Virtual Engineering in CPS
Virtual engineering everywhere

Helping over 30 different teams and skills in the company work together

Linking over 40 different EE design representations throughout the entire development process

Ensuring that the EE design flow is integrated at the same level of quality and performance as the 3D CAD system

Model based design and executable specification in the OEM/supplier chain
Virtual engineering everywhere

CAD models

Helping over 30 different teams and skills in the company work together

Linking over 40 different EE design representations throughout the entire development process

Ensuring that the EE design flow is integrated at the same level of quality and performance as the 3D CAD system

Model based design and executable specification in the OEM/supplier chain

Dassault Systèmes Delmia and Catia
http://www.3ds.com/products
Virtual engineering everywhere
Multi-Physics models

Helping over 30 different teams and skills in the company work together

Linking over 40 different EE design representations throughout the entire development process

Ensuring that the EE design flow is integrated at the same level of quality and performance as the 3D CAD system

Model based design and executable specification in the OEM/supplier chain

http://www.mathworks.fr/products/simscape/index.html
Virtual engineering everywhere
Embedded Software

Helping over 30 different teams and skills in the company work together

Linking over 40 different EE design representations throughout the entire development process

Ensuring that the EE design flow is integrated at the same level of quality and performance as the 3D CAD system

Model based design and executable specification in the OEM/supplier chain

http://www.esterel-technologies.com/products/scade-suite/
http://www.esterel-technologies.com/products/scade-display/
Helping over 30 different teams and skills in the company work together

Linking over 40 different EE design representations throughout the entire development process

Ensuring that the EE design flow is integrated at the same level of quality and performance as the 3D CAD system

Model based design and executable specification in the OEM/supplier chain

RT-Builder: modeling advanced RTOS and Bus libraries. ARINC653, CAN, OSEK, FlexRay standards. Can easily be customized or extended to provide a powerful support of customer's real-time design choices. Users can model: interactions between control and data functions, concurrency, multi-tasking, pre-emptive actions, shared resources, event routing and filtering, FIFO buffers, point-to-point data propagation time, etc...
Virtual engineering for CPS: the enablers

Seamless modeling

PLM/CAD
Multi-physics
CPS functions
Architecture
  - ECU
  - communications
Resources
  - time,
  - energy,
  - memory,
  - weight
Virtual engineering for CPS: the enablers

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

models from PLM

do not build models by hand!

Quality, certification

Executable models
- reproducible executions
- avoid unwanted non-determinism

Analyzable models
Virtual engineering for CPS: the enablers

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions

Architecture
- ECU
- Communications

Resources
- Time
- Energy
- Memory
- Weight

Quality, certification

Heterogeneity

Executable models
- Reproducible executions
- Avoid unwanted non-determinism

Analyzable models
Must reuse basic domains
- Discrete time
- Continuous, real, time
- Resource models
Virtual engineering for CPS: the enablers

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

Quality, certification

Heterogeneity

Executable models
  - reproducible executions
  - avoid unwanted non-determinism

Analyzable models

Must reuse basic domains
  - Discrete time
  - Continuous, real, time
  - Resource models
Virtual engineering for CPS: the enablers

Seamless modeling

PLM/CAD

Multi-physics

CPS functions

Architecture

- ECU
- communications

Resources

- time,
- energy,
- memory,
- weight

Quality, certification

Heterogeneity

Executable models

- reproducible executions
- avoid unwanted non-determinism

Analyzable models

Must reuse basic domains

- Discrete time
- Continuous, real, time
- Resource models
Virtual engineering for CPS: the enablers

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

Quality, certification

Heterogeneity

Executable models
- reproducible executions
- avoid unwanted non-determinism

Analyzable models
Must reuse basic domains
- Discrete time
- Continuous, real, time
- Resource models
Consequences for Hybrid Systems modeling in CPS
Is the current situation good?

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

Quality, certification

Heterogeneity

- Executable models
  - reproducible executions
  - avoid unwanted non-determinism
- Analyzable models
  - Discrete time
  - Continuous, real, time
  - Resource models
Is the current situation good?

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

Is it the same discrete time everywhere?

Simulink/Stateflow
- Simulation → development
- Two distinct engines
- Semantics & consistency: non-obvious

Two major difficulties
- Difficult to decide what is continuous/discrete (hard in Stateflow)
- Simulation is global
Is the current situation good?

- It is allowed to construct diagrams in which "continuous" and "discrete" time blocks mix
- One (often) gets complaints and hints for parameter tuning

'Sample4/Unit Delay: w' is discrete, yet is inheriting a continuous sample time; consider replacing unit delay with a memory block. When a unit delay block inherits continuous sample time, its behavior is the same as the memory block. Unit delay block's time delay will not be fixed and could change with each time step. This might be unexpected behavior. Normally, a unit delay block uses discrete sample time. You can disable this diagnostic by setting the 'Discrete used as continuous' diagnostic to 'none' in the Sample Time group on the Diagnostics panel of the Configuration Parameters dialog box.

- Discrete time

Simulink/Stateflow
- Simulation → development
- Two distinct engines
- Semantics & consistency: non-obvious

Two major difficulties
- Difficult to decide what is continuous/discrete (hard in Stateflow)
- Simulation is global
Is the current situation good?

- It is allowed to construct diagrams in which “continuous” and “discrete” time blocks mix
- One (often) gets complaints and hints for parameter tuning

Performing this yields: Warning: The configuration of the Unit Delay ‘Example4/Unit Delay: w’ is incorrect for handling a rate transition. Consider using the Rate Transition block to handle the data transfer between rates. Alternatively, you can control the diagnostic action for unspecified rate transitions on the Sample Time Diagnostics pane of the Configuration Parameters dialog box.

Simulink/Stateflow

- Simulation → development
- Two distinct engines
- Semantics & consistency: non-obvious

Two major difficulties

- Difficult to decide what is continuous/discrete (hard in Stateflow)
- Simulation is global
Is the current situation good?

- It is allowed to construct diagrams in which “continuous” and “discrete” time blocks mix
- One (often) gets complaints and hints for parameter tuning
- Running this example results in unexpected couplings between the different outputs: changing the frequency of the sine change the other outputs, although there is no interaction other than sharing the simulator!

Simulink/Stateflow
- Simulation → development
- Two distinct engines
- Semantics & consistency: non-obvious

Two major difficulties
- Difficult to decide what is continuous/discrete (hard in Stateflow)
- Simulation is global

Disclaimer: this not a complaint against Simulink/Stateflow
The subject is difficult!
Is the current situation good?

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

Is it the same discrete time everywhere?

Modelica group is currently working hard at separating

- Discrete reactions
- Continuous real-time
- Continuous dynamics

in the context of integrated virtual modeling
Is the current situation good?

Seamless modeling

PLM/CAD

Multi-physics

CPS functions

Architecture

- ECU
- communications

Resources

- time,
- energy,
- memory,
- weight

Is it the same discrete time everywhere?

The Ptolemy Project has developed directors supporting process networks (PN), discrete-events (DE), dataflow (SDF), synchronous / reactive (SR), rendezvous-based models, 3-D visualization, and continuous-time models.

Ptolemy II delegates the blending of the different models to directors.

When continuous-time models are involved, then the numerical solvers act as directors.

Nice simulation examples of difficult hybrid systems.
The physics can be difficult
Some difficulties from the physics

Non-determinism
  • which side of the corner is hit first?

Cascades of hits
  • in billiards
  • in circuit breakers

Non-determinism in cascades of hits
Some difficulties from the physics

Chattering
  • infinitely fast stop-and-go

The sticky masses (Ptolemy II)
three modes:
  • free
  • hitting
  • sticking
Some difficulties from the physics

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses
Some difficulties from the physics

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Control interferes with
- Cascades of events
- Continuous dynamics (sampling)
Some difficulties from the physics

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Unwanted couplings due to solvers
- Fast/Slow dynamics
- Zero-crossings, hot/cold restart
The problem with continuous/discrete
Which model makes sense?

Example

\[
\text{let node } \text{sum}(x) = \text{Cpt where} \\
\text{Cpt}_n = \text{Cpt}_{n-1} + x
\]

Evaluate:

\[
\text{der } \text{time} = 1.0 \text{ init } 0.0 \\
y = \text{sum}( \text{time} )
\]
Which model makes sense?

Example

let node sum(x) = Cpt where
Cpt_n = Cpt_{n-1} + x

Evaluate:
\[
\text{der } \text{time} = 1.0 \text{ init } 0.0 \\
y = \text{sum}( \text{time} )
\]

Interpretations
1. \( N \subseteq \mathbb{R} \)
Which model makes sense?

Example

let node sum(x) = Cpt where
Cpt_n = Cpt_{n-1} + x

Evaluate:
der time = 1.0 init 0.0
y = sum( time )

Interpretations
1. $\mathbb{N} \subseteq \mathbb{R}$
2. Depends on the solver
Which model makes sense?

Example

```
let node sum(x) = Cpt where
Cpt_n = Cpt_{n-1} + x
```

Evaluate:
```
der time = 1.0 init 0.0
y = sum( time )
```

Interpretations
1. $\mathbb{N} \subseteq \mathbb{R}$
2. Depends on the solver
3. Infinitesimal discrete steps
Which model makes sense?

Example

\begin{verbatim}
let node sum(x) = Cpt where
Cpt_n = Cpt_{n-1} + x
\end{verbatim}

Evaluate:
\begin{verbatim}
der time = 1.0 init 0.0
y = sum(time)
\end{verbatim}

Interpretations
1. \( \mathbb{N} \subseteq \mathbb{R} \)
2. Depends on the solver
3. Infinitesimal discrete steps
4. Reject the example
Which model makes sense?

Example

let node sum(x) = Cpt where
Cpt_n = Cpt_{n-1} + x

Evaluate:
der time = 1.0 init 0.0
y = sum( time ) every up(z)

Interpretations
1. N ⊆ R
2. Depends on the solver
3. Infinitesimal discrete steps
4. Reject the example
5. Explicit the missing synchro
Slicing Continuous/Discrete

- **D**: handled by the Discrete Engine (e.g., your favorite synchronous language)
- **C**: delegated to the solver
- **C→D**: zero-crossing detection (includes periodic sampling clocks)
- **D→C**: upon detecting end of cascade of events
- Slicing allows importing off-the-shelf
  - discrete-time modelers
  - continuous dynamics solvers
Slicing Continuous/Discrete

- **D**: handled by the Discrete Engine (e.g., your favorite synchronous language)
- **C**: delegated to the solver
- **C→D**: zero-crossing detection (includes periodic sampling clocks)
- **D→C**: upon detecting end of cascade of events
- Slicing allows importing off-the-shelf
  - discrete-time modelers
  - continuous dynamics solvers
- Discrete time models of your choice
Slicing Continuous/Discrete

- **D**: handled by the Discrete Engine (e.g., your favorite synchronous language)
- **C**: delegated to the solver
- **C→D**: zero-crossing detection (includes periodic sampling clocks)
- **D→C**: upon detecting end of cascade of events

Slicing allows importing off-the-shelf
- discrete-time modelers
- continuous dynamics solvers

- Discrete time models of your choice
- Structured, multi-solver, simulation of continuous dynamics
  - to reduce unwanted coupling
Slicing is possible only if the language is well designed

<table>
<thead>
<tr>
<th>statement</th>
<th>Assigned to D</th>
<th>Assigned to C</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y = f(x) )</td>
<td>on ( \zeta_{syst} : y = f(x) )</td>
<td>outside ( \zeta_{syst} : y = f(x) )</td>
</tr>
<tr>
<td>( y = \text{last}(x) )</td>
<td>on ( \zeta_{syst} : y = \text{last}(x) )</td>
<td>outside ( \zeta_{syst} : y = \text{last}(x) )</td>
</tr>
<tr>
<td>( \zeta = \text{up}(x) )</td>
<td></td>
<td>( \zeta = \text{up}(x) )</td>
</tr>
<tr>
<td>( \text{der}(y) = x \text{ init } y_0 \text{ reset } z )</td>
<td></td>
<td>( \text{der}(y) = x \text{ init } y_0 \text{ reset } z )</td>
</tr>
<tr>
<td>( z = x \text{ every } \zeta \text{ init } y_0 )</td>
<td>( z = x \text{ every } \zeta \text{ init } y_0 )</td>
<td>( z = \text{pre}(x) \text{ init } y_0 )</td>
</tr>
</tbody>
</table>
Slicing Continuous/Discrete

Slicing relies on typing A/C/D (A: any)
Key to good code generation

Typing A/C/D
- Variables and Wires
- Blocks (difficult)

How difficult is it on real languages?
Slicing Continuous/Discrete

Slicing relies on typing A/C/D (A: any)
Key to good code generation

Typing A/C/D
- Variables and Wires
- Blocks (difficult)

Doable in Simulink/Stateflow?

Albert Benveniste

20 January 2012
Slicing Continuous/Discrete

Slicing relies on typing A/C/D (A: any)
Key to good code generation
Typing A/C/D
  • Variables and Wires
  • Blocks (difficult)
Ongoing effort in Modelica 3.X

Albert Benveniste
20 January 2012
Slicing
Continuous/Discrete

Typing A/C/D

- Variables and Wires
- Blocks (difficult)

let hybrid ball (y₀, y'₀, start) =
let rec init y = y₀
and automaton
  | Await →
  do
    der y = 0.0
  until start then Bounce (y'₀)
  done
| Bounce (v) →
  local c, y ' in
  do
    and der y' = −9.81 init v
    and der y = y '
    and c = up(−. y)
  until c on (y' < eps) then Await
    | c then Bounce (−0.9 * . y ') 
  done
end

in
y

Albert Benveniste
20 January 2012
Any need for an effort on theory?

- time
- causality & dependencies
What is a good model of time?

Discrete time is subtle:

- **Continuous dynamics and discrete events**
- **Cascades of causally related events**
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?

- **Chattering**
- **Looks discrete but time progresses**

Control interferes with

- Cascades of events
- Continuous dynamics (sampling)

Unwanted couplings due to solvers

- Fast/Slow dynamics
- Zero-crossings, hot/cold restart

---

**D:** time is logical, not metric; events in a reaction are causally related; no date, no duration

**C:** time is metric; events are dated; actions have a duration

**H:** time can be logical or metric; events in a reaction are causally related; dates, durations
What is a good model of time?

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Control interferes with
- Cascades of events
- Continuous dynamics (sampling)

Unwanted couplings due to solvers
- Fast/Slow dynamics
- Zero-crossings, hot/cold restart


What is a good model of time?

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Control interferes with
- Cascades of events
- Continuous dynamics (sampling)

Unwanted couplings due to solvers
- Fast/Slow dynamics
- Zero-crossings, hot/cold restart


Super-dense time $\mathbb{R} \times \mathbb{N}$
What is a good model of time?

Discrete time is subtle:
• Continuous dynamics and discrete events
• Cascades of causally related events
• Non-determinism in discrete events
• Non-determinism in cascades of events

How discrete is discrete?
• Chattering
• Looks discrete but time progresses

Control interferes with
• Cascades of events
• Continuous dynamics (sampling)

Unwanted couplings due to solvers
• Fast/Slow dynamics
• Zero-crossings, hot/cold restart


H: time can be logical or metric; events in a reaction are causally related; dates, durations; almost so…
What is a good model of time?

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Control interferes with
- Cascades of events
- Continuous dynamics (sampling)

Unwanted couplings due to solvers
- Fast/Slow dynamics
- Zero-crossings, hot/cold restart

Non-standard time

\[ T = \{ n \partial | n \in \mathbb{N}^* \} \]

\( \partial \): infinitesimal base step
\( \mathbb{N}^* \): non-standard integers

T is
- totally ordered like N, hence we can refer to \( t - 1 \) and \( t + 1 \)
- dense in \( \mathbb{R}_+ \)


What is a good model of time?

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Control interferes with
- Cascades of events
- Continuous dynamics (sampling)

Unwanted couplings due to solvers
- Fast/Slow dynamics
- Zero-crossings, hot/cold restart

Non-standard time \( T = \{n\partial | n \in \mathbb{N}^*\} \)

\( \partial \) : infinitesimal base step
\( \mathbb{N}^* \) : non-standard integers
\( T \) is
- totally ordered like \( \mathbb{N} \), hence we can refer to \( t - 1 \) and \( t + 1 \)
- dense in \( \mathbb{R}_+ \)

With the two we can write:
\[
\dot{x} = f(x, t) \iff x_{t+1} = x_t + \partial \times f(x_t, t)
\]

and we can study cascades of events as in usual discrete time formalisms; we can type continuous/discrete and implement slicing

\( H \): time can be logical or metric; events in a reaction are causally related; dates, durations
Further fundamental issues: Causality and dependencies

Avoiding unwanted coupling effects
- Seemingly unrelated sub-systems influence each other through their respective time-scales: due to adaptive step size and the use of a global solver
- Countermeasure: multi-solver

Physical modeling from first principles (Modelica, Simscape)
- Infer causality and direction of flows
- For DAE: index reduction

These are all issues related to causality
- State of the art: handled globally at system level: combinatorial explosion
- Need for a compositional processing
A proof of concept tool development: Zelus

By Timothy Bourke and Marc Pouzet
Zelus
(Zero-crossings for Lustre)
Objectives

Two steps
1. No support for DAE
2. With support for DAE (future)

Technical objectives
- Simple and consistent semantics
- Slicing Continuous/Discrete
- Elegant treatment of automata
- Certifiability

```math
let hybrid ball (y_0, y'_0, start) =
  let
  rec init y = y_0
  and automaton
    | Await →
    | do
      der y = 0.0
    | until start then Bounce (y'_0)
    | done
    | Bounce (v) →
      local c, y ' in
      do
        and der y' = -9.81 init v
        and der y = y '
        and c = up(-. y)
      until c on ( y ' < eps ) then
        Await
        | c then Bounce (-0.9 * . y ')
      )
    | done
end
```
Zelus
(Zero-crossings for Lustre)

Objectives

Two steps
1. No support for DAE
2. With support for DAE (future)

Technical objectives
• Simple and consistent semantics
• Slicing Continuous/Discrete
• Elegant treatment of automata
• Certifiability

Principles

Slicing
• Compilation ensures that the discrete part of Zelus is compiled the usual way
• Off-the-shelf numerical solver (Sundials CVODE V2.6.0 from Lawrence Livermore)

Based on source-to-source transformations
• Helps for certification
• Automata transformed to data-flow constructs
Zelus
(Zero-crossings for Lustre)

Objectives

Two steps
1. No support for DAE
2. With support for DAE (future)

Technical objectives
• Simple and consistent semantics
• Slicing Continuous/Discrete
• Elegant treatment of automata
• Certifiability

Status

Typing D/C: first version, still ongoing

Highlights difficulties related to solvers
• Hot/Cold restart
• What kind of solver for physical system modeling?

Future developments
• Multi-solver (to avoid unwanted coupling)
• Symbolic non-standard simulation (to avoid bothering solvers with ODE that stop immediately)
Recap
Virtual engineering for CPS: the enablers

Seamless modeling

- PLM/CAD
- Multi-physics
- CPS functions
- Architecture
  - ECU
  - communications
- Resources
  - time,
  - energy,
  - memory,
  - weight

Quality, certification

Heterogeneity

Executable models
- reproducible executions
- avoid unwanted non-determinism

Analyzable models
Must reuse basic domains
- Discrete time
- Continuous, real, time
- Resource models
Slicing Continuous/Discrete

Typing A/C/D
- Variables and Wires
- Blocks (difficult)

Slicing
Rich structure in
- D: hierarchical and concurrent automata
- C: multi-solver / distributed solvers

Causality
- Handling cascades of events
- Support for multi-solver
- Physical modeling from first principles
Using a good model of time

Discrete time is subtle:
- Continuous dynamics and discrete events
- Cascades of causally related events
- Non-determinism in discrete events
- Non-determinism in cascades of events

How discrete is discrete?
- Chattering
- Looks discrete but time progresses

Control interferes with
- Cascades of events
- Continuous dynamics (sampling)

Unwanted couplings due to solvers
- Fast/Slow dynamics
- Zero-crossings, hot/cold restart

Non-standard time \( T = \{n\partial | n \in \mathbb{N}^*\} \)
\( \partial \) : infinitesimal base step
\( \mathbb{N}^* \) : non-standard integers

We can write:
\[
\dot{x} = f(x, t) \iff x_{t+1} = x_t + \partial \times f(x_t, t)
\]
and study cascades of events as in discrete time

We can type continuous/discrete
and implement slicing

\( \text{H: time can be logical or metric; events in a reaction are causally related; dates, durations} \)
Modeling Physics in CPS is difficult

History has shown that developing theory for real-time programming of reactive systems was needed (Synchronous Languages...)

Multi-Physics modeling embedded in systems wide virtual models is needed and requires theoretical efforts too

Useful efforts toward supporting
- Reuse and imports of models
- Certification
## Challenges for hybrid systems modeling research: a summary

### Accurate modeling

- Good language supporting D/C slicing and DAE (modeling from first principles)
- Models of D/C time reusable across the different views of virtual modeling
- Avoid unwanted coupling artifacts due to solvers
- Reproducible simulations (determinism)

### Large system modeling

- Well structured D modeling
- Well structured C modeling
  - Distributed solvers
- DAE index reduction
  - Avoid combinatorial explosion
- Efficient abstraction techniques
  - Multi-scale
  - Multiple time-scales
References related to the talk


merci