Model-Driven Engineering with **Formal Models** for **Embedded Systems**

INRIA Aoste

AOSTE

Direction : 2 (1+1)

Directeur : de SIMONE, Robert – DR

Co-directeur : SOREL, Yves – DR

AOSTE

Permanents : 6 (5+1)

ANDRE, Charles – Professeur
DEANTONI, Julien – McF
HOGIE Luc – IR
MALLET, Frédéric – McF
PERALDI-FRATI, Marie-Agnès – McF
POTOP BUTUCARU, Dumitru – CR

Ingénieurs experts : 3 (2+1)

BOUCARON, Julien – Docteur
FERRERO, Benoît – Master
DE RAUGLAUDRE, Daniel – Ingénieur

Direction : 2 (1+1)

Directeur : de SIMONE, Robert – DR
Co-directeur : SOREL, Yves – DR

Doctorants : 4 (3+1)

COADOU, Anthony – Master
LE TALLEC, Jean-François – Master
MEHMOOD KAHN, Aamir – Master
MAROUF Mohamed – Master

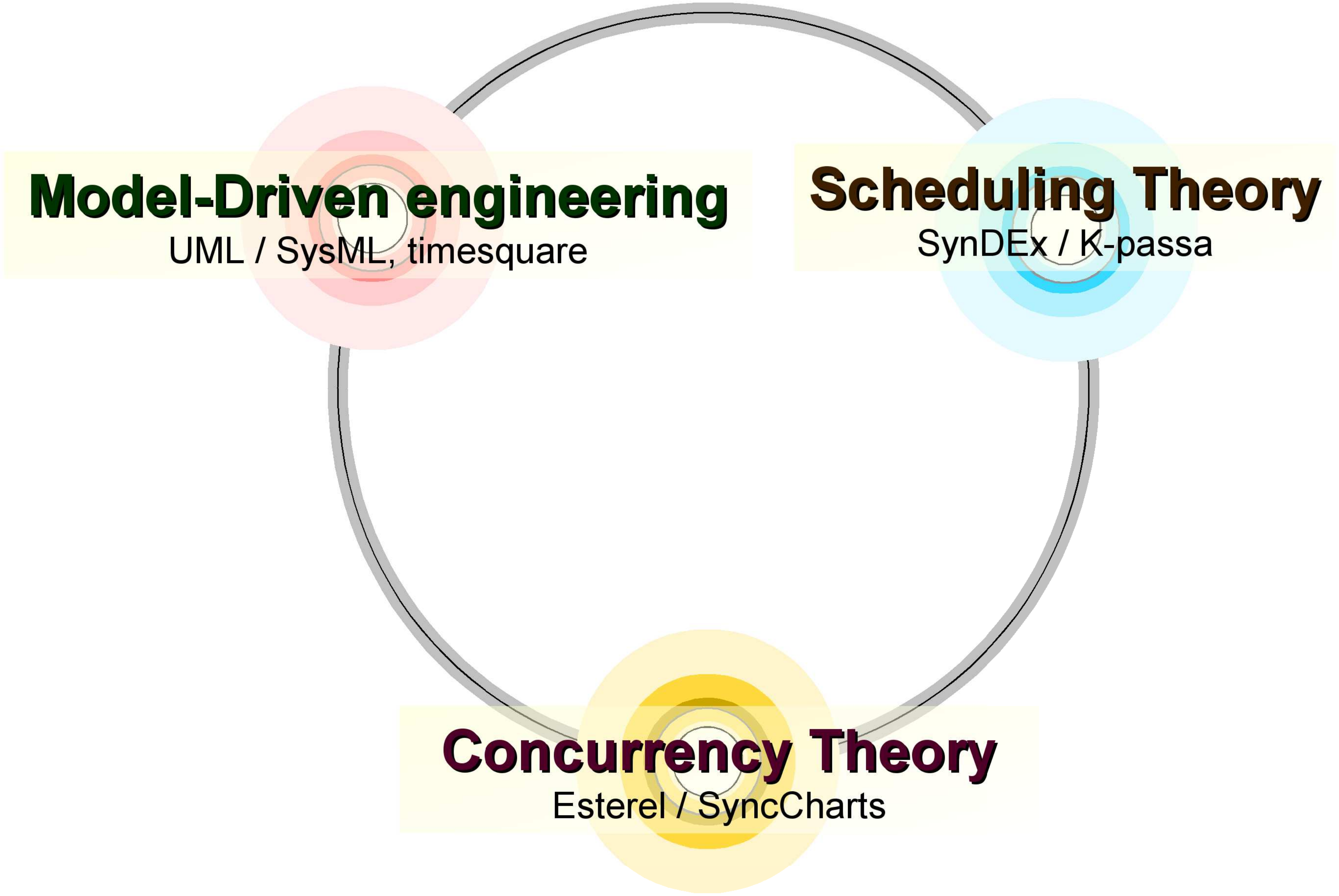
Post-Doctorants : 3 (2+1)

GASCON, Régis – Docteur
GLITIA, Calin – Docteur
MEUMEU YOMSI, Patrick – Docteur

Assistantes : >2 (2+?)

DEVAUCHELLE, Sandra – I3S
LACHAUME, Patricia – INRIA

Goal: associate 3 domains



Model-Driven engineering

UML / SysML, timesquare

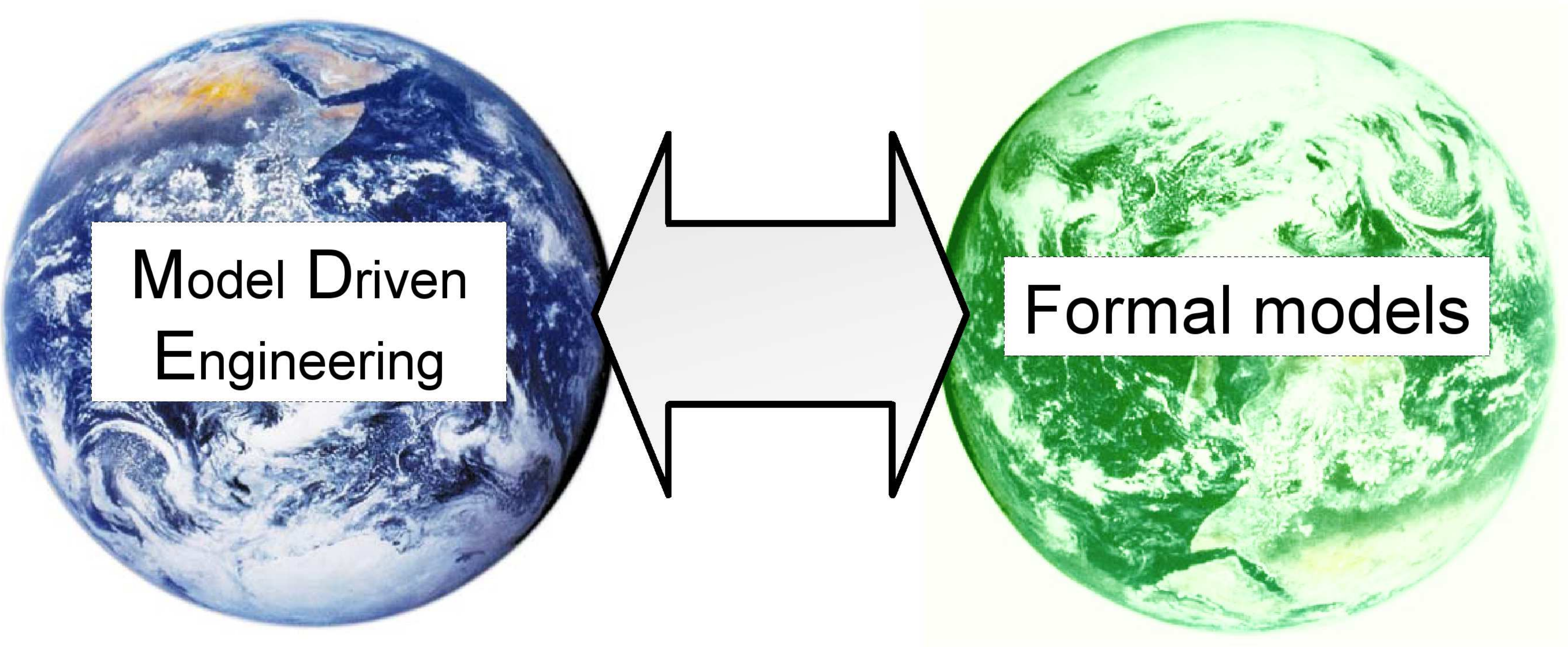
Scheduling Theory

SynDEx / K-passa

Concurrency Theory

Esterel / SyncCharts

Where are we going ?



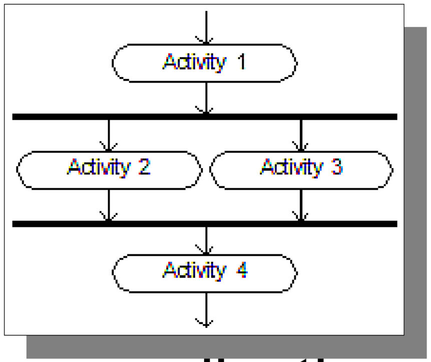
Model Driven Engineering

Formal models

Tools, standards, engineering world
→ replacement of grammars

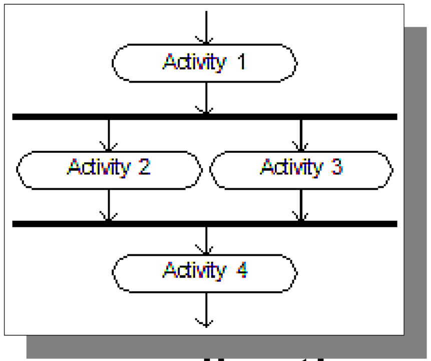
Verification / validation
→ mathematical mean to adress a problem

Domain

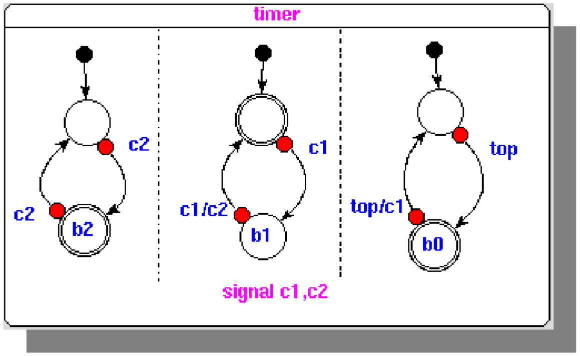


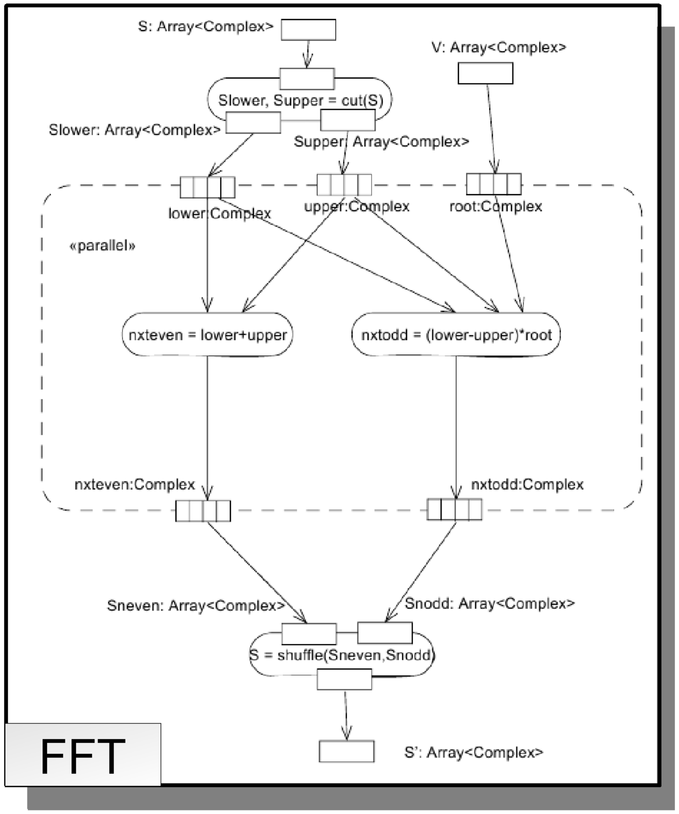
- Embedded systems
  - Concurrent and heterogeneous applications

Domain



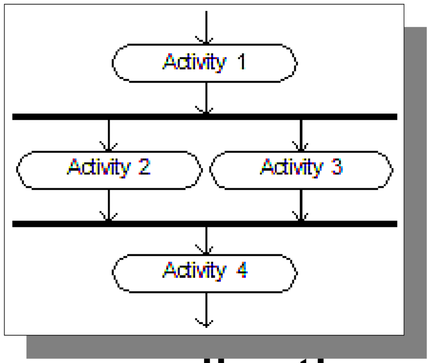
- Embedded systems
  - Concurrent and heterogeneous applications
    - Signal/image processing
    - Control software
    - …



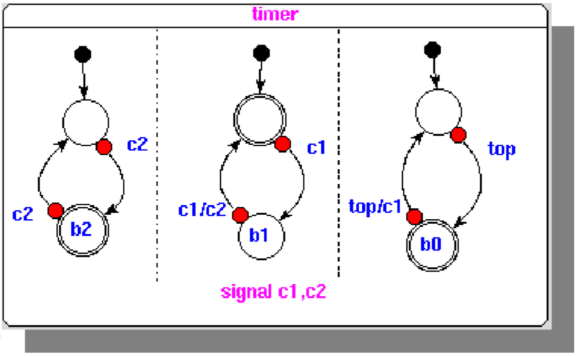


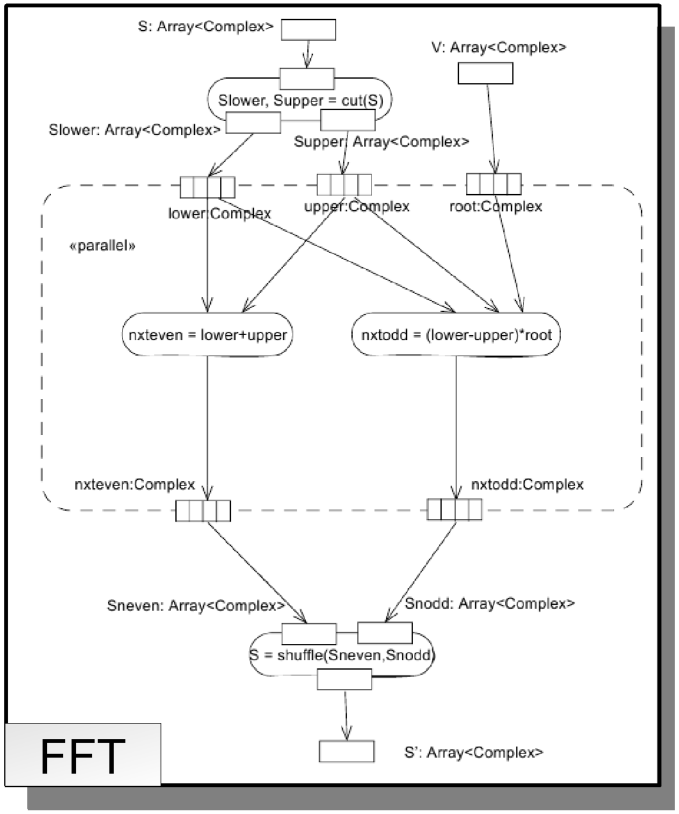
FFT

Domain

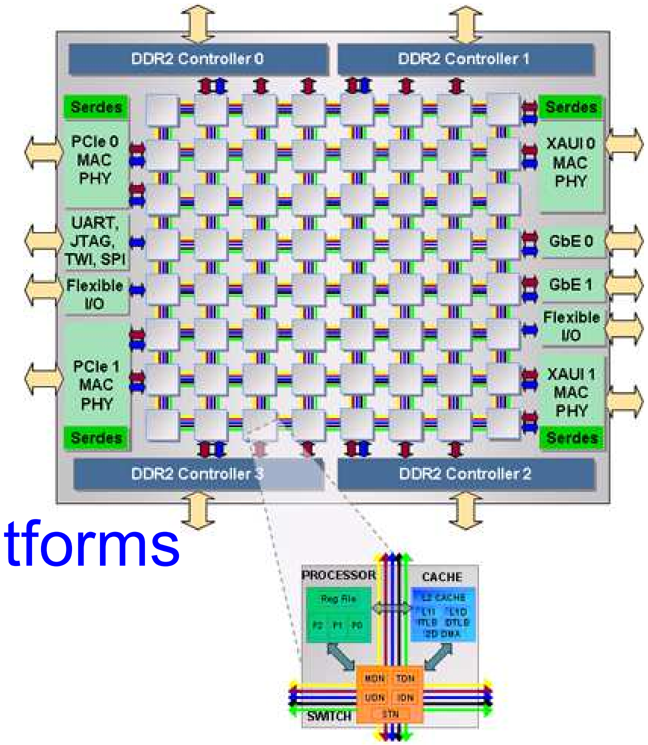


- Embedded systems
  - Concurrent and heterogeneous applications
    - Signal/image processing
    - Control software
    - …
- Constraints
  - safety-critical
  - hard real-time
  - Extra functional
    - low power
    - Cost
    - …

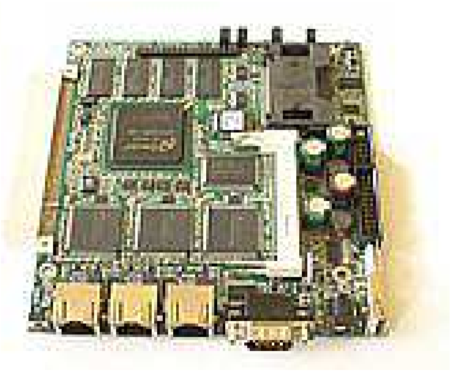


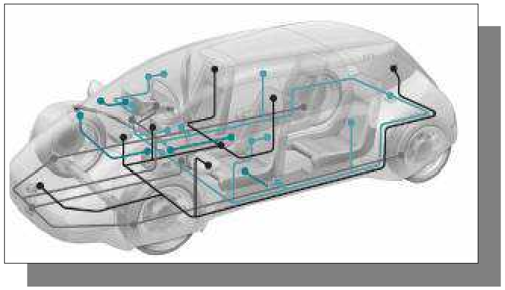


Domain



- Embedded systems
  - Concurrent and heterogeneous applications
  - Heterogeneous parallel execution platforms
- Constraints
  - safety-critical
  - hard real-time
  - Extra functional
    - low power
    - Cost
    - …



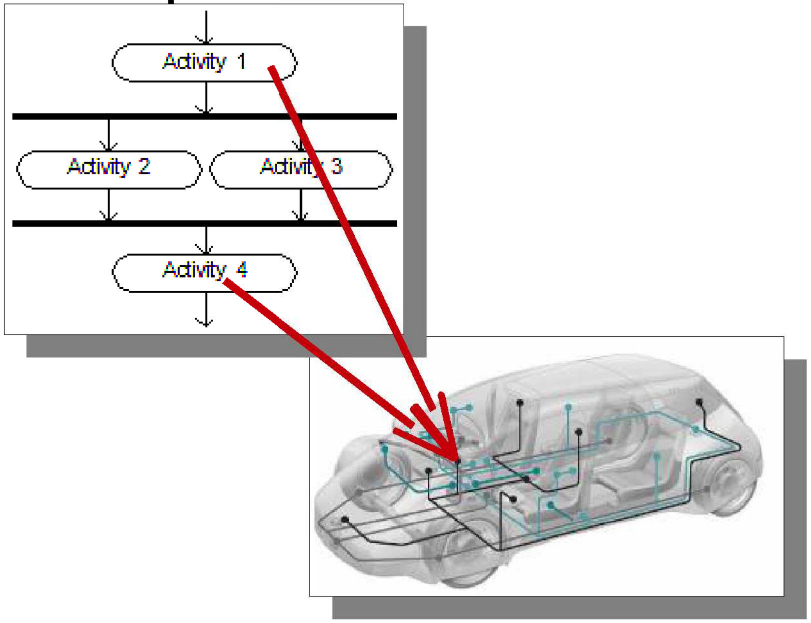


Domain

- Embedded systems
  - Concurrent and heterogeneous applications

    mapped to

  - Heterogeneous parallel execution platforms
- Constraints
  - safety-critical
  - hard real-time
  - Extra functional
    - low power
    - Cost
    - …



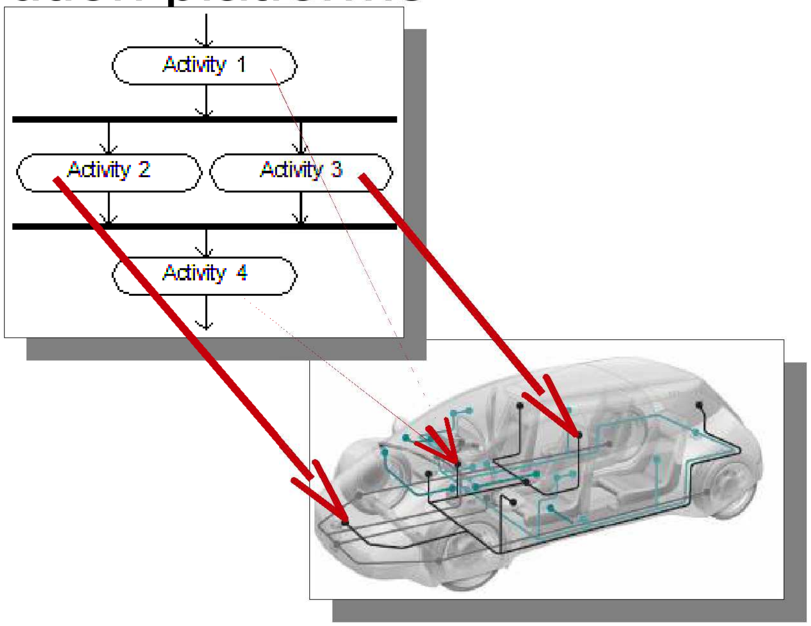
Domain

- Embedded systems
  - Concurrent and heterogeneous applications

mapped to

  - Heterogeneous parallel execution platforms

- Constraints
  - safety-critical
  - hard real-time
  - low power
  - …

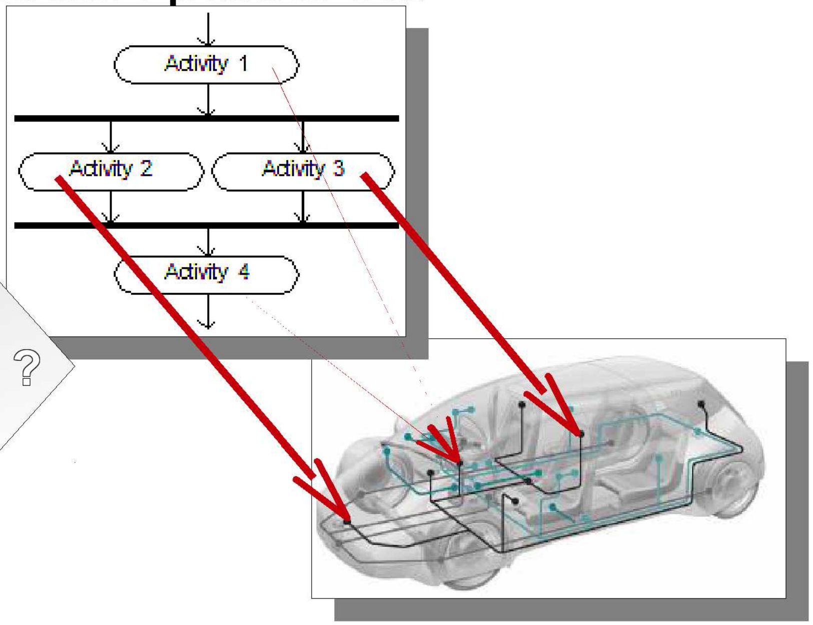


Domain

- Embedded systems
  - Concurrent and heterogeneous applications

mapped to

  - Heterogeneous parallel execution platforms



Is mapping ok ?

- Constraints
  - safety-critical
  - hard real-time
  - low power
  - …

Domain

- Embedded systems
  - 1 – Concurrent and heterogeneous applications

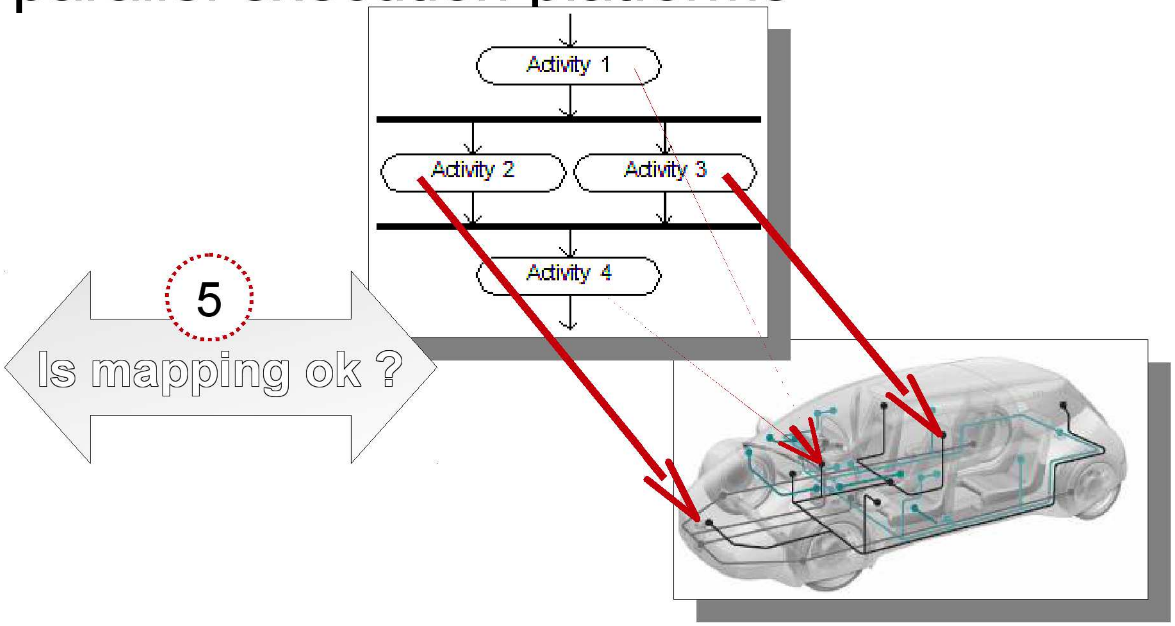
    3 *mapped to*

    Need for formalisms to adress these 5 concerns

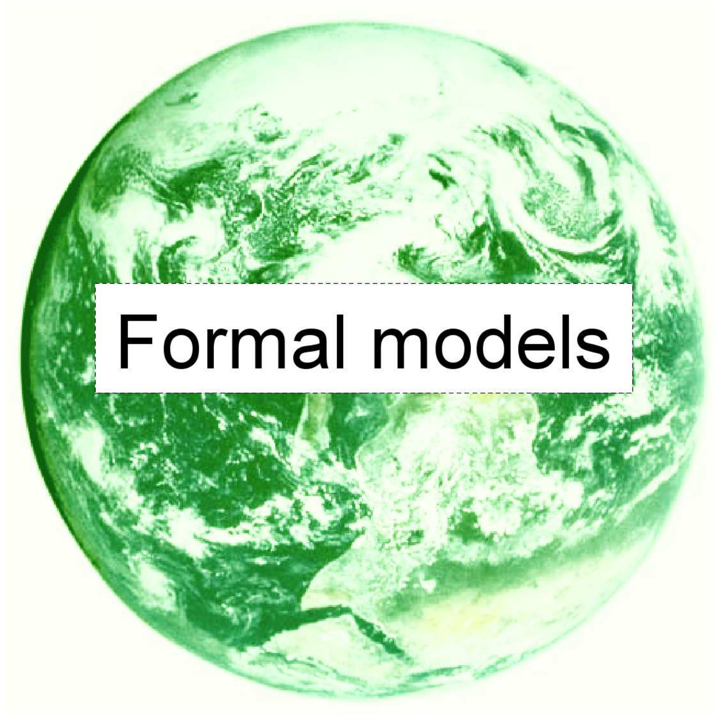
  - 2 – Heterogeneous parallel execution platforms

4 Constraints

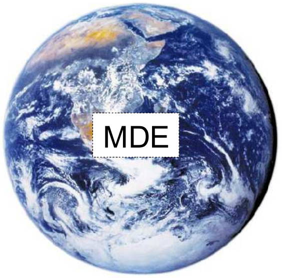
- safety-critical
- hard real-time
- low power
- …



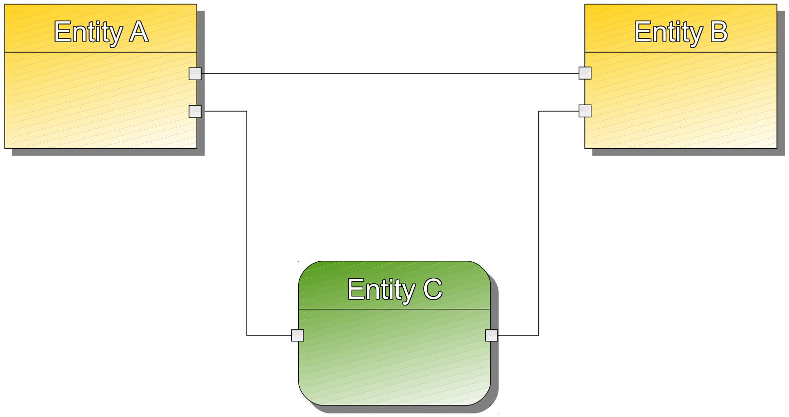
How to deal with the domain ?



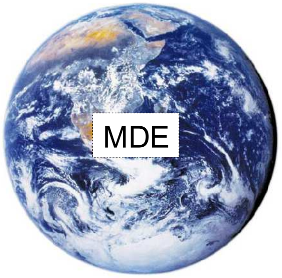
Model-Driven Engineering



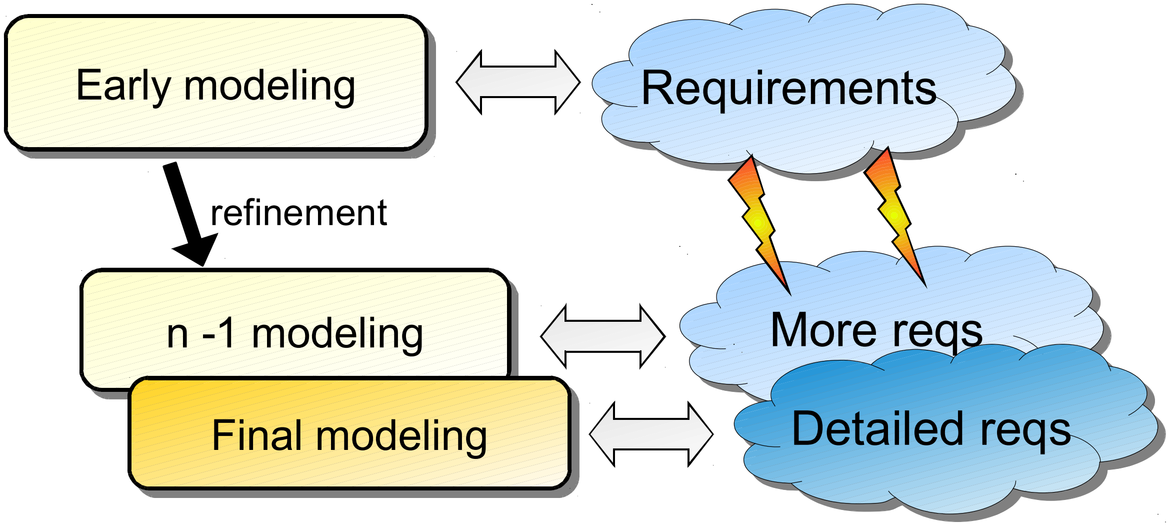
- **Goals**
  - High-level descriptions of systems and demands
    - Designing in the large !!
    - Focusing on concerns one at a time (aspects)
    - Weaving concerns
  - Support for system structuring all along design cycle (≠ code)



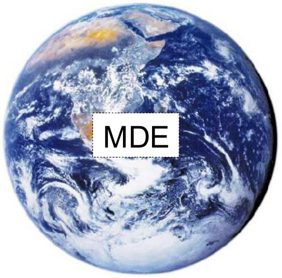
Model-Driven Engineering



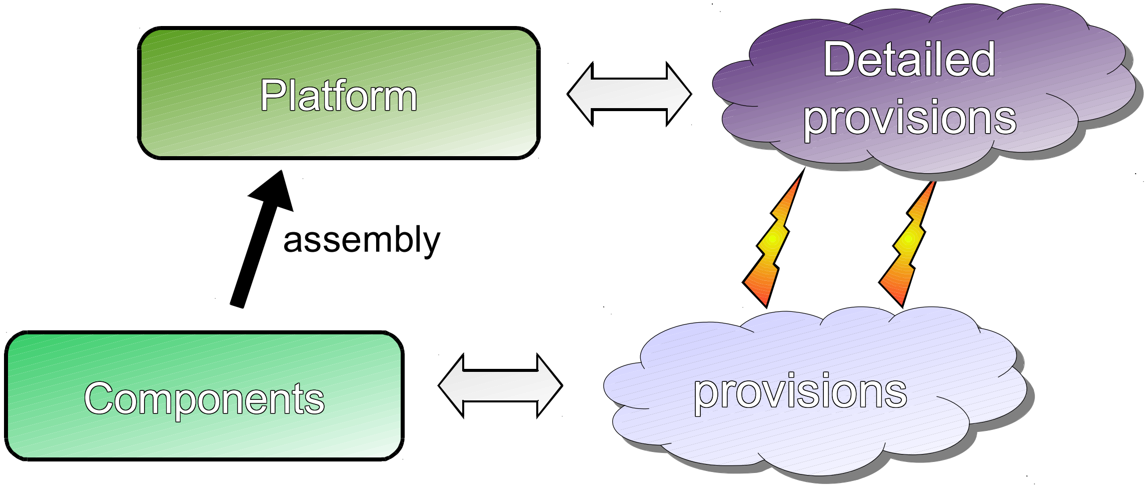
- **Goals**
  - – High-level descriptions of systems and demands
  - – Support for system architecturing all along design cycle (≠ code)
  - – Early expression of **requirements** and specifications
    - All along the design cycle
    - Requirements are detailled with the specification refinement



Model-Driven Engineering

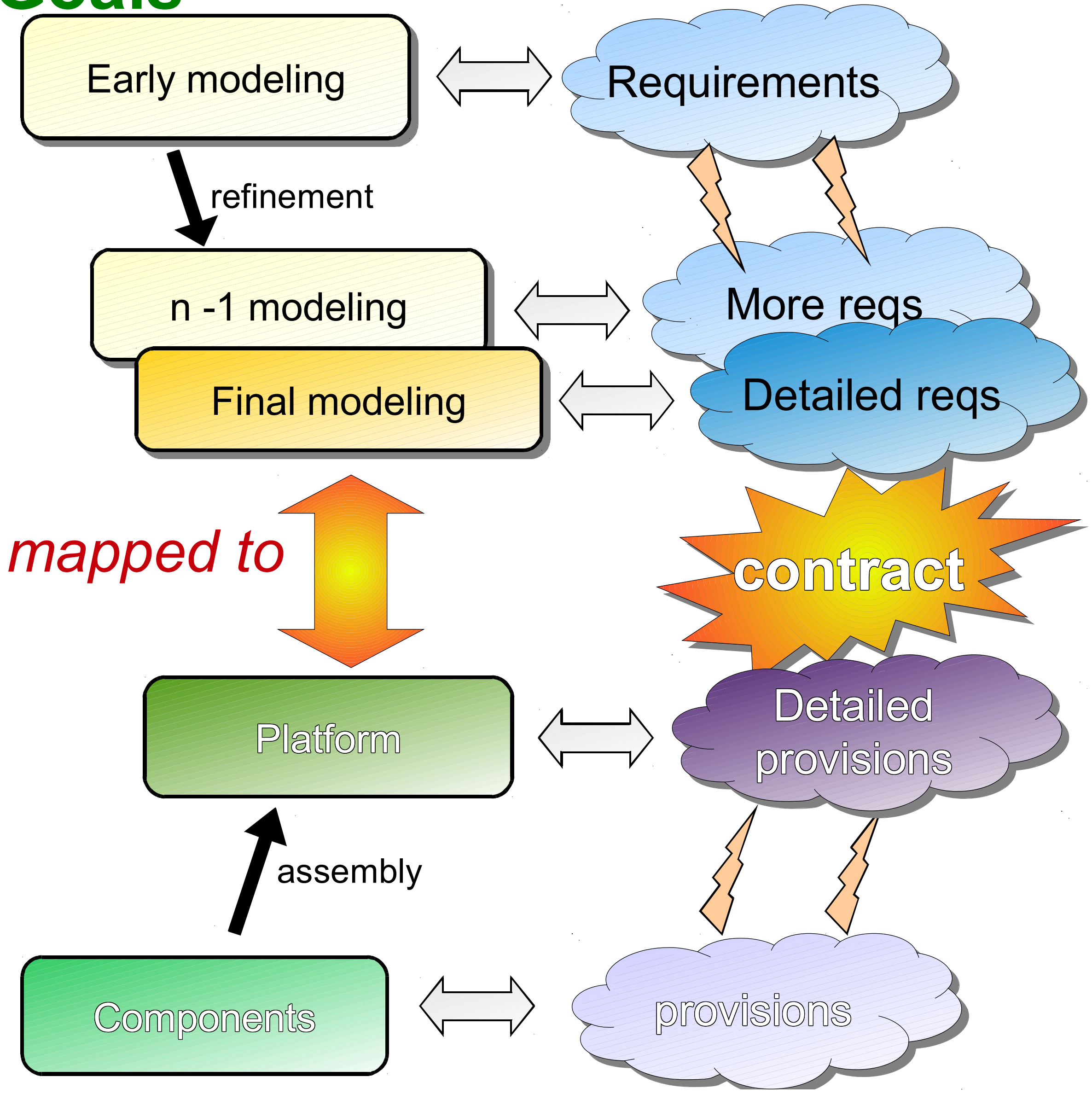


- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease **Reuse** of existing parts / **components**



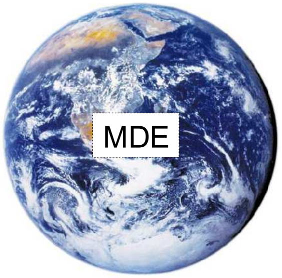
Model-Driven Engineering

- **Goals**

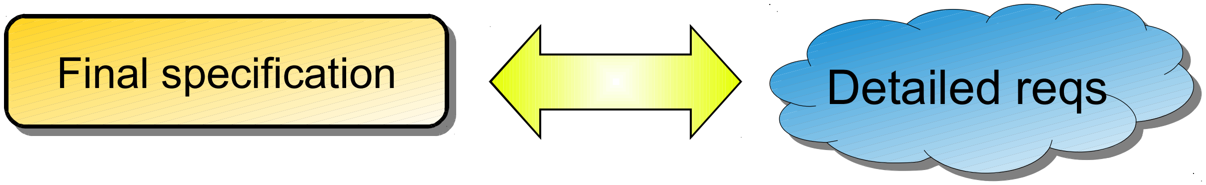


MDE

Model-Driven Engineering

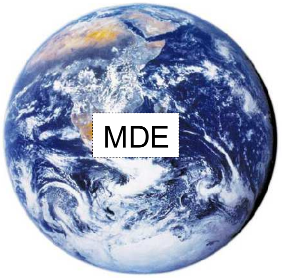


- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - Traceability

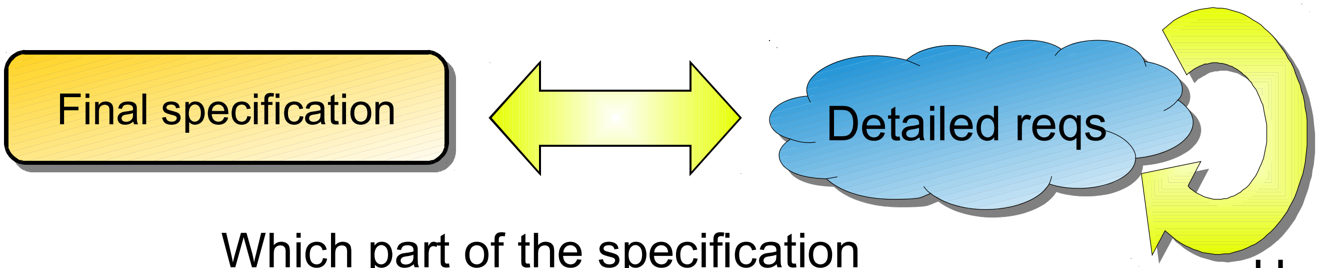


Which part of the specification satifies a specific requirement, …

Model-Driven Engineering



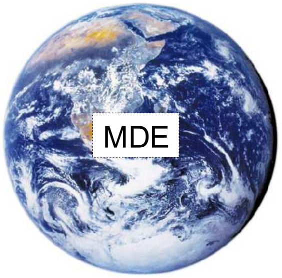
- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - **Traceability**



Which part of the specification satifies a specific requirement, …

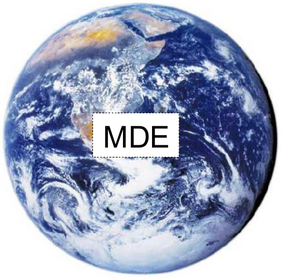
How requirements are linked together

Model-Driven Engineering



- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - Traceability
  - **Communication** between various teams (inside or across companies), **documentation**

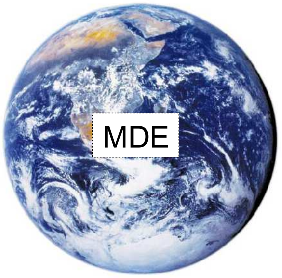
Model-Driven Engineering



- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - Traceability
  - Communication between various teams (inside or across companies), documentation

- **Current Shortcomings**
  - discrepancies, lack of semantics or even precise interpretation

Model-Driven Engineering



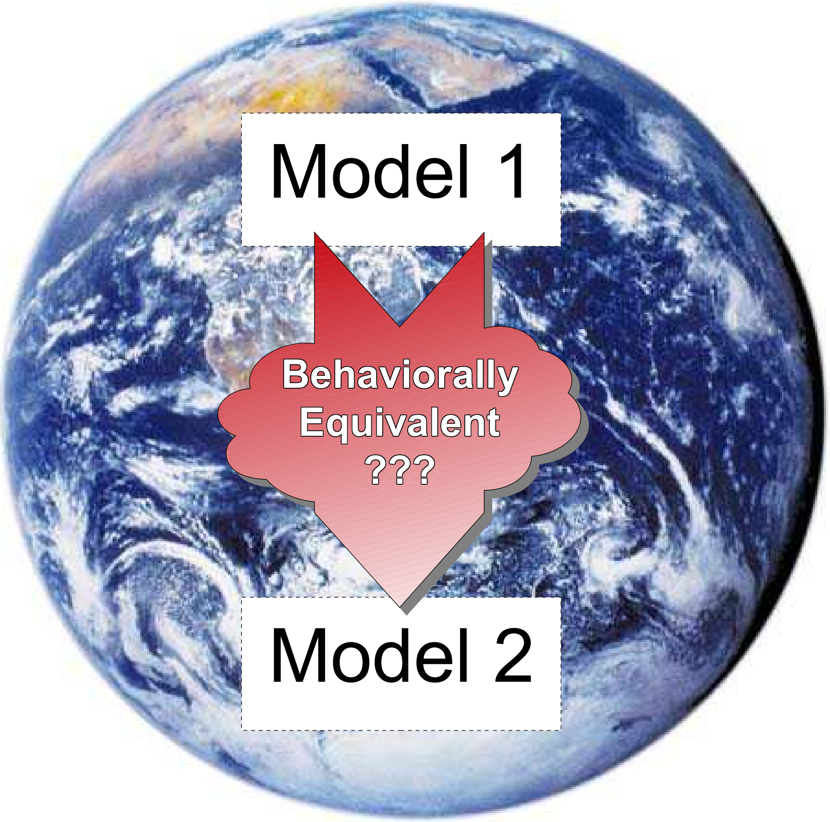
- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - Traceability
  - Communication between various teams (inside or across companies), documentation
- **Current Shortcomings**
  - discrepancies, lack of precise semantics or even of semantics

    → **tools suffer from this**

Model-Driven Engineering

Current Shortcomings

- discrepancies, lack of precise semantics or even of semantics
  - → **tools suffer from this**

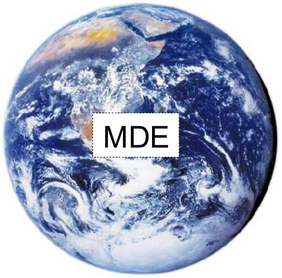


Model 1

Behaviorally Equivalent ???

Model 2

Model-Driven Engineering

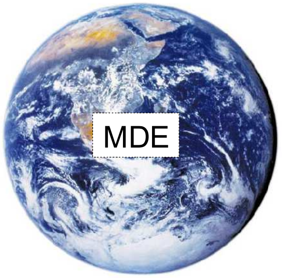


- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - Traceability
  - Communication between various teams (inside or across companies), documentation

- **Current Shortcomings**
  - discrepancies, lack of semantics or even precise interpretation

    → **tools suffer from this**
  - universality dissolves into particularisms

Model-Driven Engineering

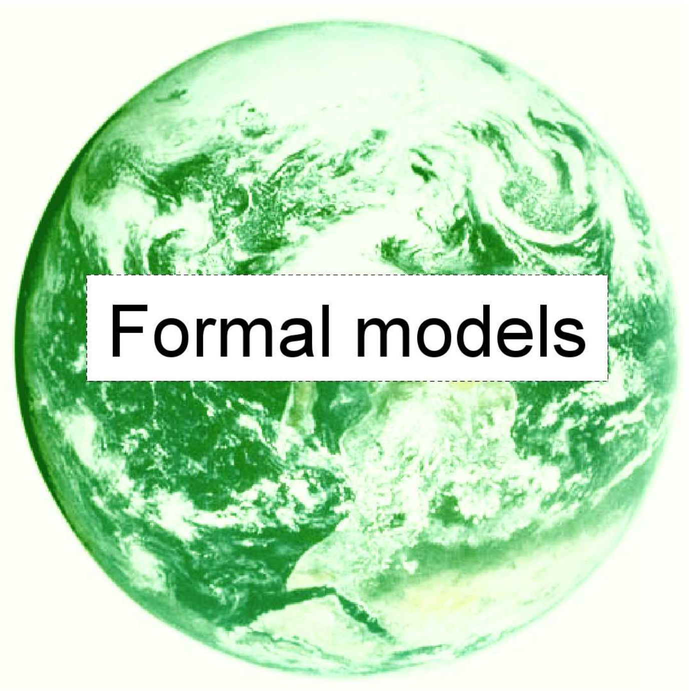


- **Goals**
  - High-level descriptions of systems and demands
  - Support for system architecturing all along design cycle (≠ code)
  - Early expression of requirements and specifications
  - Ease Reuse of existing part / components
  - Traceability
  - Communication between various teams (inside or across companies), documentation

- **Current Shortcomings**
  - discrepancies, lack of semantics or even precise interpretation

    → **tools suffer from this**
  - universality dissolves into particularisms

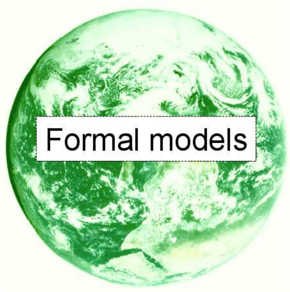
Unformal Models (and methods)



Formal models

Models of Computation and Communications (MoCCs) that exhibits explicit concurrency

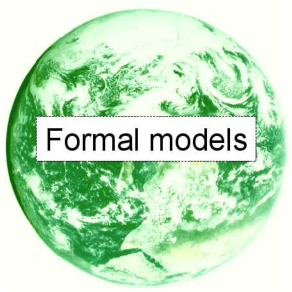
Formal Models (MoCC)



- **Goals**
  - Mathematical semantics
    - No Ambiguity

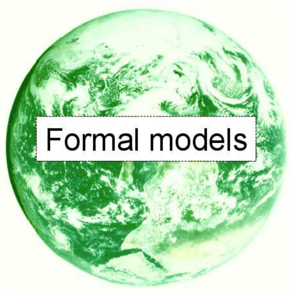
      → **Tools benefit from that !**

Formal Models (MoCC)



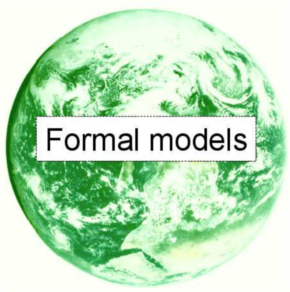
- **Goals**
  - Mathematical semantics
  - Powerful analysis and algorithmic methods

Formal Models (MoCC)



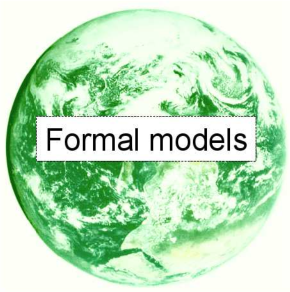
- **Goals**
  - Mathematical semantics
  - Powerful analysis and algorithmic methods
  - Optimization / verification

Formal Models (MoCC)

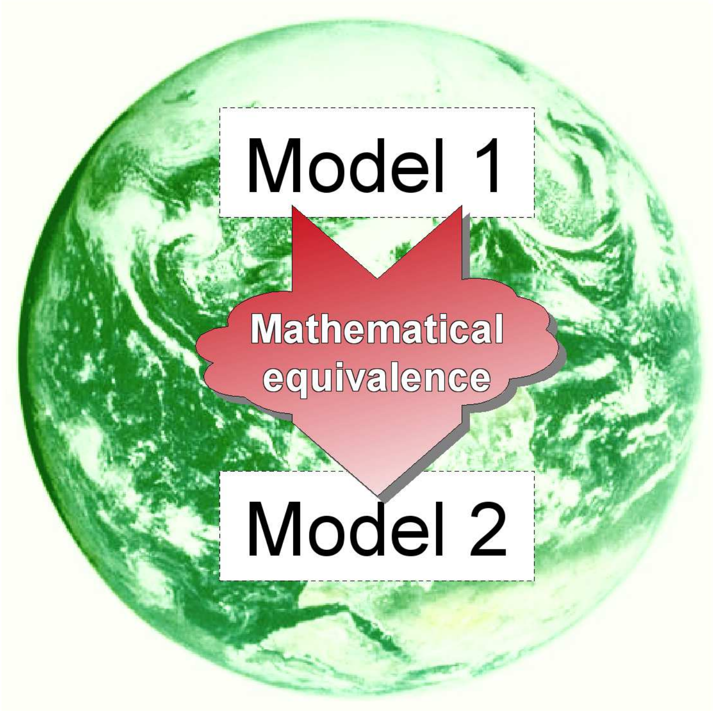


- **Goals**
  - Mathematical semantics
  - Powerful analysis and algorithmic methods
  - Optimization / verification
  - Guaranteed equivalence between code and model

Formal Models (MoCC)



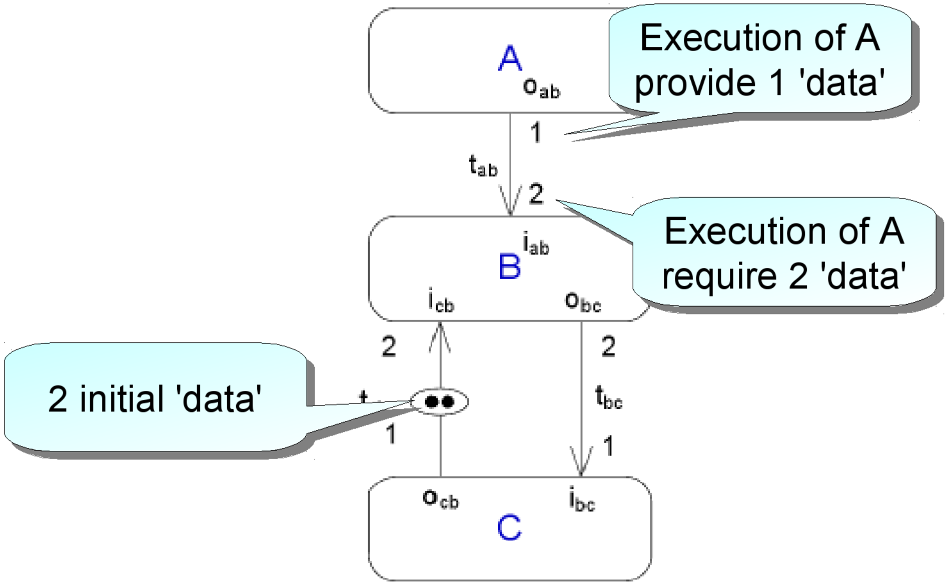
- **Goals**
  - Mathematical semantics
  - Powerful analysis and algorithmic methods
  - Optimization / verification
  - Guaranteed equivalence between code and model
  - Basis for well-founded transformations



Formal Models (MoCC)

A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



Execution of A provide 1 'data'

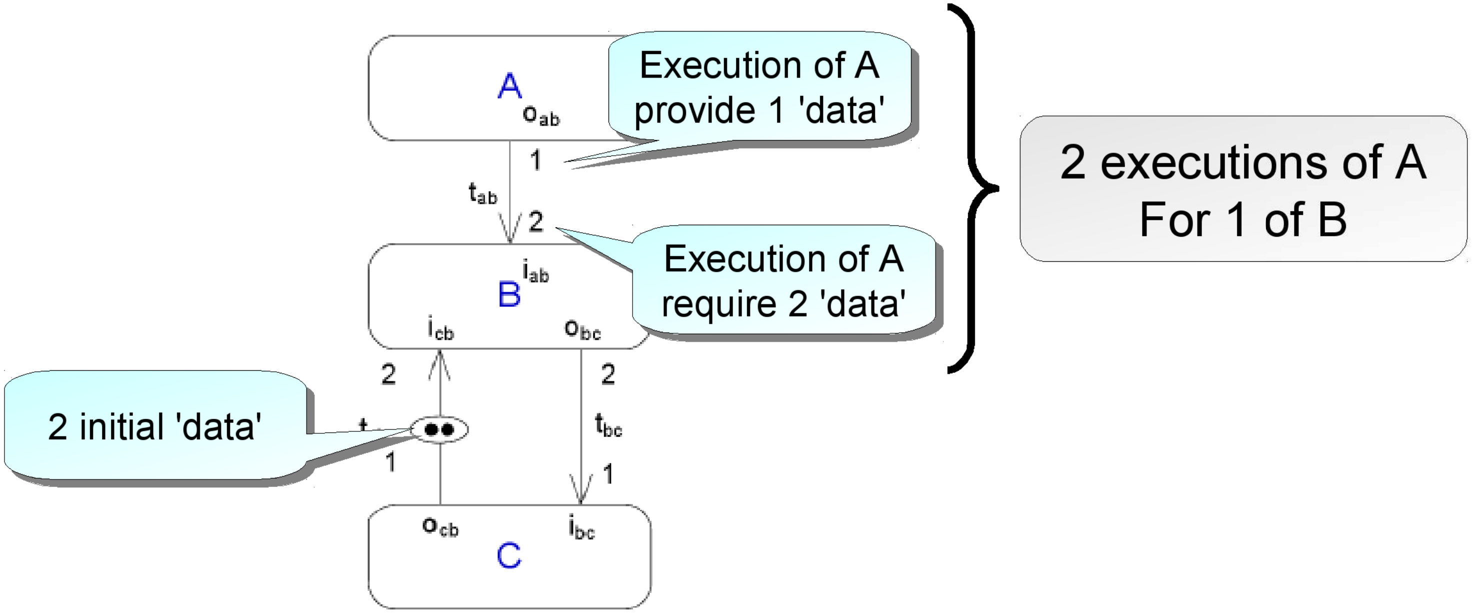
Execution of A require 2 'data'

2 initial 'data'

Formal Models

A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



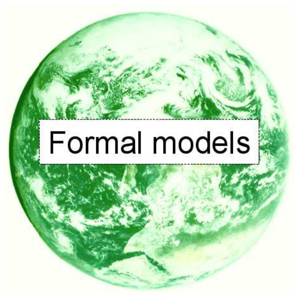
Execution of A provide 1 'data'

Execution of A require 2 'data'

2 initial 'data'

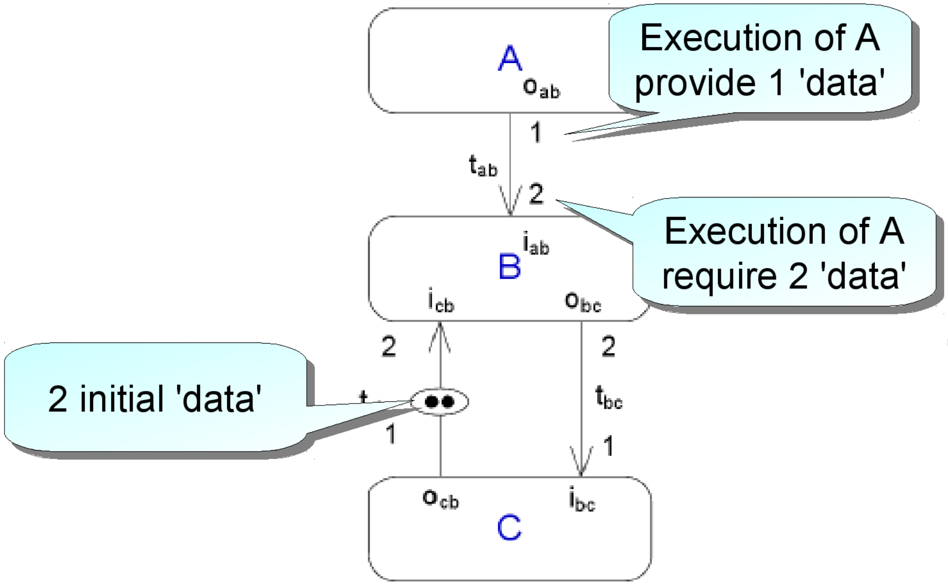
2 executions of A For 1 of B

Formal Models



A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



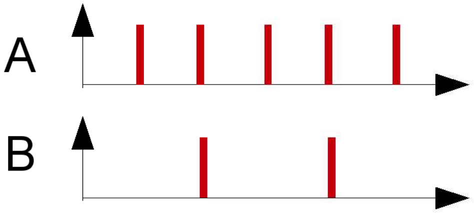
Execution of A provide 1 'data'

Execution of A require 2 'data'

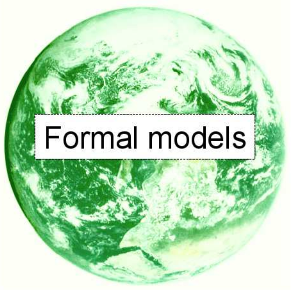
2 initial 'data'

Logical Time

2 executions of A
For 1 of B

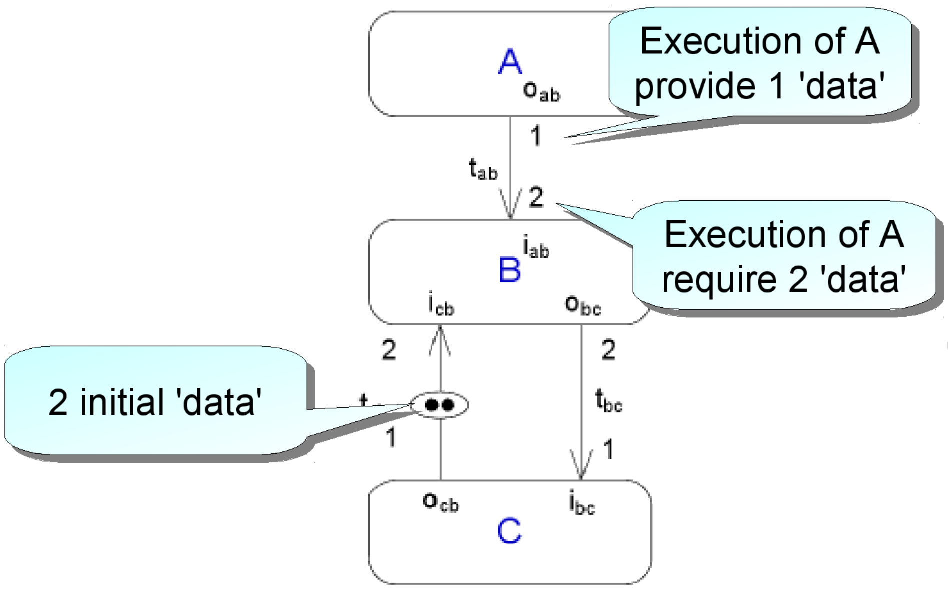


Formal Models



A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



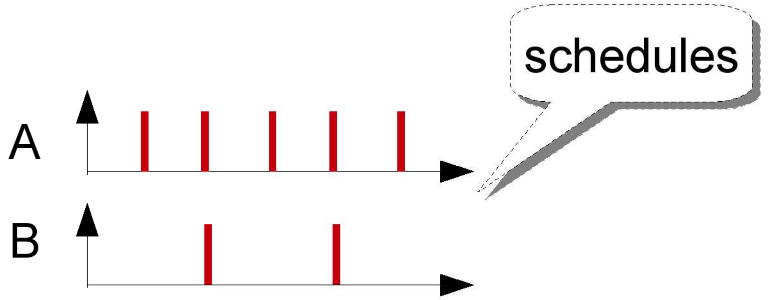
Execution of A provide 1 'data'

Execution of A require 2 'data'

2 initial 'data'

Logical Time

2 executions of A
For 1 of B

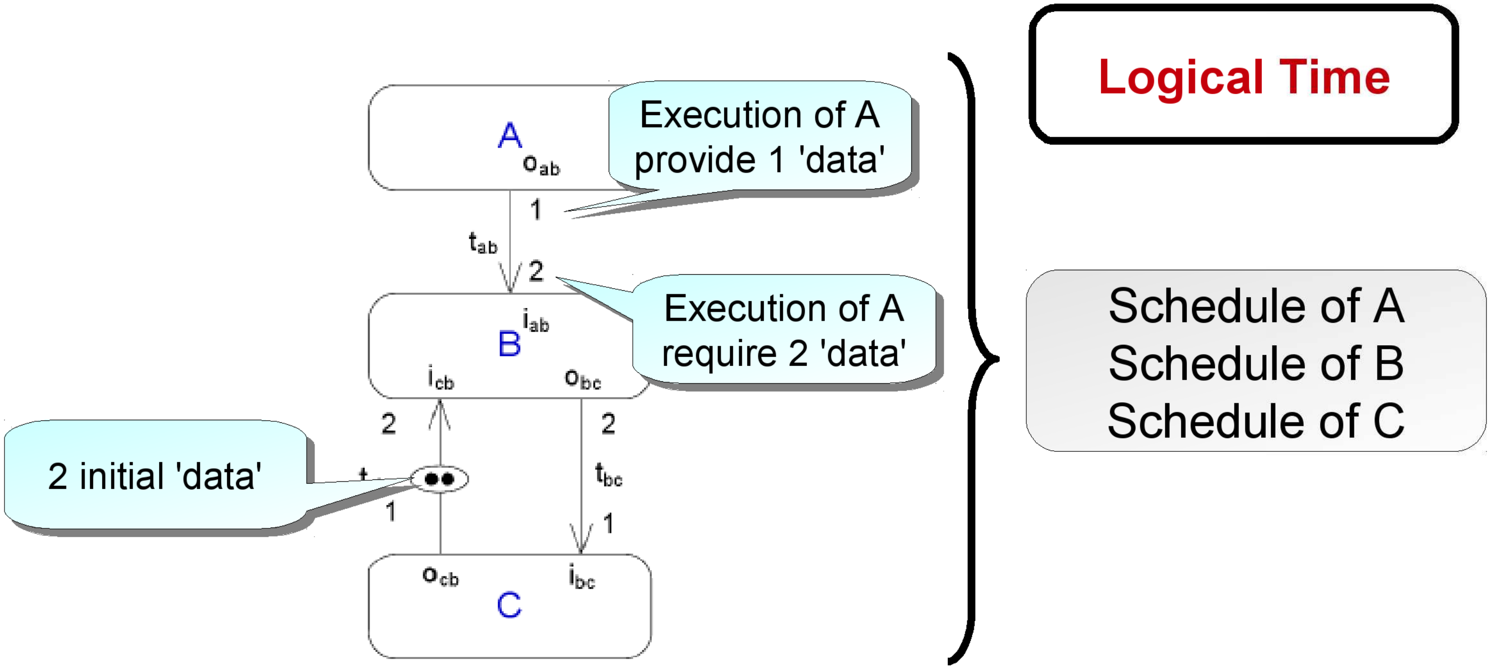


schedules

Formal Models

A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



Execution of A provide 1 'data'

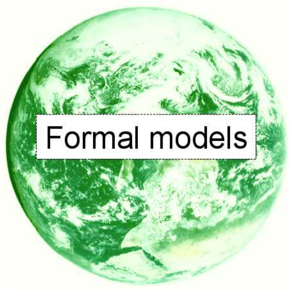
Execution of A require 2 'data'

2 initial 'data'

Logical Time

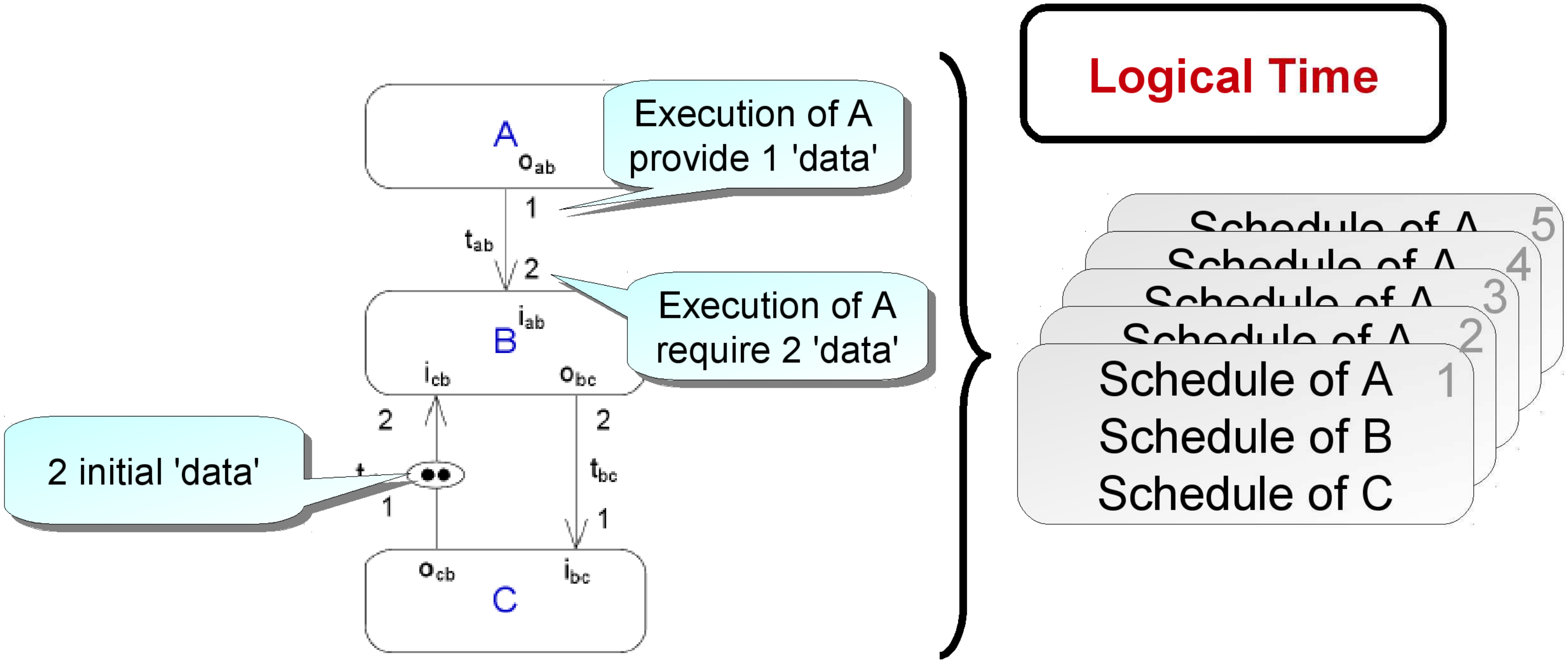
Schedule of A
Schedule of B
Schedule of C

Formal Models



A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



Execution of A provide 1 'data'

Execution of A require 2 'data'

2 initial 'data'

Logical Time

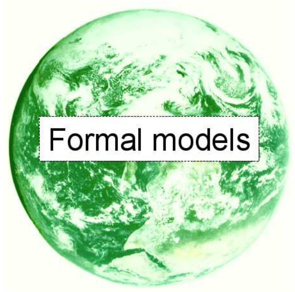
Schedule of A

Schedule of A

Schedule of A

Schedule of A

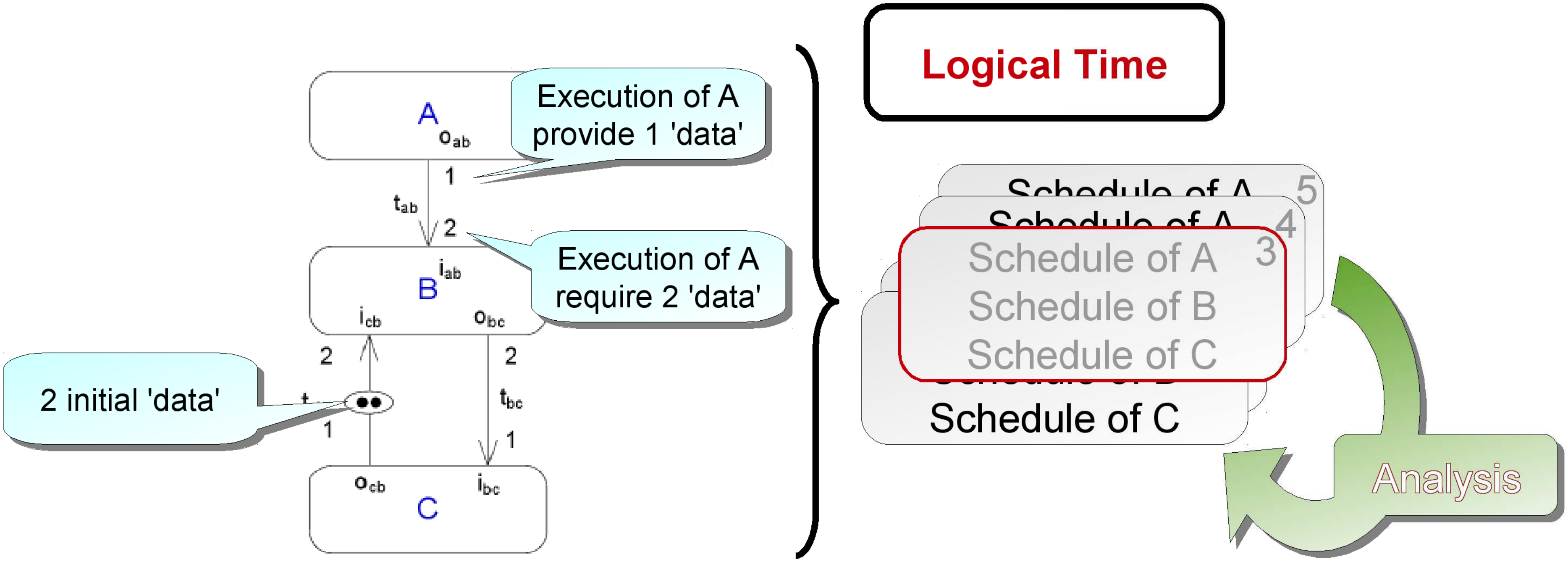
Schedule of A
Schedule of B
Schedule of C



Formal Models

A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network

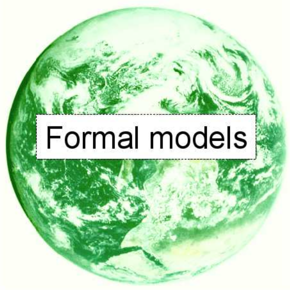

Execution of A provide 1 'data'

Execution of A require 2 'data'

2 initial 'data'

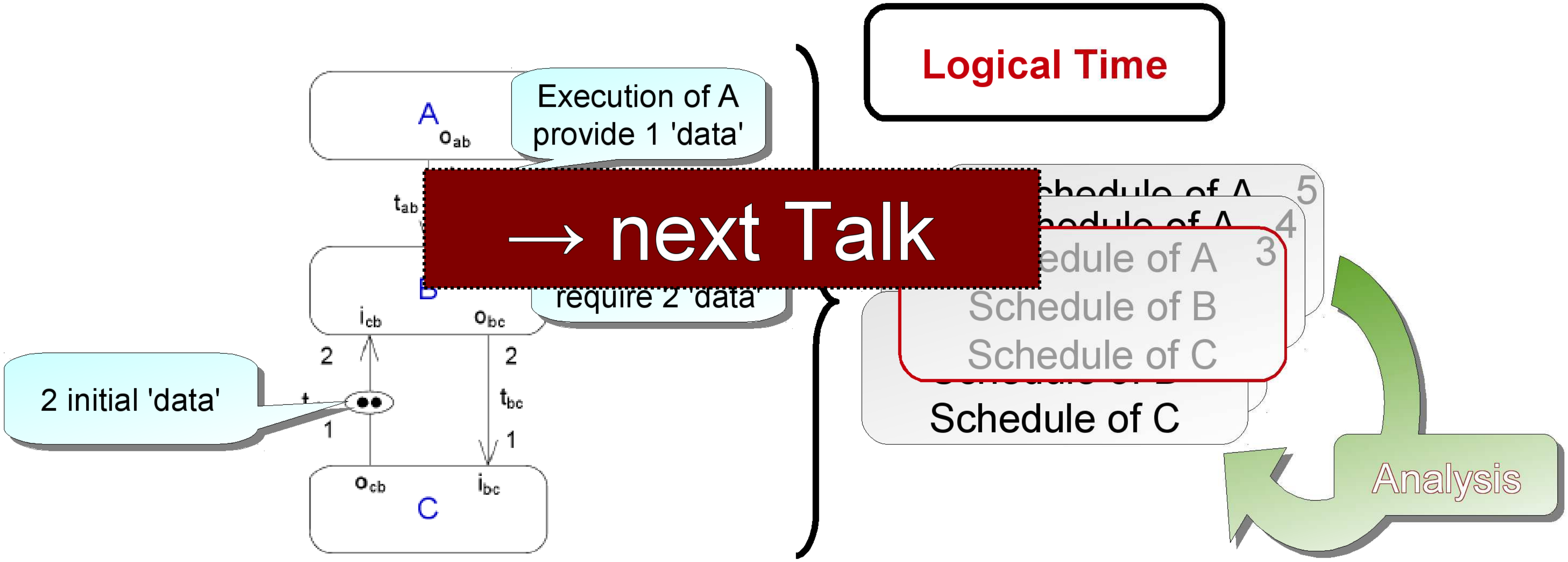
Logical Time

Formal Models



A quick snapshot of relevant MoCC

- Process Networks
  - Marked Graph
  - Synchronous Data Flow
  - Kahn Process Network



Execution of A provide 1 'data'

require 2 'data'

2 initial 'data'

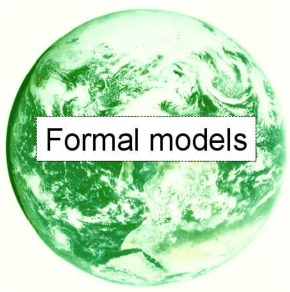
→ next Talk

Logical Time

Schedule of A
Schedule of A
Schedule of A
Schedule of B
Schedule of C
Schedule of B
Schedule of C

Analysis

Formal Models



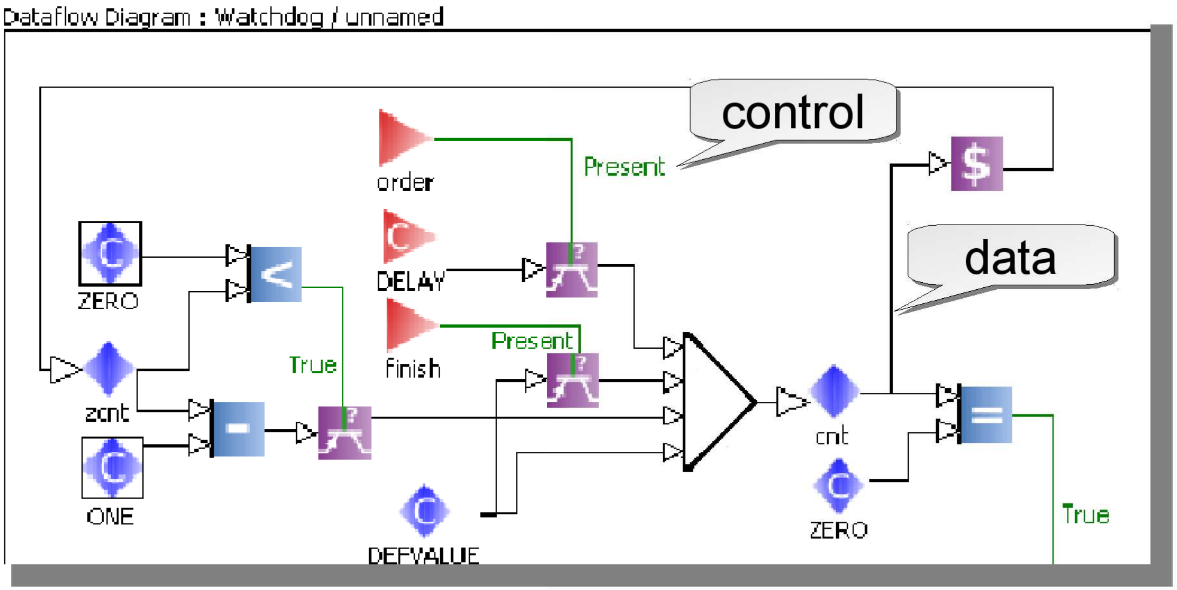
A quick snapshot of relevant MoCC

- Process Networks
- Synchronous languages

Formal Models

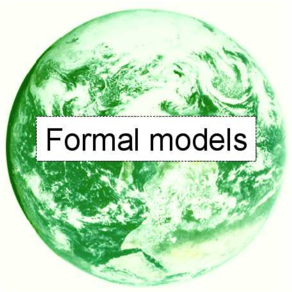
A quick snapshot of relevant formal models

- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony



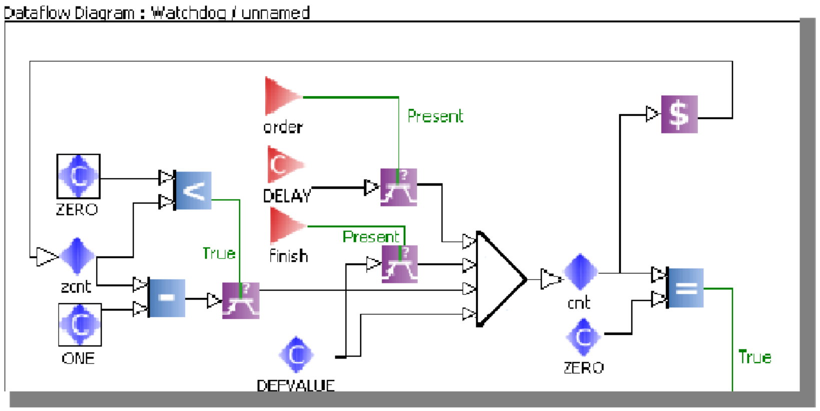
Logical Time
+
Multi-Clock

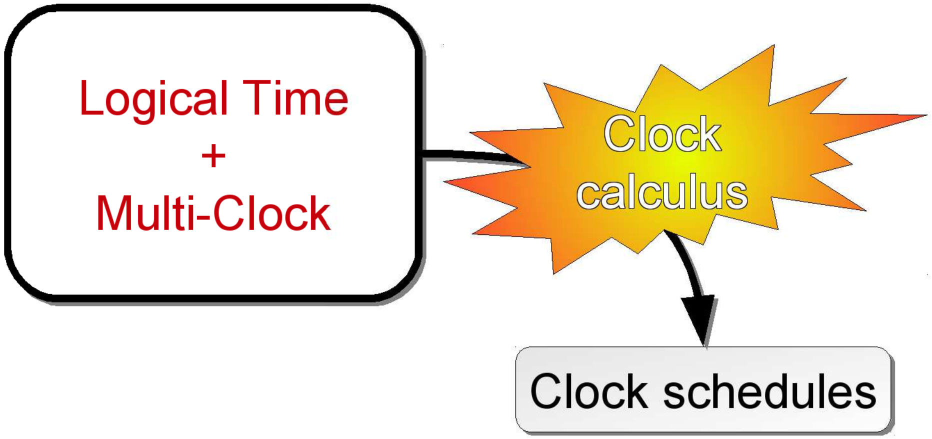
Formal Models



A quick snapshot of relevant formal models

- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony



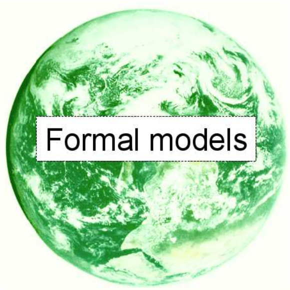


Logical Time
+
Multi-Clock

Clock calculus

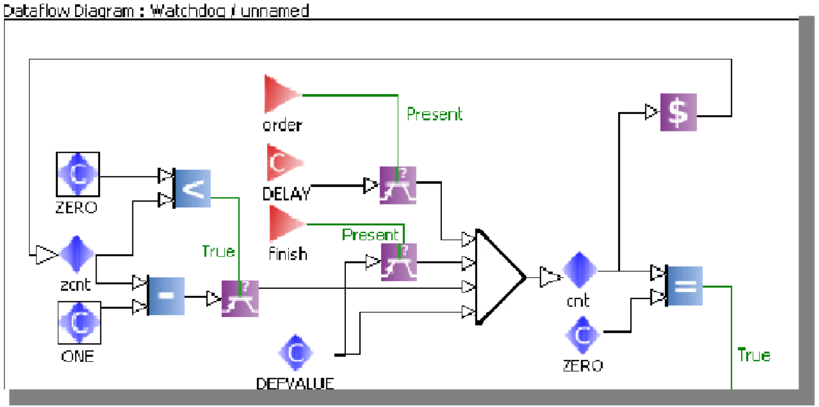
Clock schedules

Formal Models



A quick snapshot of relevant formal models

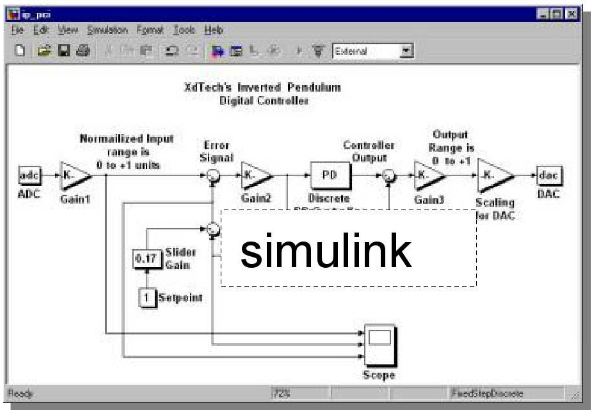
- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony



≈ equivalent

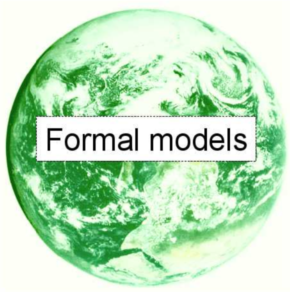
formal semantics

+



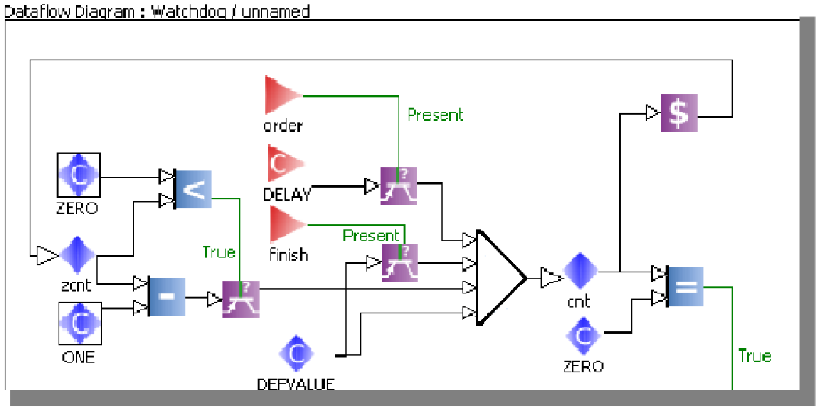
simulink

Formal Models



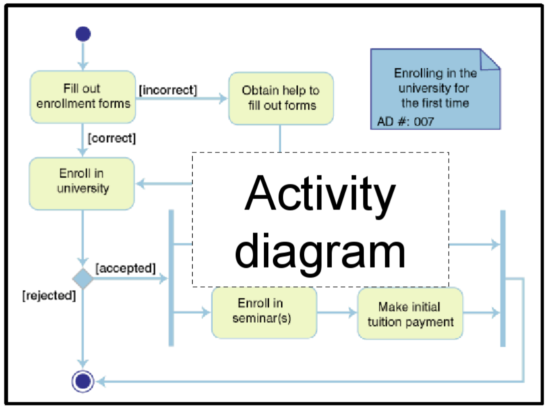
A quick snapshot of relevant formal models

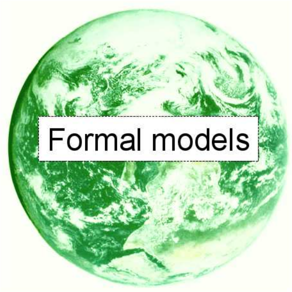
- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony



≈ equivalent

formal semantics +

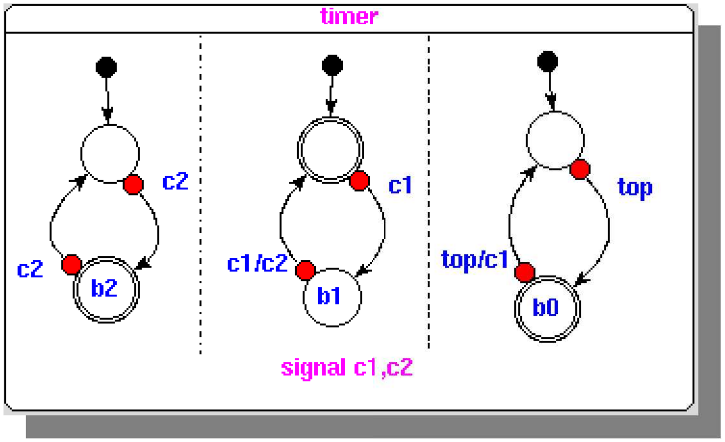

Fill out enrollment forms
[incorrect]
Obtain help to fill out forms
Enrolling in the university for the first time
AD #: 007
[correct]
Enroll in university
Activity diagram
[accepted]
[rejected]
Enroll in seminar(s)
Make initial tuition payment



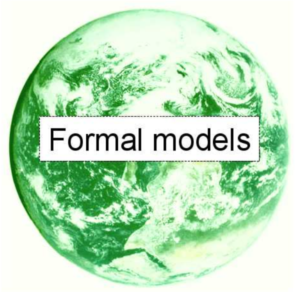
Formal Models

A quick snapshot of relevant formal models

- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony
  - Imperative: Esterel / SyncCharts

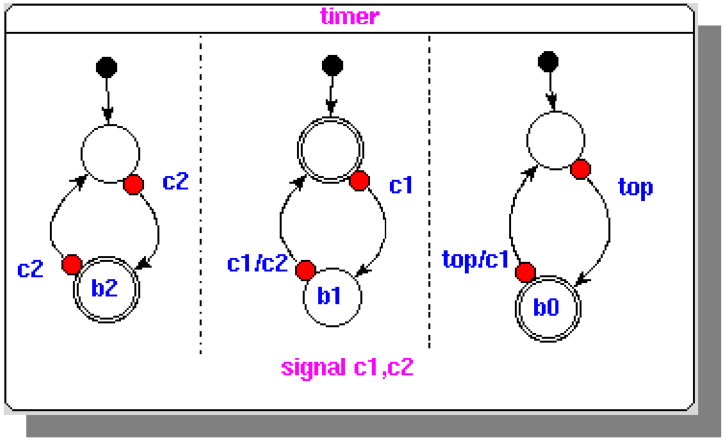


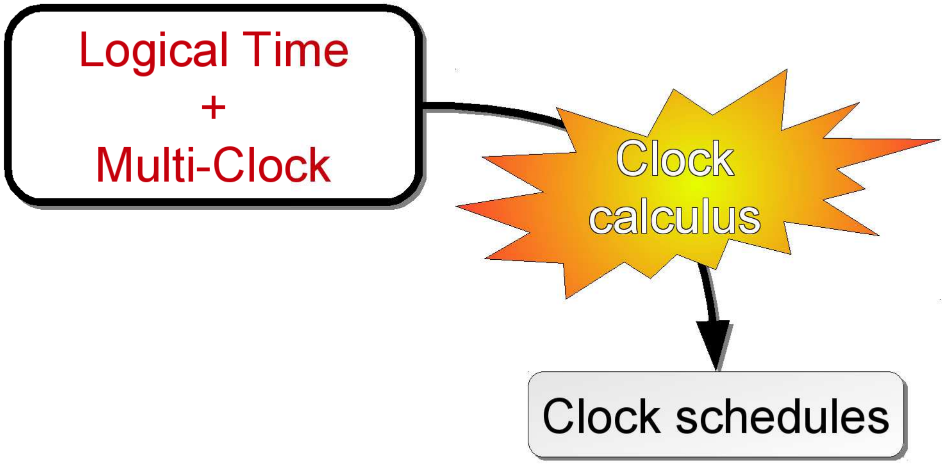
Formal Models



A quick snapshot of relevant formal models

- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony
  - Imperative Esterel / SyncCharts



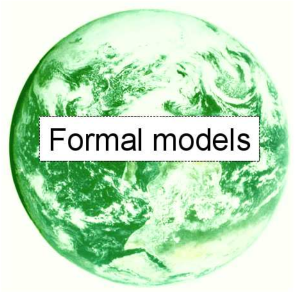


Logical Time
+
Multi-Clock

Clock calculus

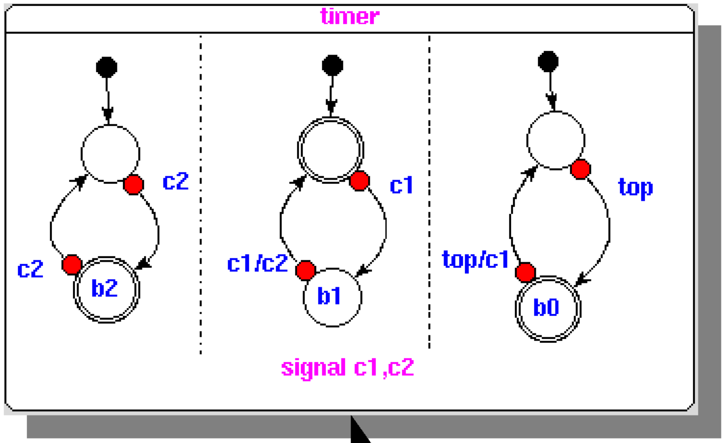
Clock schedules

Formal Models



A quick snapshot of relevant formal models

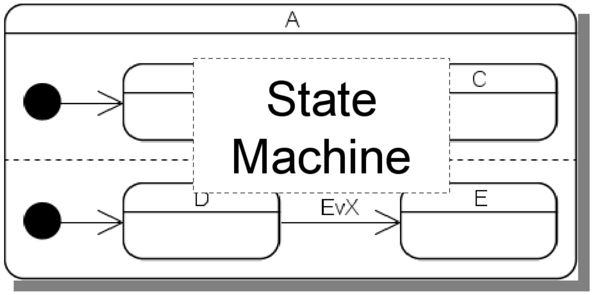
- Process Networks
- Synchronous languages
  - Declarative: Lustre / Scade, Signal / Polychrony
  - Imperative Esterel / SyncCharts



Everything is about activation conditions

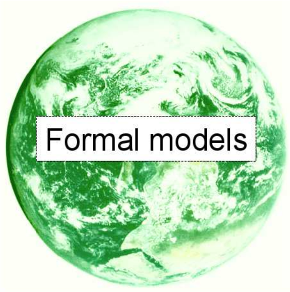
≈ equivalent

formal semantics +



State Machine

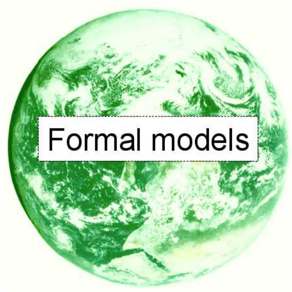
Formal Models



- **Goals**
  - Mathematical semantics
  - Powerful analysis and algorithmic methods
  - Optimization / verification
  - Guaranteed equivalence between code and model
  - Basis for well-founded transformations

- **Current Shortcomings**
  - **Distance from current mainstream engineering practice**
  - **Exotic formalisms for programmers (not C / C++ / java)**
  - **Need for a mathematical background**
  - **Most tools half-academic or confidential**

Formal Models

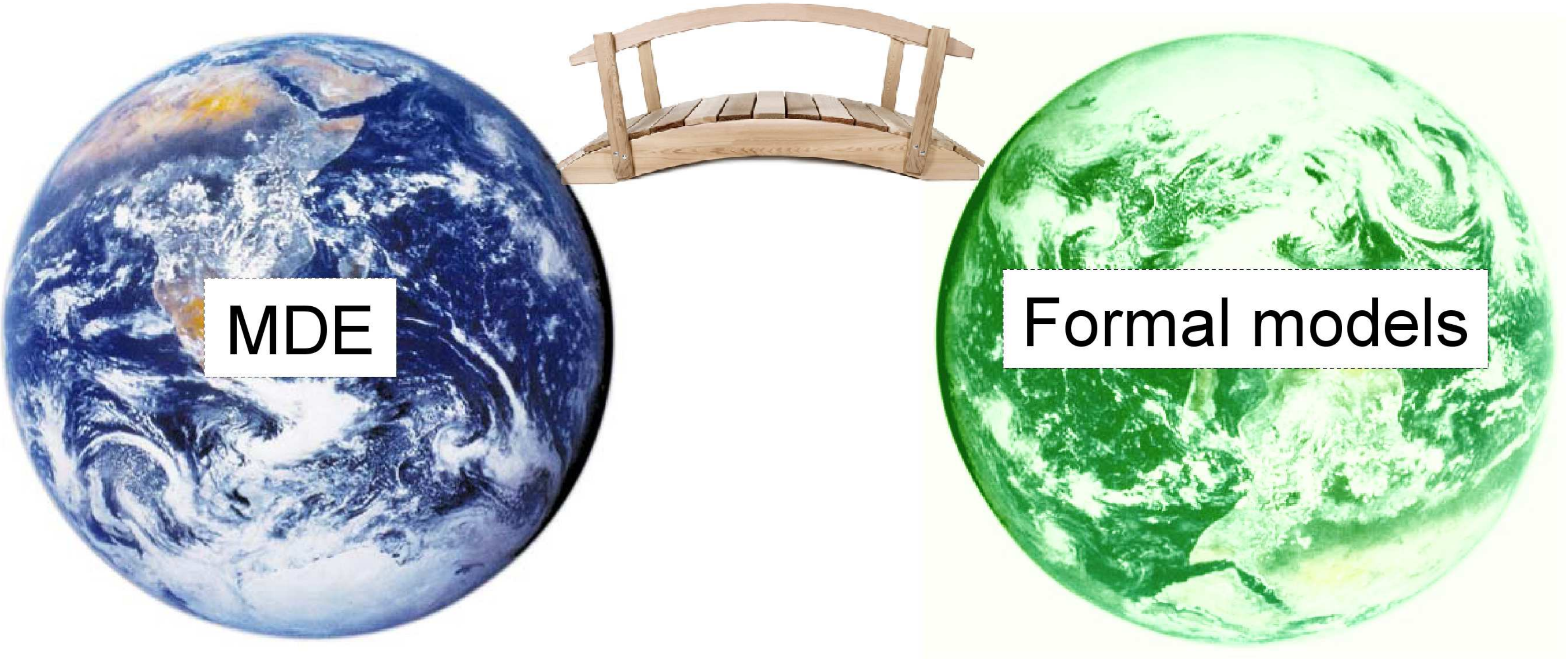


- **Goals**
  - Mathematical semantics
  - Powerful analysis and algorithmic methods
  - Optimization / verification
  - Guaranteed equivalence between code and model
  - Basis for well-founded transformations

- **Current Shortcomings**
  - **Distance from current mainstream engineering practice**
  - **Exotic formalisms for programmers (not C / C++ / java)**
  - **Need for a mathematical background**
  - **Most tools half-academic or confidential**

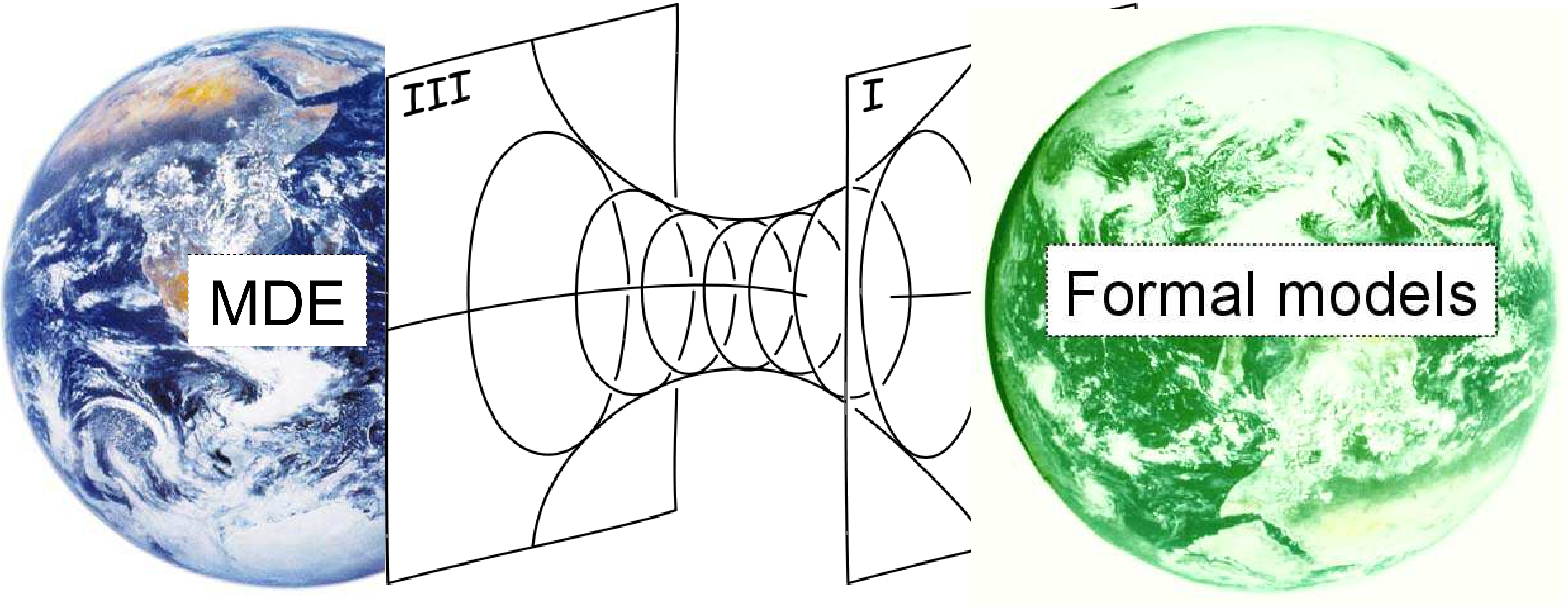
➡ **Poor integration into designflow**

So ?



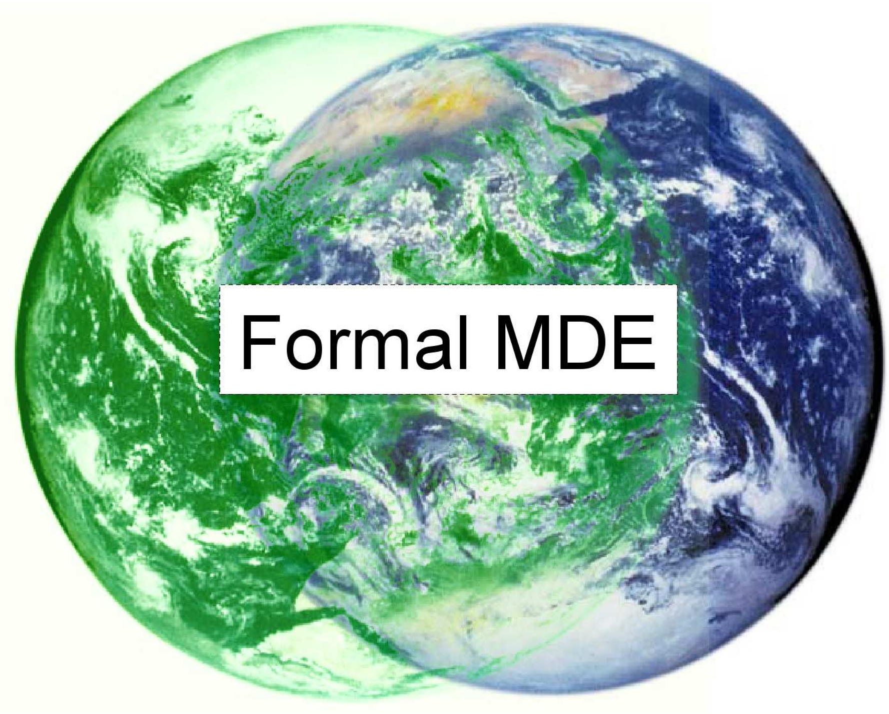
Transformations are often used from a world to another one...

So ?

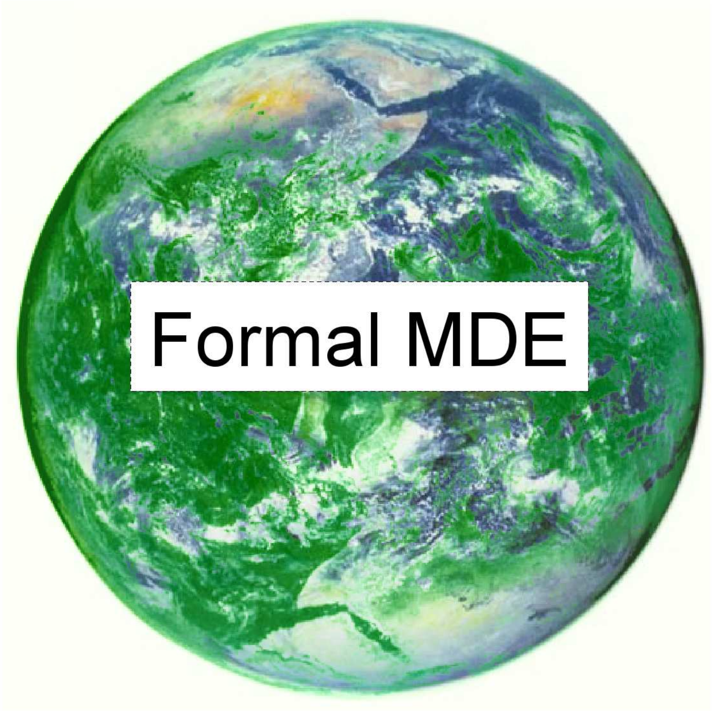


Transformations distort models and lead to hard understanding and round-trip are alsmost impossible

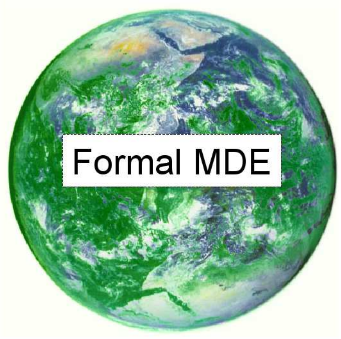
The finality ?



The finality !



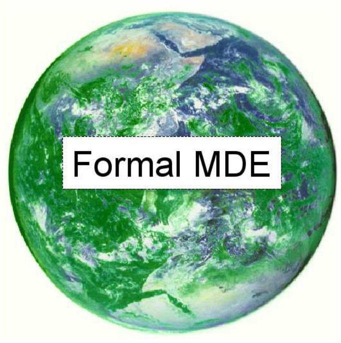
Time @ Design Time



- Expliciting the MoCC within a MDE environment
  - Adding explicit activation conditions
  - Adding relations between these activation conditions

→ formal semantics explicit within the model
(not hidden in the simulator / any transformation)

Time @ Design Time

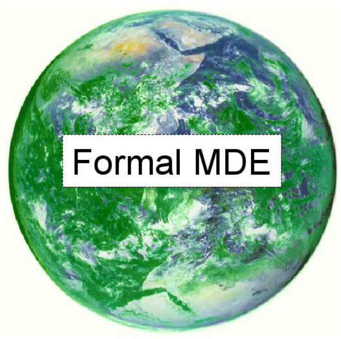


- Expliciting the MoCC within a MDE environment
  - Adding explicit activation conditions
  - Adding relations between these activation conditions

→ formal semantics explicit within the model
(not hidden in the simulator / any transformation)

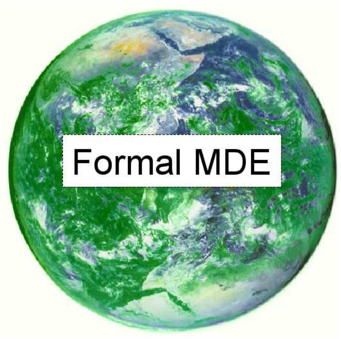
→ *By using multiform logical time*

Time @ Design Time



- MDE is good for the structural concern
- Logical Time is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time

Time @ Design Time



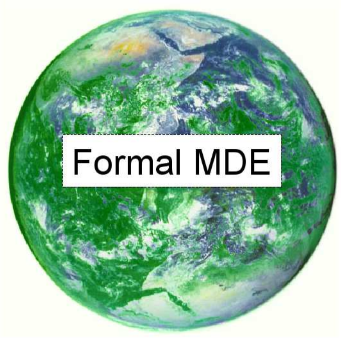
- MDE is good for the structural concern
- Logical Time is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time

→ *The MARTE time model is a first step toward that*

UML profile

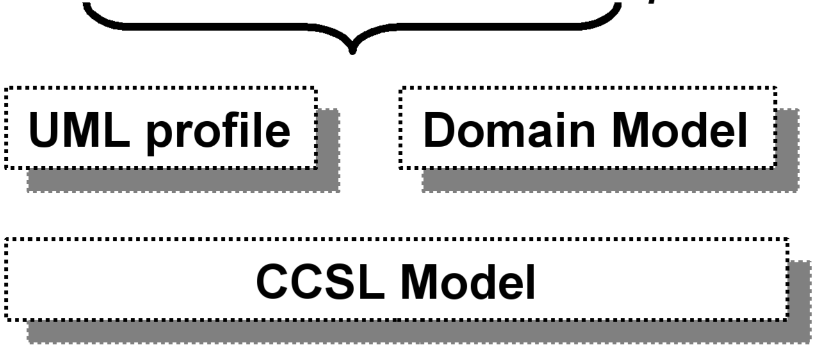
Domain Model

Time @ Design Time



- MDE is good for the structural concern
- Logical Time is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time

→ *The MARTE time model specifies logical activation conditions*


UML profile

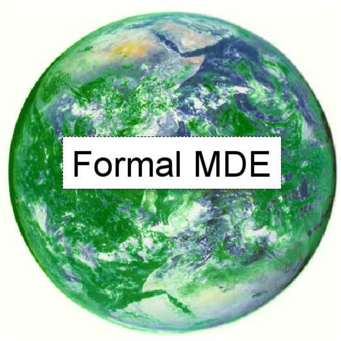
Domain Model

CCSL Model

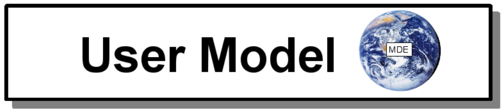
→ *CCSL specifies relations between MARTE activation conditions*

and optionnaly specifies link between logical and physical time

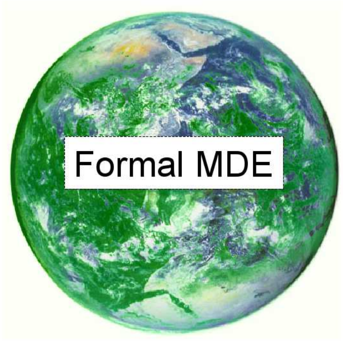
Time @ Design Time



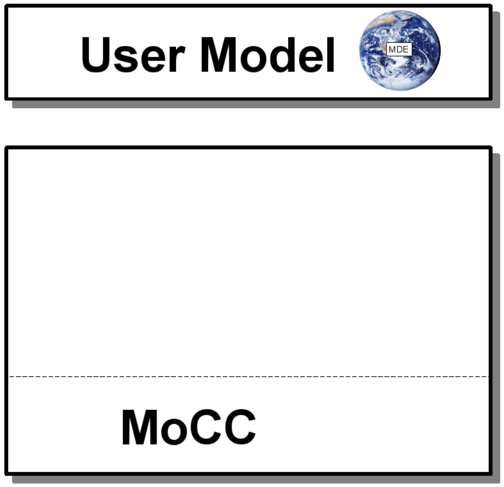
- MDE is good for the structural concern
- Logical Time is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time


User Model

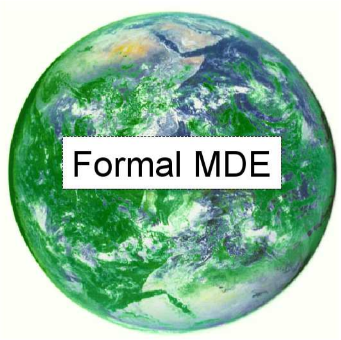
Time @ Design Time



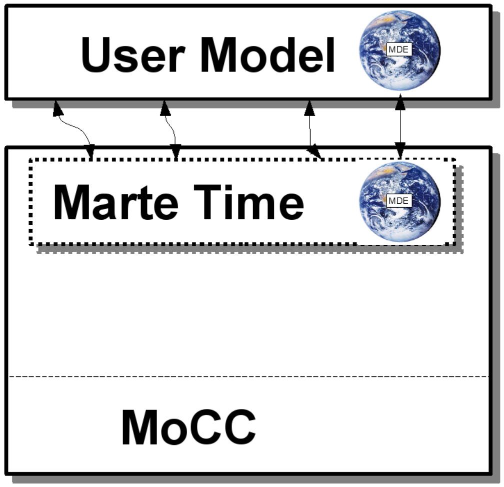
- MDE is good for the structural concern
- Logical Time MoCC is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time



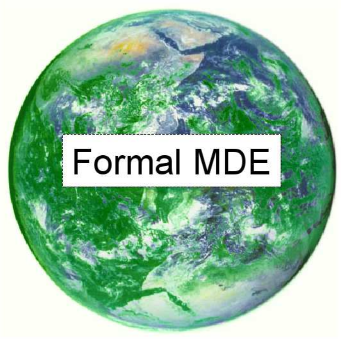
Time @ Design Time



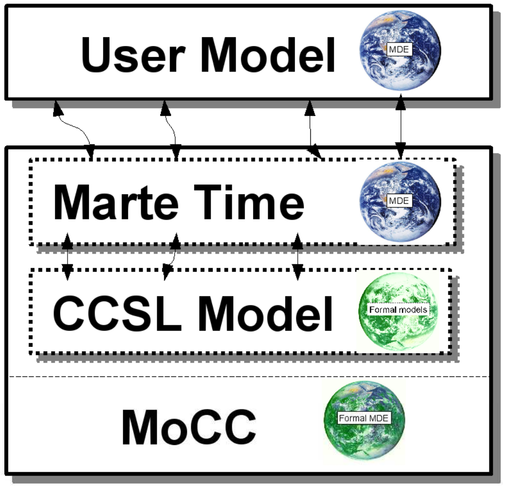
- MDE is good for the structural concern
- Logical Time is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time



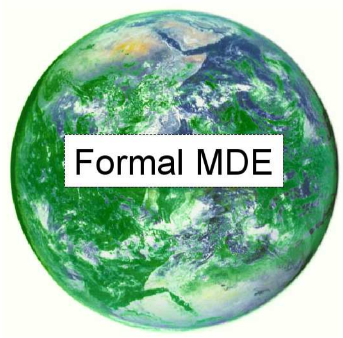
Time @ Design Time



- MDE is good for the structural concern
- Logical Time is good for the dynamic concern
  - Adding explicit activation conditions
  - Adding relations between these activation conditions
  - Optionnaly linking logical time to physical time



MARTE TIME MODEL

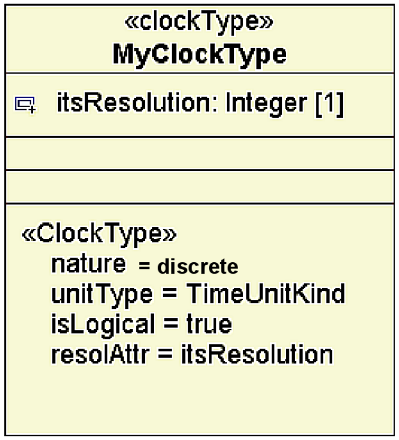


- SubProfile of the MARTE UML profile standardized by the OMG (Object Management Group)
  - Reviewed and accepted by the community
  - Implemented in Papyrus (an UML tool integrated with Eclipse)
  - Under Implementation in other UML tools

- A Domain Model integrated with eclipse and usable with Domain Specific Language

MARTE TIME MODEL

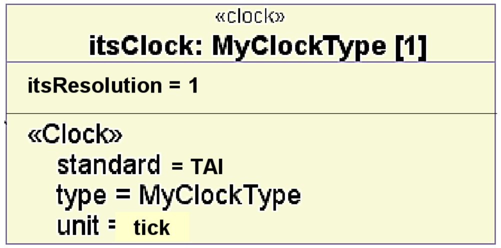
- The main concepts is the **Clock**.
  - It is a way to specify a, possibly infinite, ordered set of instant
  - It can be logical or chronometric, discrete or dense
  - Its type is a ClockType



«clockType»
MyClockType

itsResolution: Integer [1]

«ClockType»
nature = discrete
unitType = TimeUnitKind
isLogical = true
resolAttr = itsResolution



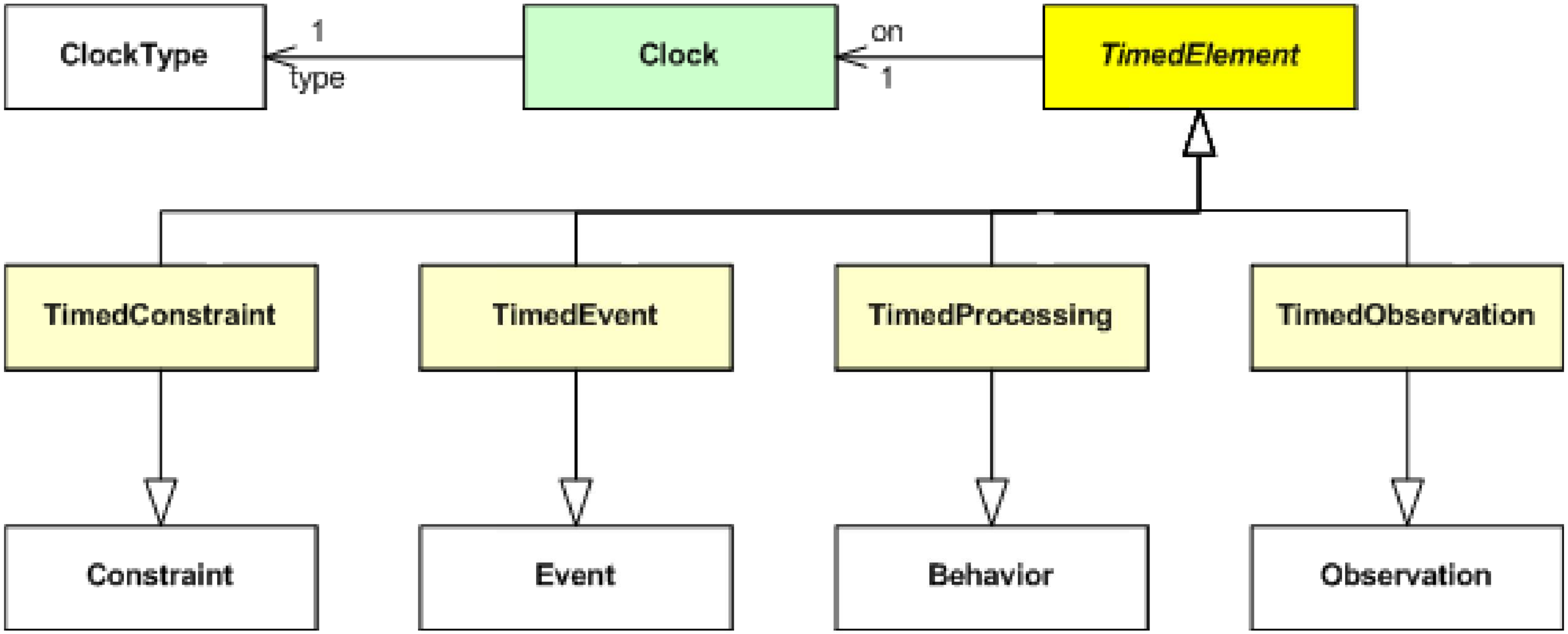
«clock»
itsClock: MyClockType [1]

itsResolution = 1

«Clock»
standard = TAI
type = MyClockType
unit = tick

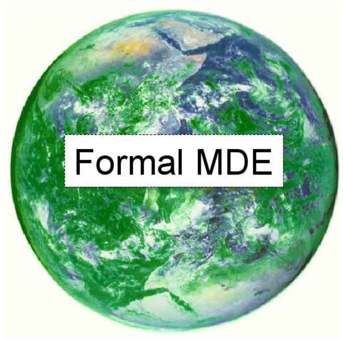
MARTE TIME MODEL

Formal MDE



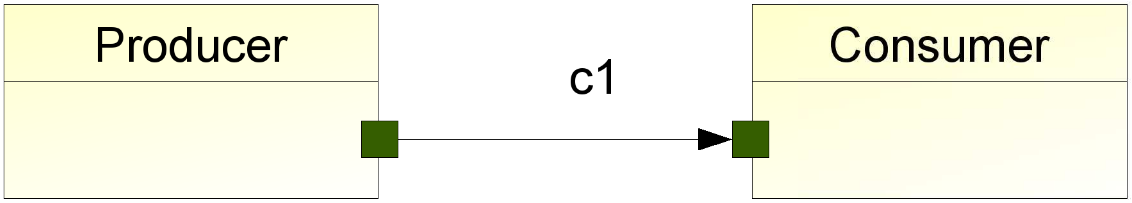
Simplified view of the MARTE Time meta-Model

MARTE TIME MODEL



- Sketchy example of its use

User model

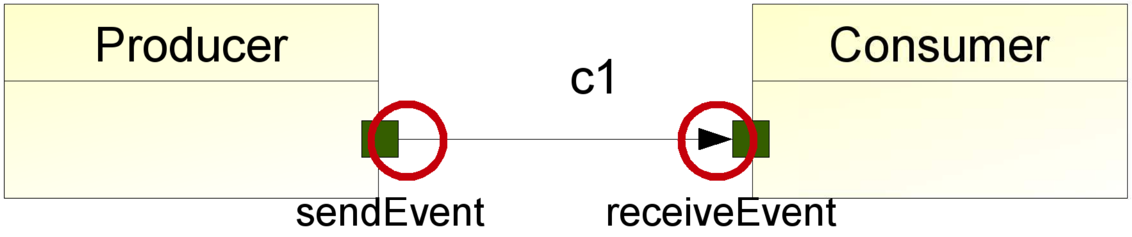


MARTE TIME MODEL

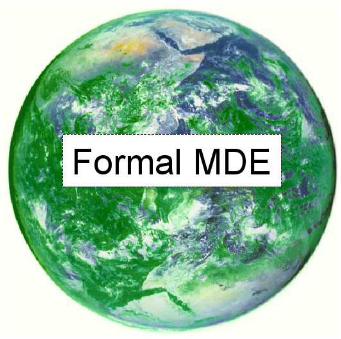
Formal MDE

- Sketchy example of its use

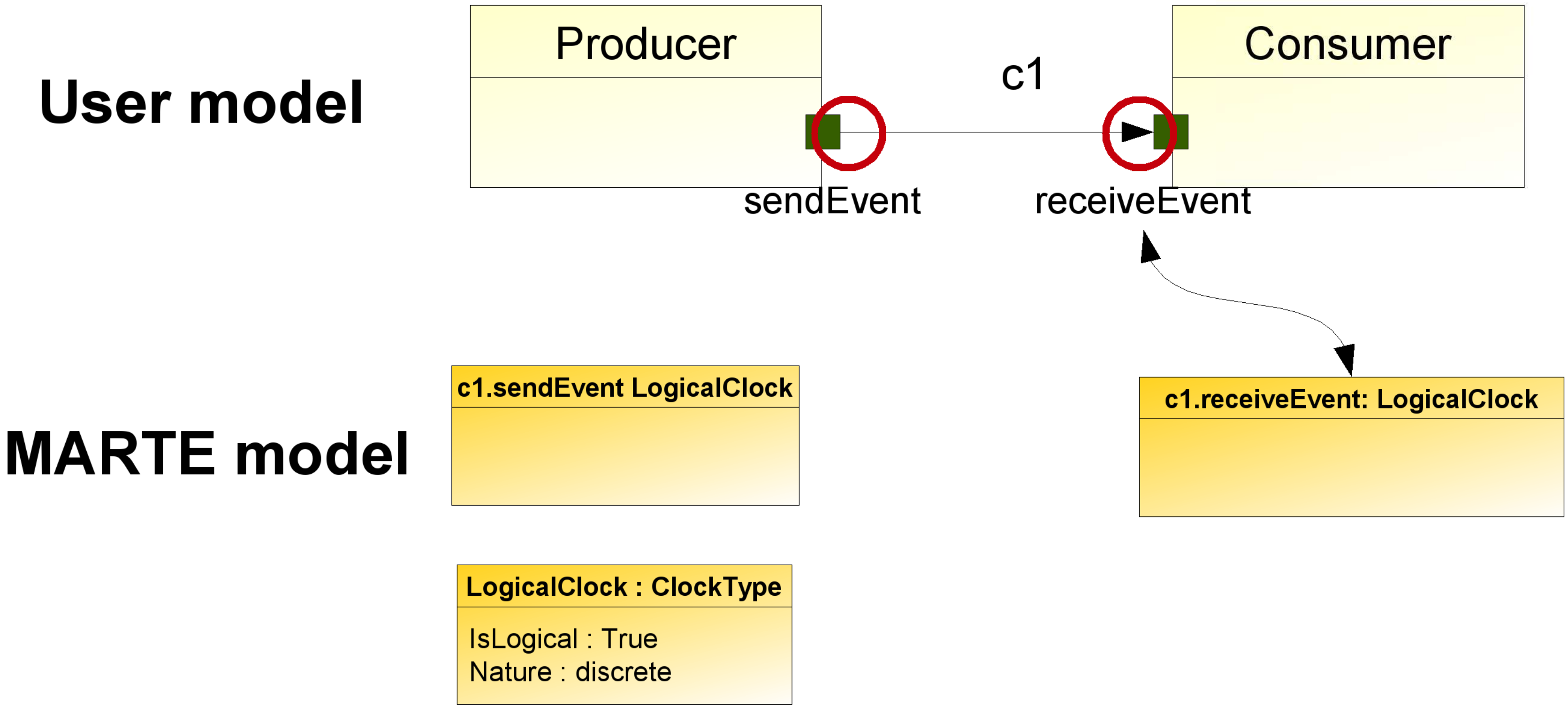
User model



MARTE TIME MODEL



- Sketchy example of its use



User model

Producer

c1

Consumer

sendEvent

receiveEvent

MARTE model

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

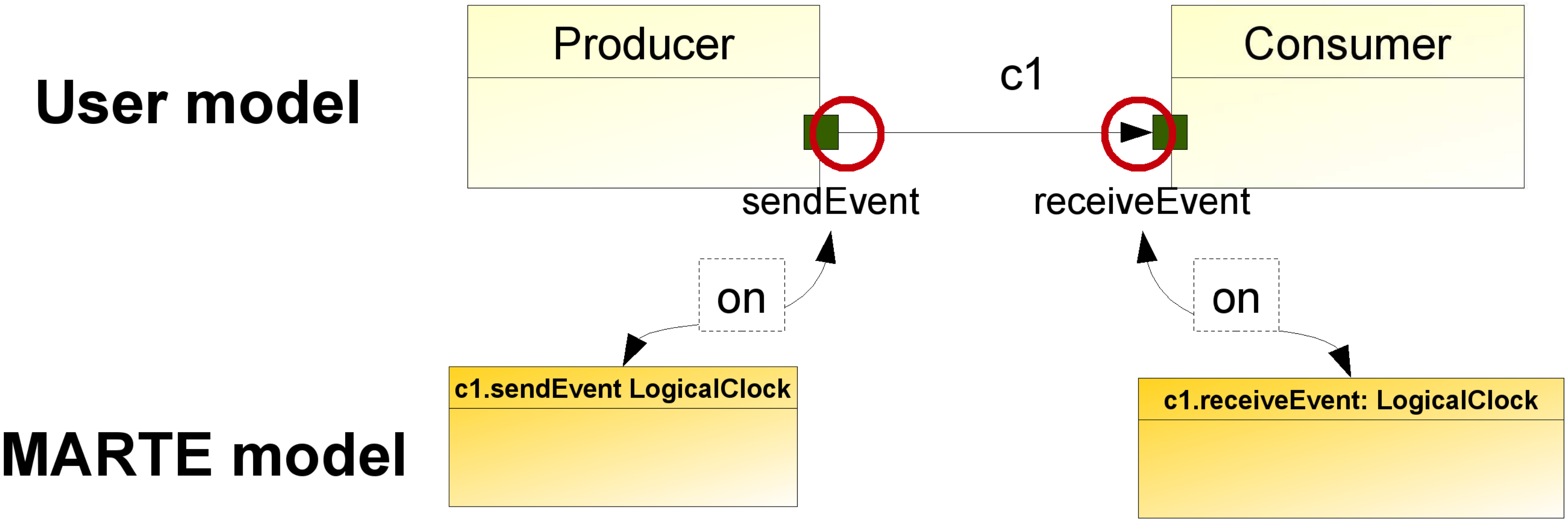
LogicalClock : ClockType

IsLogical : True
Nature : discrete

MARTE TIME MODEL

Formal MDE

- Sketchy example of its use

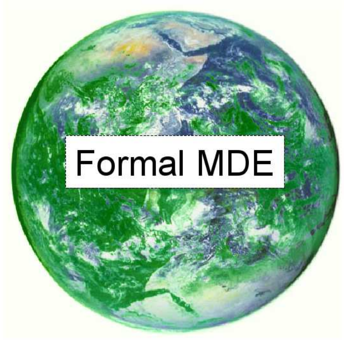


LogicalClock : ClockType

IsLogical : True
Nature : discrete

The ordered set of `sendEvent` *is bijective with the ordered set of instants of* `c1.sendEvent`

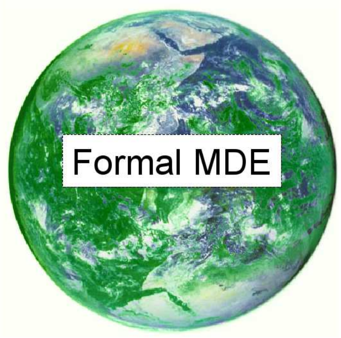
CCSL



- **C**lock **C**onstraint **S**pecification **L**anguage
  - Firstly introduced in the MARTE TIME profile
  - Declarative model-based language integrated with Eclipse
  - Formal semantics (both denotational and operational)
  - Tooled (TimeSquare)

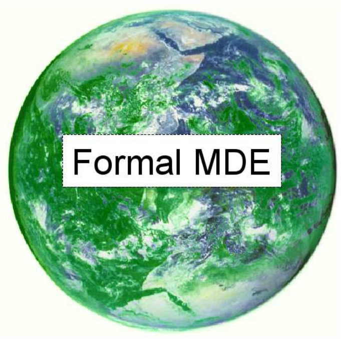
→ **Explicitly represent / specify relations between clocks**

CCSL



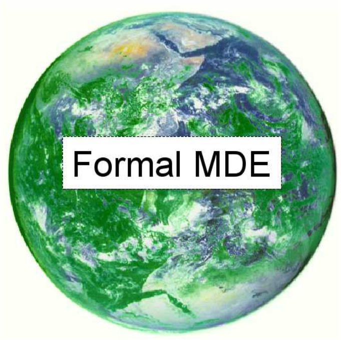
- **C**lock **C**onstraint **S**pecification **L**anguage
  - Relations: dependencies between clocks
    - Coincidence → **=**
    - Exclusion → **#**
    - Precedence → **<**
    - Alternance → **~**
  - Expressions: a mean to create new clocks from others
    - Delay → **delayedFor** *X* **on** *aClock*
    - Filtering → *aClock* **filteredBy** *aBinaryWord*
    - Union → *aClock* **union** *anotherClock*
    - Intersection → *aClock* **inter** *anotherClock*
    - Periodicity → **periodicOn** aClock **period** X **offset** Y
    - …

CCSL



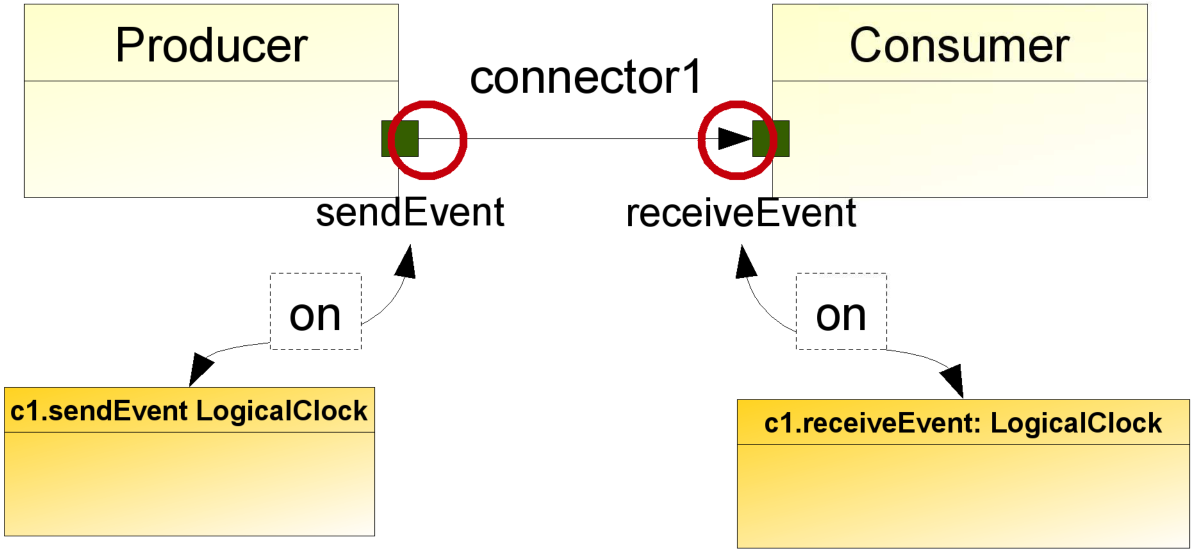
- **C**lock **C**onstraint **S**pecification **L**anguage
  - Relations: dependencies between clocks
    - Coincidence → **=**
    - Exclusion → **#**
    - Precedence → **<**
    - Alternance → **~**
  - Expressions: a mean to create new clocks from others
    - Delay → **delayedFor** *X* **on** *aClock*
    - Filtering → *aClock* **filteredBy** *aBinaryWord*
    - Union → *aClock* **union** *anotherClock*
    - Intersection → *aClock* **inter** *anotherClock*
    - Periodicity → **periodicOn** aClock **period** X **offset** Y
    - …
  - Libraries: user-defined relations and expressions

CCSL



- Complete the sketchy example

User model


Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

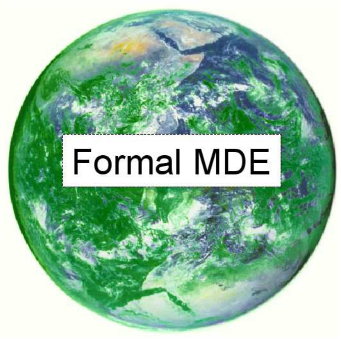
MARTE model

LogicalClock : ClockType

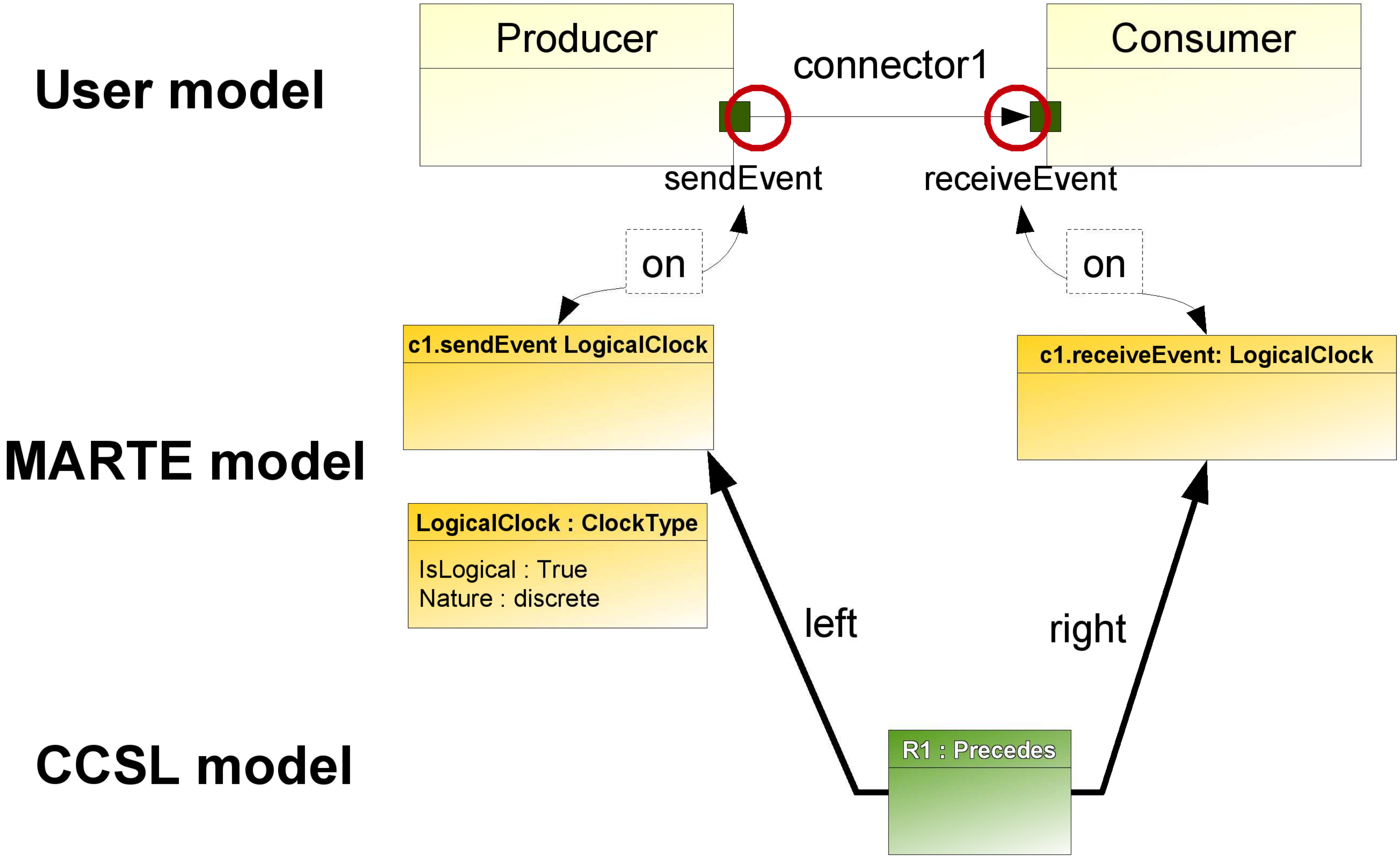
IsLogical : True
Nature : discrete

The ordered set of `sendEvent` *is bijective with the ordered set of instants of* `c1.sendEvent`

CCSL



- Complete the sketchy example



User model

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

MARTE model

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

LogicalClock : ClockType

IsLogical : True
Nature : discrete

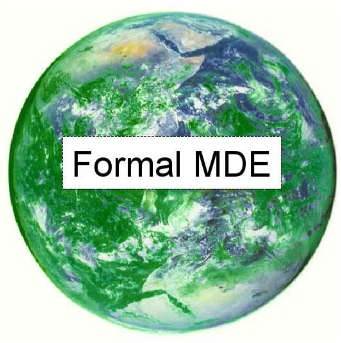
left

right

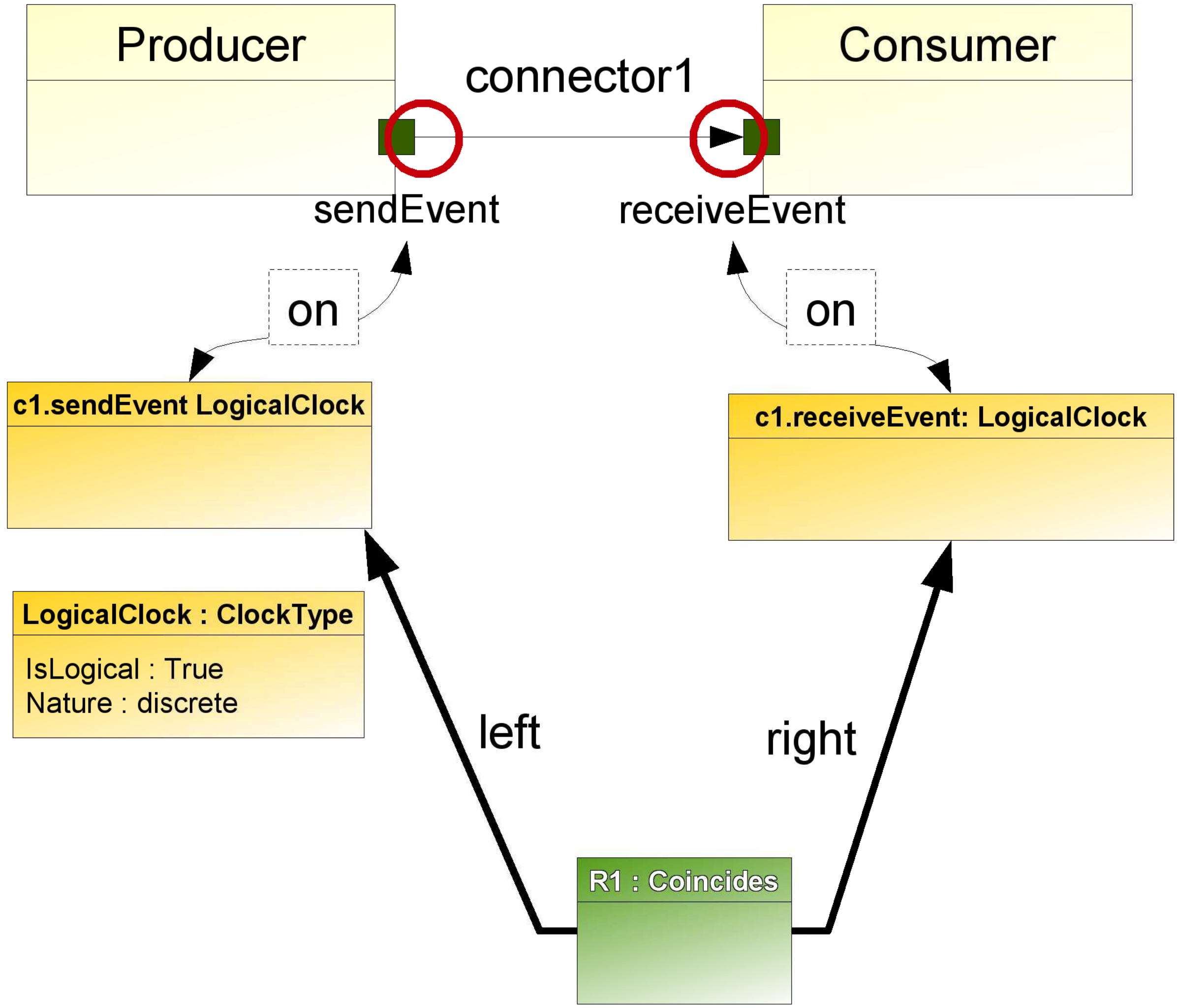
CCSL model

R1 : Precedes

CCSL



- Complete the sketchy example



User model

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

MARTE model

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

LogicalClock : ClockType

IsLogical : True
Nature : discrete

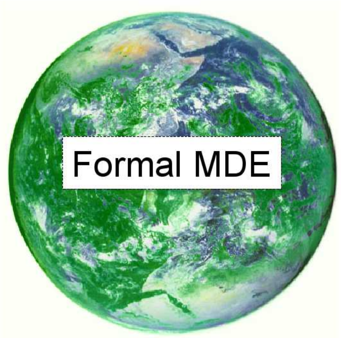
left

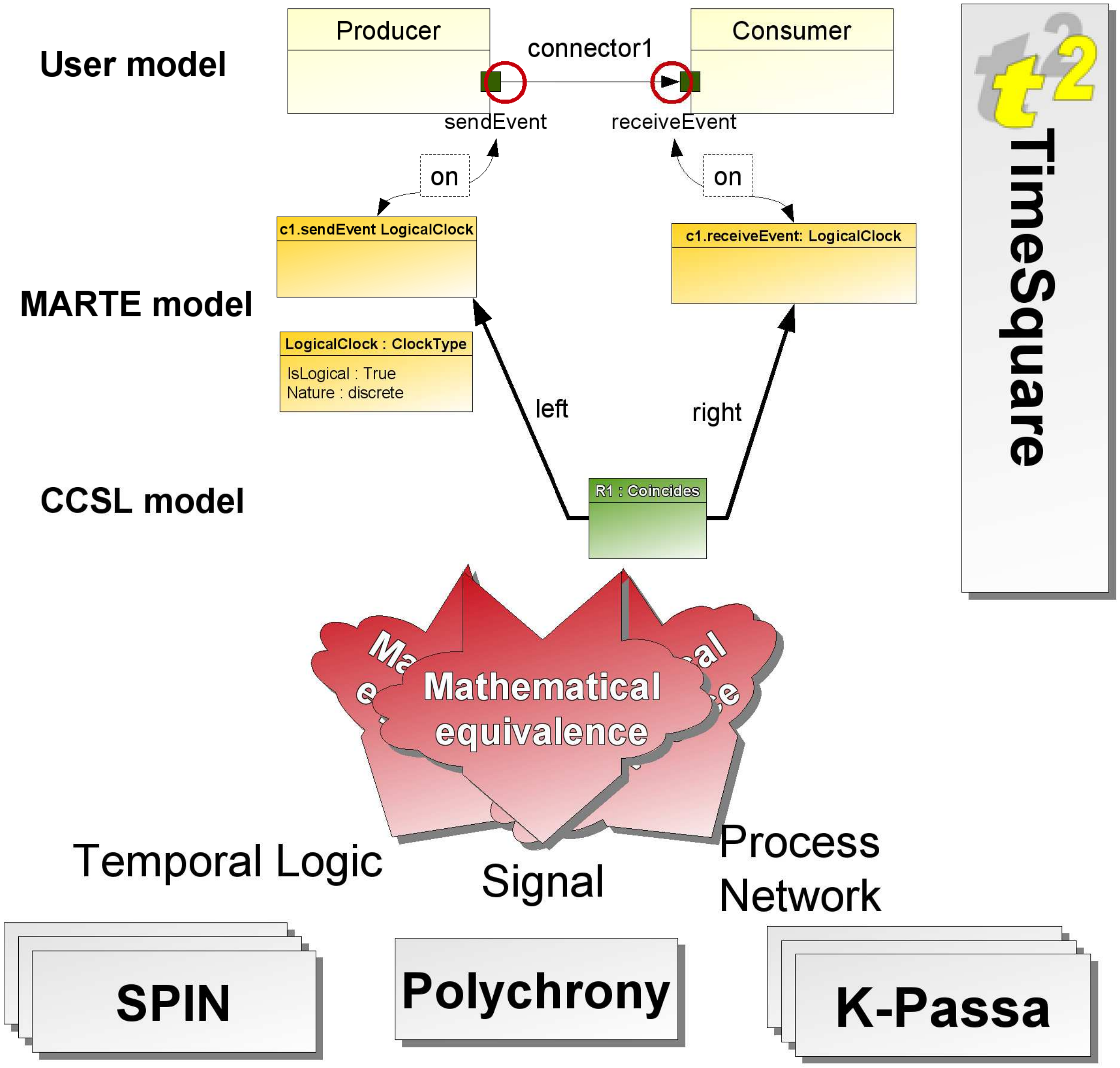
right

CCSL model

R1 : Coincides

Benefits





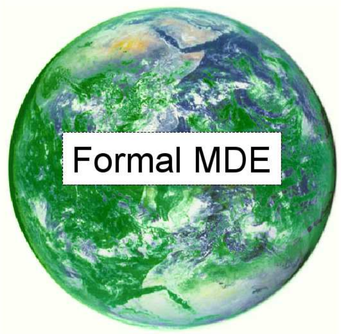
Simulation

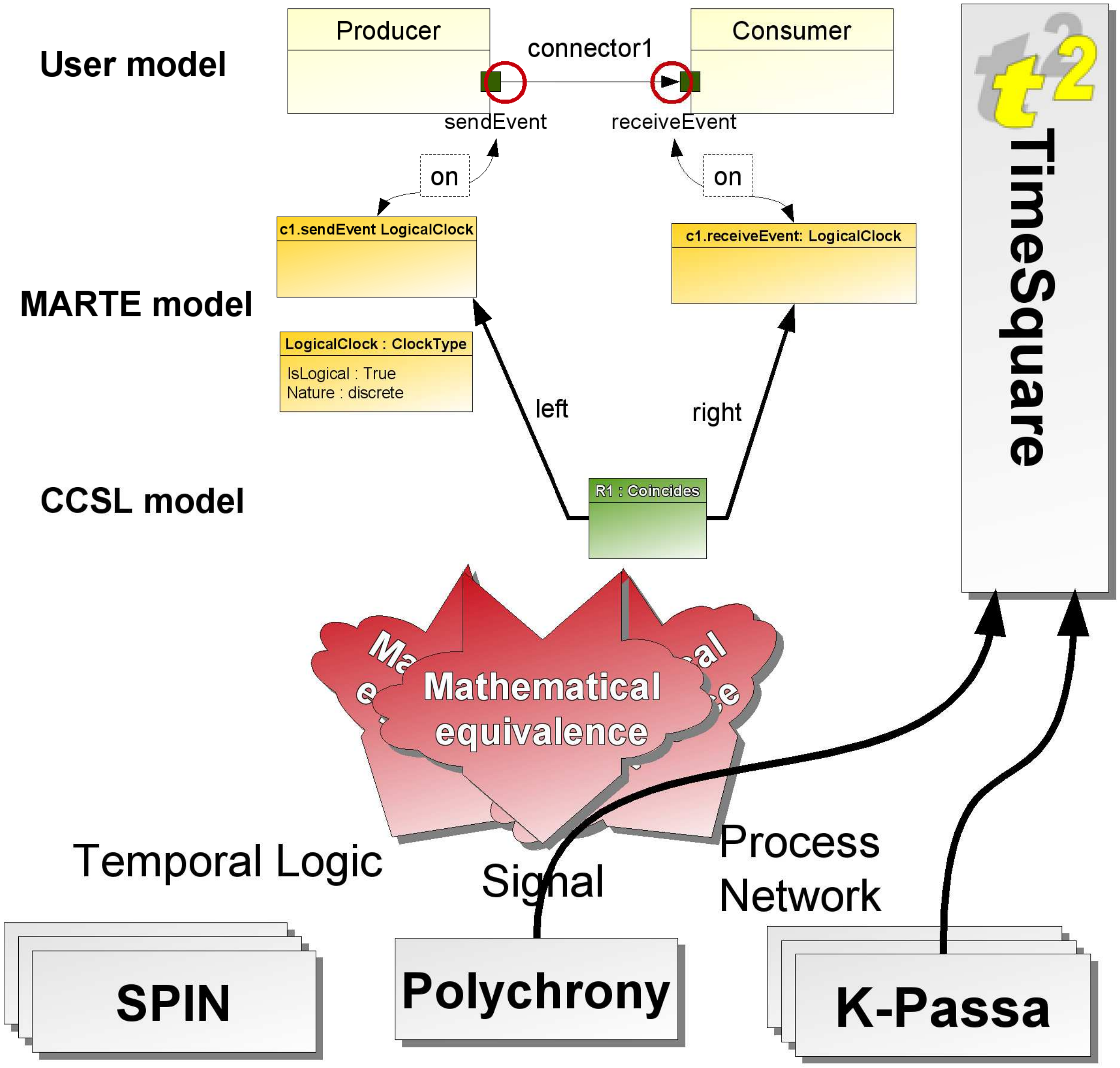
- Model animation
- Timing Diagram
- Sequence Diagram
- User Defined action

State space exploration

- Still Beta

Benefits





Simulation

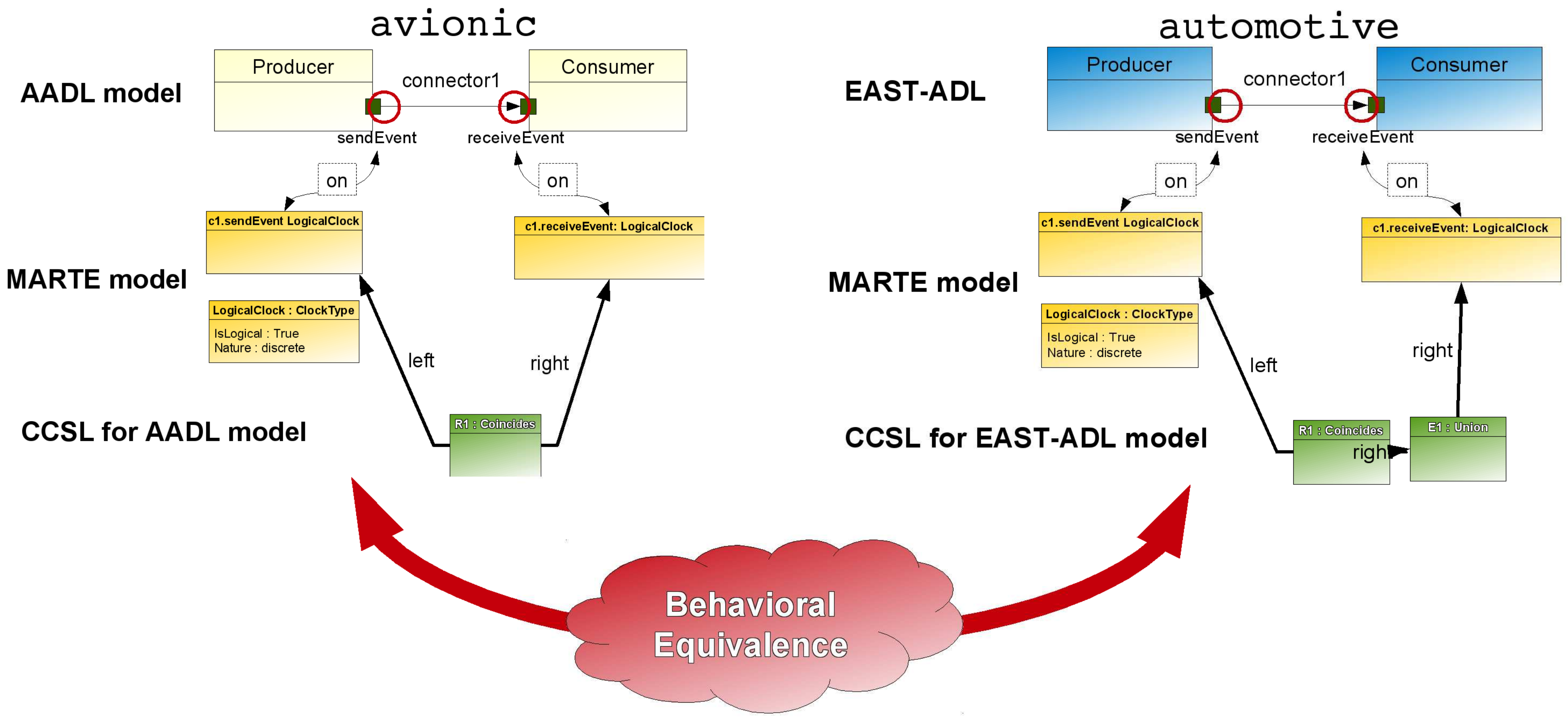
- Model animation
- Timing Diagram
- Sequence Diagram
- User Defined action

State space exploration

- Still Beta

Benefits

Formal MDE



avionic

AADL model

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

MARTE model

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

LogicalClock : ClockType

IsLogical : True
Nature : discrete

left

right

CCSL for AADL model

R1 : Coincides

automotive

EAST-ADL

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

MARTE model

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

LogicalClock : ClockType

IsLogical : True
Nature : discrete

left

right

CCSL for EAST-ADL model

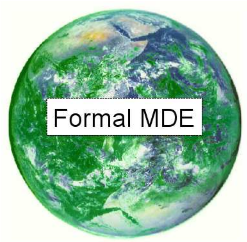
R1 : Coincides

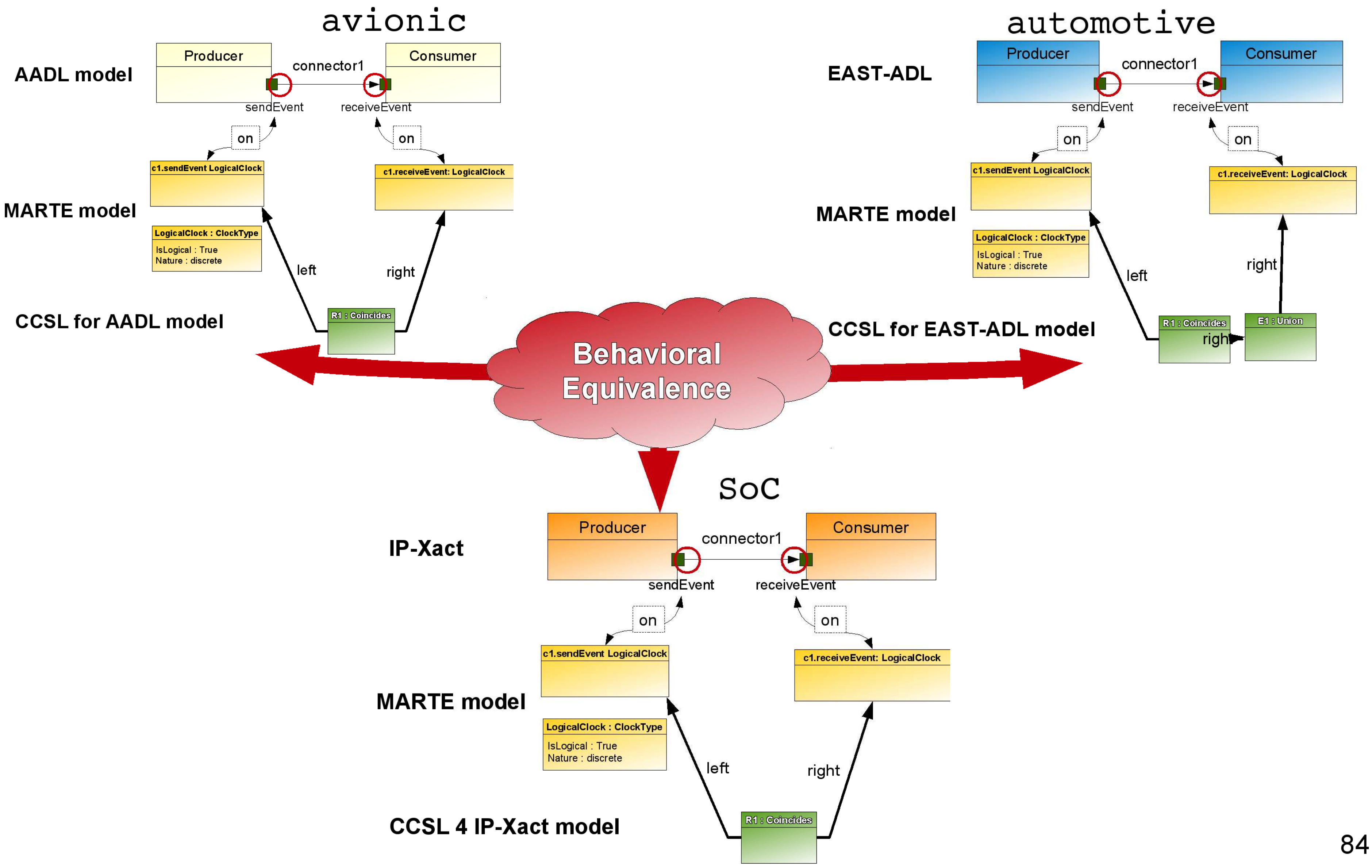
right

E1 : Union

Behavioral Equivalence

Benefits





avionic

AADL model

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

MARTE model

LogicalClock : ClockType

IsLogical : True
Nature : discrete

left

right

CCSL for AADL model

R1 : Coincides

automotive

EAST-ADL

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

MARTE model

LogicalClock : ClockType

IsLogical : True
Nature : discrete

left

right

CCSL for EAST-ADL model

R1 : Coincides

right

E1 : Union

Behavioral Equivalence

SoC

IP-Xact

Producer

connector1

Consumer

sendEvent

receiveEvent

on

on

c1.sendEvent LogicalClock

c1.receiveEvent: LogicalClock

MARTE model

LogicalClock : ClockType

IsLogical : True
Nature : discrete

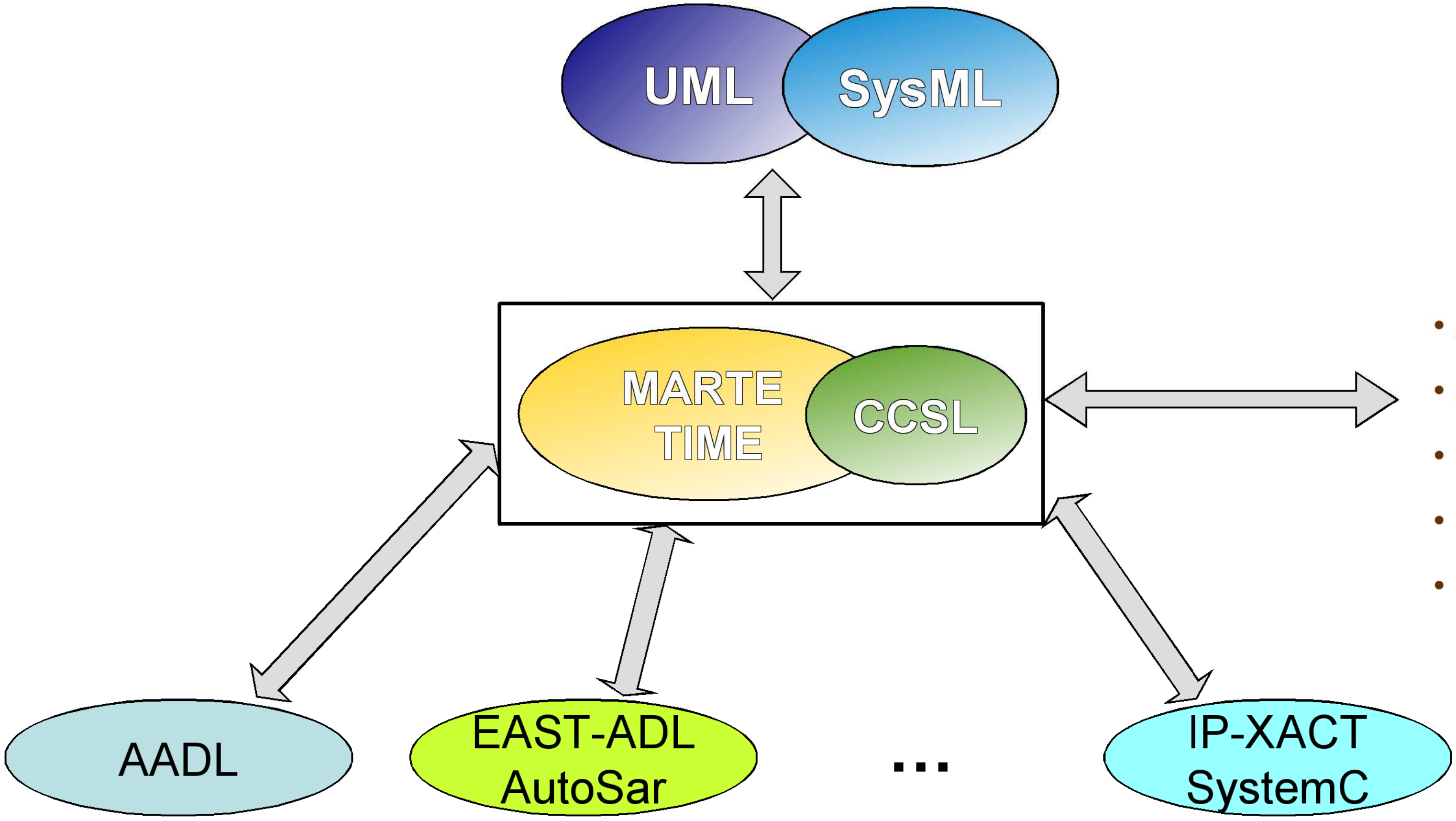
left

right

CCSL 4 IP-Xact model

R1 : Coincides

Benefits



- ***PolyChrony/SME***
- ***SynDEx***
- ***K-Passa***
- ***Gaspard2***
- ***…***

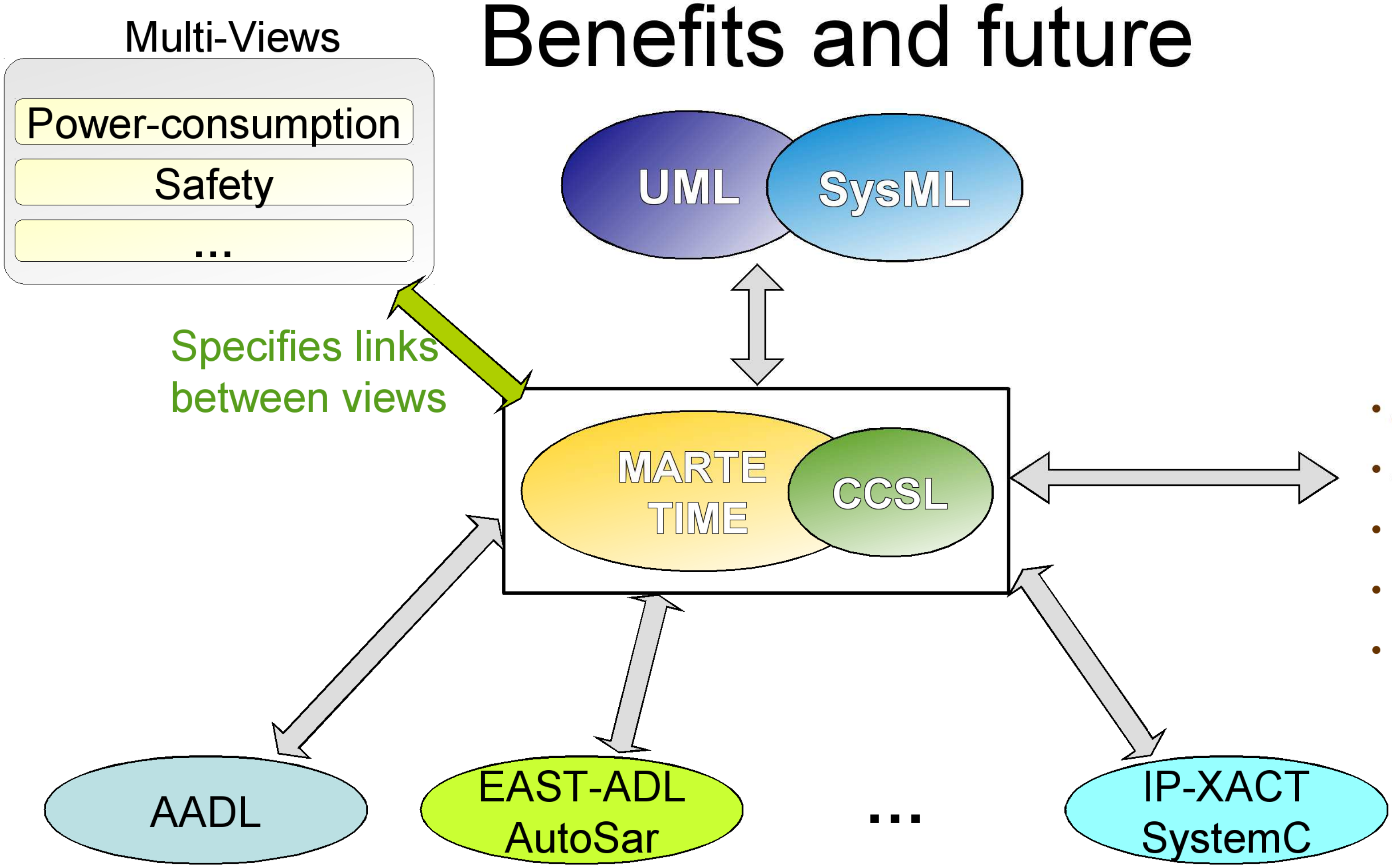
- ***Meta-H toolsuite***
- ***OSATE***
- ***Cheddar***
- ***…***

- ***Symta/S***
- ***RT@W***
- ***…***

- ***CoWare***
- ***Virtio/Innovator***
- ***VaST***
- ***…***

Formal MDE

Benefits and future



Multi-Views

Power-consumption

Safety

...

Specifies links between views

UML

SysML

MARTE TIME

CCSL

AADL

EAST-ADL AutoSar

...

IP-XACT SystemC

- *PolyChrony/SME*
- *SynDEx*
- *K-Passa*
- *Gaspard2*
- *…*

- *Meta-H toolsuite*
- *OSATE*
- *Cheddar*
- *…*

- *Symta/S*
- *RT@W*
- *…*

- *CoWare*
- *Virtio/Innovator*
- *VaST*
- *…*

Current Projects

- **ARTEMIS CESAR [01/09 – 12/11]**
  - 52 partners : CEA, Airbus, Esterel Technology, Thalès, …
  - Requirements engineering: **multi viewpoint**, multi criteria and multi level requirements,
  - Component based engineering: design space exploration, **comprising multi-view**, multi-criteria and multi level architecture trade-offs.
- **ITEA TIMMO2 [?]**
  - Continental, Delphi, Volvo, …
  - Time Model for AUTOSAR/East-ADL2
- **FUI Lambda [07/08 – 06/11]**
  - 14 partners : CEA-List, Thales TRT, Supélec, Airbus EADS, …
  - Convergences MARTE, SysML, AADL, IP-Xact, Scade/SyncCharts
- **ANR RT-Simex [12/08 – 12/11]**
  - CEA-List, Thales TRT, OBEO, UBO, Aonix
  - Retro-ingénerie de Traces d'analyse de SIMulation et d'EXécution de systèmes temps-réel
- **ANR Help [11/09 – 10/12]**
  - Verimag, STMicro Grenoble, Docea Power, LEAT
  - High Level Models for Low Power Systems : IP-Xact et UPF
- **Nano 2012 – ID-TLM [10/08 – 12/10]**
  - ST-MicroElectronics
  - UML/MARTE & IP-Xact: behavioral and timing models for IP-Xact

This is the end...

...thanks...