WISES 07

Specification for SystemC-AADL interoperability

Eduardo de las Heras Palmero
Microelectronic Engineering Group
University of Cantabria

Outline

- Motivations
- General Concepts
  - AADL
  - SystemC
  - PERFidiX and SCope
- AADL-SystemC Design Flow
- Mapping AADL to SystemC
- Example

Motivations

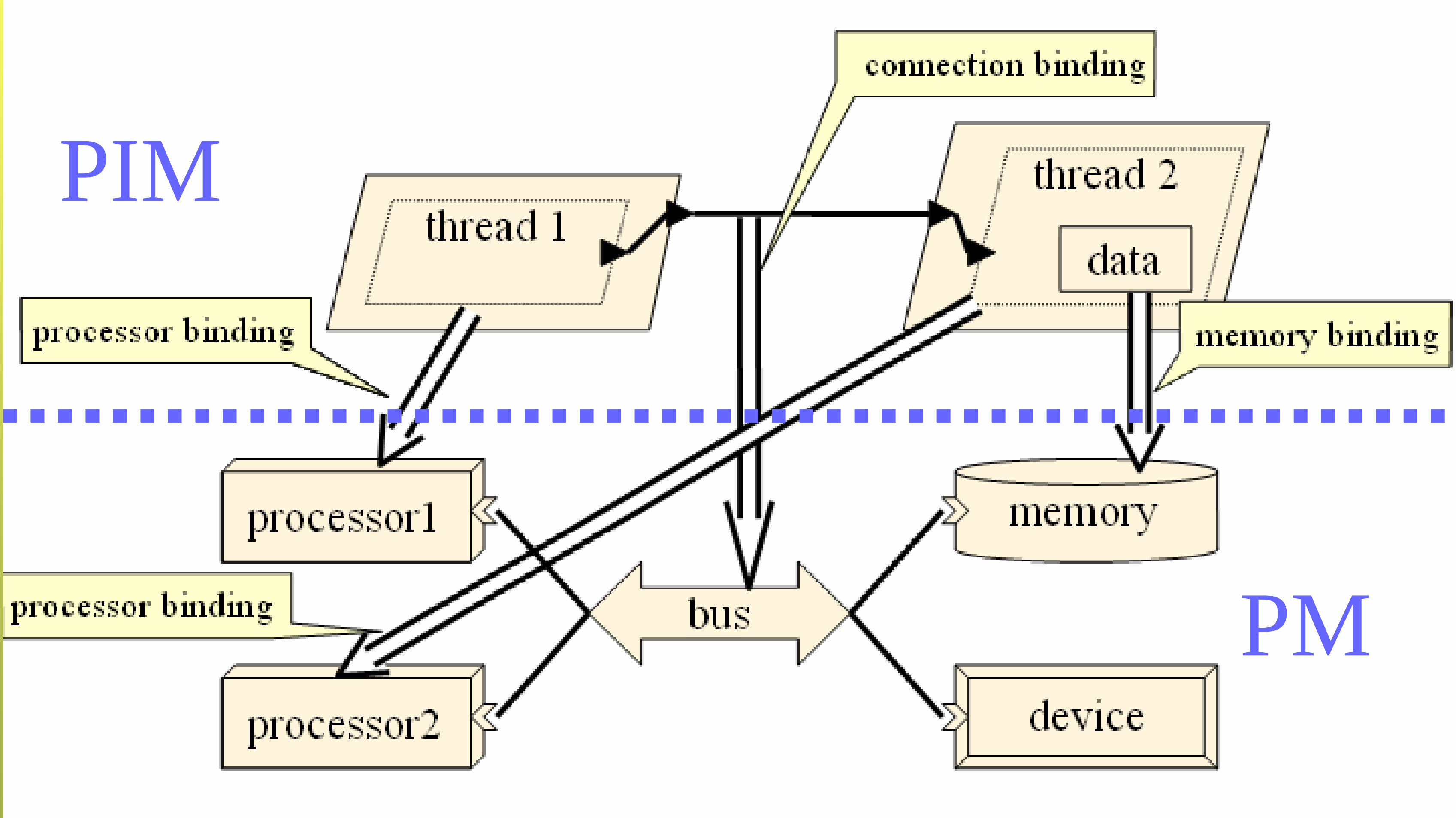
- System design issues:
  - Incomplete capture of specification
  - Need for design refinement and validation
  - Impact of functional and non-functional properties until the system integration
    - Timing properties
    - Software/Hardware co-design

AADL Concepts

Architecture Analysis & Design Language

- Standard by the SEI, November 2004
- Graphical and textual Language
- Architecture and model based design
- Precise syntax and semantics
- Specification of Tasks and communications
- Enable analysis and validation of constraints
- Large-scale architectures in a single model
- Incrementally refined
- Analyze the system structure and runtime behavior

AADL Concepts

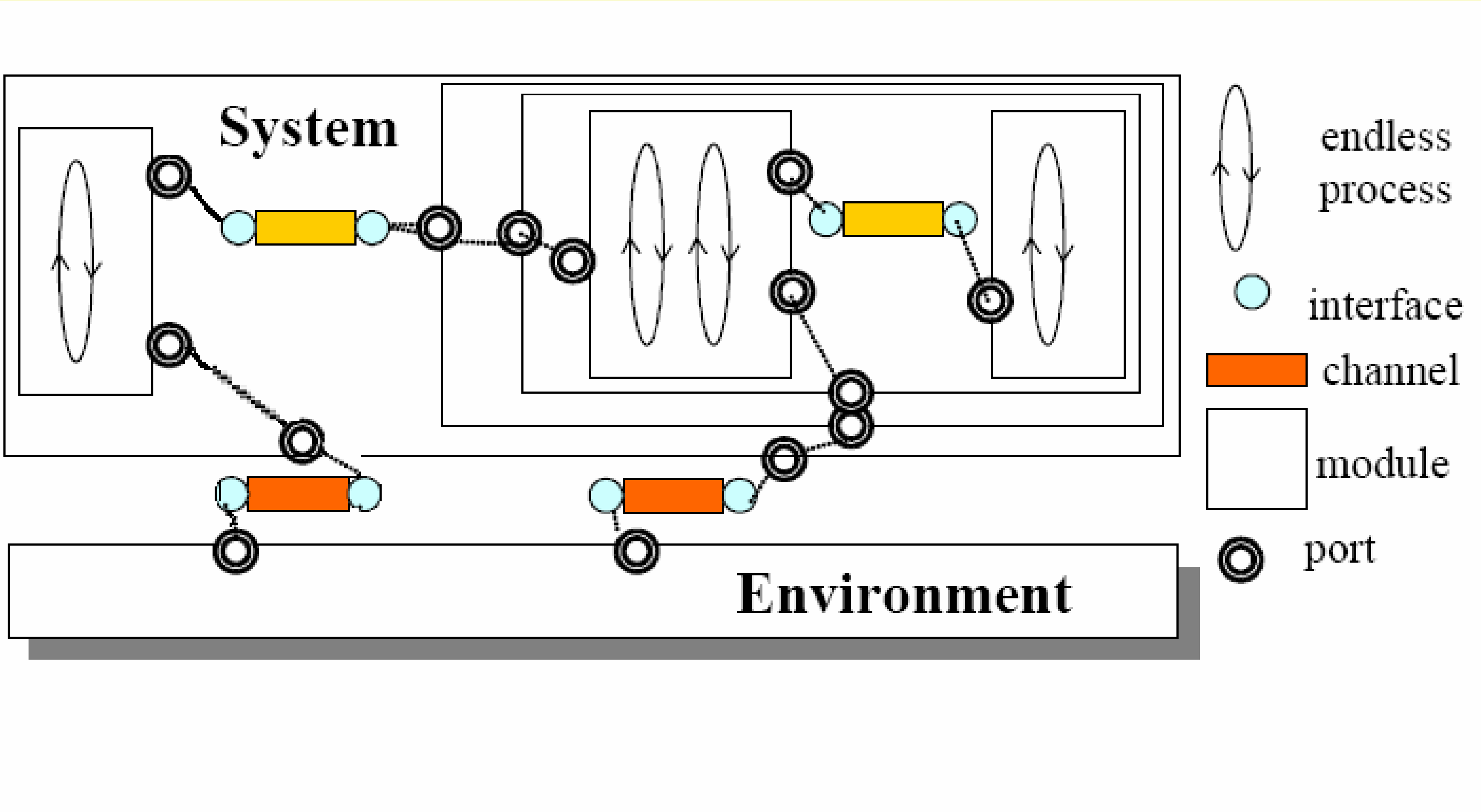


SystemC Concepts

- SystemC features
  - Standard platform for system design (IEEE 1666) developed by the OSCI
  - C++ extension
  - Strict-time, event driven simulator
  - Hierarchical Design
  - Concurrent Execution Kernel

SystemC Concepts

- SystemC Basic Elements

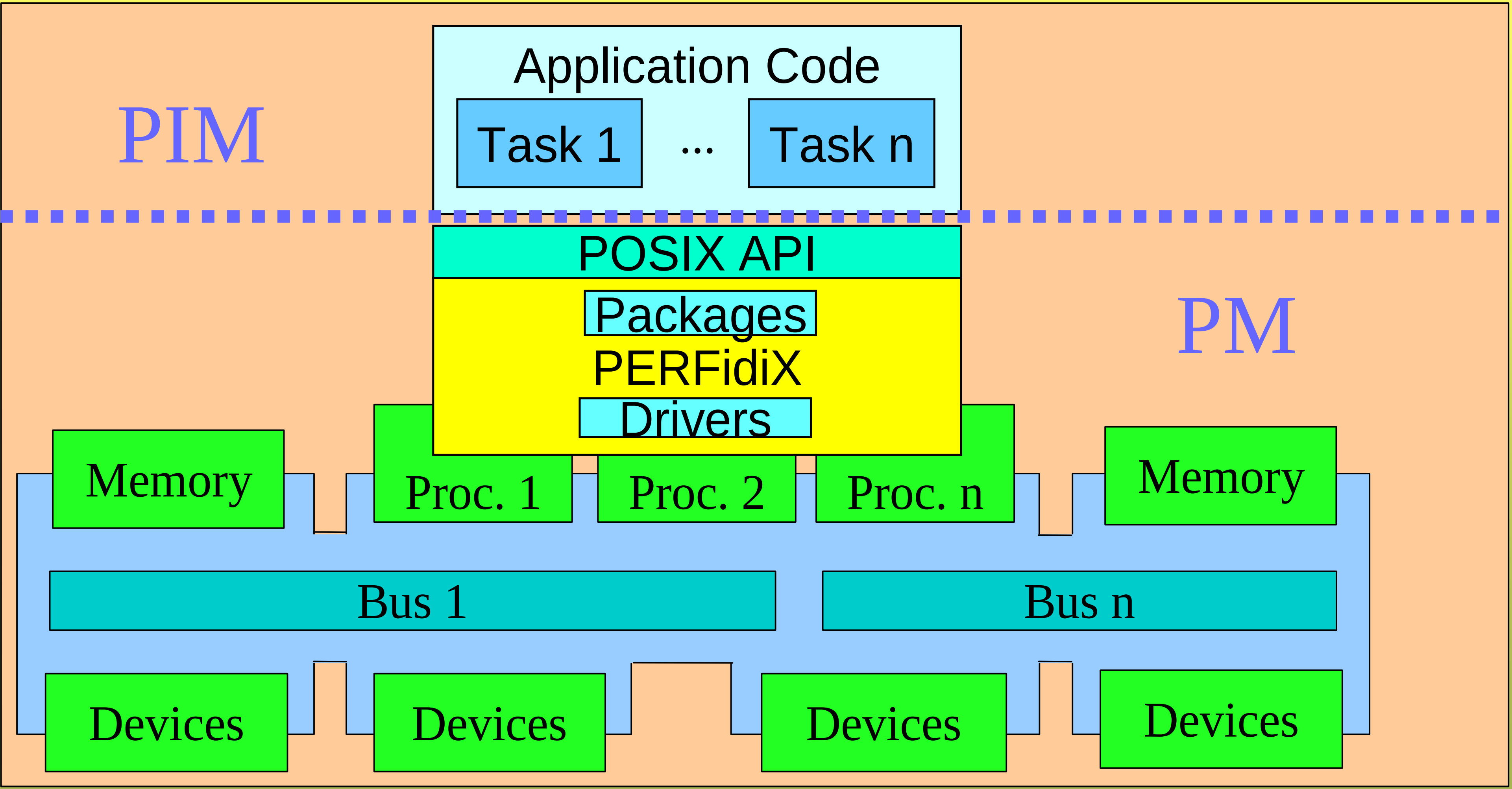


SCope Concepts

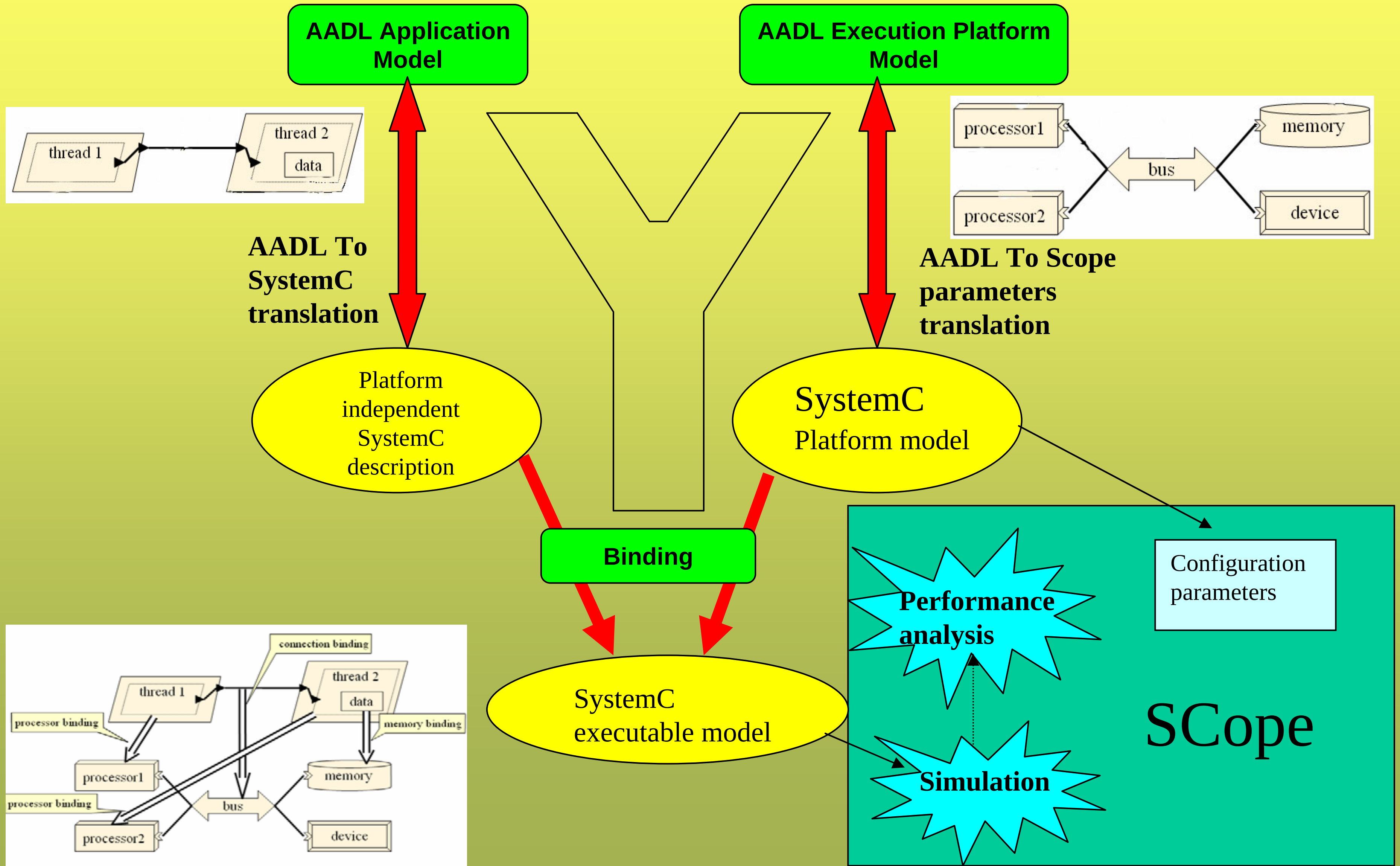
- System Co-simulation and Performance Estimation in SystemC
  - Extension of PERFidiX library
  - Multi-processor SW source-code simulation
    - OS Modelling
      - POSIX
    - Timed SW simulation
    - Performance estimation of SW code
      - Time & Power

www.teisa.unican.es/scope

SCope Concepts



AADL-SystemC Design Flow



AADL Application Model

AADL Execution Platform Model

AADL To SystemC translation

AADL To Scope parameters translation

Platform independent SystemC description

SystemC Platform model

Binding

SystemC executable model

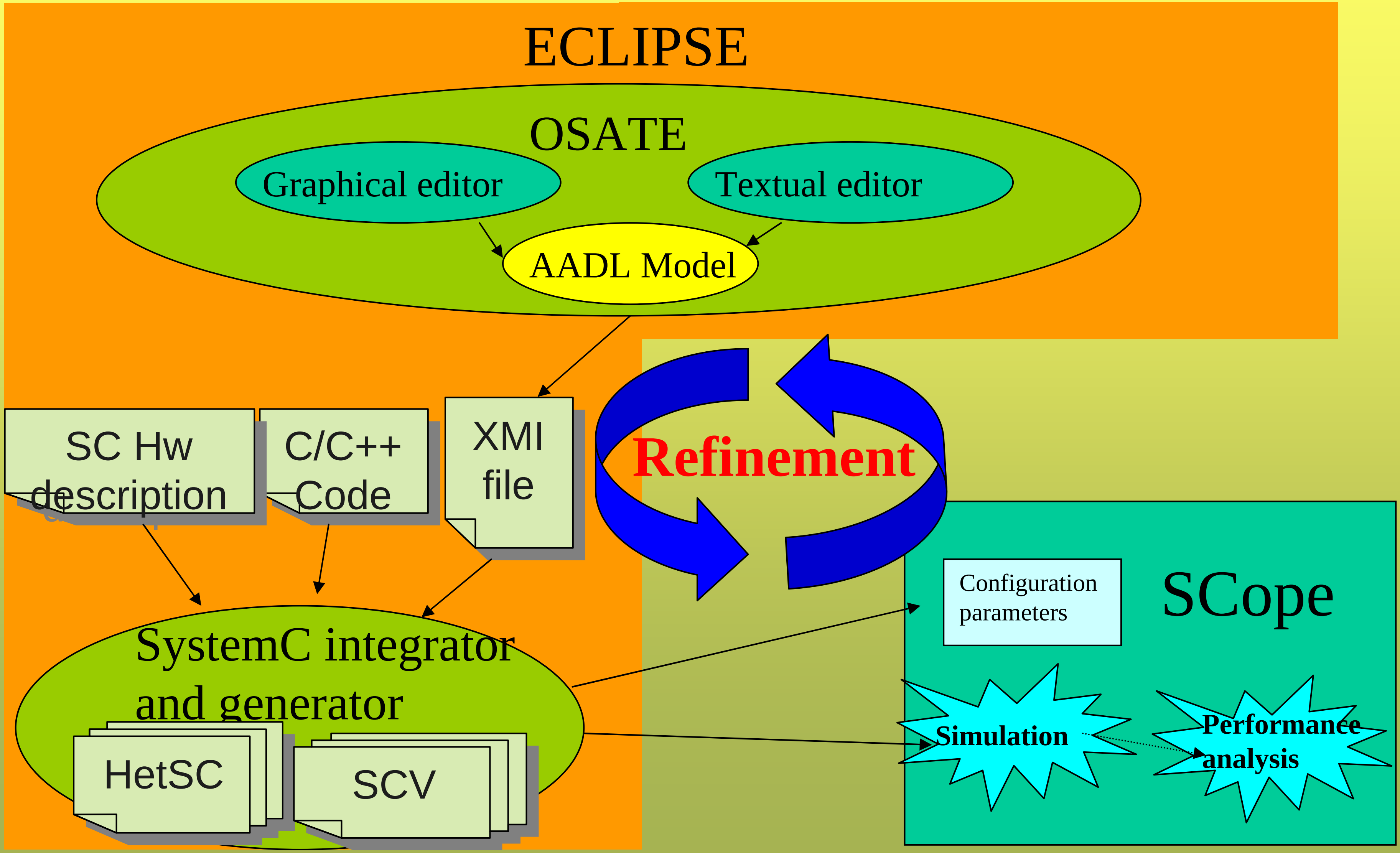
Performance analysis

Configuration parameters

SCope

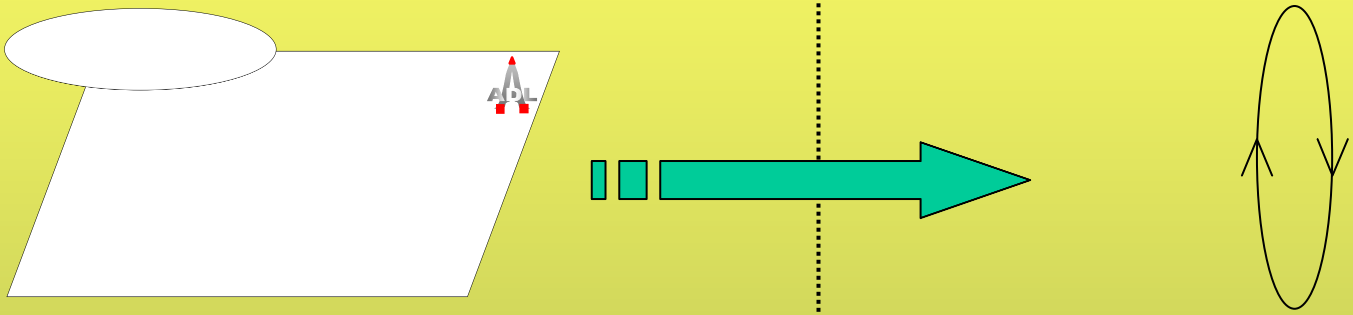
Simulation

AADL to SystemC Framework



AADL Semantics in SystemC

AADL



Thread: Schedulable unit of sequential source code.

- Properties
  - •Dispatch protocol

    100ms

    - •Period
  - •Deadline

SystemC

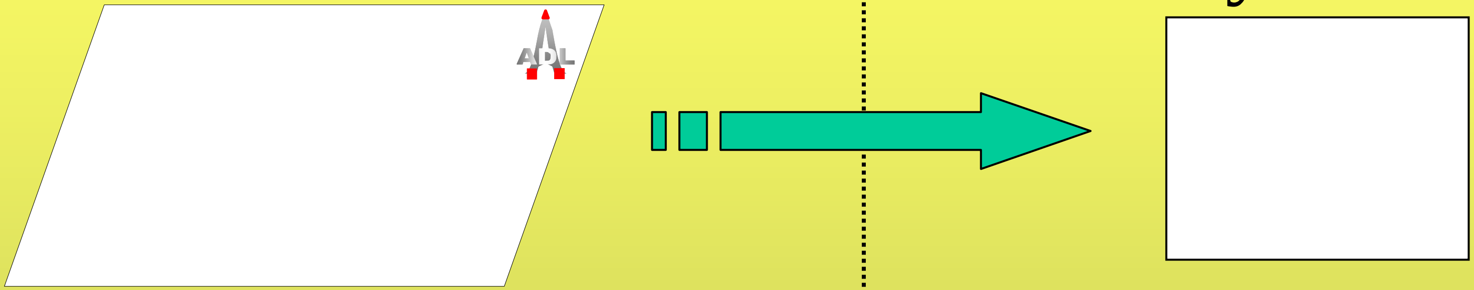
SC_THREAD: Is Called once when simulation Start.

- Properties
  - •Specific SC_THREAD implementation
  - •SC_TIME, wait (SC_TIME)
  - •Assertions SCV

AADL Semantics in SystemC

AADL

SystemC



Process: space partitioning where protection is provided

SC_MODULE: principle structural building blocks of SystemC

Subprogram: sequentially executable source text

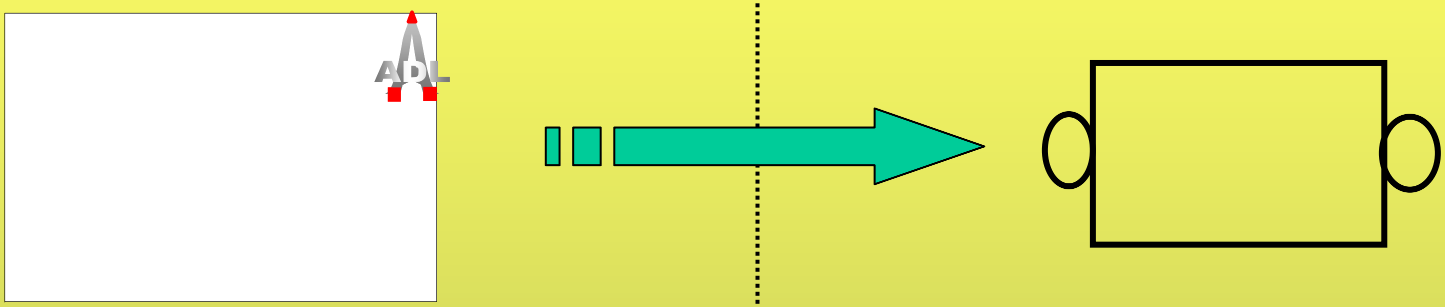
C++ function: called

from the SC_THREAD

AADL Semantics in SystemC

AADL

SystemC



Data: Enable manipulate data in concurrently in non-deterministic order.

- Properties
  - Concurrency_Control_Protocol

Channel: Enable communication between modules

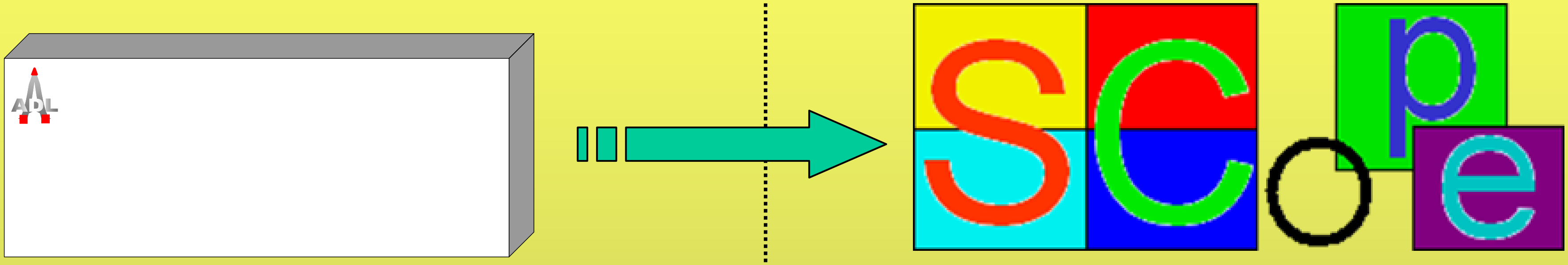
•Properties

•Semaphores, mutex, custom channels.

AADL Semantics in SystemC

AADL

SystemC



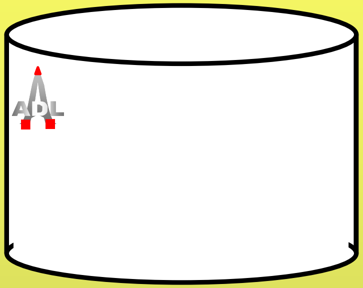
Processor: Abstraction of hardware and software responsible for scheduling and executing threads.

- Properties
  - Process_Swap_Execution_time
  - Thread_Swap_Execution_time → SCope configuration parameters
  - Scheduling_Protocols → POSIX scheduling_protocols

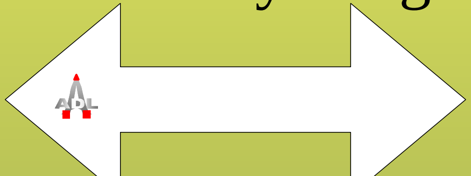
High level, POSIX simulation library and performance Analysis

AADL Semantics in SystemC

AADL



Memory: platform component that stores binary images.



Bus: platform component that can exchange control and data between modules.

- Properties
  - Transmission time, propagation delay

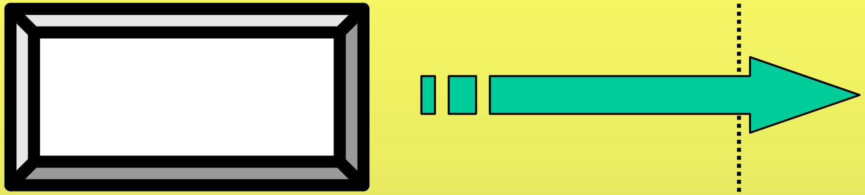
SystemC

System Co-simulation and Performance Estimation in SystemC

SCope configuration parameters

AADL Semantics in SystemC

AADL

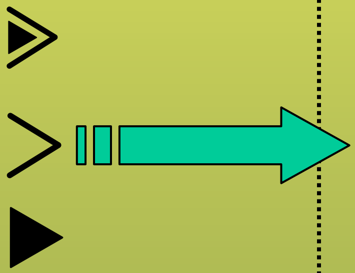


Devices: Execution platform component that interface with the exterior

Event data port

Event port

Data port



Ports and Connections: Logical Connections to exchange control and data between threads.

SystemC

SystemC description at various levels:

- TLM
- RTL
- Synthesis

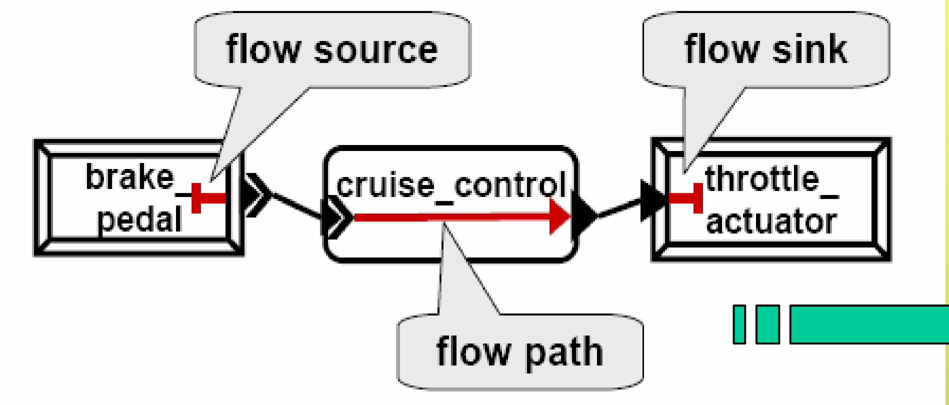
Signal channel, ports, interface

FIFO channel ports, interface

Custom Channels, ports, interface

AADL Semantics in SystemC

AADL



Flows: support for various forms of flow analysis

- •Flow source
- •Flow path
- •Flow sink

SystemC

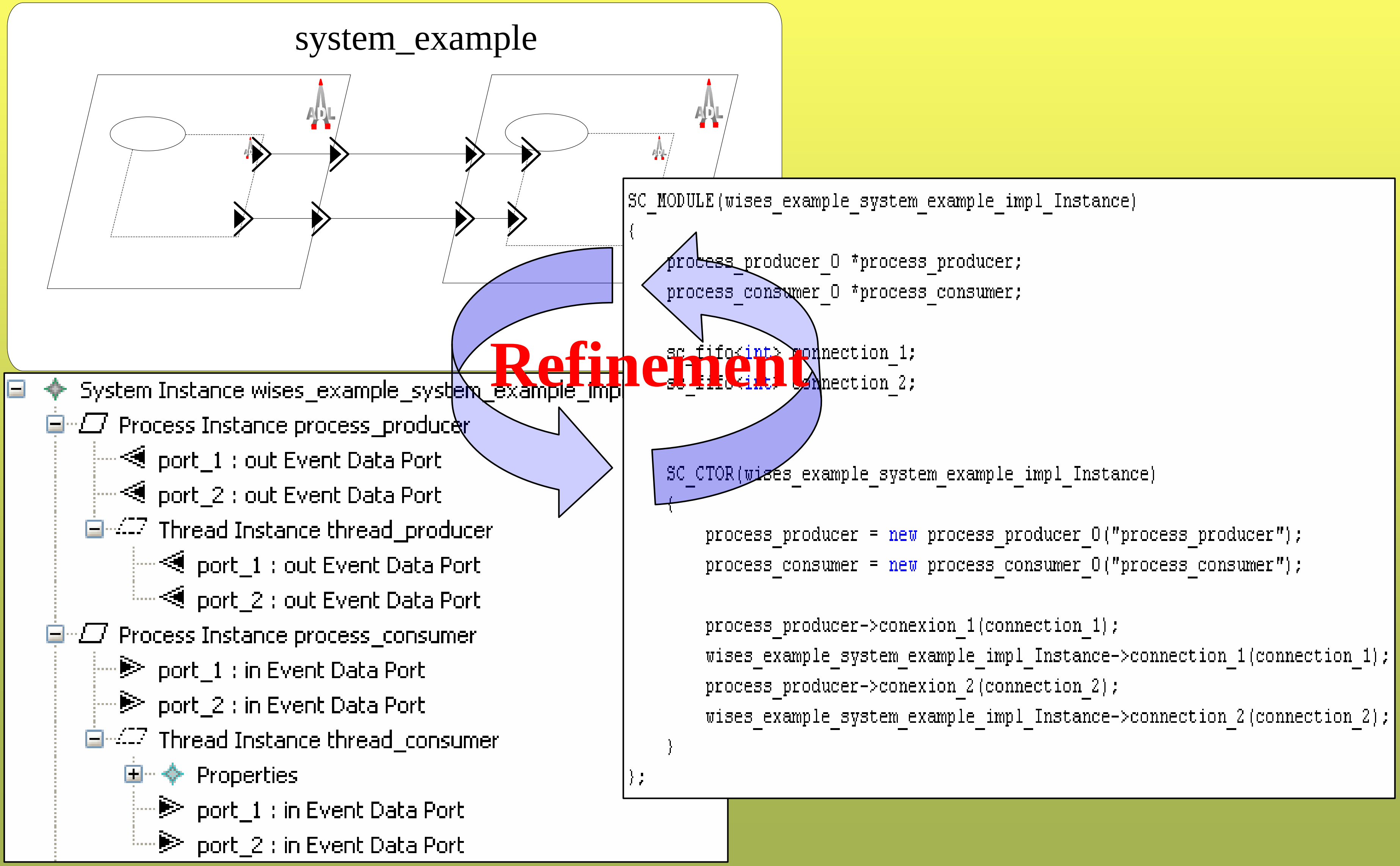
Corresponding access to subcomponents involve in the flow implementation

Random generation of tokens using SCV

Implementation of Write and read access method

Data recording for posterior analysis using SCV

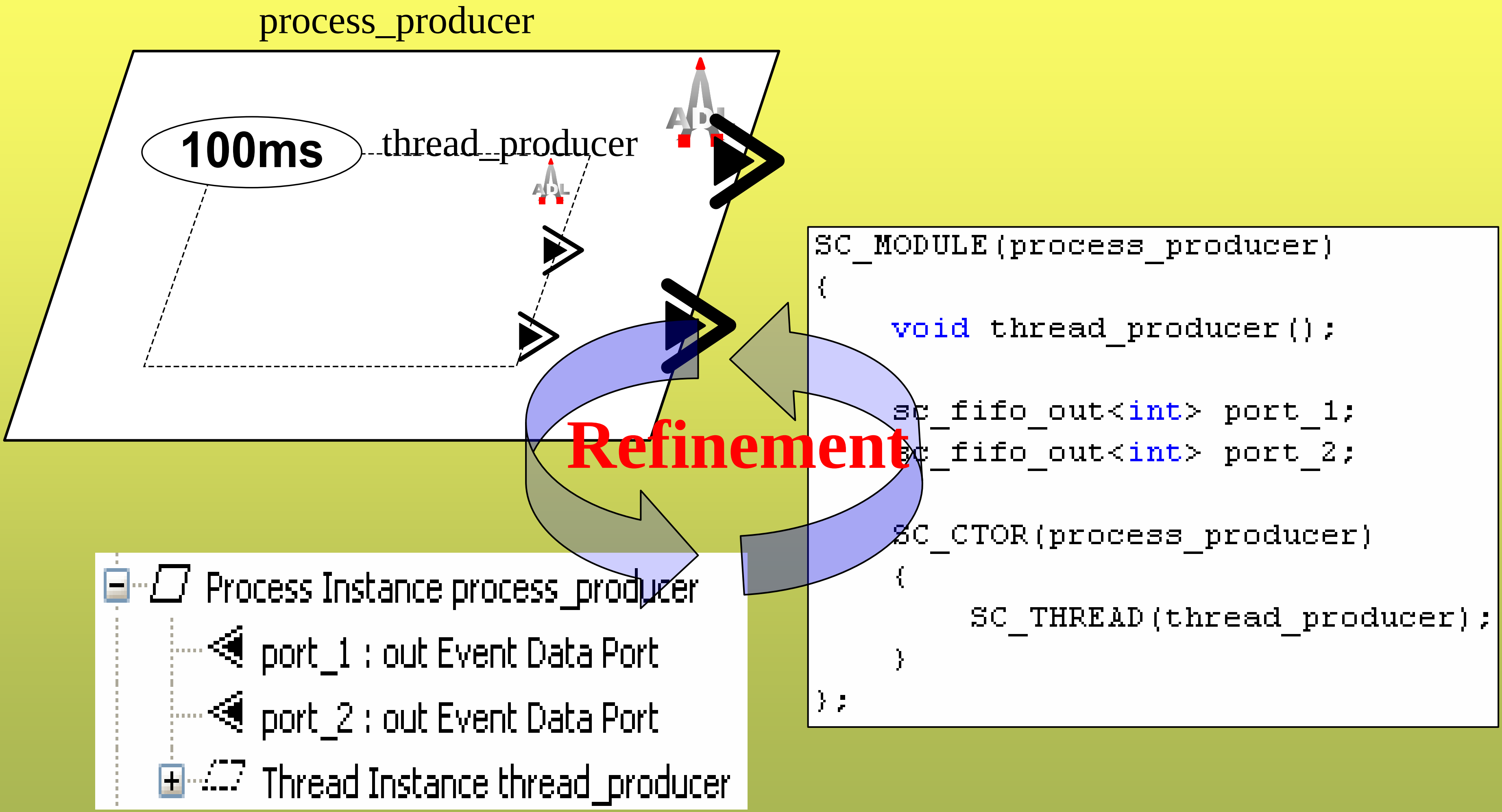
Example



- System Instance wises_example_system_example_imp
  - Process Instance process_producer
    - port_1 : out Event Data Port
    - port_2 : out Event Data Port
    - Thread Instance thread_producer
      - port_1 : out Event Data Port
      - port_2 : out Event Data Port
  - Process Instance process_consumer
    - port_1 : in Event Data Port
    - port_2 : in Event Data Port
    - Thread Instance thread_consumer
      - Properties
      - port_1 : in Event Data Port
      - port_2 : in Event Data Port

```
SC_MODULE(wises_example_system_example_impl_Instance)
{
    process_producer_0 *process_producer;
    process_consumer_0 *process_consumer;

    sc_fifo<int> connection_1;
    sc_fifo<int> connection_2;


    SC_CTOR(wises_example_system_example_impl_Instance)
    {
        process_producer = new process_producer_0("process_producer");
        process_consumer = new process_consumer_0("process_consumer");

        process_producer->conexion_1(connection_1);
        wises_example_system_example_impl_Instance->connection_1(connection_1);
        process_producer->conexion_2(connection_2);
        wises_example_system_example_impl_Instance->connection_2(connection_2);
    }
};
```

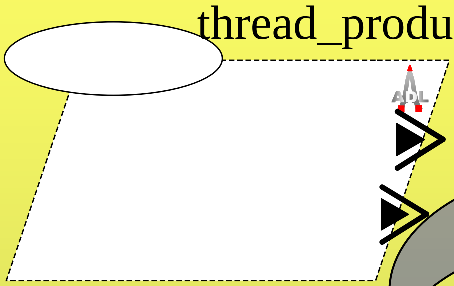
Example



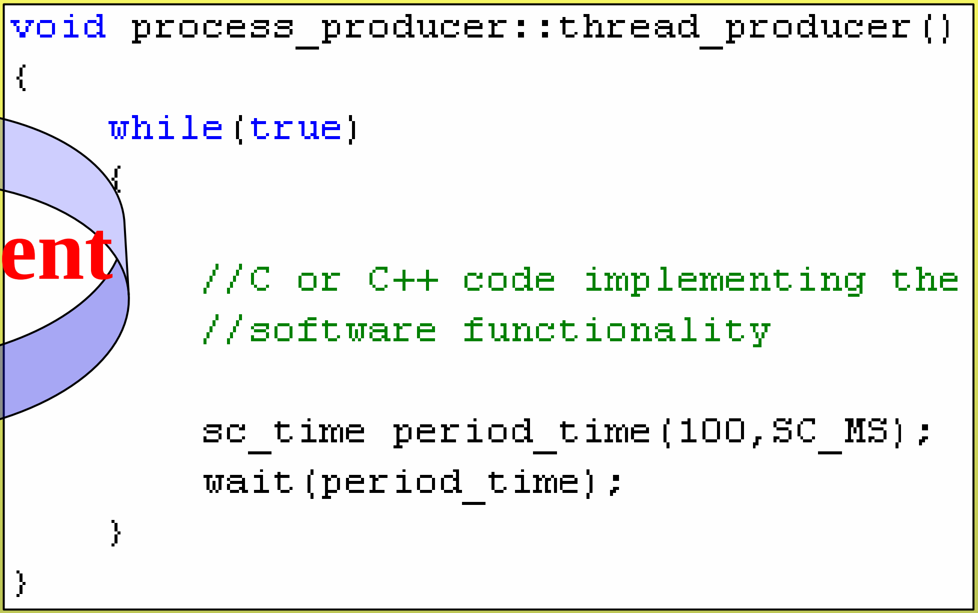
```
SC_MODULE(process_producer)
{
    void thread_producer();

    sc_fifo_out<int> port_1;
    sc_fifo_out<int> port_2;

    SC_CTOR(process_producer)
    {
        SC_THREAD(thread_producer);
    }
};
```

- Process Instance process_producer
  - port_1 : out Event Data Port
  - port_2 : out Event Data Port
  - Thread Instance thread_producer

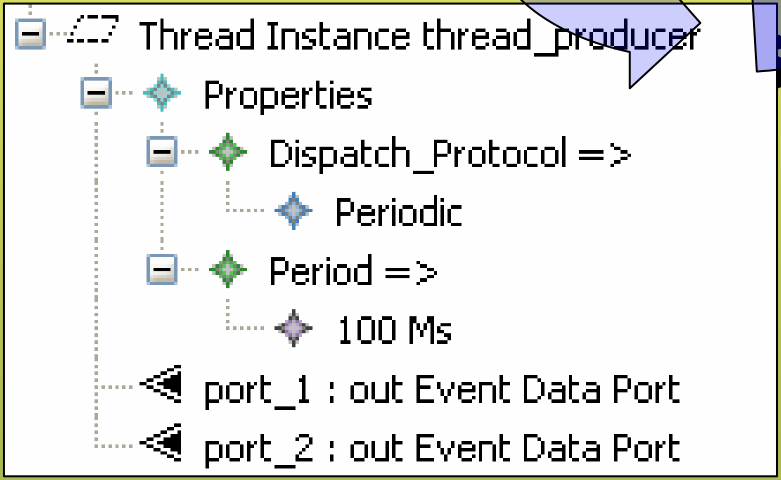
Example



Refinement



```
void process_producer::thread_producer()
{
    while(true)
    {

        //C or C++ code implementing the
        //software functionality

        sc_time period_time(100,SC_MS);
        wait(period_time);
    }
}
```



- Thread Instance thread_producer
  - Properties
    - Dispatch_Protocol =>
      - Periodic
    - Period =>
      - 100 Ms
  - port_1 : out Event Data Port
  - port_2 : out Event Data Port

Conclusions

- SystemC allows modeling AADL
  - Different abstraction levels.
  - Refinement
  - Validation
- Specification for model transformation from AADL to SystemC
- Tool proposal for embedded system design

END

THANK YOU FOR YOUR ATTENTION

QUESTIONS ?