SystemC as an Heterogeneous System Specification Language

Eugenio Villar
Fernando Herrera
University of Cantabria

November 16-17, 2006 Zurich, Switzerland

Challenges

- Massive concurrency
  - Complexity
    - PCB
    - MPSoC with NoC
  - Nanoelectronics

Challenges

- Heterogeneity
  - Algorithmic heterogeneity
    - DSP, control, protocol stacks, multimedia, ...
  - Constraints heterogeneity
    - RT, reliability, resources, performance, ...
  - Resource heterogeneity
    - ASIC, FPGA, μP, μC, ...
    - GPP, ASIP, AS-HW, eP-HW, AMS, ...
  - Methodological heterogeneity
    - Languages and tools

Main motivation

- Need for a specification methodology
  - Supporting different MoCCs
  - Heterogeneous
  - Executable
  - Link to implementation
    - HW and SW
  - Based on a standard language

SystemC as a system specification candidate

- SystemC specification methodology
  - Supporting different MoCCs 
  - Heterogeneous 
  - Link to implementation 
    - HW and SW
  - Executable 
  - Based on a standard language 

SystemC as a system specification candidate

- SystemC is committed to support system design
  - Valuable input to OSCI and the IEEE
  - Theoretical foundations to the standardization process

SystemC as a 'straw man'

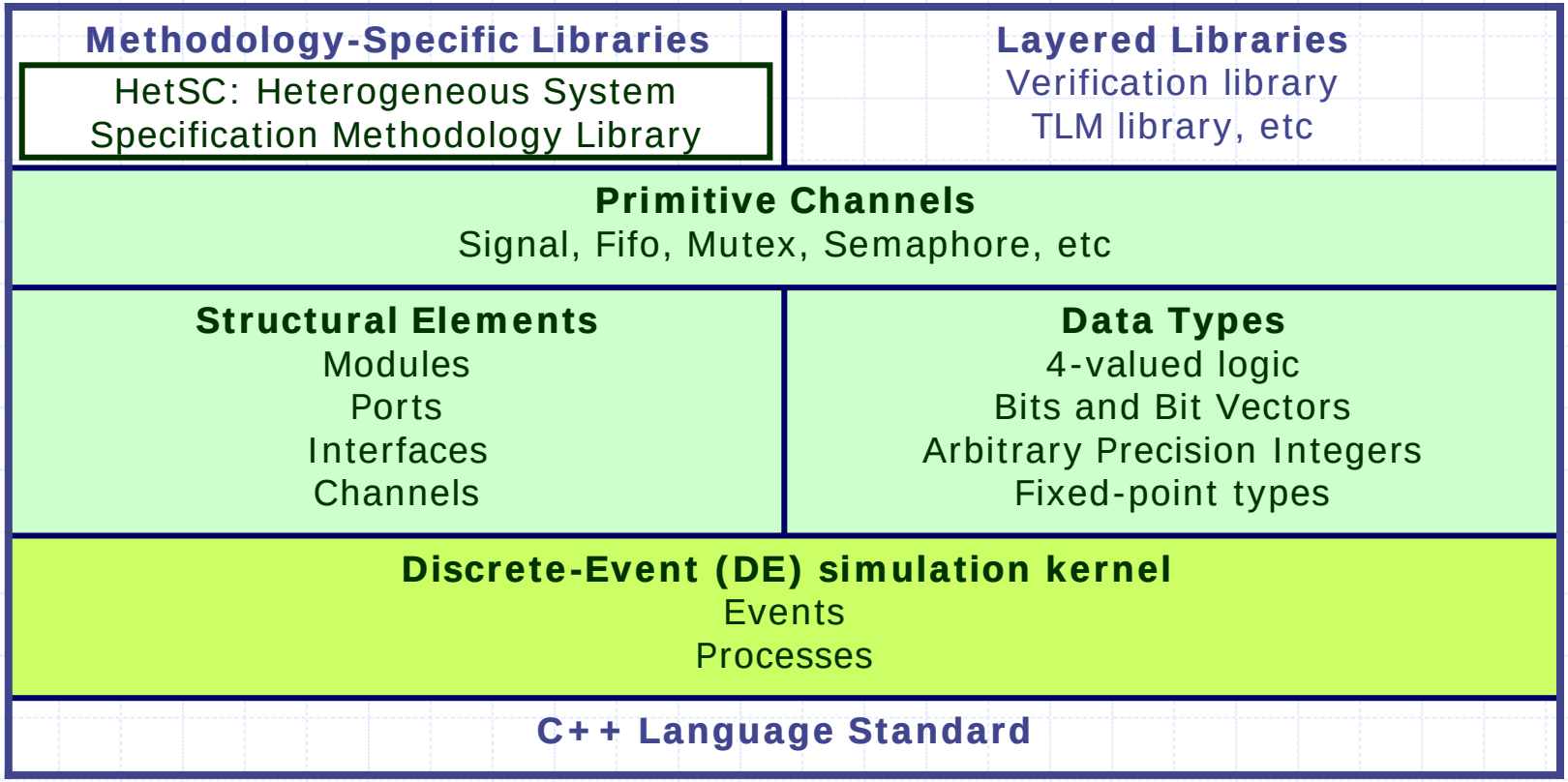
- System specification languages
  - Supporting different MoCCs
  - Heterogeneous
  - Link to implementation
    - HW and SW
  - Executable
  - Widely-used standard language

UNIFIED MODELING LANGUAGE™

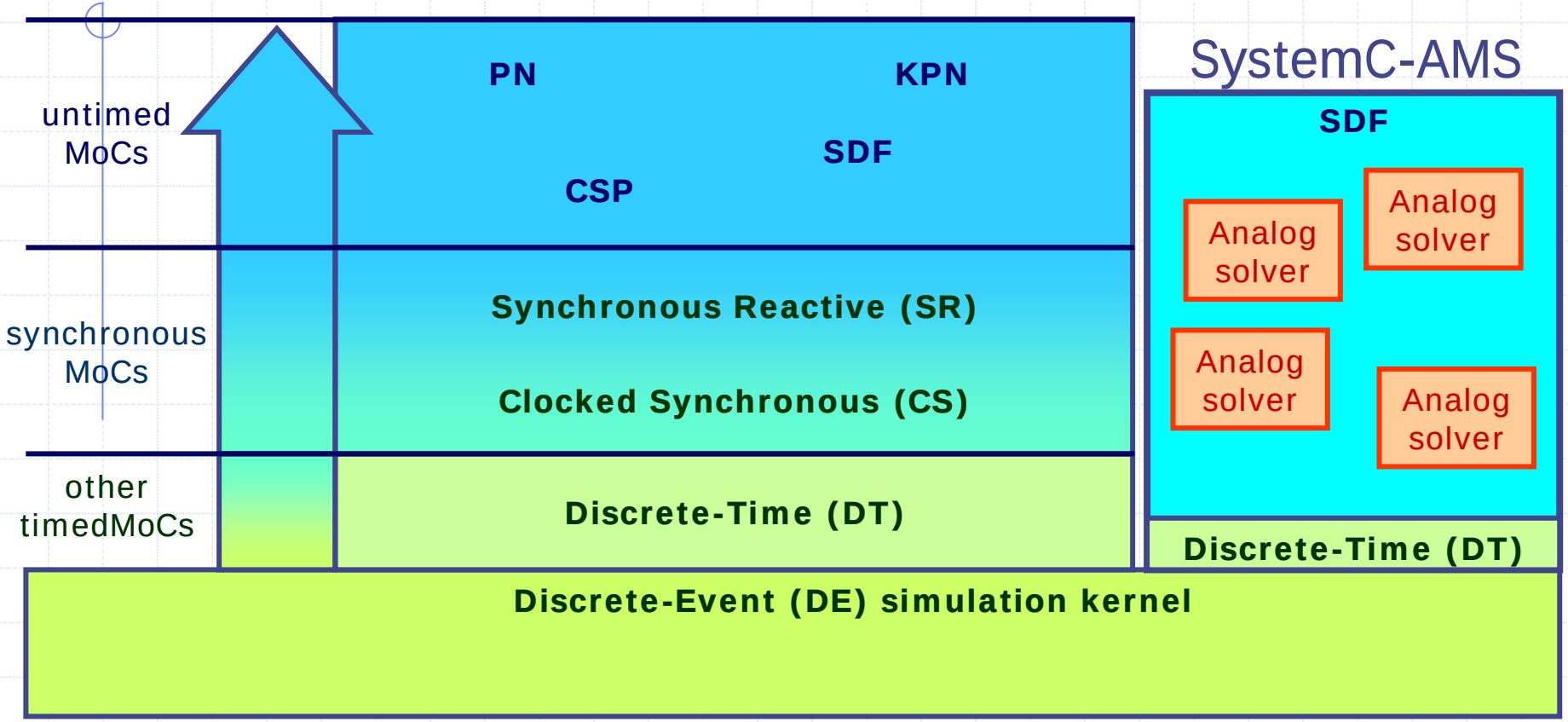
Agenda

- SystemC support for different MoCCs
- SystemC heterogeneous specification
- Link to implementation
- Future work

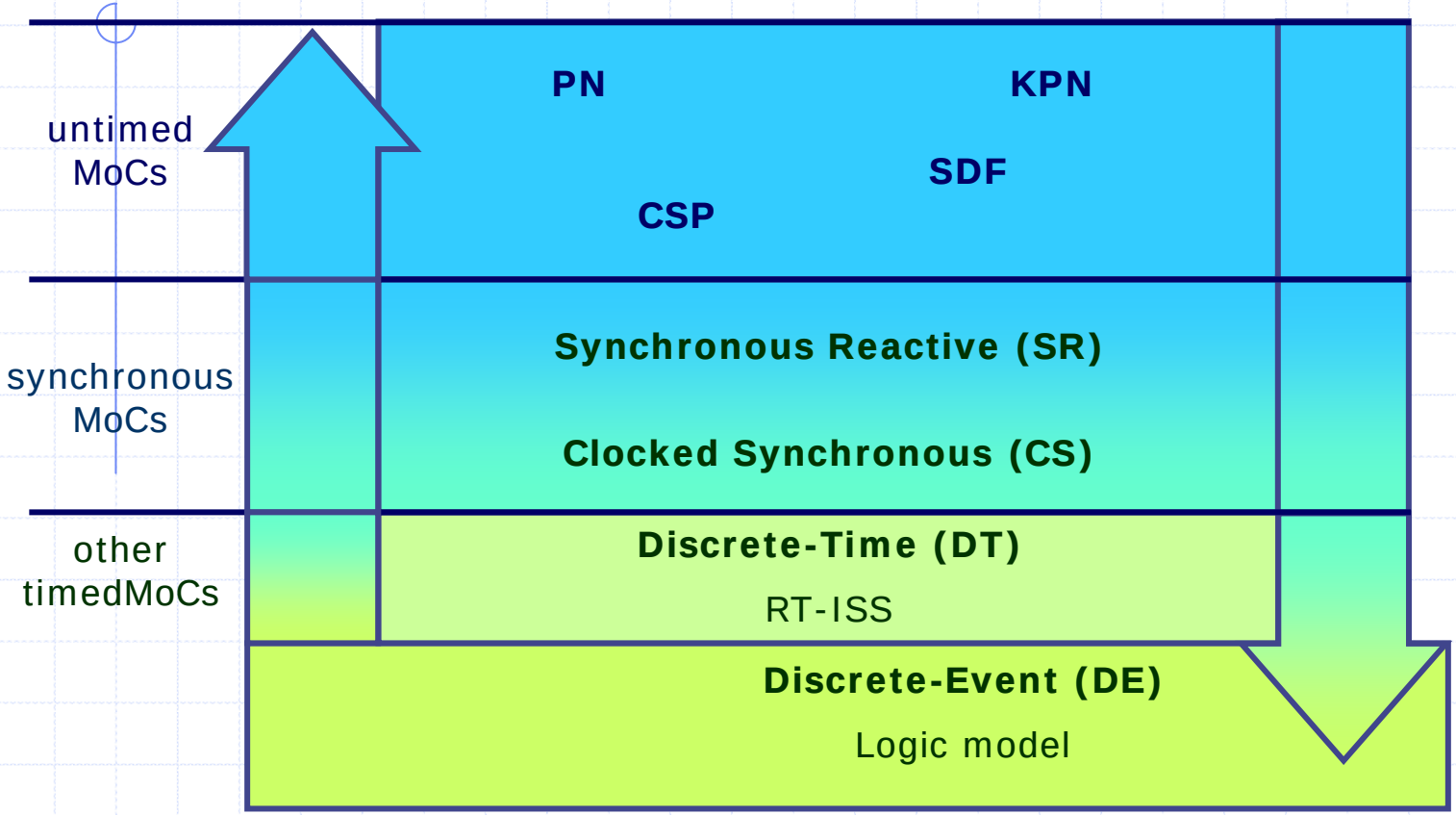
SystemC architecture



MoCC abstraction



Design refinement support



Fundamental question

- Which are the MoCCs that can be abstracted from the DE MoCC?
  - Strict answer: Only strict-timed MoCCs

$$F : MoCC \longrightarrow DE$$
$$e = (v,t) \in (V,T) \rightarrow F(e) = [F_V(v,t), F_T(v,t)]$$

$$F_T : (V,T) \longrightarrow NxN$$
$$e = (v,t) \in (V,T) \rightarrow F_T(v,t) = [F_t(v,t)\,tu, F_\delta(v,t)]$$

  - Limited design scope: RT-Logic synthesis

Fundamental question

- Which are the MoCCs that can be abstracted from the DE MoCC?
  - Strict answer: Only strict-timed MoCCs

$$F : MoCC \longrightarrow DE$$
$$e = (v,t) \in (V,T) \rightarrow F(e) = [F_V(v,t), F_T(v,t)]$$

$$F_T : (V,T) \longrightarrow NxN$$
$$e = (v,t) \in (V,T) \rightarrow F_T(v,t) = [F_t(v,t)\,tu, F_\delta(v,t)]$$

  - SystemC is committed to support system design

Fundamental question

- Which are the MoCCs that can be abstracted from the DE MoCC?
  - Relaxed answer: Any computable MoCC

Fundamental problems

- How to represent untimed events onto the DE MoCC?
  - Breaking the order relationship of δ cycles

$$e_1 = (v_1, t_1) \rightarrow F_T(e_1) = (t_{e_1}\ \text{ns}, \delta_{e_1})$$
$$e_2 = (v_2, t_2) \rightarrow F_T(e_2) = (t_{e_2}\ \text{ns}, \delta_{e_2})$$

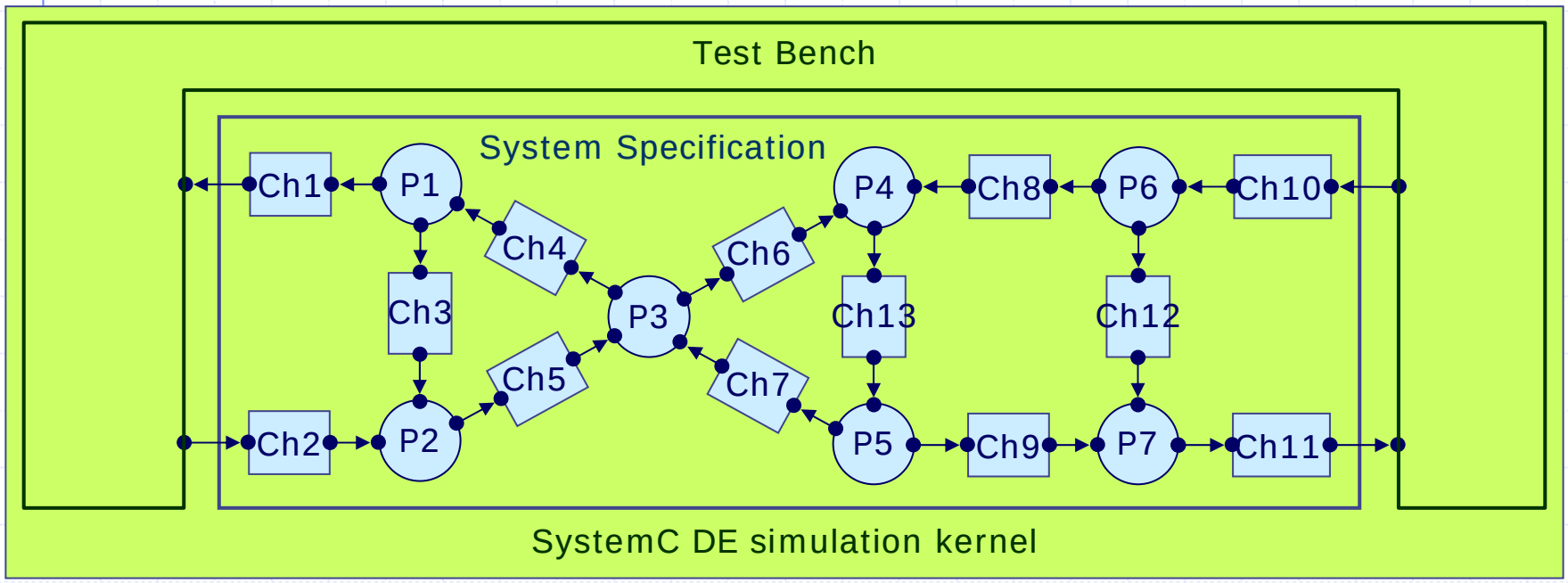
$$t_{e_1} < t_{e_2} \Rightarrow e_1 < e_2$$
$$t_{e_1} = t_{e_2} \text{ and } \delta_{e_1} < \delta_{e_2} \not\Rightarrow e_1 < e_2$$

Fundamental problems

- Why any MoCC?
  - Because DE can model any algorithm running on any computer
  - Efficiency
    - Ability to hide unnecessary details

SystemC specification structure

- SystemC processes connected by channels
  - Timing evolving during the design process
  - Strict-Timed Test Bench



SystemC specification syntax

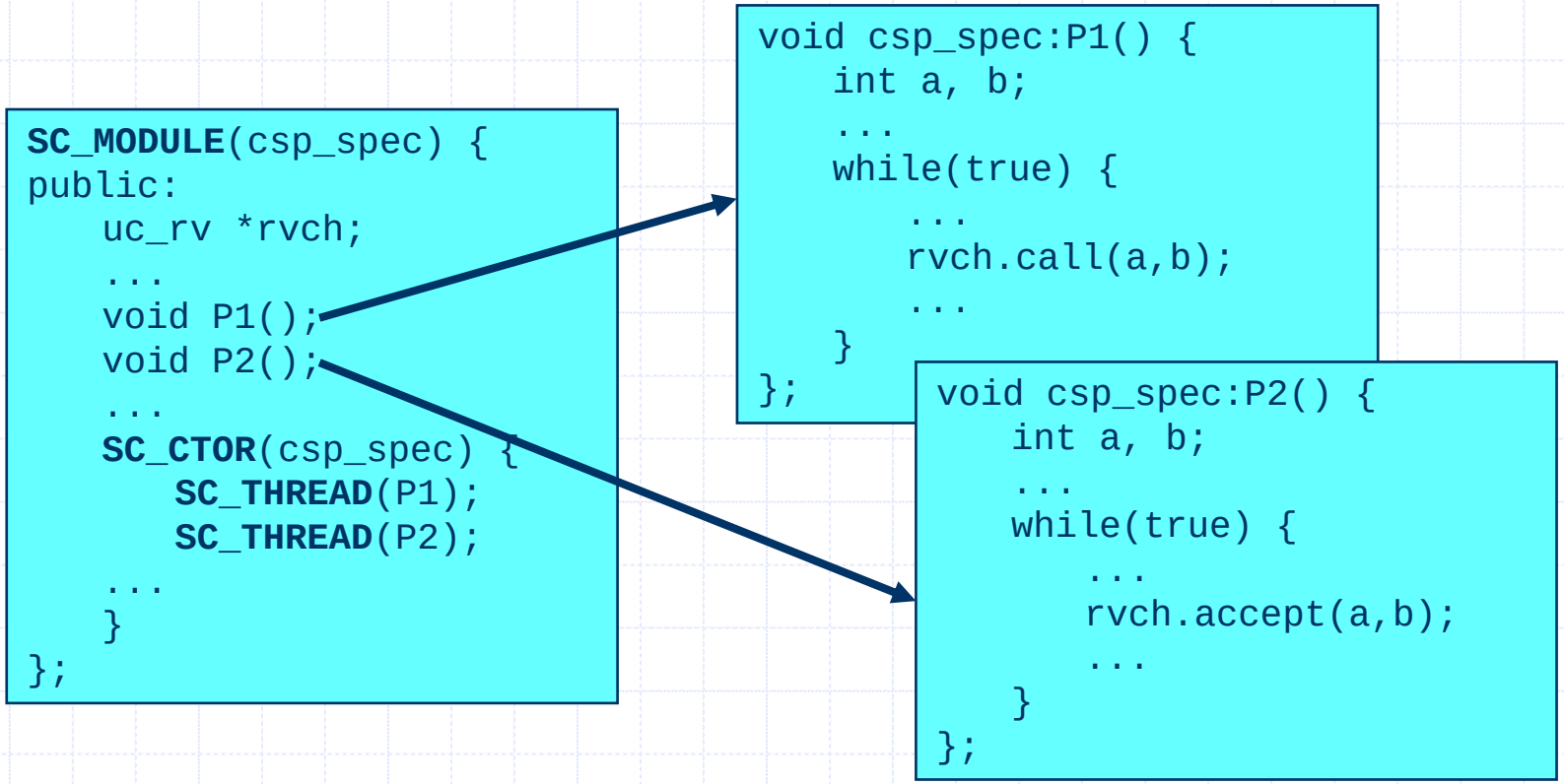
- Semantical channels
  - Associated to concrete implementations
    - uc_fifo
    - uc_rv
    - uc_sr
    - ...

SystemC specification syntax

- Concurrent processes
  - As few restrictions as possible
    - Communication and synchronization through channels

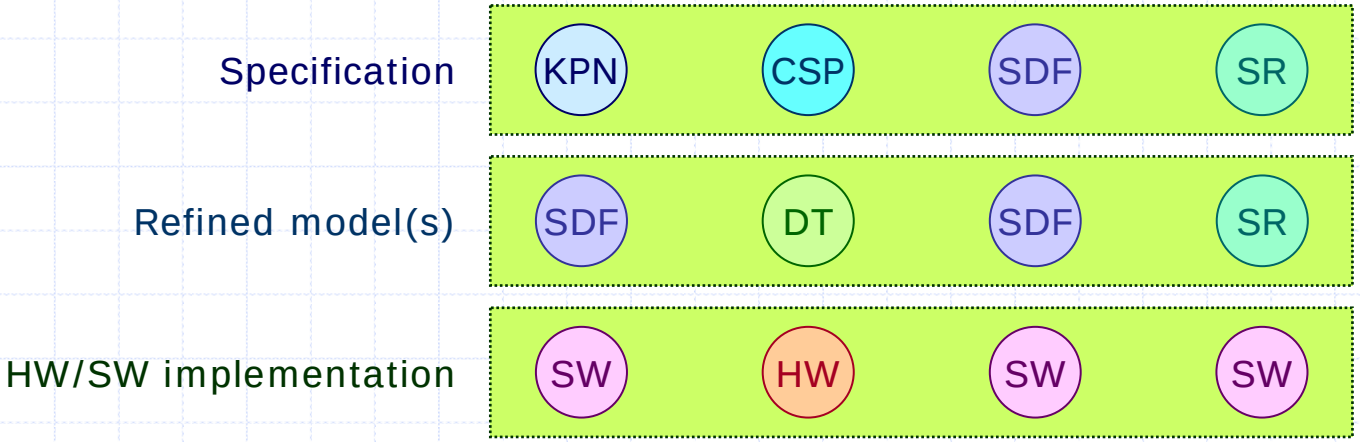
SystemC specification syntax

- Concurrent processes
  - As few restrictions as possible



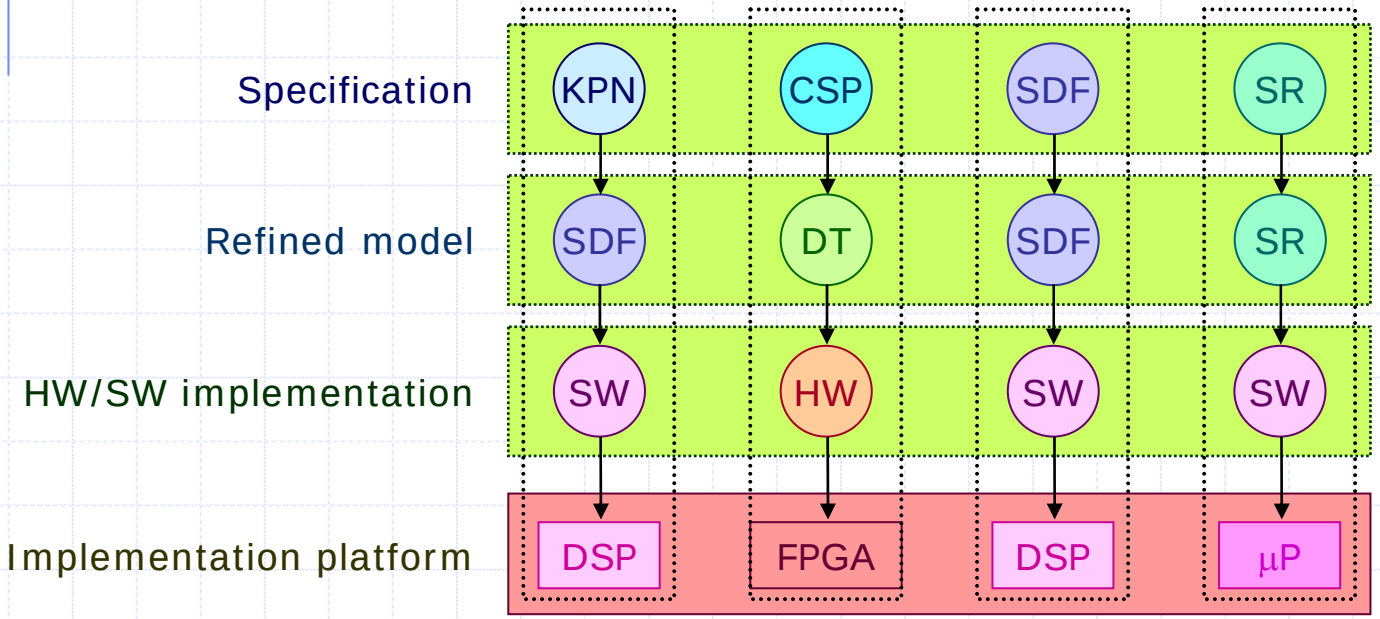
SystemC heterogeneous specification

- Horizontal heterogeneity
  - Ability to combine several MoCCs in the same specification



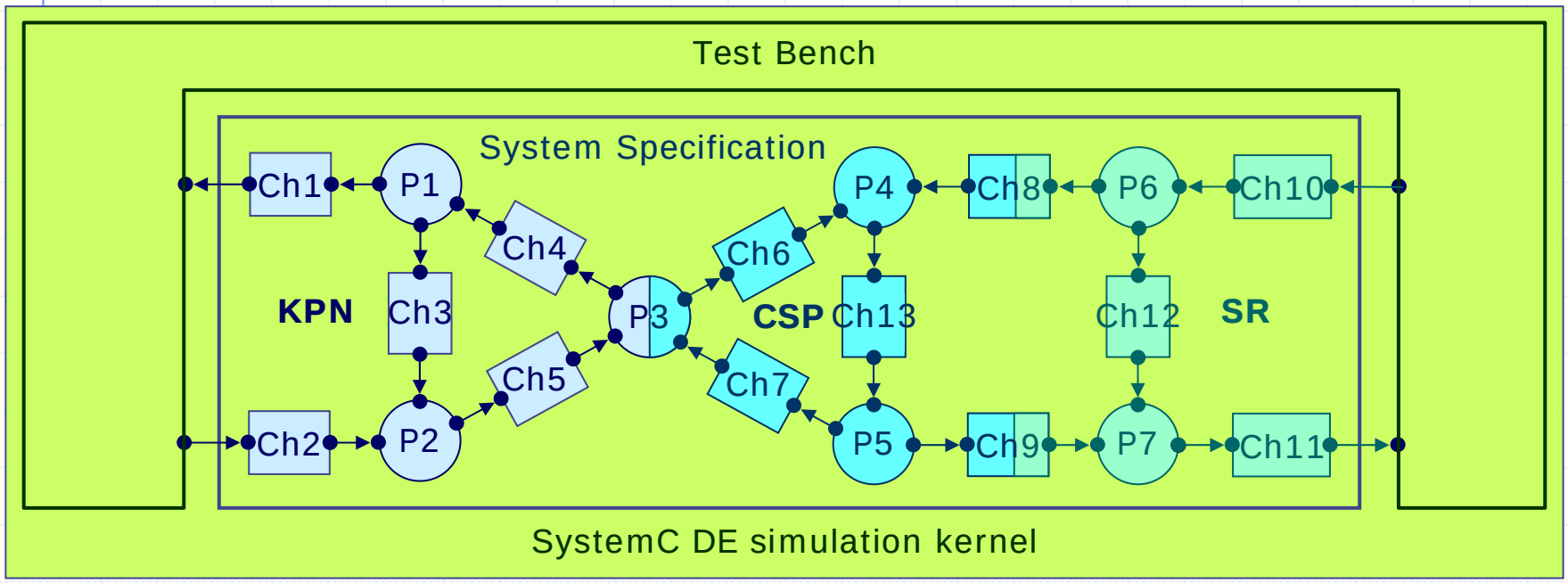
SystemC heterogeneous specification

- Vertical heterogeneity
  - Ability to transform one MoCC to another while preserving the equivalence



SystemC heterogeneous specification structure

- SystemC processes connected by channels
  - Border channels
  - Border processes



SystemC heterogeneous specification structure

- SystemC processes connected by channels
  - Border channels

```
SC_MODULE(csp_pn_spec) {
public:
    uc_rv_fifo *rvfifoch;
    ...
    void P1();
    void P2();
    ...
    SC_CTOR(csp_pn_spec) {
        SC_THREAD(P1);
        SC_THREAD(P2);
    ...
    }
};
```

```
void csp_pn_spec:P1() {
    int a, b;
    ...
    while(true) {
        ...
        rvfifoch.call(a,b);
        ...
    }
};
```

```
void csp_pn_spec:P2() {
    int a, b;
    ...
    while(true) {
        ...
        rvfifoch.read(a);
        rvfifoch.write(b);
        ...
    }
};
```

SystemC heterogeneous specification structure

- SystemC processes connected by channels
  - Border channels
  - Border processes

```
SC_MODULE(csp_pn_spec) {
public:
    uc_rv *rvch;
    uc_fifo *fifoch;
    ...
    void P1();
    void P2();
    void P3();
    ...
    SC_CTOR(csp_pn_spec) {
        SC_THREAD(P1);
        SC_THREAD(P2);
        SC_THREAD(P3);
    ...
    }};
```

```
void csp_pn_spec:P2() {
    int a, b;
    ...
    while(true) {
        ...
        fifoch.read(a);
        ...
        rvch.call(a,b);
        ...
        fifoch.write(b);
        ...
    }
};
```

HetSC

- SystemC heterogeneous specification methodology and associated library
  - Available at www.teisa.unican.es/HetSC

Link to implementation

- SW synthesis
  - Substitution of the simulation kernel by the equivalent RTOS functions
- HW synthesis
  - New generation of behavioral synthesis from C

Future work

- FPVI IST Project ANDRES
  - Formal foundation to HetSC
  - Based on ForSyDe
  - Close collaboration UC-KTH