Continuous Integration and Validation Using Closed-Loop SiL Simulation for Propulsion Control and Calibration

Mattias Eriksson, Johannes Foufas, Neda Nickmehr
Volvo Cars, Göteborg, Sweden

Michael Hartmann
QTronic GmbH, Berlin, Germany
IN THE BEGINNING

- Started 2002
- 10 model developers
- SourceSafe/Vault
  - Used as a network storage
- Manual starting bat files for code gen and building
Prior to the current engine generation, most tests were done in the car.

During the development of the current engine generation, automatic unit and system tests were introduced.

- Aftertreatment SW solely developed in Sil platform.
- One senior SW developer said: *now I know it will work when we test in the car...*

**timeline:**

- **2009**: CAE Engineers use Silver to run plant models.
- **2014**: Sil Pilot on current Engine platform.
- **2015**: Server automation and education.
- **2016**: Ramp up to 100 users, now also in more projects.
- **2017**: Pilot with explorative tests on next Engine platform.
- **2018**: Gating CI system based on git/gerrit/zuul. The first for model based development at Volvo Cars Corporation.
The Software in current generation ECMs is structured into around 500 modules.

A small part of the application code is still made by the HW suppliers.
CI/CD SYSTEM

- Ensuring the integrity is not dependent on a single individual making the right call. It is ensured by the system.
- **Fast Feedback**, small changes often, automatic testing
- **Transparency**, Follow your commit
- **CI/CD system as code**, using Python plugin Jenkins job builder and YAML files as pipeline configuration
CI/CD System

CI system structure
Build dll, unit tests, Merge tests and Exploratory tests executed by Silver and TestWeaver

Diagram:
- Developer Commit
- Code Gen
- Check Tests
- TL Check script
- MISRA, MXAM
- Complexity Analysis, MODAFY
  - Signal Consistency
- Unit Test
- Build dll
- Merge tests
- Exploratory tests executed by Silver and TestWeaver
- Gerrit Code Review +1
- Gerrit PO Review +2
- Code Gen
- Build dll
- Merge Tests
- Compile
- Build dll
- Smoke Test
- Function Test (SSP SIL)
- Integration Test (SP SIL, SP HIL)
- SF Release based on git tag
- Exploration Tests
- YES: SW for production
- NO: Post Merge Jobs

- To supplier: RxDB, integration with BSW
- Calibrating + Testing (HIL, Engine Rigs, Car)
- To supplier: RxDB, generate VBF
- Test VBF
  - From RxDB to PIE R&D
- Complete CI

Diagram details:
- Automatic step
- Manual step
- Step to be implemented
CI/CD SYSTEM
**CI/CD System**

![CI/CD System Interface](https://pt.gerrit.cm.volvocars.biz/#/r/2079/)

<table>
<thead>
<tr>
<th>Change 2079 - Merged</th>
<th>CI/CD System Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Owner</strong></td>
<td>Nilsson, Zoul</td>
</tr>
<tr>
<td><strong>Uploader</strong></td>
<td>Nilsson, Zoul</td>
</tr>
<tr>
<td><strong>Assignees</strong></td>
<td>Nilsson, Olsson, Zoul</td>
</tr>
<tr>
<td><strong>Reviewers</strong></td>
<td>Nilsson, Zoul</td>
</tr>
<tr>
<td><strong>Project</strong></td>
<td>pt Proj</td>
</tr>
<tr>
<td><strong>Branch</strong></td>
<td>master</td>
</tr>
<tr>
<td><strong>Topic</strong></td>
<td>Fix</td>
</tr>
<tr>
<td><strong>Updated</strong></td>
<td>16 hours ago</td>
</tr>
<tr>
<td><strong>Cherry Pick</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Files**

- Commit Message: Merge _Parfus3.m_
- ScriptFunctions/merge_Parfus3.m
- ScriptFunctions/TL208_Gui.m

**History**

<table>
<thead>
<tr>
<th>Author</th>
<th>Action</th>
<th>Comments Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nilsson</td>
<td>Uploaded patch set 1</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Starting jobs. github/1415-got.volvocars 0098</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Opcheck Results. github/1420-got.volvocars 0098</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Opcheck Results. github/1420-got.volvocars 0140</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Opcheck Results. github/1420-got.volvocars 0140</td>
<td></td>
</tr>
<tr>
<td>Olsson</td>
<td>Patch Set 1: Code-Review+1</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Build succeeded. github/1420-got.volvocars 0144</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Verified starting jobs. github/1415-got.volvocars 0098</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Merge Tests Report. github/1420-got.volvocars 0139</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Merge Tests Report. github/1420-got.volvocars 0139</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Patch Set 1: Build succeeded. github/1420-got.volvocars 0140</td>
<td></td>
</tr>
<tr>
<td>Zoul</td>
<td>Change has been successfully released and submitted. github/1420-got.volvocars 0140</td>
<td></td>
</tr>
</tbody>
</table>
Zuul Status

Real-time status monitor of Zuul, the pipeline manager between Gerrit and Workers.

Queue lengths: 0 events, 0 results.

Filters: Expand by default: 0

check

Nearly uploaded patchsets enter this pipeline to receive an initial +A1 Verified vote from Jenkins. You can retrigger this check by commenting “recheck” in gerrit.

Queue: pt_pcc

Gate

Changes that have been approved by core developers are enqueued in order in this pipeline.

Queue: pt_pcc

build_mxam_report

Generate MXAM report, start this by commenting “build_mxam_report” in gerrit.

build_gep3_...

Build GEP3_SPA_TODC, start this by commenting “build_gep3_spa_TODC” in gerrit.

build_gep3_...

Build GEP3_SPA, start this by commenting “build_gep3_spa” in gerrit.

build_gep3_...

Build GEP3_HRP, start this by commenting “build_gep3_HRP” in gerrit.

build_gep3_...

Build GEP3_HRP, start this by commenting “build_gep3_HRP” in gerrit.

build_gep3_...

Build GEP3_HRP, start this by commenting “build_gep3_HRP” in gerrit.

build_gep3_...

Build GEP3_HRP, start this by commenting “build_gep3_HRP” in gerrit.

build_gep3_...

Build GEP3_HRP, start this by commenting “build_gep3_HRP” in gerrit.
SIL, THE CORE OF THE CI SYSTEM

Physical Testing

EP   CD   FPD   HIL   SIL

Number of tests

Virtual Testing

Standardized Data Collection
Plant Modelling
Model Calibration
Conceptual SW design
Integrated SW design
Diagnosis development
SW Validation
SW Calibration
Attribute Validation
HW and diagnostic validation

expensive

SIM   MIL   SIL   HIL   RIGS   ROAD
**SIL TEST LEVELS**

- Unit-, Module- and ECM-level tests
- ECM includes supplier SW
- With or without plant models
• All SIL tests – from open-loop unit tests to closed-loop ECM tests are using the same toolchain
• Tests defined in TestWeaver
  • Stimulus files in Python
  • Test-level invariant requirement watchers (RML)
• Test execution in Silver

```python
w = WatcherFSM( tw,
    'Watcher_engine_max_speed',  # name in TestWeaver
    description = ('Checks if the engine RPM does not overspeed'),
    while_condition = (lambda: True, ''),
    pass_condition = (lambda: rpm.Value < 5000, 'Check engine RPM.'),
    autoreset = False,  # continue to check the pass_condition
    tolerance_time = 1)
w.set_req_id('')
```
ECM SIL - ENGINE PLANT MODEL

sensors

actuators

Engine Control Module
SIL dll
In-house developed Dymola model for torque, air-charge and cooling systems
Supplier catalyst models, blackbox
Data-driven emission models
ETAS ASCMO, Neural networks
ECM SIL - THE FULL SYSTEM

Engine

Engine Control Module
SIL dll

Transmission

Vehicle

Driver
ECM SIL – THE FULL SYSTEM

How to integrate and simulate all this together?

- Plant models run as Functional Mockup Units (FMU:s)
  https://fmi-standard.org/
- Integrated in QTronic Silver
Results, WLTC

- WLTC cycle comparison
- ECM SIL vs. Prototype vehicle rig measurement
- Input: WLTC velocity profile
- Including emissions
- Focus on triggering software functionality, not absolute quantities

Vehicle speed, WLTC
RESULTS, WLTC

- EGR flow
- LNT mode
- LNT NOx storage
- Ad-blue dosing
- NOx Engine out
- NOx Tailpipe
THANK YOU!