26th annual INCOSE international symposium
Edinburgh, UK
July 18 - 21, 2016

Collaborative MBSE for Cyber-Physical Systems with a Building Automation Case Study

John Fitzgerald, Carl Gamble, Richard Payne
Newcastle University, UK
Firstname.Lastname@newcastle.ac.uk

Peter Gorm Larsen
Aarhus University, Denmark
pgl@eng.au.dk

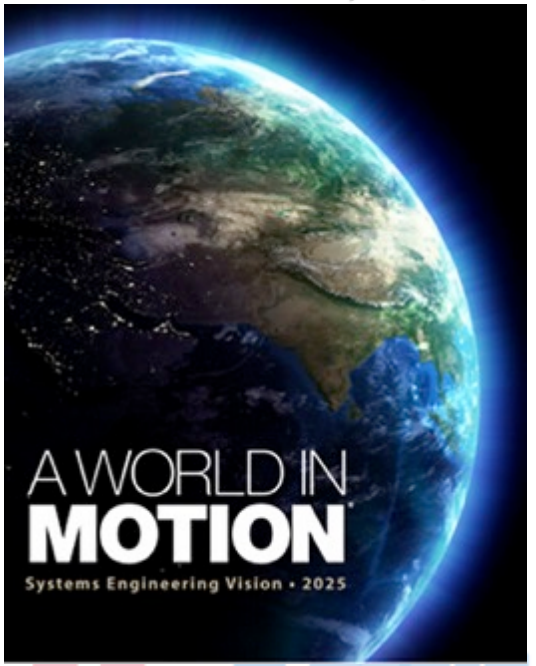
Stylianos Basagiannis, Alie El-Din Mady
United Technologies, Cork, Ireland
BasagiS@utrc.utc.com
MadyAA@utrc.utc.com

Overview

1. **Engineering of Cyber-Physical Systems**
2. Co-models and Co-simulation
3. A Building Automation Study
4. Towards Multi-models
5. Concluding Remarks

A Striking Vision

- Systems of Systems enabled by networked, autonomous computing elements
- Collaborative and model-based engineering methods as a means of managing risk
- Supporting cross-disciplinary analyses, design space explorations and optimisations



Engineering Cyber-Physical Systems

Newcastle University

Advanced Model-Based Engineering & Reasoning (AMBER)

- Model-based Systems Engineering for demanding systems
  - Systems of Systems (www.thecompassclub.org)
  - Cyber-Physical Systems (www.cpse-labs.eu)
- Models as a basis for collaborative multi-disciplinary development
- Exploiting rigorous mathematical semantics
- Benefits:
  - Traceability to requirements and assurance arguments
  - Design Space Exploration
  - Models as test oracles
  - Design-time V&V of performance, safety, security properties …
- Demonstrated reductions in development effort and defect rates

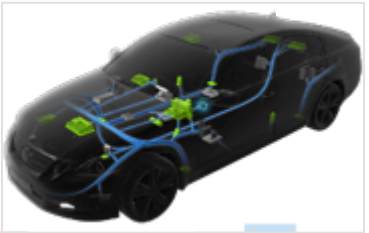
SONY FeliCa

NYSE

Engineering Cyber-Physical Systems (CPSs)

Networked systems integrating embedded computational and physical processes, and human users.

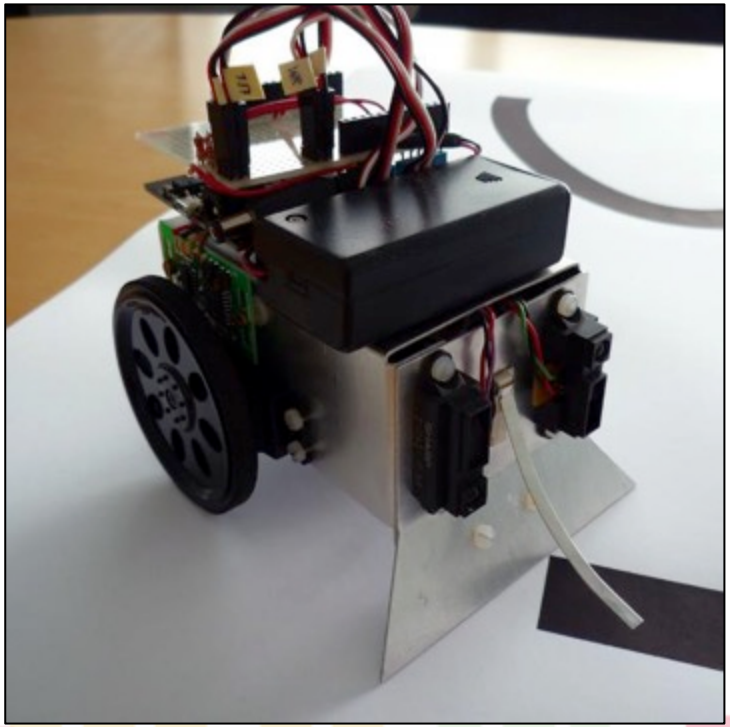
- Examples: agile manufacturing, responsive infrastructure, buildings, transport, cooperative robotics
- Step up from technical processes to organisation level
- Inherently multidisciplinary
- Reliance is placed on joint behaviour of cyber and physical elements
    - Confidence in their interaction is a research focus



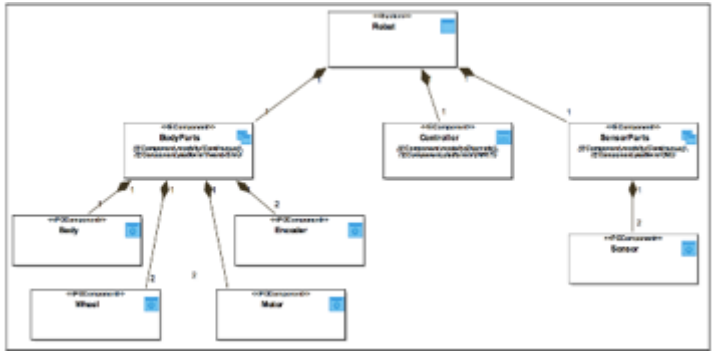
Overview

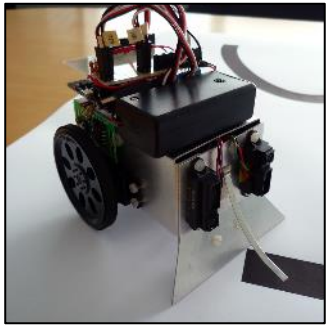
1. Engineering of Cyber-Physical Systems
2. **Co-models and Co-simulation**
3. A Building Automation Study
4. Towards Multi-models
5. Concluding Remarks

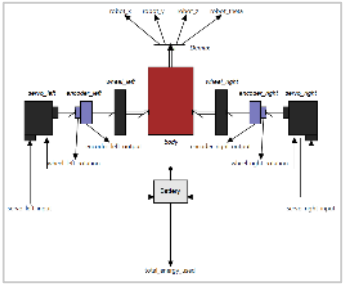
Co-modelling and Co-simulation

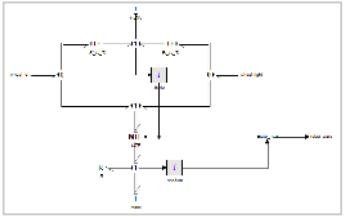


Co-modelling and Co-simulation





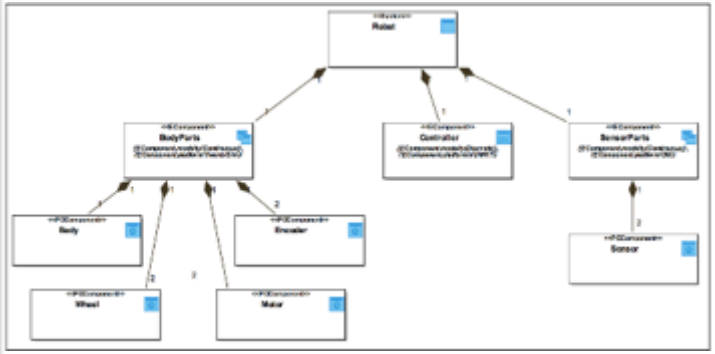




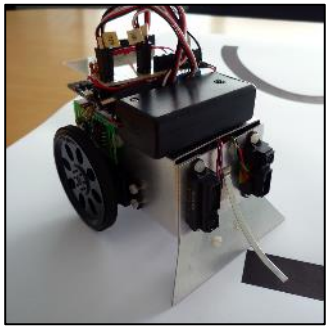
```
class AbstractModalController
instance variables
-- servos (as abstract classes)
private servoLeft:  IActuatorRealPercent;
-- current mode and map of all possible modes
protected modes: map Mode to IMode;
-- control turn aggression
private fastSpeed: IActuatorRealPercent`Percent;
operations
public Step: () ==> ()
Step() == (if init then (
            modes(mode).Enter();
            init := false);
            let m = CheckModeChange() in
            if m <> nil then ChangeMode(m);
            let m = modes(mode).Step() in
            if m <> nil then ChangeMode(m);     );
```

```
externals
  real global export pos_x;
  real global export pos_y;

parameters
  real global initial_position [3];

equations
  position = int (velocity) +
                initial_position[1:2];
  pos_x = position[1];
  pos_y = position[2];
```

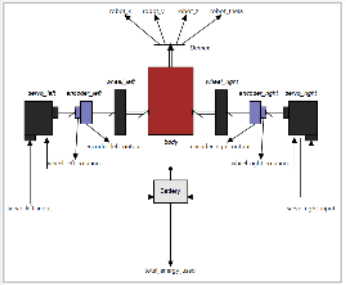
Co-modelling and Co-simulation

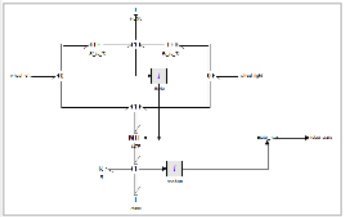


```
class AbstractModalController
instance variables
-- servos (as abstract classes)
private servoLeft:  IActuatorRealPercent;
-- current mode and map of all possible modes
protected modes: map Mode to IMode;
-- control turn aggression
private fastSpeed: IActuatorRealPercent`Percent;
operations
public Step: () ==> ()
Step() == (if init then (
            modes(mode).Enter();
            init := false);
            let m = CheckModeChange() in
            if m <> nil then ChangeMode(m);
            let m = modes(mode).Step() in
            if m <> nil then ChangeMode(m);     );
```



Mind the (Semantic) Gap!





```
externals
  real global export pos_x;
  real global export pos_y;

parameters
  real global initial_position [3];

equations
  position = int (velocity) +
                initial_position[1:2];
  pos_x = position[1];
  pos_y = position[2];
```

Co-modelling and Co-simulation

- **Discrete-event (DE)**, e.g. VDM-RT
- In simulation, only represent points in time at which the state changes
- Good abstractions for software
- Less suited for physical system modelling

Software:
- Discrete
- Complex logic

DE Model

Physics:
- Continuous
- Numerical

CT Model

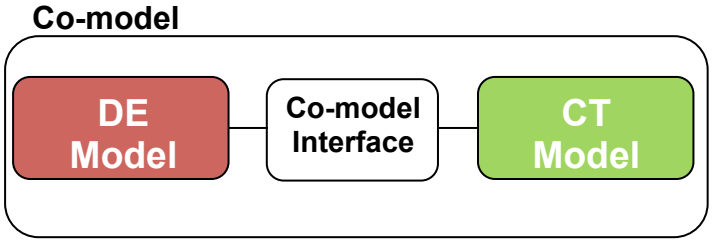
- **Continuous-time (CT)**, e.g. differential equations
- In simulation, the state changes continuously through time
- Good abstractions for physical system disciplines
- Poor software modelling support

Co-modelling and Co-simulation

- **Discrete-event (DE)**, e.g. VDM-RT
- In simulation, only represent points in time at which the state changes
- Good abstractions for software
- Less suited for physical system modelling

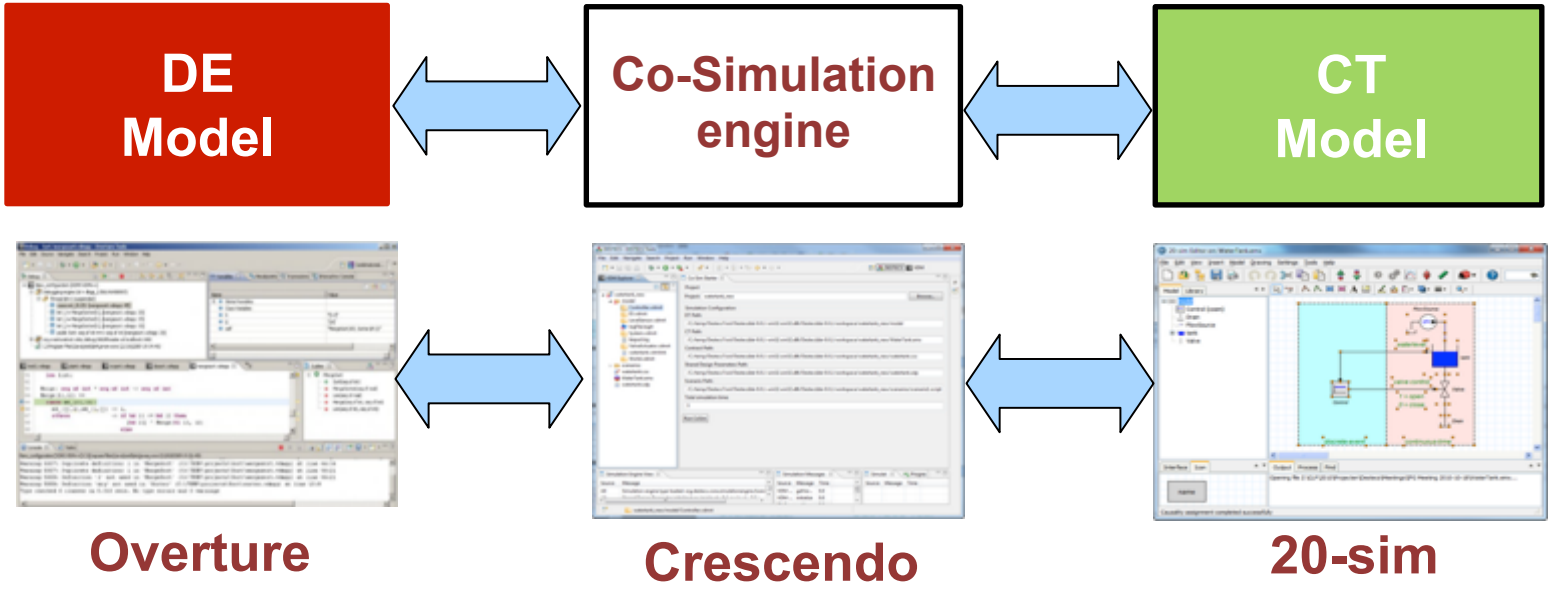
Software:
- Discrete
- Complex logic

Physics:
- Continuous
- Numerical

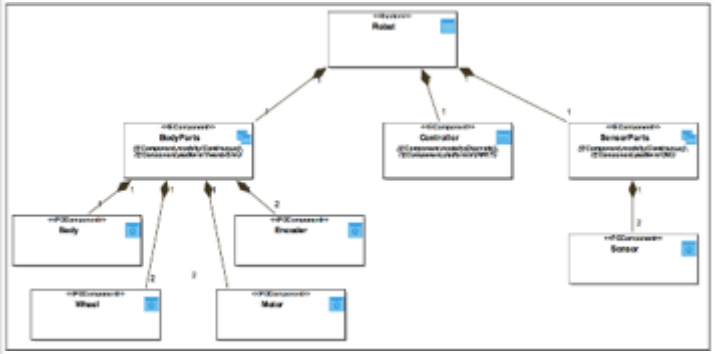


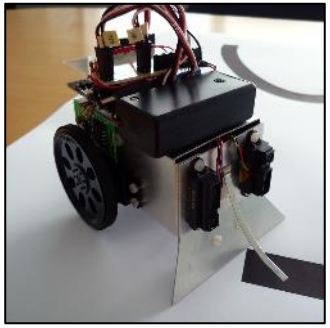
- **Continuous-time (CT)**, e.g. differential equations
- In simulation, the state changes continuously through time
- Good abstractions for physical system disciplines
- Poor software modelling support

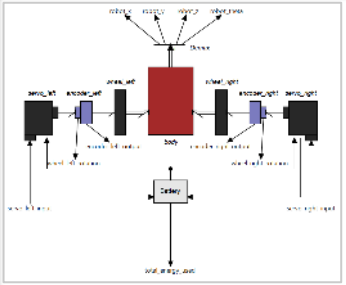
Co-modelling and Co-simulation

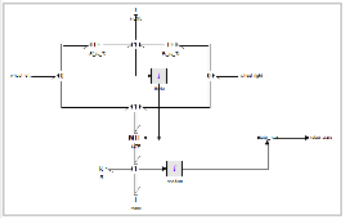


Co-modelling and Co-simulation









```
class AbstractModalController
instance variables
-- servos (as abstract classes)
private servoLeft:  IActuatorRealPercent;
-- current mode and map of all possible modes
protected modes: map Mode to IMode;
-- control turn aggression
private fastSpeed: IActuatorRealPercent`Percent;
operations
public Step: () ==> ()
Step() == (if init then (
            modes(mode).Enter();
            init := false);
            let m = CheckModeChange() in
            if m <> nil then ChangeMode(m);
            let m = modes(mode).Step() in
            if m <> nil then ChangeMode(m);     );
```

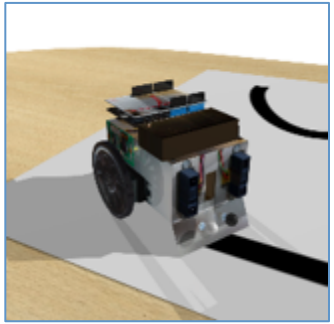
```
externals
  real global export pos_x;
  real global export pos_y;

parameters
  real global initial_position [3];

equations
  position = int (velocity) +
                initial_position[1:2];
  pos_x = position[1];
  pos_y = position[2];
```

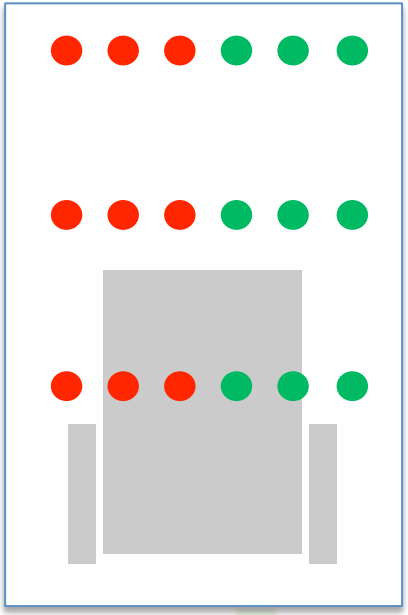
Co-modelling and Co-simulation

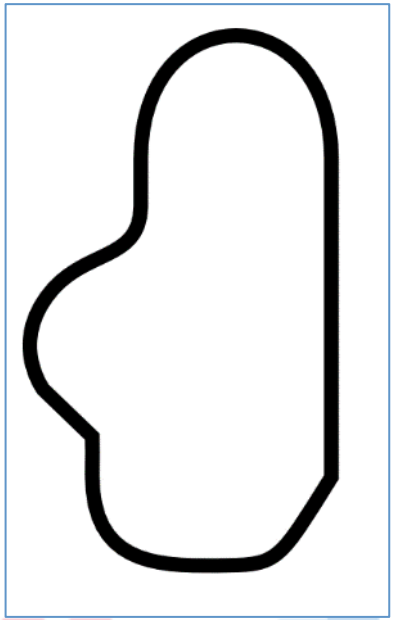
Design Space Exploration

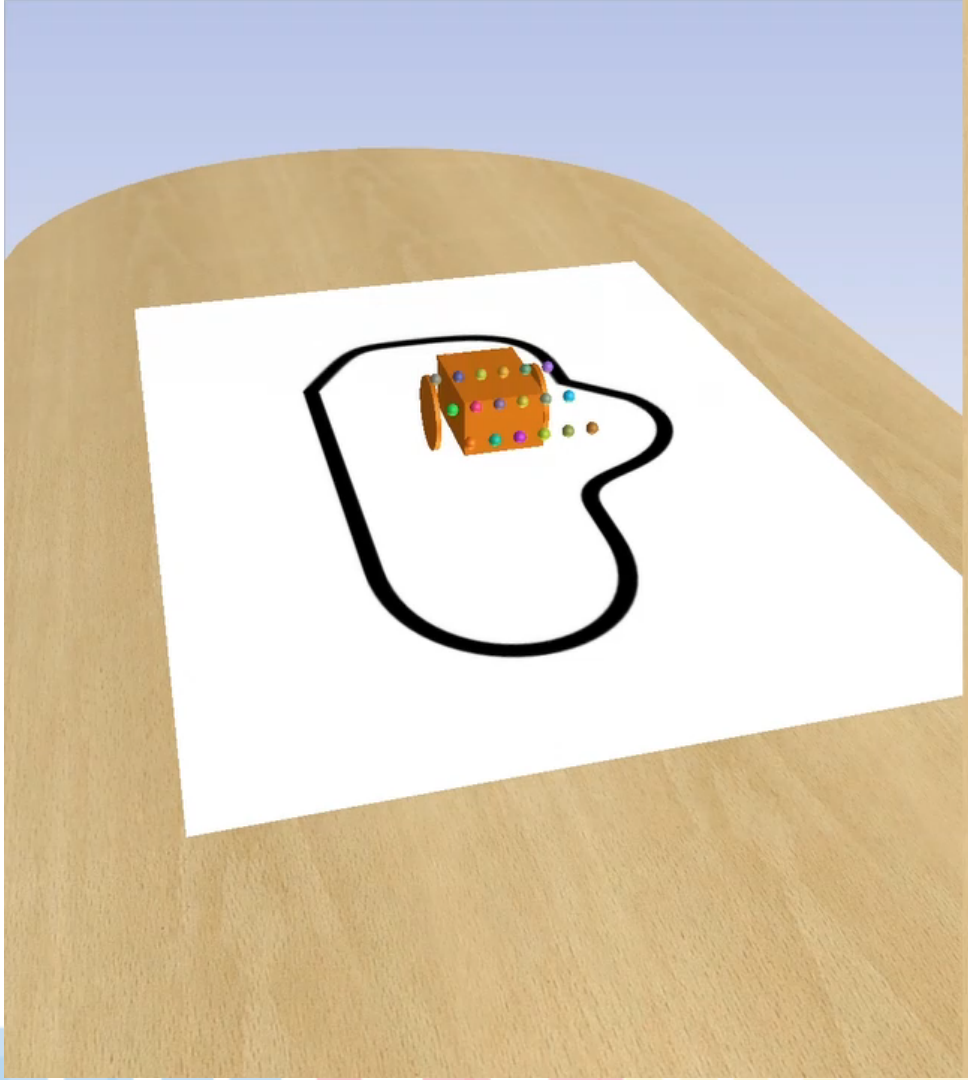
Sweeping over design parameters

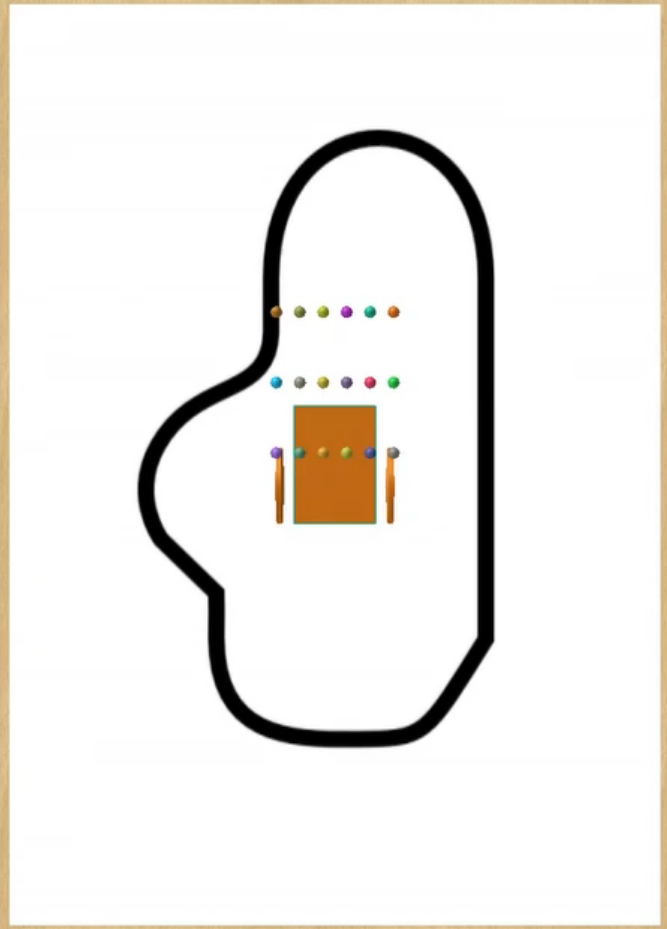
Could be in the DE or CT models



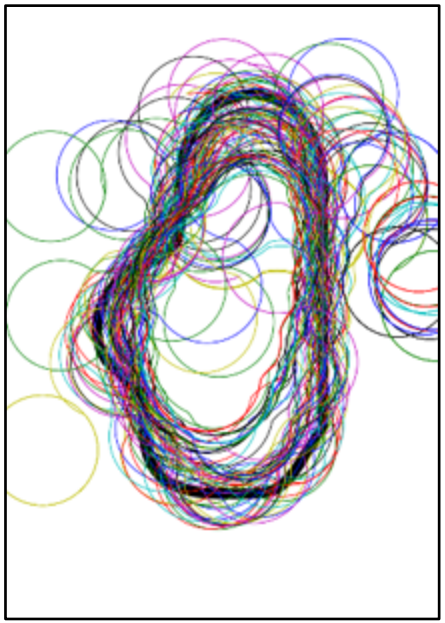


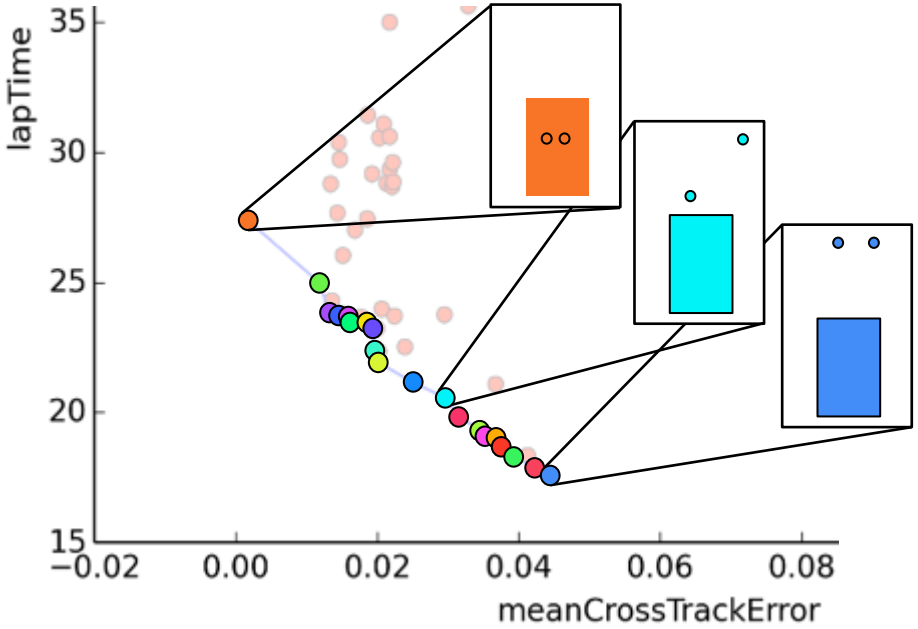


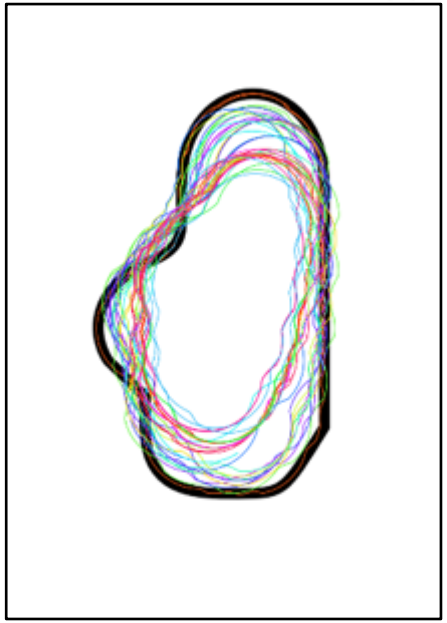




Co-modelling and Co-simulation







Overview

1. Engineering of Cyber-Physical Systems
2. Co-models and Co-simulation
3. **A Building Automation Study**
4. Towards Multi-models
5. Concluding Remarks

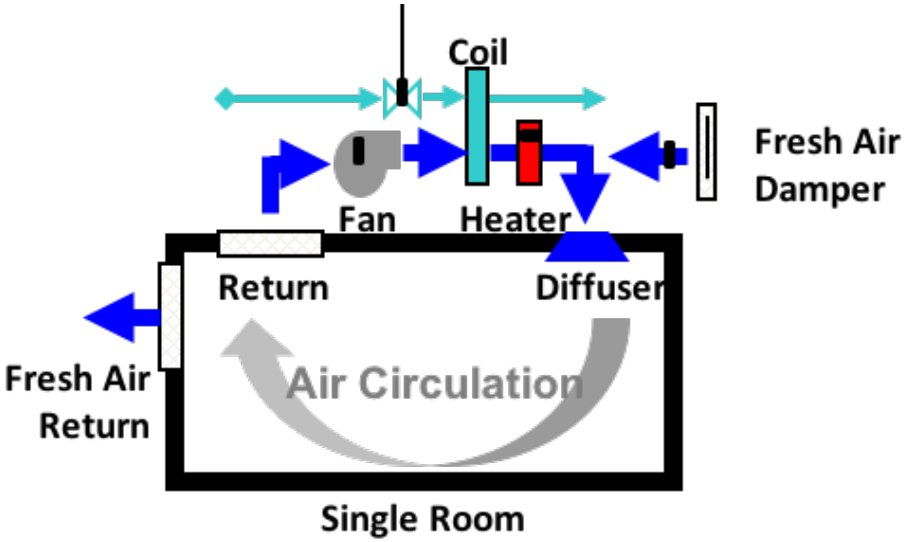
A Building Automation Study with UTRC

- Buildings: 40% of energy consumption, 36% of carbon emissions
- Potential benefits via, e.g. IoT-based user profiling
- Complex interconnected control systems interacting with physical environment
- Can co-modelling help?

- Case Study
- UTRC identified targets:
  - Co-modelling
  - Co-simulation performance
  - Co-simulation accuracy and precision

A Building Automation Study

- Fan Coil Unit (FCU)
- Water heated/cooled, passes through Coil
- Room air blow through Fan, returned to room
- Software Controller



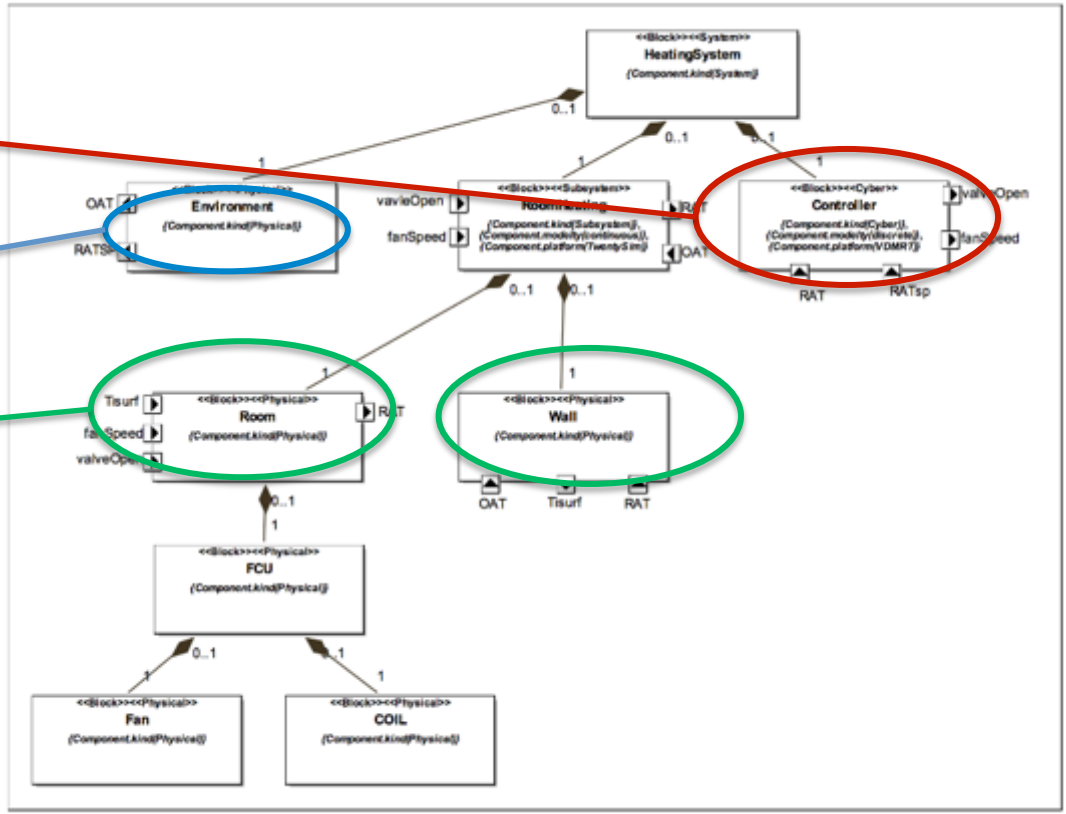
A Building Automation Study: Co-model Architecture

Architectural Structure Diagram

Cyber

Envt. handled as time-indexed tables

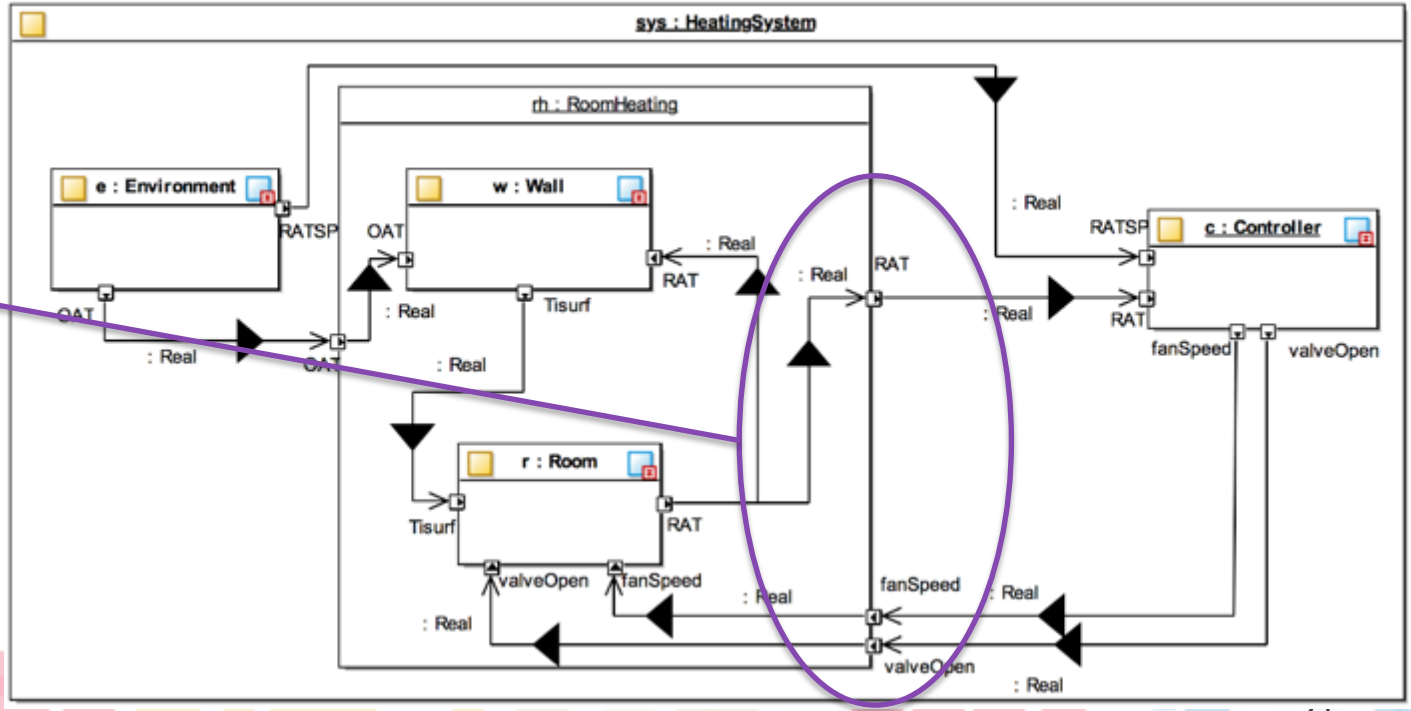
Physical



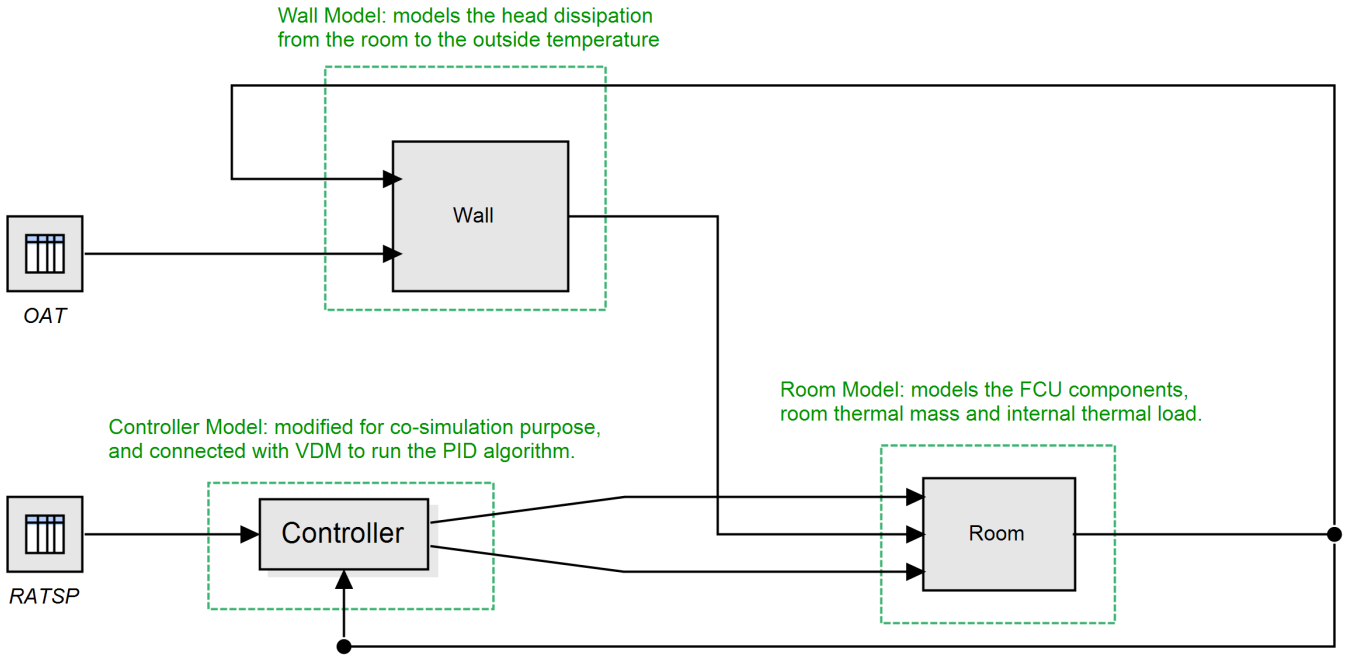
A Building Automation Study: Co-model Architecture

Connection Diagram

Shared Parameters



A Building Automation Study: Co-model: CT



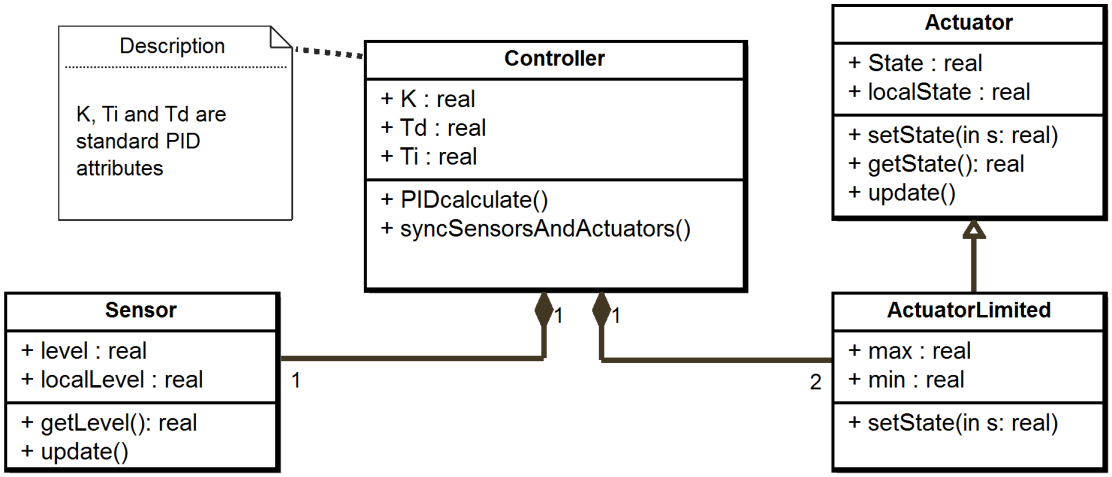
A Building Automation Study: Co-model: CT

- ODEs within the Wall block in the CT model

```
parameters
real rhoWall = 1312.0; -- wall density
real cWall = 1360.71; -- specific heat capacity
real lambdaWall = 0.1192; -- wall thermal conductivity
real lWall = 0.001; -- wall thickness
…
variables
real Tosurf; -- external surface temperature
real R; -- wall resistance
real C; -- wall thermal capacity

equations
R = lWall / (lambdaWall * aWall);
C = 0.5 * rhoWall * cWall * lWall * aWall;
Tisurf = int ((hi * aWall * (RAT - Tisurf) + (Tosurf - Tisurf) / R) / C, TisurfInit);
Tosurf = int((ho * aWall * (OAT - Tosurf) + (Tisurf - Tosurf) / R) / C, TosurfInit);
```

A Building Automation Study: Co-model: DE

- Object-oriented
- e.g. general Actuator model overridden to limit outputs from the control algorithm to suit physical valve and fan

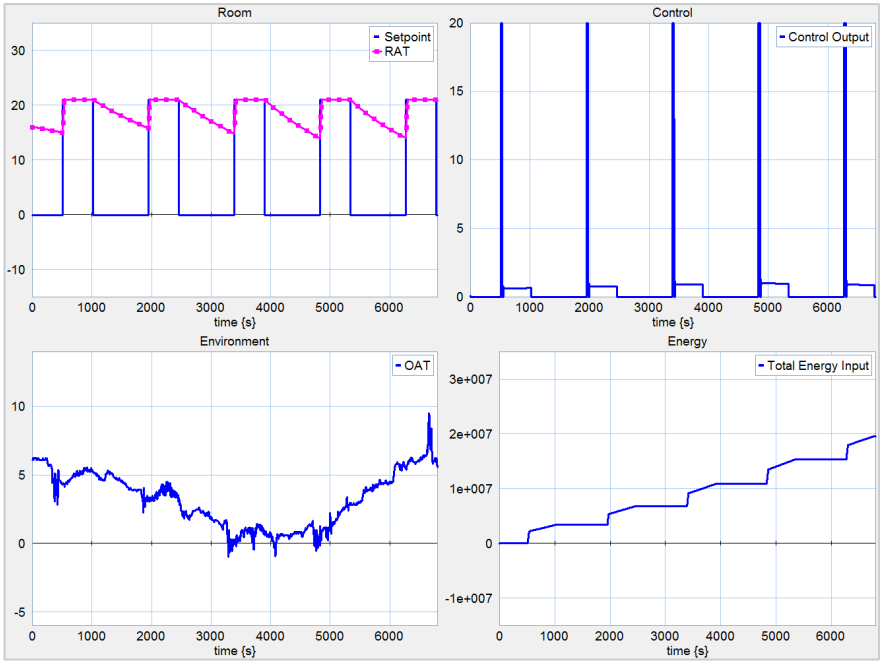


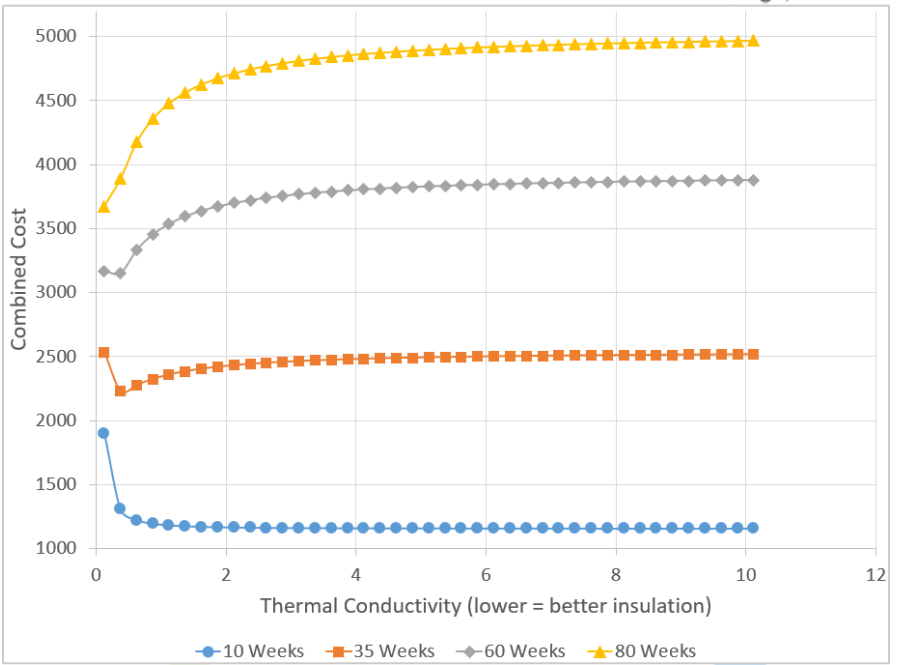
A Building Automation Study: Co-model: DE

- Object-oriented
- e.g. general Actuator model overridden to limit outputs from the control algorithm to suit physical valve and fan
- Could be arbitrarily sophisticated (this PID controller isn't!)

```
private PIDcalculate:()==>()
PIDcalculate()==
(
      syncSensorsAndActuators();

      MV:=RAT.getLevel();
      err:=RATSP-MV;
      factor:=Td/(sampletime+(Td/N));
      uP:=K*(b*RATSP-RAT.getLevel());
      uI:=previousuI+sampletime*(K*err/Ti);
      previousuI:=uI;
      uDin:=c*RATSP-MV;
      previousuDin:=uDin;
      uD:=factor*(uD/N+K*(uDin-previousuDin));
      control:=uP+uI+uD;
      valveOpen.setState(control);
      fanSpeed.setState(control);
);
thread - period 80 ms
periodic(80E6/*ms*/,0,0,0)(PIDcalculate);
```

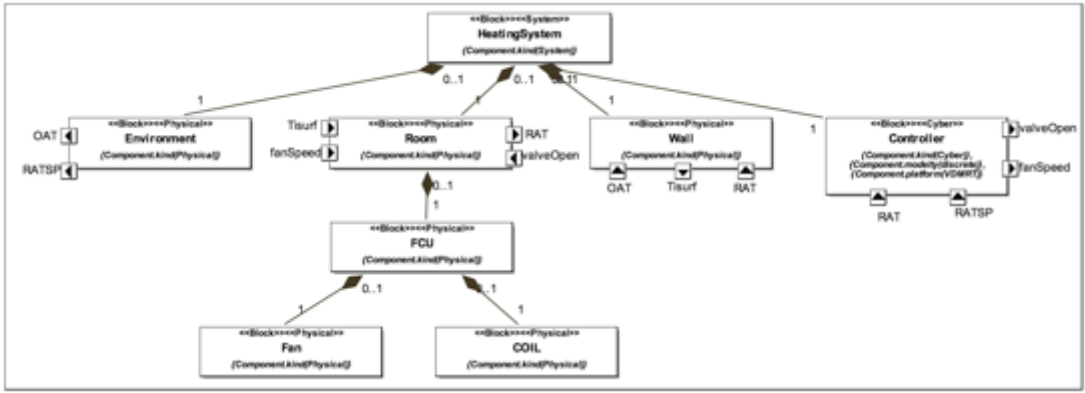
A Building Automation Study: Co-model: DSE





A Building Automation Study: Limitations

- Binary co-model is limiting
  - N-ary multimodel more realistic
- Performance acceptable (one week in 7.5 minutes on a desktop PC)
- Ability to set precision DE side

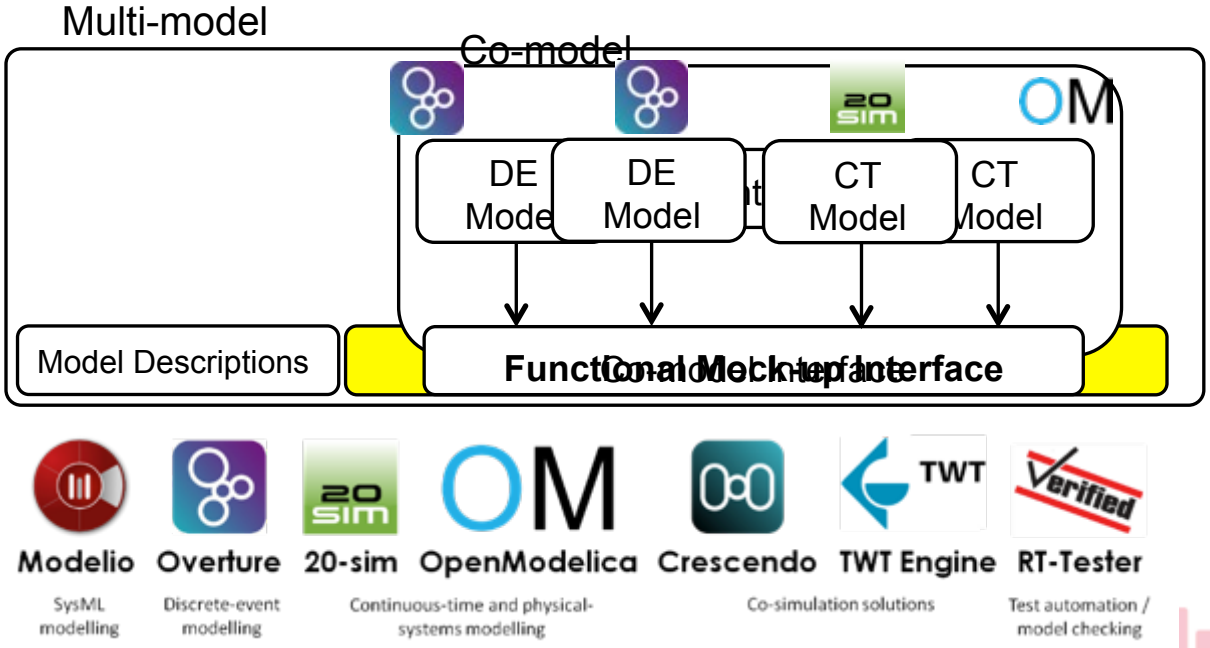


Overview

1. Engineering of Cyber-Physical Systems
2. Co-models and Co-simulation
3. A Building Automation Study
4. **Towards Multi-models**
5. Concluding Remarks

Towards Multi-models

Integrated Toolchains INTO-CPS (www.into-cps.au.dk)

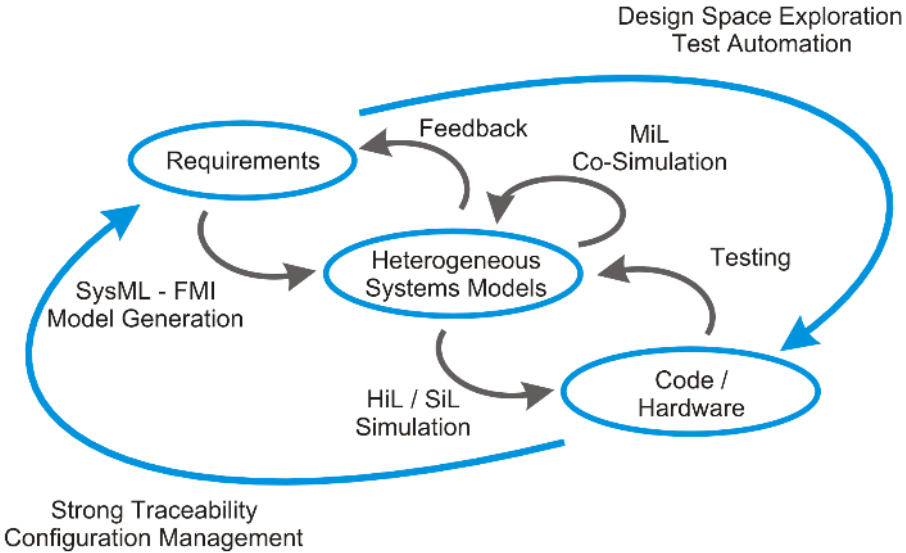


Towards Multi-models

Integrated Toolchains INTO-CPS (www.into-cps.au.dk)

- Modelling Guidelines, Frameworks, Profiles, Patterns

- No “factotum” tools!
- Traceability, provenance and model management
- Extensible range of plug-ins



Overview

1. Engineering of Cyber-Physical Systems
2. Co-models and Co-simulation
3. A Building Automation Study
4. Towards Multi-models
5. **Concluding Remarks**

Concluding Remarks

- Collaborative Systems need Collaborative Engineering!
- Model-based methods for multidisciplinary CPSE
- Challenges:
  - Integration of heterogenous models in workflows and toolchains
  - Semantic links
  - Assurance through traceability
- User-driven case study
- Co-modelling and co-simulation is feasible fro DE/CT binary co-models
- Integration of co-modelling with SE processes
- Increasing number and diversity of constituent models
- Need for practical guidelines


26th annual INCOSE
international symposium
Edinburgh, UK
July 18 - 21, 2016

Thank You!