Modelling Network Connections in FMI with an Explicit Network Model

Luis Diogo Couto    Ken Pierce

UTRC-I

University of Newcastle

15th Overture Workshop
15-09-2017

Created at UTRC-I

This document does not contain any export controlled technical data.
Outline

Introduction

Comms Modelling Challenges

Ether Pattern

Case Study

Conclusions
Outline

Introduction

Comms Modelling Challenges

Ether Pattern

Case Study

Conclusions
Introduction

- CPS design is challenging
- Multiple stakeholders, disciplines, notations...
- FMI can help couple tools
- Some challenges remain (ex: comms)
Introduction

- CPS design is challenging
- **Multiple** stakeholders, disciplines, notations...
- FMI can help couple tools
- Some challenges remain (ex: comms)

Created at UTRC-I – This page does not contain any export controlled technical data.
Outline

Introduction

Comms Modelling Challenges

Ether Pattern

Case Study

Conclusions
Communications Modelling

Fundamental Challenges

- inter-FMI comms
- Models exported as whole
- Native comms reasoning hidden
- Have to work at FMI level

- No events or message passing
- All signal exchanges continuous
- Can work around but it’s hacky
- Time issues
Communications Modelling
Practical Challenges

Embedding Explicit Comms in FMUs

- Model pollution
- Difficult to see complete picture
- 1:1 FMU data exchange... too many connections

Communication Messages Representation

- Only primitive types (bools, strings reals)
- No structured types 😞
  - Multi-ports is too many ports
  - String encoding adds overhead
Outline

Introduction

Comms Modelling Challenges

Ether Pattern

Case Study

Conclusions
Ether Pattern

Network Topology in FMI

- Direct connection
- Unwieldy
- Inflexible

- Ether pattern
- Communications medium
- Flexible
Ether Pattern
Network Topology in FMI

- Producer/consumer example: one output, one input
- Ether sits between and passes messages
- Requires additional ports for new FMUs
Ether Pattern
Network Topology in FMI

Sender  Generates messages, encodes to strings using VDMUtil, sets its output.

Receiver Receives messages, decodes using VDMUtil.

Ether Each input / output port assigned unique identifier as a map Id to StringPort. Mapping from input to output: set of (Id * Id). Gathers messages from each input and passes them to outputs.

- Not currently considered: sender identification, confirmation of delivery, maximum transmission unit, message timing
Outline

Introduction

Comms Modelling Challenges

Ether Pattern

Case Study

Conclusions
Case Study
Building HVAC

- 4 Fan Coil Units (FCUs) in rooms and zones
- Networked FCU Controllers
- Supervision

- 4x Simulink FCU Controller FMU
- 1x VDM Supervisor FMU
- Multi-port approach
`system` System

**instance variables**

`public static` super : [Supervisor] := nil;

`public static` sr1 : [FCU] := nil;

-- ...

**operations**

`public` System : () ==> System

System () == (  
srl := `new` FCU(1);

srl.primeFmi(hwi.srl_spIn,hwi.srl_spOut,
            hwi.srl_mode);

cpu1.deploy(srl,"FCU_SR1");

-- ...

super := `new` Supervisor({srl,sr2,z1,z2});

cpu5.deploy(super,"Supervisor");)

end System
Outline

Introduction

Comms Modelling Challenges

Ether Pattern

Case Study

Conclusions
As it currently stands, FMI is limited for CPS (DE comms)  
Ether helped structure the multi-model and study networked controllers for the industrial case study  
The best workaround is still a workaround  
FMI extension can help but broad support is key  
Adding real value will drive adoption