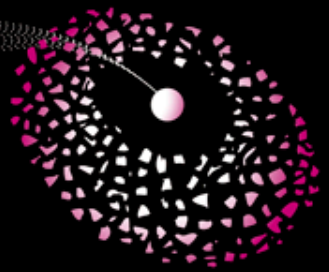
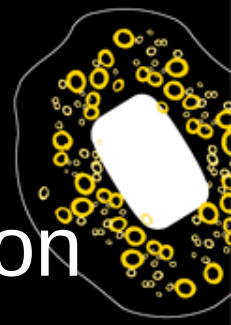
UNIVERSITY OF TWENTE.

Model-Driven Design of Simulation Support for the TERRA Robot Software Tool Suite

Zhou Lu, Maarten Bezemer, Jan Broenink

Robotics & Mechatronics, University of Twente



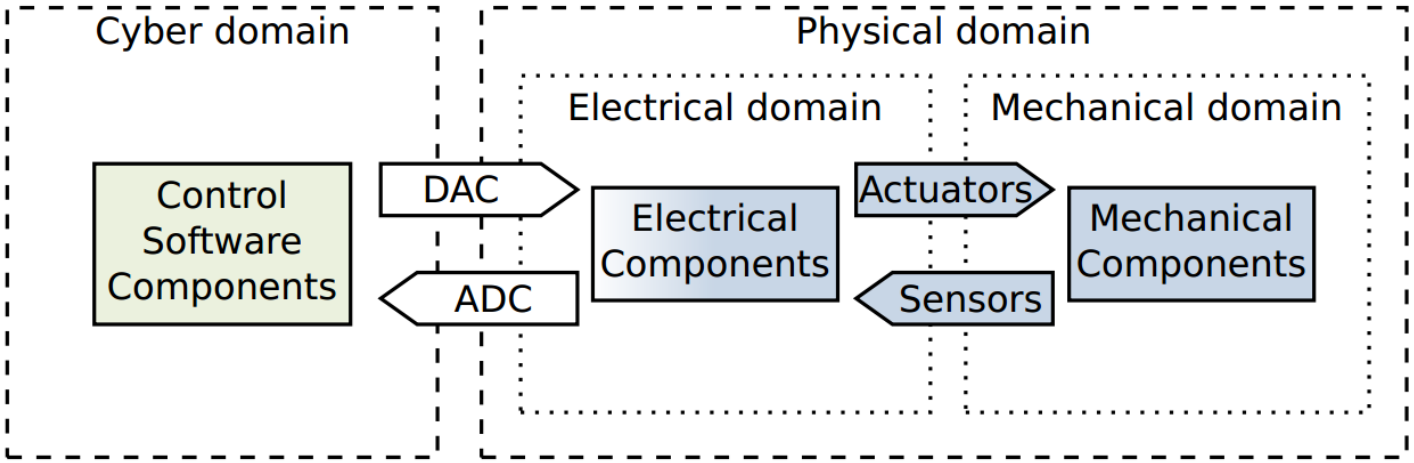


Outlines

- Introduction
  - Context
  - Motivation
  - Goals
- Background
  - Modeling Approach
  - Existing Tool Suite
- Simulation Support for TERRA
  - Design Methodology
  - Simulation Meta-model
  - Implementation
- Example
- Conclusion and Future Work

Introduction – Context

- Cyber-Physical System
  - CPS: cyber domain and physical domain are closely integrated
  - Most of them are safety-critical systems
  - Design challenges of CPS: complex design spaces, too costly
    - Joint dynamics of software, networks, and physical processes, etc.
    - Operate dependably, safely, efficiently and in real-time w.r.t. requirements
    - Gap, between C&P parts when integrating in the end of the design phase



(M.M.Bezemer, PhD Thesis)

Introduction – Motivation

- Concurrency in CPS
  - Concurrency is intrinsically present in CPS
    - Physical: compositions of many things occurring at the same time
    - Cyber: measuring and controlling the dynamics of physical processes
- Model-Driven Development
  - Model transformation – formal verification
  - Meta-model – reusability and extensibility
  - Off-target simulation and testing, first-time-right implementation
- Communicating Sequential Processes
  - Solution for formalization and concurrency issues in CPS
- Simulation needs
  - Precise modeling for C&P parts are required in CPS design
  - FDR: check the software architecture problems
  - The executing order of processes is required as well

Introduction – Goals

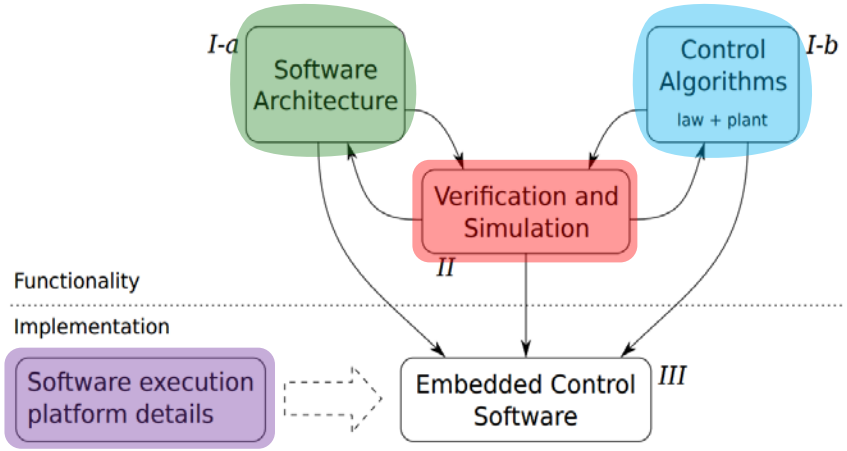
- Project Goals
  - Develop methods and tools for model-driven C&P co-design
    - Integrally treat C&P parts, co-modeling and co-simulation
    - Eliminate the gap between C&P parts during the early design phase
    - Achieve first-time-right designs

- Our work reported in this paper
  - Design and develop simulation support for the TERRA tool suite
    - Make the TERRA model ‘executable’
    - Determine the order of software processes

Background – Modeling Approach

UNIVERSITY OF TWENTE.

- TERRA
  - Graphically CSP-based modeling for discrete-event domain [CPA2012]
- 20-sim
  - Graphically modeling for continuous-time domain
    - Control algorithms
    - Plant dynamics
- LUNA
  - C++ execution framework for QNX/Xenomai [CPA2011]
- Simulation/Co-simulation

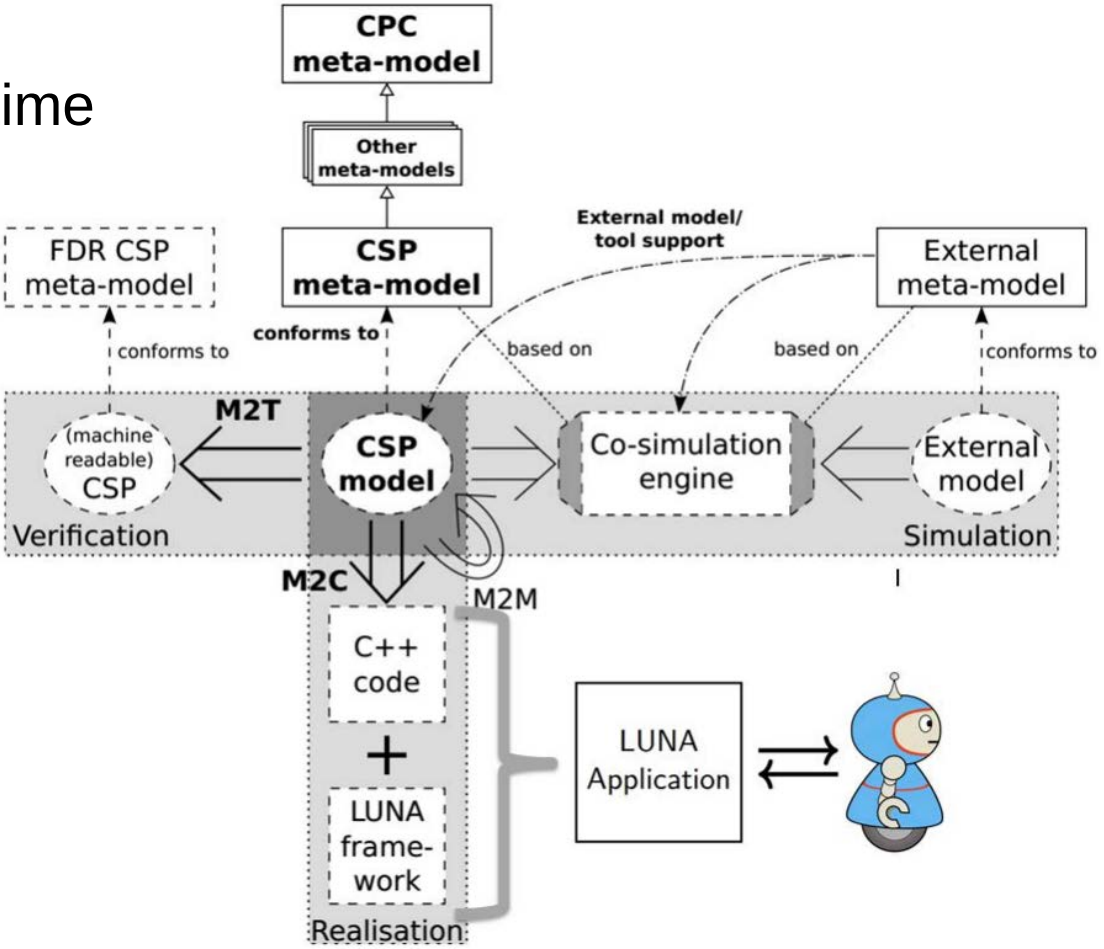


(M. M. Bezemer, PhD thesis)

Background – Existing Tool Suite

UNIVERSITY OF TWENTE.

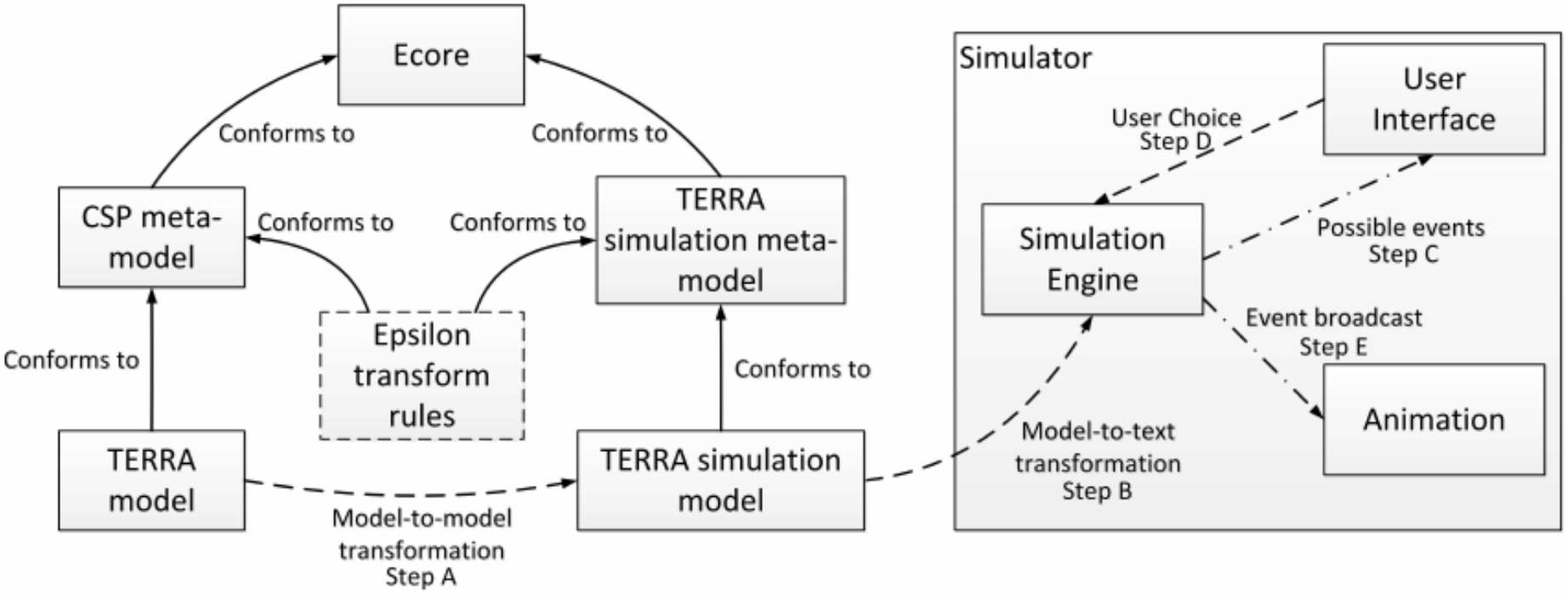
- TERRA
  - Twente Embedded Real-time Robot Application
- Eclipse-based tool suite
  - EMF
    - Meta-model design
      - Explicit CSP meta-model
  - Epsilon Framework
    - Model transformation
    - Code generation
    - Model validation



(M. M. Bezemer, PhD thesis)

- Design methodology
  - Based on
    - MDE
    - Meta-models
    - EMF, Epsilon ETL, EGL

A: Transform TERRA model to simulation model
B: Generate simulation trace text
C: Present status to user
D: Simulate by user choice
E: Broadcast event for animation



- Simulation meta-model

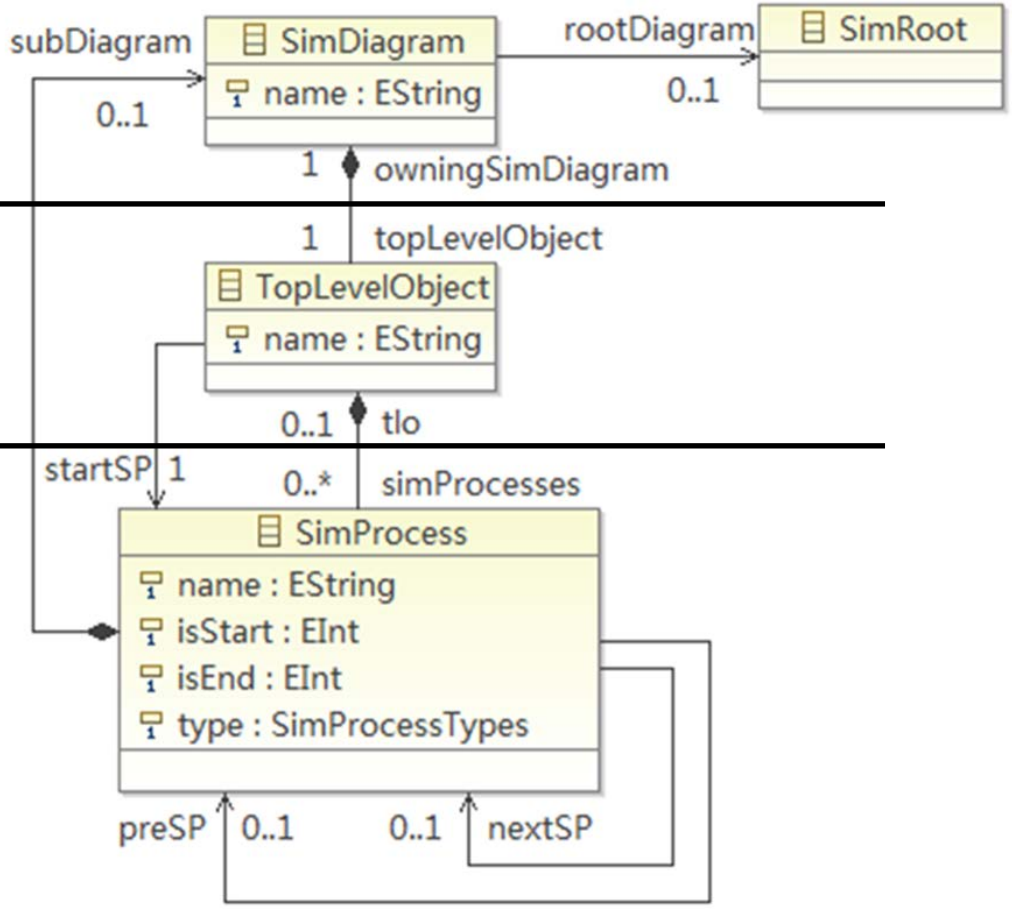
CSP Meta-model

CSPDiagram

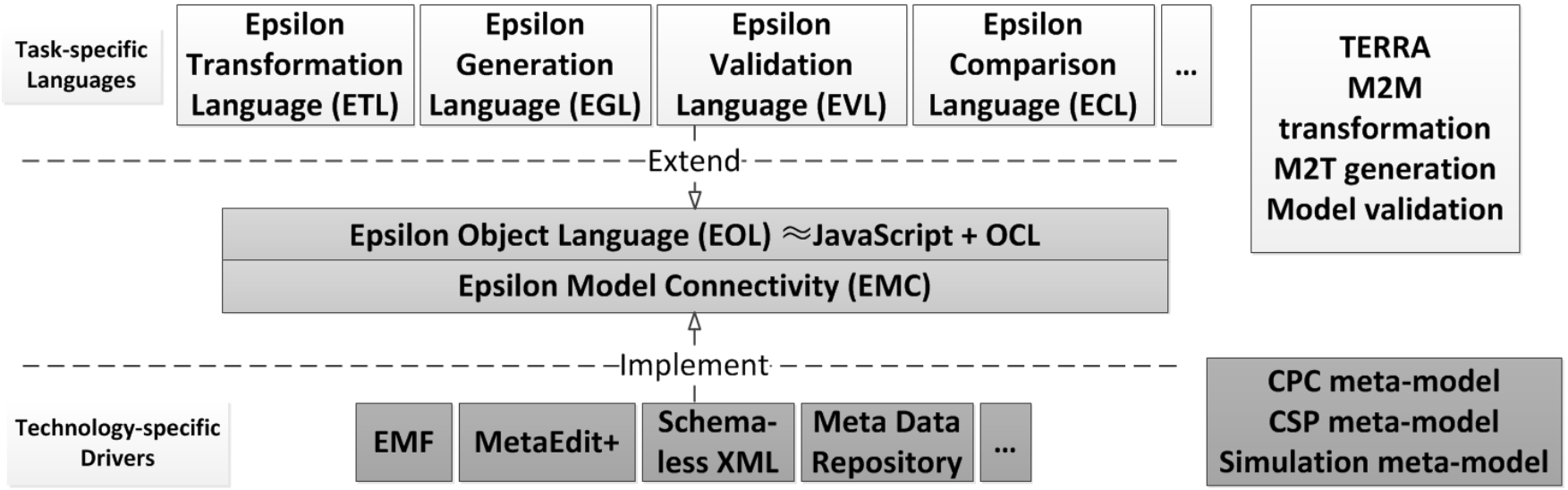
CSPCompositionalGroup
CSPModel
CSPWriter
CSPReader

CSPModel
CSPWriter
CSPReader
CSPCompositionalGroup

Simulation Meta-model



- Implementation

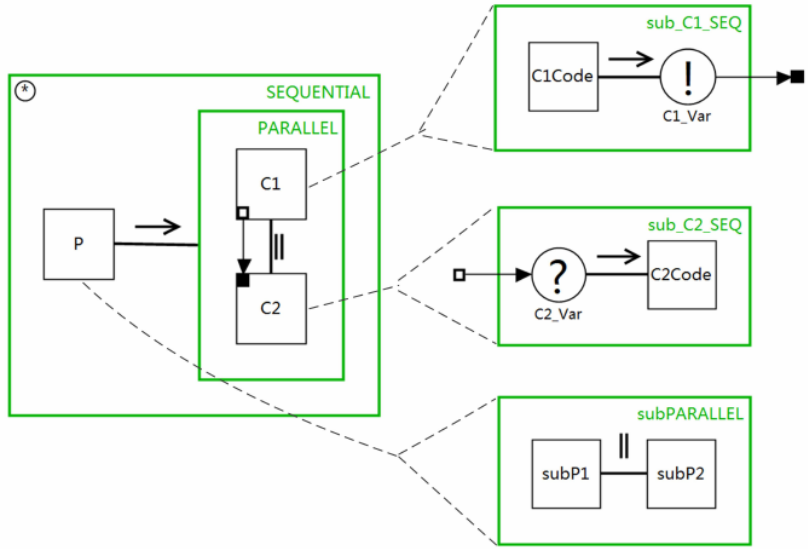


- Why adding a new simulation meta-model?
  - No conflict/influence with/to the existing CSP meta-model
  - Simplified abstraction of a TERRA CSP model, w.r.t simulation needs
  - More flexibility, extensibility

Example

- TERRA model example

Source TERRA model

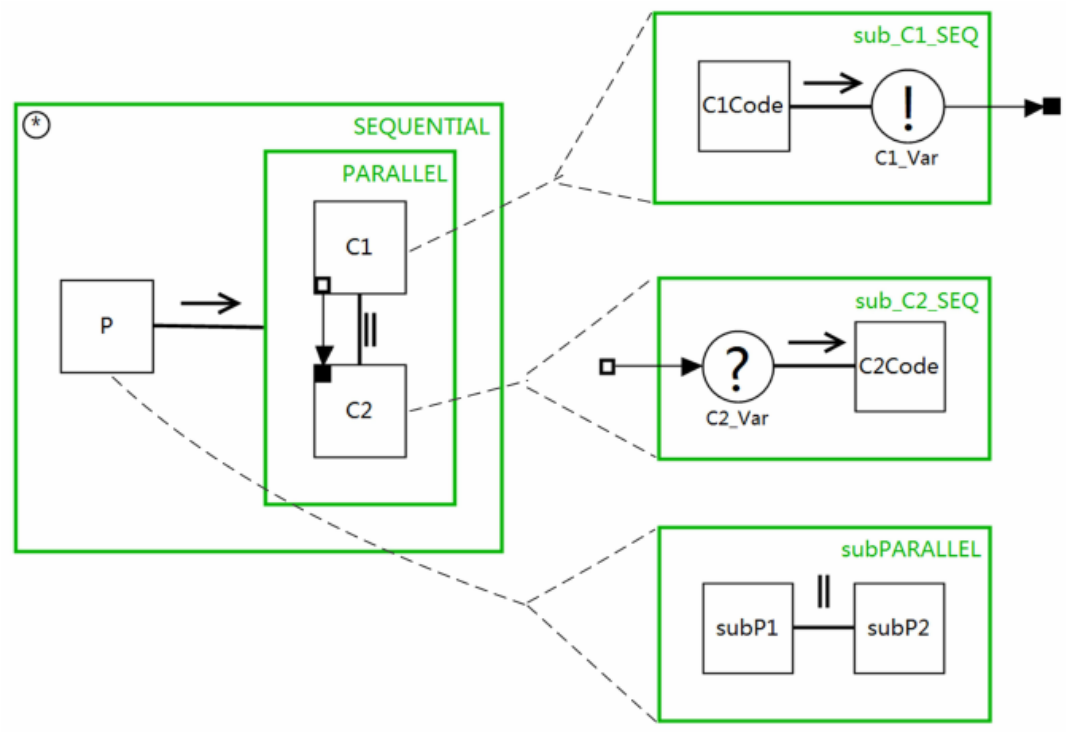


Generated CSPm

```
-- Channels
channel channel_C1_Wr_C1_to_C2_Rd_C2

-- Processes
MainModel_SEQUENTIAL = if (true) then
(MainModel_P ; MainModel_PARALLEL ) ; MainModel_SEQUENTIAL else SKIP

MainModel_P = P_subPARALLEL
P_subPARALLEL = P_subP1 ||| P_subP2
P_subP1 = SKIP
P_subP2 = SKIP

MainModel_PARALLEL =
MainModel_C1 [| {| channel_C1_Wr_C1_to_C2_Rd_C2 |} |] MainModel_C2
MainModel_C1 = C1_sub_C1_SEQ
C1_sub_C1_SEQ = C1_C1Code ; C1_Wr_C1
C1_C1Code = SKIP
C1_Wr_C1 = channel_C1_Wr_C1_to_C2_Rd_C2 -> SKIP
MainModel_C2 = C2_sub_C2_SEQ
C2_sub_C2_SEQ = C2_Rd_C2 ; C2_C2Code
C2_Rd_C2 = channel_C1_Wr_C1_to_C2_Rd_C2 -> SKIP
C2_C2Code = SKIP
```

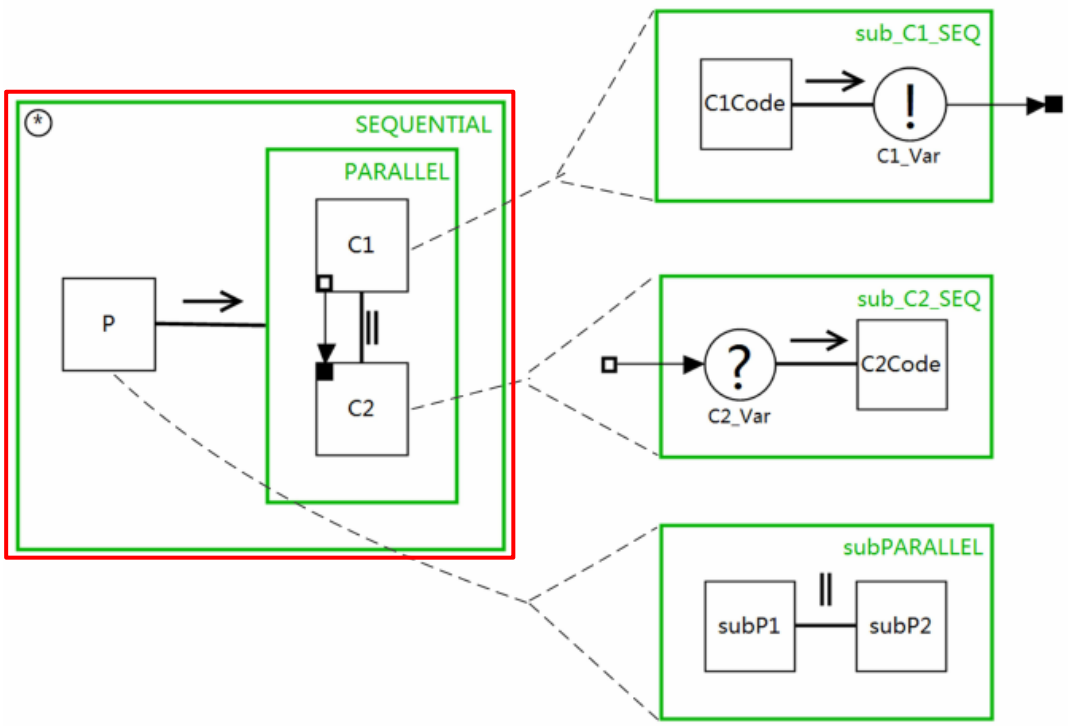
Example

- Transformed to a simulation model



Example

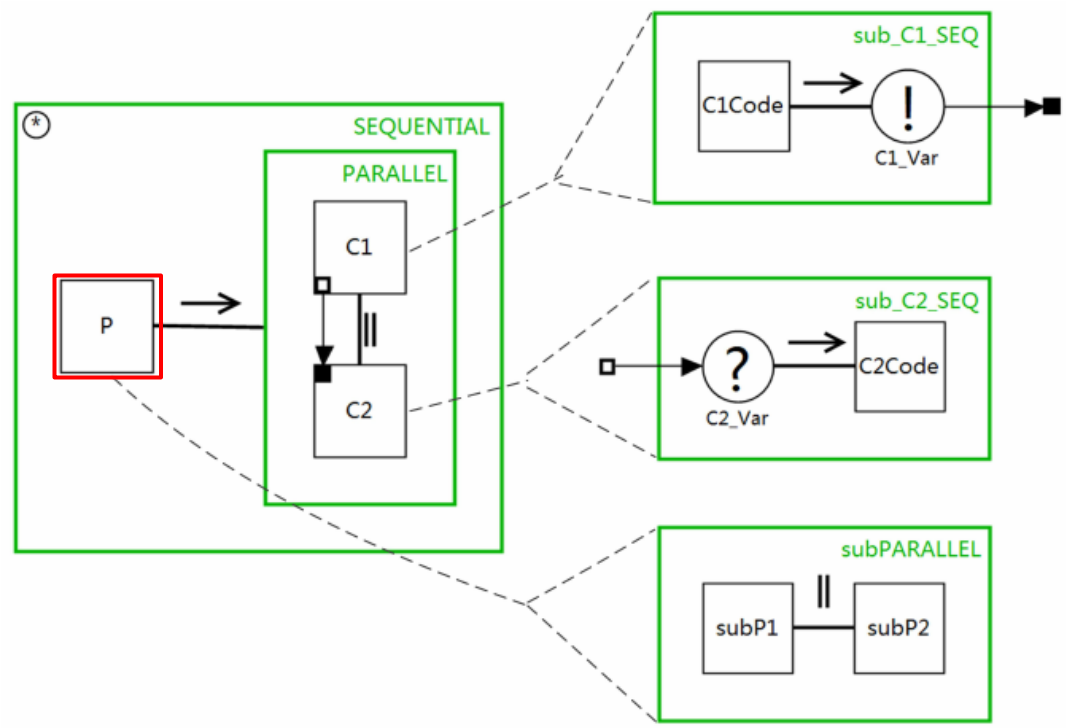
- Transformed to a simulation model



Is End	0
Is Start	1
Name	SEQUENTIAL
Next SP	Sim Process P
Pre SP	
Tlo	Top Level Object SEQUENTIAL
Type	Group

Example

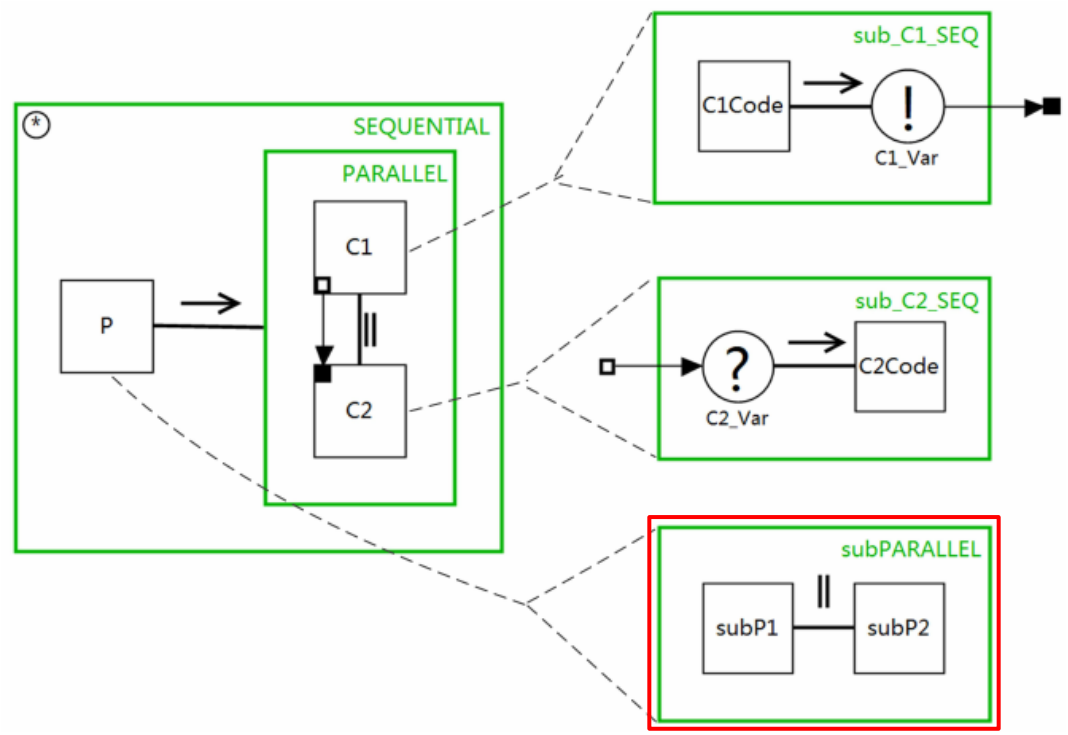
- Transformed to a simulation model



Is End	0
Is Start	0
Name	P
Next SP	Sim Process subPARALLEL
Pre SP	Sim Process SEQUENTIAL
Tlo	Top Level Object SEQUENTIAL
Type	Process

Example

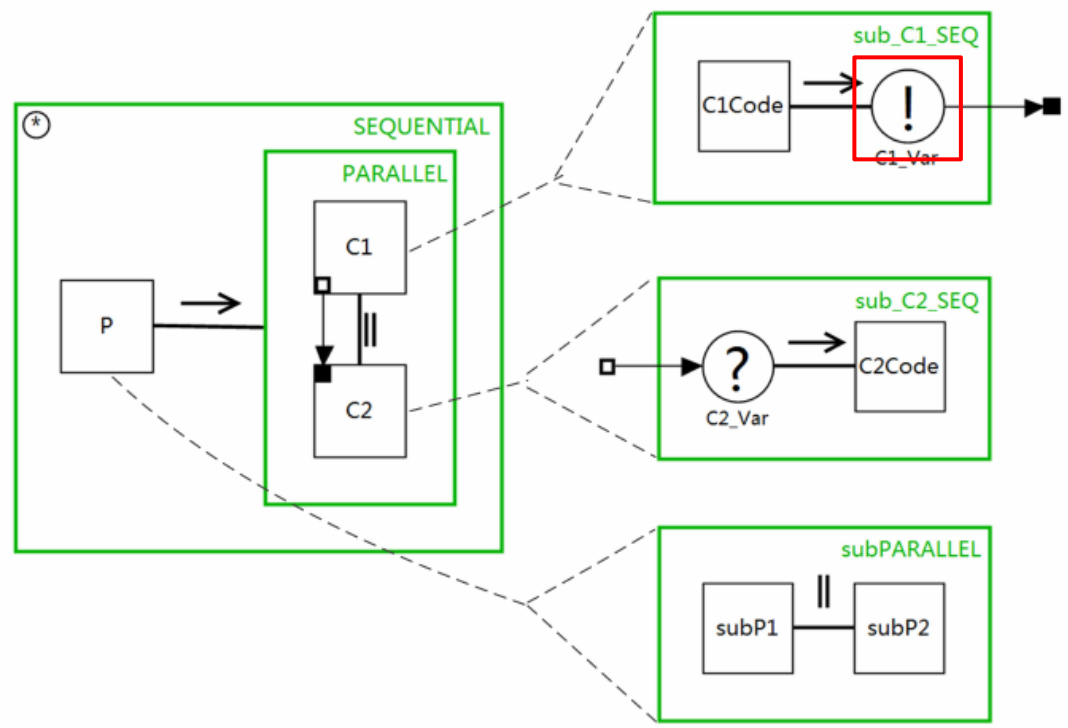
- Transformed to a simulation model



Is End	0
Is Start	1
Name	subPARALLEL
Next SP	Sim Process subP1
Pre SP	Sim Process P
Tlo	Top Level Object subPARALLEL
Type	Group

Example

- Transformed to a simulation model



Is End	1
Is Start	0
Name	Wr_C1
Next SP	Sim Process C2
Pre SP	Sim Process subP1
Tlo	Top Level Object sub_C1_SEQ
Type	Writer

Example

- Generated simulation trace text
  - Indicate the execution order of processes
  - Random selection

```
-- Generated by TERRA SIM to simTrace version 0.0.4
-- Input file: test.sim

-- Execution Queue
SimDiagram = MainModel
Top Level Object = SEQUENTIAL
SEQUENTIAL is Recursive
SEQUENTIAL -> Start SP = SEQUENTIAL
SEQUENTIAL -> Next SP = P

SimDiagram = P
Top Level Object = subPARALLEL
subPARALLEL -> Start SP = subPARALLEL
subPARALLEL -> Next SP = subP1
subP1 -> Next SP = subP2
subP2 = isEnd

SimDiagram = C1
Top Level Object = sub_C1_SEQ
sub_C1_SEQ -> Start SP = sub_C1_SEQ
sub_C1_SEQ -> Next SP = C1Code
C1Code -> Next SP = Wr_C1
Wr_C1 = isEnd

SimDiagram = C2
Top Level Object = sub_C2_SEQ
sub_C2_SEQ -> Start SP = sub_C2_SEQ
sub_C2_SEQ -> Next SP = Rd_C2
Rd_C2 -> Next SP = C2Code
C2Code = isEnd
```

Example

- Generated simulation trace text
  - Indicate the execution order of processes
  - Random selection

```
SimDiagram = P
Top Level Object = subPARALLEL
subPARALLEL -> Start SP = subPARALLEL
subPARALLEL -> Next SP = subP2
subP2 -> Next SP = subP1
subP1 = isEnd
```

```
operation CSPCompositionalGroup getGroupedObjects() : Sequence {
  var objects: Sequence;
  for(relation in self.relations) {
    for(object in relation.compositeObjects) {
      objects.add(object);
    }
  }
    if(self.getGroupSymbol() =="||" and Sequence{1..2}.random()==1) {
    return objects.invert();
  }
  else {
    return objects;
  }
}
```

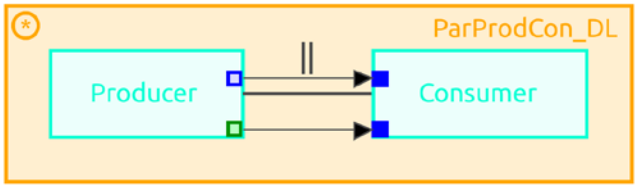
```
-- Generated by TERRA SIM to simTrace version 0.0.4
-- Input file: test.sim

-- Execution Queue
SimDiagram = MainModel
Top Level Object = SEQUENTIAL
SEQUENTIAL is Recursive
SEQUENTIAL -> Start SP = SEQUENTIAL
SEQUENTIAL -> Next SP = P

SimDiagram = P
Top Level Object = subPARALLEL
subPARALLEL -> Start SP = subPARALLEL
subPARALLEL -> Next SP = subP1
subP1 -> Next SP = subP2
subP2 = isEnd

SimDiagram = C1
Top Level Object = sub_C1_SEQ
sub_C1_SEQ -> Start SP = sub_C1_SEQ
sub_C1_SEQ -> Next SP = C1Code
C1Code -> Next SP = Wr_C1
Wr_C1 = isEnd

SimDiagram = C2
Top Level Object = sub_C2_SEQ
sub_C2_SEQ -> Start SP = sub_C2_SEQ
sub_C2_SEQ -> Next SP = Rd_C2
Rd_C2 -> Next SP = C2Code
C2Code = isEnd
```

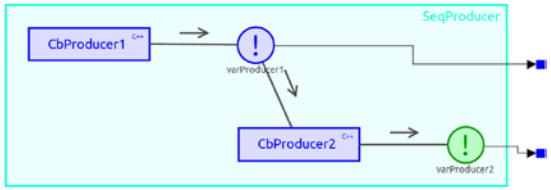
Conclusion and future work

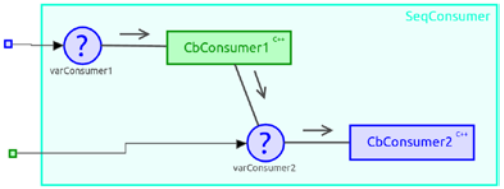
- ☑ Sim meta model
- ☑ TERRA2Sim transformation
- ☑ Produce simulation trace text
- ☑ Show process orders
- ☑ Show random selections
- ? Graphical animation
- ? Show computations
- ? Co-simulation interface (between TERRA & 20-sim)

Conclusion and future work

- ☑ Sim meta model
- ☑ TERRA2Sim transformation
- ☑ Produce simulation trace text
- ☑ Show process orders
- ☑ Show random selections
- ? Graphical animation
- ? Show computations
- ? Co-simulation interface (between TERRA & 20-sim)







Thanks!