Verified and robust integration of safety critical functions

Reinder J. Bril
Overview

• Background and motivation
  – Historic perspective
  – Examples
  – Automotive architectures
  – From *federated* to *integrated* architectures

• Requirements
  – Verification
  – Robustness

• An example application: Active suspension

• Conclusion
Historic perspective

• 1977 General Motors Oldsmobile Tornado:
  – 1\textsuperscript{st} production automotive ECU
    • Electronic spark timing.

• $\approx$ 1990 (Mercedes, BMW, Audi, and VW)
  – Up to 5 ECUs, point-to-point connections;
  – Issues: expensive to fit, heavy, faulty connectors.

• 1991 Mercedes S-class:
  – 1\textsuperscript{st} deployed CAN (Controller Area Network).

• $\approx$ 2000 (Mercedes, BMW, Audi, and VW)
  – Around 40 ECUs and multiple busses.
Historic perspective

• Today:
  – 100+ ECUs and 5-7 busses;
  – > 100 MLOCS (Million Lines Of Code).

Manufacturing costs: 15 – 40%;
Innovation costs: 80%.

(by courtesy of Daren Buttle)
Examples

(by courtesy of Thomas Nolte)
Automotive architectures

- Example network architectures

- Trends:
  - Reduction of the number of ECUs
  - Functionalities sharing ECUs
  - Integration of functionalities and domains
From *federated* to *integrated* architectures

Step 1: Inter-application isolation on ECU.

(by courtesy of Martijn M.H.P. v.d. Heuvel)
From *federated* to *integrated* architectures

Step 1: Inter-application isolation on ECU.
Step 2: Replace CAN by FlexRay, isolating network traffic.
Requirements

• Verification:
  – Individual applications;
  – Composition of applications;
  – Temporal analysis of components and platforms
  – Protocol analysis (e.g. network specifications).

• Robustness:
  – Inter-application isolation on ECU;
  – Inter-application isolation on network;
  – Invalid input data;
  – Network loss.
Verification

• Model-based development (TNO):
  – Matlab/Simulink: control algorithms verified independent of target platform;

• ASD (Analytical Software Design) (Verum):
  – Behavioral correctness guaranteed for composition of software components;

• Temporal analysis for virtual platforms (TU/e);

• FlexRay start-up protocol (TU/e).
Robustness

• Inter-application
  – ECU (TU/e):
    • Virtual processor: isolate applications on ECU;
  – Network (NXP):
    • FlexRay: isolate applications on network;
    • Active Star: isolation of an erroneous branch.

• Distributed AS Application Functionality:
  – Control loop values monitoring (TNO):
    • Central/local control: application mode change;
  – Network loss detection (Verum):
    • Supervisory control: application mode change.
An example: Active suspension

- Mechanics: Suspension struts
An example: Active suspension

- Mechanics: Suspension struts & hydraulic system

Hydraulics system

Jaquar XF
An example: Active suspension

- Mechanics: Suspension struts & hydraulic system
- Control system
  - Step 0: Centralized $\rightarrow$ distributed;
  - Step 1: Inter-application isolation on ECU;

Jaquar XF
An example: Active suspension

- Mechanics: Suspension struts & hydraulic system
- Control system
  - Step 0: Centralized → distributed;
  - Step 1: Inter-application isolation on ECU;
  - Step 2: CAN → FlexRay, isolation on network.

Jaquar XF
An example: Active suspension

• “Embedded” demonstrations:
  – December 2012:
    • Step 0: Centralized → distributed;
      – Integrated functionality (Verum, TNO, TU/e);
      – Distributed AS functionality:
        » CAN: Network-loss detection (Verum);
        » Mode change (TNO);
    • Step 1: Inter-applications isolation on ECU (TU/e).
  – November 2013:
    • Step 2: CAN → FlexRay, isolation on network.
An example: Active suspension

• “Embedded” demonstrations:
  – November 2013:
    • Step 2: CAN → FlexRay, isolation on network
An example: Active suspension

• “Embedded” demonstrations:
  – November 2013:
    • Step 2: CAN → FlexRay, isolation on network
    • Scenarios:
      1. FlexRay: Disconnect and reconnect;

Central dSpace box

“auxiliary” ECU

Active Star

“embedded” ECU
An example: Active suspension

- “Embedded” demonstrations:
  - **November 2013:**
    - Step 2: CAN → FlexRay, isolation on network
    - Scenarios:
      1. FlexRay: Disconnect and reconnect;
      2. Active star: isolate erroneous branch.
An example: Active suspension

• Acknowledgements:
  – NXP: Abhijit Deb;
  – Verum: Henk Katerberg;
  – TNO: Can Cilli;
  – TU/e: Erik Luit, Richard Verhoeven, Mike Holenderski, Sjoerd Cranen, Daan Gerrits.
Conclusion

• Vision
  – From *federated* to *integrated* architectures

• Requirements
  – Verification and robustness

• Demonstrator
  – Verification
    • Model-based development (TNO);
    • Analytical Software Design (ASD, Verum);
  – Robustness:
    • Inter-application isolation on ECU (TU/e);
    • Isolation on network (NXP);
    • Control-loop values monitoring (TNO);
    • Network-loss (Verum).