Extending OMNeT++ Towards a Platform for the Design of Future In-Vehicle Network Architectures

Till Steinbach    Philipp Meyer    Stefan Buschmann    Franz Korf

Hamburg University of Applied Sciences
philipp.meyer@haw-hamburg.de

OMNeT++ Community Summit
15. September 2016, Brno University of Technology, Czech Republic
1 Introduction

2 Simulation Models

3 Toolchain

4 Workflow

5 Conclusion & Outlook
Introduction

- In-vehicle networks face a paradigm change
- Communication architectures today:
  - CAN
  - FlexRay
  - LIN
  - MOST
- Switched real-time Ethernet is promising candidate for future communication architectures\(^1\)
- Stepwise transition from heterogeneous bus architecture towards a single flat Ethernet topology

\(^1\) Kirsten Matheus and Thomas Königseder: *Automotive Ethernet*. Jan. 2015.
Introduction

- Current tools focus on bit-correct fieldbus simulation
- New tools are required for design and evaluation
- These environments have to support analysis of congestion and jitter
- The OMNeT++ platform provides a perfect base
- We want to provide an easy to use environment
- In this work we contribute:
  - Simulation models
  - Tools to design
  - Tools to evaluate
  - A uniform workflow
1 Introduction

2 Simulation Models

3 Toolchain

4 Workflow

5 Conclusion & Outlook
Extending OMNeT++ Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook

Simulation Models
Overview

Vehicle network model

<table>
<thead>
<tr>
<th>CoRE4INET</th>
<th>FiCo4OMNeT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real-time Ethernet</td>
<td>Fieldbusses (CAN, FlexRay)</td>
</tr>
</tbody>
</table>

INET Framework
Internet Technologies / Protocols

OMNeT++
IDE and Simulation kernel

- CoRE4INET
  (Communication over Real-time Ethernet for INET)
- FiCo4OMNeT
  (Fieldbus Communication for OMNeT++)
- SignalsAndGateways

Simulation Models
Overview

SignalsAndGateways
Signal sources, Gateways

Vehicle network model

provided       contributed
Simulation Models
CoRE4INET

- Currently supported standards:
  - TTEthernet protocol suite (AS6802)
  - AVB traffic shapers (IEEE 802.1Qav)
  - Ethernet with priorities (IEEE 802.1Q)

- Currently supported features:
  - Models to map IP traffic to real-time traffic classes
  - Incoming traffic selection and constraint checks
  - Models for oscillators, timers and schedulers
  - Application models for traffic patterns
  - Flexible combining of media access strategies

- Checked against analytical models and empirical tests
Simulation Models
FiCo4OMNeT

- Currently supported standards:
  - CAN
  - FlexRay

- Currently supported features:
  - Models for oscillators and timers
  - Application models for traffic patterns

- Checked against results of CANoe simulation environment
Simulation Models
SignalsAndGateways

- Fills the gap between CoRE4INET and FiCo4OMNeT
- Gateway translate between (real-time) Ethernet and fieldbusses
- For flexibility it contains three submodules:
  - Routing
  - Buffering
  - Transformation
- Gateway can host applications
Extending OMNeT++ Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook
**Toolchain**

**Overview**

- Eclipse Updatesite (https://sim.core-rg.de/updates)  
  -> get plugins
- CoRE model installer (OMNeT++ plugin)  
  -> get simulation models
Toolchain
Abstract Network Description Language

- Configuring large heterogeneous networks is complex and lengthy
- Domain Specific Language (DSL) reduces effort
- Eclipse plugin using Xtext technology
- Supported features:
  - Syntax highlighting
  - Code completion
  - Scheduling algorithms (for TDMA technologies\(^2\))
  - Simple inheritance
  - Inline ini configuration

Toolchain
Gantt Chart Timing Analyzer

- Specialized analysis tool as OMNeT++ plugin
- Traces jitter and delay in cyclic communication
- Uses a timing log (.tlog) file written during simulation
- GCTA compresses all occurrences of a cyclic message into one single chart
Toolchain
Gantt Chart Timing Analyzer

Extending OMNeT++
Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook
Toolchain

oppResultManagers

- Set of modules for OMNeT++ simulations: called ResultManagers
- OMNeT++ vector and scalar recording are build-in instances of ResultManagers
- Contributed in oppResultManagers:
  - PCAPng
  - SQLite & postgreSQL
  - Constraint Checks
  - Multiple
- Functionality is not restricted to our simulation models
Introduction

Simulation Models

Toolchain

Workflow

Conclusion & Outlook
Extending OMNeT++ Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences
Abstract Network Description Language File:

```plaintext
types std { // Types can be defined and reused
  ethernetLink ETH { // Definition for Ethernet link
    bandwidth 100Mb/s; // Link has bandwidth of 100MBit/s
  }
} // It is also possible to define types in a separate file

network smallNetwork { // Network name is smallNetwork
  inline ini { // Inline ini for special parameters
    record—eventlog = false
  } // Parameters are inserted into .ini

  devices { // Define all devices in the network
    canLink bus1; // First CAN bus
    canLink bus2; // Second CAN bus
    node node1; // First CAN node
    node node2; // Second CAN node
    gateway gw1; // Gateway for first CAN bus
    gateway gw2; // Gateway for second CAN bus
    switch switch1; // Real—time Ethernet Switch
  }
}
```

Continued on next slide...
Workflow
Network Description

```c
connections{ // Physical connections (Segments = groups)
    segment backbone { // Ethernet Backbone part
        gw1 <-> {new std.ETH} <-> switch1; // Ethernet Link
        gw2 <-> {new std.ETH} <-> switch1; // Ethernet Link
    }
    segment canbus{ // CAN bus part (busses share config)
        node1 <-> bus1; // CAN node connected to first bus
        gw1 <-> bus1; // Gateway connected to first bus
        node2 <-> bus2; // CAN node connected to second bus
        gw2 <-> bus2; // Gateway connected to second bus
    }
}

communication{ // Communication in the network
    message msg1{ // Message definition
        sender node1; // First CAN node is sender
        receivers node2; // Second CAN node is receiver
        payload 6B; // Message payload is 6 Bytes
        period 5ms; // 5ms cyclic transmission
        mapping{ // mapping to traffic class, id, gw strategy
            canbus: can{id 37;}; // Message ID 37 on CAN
            backbone: tt{ctID 102;}; // TT traffic on backbone
            gw1: pool gw1_1{holdUp 10ms;}; // Aggregation time
        }
    }
}
```

Extending OMNeT++
Towards a Platform for the Design of Future In-Vehicle Network Architectures
Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook
Extending OMNeT++ Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook
Workflow
Simulation Configuration / Model

- Uses all three simulation models and INET
- Generated config (.ini/.ned/.xml) > 250 lines
- Resulting network:

```
additonal configuration of ResultManagers in ini file:

- postgreSQL:

  outputscalarmanager — class="cPostgreSQLOutputScalarManager"
  outputvectormanager — class="cPostgreSQLOutputVectorManager"
  postgresqloutputmanager — connection="dbname=testdb _user=
                             testuser _password=testuser _port=15432"

- PCAPng:

  eventlogmanager — class = ”PCAPNGEventManager”
```

Workflow
Simulation Configuration / Model

- Uses all three simulation models and INET
- Generated config (.ini/.ned/.xml) > 250 lines
- Resulting network:

```
additonal configuration of ResultManagers in ini file:

- postgreSQL:

  outputscalarmanager — class="cPostgreSQLOutputScalarManager"
  outputvectormanager — class="cPostgreSQLOutputVectorManager"
  postgresqloutputmanager — connection="dbname=testdb _user=
                             testuser _password=testuser _port=15432"

- PCAPng:

  eventlogmanager — class = ”PCAPNGEventManager”
```
Workflow
Simulation Results

- OMNeT++ (scalar, vector and eventlog)
- WireShark (PCAPng)
- GCTA (timinglog)
- Database (mysql, postgresQL)

Use-case postgresQL database:
Extending OMNeT++
Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Agenda

1. Introduction
2. Simulation Models
3. Toolchain
4. Workflow
5. Conclusion & Outlook
Conclusion

- In-car communication technologies are changing
- Simulation on system-level supports the process
- We contribute a simulation environment with:
  - Simulation models
  - Development tools
  - Analysis tools
- Specialized tools can support the workflow
- OMNeT++ is a solid foundation for the development of such plugin tools
Outlook

- Adding new technologies to our simulation suite:
  - Ethernet with frame preemption (discussed in IEEE 802.1Qbu)
  - CAN with flexible data rate (CAN FD)
- Refinement of result analysis tools
Thank you for your attention!

- Website of CoRE research group: https://core-rg.de/
- Website of simulation models: https://sim.core-rg.de/
Extending OMNeT++
Towards a Platform for the Design of Future In-Vehicle Network Architectures

Philipp Meyer

Introduction
Simulation Models
Toolchain
Workflow
Conclusion & Outlook

References I
