

Guidelines for the Development of Dependable Integrated Safety Systems

– Results of the EU Project EASIS –

Marko Auerswald

Robert Bosch GmbH
Corporate Sector Research and Advance Engineering
Frankfurt am Main

Contents

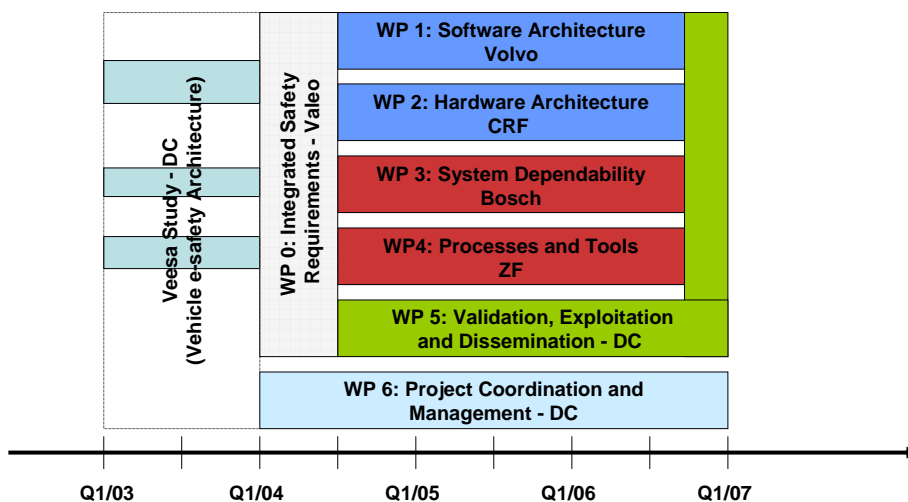
- The EASIS Project
- Enabling Technology for ISS
- The EASIS Dependability Activity Framework
- Guidelines for the Dependability Activities
- The ISS Application Development Process
- Application of Formal Methods
- Conclusions

EC White Paper on Transport 2001

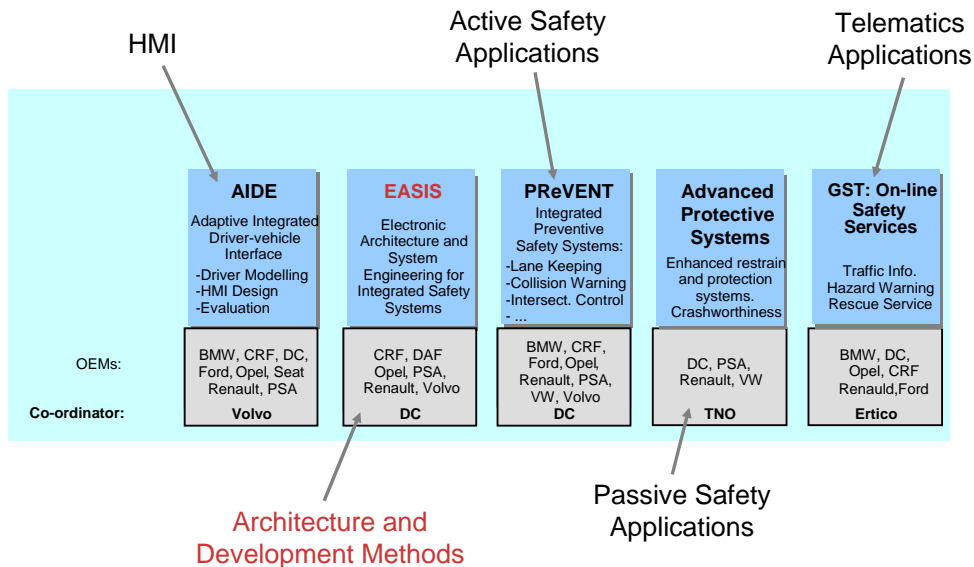
- „Transport by road is the most dangerous and most costly in terms of human lives“
 - 1.64 million killed in EU since 1970 in road accidents
 - in 2000: 40.000 killed, 1.7 million injured
 - for 14-25 year olds road accidents are the prime cause of death
 - indirect costs of road accidents: 160 billion € (2% of EU's GNP)
- Target: halve the number of road deaths by 2010
 - „The European Union has considerable, even sole, responsibility for encouraging the deployment of innovative technologies which should lead to the introduction of safe new vehicles on the market“
- Addressed by EUCAR program on integrated safety

Source: White Paper „European transport policy for 2010: time to decide“, European Commission, 2001

EASIS Project Roadmap and Structure



EUCAR Program „Integrated Safety“



The EASIS Consortium

Vehicle Manufacturers

DaimlerChrysler, DAF Trucks, Fiat (CRF), Opel, PSA, Renault, Volvo (VTEC)

Suppliers

Bosch, ContiTeves, Lear, Freescale, Philips, TRW, Valeo, ZF

Tool and Middleware Suppliers

DECOMSYS, dSPACE, ETAS, Vector

Research Institutes

MIRA, OFFIS, University Duisburg/Essen

EASIS Organizational Information

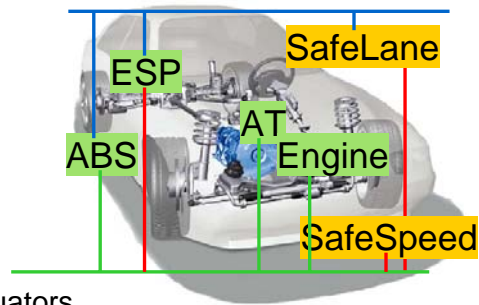
Cost	9.400 T€
Funding	5.000 T€
Workload	673 Person Month
Starting Date	1 January 2004
Duration	3 Years
Coordinator	DaimlerChrysler

Contents

- The EASIS Project
- **Enabling Technology for ISS**
- The EASIS Dependability Activity Framework
- Guidelines for the Dependability Activities
- The ISS Application Development Process
- Application of Formal Methods
- Conclusions

Integrated Safety Systems

- Functions are integrated to functional networks
 - Advanced vehicle dynamics systems
 - Advanced safety systems e.g. SafeLane, SafeSpeed, ...



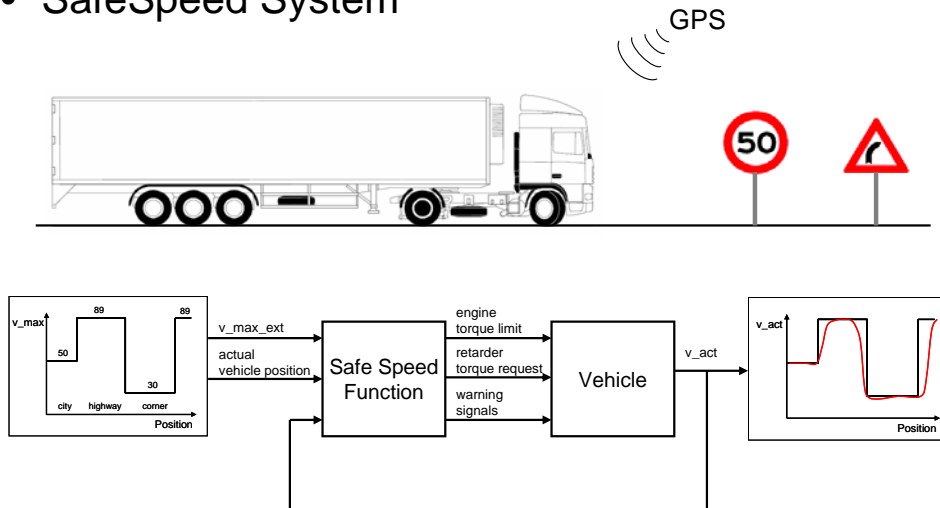
- Common use of Sensors and Actuators
 - Increased requirements on signal timing / availability
 - Increased functional interaction
 - Functions may give conflicting inputs to actors
 - Coordination of functions is indispensable

ISS – An Automotive Dependability Challenge

- Safety-critical fail-operational software-based systems
 - transition took place in avionics industry, not yet in automotive industry
 - automotive industry needs adapted approaches
- High system complexity
 - high number of connected ECUs providing a function
 - high complexity of the functions (#inputs, #failure modes)
 - multiple interrelationships across vehicle domains
 - possibility of actuator control conflicts between different safety functions
 - complexity of the control algorithms with different levels of control
- Shared responsibilities of suppliers/vehicle manufacturers

ISS Example: EASIS WT5.2 Validator

- SafeSpeed System



The **SafeSpeed** application, although being comparatively simple, already shows many typical ISS issues.

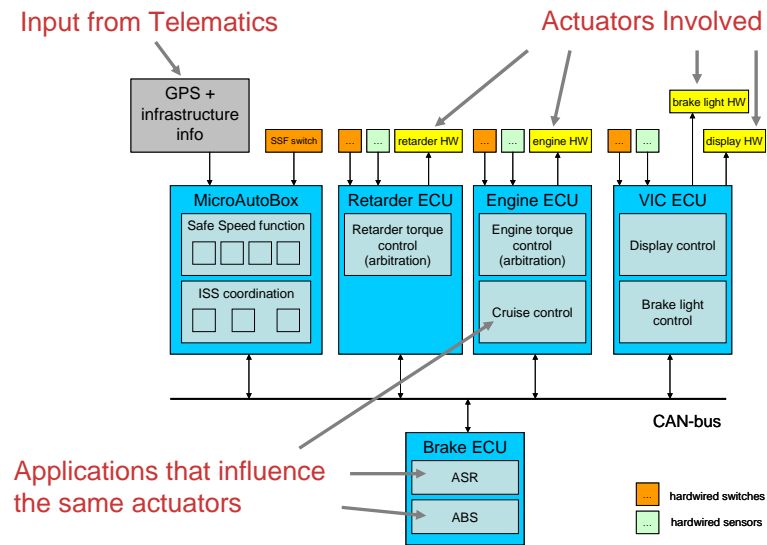
The purpose of the function is to limit the speed of a truck according to existing speed limits and dangerous road conditions. The information about the maximum safe speed is provided by the telematics domain. The SafeSpeed System tries to enforce the speed limit by issuing torque requests for the engine control and the retarder control.

ISS: Integrated Safety System

SSF: SafeSpeed Function

Retarder: Device for slowing down trucks and other vehicles

SafeSpeed: Topology



Typical ISS Properties

- Safety-relevant applications across domain and network borders
- Integration of functions with different levels of safety criticality
- Integration of sensor input from different domains
- Coherent actions and reactions over different domains

EASIS provides Enabling Technology for ISS

- Software Architecture and Basic Services
- Hardware Platform
- Guidelines for System Dependability
- Processes and Tools

Current state of automotive dependability methodology

- Absence of a broadly accepted approach for hazard identification in the early phases of development
- Unclear responsibilities with respect to hazard identification in the development of integrated systems
- Lack of a broadly accepted and mature approach for hazard classification
- Lack of broadly accepted safety approach (tolerable risk, ALARP, prescriptive approaches)
- Absence of a comprehensive and systematic approach for the establishment of dependability requirements
- Absence of broadly accepted verification techniques that are suitable to tackle systems of such high complexity
- Lack of an accepted approach for comprehensively and consistently demonstrating that safety was sufficiently considered in system development (e.g. safety case)

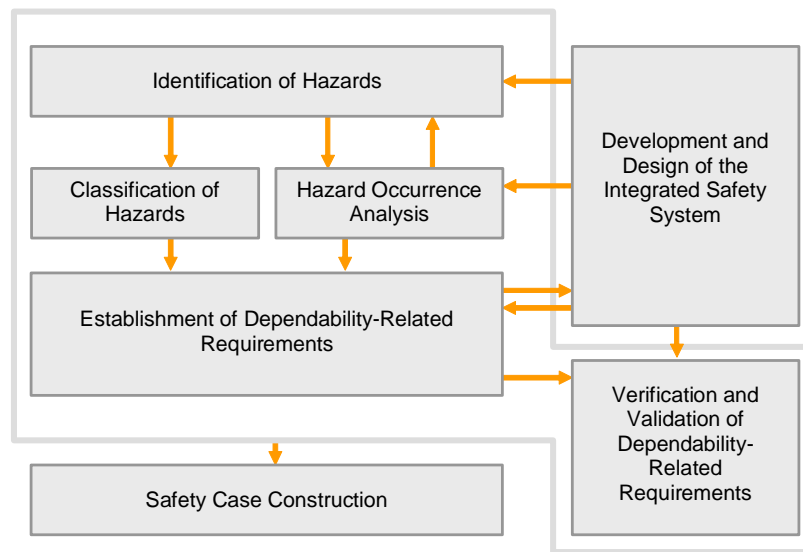
Contents

- The EASIS Project
- Enabling Technology for ISS
- **The EASIS Dependability Activity Framework**
- Guidelines for the Dependability Activities
- The ISS Application Development Process
- Application of Formal Methods
- Conclusions

The EASIS Dependability Framework

- The framework was worked out in due consideration of the following state-of-the-art approaches
 - **IEC 61508** (process industry and generic standard)
 - **ARP 4754, RTCA DO-178B, DO-254** (Aerospace)
 - **MISRA Guidelines** and **MISRA Safety Analysis** Guidelines (Automotive)
 - **ISO/WD 26262** Activity (Automotive)
 - **UK MoD Def Stan 00-56**
 - **US MIL-STD 882D**
- See: EASIS Deliverable D3.2, Part 1, Appendix A:
„Process Frameworks for Dependability“

The EASIS Dependability Framework



Hazard identification: Identification of the undesirable vehicle-level states and behaviours that may be caused by the system being considered

Hazard classification: Qualitative classification of the degree of undesirability associated with each identified hazard

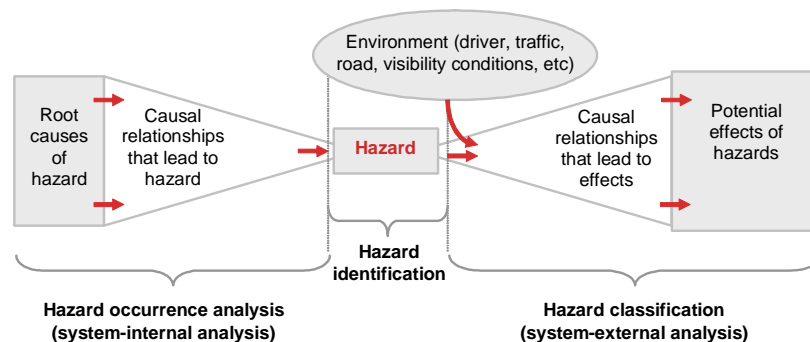
Hazard occurrence analysis: Identification and investigation of the cause/consequence chains that may lead to a given hazard

Establishment of dependability-related requirements: Formulation of requirements on the design process and on the finally implemented system

Verification and validation: Checking of whether or not the dependability-related requirements are fulfilled (verification), and checking of whether or not the system fulfils the dependability expectations of typical users and other stakeholders (validation)

Final dependability assessment: Dependability analysis of the developed system. This includes the production of a Safety Case which is the topic of a separate subtask within the EASIS WT 3.1 work.

Meaning of „Hazard“ in the EASIS Framework



A hazard is an **undesirable condition or state** of a **vehicle** that could lead to an undesirable outcome depending on other factors that can influence the outcome

The chosen approach enables the system-internal and system-external issues associated with a hazard to be analyzed separately from each other.

This hazard concept covers any deviation from the desired behaviour, regardless of whether this behaviour is safety-critical or not. Whether or not a particular such condition really affects the safety is analyzed in the Hazard Classification activity, not in the Hazard Identification activity

Hazards are defined as “undesirable conditions or states”. If hazards were defined as undesired behaviour, the driving situation would influence both the occurrence and the outcome of the hazards, generally resulting in a complex analysis which does not allow the occurrence and the outcome to be studied independently. This would prevent a clear separation between system-internal and system-external analysis.

Hazards are defined with respect to the vehicle and not with respect to the system of concern. When the system is analyzed, only those hazards are of interest that can be caused by the system of concern.

Contents

- The EASIS Project
- Enabling Technology for ISS
- The EASIS Dependability Activity Framework
- **Guidelines for the Dependability Activities**
- The ISS Application Development Process
- Application of Formal Methods
- Conclusions

Hazard Identification Methods

- Checklists
- Hazards and Operability Analysis (HAZOP)
- Functional Hazard Assessment (FHA)
- Hazard Identification Based on State Transition Models
- FMEA

All relevant hazard characteristics should be included in the description of a specific hazard. Thus, a hazard can often be broken down into several different hazards with different characteristics. Examples of such characteristics are:

- **The magnitude of the potential deviation** from the desired behavior (force, velocity, etc)
- **The duration of the hazard.** (Some hazards are characterized by a specific duration because of the way the system is, or may be, implemented)
- **Information provided to the driver** about the existence of the hazard. (For the airbag example, there is obviously a large difference between "*airbag inoperable and driver is informed about this*" and "*airbag inoperable and driver is not informed about this*".)"

Some examples of hazard formulations are given below:

- Inflation of the airbag in a non-collision situation
- Inability to inflate the airbag
- Time-limited inability to inflate the airbag

The most effective way to identify hazards appears to be to use several methods since they complement each other. Checklists can be used for an easy start followed by HAZOP, FHA and 'Hazard Identification Based on State Transition Models' for the different aspects they bring into the analysis. FMEAs are already a part of typical automotive development processes and it is a simple task to transfer the "Effects" listed in the FMEA tables to the hazard list. Identification of hazards that are not related to failures should also be made and the results incorporated in the list of hazards.

Hazard Identification

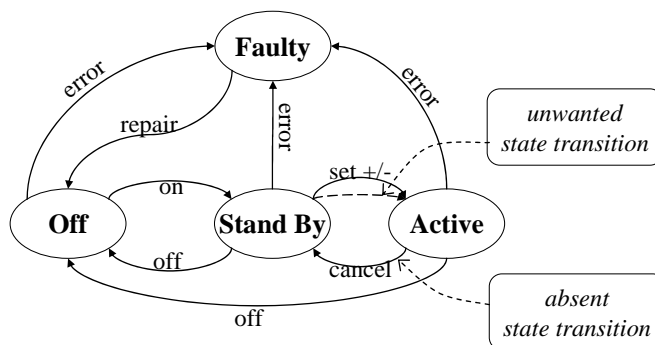
- Hazards and Operability Analysis (HAZOP)

Entity	Attribute	Guideword	Interpretation	Causes	Consequences
Brake Actuator	Force	None	No Brake Force	...	No Deceleration of Vehicle

- **Example attributes:** force, expansion, data flow, value, timing, level
- **Example guidewords:** missing, too high/low, as well as, part of reverse, other than, early, late, before, after, inadvertent stuck

Hazard Identification

- Hazard Identification Based on State Transition Models



Hazard Identification

- Identification of hazards that are **not** related to undesirable conditions or states of the system
 - not within the main scope of the EASIS dependability work package
- Combined Hazards
 - have to be considered if they could appear in combination
 - the effects might be much more critical than the effects of the separate hazards

It should be noted that a fault may lead to more than one identified hazard. The resulting combination can be considered as a separate hazard and should be included in the list of hazards. The possibility of such combined hazards should be considered when the hazard identification is carried out. This implies that an iteration is necessary between the Hazard Identification and the Hazard Occurrence analysis, since the latter investigates the relation between causes and the resulting hazards.

As a simple example, let us assume that the following hazards have been identified for the SafeSpeed system:

- H1: unnecessary reduction of engine torque
- H2: unnecessary activation of retarder

It is quite obvious that any fault that makes the CAS falsely believe that a collision is about to occur would lead to a simultaneous occurrence of H1 and H2. Thus, it is reasonable to assume that sensor faults and faults in the CAS software or hardware may lead to the combination H1+H2. A third hazard should therefore be defined as follows:

- H3: unnecessary reduction of engine torque and simultaneous activation of the retarder

Some hazard identification techniques are less effective than others for identification of combined hazards. For example, a HAZOP performed on the outputs of a system would typically not find combined hazards since it considers one output at a time.

Hazard Classification

- Evaluation of Existing Approaches
 - Risk Graph (IEC 61508 Part 5 Annex D Example)
 - Controllability
 - MISRA Safety Analysis Guidelines Risk Graph
 - ISO/WD 26262 ASIL Classification
 - MIL-STD-882D
- Proposal for an Alternative HC Approach
- Benchmarking of HC Approaches

Proposal for an Alternative Hazard Classification Approach

Hazard Criticality is determined by considering the effects of a hazard in terms of

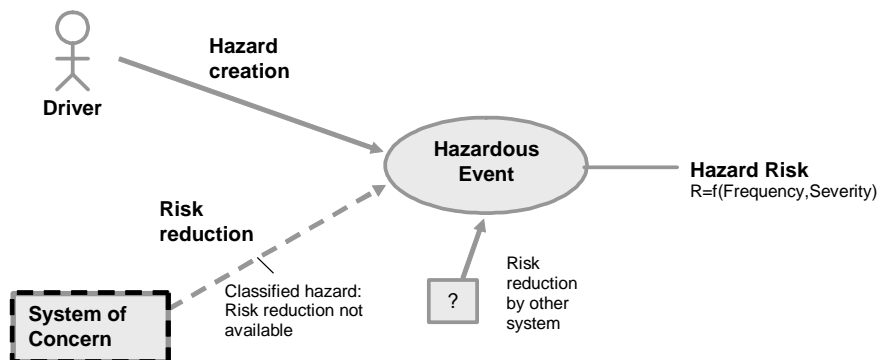
- Severity
- Exposure:
Prob (driving situation in which the hazard could lead to severity S)
- Possibility of non-avoidance:
Prob (Severity S | hazard exists and driving situation as above)

The proposed approach has many similarities to the ISO/WD 26262 approach.

Major deviations are:

- Instead of „controllability“ the term „possibility of non-avoidance“ is used to account for the fact that a high value here represents a high criticality (i.e. possibility of **non**-avoidance and not a high controllability)
- The ranges of Exposure E and Possibility of Non-Avoidance P are not discretized. Only the product of both ($E \cdot P$) is discretized. This avoids paradoxical outcomes that can be achieved otherwise.
- The approach accounts for the fact that the same hazard, depending on the driving situation, may lead to different Severities S with different Exposures E and different Probabilities of Non-Avoidance P. A sound approach for combining the results of distinct driving situations is provided.

Classification of ISS Unavailability Hazard



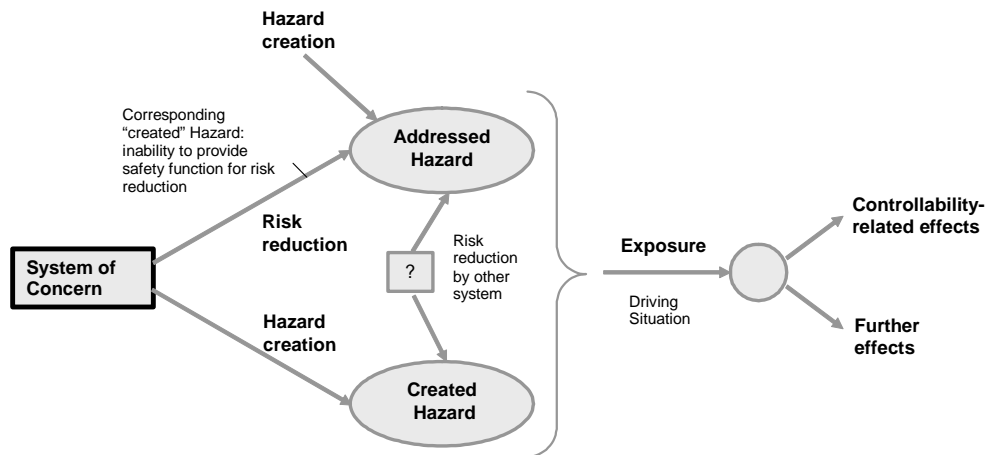
In the SafeSpeed example, a new function is added to a system that is considered to be acceptably safe without this function. The function provides risk reduction but does not seem to be necessary for risk reduction.

Nevertheless, the unavailability of this system might be justifiably classified as non-acceptable from a safety perspective. The reason is that the broadly accepted risk of today's road traffic is not necessarily the risk margin for integrated safety systems.

On the other hand, if a hazard classification approach comes to the result, that a significant amount of risk reduction is necessary for the hazardous event addressed, then the system of concern might be valuable and fit for use even if it does not provide the required amount of risk reduction given that there is no reasonable approach to achieve the necessary risk reduction.

The benchmarked hazard classification approaches cope differently with this issue and thus come to different results especially for this class of hazards.

Hazard Classification - A Generalized View



A deeper understanding and modeling of the causal chains associated with the hazard might help to understand how the parameters of the chosen classification scheme have to be interpreted.

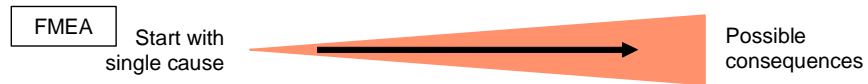
There is an important difference between hazards that are created by the system of concern and hazards that are not created but addressed by the system of concern (of course failing to do so is a hazard „created by the system of concern“).

While the risk of hazards that are created by the system of concern can be **effectively** reduced by a proper design, the risk of hazards that are addressed by the system of concern often can only be reduced by a limited amount; e.g. a collision avoidance function with recent technology can reduce the collision probability only by a comparatively low factor, not by orders of magnitude.

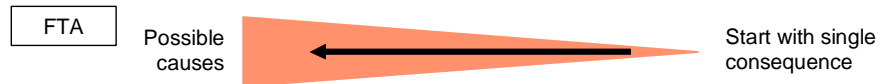
Note: The EASIS scope of hazards, as described earlier, focuses on the “created hazards”. Nevertheless, for every “addressed hazard” there is a “created hazard”: the inability to provide the risk reduction function for the “addressed hazard”.

Hazard Occurrence Analysis

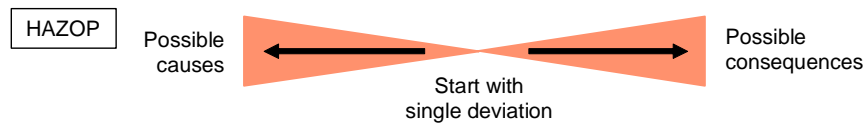
Inductive reasoning (bottom-up)



Deductive reasoning (top-down)



Exploratory reasoning



The Figure is adapted from: F. Redmill, M. Chudleigh, and J. Catmur, System Safety: *HAZOP and Software HAZOP*, John Wiley and Sons, 1999.

Inductive and explanatory approaches are valuable to achieve a better coverage in hazard identification. On the other hand, deductive approaches are valuable to achieve the necessary coverage in establishing dependability related requirements.

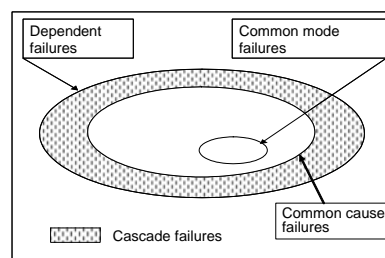
Therefore, all approaches have their right to exist and should be combined appropriately.

Hazard Occurrence Analysis

- Approaches studied:
 - FME(C)A, FTA, HAZOP, ETA, CCA
 - Dynamic Approaches: Markov Modelling, Dynamic ETA
 - Approaches for investigating dependent failures
 - Approaches tailored for investigation of software

FMEA: Failure Mode and Effects Analysis
FTA: Fault Tree Analysis
ETA: Event Tree Analysis
CCA: Cause-Consequence Analysis

Investigation of Dependent Failures



- FTA, FMEA
- SAE-ARP 4754 and SAE-ARP 4761 (Aerospace)
- IEC 61508 β factor method

Software Hazard Occurrence Analysis

- Approaches studied: Software FMEA, System Software Block Failure Analysis
- Recommended: **High abstraction level:**
 - System FMEA including software components most effective
 - Detailed SW FMEA
 - does not fit iterative development and timing constraints in automotive projects
 - is only of limited value, as effects are often arbitrary and thus potential issues are usually resolved by process and coding standards
 - only suitable for small, very safety-critical parts of Software

System level FMEA (or ETA) methods including software at the architectural level, together with additional process and product requirements that result from a very generic software FMEA can provide the vast majority of the value which may be available from performing a very detailed, product specific software FMEA

Recommendations developed from a detailed automotive software FMEA will tend to be similar, independent of the actual product being developed. That is to say that the recommendations would be the same if the product is a windscreen wiper, rear heated window or steering system. Furthermore, it is suggested that the methods to resolve the potential issues that arise are well documented in numerous process, coding standards and other types of documentation already available.

Although most of the value provided by software FMEAs is provided by the system level evaluation, evaluation only at the system level cannot completely ensure that single point failure paths are not included in the system and software design.

Establishment of Dependability Requirements

- Typical Dependability Related requirements can be assigned to one of thirteen categories
- For each of the categories it has been worked out
 - How to determine suitable requirements
 - How to express the requirements
 - How the requirements of this category are related to other requirements
 - How to verify the requirements

Categories of Dependability Requirements

- Hazard probability requirements
- Fault tolerance and functional degradation requirements
- Requirements on System Architecture
- Requirements on specific detection mechanisms and corresponding reactions
- Quantitative requirements for hardware architecture
- Requirements on the avoidance of non-systematic faults
- Critical functional requirements
- Requirements for functional limitations
- Requirements on the design process
- Requirements on isolation and diversity
- Requirements on the manufacturing process
- Requirements for systems external to the system of concern
- Requirements on user manual and service manual

Safe Speed Examples

- Fault Tolerance and Functional Degradation Requirements
 - “If it is found that the accuracy of the measured vehicle speed is poor or unknown, SSF shall be disabled” (REQ 2.008)
- Requirements for Functional Limitations
 - “The SSF deceleration (of wheels and vehicle) shall be limited to a safe level – safe both to the possibility of causing a rear-end crash and to the possibility that the vehicle might become unstable” (REQ 2.028)
- Requirements on Systems External to the System of Concern
 - “In the retarder controller, requests from ABS for reduced torque shall have a higher priority than Cruise/ACC requests” (REQ 2.003)

Ideal Properties of Requirements

Completeness: All relevant requirements are included

Non-ambiguity: The requirements are not open to interpretation

Consistency: The requirements do not contradict each other

Correctness: The requirements represent the desired behaviour or properties

Atomicity: Each requirement should represent a single "designable" entity

Verifiability: The requirements are formulated in a way that makes it possible to verify that they are fulfilled

Traceability: It shall be possible to trace between requirements at different hierarchical levels (for example system-level requirements and requirements on individual components) as well as between a hazard and the requirements related to this hazard. This tracing shall be possible in both directions.

A Safety Case

- Should „communicate a clear, comprehensive and defensible argument that a system is acceptably safe to operate in a given context“
- Reflects the Applied Safety Approach
 - prescriptive
 - goal based

Definition adapted from: T. Kelly and R. Weaver: *The Goal Structuring Notation – a Safety Argument Notation*.

Acceptably safe: Means that there is a tolerable risk remaining. The Safety Case should explain that the system is safe enough to operate.

Context: Context-free safety is impossible. It should be exactly declared which application in which environment is discussed.

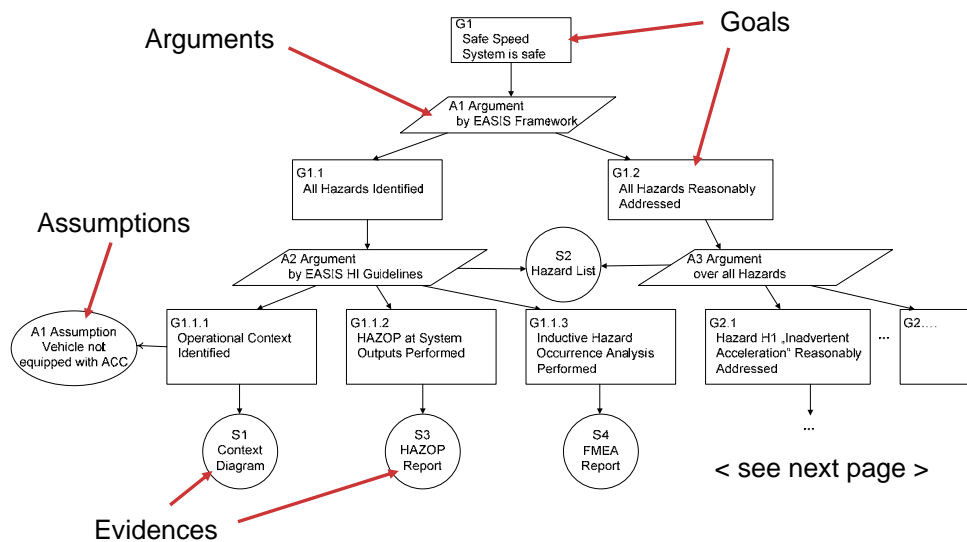
Clear: is about being understandable and having good structure.

Comprehensive: Comprehensive is about being acceptably complete, and will contribute to clarity by ensuring that the full argument is present.

Prescriptive Approach: Existing safety standards prescribe the requirements that have to be met in order to achieve the desired level of safety.

Goal-based Approach: Provide a claim and set up the requirements as needed by a specific argumentation that is not specified by a standard.

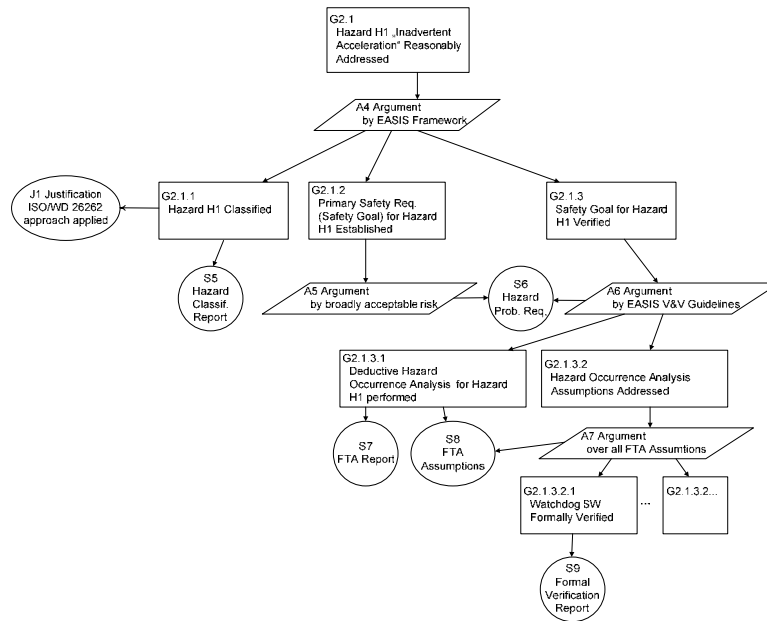
Safety Case Construction



The figure gives an example of a graphical goal-based safety case which uses the goal-structuring notation (GSN). A safety case is basically a chain of valid and sound arguments that support the safety claim by incorporating all necessary evidence and identifying the underlying assumptions

We call an argument **valid** if the conclusion can be logically derived from its premises. Otherwise we call the argument invalid.

We call the Argument **sound** if it is valid and all premises are true.



Safety Case Construction

- Modular Safety Cases
 - Allow to split Safety Case construction between OEMs and suppliers
 - Links between modules are defined by Safety Contracts
 - Also for COTS integration

Contents

- The EASIS Project
- Enabling Technology for ISS
- The EASIS Dependability Activity Framework
- Guidelines for the Dependability Activities
- **The ISS Application Development Process**
- Application of Formal Methods
- Conclusions

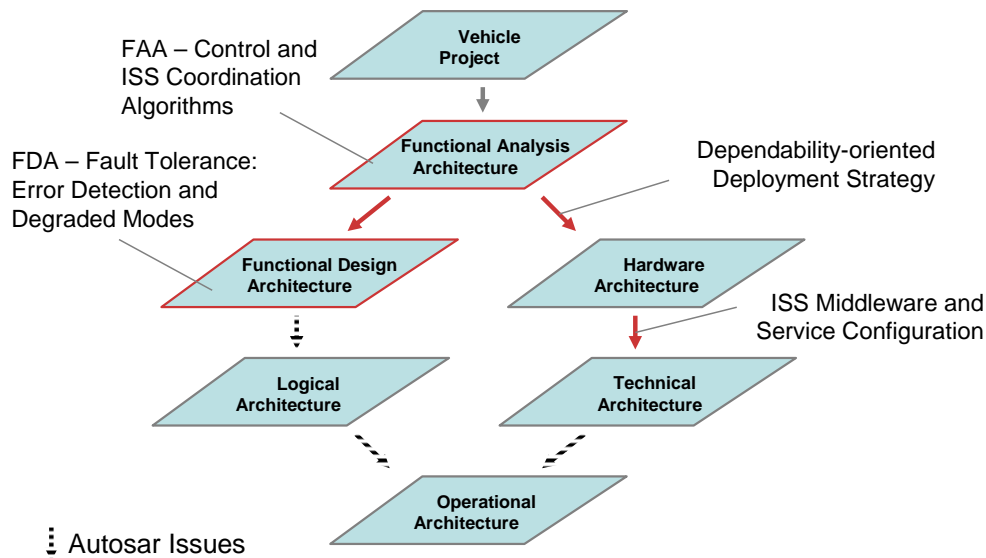
Mastering the Complexity

- Model-based Application Development Approach:
„EASIS Engineering Process“ (EEP)
- Based on EAST-ADL ideas as underlying ontology
- Tailored for complex ISS control algorithms with different levels of control
- Integrates dependability activity framework
- Enables formal verification

Goals of a Development Process for ISS

- The development must consider both functionality and safety
 - Integration of safety related requirements as an integral part of the requirements
- Process must be under control
 - Planning of activities with realistic time frames
- Effective consideration of safety related requirements
 - Perform the right methods using the right tools
- Efficient consideration of safety related requirements
 - Perform the methods at the right time

Software Artifact Ontology (EAST-ADL)

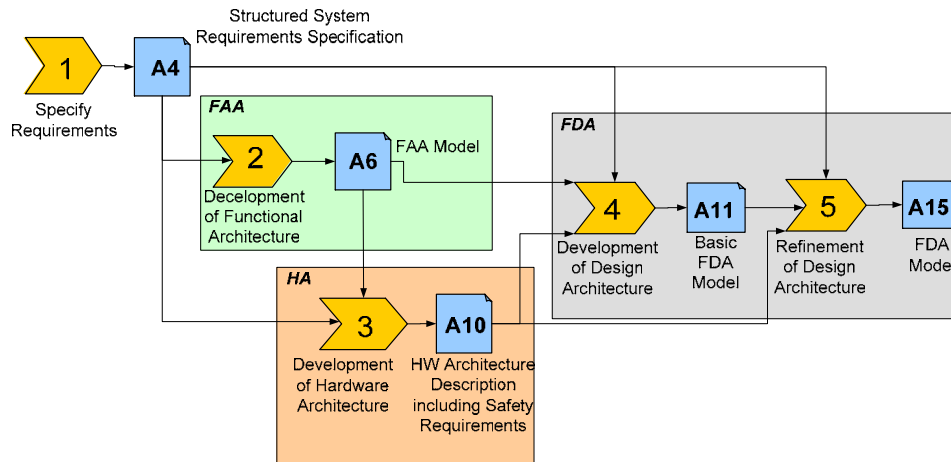


In philosophy, ontology is the most fundamental branch of metaphysics. It studies being or existence as well as the basic categories thereof, trying to find out what entities and what types of entities exist. Ontology has strong implications for the conceptions of reality.

Here, the ontology is a vocabulary or a conceptual framework to describe the functionality and features of an automotive electronic system. We use meta models to describe the structures and attributes of entities together with relations to other entities.

Using formalized models, reliable transitions from one process step to the other and from one development partner to another are possible.

The EASIS Engineering Process (EEP)



EEP helps to handle **complexity** by splitting up the system development into manageable parts:

1. **Specify Requirements:** here the dependability framework is instantiated for the first time („hazard analysis“)
2. **Development of functional architecture:** Consideration of basic functional behavior, ISS coordination
3. **Development of Hardware Architecture:** Safety-related requirements corresponding to hardware
4. **Development of Design Architecture:** Joining functional model and hardware architecture, initially without considering the safety requirements from step 3
5. **Refinement of Design Architecture:** Consideration of safety requirements from step 3

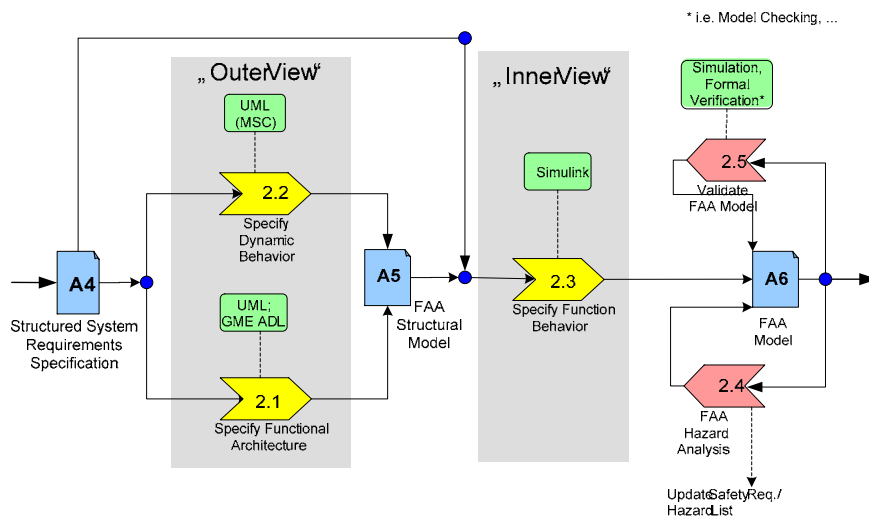
Frontloading of hazard analysis activities allows to avoid critical design flaws from the beginning. This reduces repetition of development cycles.

FAA: Functional Analysis Architecture

FDA: Functional Design Architecture

HA: Hardware Architecture

EFP: Develop Functional Architecture



To close the tool gap between the FAA Structural Model (UML) and the FAA Model (Simulink), a tool prototype („FAA enabler“) was developed within EASIS. The tool performs consistency checks on the structural model and does the transformation to Simulink. Furthermore, the tool allows to generate templates for an FMEA based on the structural model.

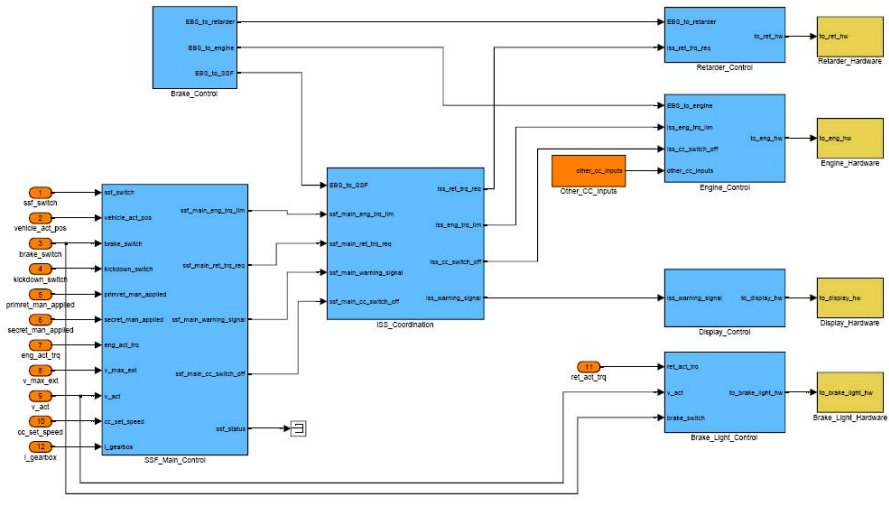
The ontology is a prerequisite for automating these transformation steps and the included consistency checks.

Already on FAA level, ISS coordination can be addressed. By ISS coordination we mean to introduce a higher-level control instance that mitigates conflicting actuator requests of different (safety-related) functions.

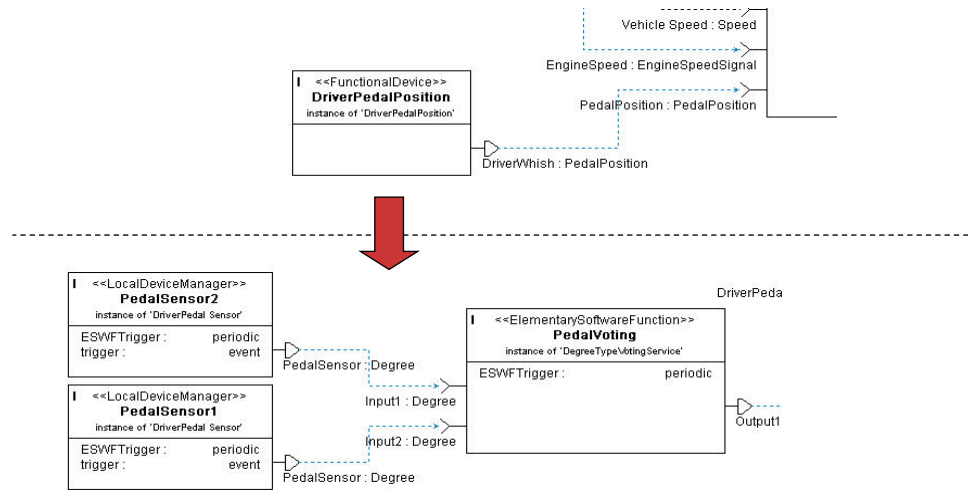
Thus the SafeSpeed system consists of two main sub-functions. One is the “SSF main control” which calculates an engine torque limit and a retarder torque request that are necessary to keep the vehicle speed limited to a maximum speed. Moreover, this sub-function can generate a cruise control deactivation command, a brake light activation command and a warning signal to the driver.

The second main sub-function is the “ISS coordination”. Here, some arbitration issues are dealt with. For instance, the retarder torque request from “SSF main control” is compared with the torque limit as demanded from the Antilock Brake System (ABS).

EFP: SafeSpeed FAA in Simulink



From FAA to FDA – Example

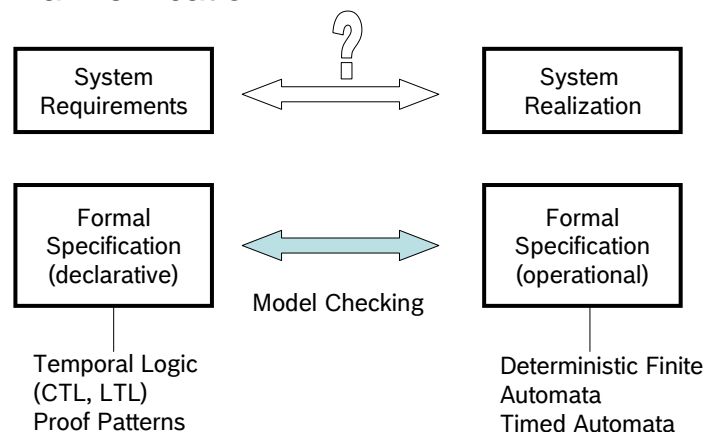


Contents

- The EASIS Project
- Enabling Technology for ISS
- The EASIS Dependability Activity Framework
- Guidelines for the Dependability Activities
- The ISS Application Development Process
- **Application of Formal Methods**
- Conclusions

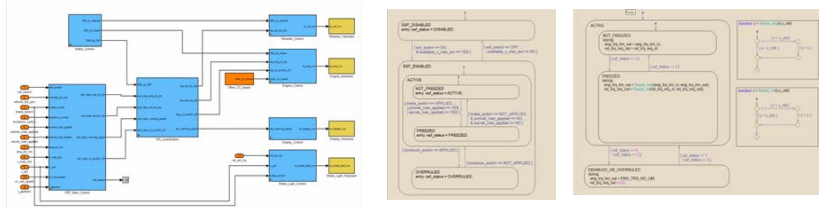
Formal Methods - Mastering Systematic Faults

- Formal Verification



Verification of the SafeSpeed FAA Model

- FAA Model in Simulink/Stateflow



- Model Checker: OSC EmbeddedValidator
- All textual requirements were investigated for feasibility of formal verification

Conclusions

- Positive
 - Formal verification helps finding ambiguousness in the original textual requirements
 - Counter-examples generated in case of a “FALSE” are very helpful
- Problems
 - Translation of textual requirements into temporal logic is difficult and requires a lot of experience

Textual: “The SSF is enabled when the dashboard switch is in the ON position AND the external commanded maximum safe vehicle speed is available”

Temporal logic:

$\text{inv_P_implies_finally_Q_B_after_reaching_R}$, where
P : $\text{ssf_switch} \ \&\& \ \text{available_v_max_ext}$
Q : $\text{in}(\text{SSF_ENABLED})$
R : $\text{in}(\text{SSF_DISABLED})$
max_X: 10

- Formal verification seems only applicable to verify parts of safety critical systems
- Formal verification can not verify requirements on “control characteristics” (overshoot etc.)

Formal Methods

- Guidelines for Application (“How”)
 - Check and improve applicability of formal verification
 - Choice of tool and integration into design process
 - Training and skills
 - Coverage analysis with respect to formal properties
 - Develop confidence in verification results (see below)
 - Management of proof obligations and results
- Maturity is still an issue
- Only a limited number of model-based tool chains is supported by now

Guideline Example: Develop Confidence in Verification Results

- If an invariant is violated, the model-checker produces a counter-example
- If the variant holds, the model-checker indicates success but **there is no further evidence available** (as e.g. a formal paper proof)
- To obtain further confidence in the verification result in case of a success result (no counterexample found), the “Mutation method” was proposed and validated at the SafeSpeed validator:
 - systematically weaken the assumptions and strengthen the commitments
 - in each case the model checker produces a counter-example
 - examine whether the counter-example exploits the weakened assumption or the strengthened commitment

Conclusions

- ISS will demand for a unified automotive dependability methodology
- EASIS gave a first sketch of which methods are
 - relevant
 - applicable
 - used in other industries
 - on the edge for transfer from academia to industry

Conclusions

EASIS provided

- A simple Framework for Dependability Activities
- Guidelines for the practitioner
 - Include state-of-the-art survey on major dependability activities
 - Include experience reports on formal method approaches
- A Process Framework for model-based ISS application development

Conclusions

EASIS Deliverables

- Dependability
 - D3.2, Part 1: Guidelines for establishing dependability requirements and performing hazard analysis, and constructing the safety case
 - D3.2, Part 2: Guidelines for verification and validation of dependability requirements – formal verification techniques
- Processes and Tools
 - D4.1: A software specific system engineering process for safety critical functions in the automobile
 - D4.2: A prototype tool interaction software layer
 - D4.3: Specification of the certification criteria for tools and tool interaction layers.

The EASIS results are available on www.easis.org

Acknowledgments

My et al. consists of:

Jürgen Lucas, ZF – **Olof Bridal**, VTEC – **Bernhard Josko**, OFFIS –
Malte Jacobs, Opel – **Joachim Stroop**, dSpace – **Ulrich Freund**, ETAS –
Henk Voets, DAF – **David Ward**, MIRA – **Michael Amann**, ZF – **Henning
Dierks**, OFFIS – **Andrea Piovesan**, CRF – **Christian Scheidler**,
DaimlerChrysler – **Paul Tiplady**, TRW –
Thorsten Kimmeskamp, University Duisburg/Essen –
Michel Leeman, Valeo – **Rosaria Anna Bray**, CRF –
Rudolf Huisman, DAF – **Alastair Ruddle**, MIRA –
Marc Graniou, PSA