Automotive System Testing by Independent Guarded Assertions

Thomas Gustafsson
thomas.gustafsson@scania.com
Scania CV

Mats Skoglund, Avenir Kobetski, Daniel Sundmark
firstname.lastname@sics.se
Swedish Institute of Computer Science

Outline

- Context, problem formulation
- Background, ”traditional” test scripts
- Independent guarded assertions (IGA)
- Experiment, results, conversion trad. to IGA
- Conclusions and future work

Context

Why

- Better real-world representativeness
- Shorter testing time
- Improved defect detection

Background

- Scripted scenario based testing of user functions
- Regression testing in batch runs during the night
- The HIL is configured as a vehicle individual out of millions of possible ones

Test script

- Linked to user function, use case, and scenario
- A test script has the following structure
  - `pre – perform actions so testing can start`
  - `act1 – mix of stimuli and testing`
  - `…`
  - `actN`
  - `post – perform actions so other scripts can reach their states`

Test script - example

- UF068: Work light
- `act1`
  - Engage reverse gear and work light
  - Test that work light is lit
- `act2`
  - Disable work light
  - Test that work light is not lit
- `act3`
  - Enable ”sticky” work light – work light shall be enabled if reverse is enaged – by pressing button more than 3 sec
  - Test that work light is lit

Shortcomings

- Test scripts always execute the same sequences of stimuli and tests
- Test scripts test each act only one time
- Test scripts do not test combinations of user functions
  - A new test script explicitly doing so is needed

- Solution: execute test scripts concurrently during a predefined course
  - Better real-world representativeness
  - Shorter testing time
  - Improved defect detection

Independent Guarded Assertions

- An act can be written as a traditional guarded command
  - if precondition for testing is fulfilled => perform tests

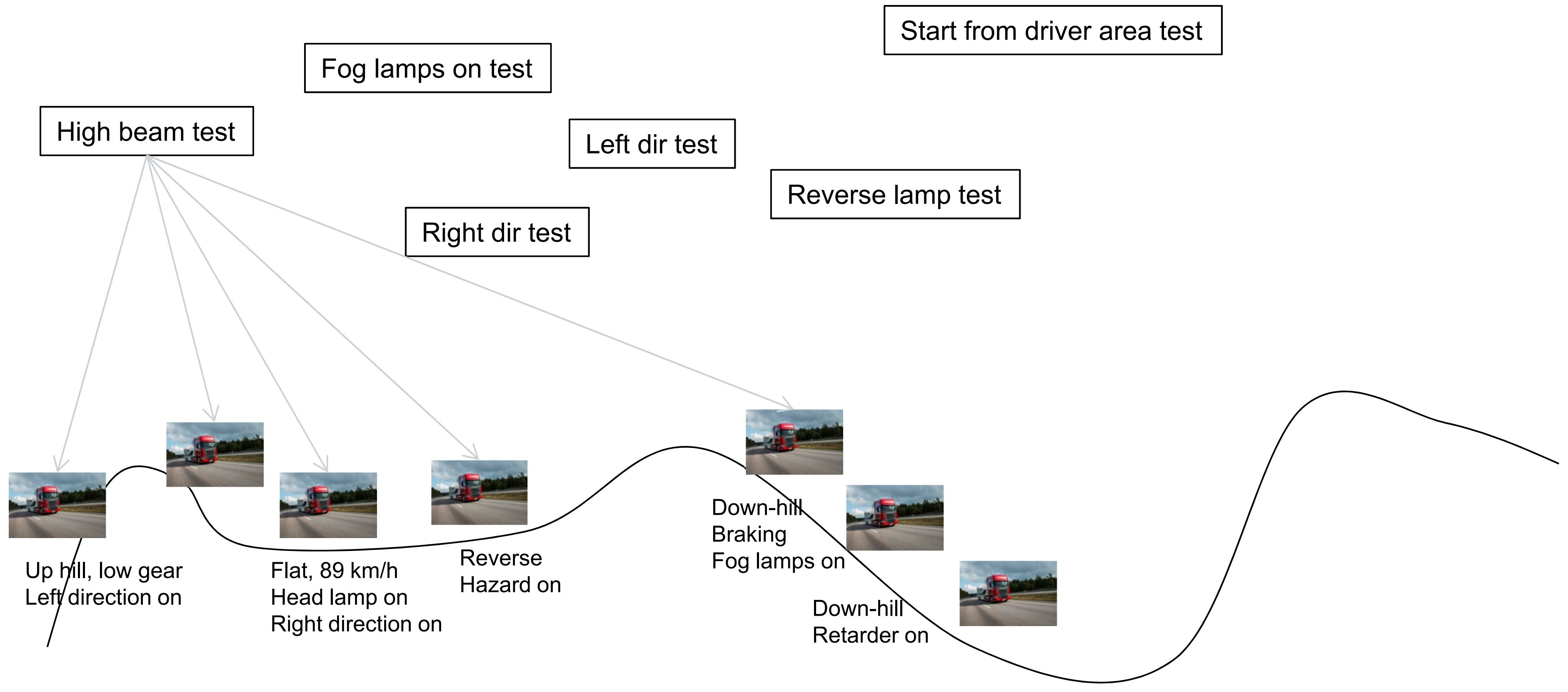
- No stimuli is needed in such a script
  - Each act becomes read-only

- **Independent**: acts no longer depend on stimuli from itself or other acts
- **Guarded**: acts decide when to perform testing
- **Assertions**: the actual testing of the SUT

Independent Guarded Assertions - Example

- Thread 1
  - Loop
    - status = button is pressed
    - ts = time
- Loop
  - EventWait CAN.msg2.signal3 == 1
  - Thread 1’s status == 0 and ts > time + 0.2
  - assertEqual CAN.msg1.sig1 == 1

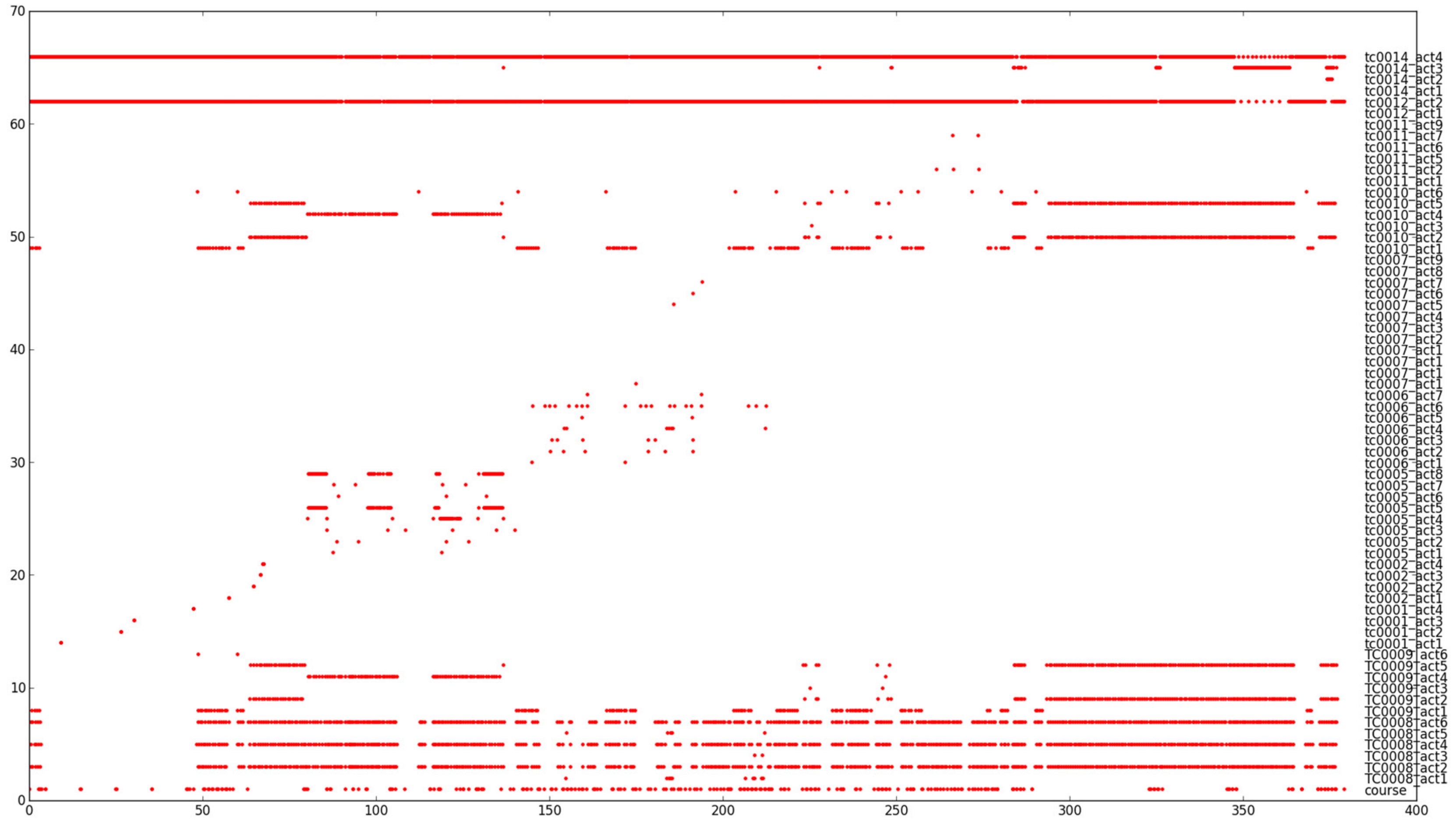
Overview


Start from driver area test
Fog lamps on test
High beam test
Left dir test
Reverse lamp test
Right dir test
Up hill, low gear
Left direction on
Flat, 89 km/h
Head lamp on
Right direction on
Reverse
Hazard on
Down-hill
Braking
Fog lamps on
Down-hill
Retarder on

Experiment

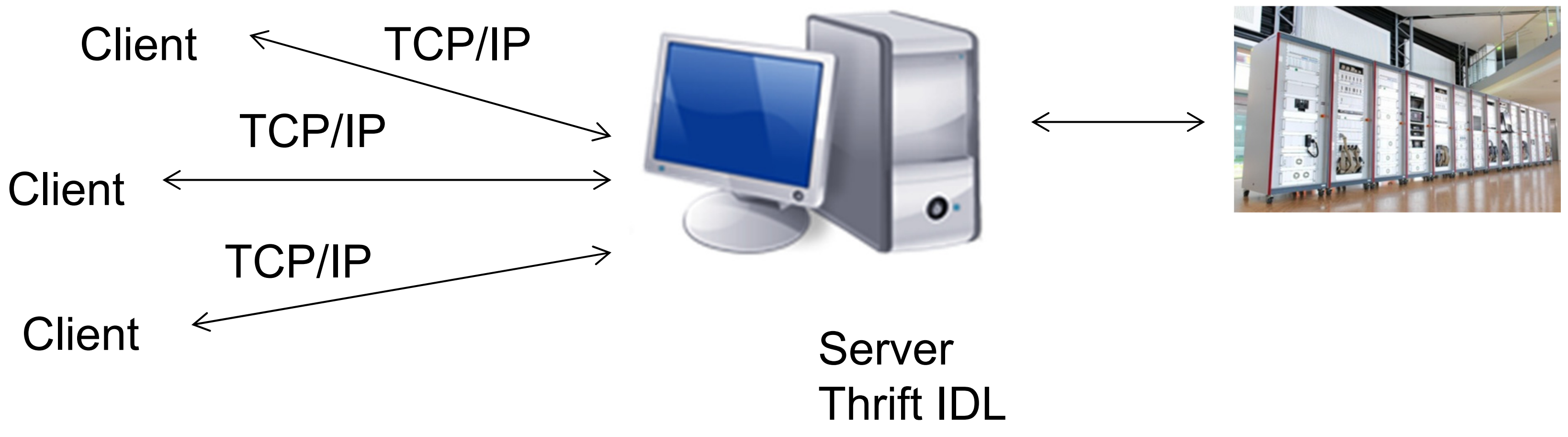
- 10 of Scania’s test scripts were transformed into 68 independent guarded assertions. One per act
- One course was constructed by taking the stimuli from the 10 test scripts and executing it in the same order
- The course is not optimized, but is our baseline
- Scania’s test automation framework executed the 69 scripts concurrently

Results


70
60
50
40
30
20
10
0
0
50
100
150
200
250
300
350
400
tc0014_act4
tc0014_act3
tc0014_act2
tc0014_act1
tc0012_act2
tc0012_act1
tc0011_act9
tc0011_act7
tc0011_act6
tc0011_act5
tc0011_act2
tc0011_act1
tc0010_act6
tc0010_act5
tc0010_act4
tc0010_act3
tc0010_act2
tc0010_act1
tc0007_act9
tc0007_act8
tc0007_act7
tc0007_act6
tc0007_act5
tc0007_act4
tc0007_act3
tc0007_act2
tc0007_act1
tc0007_act1
tc0007_act1
tc0007_act1
tc0006_act7
tc0006_act6
tc0006_act5
tc0006_act4
tc0006_act3
tc0006_act2
tc0006_act1
tc0005_act8
tc0005_act7
tc0005_act6
tc0005_act5
tc0005_act4
tc0005_act3
tc0005_act2
tc0005_act1
tc0002_act4
tc0002_act3
tc0002_act2
tc0002_act1
tc0001_act4
tc0001_act3
tc0001_act2
tc0001_act1
TC0009_act6
TC0009_act5
TC0009_act4
TC0009_act3
TC0009_act2
TC0009_act1
TC0008_act6
TC0008_act5
TC0008_act4
TC0008_act3
TC0008_act2
TC0008_act1
course

Test automation framework

- Client/server architecture. The server accepts multiple clients. The server is thread safe
- The clients are indepedent and run in their own processes
- The clients execute on the same PC as the server, but could execute on other PCs



SCANIA

Transformation to guarded assertions

- Mismatch in results and some acts did not execute at all
- The mistake is in the guards
- Studied tc0005_act4 in more detail, which is one of the more complex guards in the 10 acts
  - *Test that if the work light is disabled, the gear is changed from reverse to neutral, and regardless of state of ”sticky” work light, then the reverse light shall be off*
- Mix of stimuli and test
  - Set gear to neutral
  - Disable work light function
  - Test that work light is off

Transformation to guarded assertions

- The guard is implemented as
  - one thread monitors the state of the work light function
  - one thread monitors the state of the gear
  - one thread fuses the information and decides when to test
- The threads are part of the same process, and need to be thread safe
- The guard must correctly describe all possible states where testing can be performed
- Moreover, the guard must consider propagation times of signals/functions

Conclusions

- Concurrent execution of scripts is a prerequisite for achieving the stated goals
- Independent guarded assertions
  - Concurrency
  - Multiple tests, but coverage depends on the course
  - The guards can be complicated to express in a general way

Future work

- Automatic construction of a course given an objective
- Execute a script against offline data
  - Set up HIL, install all ECU SW, and parameterize it
  - Run a course and collect data
  - Run data in a cloud, while switching to the next vehicle individual
- Independent guarded assertions focus on *what and when* to test, not *how* to test
  - What abstraction level is best suited for describing guards