference, an oracle or golden reference model, upon which we rely and put our trust in that it has our desired properties. Verification is thus the process to check how well our software correlates to, or fulfills, those oracle properties. Our approach with safety contracts that are checked in run time is really no different. We have a random environment, the real operational situation, which supplies stimulus and we have our safety properties as references that we would like to assert. What differs is how we handle the situation when the properties are not ok. In our case we can, and must, shift our operational situation to one that is safe under the systems current state. What is considering correct behavior and what is considering safe are situation, or context, dependent properties.

Some properties are hard to assert statically e.g. a compiler may, falsely, reject a correct program. This is because the compiler internally works with an abstraction of the program and does not use all the available information. The design tradeoff taken is that it is better, easier, to reject difficult programs and only approve simpler ones than to spend time on a complex analysis process. From this discussion we realize that abstractions, in the software but also in the operational situation, are important and what matters is what abstractions we make and what information is kept and thus can be analyzed. We need to better understand how to use an incremental design process, one that is supported with techniques like safety analysis in form of a fault tree analysis (FTA) to realize what kind of abstractions we need to define our safety contracts.

Much of the data produced in a vehicle comes from sensors. A sensor has two important attributes, its accuracy and precision. Any measurement has uncertainties. Mean value and standard deviation are important statistical attributes of a series of uncertain measurements. Accuracy can be defined as how well a measurement relates to the mean value. Precision is related to standard deviation. The uncertainty can be viewed as a perfect measurement added with noise, where the noise can be both in the spatial domain e.g. quantization noise, and the temporal domain e.g. phase noise or jitter (phase noise is described in the frequency-domain and jitter in the time-domain)

We conclude that there are three different dimensions on how to understand data, its quality and integrity:

- A spatial dimension
- A temporal dimension
- An uncertainty dimension. In some occasion it can be difficult to understand the cause of the uncertainty so we give it a separate domain.

The formalism needed in form of different logic systems, first order logic, temporal logic, fuzzy logic, Bayesian logic etc. is not our primary research question. We will choose the relevant logic systems according to our needs and abstractions. Our contribution is the research about dynamic run time safety contracts in an environment that is uncertain and dynamic. For that we need more basic understanding in how to formalize the concept of operational situations and safety properties in a format suitable for run time monitoring.

6 Conclusion & future work

In the future our vehicles will operate in an open and adaptive environment, both in autonomous mode and in cooperative mode. In this paper we have analyzed this from a safety perspective and discussed and motivated the need to move parts of the safety assessment into run time. Any solution will probably have an impact on current standard efforts but it is our ambition to follow and harmonize as much as possible with existing standards.

We have researched the domain for relevant standards and have a hypothesis that AUTOSAR and ITS Station should be more tightly integrated to be able to provide vehicles with run time safety monitoring capability. We have suggested an initial experimental platform for our future research work. We have discussed the advantage of a uniform database technology suitable for storing information about operational situations, safety properties and run time quality metrics. We define a safety contract as a formalized description of an operational situation and its related safety properties. These contracts are dynamic as they relate to different operational situations and thus are not always applicable. We have discussed a safety manager that selects appropriate safety contracts dependent on the current operational situation and monitors their safety properties.

Future research work is to understand what kind of safety requirement emerges when the vehicles becomes autonomous and cooperative. How to describe an operational situation and safety properties in a formalized way to be included in the safety contracts. How to select and apply contracts dynamically while still maintaining safe operation. We will evaluate the proposed system from different aspects. The most important ones are: overall safety, impact on current standard and cost. It is also natural to investigate to what extent we can substitute the passive temporal checking performed by the watchdog manager with active checking performed by our active safety manager.

We have excluded discussion about what kind of formal notations are needed for such system as that will be discovered during our research process.

Acknowledgement.

This work as been supported by the EU under the FP7-ICT programme, through project 288195 "Kernel-based ARchitecture for safetY-critical cONtrol" (KARYON).

References.

1. Rushby, John. "Kernels for safety." Safe and Secure Computing Systems (1989): 210-220.
2. ETSI TR 102 863 Intelligent Transport Systems (ITS);Local Dynamic Map (LDM);
3. Gerwin Klein, Kevin Elphinstone, Gernot Heiser, June Andronick, David Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch and Simon Winwood “seL4: Formal verification of an OS kernel” Proceedings of the 22nd ACM Symposium on Operating Systems Principles, pp. 207–220, Big Sky, MT, USA, October, 2009
4. Schneider, D.; Trapp, M., "A Safety Engineering Framework for Open Adaptive Systems," Self-Adaptive and Self-Organizing Systems (SASO), 2011 Fifth IEEE International Conference on , vol., no., pp.89,98, 3-7 Oct. 2011 doi: 10.1109/SASO.2011.20
5. Avizienis et al, "Basic Concepts and Taxonomy of Dependable and Secure Computing," IEEE Transactions on Dependable and Secure Computing, vol. 1, pp. 11-33, 2004.