ASSESSMENT OF THE ISO 26262 STANDARD, "ROAD VEHICLES – FUNCTIONAL SAFETY"

Dr. Qi Van Eikema Hommes

SAE 2012 Government/Industry Meeting

January 25, 2012

Outline

- ISO 26262 Overview
- Scope of the Assessment
- Strengths
- Considerations for Improvements
- Industry Feedbacks
- Summary

ISO 26262 Overview

- Adaptation of IEC 61508 to road vehicles
- Influenced by ISO 16949 Quality Management System
- The first comprehensive standard that addresses safety related automotive systems comprised of electrical, electronic, and software elements that provide safety-related functions.
- It intends to address the following important challenges in today's road vehicle technologies:
  - The safety of new E/E and Software functionality in vehicles
  - The trend of increasing complexity, software content, and mechatronics implementation
  - The risk from both systematic failure and random hardware failure

General Structure of ISO 26262

ISO 26262 affects all areas

1. Vocabulary

2. Management of functional safety

- **2-5** Overall safety management
- **2-6** Safety management during item development
- **2-7** Safety management after release for production

3. Concept phase

- **3-5** Item definition
- **3-6** Initiation of the safety lifecycle
- **3-7** Hazard analysis and risk assessment
- **3-8** Functional safety concept

4. Product development: system level

- **4-5** Initiation of product development at the system level
- **4-6** Specification of the technical safety requirements
- **4-7** System design
- **4-8** Item integration and testing
- **4-9** Safety validation
- **4-10** Functional safety assessment
- **4-11** Release for production

5. Product development: hardware level

- **5-5** Initiation of product development at the hardware level
- **5-6** Specification of hardware safety requirements
- **5-7** Hardware design
- **5-8** Hardware architectural metrics
- **5-9** Evaluation of violation of the safety goal due to random HW failures
- **5-10** Hardware integration and testing

6. Product development: software level

- **6-5** Initiation of product development at the software level
- **6-6** Specification of software safety requirements
- **6-7** Software architectural design
- **6-8** Software unit design and implementation
- **6-9** Software unit testing
- **6-10** Software integration and testing
- **6-11** Software verification

7. Production & Operation

- **7-5** Production
- **7-6** Operation, service and decommissioning

Core processes

Management

8. Supporting processes

- **8-5** Interfaces within distributed developments
- **8-6** Overall management of safety requirements
- **8-7** Configuration management
- **8-8** Change management
- **8-9** Verification
- **8-10** Documentation
- **8-11** Qualification of software tools
- **8-12** Qualification of software components
- **8-13** Qualification of hardware components
- **8-14** Proven in use argument

Support

9. ASIL-oriented and safety-oriented analyses

- **9-5** Requirements decomposition with respect to ASIL tailoring
- **9-6** Criteria for coexistence of
- **9-7** Analysis of dependent failures
- **9-8** Safety analyses

10. (Informative) Guidelines on ISO 26262

Scope of This Assessment

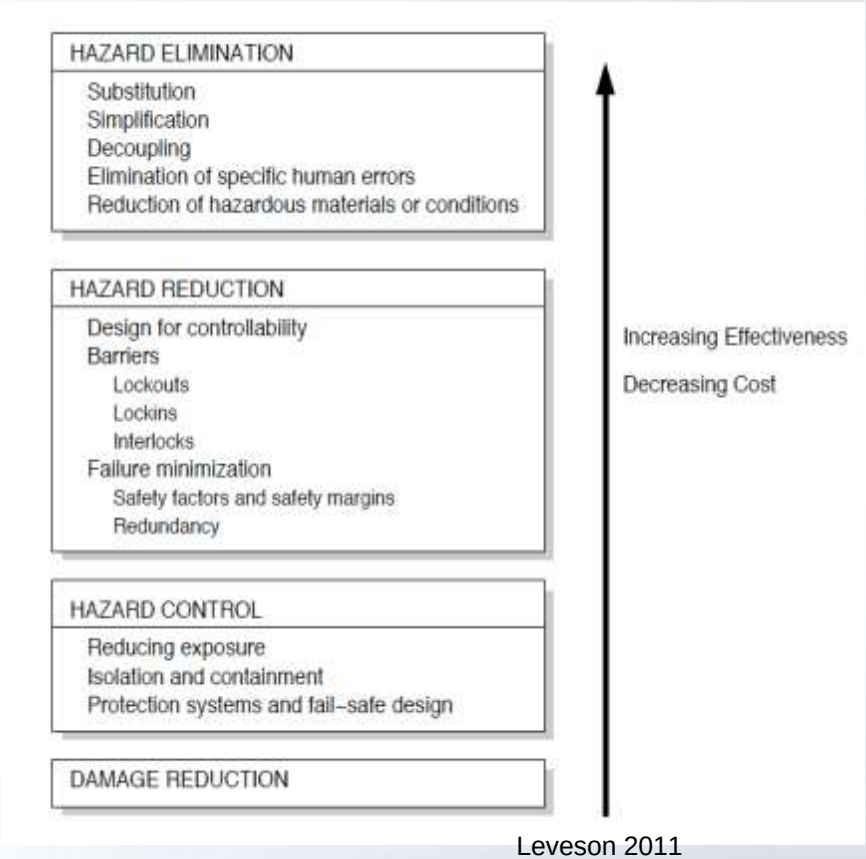
- Conducted in June-July 2011, based on DSI draft published in 2009.
- Final standard (FDIS) was published in November 2011.
- Future discussions should be based on the FDIS version of the standard.
- Review Focus—How well can the standard provide safety assurance for the complex software-intensive automotive electronics and electrical systems?

Strengths

- Emphasizing safety management and safety culture
  - Major accidents in large complex systems are not caused by the failure of technical components, but rather organization factors influencing the design, manufacturing, and operation of the system.
- Prescribes a systems engineering process
  - Safety is an emergent property of the system, and requires systems engineering approach.

Strengths

- Departure from safety as an after-thought:
  - IEC 61508: safety function
  - ISO 26262: provides the framework and vocabulary for hazard elimination in the first place
    - Systems engineering framework
    - Safety measure vs. safety mechanisms



Leveson 2011

Strengths

- Disassociate hazard risk level from probabilistic failure rate:
  - IEC 61508: SIL uses component failure rate

Severity

Class	S0	S1	S2	S3
Description	No Injuries	Light and Moderate Injuries	Severe and Life-threatening Injuries(survival probable)	Life-threatening injuries (survival uncertain), fatal injuries

Exposure

Class	E0	E1	E2	E3	E4
Description	Incredible	Very low probability	Low probability	Medium Probability	High Probability

Controllability

Class	C0	C1	C2	C3
Description	Controllable in general	Simply controllable	Normally controllable	Difficult to control or uncontrollable

ASIL

		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

D: highest safety integrity level
A: lowest safety integrity level
QM: quality management

ASIL Assessment

- Consider only use S (severity)
  - Estimation of E can be subjective
- Standardize ASIL assessment among OEM and Suppliers
  - Legal liability of different ASIL assessment
  - Development cost affecting industry competitiveness
  - OEM inconsistency among the same component from the suppliers.
  - Must be careful with component ASIL standardization—safety of a component can only be assessed in the context of the specific system implementation.
  - The government may play a role in ASIL classification.

Provide More Guidance on Hazard Elimination

- Safety measure is not clearly explained in the document. But this concept is the key to hazard elimination in the first place—the most effective and least costly approach.
- Safety Mechanism is explained in detail throughout the document. But this concept is like the safety function concept in IEC 61508, and is a less effective safety measure.
- The standard may want to add a section in Part 1 to further clarify the departure from IEC 61508's design philosophy.
- Investigate other hazard analysis methods that can provide more effective guidance on how to identify and eliminate hazards in design. One such method is the System Theoretic Process Analysis (STPA) based on System Theoretic Accident Modeling Process (STAMP).

Separate System Safety from Reliability Engineering

Safety ≠ Reliability

- Reliability engineering focuses on component failures.
- System can be unsafe when none of the component fails.
  - The required function may be unsafe.
  - Software or human do not fail, and have no failure rate.
- System can still be safe when components fail.

Reliability Engineering Methods in ISO 26262

- **Hardware Architecture Metrics--**Based on random failure of components.
- **Failure Modes and Effects Analysis (FMEA)**:
  - Developed to predict equipment reliability.
  - Forward search based on underlying single chain-of-events and failure models
  - Initiating events are failures of individual components
  - Quickly become impractical for complex systems
- **Fault Tree Analysis (FTA)**:
  - Top-down search method
  - Based on converging chains-of-events accident model.
  - Tree is simply record of results; analysis done in head. Lack of guidance.
  - Assume independence of the failure, which is not always true.
- **Safety Case Approach**
  - Confirmation bias
  - the use of Quantitative Risk Assessment
  - Independent reviewers are less familiar with the design

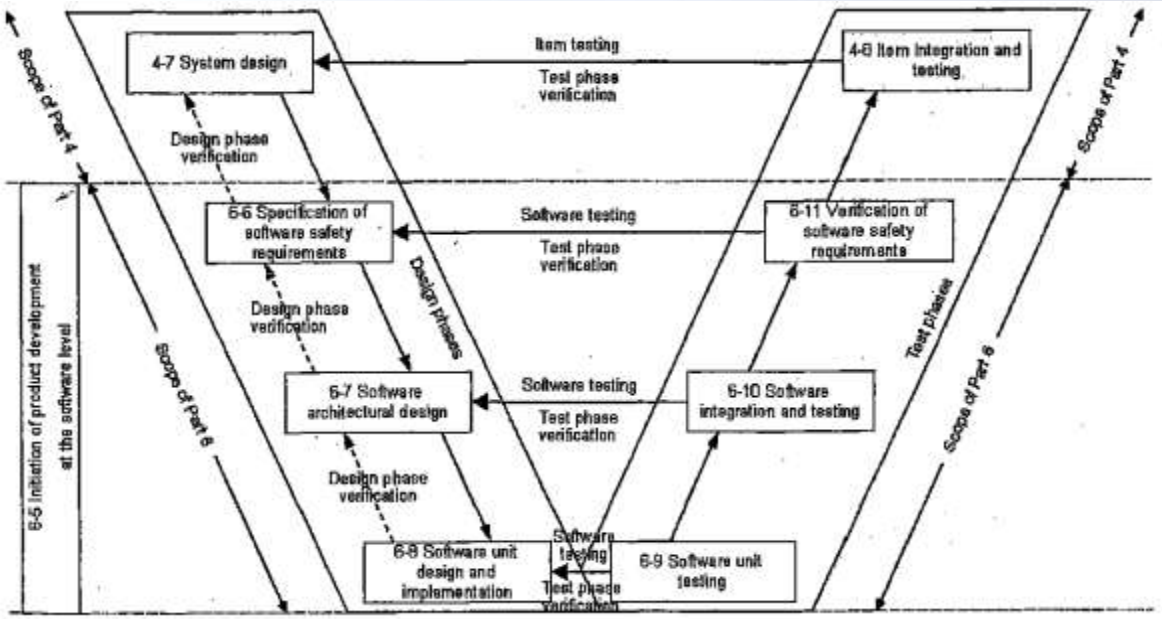
The FMEA and FTA comments are based on Professor Nancy Leveson's system safety class lecture notes.

Recommendations for Strengthening Safety Engineering

- Independent review team may not want to focus on the correctness of the safety case, but rather independently conduct hazard analysis to find out whether all causes of the hazards have been analyzed and addressed.
- Investigate the effectiveness of STPA and other system safety methods in order to adapt them into the standards to provide guidance on how to design for system safety.

Software Safety

- Follows software system engineering process
- Promotes good software architecture practices
- Best practices in software design
- Addresses hardware failure
- On Par with other software safety standards such as DO-178



Comments:

- Unlike hardware, software does not fail.
- Software faults are due to design errors, but the standard does not offer a way to identify design errors that can cause hazard.
- Good systems engineering process and software architecture design are necessary but not sufficient to ensure system safety.

Production and Operation

- Great to have a complete lifecycle view of the safety.
- Unfortunately, the standard provide almost no guidance on how to ensure the safety in these two important stages.
- Thought starters:
  - How to identify key characteristics in manufacturing that are critical to safety?
  - Aftermarket software and components safety?
  - How to check and ensure sensors, actuators, and communication channels are safe throughout the lifecycle of the vehicle?
  - Is there a need for Government-mandated yearly safety checkup?
  - Education and training for service technicians?

Incorporate Design Guidelines for Safe User Interaction

- The standard has no mention of safe user interactions
- Automation in the airplane cockpit has led to some major accidents
- Automotive companies do have Human-Machine Interface design groups
- Recommendations:
  - Learn from the accidents in cockpit automation literatures
  - Incorporate guidelines for safety user interaction design in the future

Industry's Views—Pro's

- ISO 26262 is well regarded by industry and is seen as necessary.
- Many companies have at least tried it on pilot projects.
- GM has used it to ensure Volt's battery functional safety.
- Industry recognizes it is valuable to have safety standard to address the growing complexity of Cyber-Physical Systems.
- No discrepancy with mature product development process, and it is easy to implement.
- Aligns well with the model-based development process.

Industry's Views--Cons

- Amount of documentation efforts
- Not convinced that the software development methods are sufficient to guarantee safety
- Since the standard is about the entire product life cycle, the effect of the standard will take some time to show.
- The concept phase is easy to implement, but there is difficulty to integrate a pilot project into the rest of the system that was not developed based on the standard.
- ASIL classification harmonization
- "Proven in use" argument is not useful
  - Takes too long to collect sufficient data
  - The definition in the standard makes it a step that will never be visited
- Qualification of software tools
  - The large number of software tools used in development
  - Comment: software tools are software. How will one quantify the probability of software making mistakes?

Summary of Recommendations

1. Consider only using severity for ASIL assessment
2. Government may want to consider playing a role in ASIL standardization
   - However, the ASIL assessment must depend on the context and the design configuration of the system.
3. The standard may want to add a section to emphasize hazard elimination before detection and control
4. Research activities may want to investigate the effectiveness of system theory based hazard causal analysis in automotive complex cyber-physical systems
   - E.g. STAMP model and STPA.
5. Fundamental research is needed for the safety of complex software-intensive systems today, including those in the current automobiles:
   - The effect of complexity on safety is not well quantified
   - The effects of software engineering best practices on safety may be insufficient to ensure safety. New and different approaches may need to be developed.
6. Government may want to play a role in certifying software tools used for the development of safety critical systems
7. Government may want to consider regulating the safety of E/E systems after the vehicle is sold