Eugenio Villar – University of Cantabria

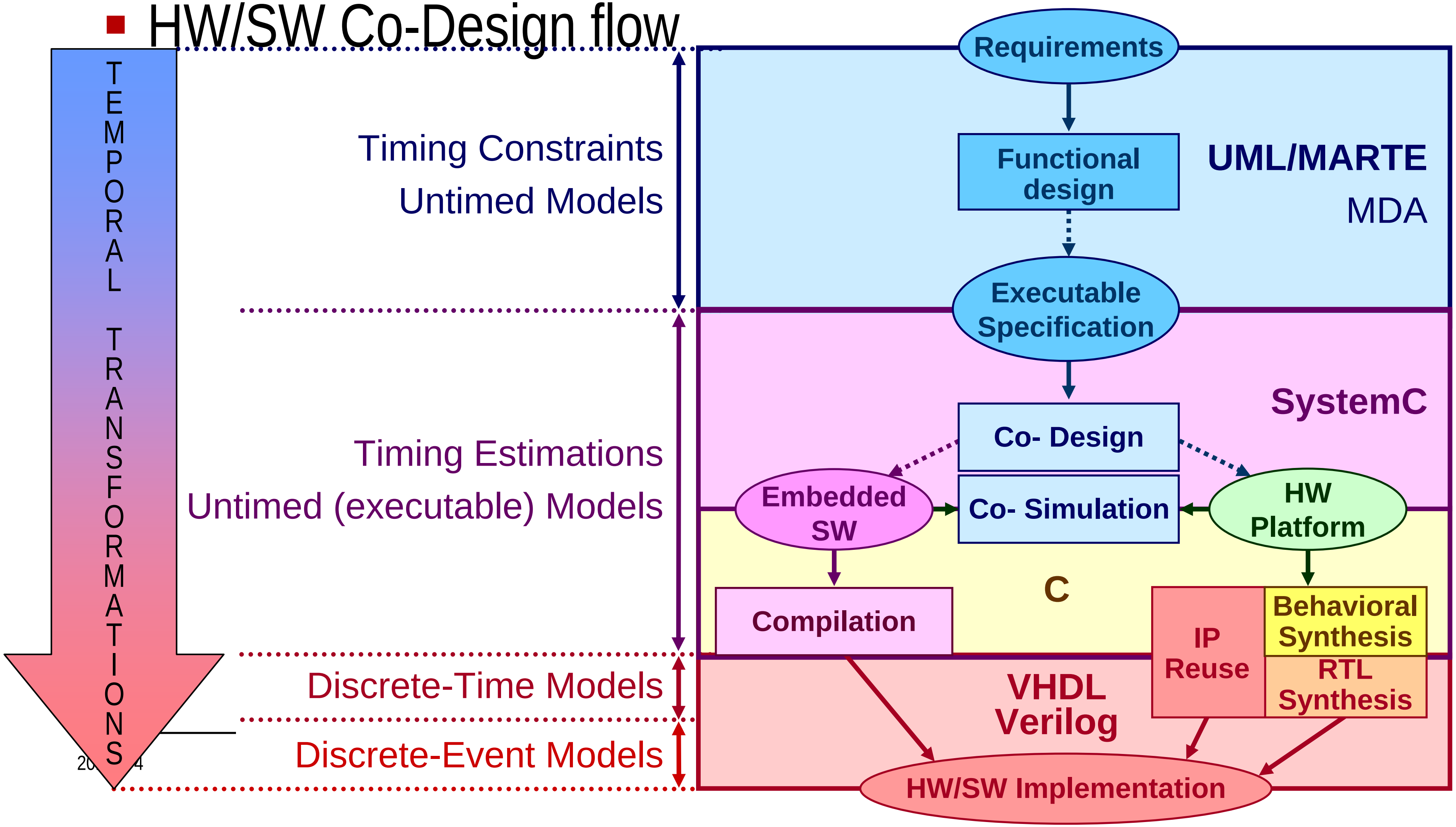
FORMALIZATION OF THE MARTE/SYSTEMC INTEROPERABILITY FOR HW/SW CO-DESIGN

Context

- SystemC
  - Wide use for system modeling
  - Increasing interest for system specification
  - Untimed modeling > 25%
- Increasing interest in MARTE
- Increasing interest in MARTE/SystemC interoperability

Context

- HW/SW Co-Design flow



Agenda

- Motivation
- Formalization of the Saturn design methodology
- MARTE/SystemC interoperability formalization
- Conclusions

Motivation

- Mature HW design process
  - RTL description
  - RTL simulation (Discrete-Time MoC)
  - RTL synthesis
  - Logic simulation (Discrete-Event MoC)
  - Placement & Routing
  - Static Timing Analysis
  - Back-annotated logic simulation (Discrete-Event MoC)

Motivation

- The MPSoC
  - Multi-processing platform
  - With ‘some’ additional HW
  - Most of the functionality implemented as Embedded SW

Motivation

- Software Reliability

WASHINGTON (COMPUTERWORLD) - Software bugs are costing the U.S. economy an estimated $59.5 billion each year, with more than half of the cost borne by end users and the remainder by developers and vendors, according to a new federal study.

Improvements in testing could reduce this cost by about a third, or $22.5 billion, but it won't eliminate all software errors, the study said. Of the total $59.5 billion cost, users incurred 64% of the cost and developers 36%.

- http://www.cse.lehigh.edu/~gtan/bug/softwarebug.html
- http://www.sereferences.com/software-failure-list.php
- http://www5.in.tum.de/~huckle/bugse.html ...

Motivation

- Clear need for sound HW/SW design methods
  - Formal methods in HW/SW System Engineering

- Formalization of the HW/SW Co-Design process
  - Formalization of the MoCs at each level
    - Horizontal heterogeneity
  - Formalization of the transformations among MoCs
    - Vertical heterogeneity

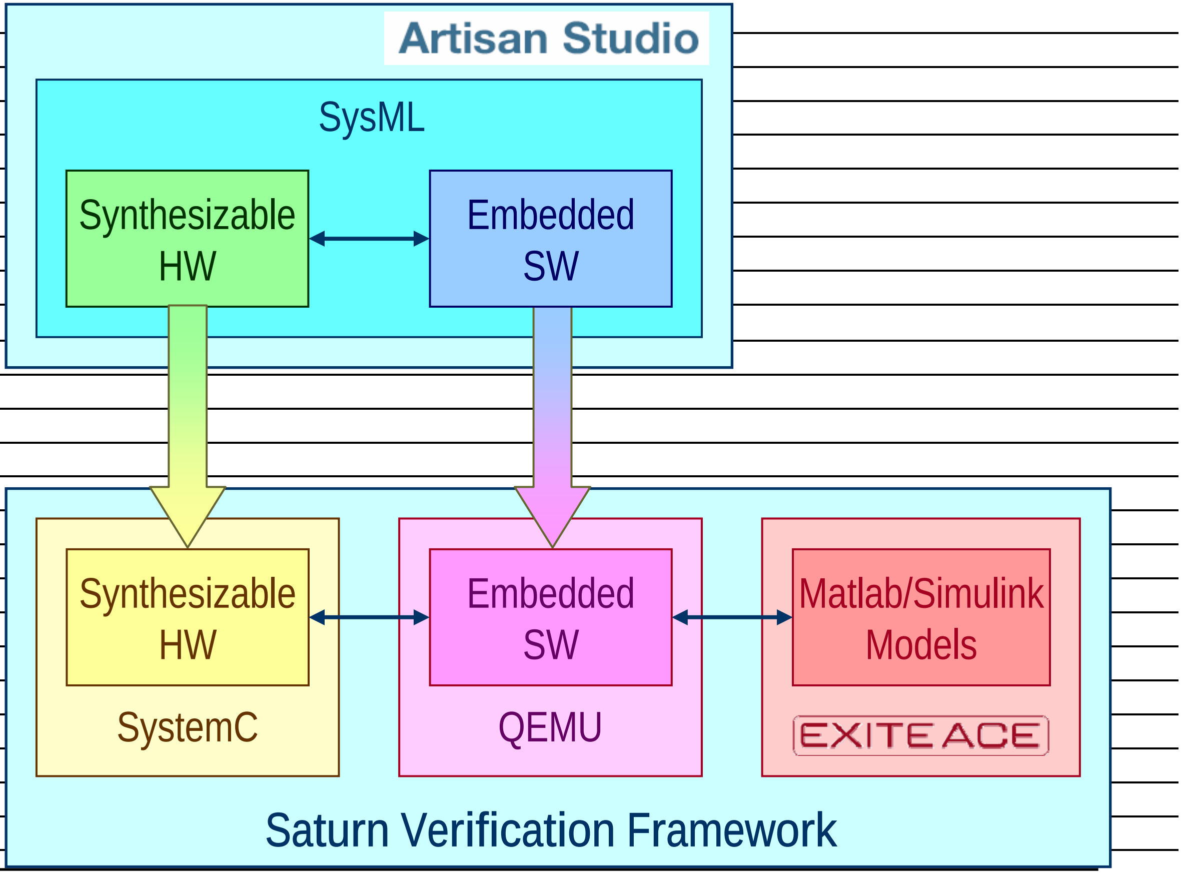
Objective

- Formalization of the MARTE/SystemC interoperability

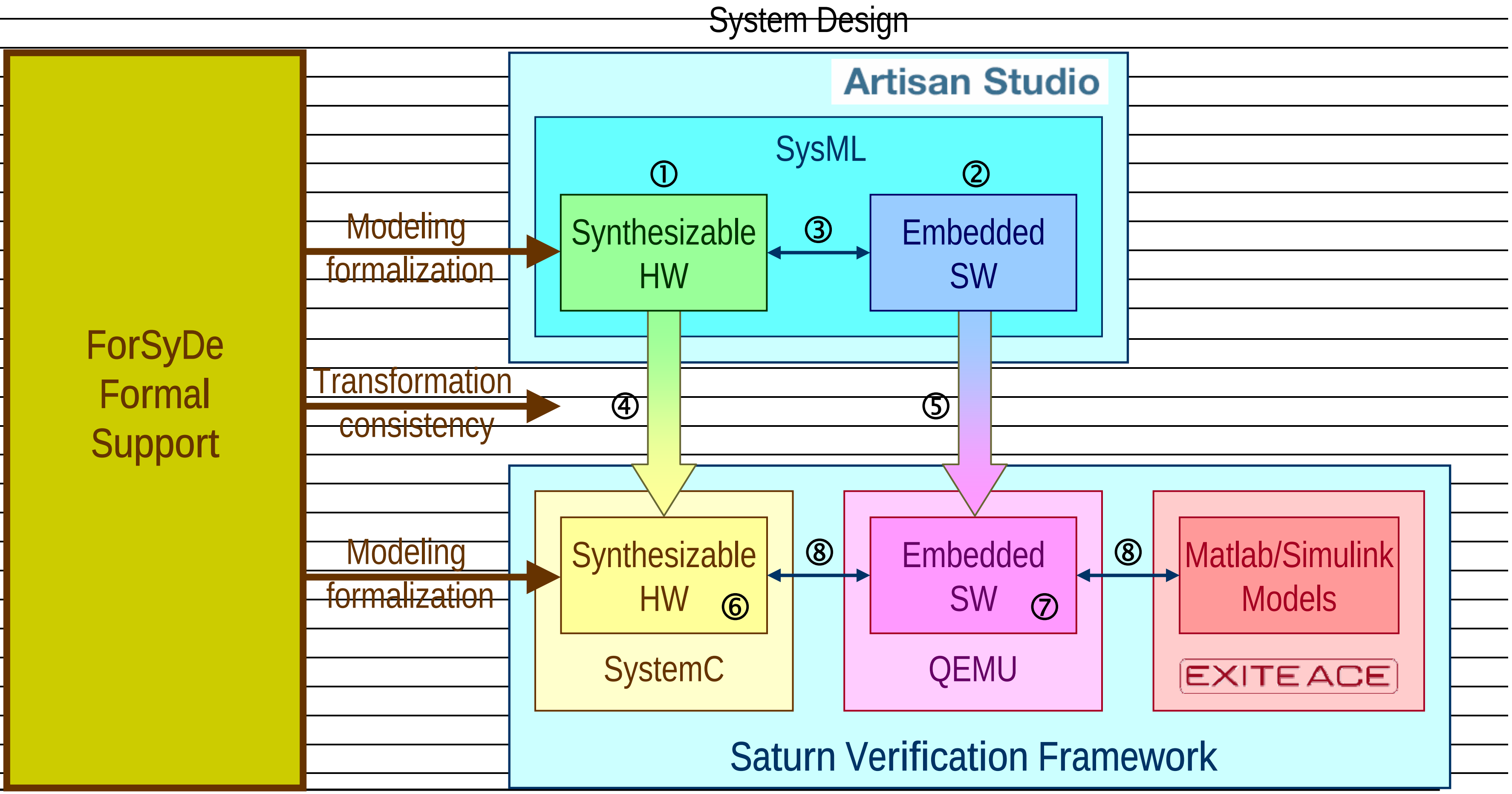
- ForSyDe as formal framework
  - Untimed, Synchronous, Timed and Continuous MoCs
  - Unified modeling
    - Functionality
    - Timing

The Saturn design methodology

System Design

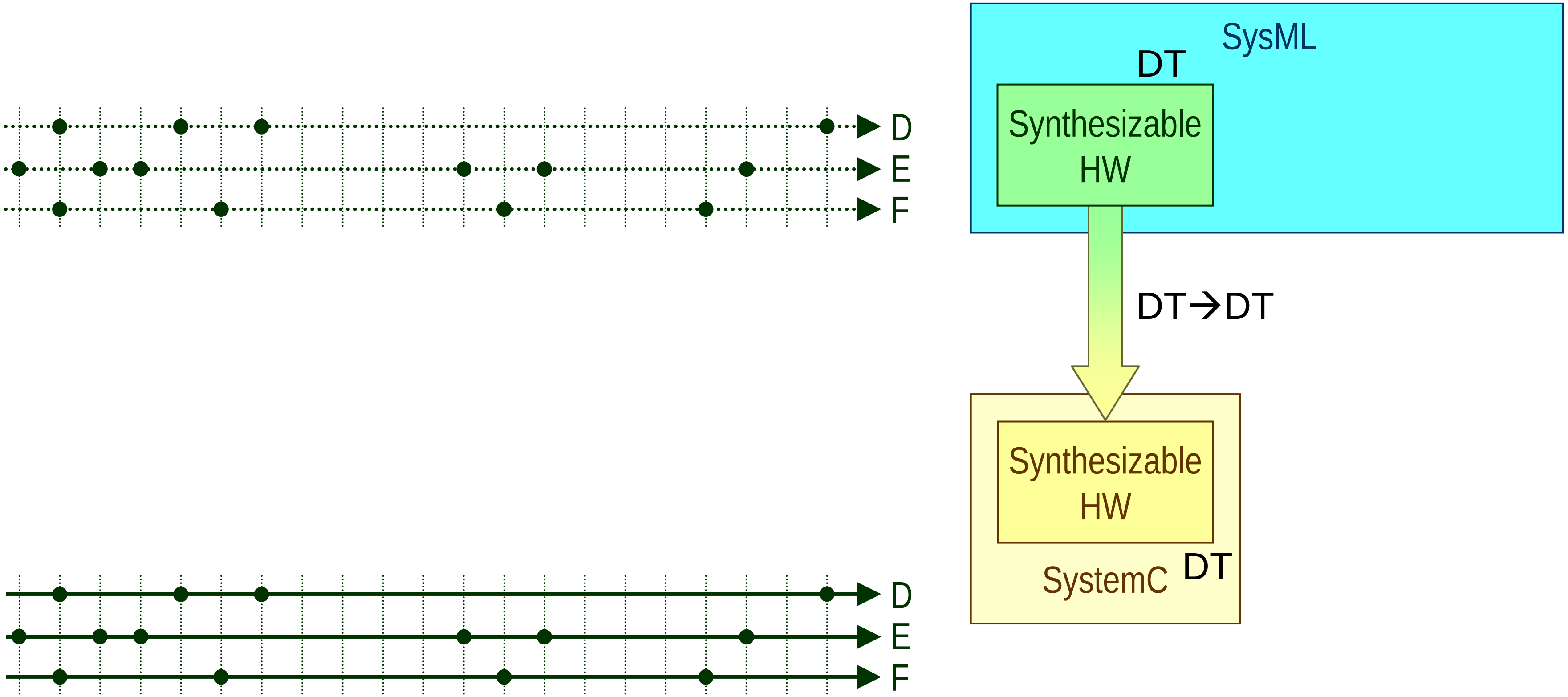


The ForSyDe Formal Support



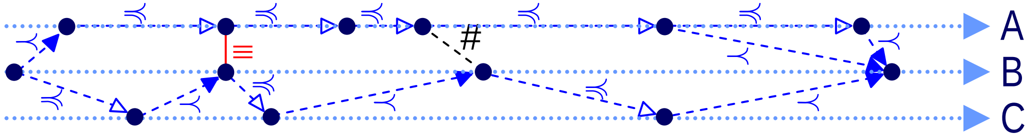
The ForSyDe Formal Support

- SystemC/RTL synthesizable code

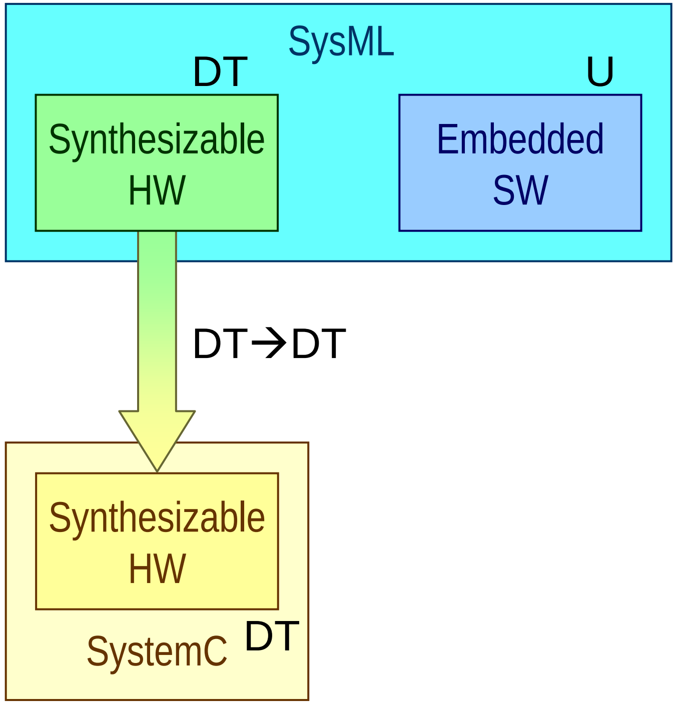


The ForSyDe Formal Support

- SysML SW model

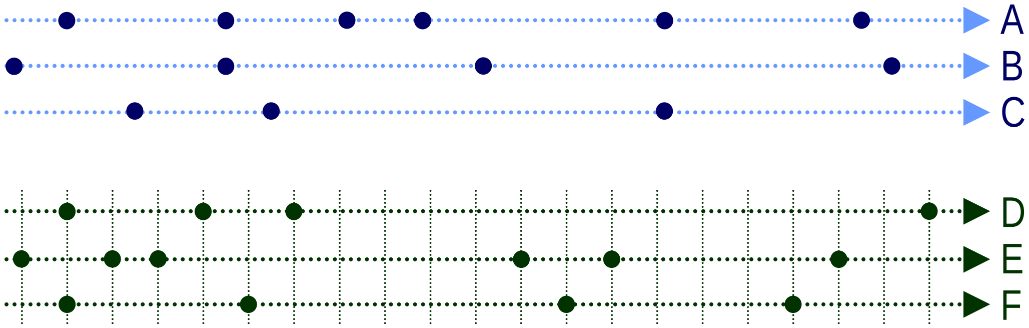


B alternatesWith C



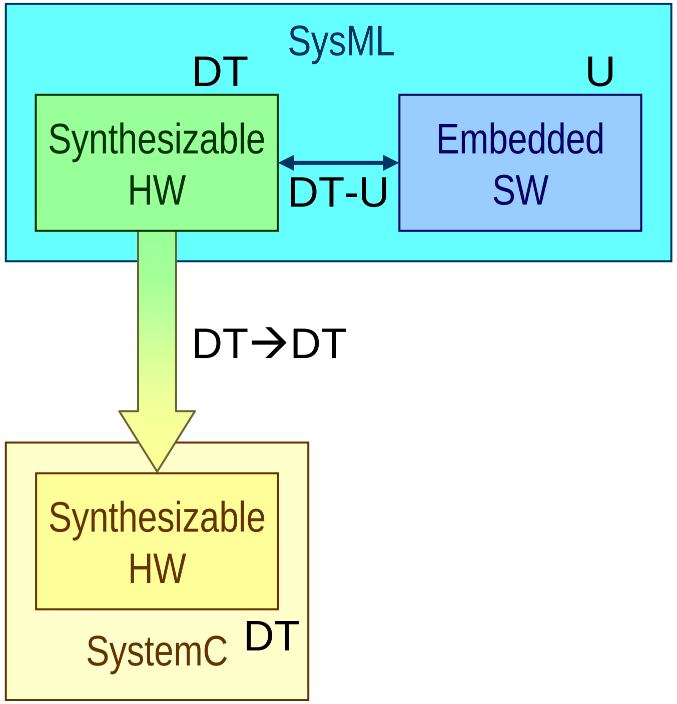
The ForSyDe Formal Support

- SysML HW/SW interaction



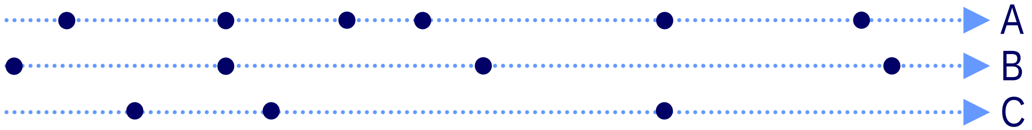
B ≡ F

D alternatesWith C

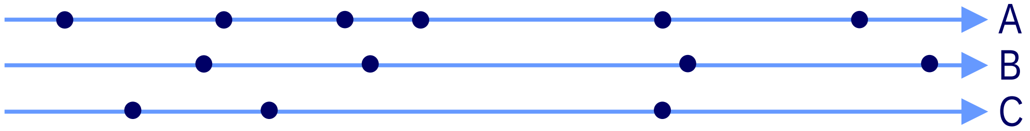


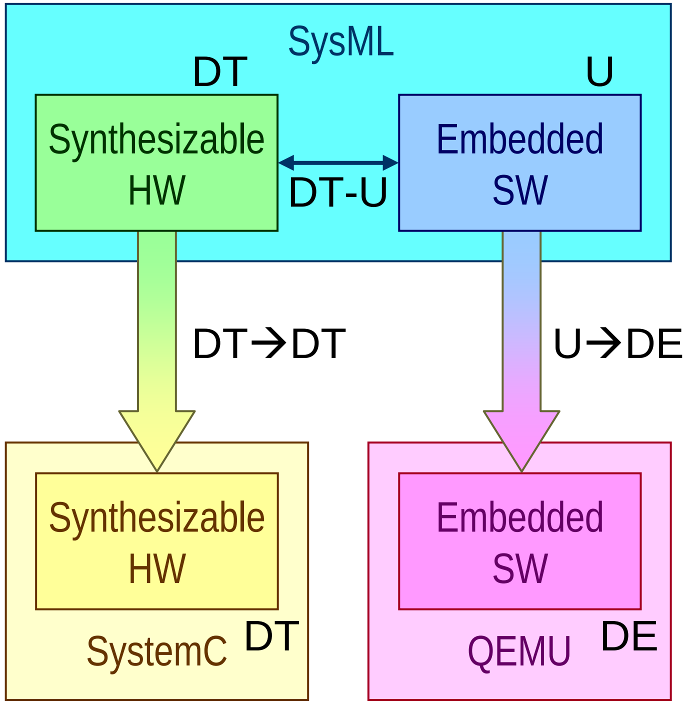
The ForSyDe Formal Support

- SW generation



B alternatesWith C, doesn't?





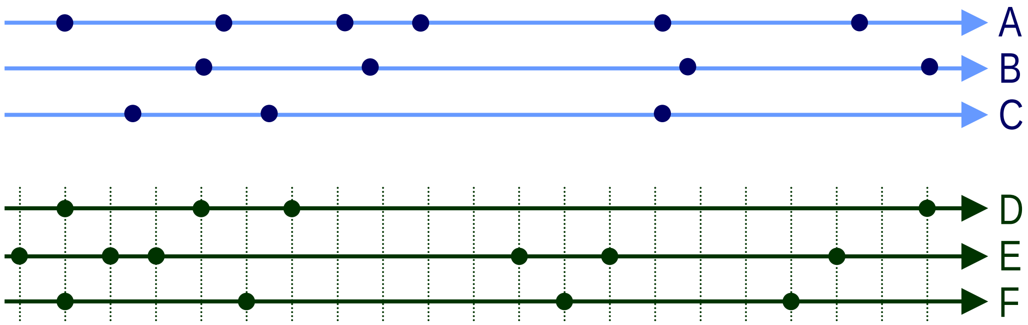
The ForSyDe Formal Support

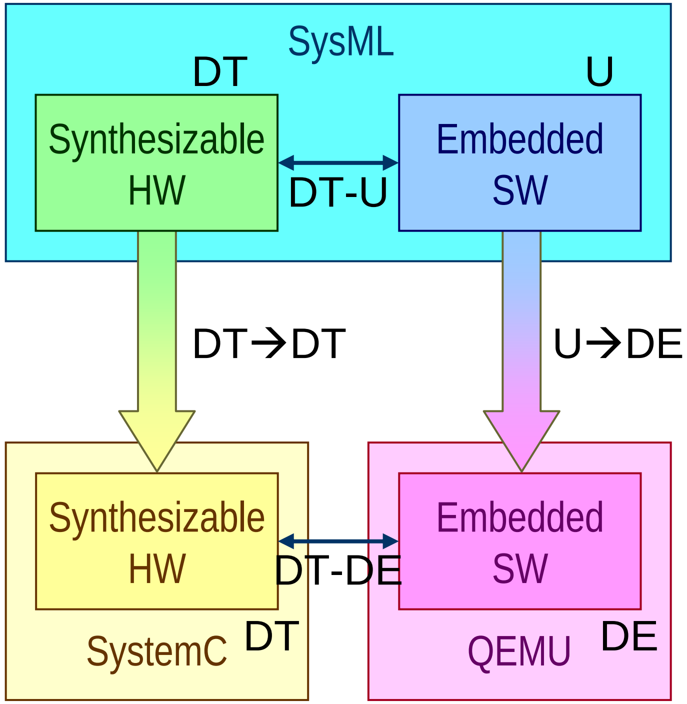
- HW/SW co-simulation

B ≡ F, doesn't't?

B alternatesWith C, doesn't't?

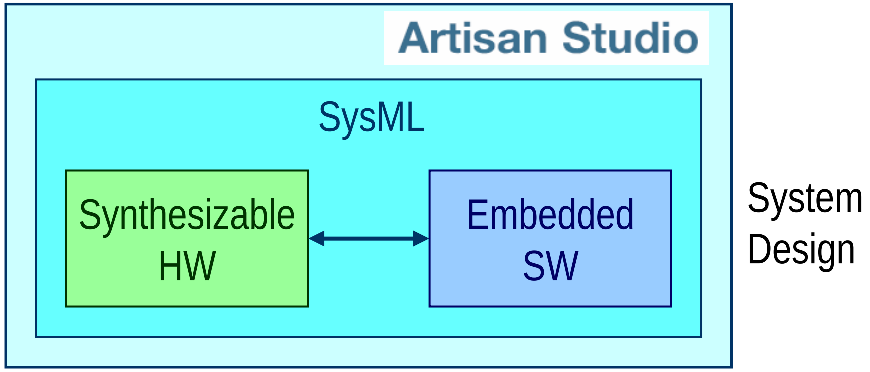
D alternatesWith C, doesn't't?





Extension of the Saturn SysML profile

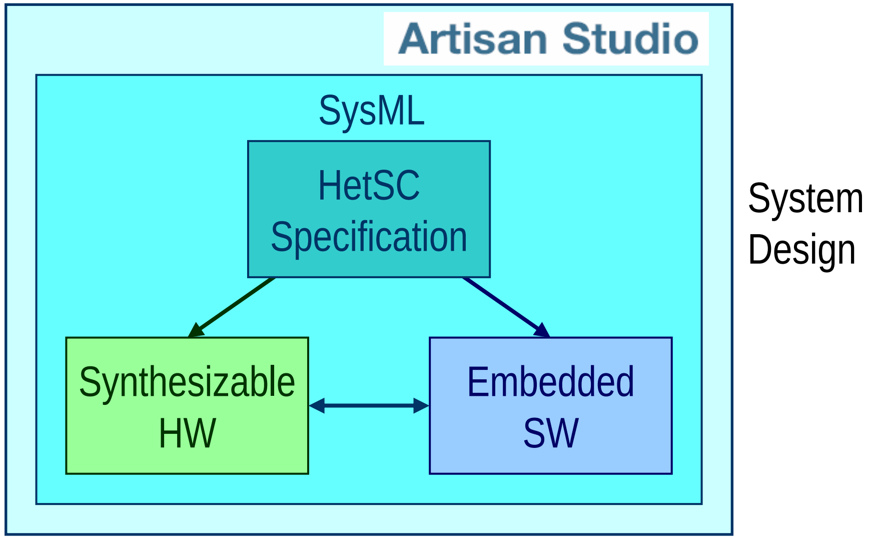
- System-level specification capabilities
  - System-level modeling
  - Reference to any HW/SW implementation



CopyRight by
University of Cantabria©

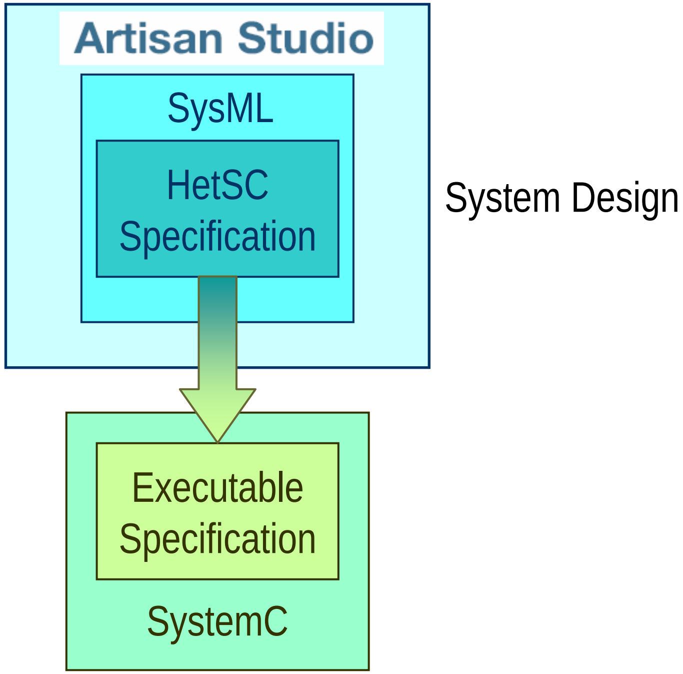
Extension of the Saturn SysML profile

- System-level specification capabilities
  - System-level modeling
  - HetSC profile



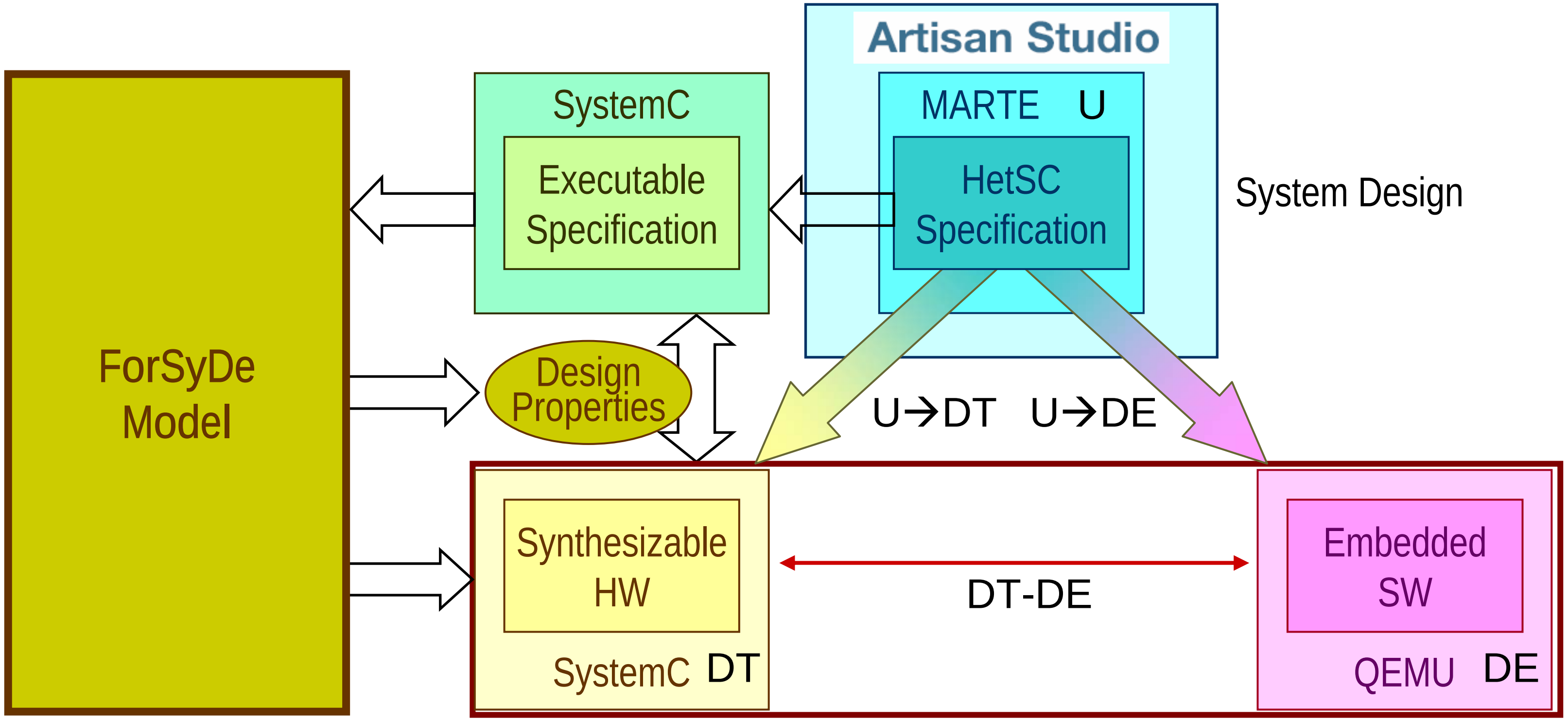
Extension of the Saturn SysML profile

- SystemC HetSC generation



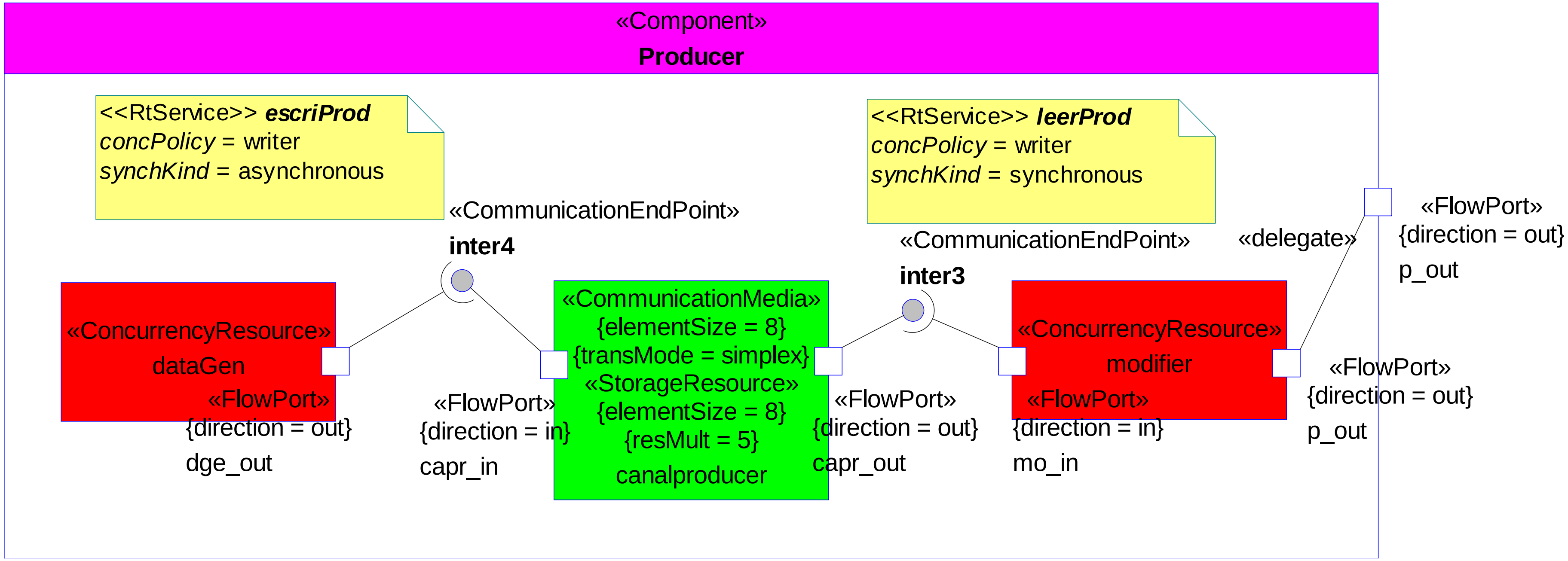
MARTE/SystemC interoperability formalization

- MARTE computation and communication stereotypes
  - Generic Resource Modeling



MARTE/SystemC interoperability formalization

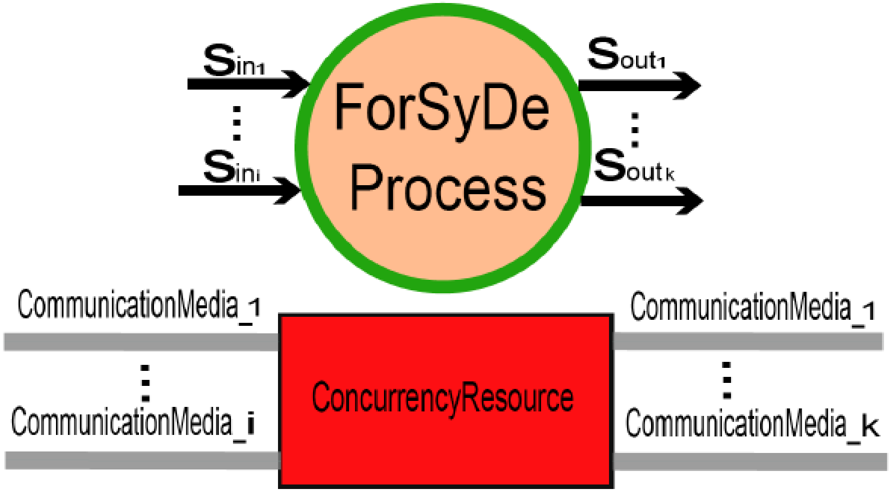
- MARTE computation and communication stereotypes
  - Generic Resource Modeling



  - Different MoCs can be identified

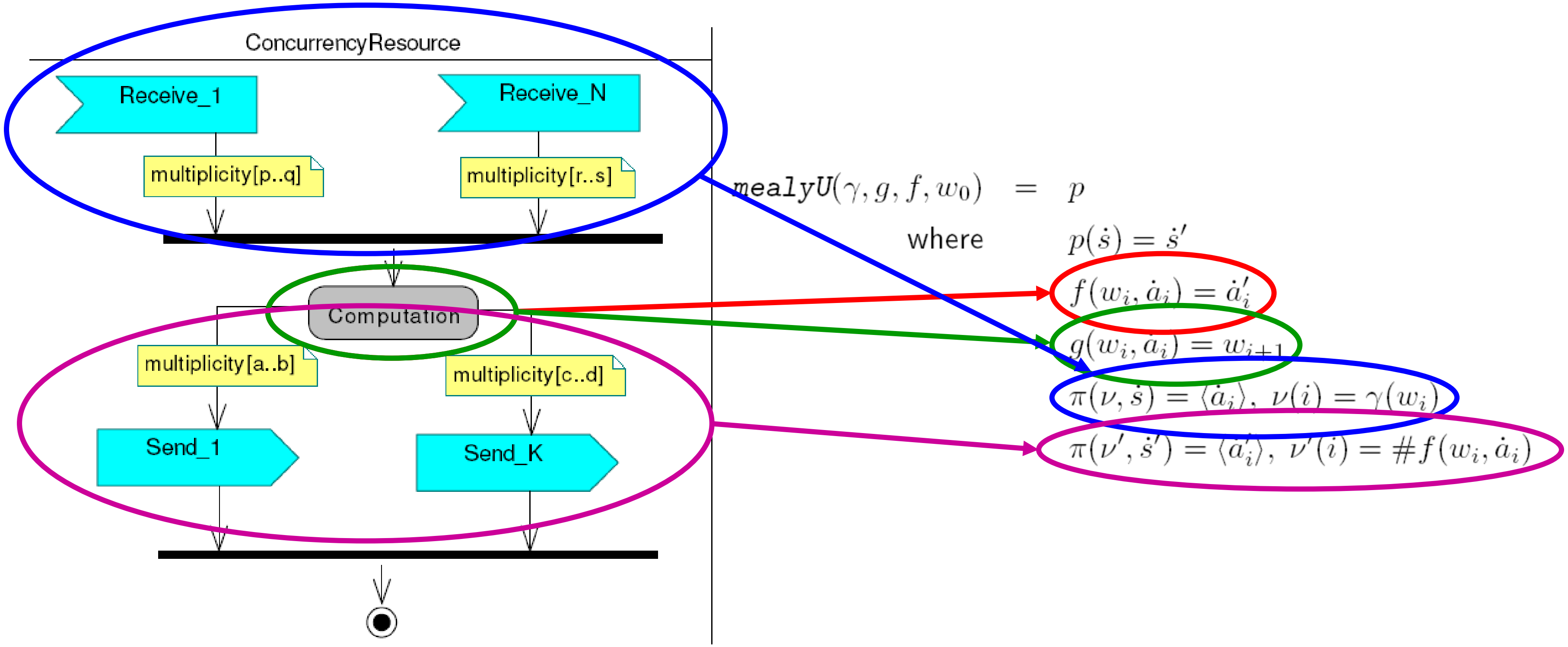
MARTE/SystemC interoperability formalization

- MARTE computation and communication stereotypes
  - Generic Resource Modeling



MARTE/SystemC interoperability formalization

- MARTE computation
  - Activity Diagrams



$$\texttt{mealyU}(\gamma, g, f, w_0) \quad = \quad p$$

where

$$p(\dot{s}) = \dot{s}'$$

$$f(w_i, \dot{a}_i) = \dot{a}'_i$$

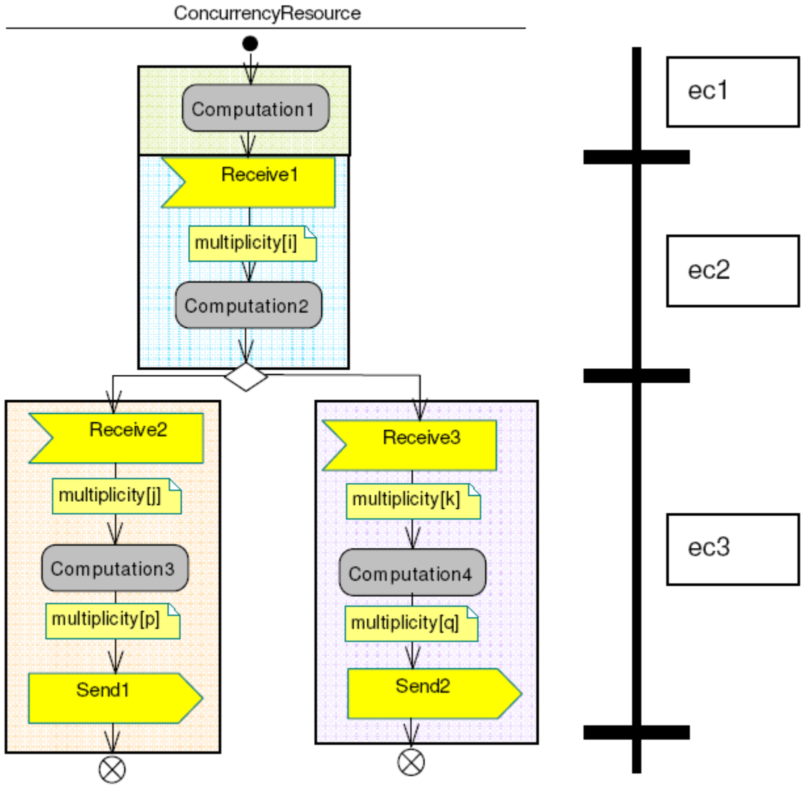
$$g(w_i, \dot{a}_i) = w_{i+1}$$

$$\pi(\nu, \dot{s}) = \langle \dot{a}_i \rangle, \ \nu(i) = \gamma(w_i)$$

$$\pi(\nu', \dot{s}') = \langle \dot{a}'_i \rangle, \ \nu'(i) = \#f(w_i, \dot{a}_i)$$

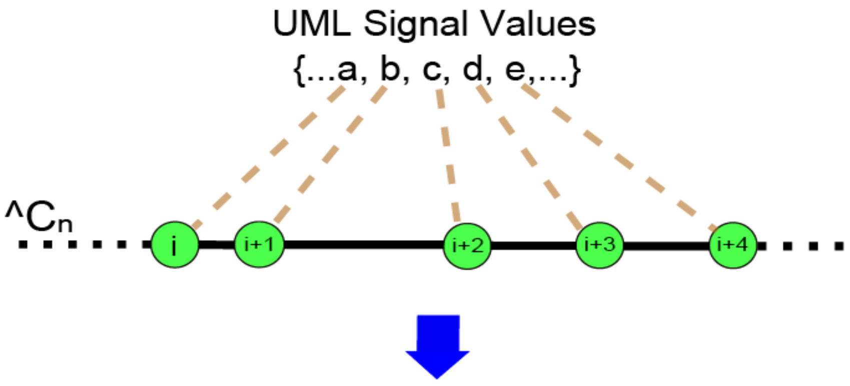
MARTE/SystemC interoperability formalization

- MARTE computation
  - Activation Cycles



MARTE/SystemC interoperability formalization

- MARTE timing
  - UML signals and MARTE clocks as ForSyDe signals
  - Clock relations
    - Clock Constraints Specification Language



$S_{ForSyDe} = <\ldots e_i, e_{i+1}, e_{i+2}, e_{i+3}, e_{i+4}, \ldots>$

MARTE/SystemC interoperability formalization

- Verification of design constraints
  - MARTE clock relations

B $\equiv$ F	D alternatesWith C

-- psl property B_synchonous_F is always (B.write_CALL() and F.write_CALL());	*-- psl property D_alternates_C is always (D.write_CALL() -> next(C.write_CALL() before D.write_CALL()));*
-- psl assert B_synch_F;	*-- psl assert D_alternates_C;*

  - SystemC assertions

Conclusions

- ForSyDe as formalization metamodel for Saturn
  - Providing a formal framework
  - Defining design rules and constraints
- HetSC extension to the Saturn profile
- MARTE/SystemC interoperability formalization
  - First step towards a SystemC generation methodology
    - Based on a formal framework
    - Predictable executable specification results
  - Application to system design verification