Process Algebra to Model Smart Distributed Mobile Real-Time IoT

Implementation Perspective for Cyber-Physical Systems

20 Sept 2019

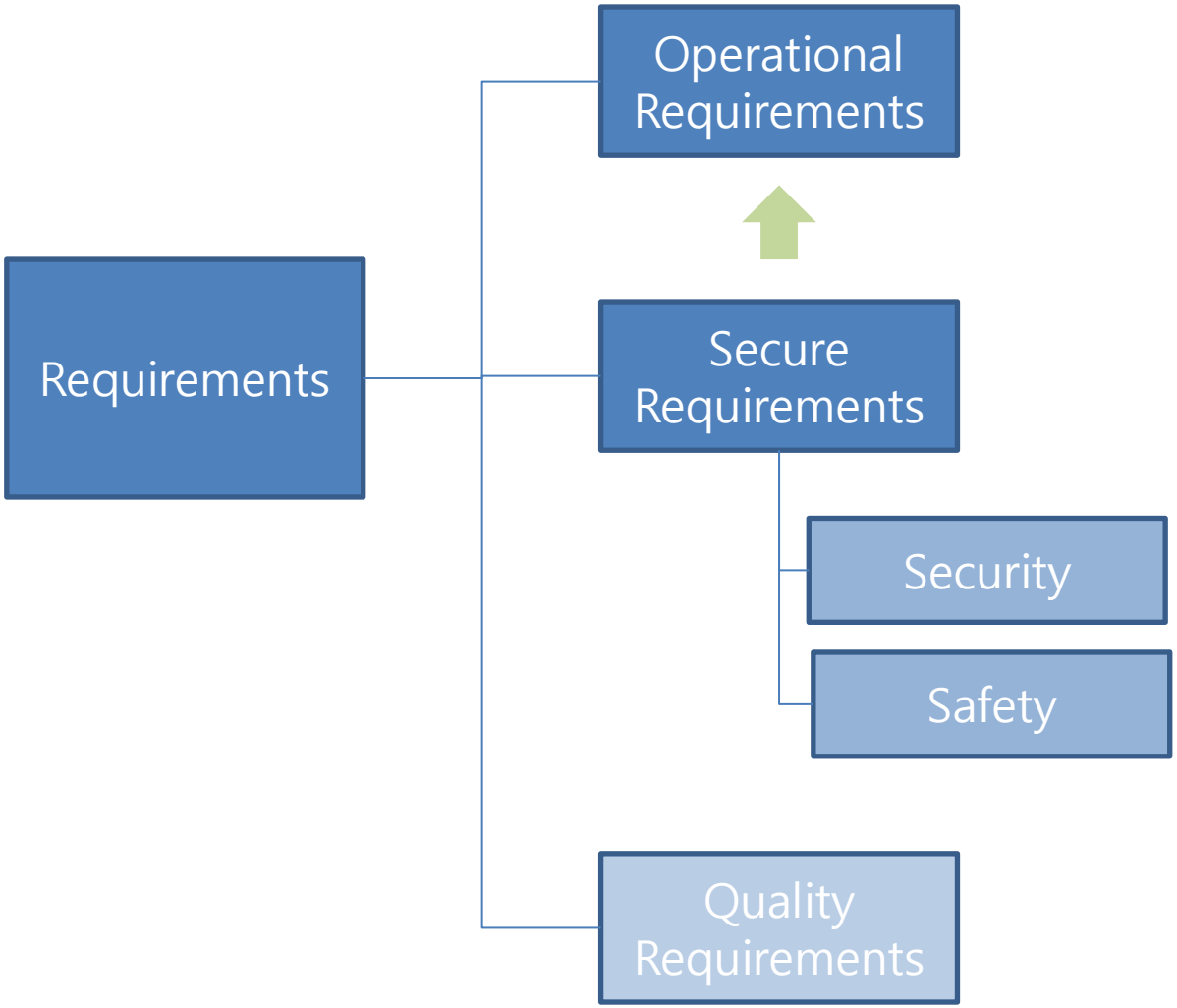
Prof. Moonkun Lee, Ph.D.

Division of Computer Science and Engineering
Chonbuk National University
Republic of Korea

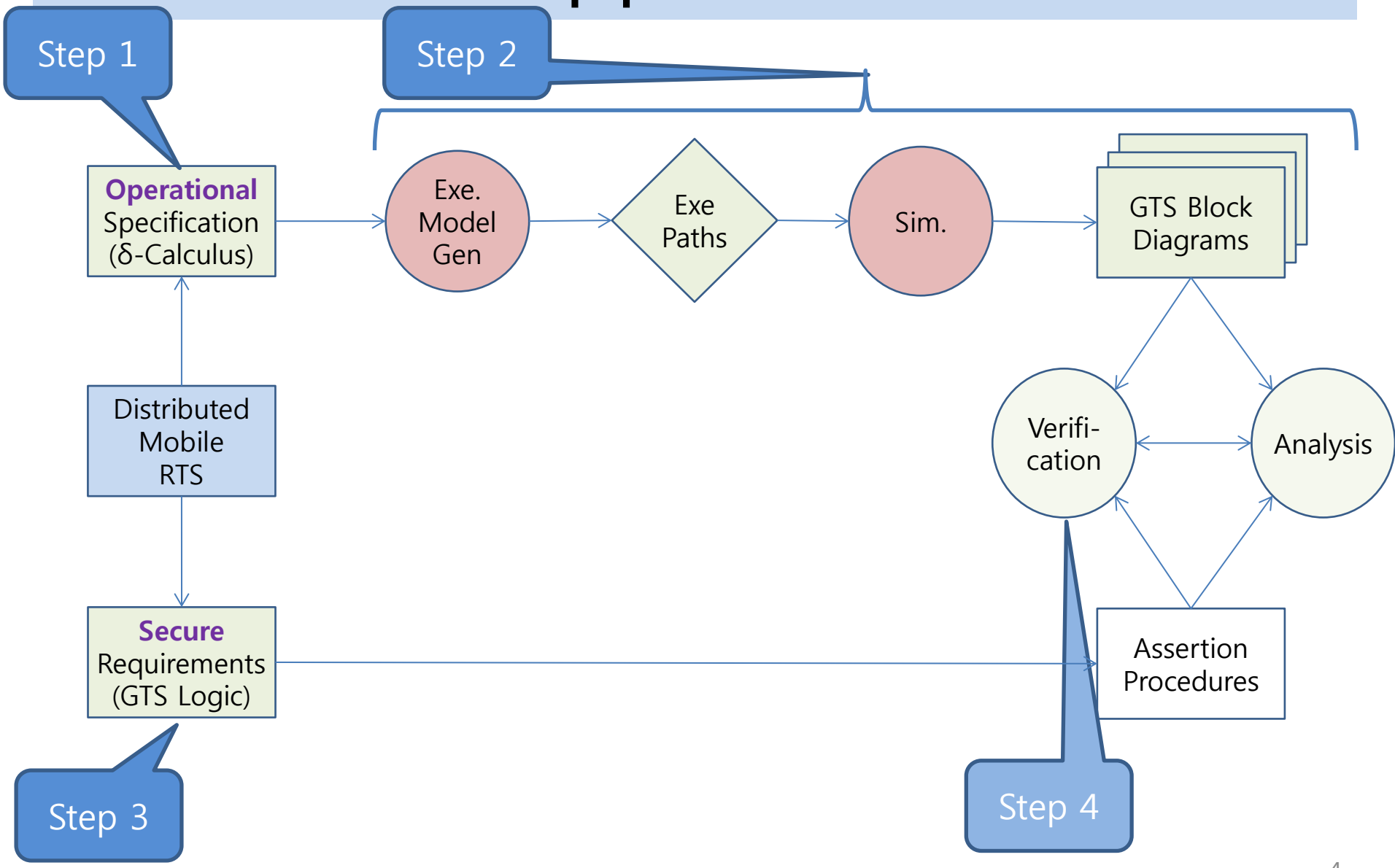
1. Motivation
2. **Modeling**
    1) Specification: δ-Calculus
    2) Verification: GTS Logic
    3) EMS Example
3. SAVE Tool
4. Cyber-Physical Systems Application
5. Summary
6. Discussion

2. MODELING: SPECIFICATION & VERIFICATION

Dual Approach

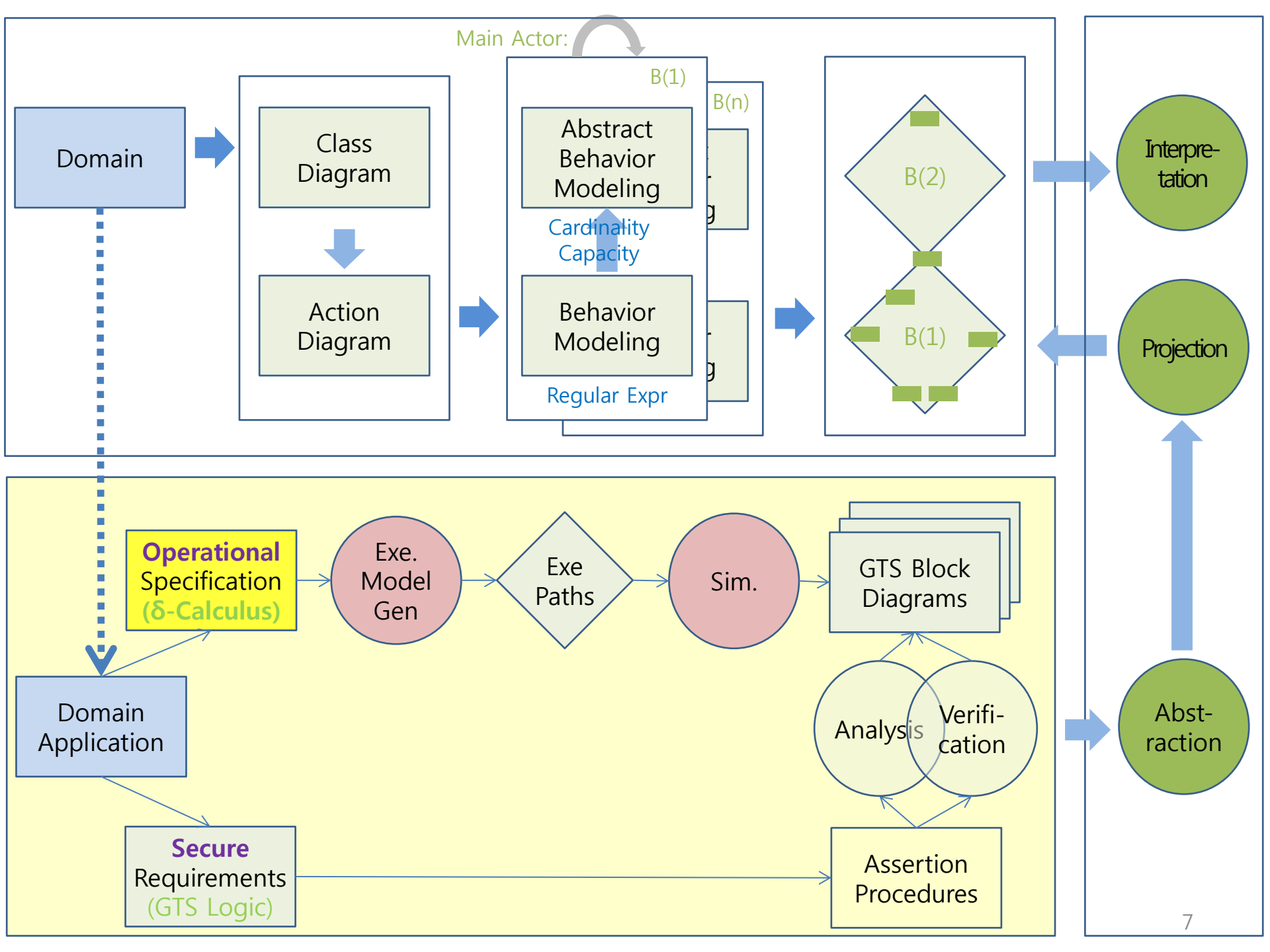


Approach


Step 1
Step 2
Operational Specification (δ-Calculus)
Exe. Model Gen
Exe Paths
Sim.
GTS Block Diagrams
Distributed Mobile RTS
Verifi-cation
Analysis
Secure Requirements (GTS Logic)
Assertion Procedures
Step 3
Step 4

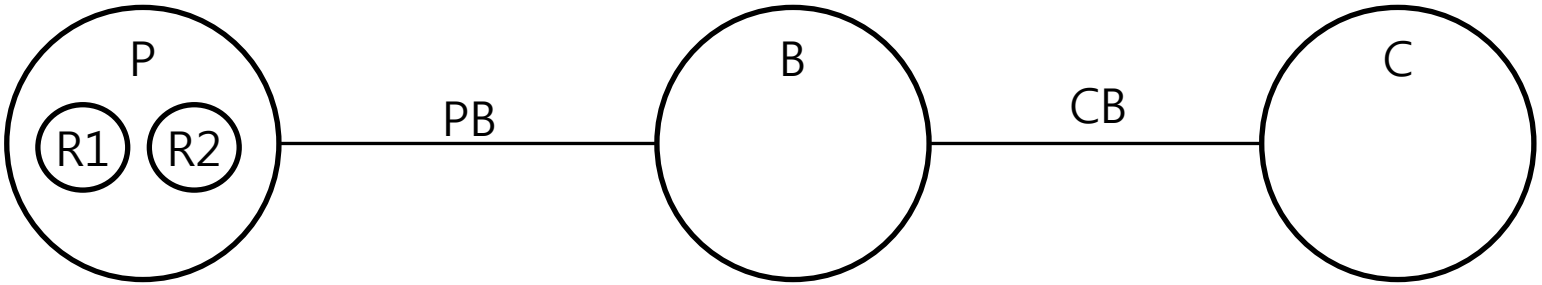
2.1 δ-Calculus: Operational Specification

Step 1
Operational Requirements


Main Actor:
Domain
Class Diagram
Action Diagram
B(1)
B(n)
Abstract Behavior Modeling
Cardinality Capacity
Behavior Modeling
Regular Expr
B(2)
B(1)
Interpre-tation
Projection
Abst-raction
Operational Specification (δ-Calculus)
Exe. Model Gen
Exe Paths
Sim.
GTS Block Diagrams
Domain Application
Analysis
Verifi-cation
Secure Requirements (GTS Logic)
Assertion Procedures

Requirements: PBC Example

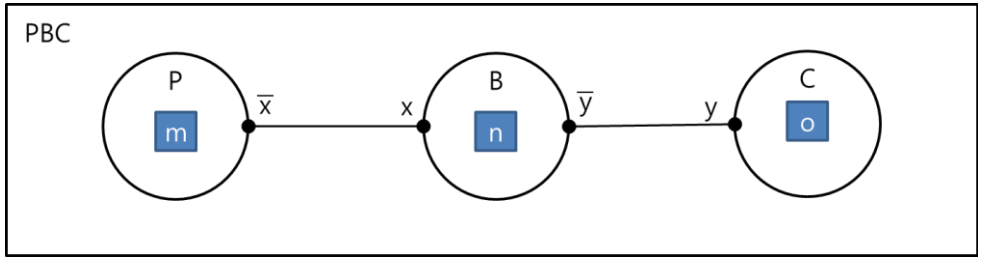
- **Operational Requirements**
  - *Producer* produces two resources, R1 and R2.
  - *Producer* stores the resources in *Buffer* in order.
  - *Producer* informs *Buffer* of the order of R1 and R2, or R2 and R1.
  - *Consumer* consumes the resources from *Buffer* in order.
  - The order of the consumption is informed to *Buffer* by *Consumer*.
- Secure Requirements
  - Security:
    - The order should not be violated, since the first resource contains security information to decode the second resource.
    - The propagation between the first and the second should be less than 30 seconds.
  - Safety
    - The resources produced by *Producer* should be consumed by *Consumer* less than 5 minutes.



Specification: Process Algebra

$$1+2+3=1+(2+3)$$
$$=1+(3+2)$$
$$=1+5$$
$$=(1+5)$$
$$=6$$

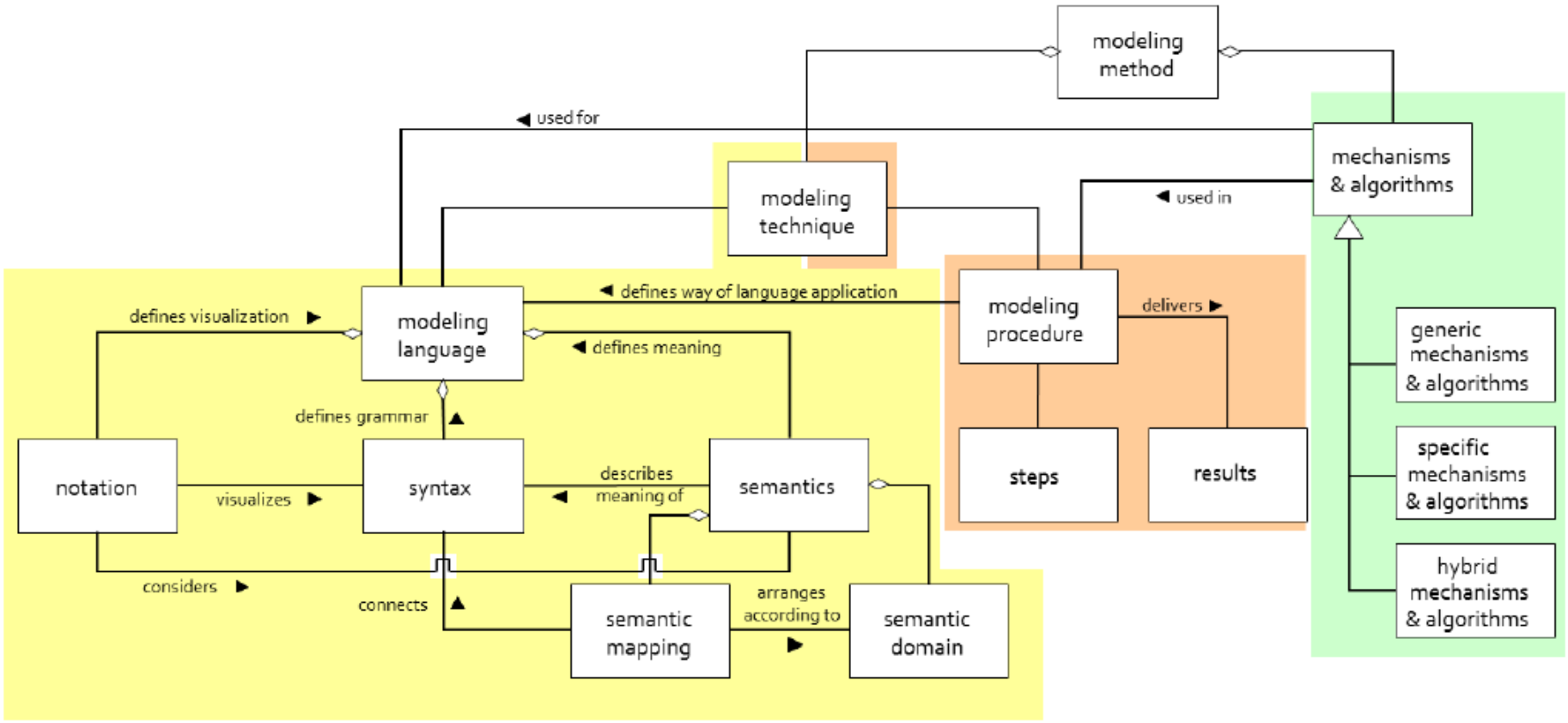
- Specification:
  - $PBC \overset{\text{def}}{=} P|C|B$
  - $P \overset{\text{def}}{=} \bar{x}(m).P$
  - $B \overset{\text{def}}{=} x(n).\bar{y}(n).B$
  - $C \overset{\text{def}}{=} y(o).C$



- Execution:
  - $PBC \overset{\text{def}}{=} P|C|B$ (Commutative Law)
    $= P|B|C$
    $= (\bar{x}(m).P\ |x(m/n).\bar{y}(m/n).B)\ |y(o).C$ (Associative Law)
    $\rightarrow \quad P\ |\quad (\bar{y}(m)\quad .B\ |y(m/o).C)$ (Binary; Synch)
    $\rightarrow \quad P\ |\quad B\ |\quad C$

ADOxx Meta-Modeling Platform

Method = (T+MA)·(L+P)



Karagiannis, D., Kühn, H.: „Metamodelling Platforms". In Bauknecht, K., Min Tjoa, A., Quirchmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2002, LNCS 2455, Springer, Berlin/Heidelberg, p. 182.

Process Algebra

System Representation

- System:
  - System: A set of processes.
  - Process: A sequence of actions or interactions.
  - Medium:
    - A set of ports or channels/buses for communication.
    - A set of gates or entrance for movements
- Operations
  - Process operations:
    - Parallel (|)
    - Choice (+)
    - Sequencing (;)
  - Actions:
    - Actions: Independent actions.
      - Communication
      - Movements
      - Usage of resources
    - Interactions: Dependent binary actions between two independent processes.
      - Communication
      - Movements
- Other properties;
  - Time
  - Probability: Choice (+)

Behavior Representation

- Labeled Transition System:
  - Transition rules
- Traces:
  - Process: A sequence of transitions.
  - System: A set of processes in transitions.
- Criteria:
  - Order of actions or interactions.
  - Properties related to the execution:
    - Time
    - probability
- Equivalence: Bisimulation
  - Strong bisimulation: Strong equivalence
  - Weak bisimulation: Observable equivalence

Language

- Syntax
  - Process
  - Action: Operation
    - Unary:
      - Asynchronous
    - Binary:
      - Synchronous
      - Asynchronous
- Semantics: Labelled transition systems
  - Operations: Transition rules
    - Unary:
      - Asynchronous
    - Binary:
      - Synchronous
      - Asynchronous
  - Laws
    - Operations
    - Precedency
    - Ordering
    - Equivalence

Example: CCS Syntax

$P ::= 0$	Empty process	A valid CCS process.
$\mid P_1 \mid P_2$	Parallel Composition	The process P_1 and the process P_2 exist simultaneously.
$\mid P_1 + P_2$	Choice	Proceed either as the process P_1 or the process P_2.
$\mid a.P_1$	Action Prefix	Perform action a and continue as the process P_1.
$\mid P_1[b/a]$	Renaming	All actions named a to be renamed as b.
$\mid P_1 \setminus a$	Restriction	The process P_1 without action a.
$\mid A$	Process identifier	Use A to refer the process P_1.

Example: CCS Semantics

$$\frac{\text{Premises}}{\text{Conclusion}}\,[\text{Side Conditions}]$$

Nil	No transition
Action Prefix	$\dfrac{}{a.P \xrightarrow{a} P}$
Choice	$\dfrac{P \xrightarrow{a} P',\ Q \xrightarrow{a} Q'}{P+Q \xrightarrow{a} P'}$
	$\dfrac{P \xrightarrow{a} P',\ Q \xrightarrow{a} Q'}{P+Q \xrightarrow{a} Q'}$
Restriction	$\dfrac{P \xrightarrow{a} P'}{P \setminus L \xrightarrow{a} P' \setminus L}\left[a, \bar{a} \notin L\right]$
Relabeling	$\dfrac{P \xrightarrow{a} P'}{P[f] \xrightarrow{f(a)} P'[f]}$

Parallel Composition	$\dfrac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q}$
	$\dfrac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'}$
	$\dfrac{P \xrightarrow{a} P',\ Q \xrightarrow{\bar{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$
Process Variable	$\dfrac{P \xrightarrow{a} P'}{c \xrightarrow{a} P'}\left[c \triangleq P\right]$

Example: CCS Basic Laws

Laws	
Choice: Commutative	$x + y = y + x$
Choice: Associative	$(x + y) + z = y + (x + z)$
Choice: Idempotent	$x + x = x$
Distribution of + over .	$(x + y).z = x.z + y.z$
Sequence: Associative	$(x.y).z = y.(x.z)$
Parallel: Commutative	$x \parallel y = y \parallel x$
Parallel: Associative	$(x \parallel y) \parallel z = y \parallel (x \parallel z)$

Producer-Buffer-Consumer Example

- Specification:
  - $PBC \stackrel{\text{def}}{=} P|C|B$
  - $P \stackrel{\text{def}}{=} \bar{x}(m).P$
  - $B \stackrel{\text{def}}{=} x(n).\bar{y}(n).B$
  - $C \stackrel{\text{def}}{=} y(o).C$

$$1+2+3=1+(2+3)$$
$$=1+(3+2)$$
$$=1+5$$
$$=(1+5)$$
$$=6$$

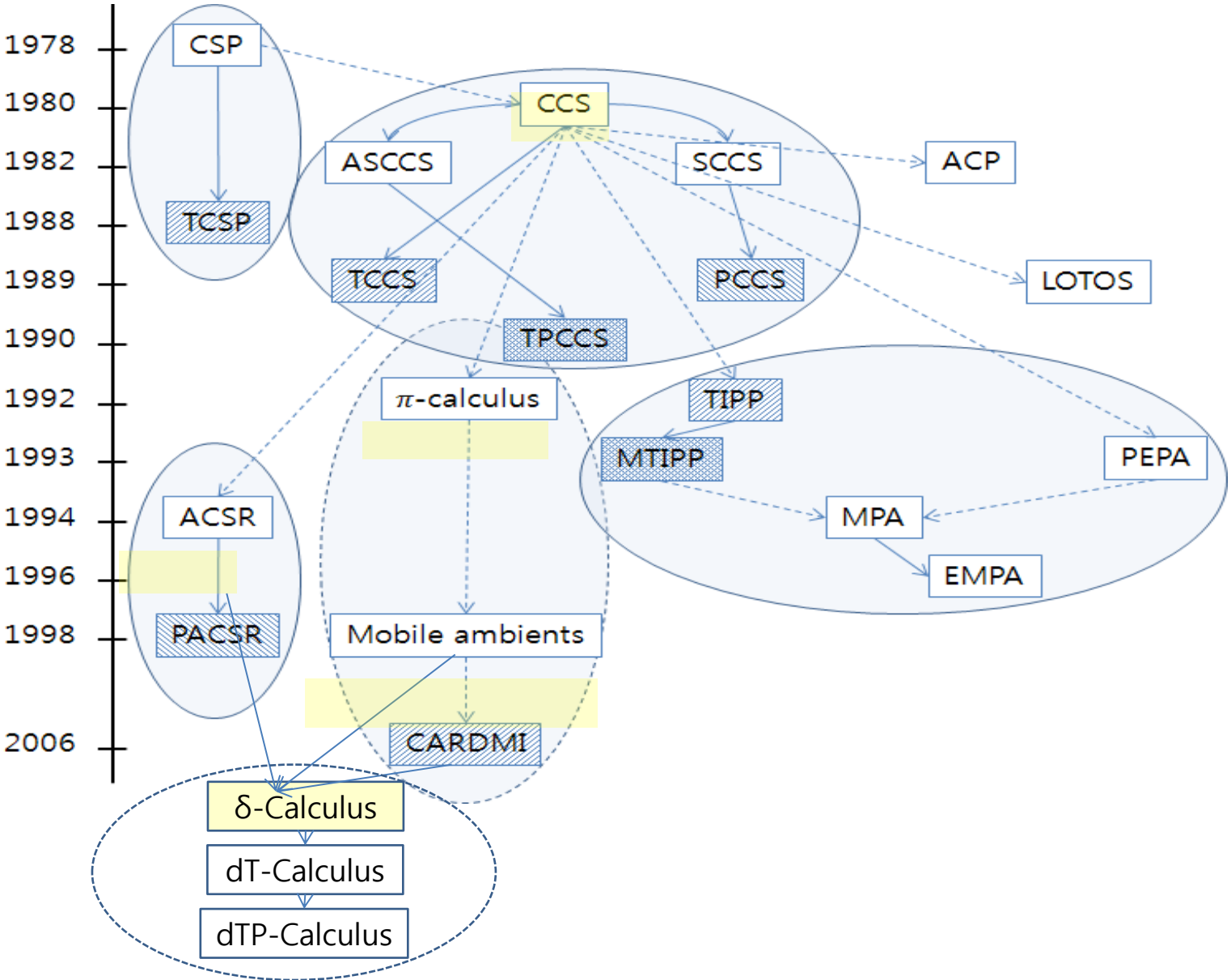
- Execution:
  - $PBC \stackrel{\text{def}}{=} P|C|B$ (Commutative Law)
    $= P|B|C$
    $= (\bar{x}(m).P \,|x(m/n).\bar{y}(m/n).B) \,|y(o).C$ (Associative Law)
    $\rightarrow \quad P \,| \quad (\bar{y}(m) \quad .B \quad |y(m/o).C)$ (Binary; Synch)
    $\rightarrow \quad P \,| \quad B \,| \quad C$

δ-Calculus

Overview

- A process calculus:
  - Based on ACSR, extended from CCS.
  - Extending its feature for movements on space.
- Purpose
  - Mathematical formalisms for describing and analyzing properties of concurrent computation with resource and priority, including movements over geo-temporal space.
- Main characteristics:
  - Space
  - Movement
  - Time
- Applications:
  - Distributed Mobile (Real-time) systems

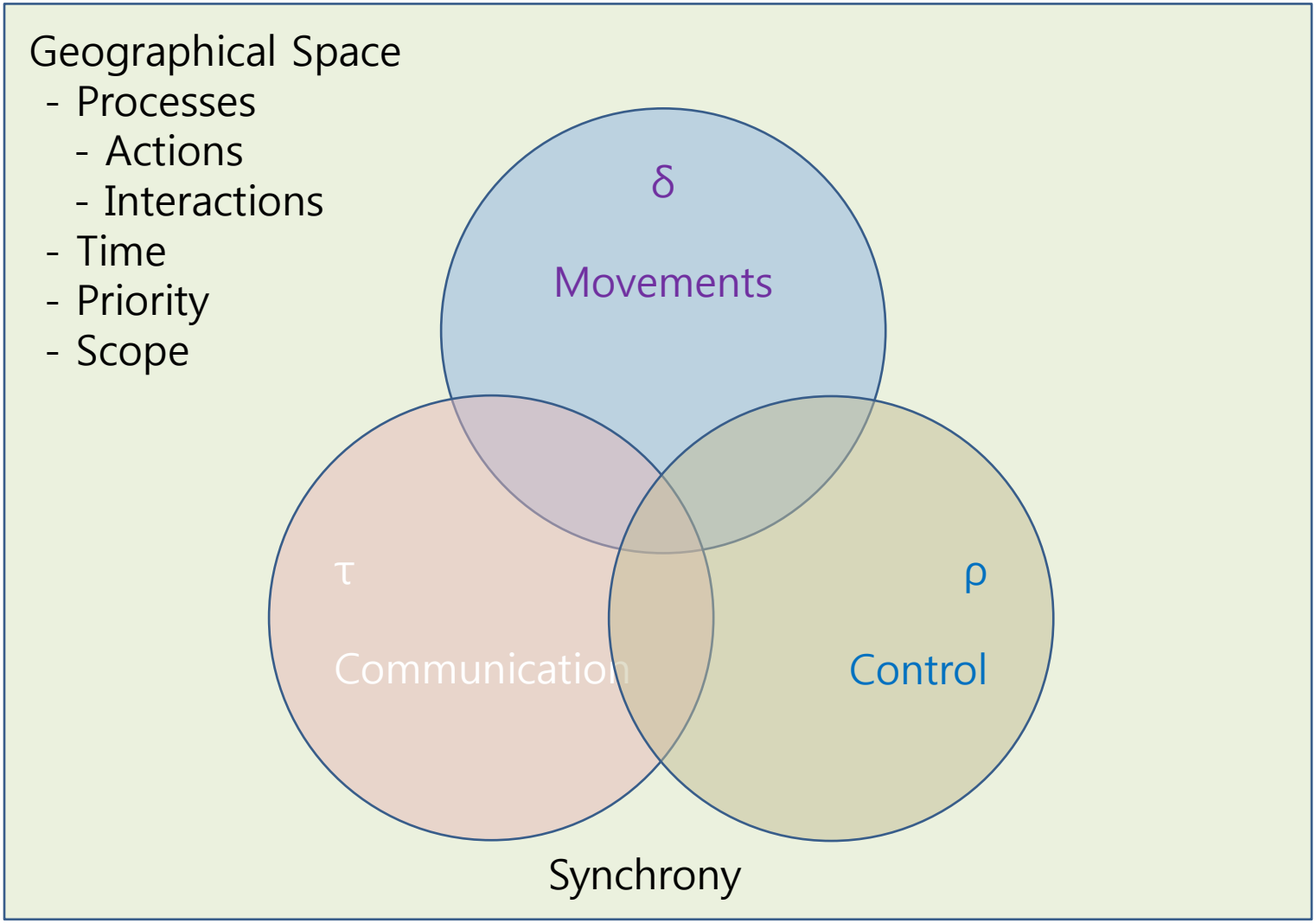
Map


1978
1980
1982
1988
1989
1990
1992
1993
1994
1996
1998
2006
CSP
TCSP
CCS
ASCCS
SCCS
ACP
TCCS
PCCS
LOTOS
TPCCS
π-calculus
TIPP
MTIPP
PEPA
MPA
EMPA
ACSR
PACSR
Mobile ambients
CARDMI
δ-Calculus
dT-Calculus
dTP-Calculus

Main Characteristics

Process Algebra	Features
CCS	• Concurrency • Communication (τ)
π-Calculus	+ Channel Passing
Mobile Ambient	+ Asynchronous Movement
ACSR	+ Resource + Time + Priority
δ-Calculus	++ Space ++ Synchronous Movement (δ) ++ Synchronous Control (ρ)

δ-Calculus: Overview



System & Process

- System $\mathcal{S} = \langle \mathcal{P}, \mathcal{I}, \mathcal{C} \rangle$, where
  - $\mathcal{P} = \{P_{1,r_1,T_1}, \cdots, P_{n_p,r_p,T_{n_p}}\}$: A set of processes
    - $T_i = [t_l, t_u]; t_l \leq t_u$; Optional
    - $P \in \mathcal{P}; P = a_{1;T_1} \circ \cdots \circ a_{m_{p;}T_{mp}}$: A set of actions
      - $T_i = [t_l, t_u]; t_l \leq t_u$; Optional
    - r_i: Priority
  - $\mathcal{I} = \{I_1, \cdots, I_{n_i}\}$: A set of process inclusions
    - $I \in \mathcal{I};\ c = (P, Q) = P[Q]; P, Q \in \mathcal{P}$
  - $\mathcal{C} = \{c_1, \cdots, c_{n_c}\}$: A set of channels
    - $c \in \mathcal{C};\ c = (P, Q);\ P, Q \in \mathcal{P}$

Actions

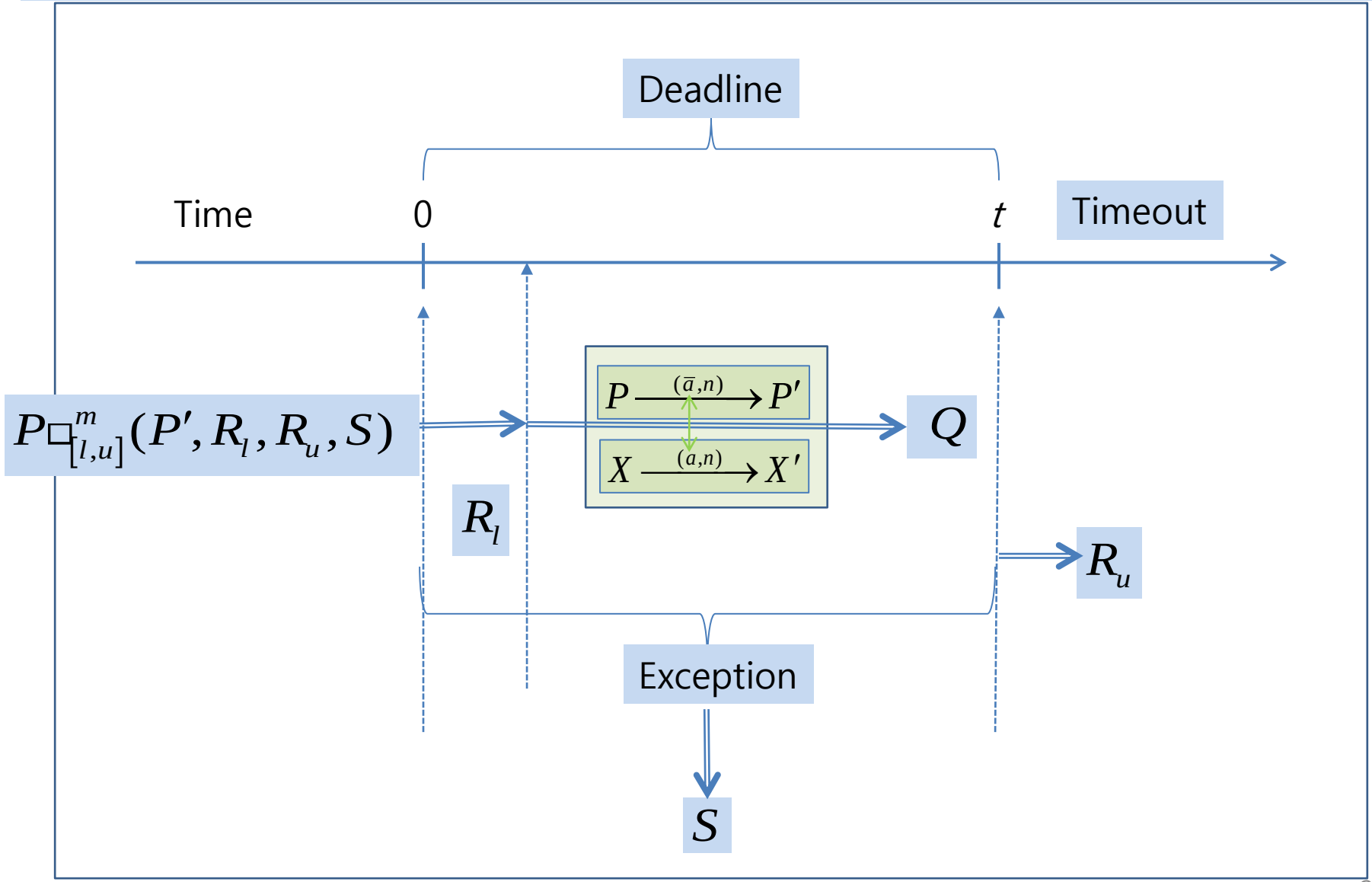
- $P \in \mathcal{P}; P = a_{1;r_1;T_1} \circ \cdots \circ a_{m;r_{m;}T_m}$: A set of actions
  - $T_i = [t_l, t_u]; t_l \leq t_u$; Optional
- Type:
  - τ : Communication
    - Send:
    - Receive:
  - δ : Movement
    - Active
      - In:
      - Out:
    - Passive
      - Get
      - put
  - ρ: Control
    - Start
    - Terminate
    - Exit
    - Suspend
    - wake

Autonomous

Interaction			Action	Co-action
τ			$P: c(m);$	$Q: c(\overline{m});$
δ	Active	In	$P: (in\ Q);$	$Q: (P\ in);$
		out	$P: (out\ Q);$	$Q: (P\ out);$
	Passive	in	$P: (get\ Q);$	$Q: (P\ get);$
		out	$P: (put\ Q);$	$Q: (P\ put);$
ρ	Start		$P: (start\ Q);$	$Q.start;$
	Terminate		$P: (terminate\ Q);$	$Q.terminate;$
	Exit		$P: (exit);$	$P.terminate;$
	Suspend		$P: (suspend\ Q);$	$Q.suspend;$
	Wake		$P: (wake\ Q);$	$Q.wake;$

Heteronomous

Movement Scope



Syntax

$$P ::= nil \mid A \mid P_{(n)} \mid P[Q] \mid P\langle r_t \rangle \mid P + Q \mid P \mid Q \mid A \bullet P$$

$$A ::= \varnothing \mid r_t(\overline{msg}) \mid r_t(msg) \mid M$$

$$M ::= m_t^p(k)\ P \mid P\ m(k)_t^p$$

$$m ::= in \mid out \mid get \mid put$$

Communication (τ) Semantics

Action	$\dfrac{-}{r(a)\bullet P \xrightarrow{r(a)} P}$	ChoiceL	$\dfrac{P \xrightarrow{A} P'}{P+Q \xrightarrow{A} P'}$
ChoiceR	$\dfrac{Q \xrightarrow{A} Q'}{P+Q \xrightarrow{A} Q'}$	ParlL	$\dfrac{P \xrightarrow{A} P'}{P \mid Q \xrightarrow{A} P' \mid Q}$
ParlR	$\dfrac{Q \xrightarrow{A} Q'}{P \mid Q \xrightarrow{A} P \mid Q'}$	ParCom	$\dfrac{P \xrightarrow{A} P', Q \xrightarrow{\bar{A}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$
NestO	$\dfrac{P \xrightarrow{A} P'}{P[Q] \xrightarrow{A} P'[Q]}$	NestI	$\dfrac{Q \xrightarrow{A} Q'}{P[Q] \xrightarrow{A} P[Q']}$
Nest-Com	$\dfrac{P \xrightarrow{A} P', Q \xrightarrow{\bar{A}} Q'}{P[Q] \xrightarrow{\tau} P'[Q']}$		

Movement (δ) Semantics

In	$\dfrac{P \xrightarrow{in\ Q} P', Q \xrightarrow{P\ in} Q'}{P \mid Q \xrightarrow{\delta} Q'[P']}$	Out	$\dfrac{P \xrightarrow{out\ Q} P', Q \xrightarrow{Q\ out} Q'}{Q[P] \xrightarrow{\delta} P' \mid Q'}$
Get	$\dfrac{P \xrightarrow{get\ Q} P', Q \xrightarrow{P\ get} Q'}{P \mid Q \xrightarrow{\delta} P'[Q']}$	Put	$\dfrac{P \xrightarrow{put\ Q} P', Q \xrightarrow{Q\ put} Q'}{P[Q] \xrightarrow{\delta} P' \mid Q'}$
InP	$\dfrac{P_{(n)} \xrightarrow{in^p\ Q} P_{(n)}'}{P_{(n)} \mid Q_{(m)} \xrightarrow{in^p\ Q} Q_{(m)}[P_{(n)}']} (n > m)$		
OutP	$\dfrac{P_{(n)} \xrightarrow{out^p\ Q} P_{(n)}'}{Q_{(m)}[P_{(n)}] \xrightarrow{out^p\ Q} P_{(n)}' \mid Q_{(m)}} (n > m)$		
GetP	$\dfrac{P_{(n)} \xrightarrow{get^p\ Q} P_{(n)}'}{P_{(n)} \mid Q_{(m)} \xrightarrow{get^p\ Q} P_{(n)}'[Q_{(m)}]} (n > m)$		
PutP	$\dfrac{P_{(n)} \xrightarrow{put^p\ Q} P_{(n)}'}{P_{(n)}[Q_{(m)}] \xrightarrow{out^p\ Q} P_{(n)}' \mid Q_{(m)}} (n > m)$		
InN	$\dfrac{P \xrightarrow{in\ Q} P', Q \xrightarrow{P\ in} Q'}{P \mid Q[R] \xrightarrow{\delta} Q'[P' \mid R]}$	GetN	$\dfrac{P \xrightarrow{get\ Q} P', Q \xrightarrow{P\ get} Q'}{P[R] \mid Q \xrightarrow{\delta} P'[R \mid Q']}$

Control (ρ) Semantics

Start	$\dfrac{P \xrightarrow{\text{start } Q} P',\ ( ) \xrightarrow{Q.\text{start}} Q}{P \xrightarrow{\gamma} P[Q]}$
Terminate	$\dfrac{P[Q],\ P \xrightarrow{\text{terminate } Q} P',\ Q \xrightarrow{Q.\text{terminate}} ( )}{P[Q] \xrightarrow{\gamma} P'}$
Exit	$\dfrac{P \xrightarrow{\text{exit}} P',\ P \xrightarrow{P.\text{exit}} ( )}{P \xrightarrow{\gamma} ( )}$
Suspend	$\dfrac{P[Q],\ P \xrightarrow{\text{suspen } Q} P',\ Q \xrightarrow{Q.\text{suspend}} Q_S}{P[Q] \xrightarrow{\gamma} P'[Q']}$
Wake	$\dfrac{P \xrightarrow{\text{wake } Q} P',\ Q_S \xrightarrow{Q.\text{wake}} Q'}{P[Q_S] \xrightarrow{\gamma} P'[Q]}$

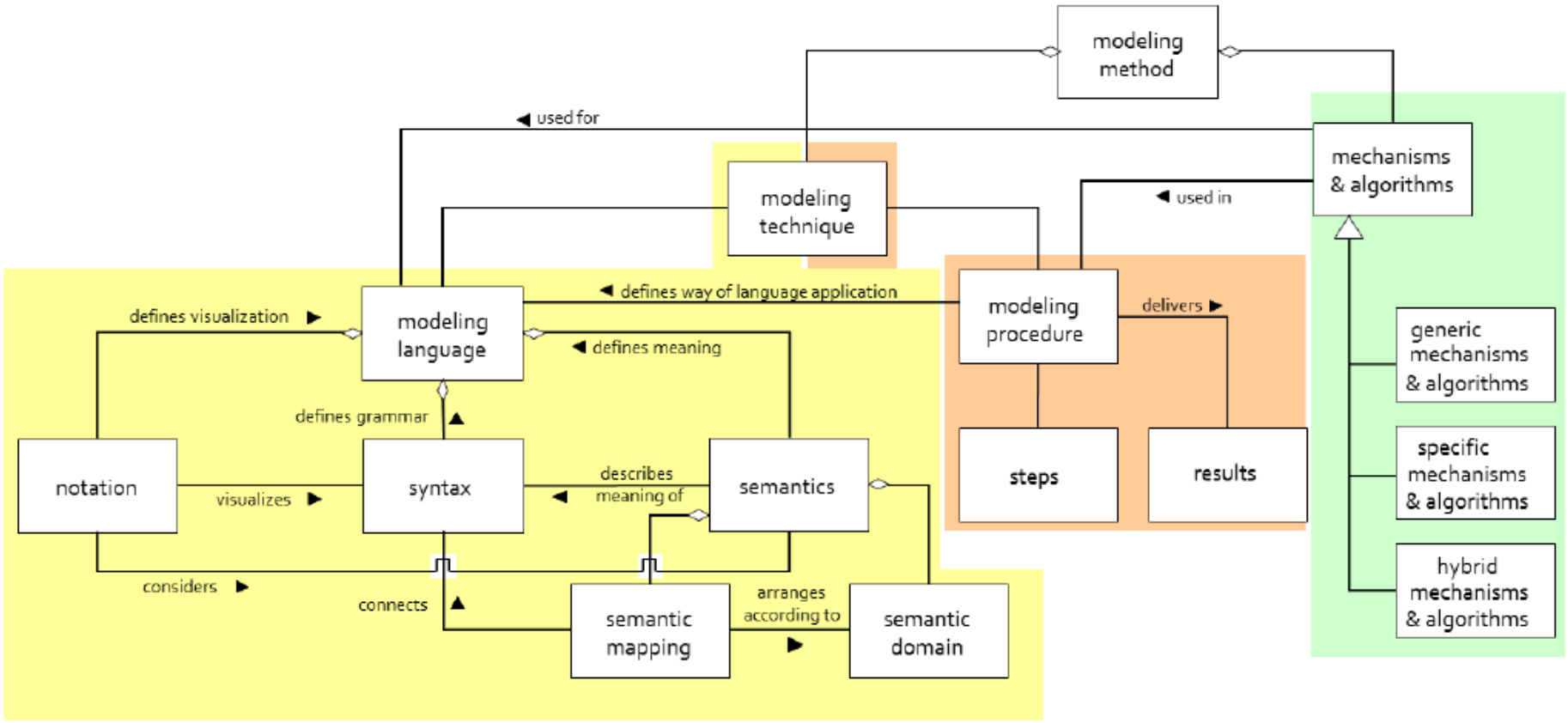
Laws

Choice(1)	$P + P \equiv P$	Choice(2)	$P + Q \equiv Q + P$
Choice(3)	$(P + Q) + R \equiv P + (Q + R)$	Parallel(1)	$P \mid nil \equiv P$
Parallel(2)	$P \mid Q \equiv Q \mid P$	Parallel(3)	$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$
Nesting(1)	$P[nil] \equiv P$	Nesting(2)	$R[P] + R[Q] \equiv R[P + Q]$
Distributive (1)	$P \mid (Q + R) \equiv (P \mid Q) + (P \mid R)$	Distributive (2)	$(A_1 + A_2).P \equiv A_1.P + A_2.P$

VISUALIZATION

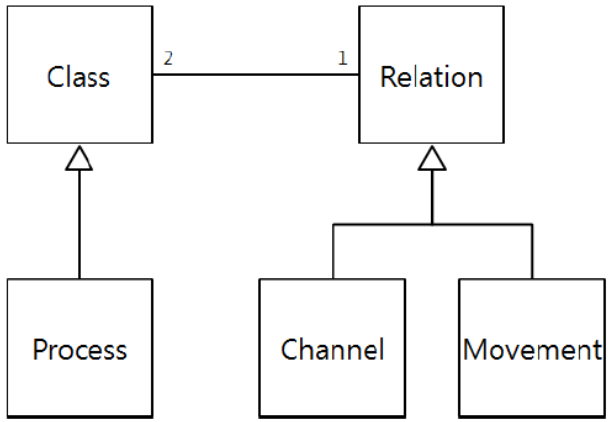
ADOxx Meta-Modeling Platform

Method = (T+MA)·(L+P)



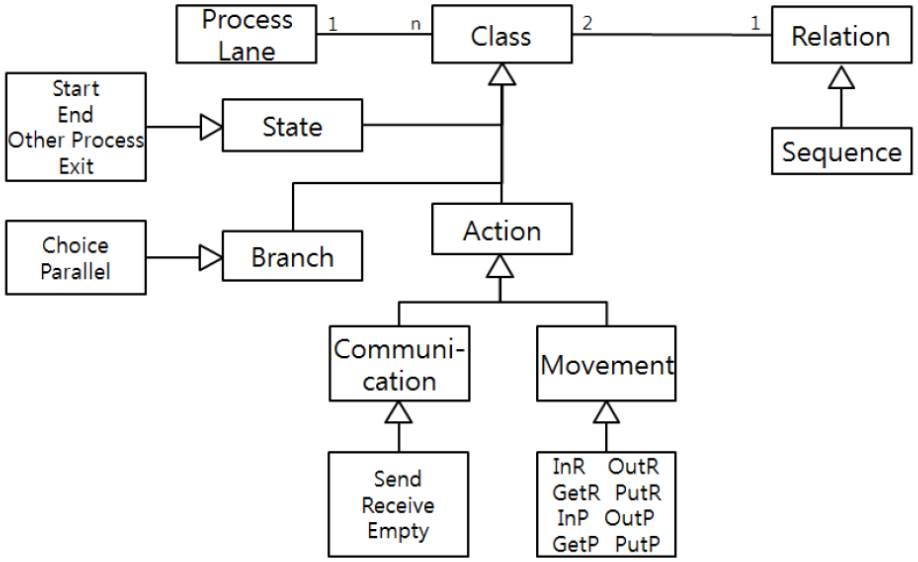
Karagiannis, D., Kühn, H.: „Metamodelling Platforms". In Bauknecht, K., Min Tjoa, A., Quirchmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2002, LNCS 2455, Springer, Berlin/Heidelberg, p. 182.

ITL Icons

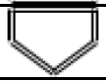
Meta-Model	Icon	
	Process	
	Channel	
	Movement	

ITS Icons

Meta-Model



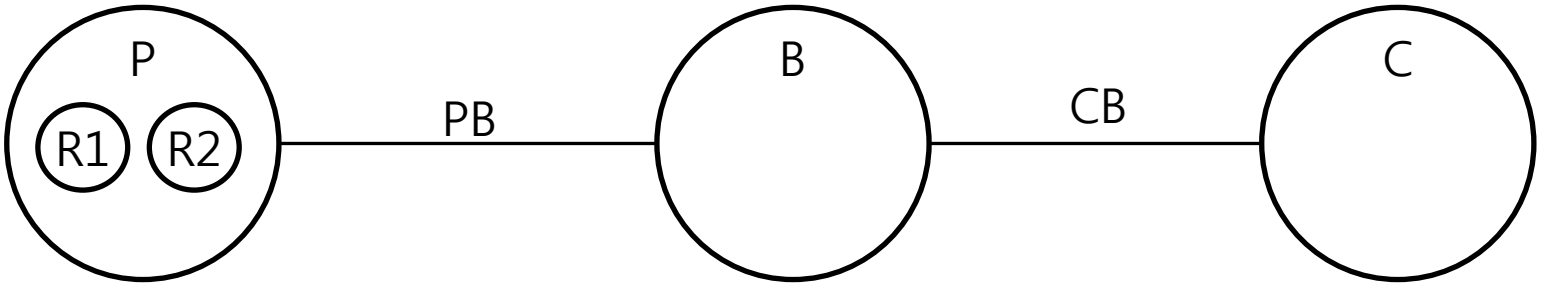
Icon

Process Lane		Start		End		Other Process	
Exit		Choice		Parallel		Send	
Receive		Empty		In R		Out R	
Get R		Put R		In P		Out P	
Get P		Put P		Sequence			

EXAMPLE: PBC

Example: PBC

- Operational Requirements
  - *Producer* produces two resources, R1 and R2.
  - *Producer* stores the resources in *Buffer* in order.
  - *Producer* informs *Buffer* of the order of R1 and R2, or R2 and R1.
  - *Consumer* consumes the resources from *Buffer* in order.
  - The order of the consumption is formed to *Buffer* by *Consumer*.
- Secure Requirements
  - Security:
    - The order should not be violated, since the first resource contains security information to decode the second resource.
    - The propagation between the first and the second should be less than 30 seconds.
  - Safety
    - The resources produced by *Producer* should be consumed by *Consumer* less than 5 minutes.



PBC Example (0)

- δ-Calculus

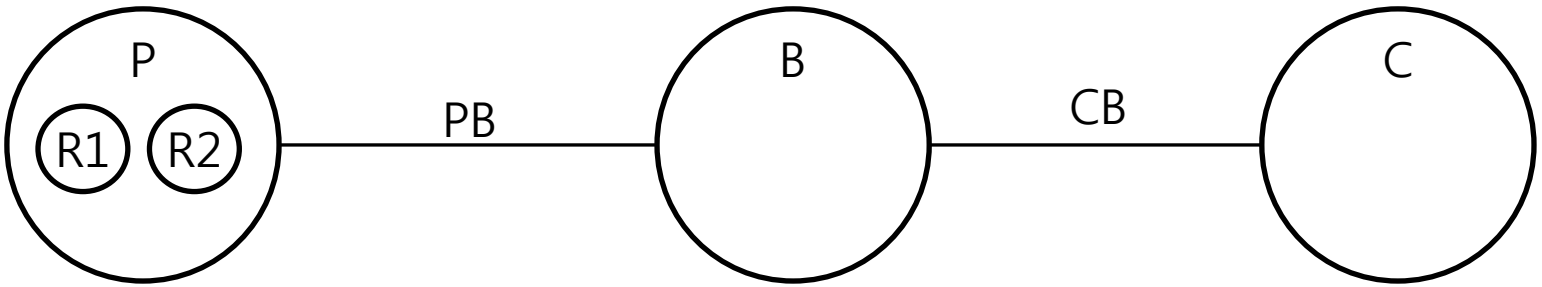
$PBC = P[R1 \parallel R2] \parallel B \parallel C$

$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$
$R2 = P\ put.B\ get.B\ put.C\ get.exit$



CASE 1

PBC Example (1) : τ_1

- δ-Calculus

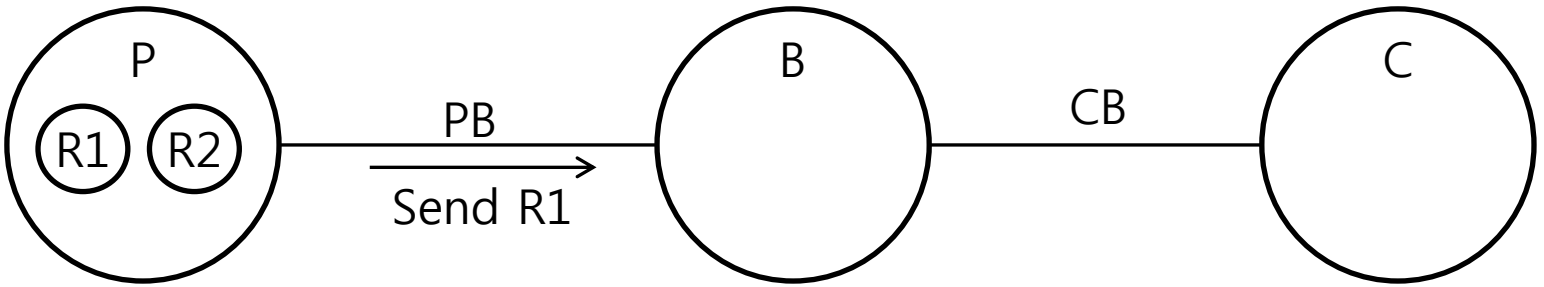
$PBC = P[R1 \parallel R2] \parallel B \parallel C$

$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$
$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (2) : $\delta_{1,1}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

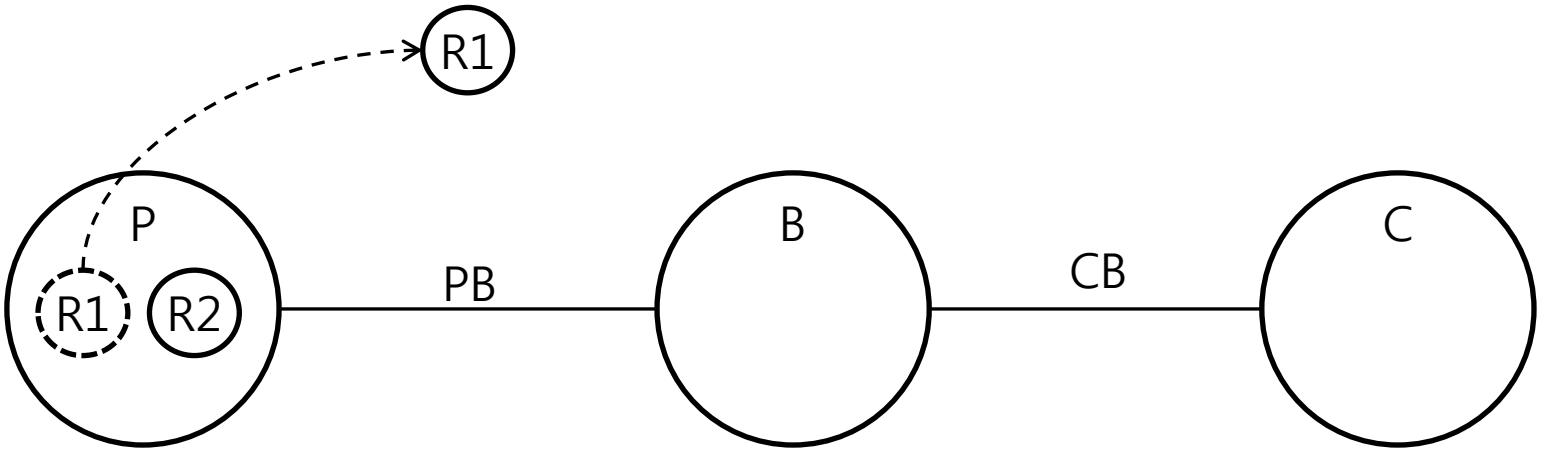
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (3) : $\delta_{1,2}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

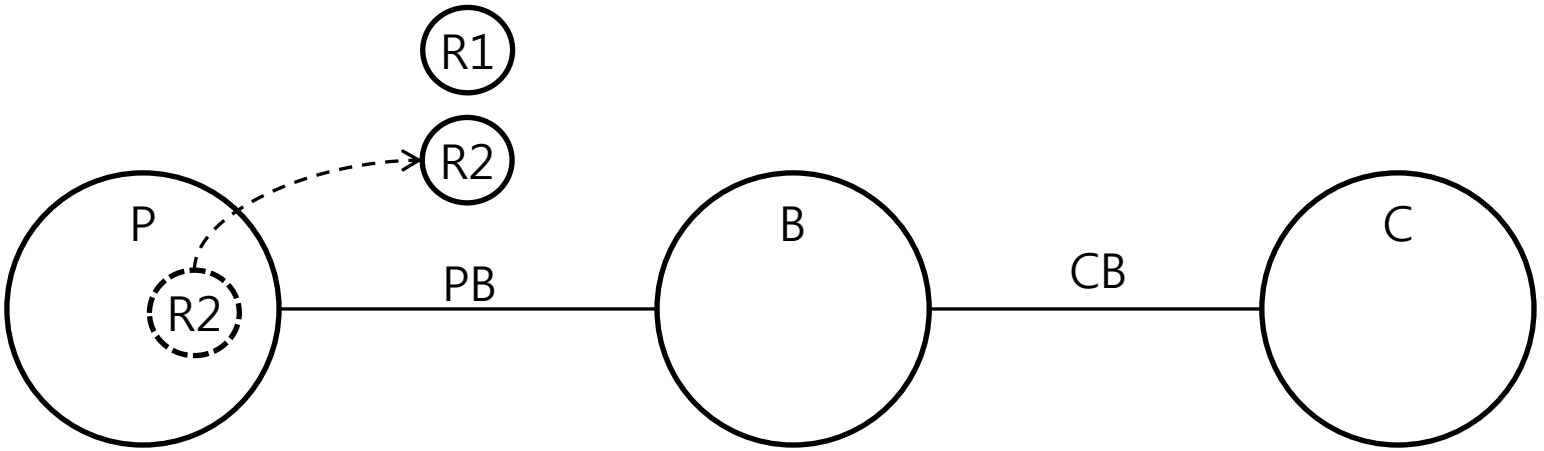
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (4) : $\delta_{2,1}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

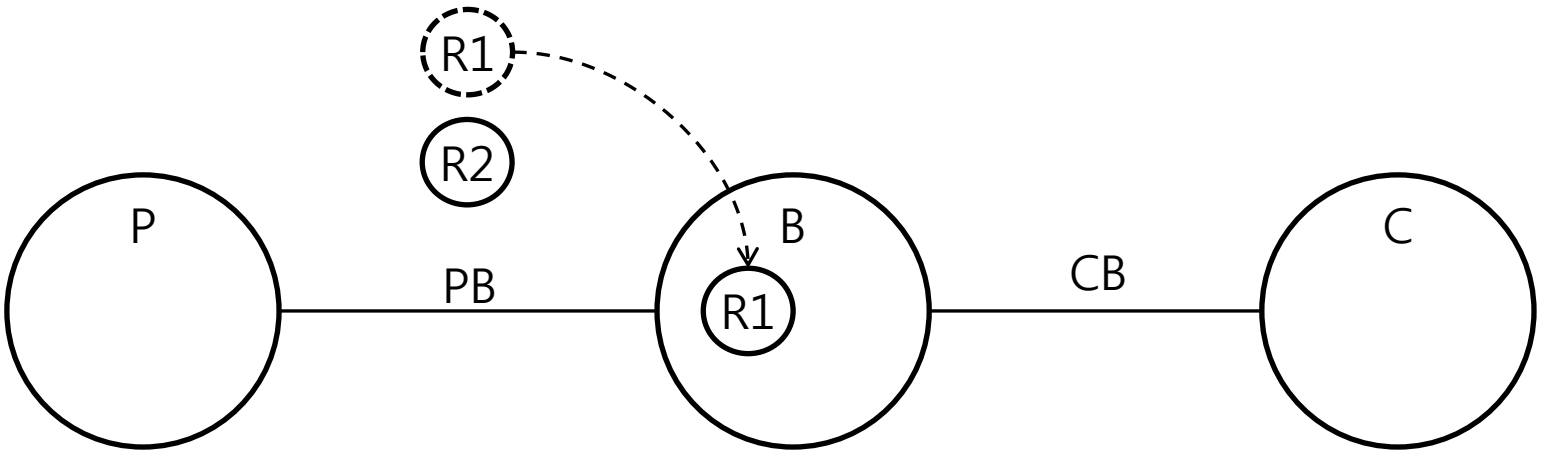
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (5) : $\delta_{2,2}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

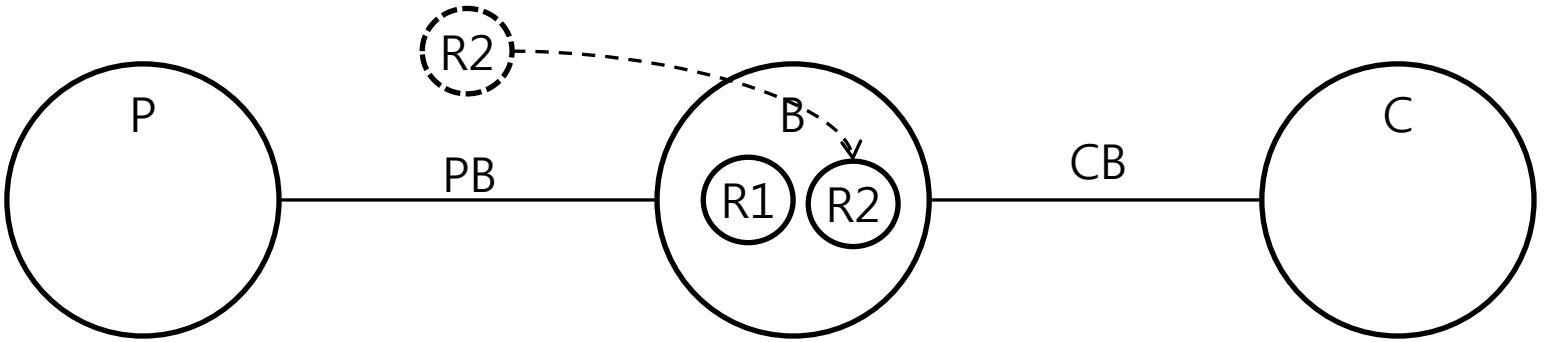
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (6) : $\delta_{3,1}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

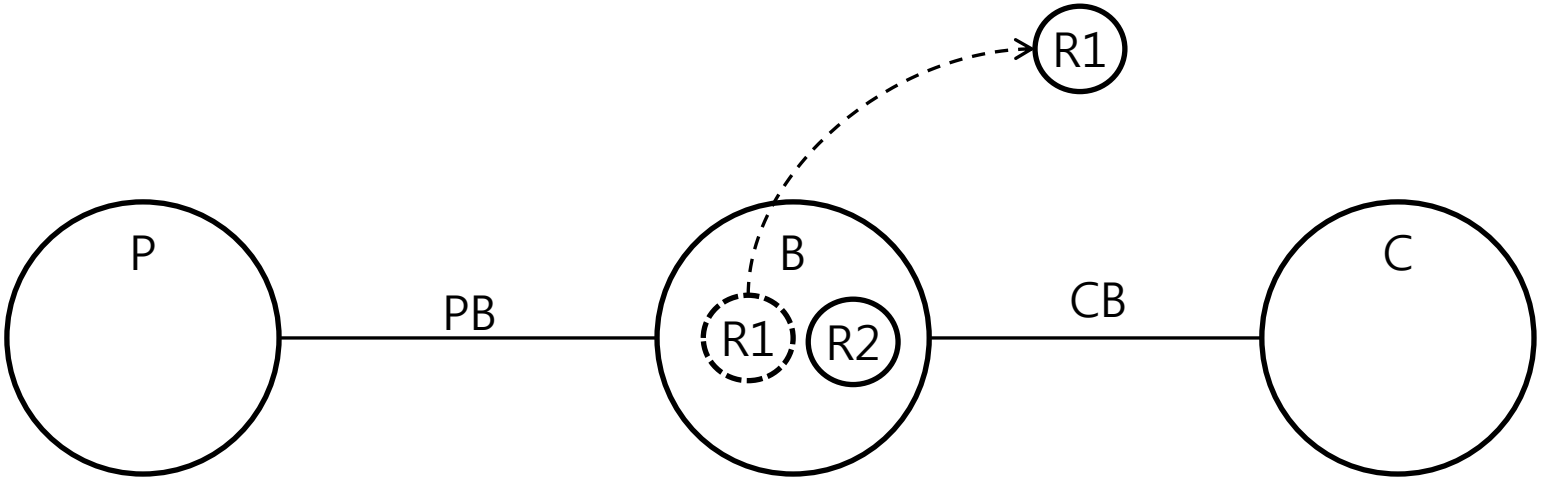
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (7) : $\delta_{3,2}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

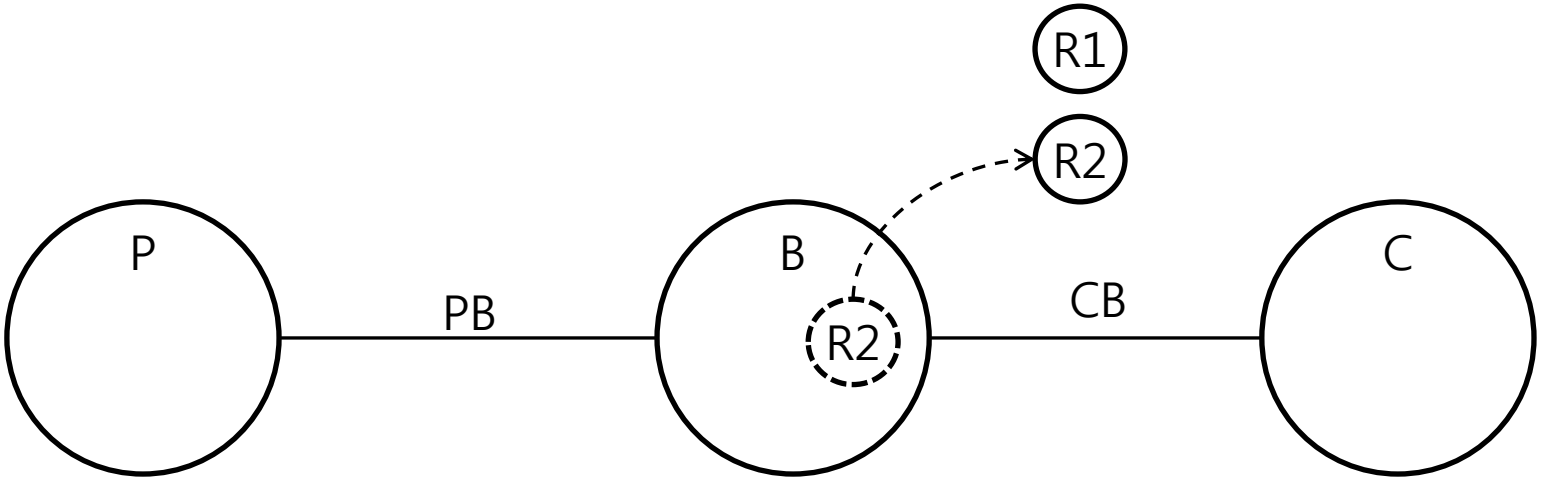
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (8) : $\delta_{4,1}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

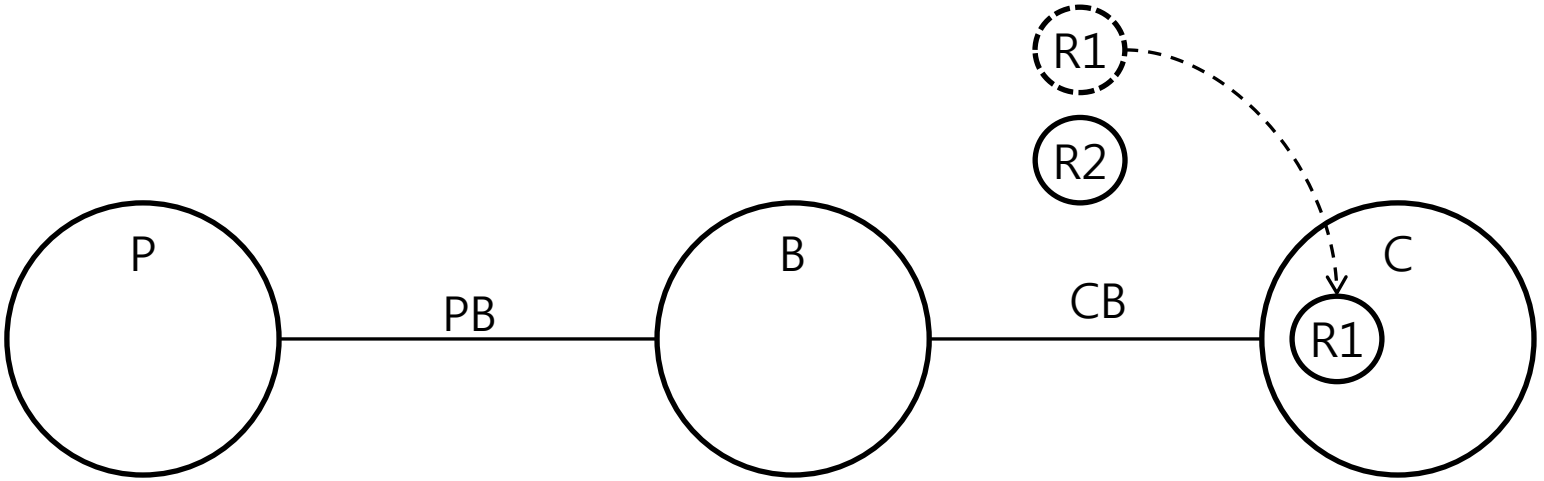
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (9) : $\delta_{4,2}$

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

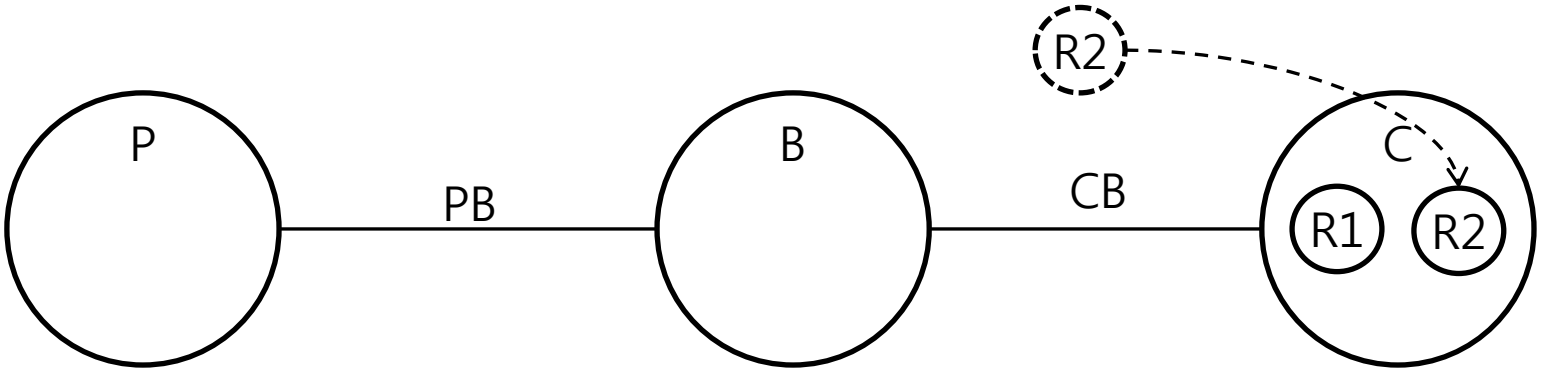
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



PBC Example (10)

- δ-Calculus

$PBC = P[R1 \parallel R2] \parallel B \parallel C$

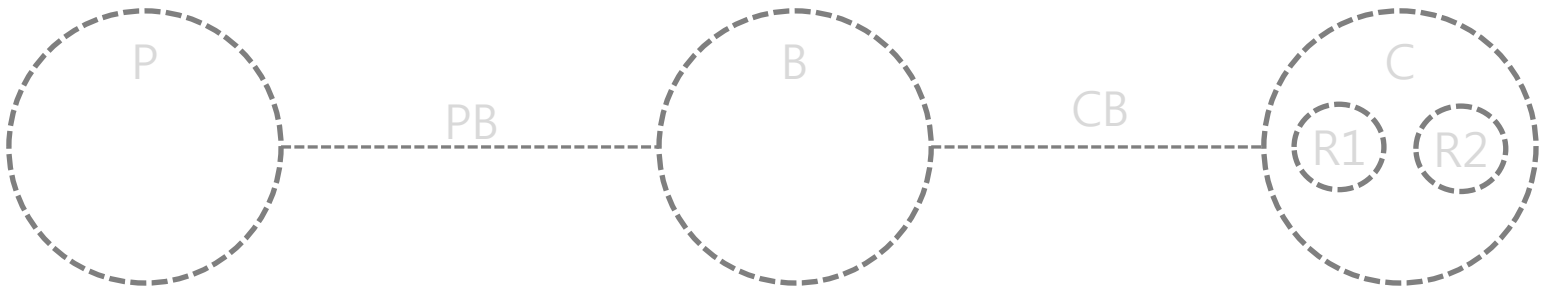
$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$

$R2 = P\ put.B\ get.B\ put.C\ get.exit$



CASE 2

PBC Example (1) : τ_2

- δ-Calculus

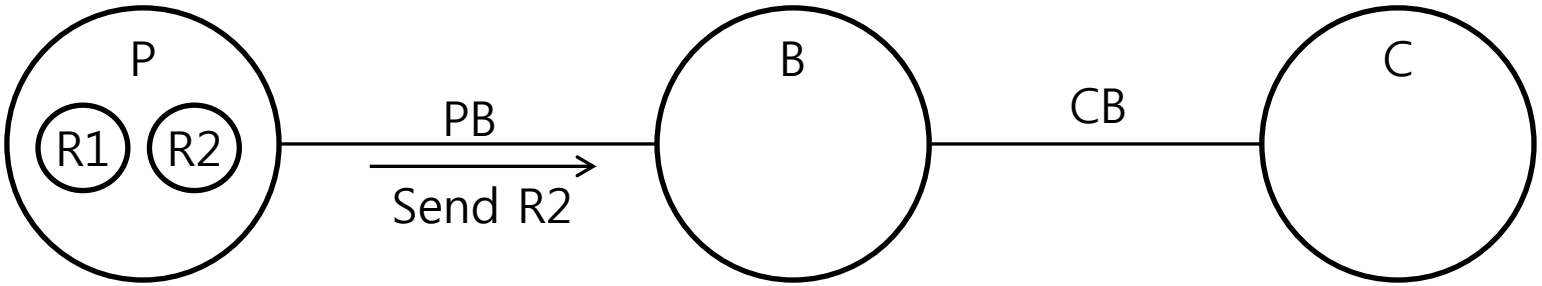
$PBC = P[R1 \parallel R2] \parallel B \parallel C$

$P = (PB(\overline{Send\ R1}).put\ R1.put\ R2 + PB(\overline{Send\ R2}).put\ R2.put\ R1).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$
$R2 = P\ put.B\ get.B\ put.C\ get.exit$



CASE 3: DEADLOCK

PBC Example (1) : τ_3

- δ-Calculus

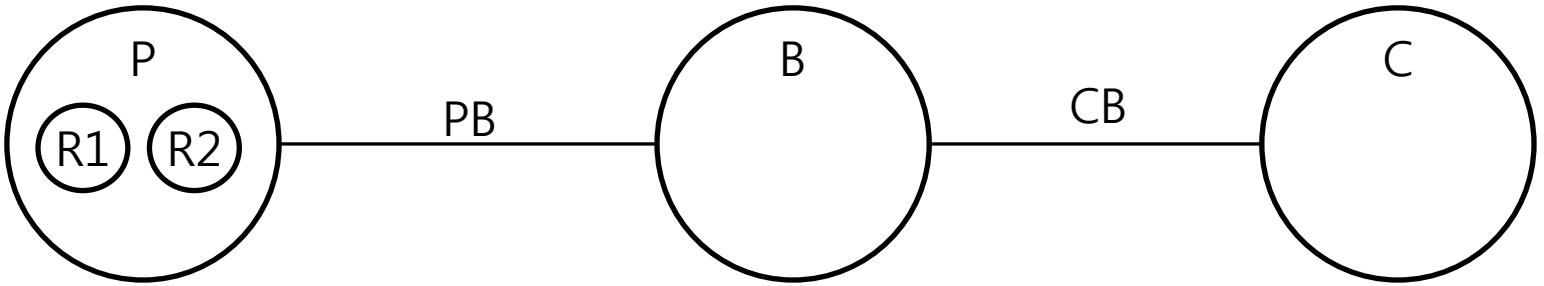
$PBC = P[R1 \parallel R2] \parallel B \parallel C$

$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

$R1 = P\ put.B\ get.B\ put.C\ get.exit$
$R2 = P\ put.B\ get.B\ put.C\ get.exit$



CASE 4: DEADLOCK

PBC Example (1) : τ_4

- δ-Calculus

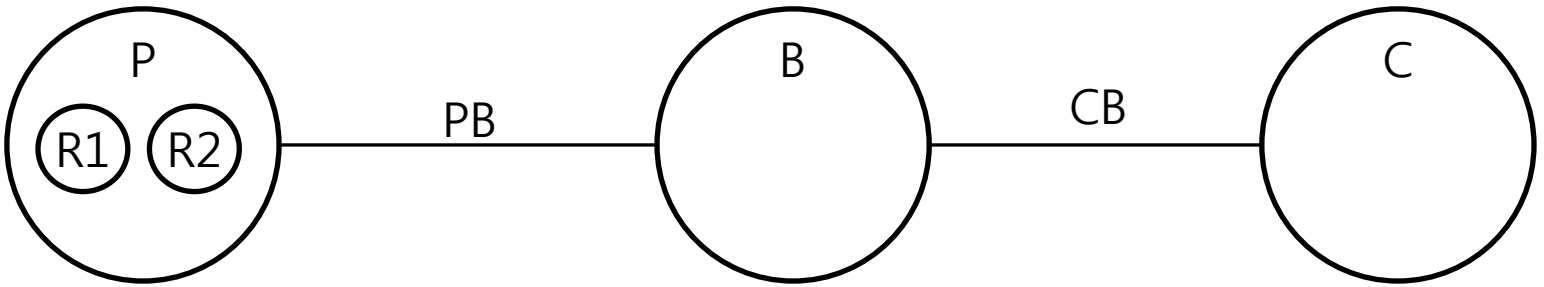
$PBC = P[R1 \parallel R2] \parallel B \parallel C$

$P = \left(PB\left(\overline{Send\ R1}\right).put\ R1.put\ R2 + PB\left(\overline{Send\ R2}\right).put\ R2.put\ R1\right).exit$

$B = (PB(Send\ R1).get\ R1.get\ R2 + PB(Send\ R2).get\ R2.get\ R2).put\ R1.put\ R2.exit$

$C = get\ R1.get\ R2.exit$

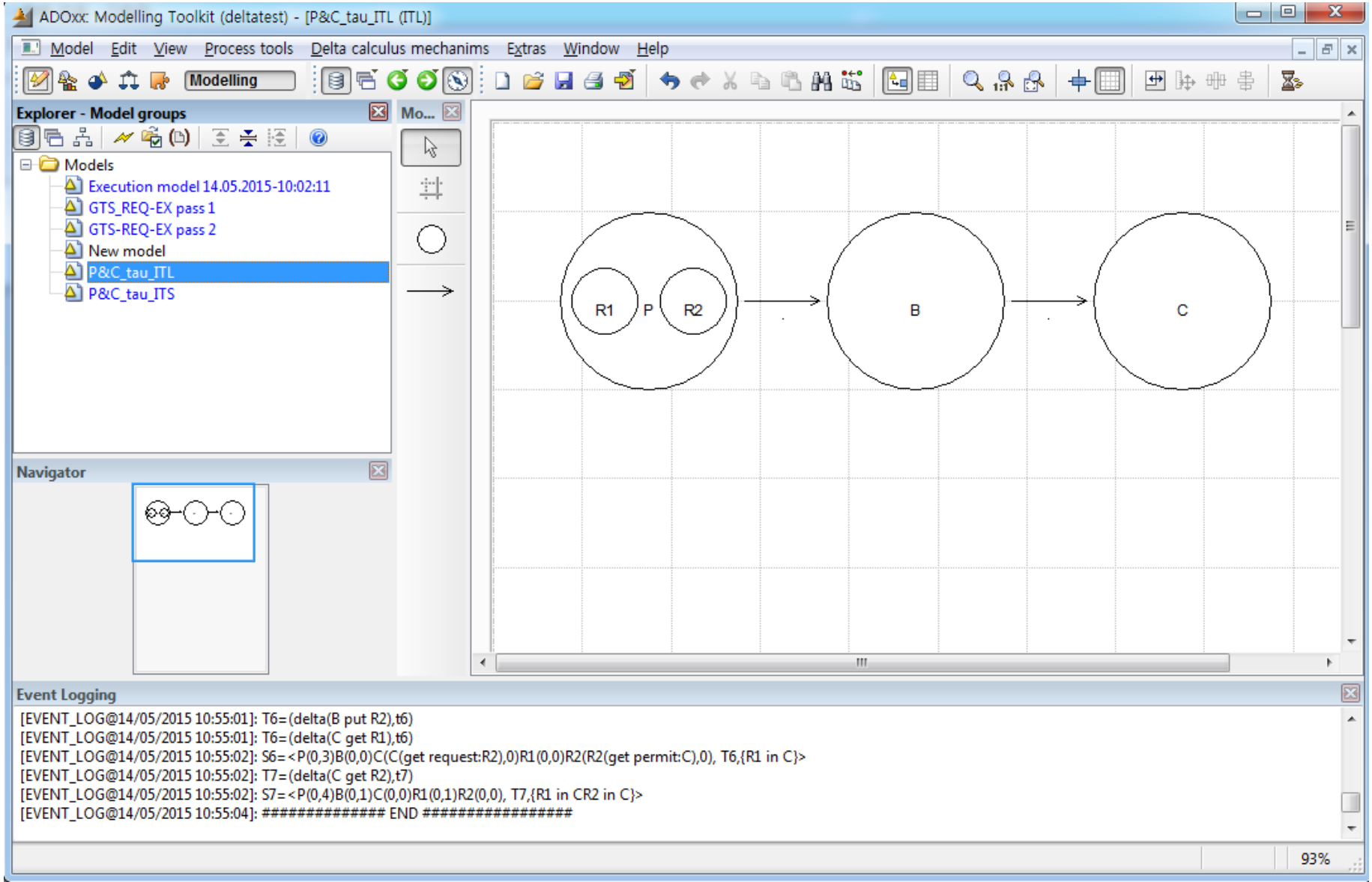
$R1 = P\ put.B\ get.B\ put.C\ get.exit$
$R2 = P\ put.B\ get.B\ put.C\ get.exit$



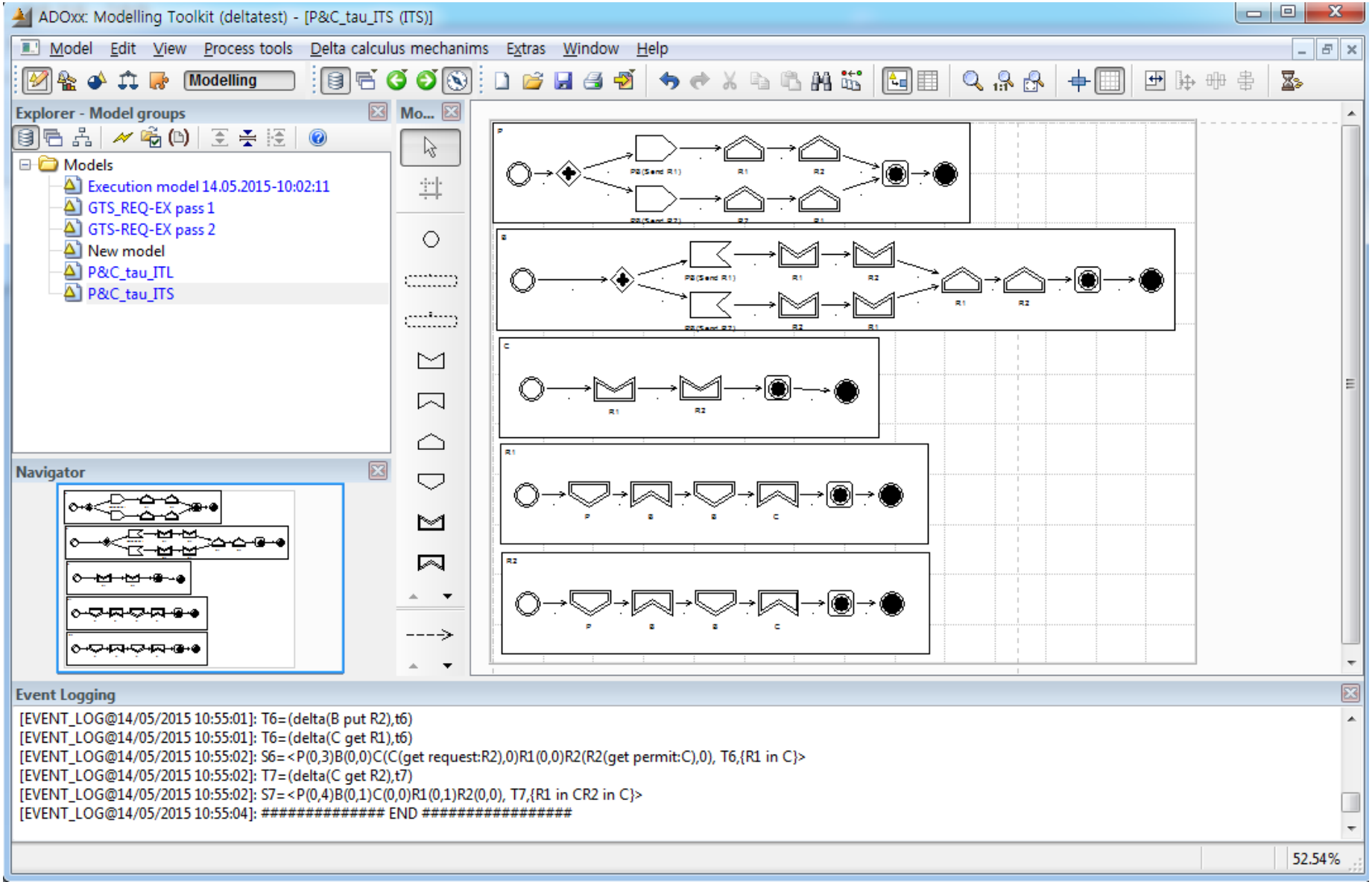
Temporal Properties

Interactions	Case 1	Case 2
$\tau 1 = P{:}PB, Send\ R1, B{:}PB(Send\ R1)$	0-1	X
$\tau 2 = P{:}PB, Send\ R2, B{:}PB(Send\ R2)$	X	0-1
$\delta 1,1 = P{:}put\ R1, R1{:}P\ put$	1-2	2-3
$\delta 1,2 = P{:}put\ R2, R2{:}P\ put$	2-3	1-2
$\delta 2,1 = B{:}get\ R1, R1{:}B\ get$	2-3	3-4
$\delta 2,2 = B{:}get\ R2, R2{:}P\ get$	3-4	2-3
$\delta 3,1 = B{:}put\ R1, R1{:}B\ put$	4-5	4-5
$\delta 3,2 = B{:}put\ R2, R2{:}B\ put$	5-6	5-6
$\delta 4,1 = C{:}get\ R1, R1{:}C\ get$	5-6	5-6
$\delta 4,2 = C{:}get\ R2, R2{:}C\ get$	6-7	6-7

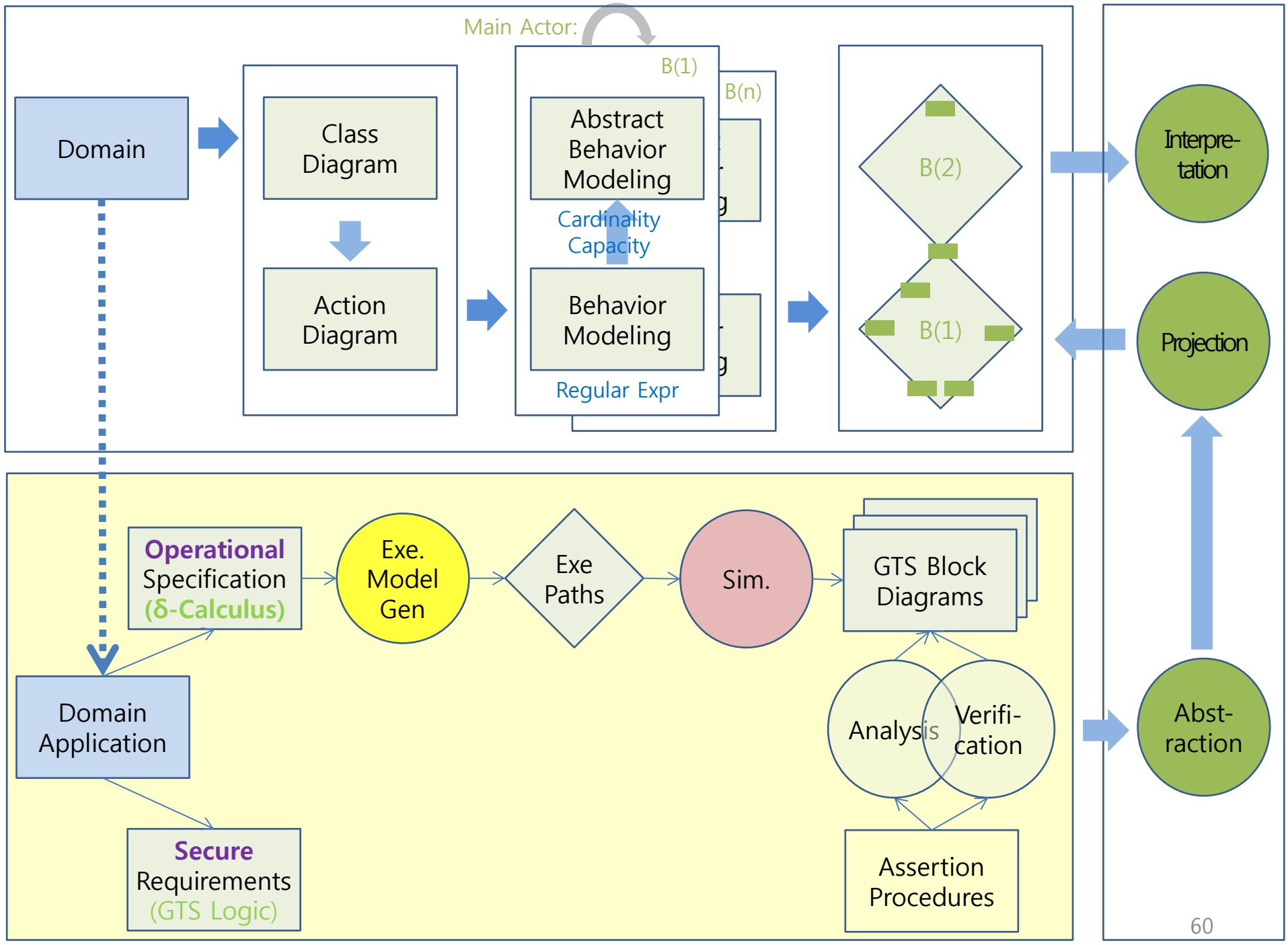
ITL View


ADOxx: Modelling Toolkit (deltatest) - [P&C_tau_ITL (ITL)]
Model Edit View Process tools Delta calculus mechanims Extras Window Help
Modelling
Explorer - Model groups
Models
Execution model 14.05.2015-10:02:11
GTS_REQ-EX pass 1
GTS-REQ-EX pass 2
New model
P&C_tau_ITL
P&C_tau_ITS
Navigator
R1
P
R2
B
C
Event Logging
[EVENT_LOG@14/05/2015 10:55:01]: T6=(delta(B put R2),t6)
[EVENT_LOG@14/05/2015 10:55:01]: T6=(delta(C get R1),t6)
[EVENT_LOG@14/05/2015 10:55:02]: S6=<P(0,3)B(0,0)C(C(get request:R2),0)R1(0,0)R2(R2(get permit:C),0), T6,{R1 in C}>
[EVENT_LOG@14/05/2015 10:55:02]: T7=(delta(C get R2),t7)
[EVENT_LOG@14/05/2015 10:55:02]: S7=<P(0,4)B(0,1)C(0,0)R1(0,1)R2(0,0), T7,{R1 in CR2 in C}>
[EVENT_LOG@14/05/2015 10:55:04]: ############## END #################
93%

ITS Views

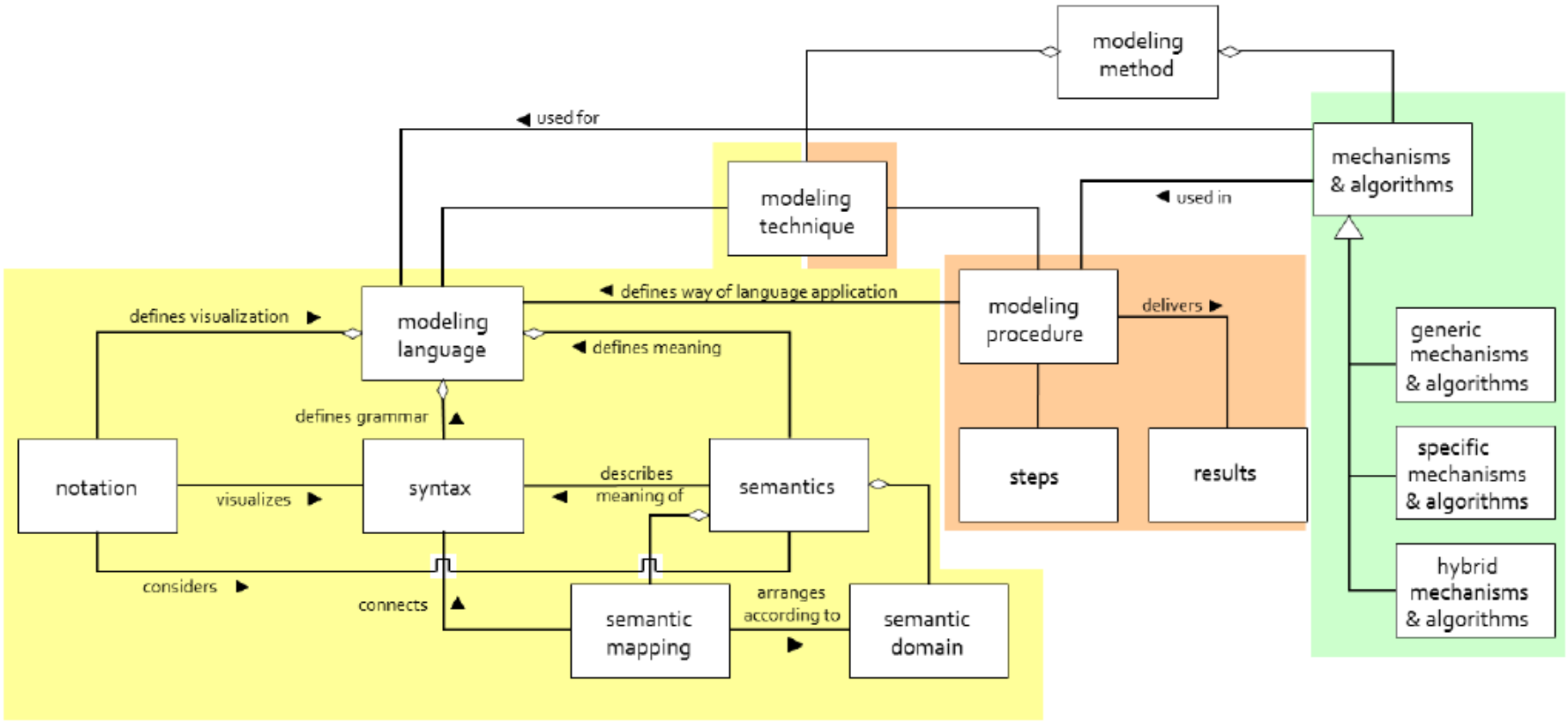

ADOxx: Modelling Toolkit (deltatest) - [P&C_tau_ITS (ITS)]
Model Edit View Process tools Delta calculus mechanims Extras Window Help
Modelling
Explorer - Model groups
Models
Execution model 14.05.2015-10:02:11
GTS_REQ-EX pass 1
GTS-REQ-EX pass 2
New model
P&C_tau_ITL
P&C_tau_ITS
Navigator
Event Logging
[EVENT_LOG@14/05/2015 10:55:01]: T6=(delta(B put R2),t6)
[EVENT_LOG@14/05/2015 10:55:01]: T6=(delta(C get R1),t6)
[EVENT_LOG@14/05/2015 10:55:02]: S6=<P(0,3)B(0,0)C(C(get request:R2),0)R1(0,0)R2(R2(get permit:C),0), T6,{R1 in C}>
[EVENT_LOG@14/05/2015 10:55:02]: T7=(delta(C get R2),t7)
[EVENT_LOG@14/05/2015 10:55:02]: S7=<P(0,4)B(0,1)C(0,0)R1(0,1)R2(0,0), T7,{R1 in CR2 in C}>
[EVENT_LOG@14/05/2015 10:55:04]: ############## END #################
52.54%

Step 2
Execution Model Generation


Main Actor:
Domain
Class Diagram
Action Diagram
B(1)
B(n)
Abstract Behavior Modeling
Cardinality Capacity
Behavior Modeling
Regular Expr
B(2)
B(1)
Interpre-tation
Projection
Abst-raction
Operational Specification (δ-Calculus)
Exe. Model Gen
Exe Paths
Sim.
GTS Block Diagrams
Domain Application
Analysis
Verifi-cation
Secure Requirements (GTS Logic)
Assertion Procedures

ADOxx Meta-Modeling Platform

Method = (T+MA)·(L+P)



Karagiannis, D., Kühn, H.: „Metamodelling Platforms". In Bauknecht, K., Min Tjoa, A., Quirchmayer, G. (Eds.): Proceedings of the Third International Conference EC-Web 2002 – Dexa 2002, Aix-en-Provence, France, September 2002, LNCS 2455, Springer, Berlin/Heidelberg, p. 182.

Execution Model

- Labelled Transition System
  - System States
  - Transitions
  - Global and local clocks.
- The process state s_j for Process P_i:
  - $s_{i,j} = \langle a_j, t_{i,j} \rangle$
  - $t_{i,j}$: the local time at the state.
- The system state:
  - $S_i = \langle s_{1,i_1}, s_{2,i_2}, \cdots, s_{n,i_n}, T_i, I_i \rangle$
  - T_i: The global time at the system state S_i and
  - I_i is the inclusion relations of the processes for the system at the state.
- E.g.: $S = \langle \{P_1, P_2, \cdots, P_n\}, C, I \rangle$
  - $S_0 = \langle s_{1,i_0}, s_{2,i_0}, \cdots, s_{n,i_0}, T_0, I_0 \rangle$
  - $S_n = \langle s_{1,i_n}, s_{2,i_n}, \cdots, s_{n,i_n}, T_n, I_n \rangle$

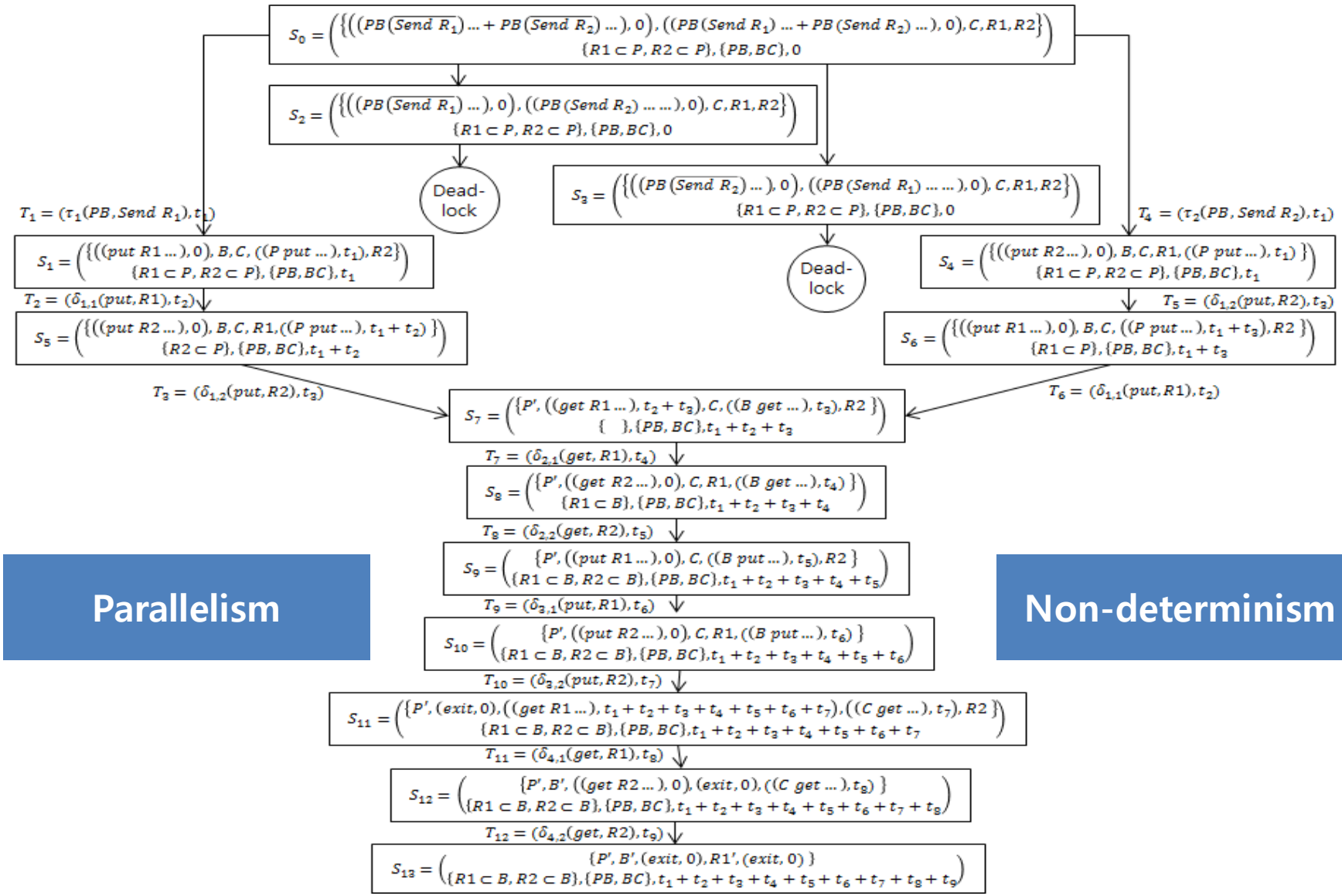
Communication (τ) Semantics

Action	$\dfrac{-}{r(a)\bullet P \xrightarrow{r(a)} P}$	ChoiceL	$\dfrac{P \xrightarrow{A} P'}{P+Q \xrightarrow{A} P'}$
ChoiceR	$\dfrac{Q \xrightarrow{A} Q'}{P+Q \xrightarrow{A} Q'}$	ParlL	$\dfrac{P \xrightarrow{A} P'}{P \mid Q \xrightarrow{A} P' \mid Q}$
ParlR	$\dfrac{Q \xrightarrow{A} Q'}{P \mid Q \xrightarrow{A} P \mid Q'}$	ParCom	$\dfrac{P \xrightarrow{A} P', Q \xrightarrow{\bar{A}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$
NestO	$\dfrac{P \xrightarrow{A} P'}{P[Q] \xrightarrow{A} P'[Q]}$	NestI	$\dfrac{Q \xrightarrow{A} Q'}{P[Q] \xrightarrow{A} P[Q']}$
Nest-Com	$\dfrac{P \xrightarrow{A} P', Q \xrightarrow{\bar{A}} Q'}{P[Q] \xrightarrow{\tau} P'[Q']}$		

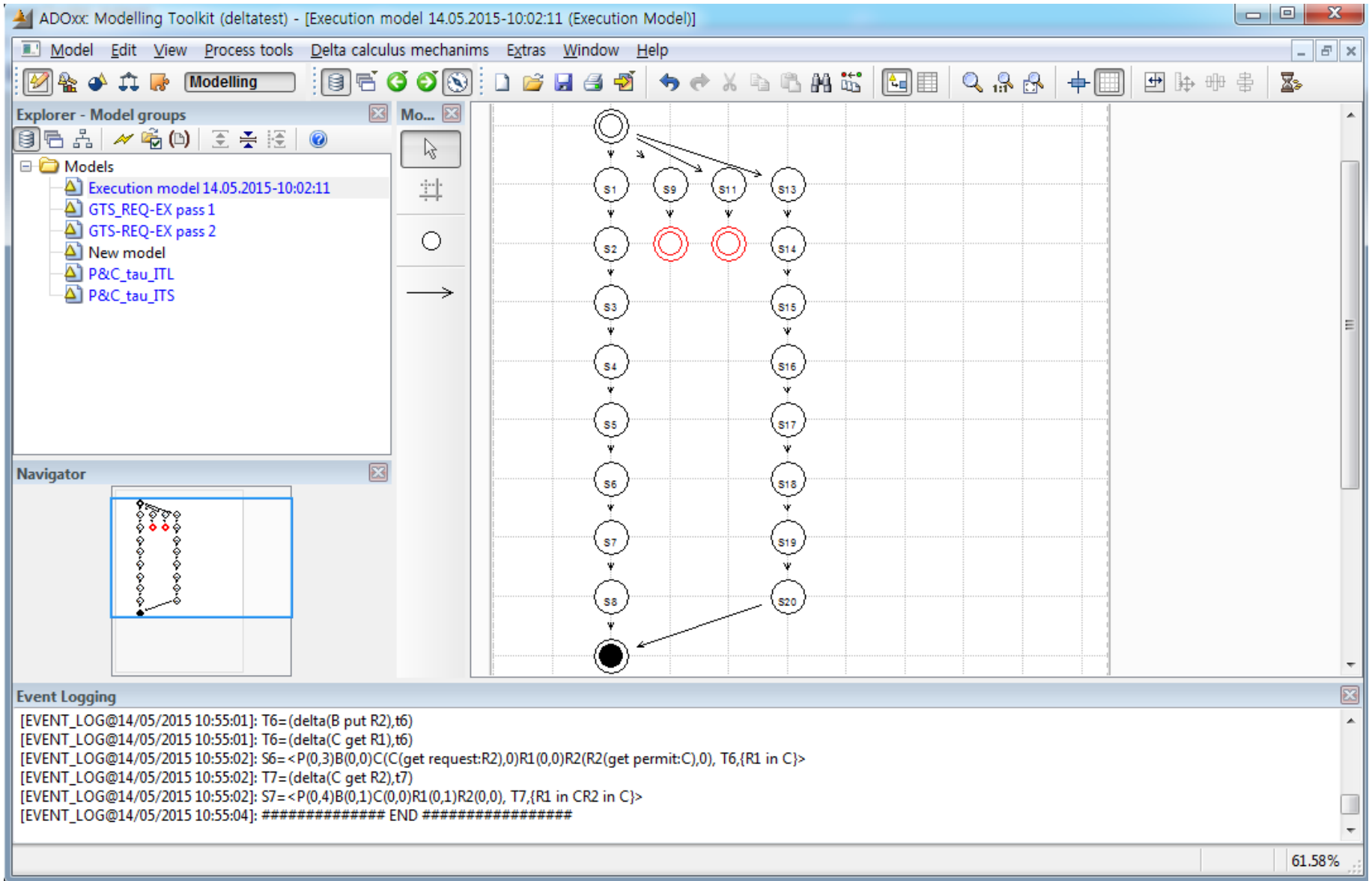
Movement (δ) Semantics

In	$\dfrac{P \xrightarrow{in\ Q} P', Q \xrightarrow{P\ in} Q'}{P \mid Q \xrightarrow{\delta} Q'[P']}$	Out	$\dfrac{P \xrightarrow{out\ Q} P', Q \xrightarrow{Q\ out} Q'}{Q[P] \xrightarrow{\delta} P' \mid Q'}$
Get	$\dfrac{P \xrightarrow{get\ Q} P', Q \xrightarrow{P\ get} Q'}{P \mid Q \xrightarrow{\delta} P'[Q']}$	Put	$\dfrac{P \xrightarrow{put\ Q} P', Q \xrightarrow{Q\ put} Q'}{P[Q] \xrightarrow{\delta} P' \mid Q'}$
InP	$\dfrac{P_{(n)} \xrightarrow{in^p\ Q} P_{(n)}'}{P_{(n)} \mid Q_{(m)} \xrightarrow{in^p\ Q} Q_{(m)}[P_{(n)}']} (n > m)$		
OutP	$\dfrac{P_{(n)} \xrightarrow{out^p\ Q} P_{(n)}'}{Q_{(m)}[P_{(n)}] \xrightarrow{out^p\ Q} P_{(n)}' \mid Q_{(m)}} (n > m)$		
GetP	$\dfrac{P_{(n)} \xrightarrow{get^p\ Q} P_{(n)}'}{P_{(n)} \mid Q_{(m)} \xrightarrow{get^p\ Q} P_{(n)}'[Q_{(m)}]} (n > m)$		
PutP	$\dfrac{P_{(n)} \xrightarrow{put^p\ Q} P_{(n)}'}{P_{(n)}[Q_{(m)}] \xrightarrow{out^p\ Q} P_{(n)}' \mid Q_{(m)}} (n > m)$		
InN	$\dfrac{P \xrightarrow{in\ Q} P', Q \xrightarrow{P\ in} Q'}{P \mid Q[R] \xrightarrow{\delta} Q'[P' \mid R]}$	GetN	$\dfrac{P \xrightarrow{get\ Q} P', Q \xrightarrow{P\ get} Q'}{P[R] \mid Q \xrightarrow{\delta} P'[R \mid Q']}$

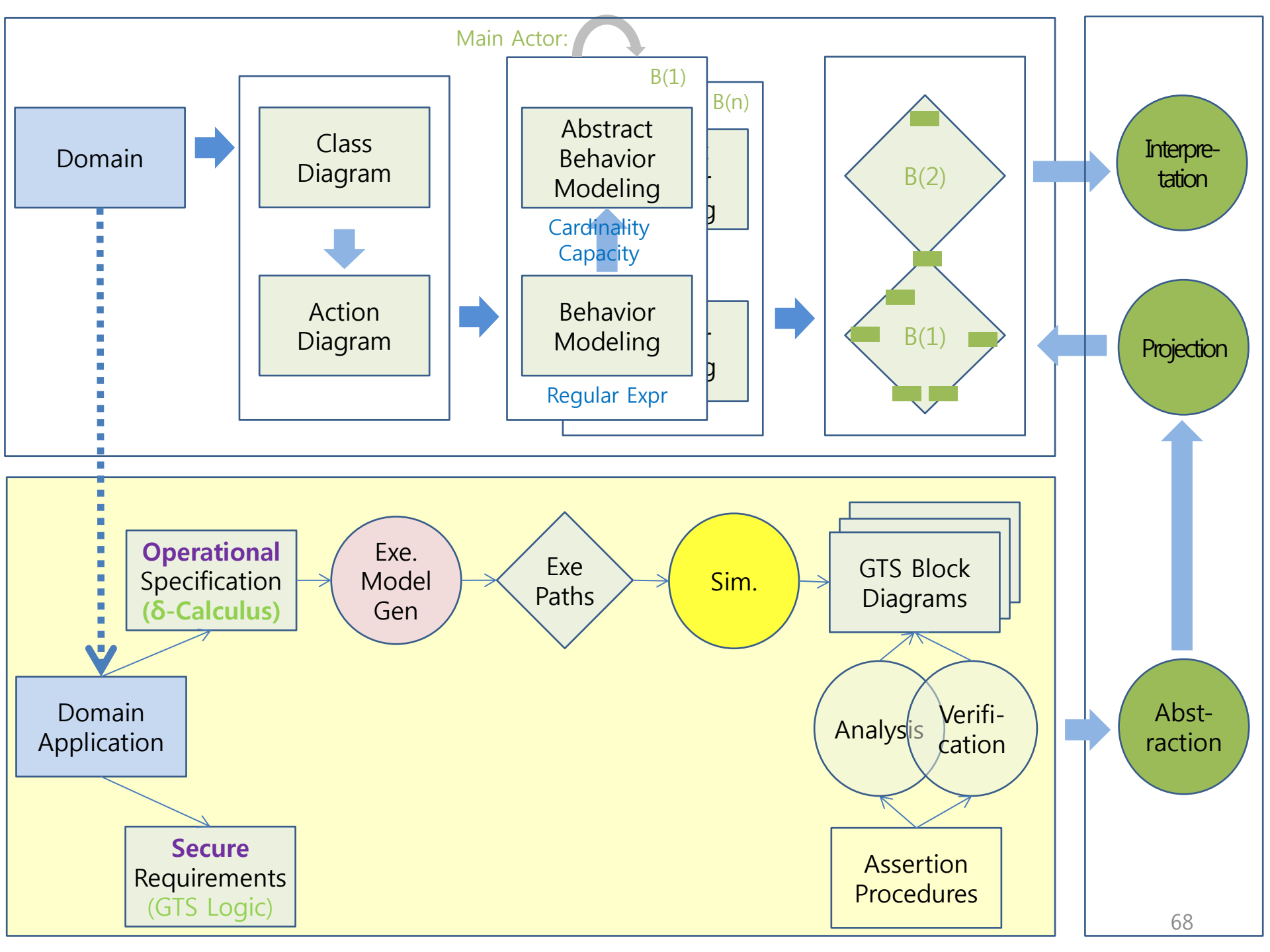
Example: PBC


$S_0 = \left(\{((PB(\overline{Send\ R_1}) \ldots + PB(\overline{Send\ R_2}) \ldots), 0), ((PB(Send\ R_1) \ldots + PB(Send\ R_2) \ldots), 0), C, R1, R2\}, \{R1 \subset P, R2 \subset P\}, \{PB, BC\}, 0\right)$
$S_2 = \left(\{((PB(\overline{Send\ R_1}) \ldots), 0), ((PB(Send\ R_2) \ldots \ldots), 0), C, R1, R2\}, \{R1 \subset P, R2 \subset P\}, \{PB, BC\}, 0\right)$
Dead-lock
$S_3 = \left(\{((PB(\overline{Send\ R_2}) \ldots), 0), ((PB(Send\ R_1) \ldots \ldots), 0), C, R1, R2\}, \{R1 \subset P, R2 \subset P\}, \{PB, BC\}, 0\right)$
Dead-lock
$T_1 = (\tau_1(PB, Send\ R_1), t_1)$
$T_4 = (\tau_2(PB, Send\ R_2), t_1)$
$S_1 = \left(\{((put\ R1 \ldots), 0), B, C, ((P\ put \ldots), t_1), R2\}, \{R1 \subset P, R2 \subset P\}, \{PB, BC\}, t_1\right)$
$S_4 = \left(\{((put\ R2 \ldots), 0), B, C, R1, ((P\ put \ldots), t_1)\}, \{R1 \subset P, R2 \subset P\}, \{PB, BC\}, t_1\right)$
$T_2 = (\delta_{1,1}(put, R1), t_2)$
$T_5 = (\delta_{1,2}(put, R2), t_3)$
$S_5 = \left(\{((put\ R2 \ldots), 0), B, C, R1, ((P\ put \ldots), t_1 + t_2)\}, \{R2 \subset P\}, \{PB, BC\}, t_1 + t_2\right)$
$S_6 = \left(\{((put\ R1 \ldots), 0), B, C, ((P\ put \ldots), t_1 + t_3), R2\}, \{R1 \subset P\}, \{PB, BC\}, t_1 + t_3\right)$
$T_3 = (\delta_{1,2}(put, R2), t_3)$
$T_6 = (\delta_{1,1}(put, R1), t_2)$
$S_7 = \left(\{P', ((get\ R1 \ldots), t_2 + t_3), C, ((B\ get \ldots), t_3), R2\}, \{\ \}, \{PB, BC\}, t_1 + t_2 + t_3\right)$
$T_7 = (\delta_{2,1}(get, R1), t_4)$
$S_8 = \left(\{P', ((get\ R2 \ldots), 0), C, R1, ((B\ get \ldots), t_4)\}, \{R1 \subset B\}, \{PB, BC\}, t_1 + t_2 + t_3 + t_4\right)$
$T_8 = (\delta_{2,2}(get, R2), t_5)$
$S_9 = \left(\{P', ((put\ R1 \ldots), 0), C, ((B\ put \ldots), t_5), R2\}, \{R1 \subset B, R2 \subset B\}, \{PB, BC\}, t_1 + t_2 + t_3 + t_4 + t_5\right)$
$T_9 = (\delta_{3,1}(put, R1), t_6)$
$S_{10} = \left(\{P', ((put\ R2 \ldots), 0), C, R1, ((B\ put \ldots), t_6)\}, \{R1 \subset B, R2 \subset B\}, \{PB, BC\}, t_1 + t_2 + t_3 + t_4 + t_5 + t_6\right)$
$T_{10} = (\delta_{3,2}(put, R2), t_7)$
$S_{11} = \left(\{P', (exit, 0), ((get\ R1 \ldots), t_1 + t_2 + t_3 + t_4 + t_5 + t_6 + t_7), ((C\ get \ldots), t_7), R2\}, \{R1 \subset B, R2 \subset B\}, \{PB, BC\}, t_1 + t_2 + t_3 + t_4 + t_5 + t_6 + t_7\right)$
$T_{11} = (\delta_{4,1}(get, R1), t_8)$
$S_{12} = \left(\{P', B', ((get\ R2 \ldots), 0), (exit, 0), ((C\ get \ldots), t_8)\}, \{R1 \subset B, R2 \subset B\}, \{PB, BC\}, t_1 + t_2 + t_3 + t_4 + t_5 + t_6 + t_7 + t_8\right)$
$T_{12} = (\delta_{4,2}(get, R2), t_9)$
$S_{13} = \left(\{P', B', (exit, 0), R1', (exit, 0)\}, \{R1 \subset B, R2 \subset B\}, \{PB, BC\}, t_1 + t_2 + t_3 + t_4 + t_5 + t_6 + t_7 + t_8 + t_9\right)$
Parallelism
Non-determinism

Execution Model: Paths for PBC



Step 3
Simulation

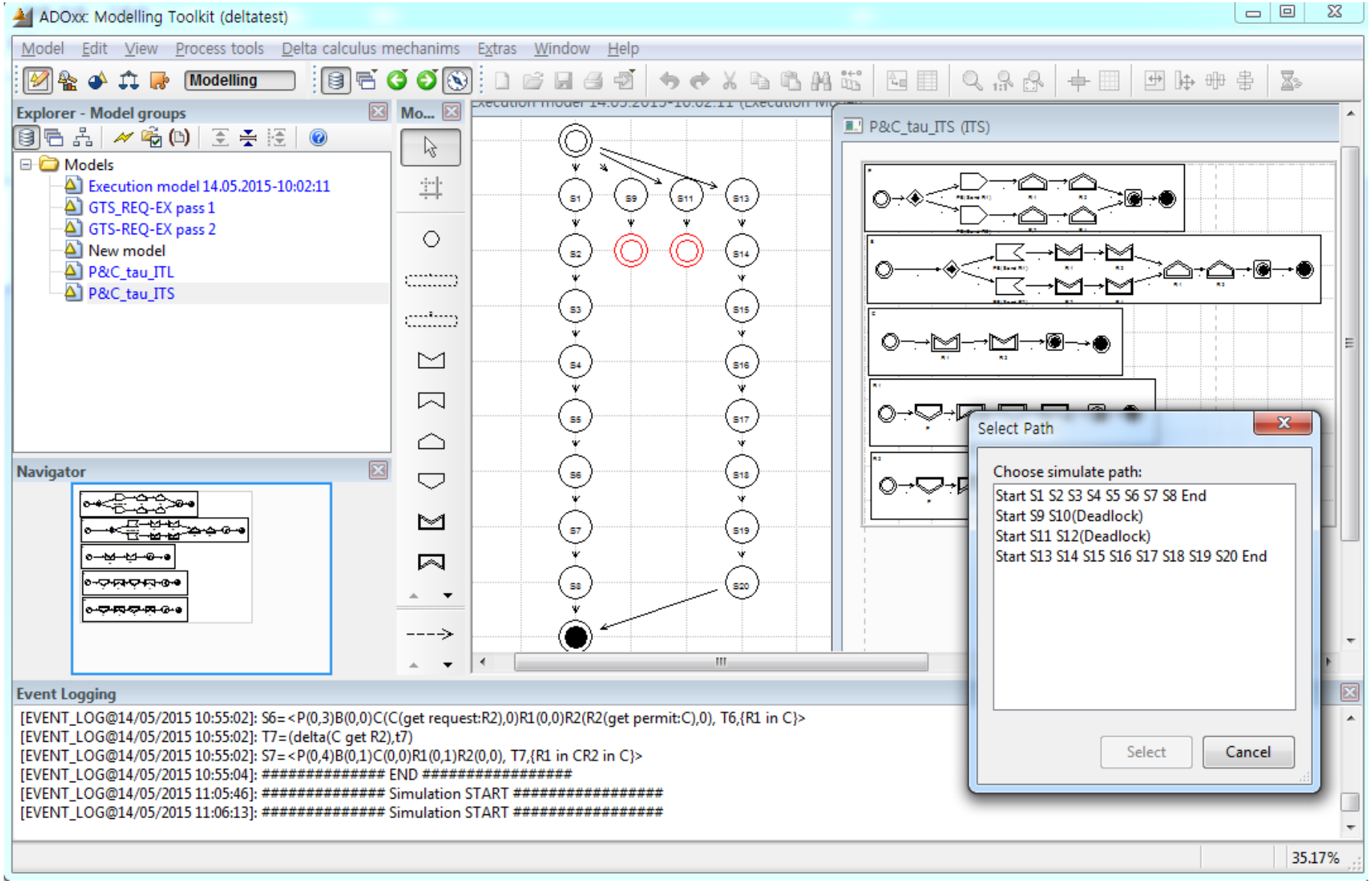

Main Actor:
Domain
Class Diagram
Action Diagram
B(1)
B(n)
Abstract Behavior Modeling
Cardinality Capacity
Behavior Modeling
Regular Expr
B(2)
B(1)
Interpre-tation
Projection
Abst-raction
Operational Specification (δ-Calculus)
Exe. Model Gen
Exe Paths
Sim.
GTS Block Diagrams
Domain Application
Analysis
Verifi-cation
Secure Requirements (GTS Logic)
Assertion Procedures

Algorithm

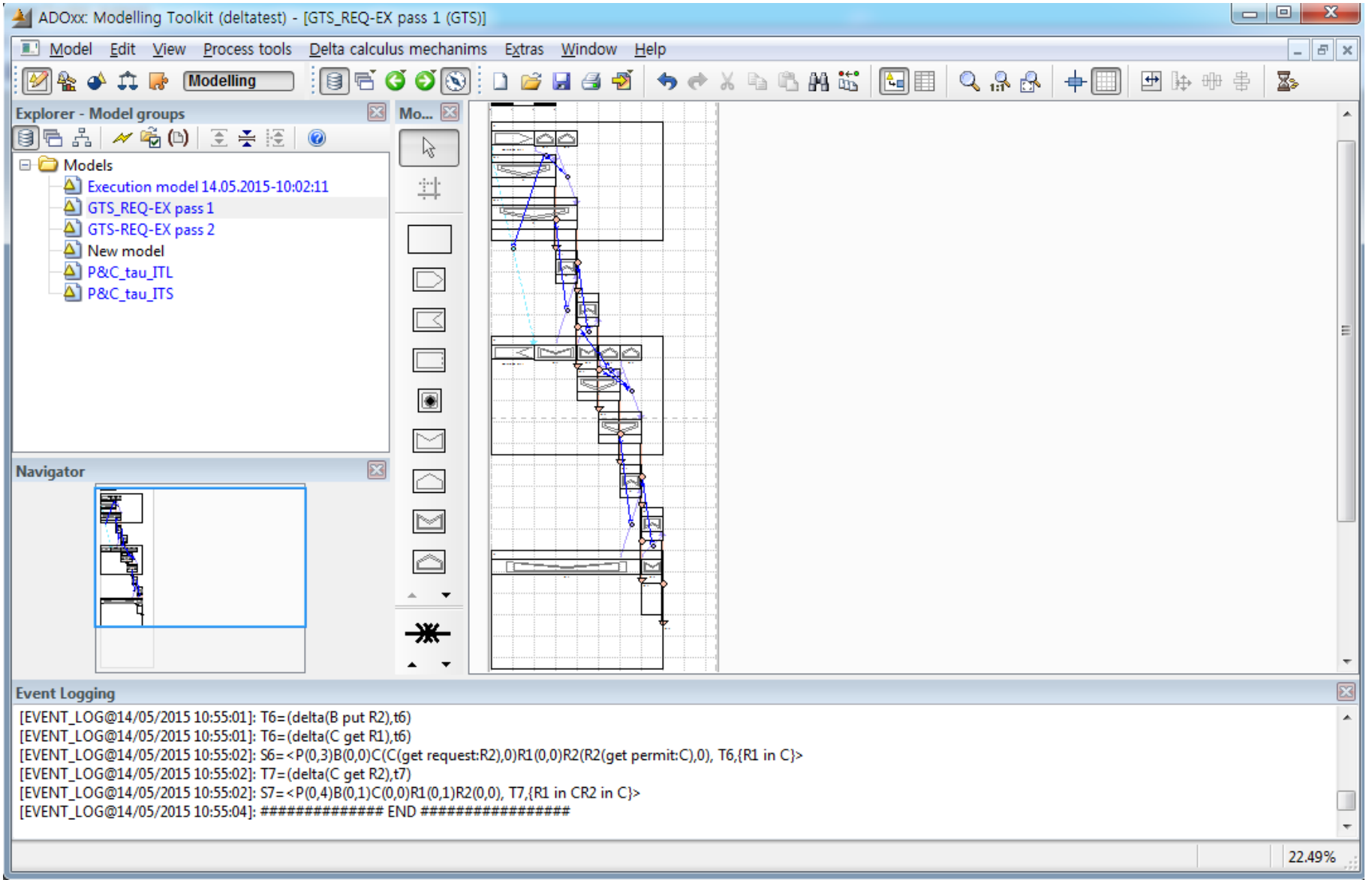
```
Simulation(M: Model)
{
   IF ((M ∈ Models) & (type (M) = ITL))
   {
      Select ITS model of M;
   }
   Else
   {
      Select ITL_M Model of M;
      Select ITS_M Model of M;
   }
   M_ITL = ITL_Loader(ITL_M);        // Load the ITL Model
   M_ITS = ITS_Loader(ITS_M);        // Load the ITS Model
   ITL_ITS_Mapper(M_ITL, M_ITS);    // Map the ITS to the ITL

   {
      P = Select_Simulation_Path(Execution_Model(M));
   } while (!P);

   PI = Load_Path_Info(P);
   Data = Sim_Init(); // Initialization for Simulation Environment

   //Search for Start node and set Start state for all processes
   Sim(Data);
}
```

```
Sim(Data) {
   while()    {
      while (Search Branch)  {    // branch = choice or parallel
         Set the next node environment of Branch;
         IF (choice Node) {
            IF(∃ path info){
               Select a path from path info;
            }
            ELSE {
               Select a Branch after the select-path pop-up window;
            }
         }
      }

      If (All processes are terminated) {
         Exit();
      }

      For(Nodes of the current_node_list) {
         Action(node);
      }
      Update the current_node_list to thenext state;
      Change the node state;   //Standing, Ready, Waiting, Executing,
Deadlock

      IF (Check Deadlock) {
         Exit();
      }
      Time++;
  }
}
```

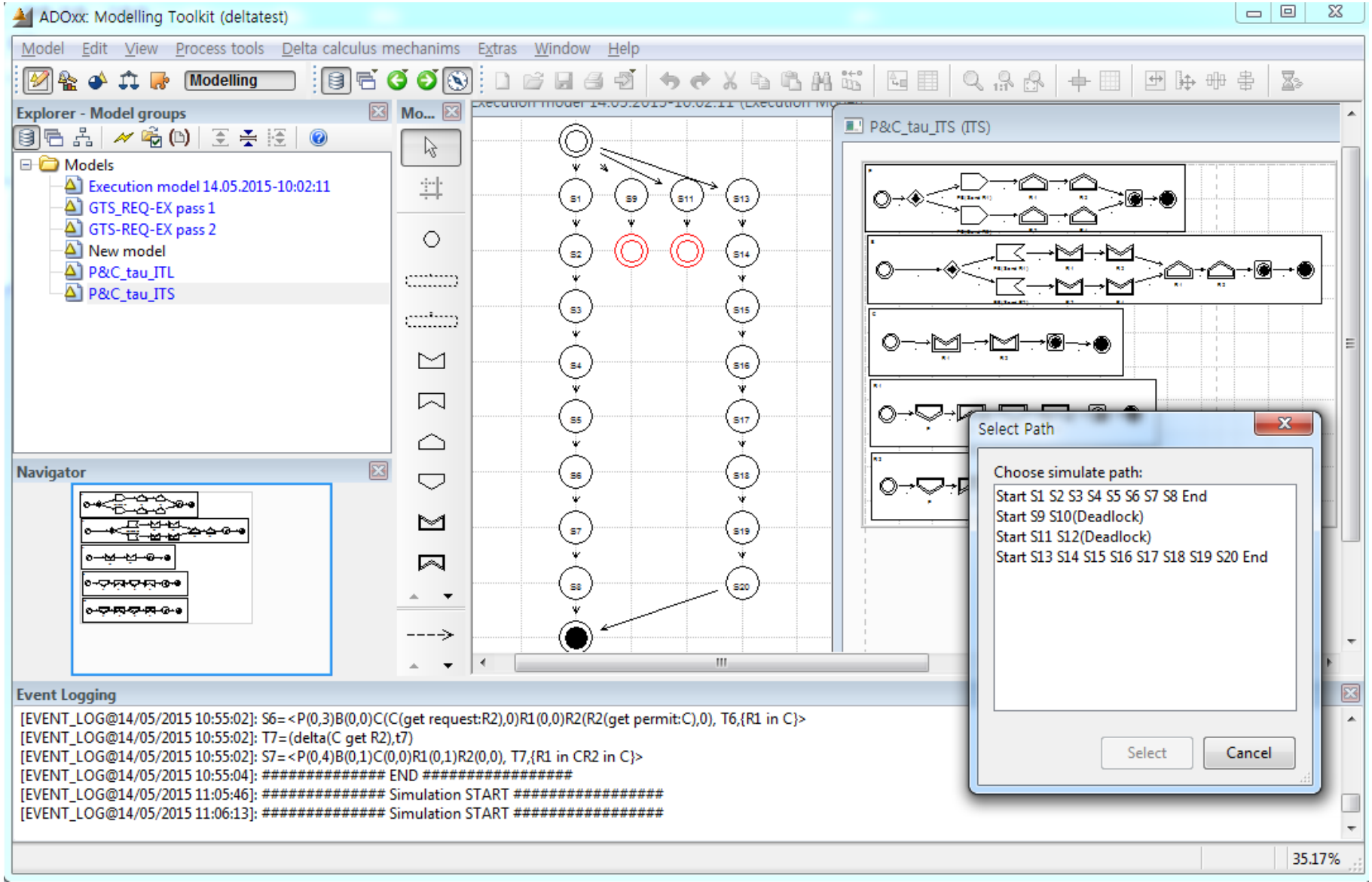
Selecting a Path



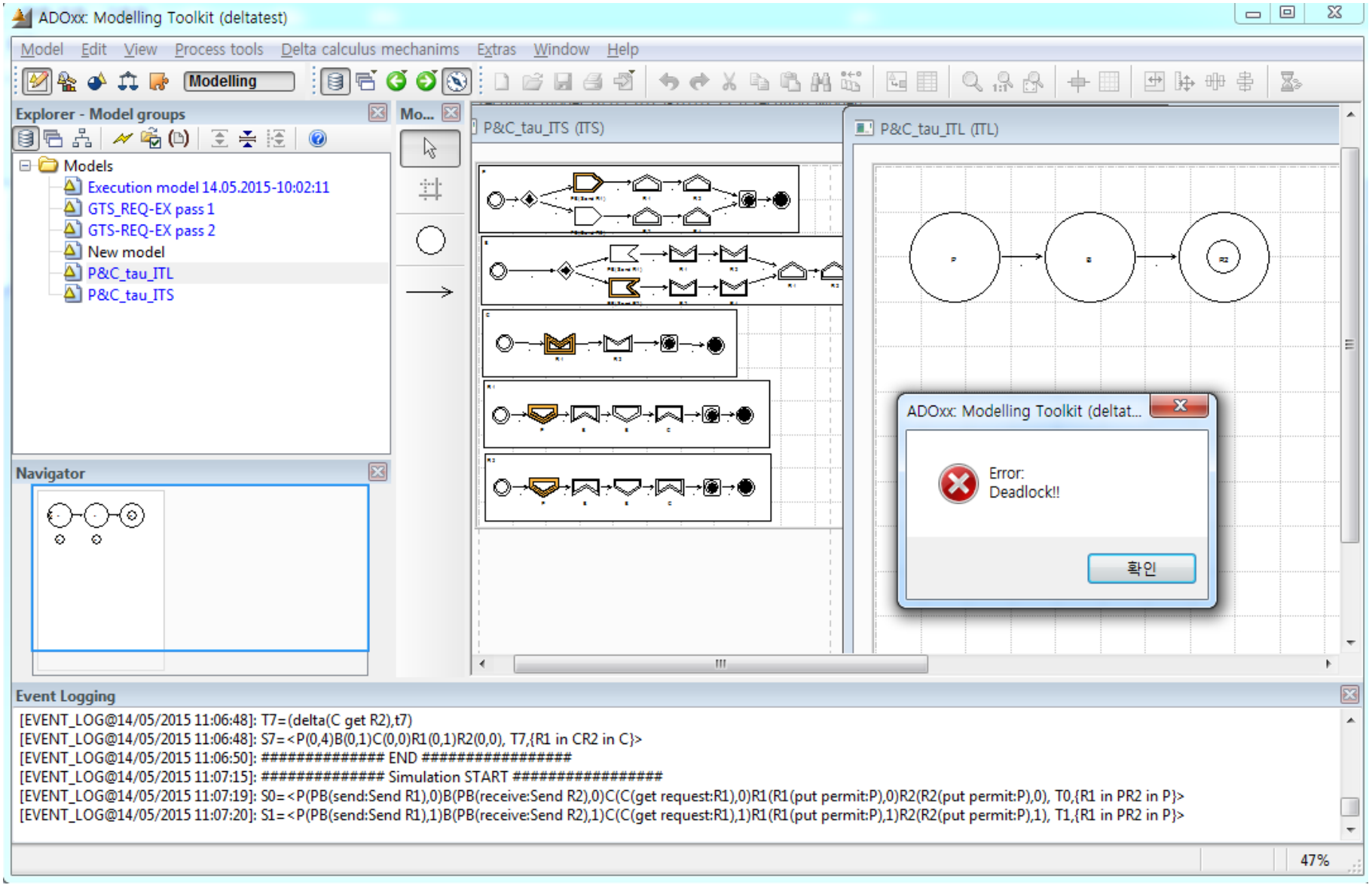
Simulation


ADOxx: Modelling Toolkit (deltatest) - [GTS_REQ-EX pass 1 (GTS)]
Model Edit View Process tools Delta calculus mechanims Extras Window Help
Modelling
Explorer - Model groups
Models
Execution model 14.05.2015-10:02:11
GTS_REQ-EX pass 1
GTS-REQ-EX pass 2
New model
P&C_tau_ITL
P&C_tau_ITS
Navigator
Event Logging
[EVENT_LOG@14/05/2015 10:55:01]: T6=(delta(B put R2),t6)
[EVENT_LOG@14/05/2015 10:55:01]: T6=(delta(C get R1),t6)
[EVENT_LOG@14/05/2015 10:55:02]: S6=<P(0,3)B(0,0)C(C(get request:R2),0)R1(0,0)R2(R2(get permit:C),0), T6,{R1 in C}>
[EVENT_LOG@14/05/2015 10:55:02]: T7=(delta(C get R2),t7)
[EVENT_LOG@14/05/2015 10:55:02]: S7=<P(0,4)B(0,1)C(0,0)R1(0,1)R2(0,0), T7,{R1 in CR2 in C}>
[EVENT_LOG@14/05/2015 10:55:04]: ############### END #################
22.49%

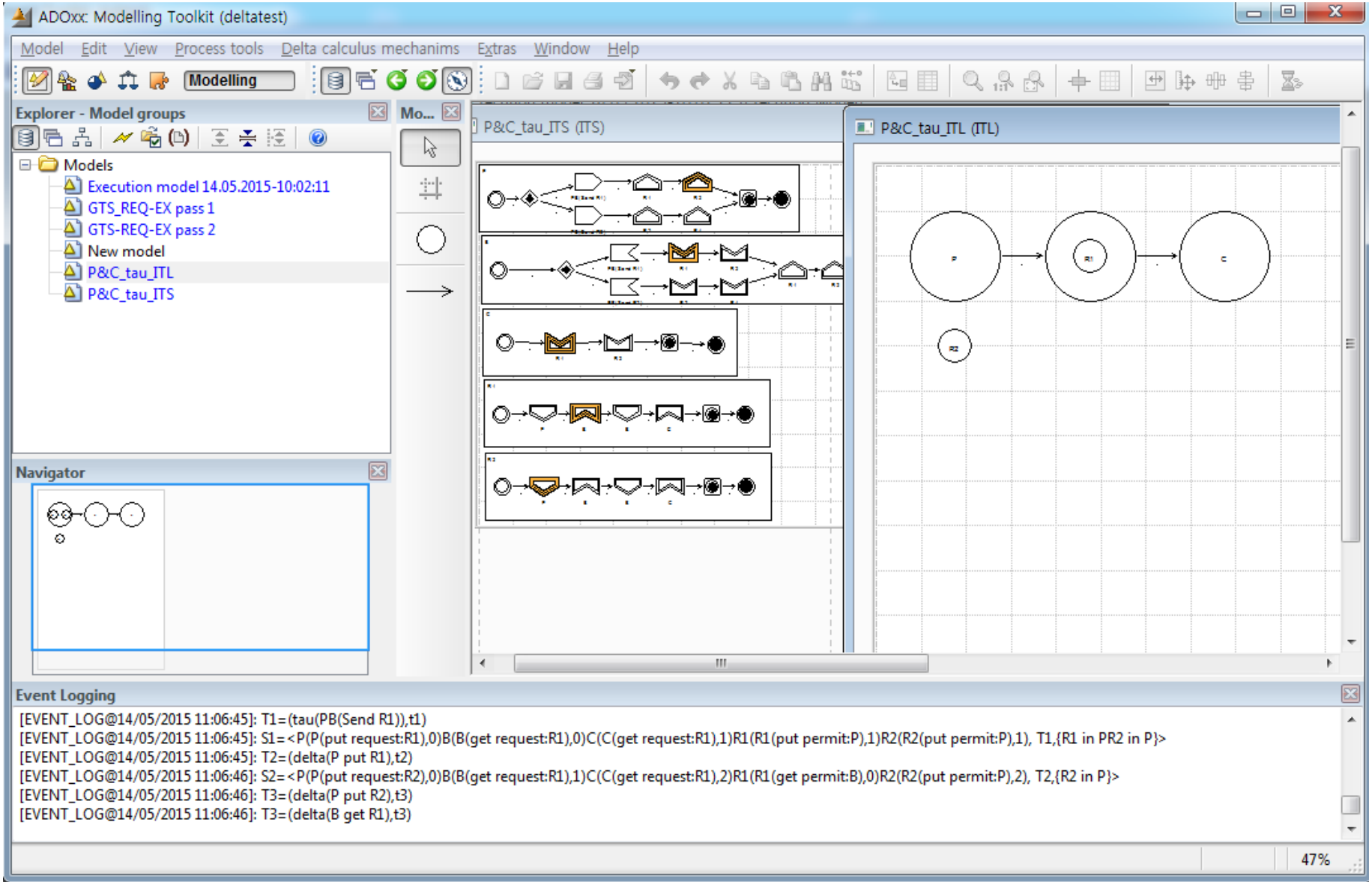
Selecting Another Path



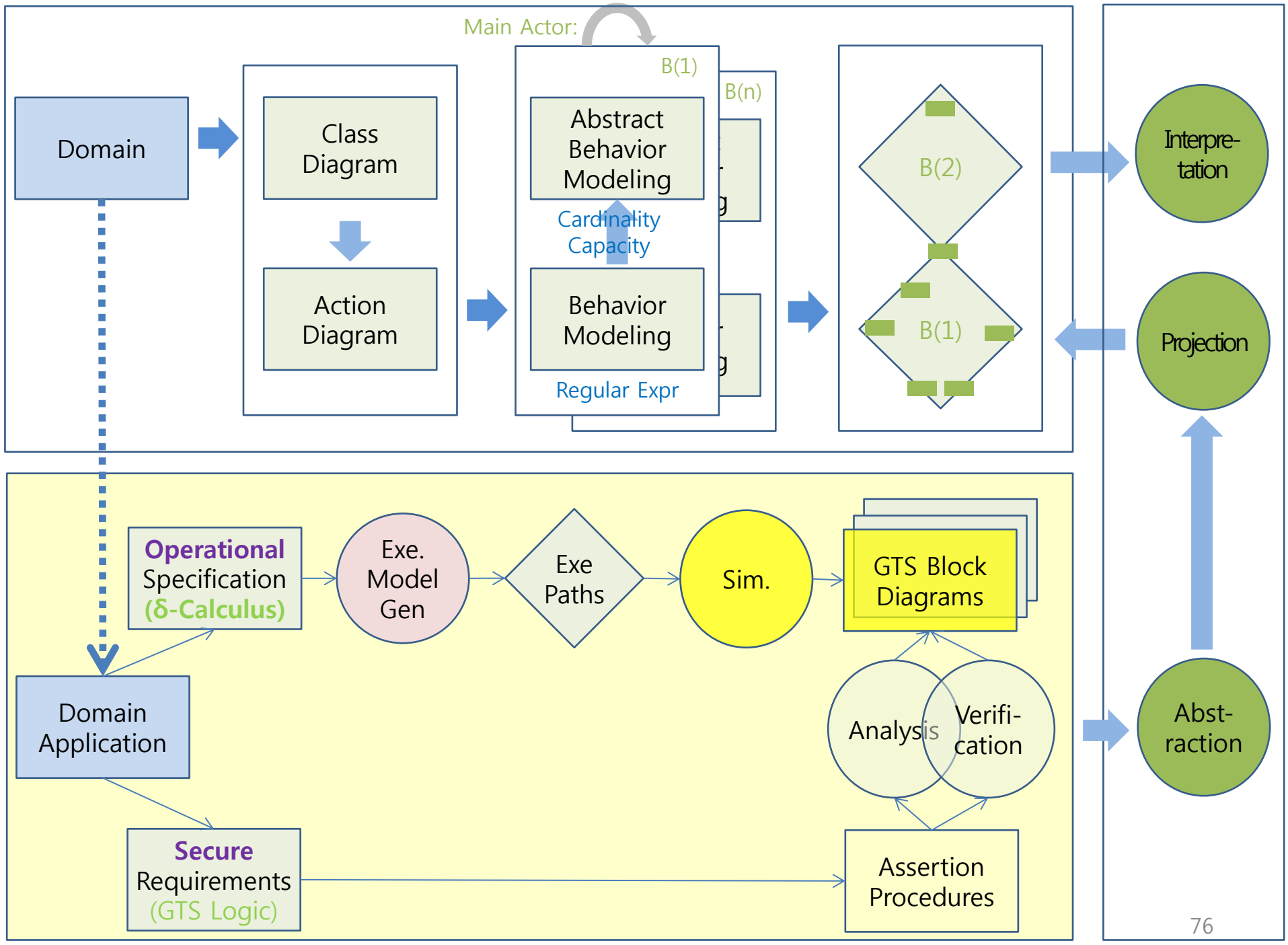
Deadlock


ADOxx: Modelling Toolkit (deltatest)
Model Edit View Process tools Delta calculus mechanims Extras Window Help
Modelling
Explorer - Model groups
Models
Execution model 14.05.2015-10:02:11
GTS_REQ-EX pass 1
GTS-REQ-EX pass 2
New model
P&C_tau_ITL
P&C_tau_ITS
Navigator
P&C_tau_ITS (ITS)
P&C_tau_ITL (ITL)
ADOxx: Modelling Toolkit (deltat...
Error:
Deadlock!!
확인
Event Logging
[EVENT_LOG@14/05/2015 11:06:48]: T7=(delta(C get R2),t7)
[EVENT_LOG@14/05/2015 11:06:48]: S7=<P(0,4)B(0,1)C(0,0)R1(0,1)R2(0,0), T7,{R1 in CR2 in C}>
[EVENT_LOG@14/05/2015 11:06:50]: ############## END #################
[EVENT_LOG@14/05/2015 11:07:15]: ############## Simulation START #################
[EVENT_LOG@14/05/2015 11:07:19]: S0=<P(PB(send:Send R1),0)B(PB(receive:Send R2),0)C(C(get request:R1),0)R1(R1(put permit:P),0)R2(R2(put permit:P),0), T0,{R1 in PR2 in P}>
[EVENT_LOG@14/05/2015 11:07:20]: S1=<P(PB(send:Send R1),1)B(PB(receive:Send R2),1)C(C(get request:R1),1)R1(R1(put permit:P),1)R2(R2(put permit:P),1), T1,{R1 in PR2 in P}>
47%

Deadlock State

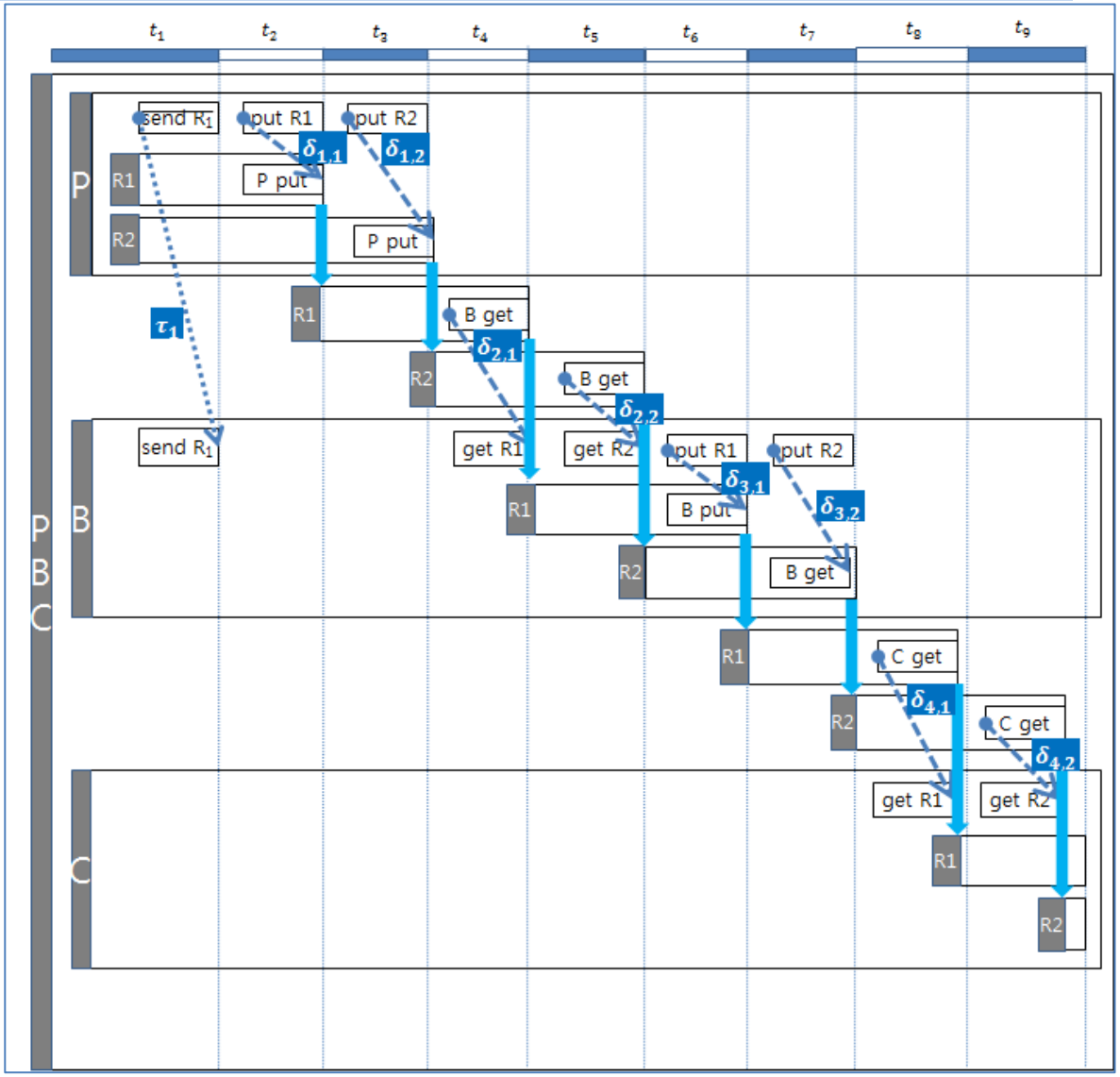


SIMULATION OUTPUT


Main Actor:
Domain
Class Diagram
Action Diagram
B(1)
B(n)
Abstract Behavior Modeling
Cardinality Capacity
Behavior Modeling
Regular Expr
B(2)
B(1)
Interpre-tation
Projection
Abst-raction
Operational Specification (δ-Calculus)
Exe. Model Gen
Exe Paths
Sim.
GTS Block Diagrams
Domain Application
Analysis
Verifi-cation
Secure Requirements (GTS Logic)
Assertion Procedures

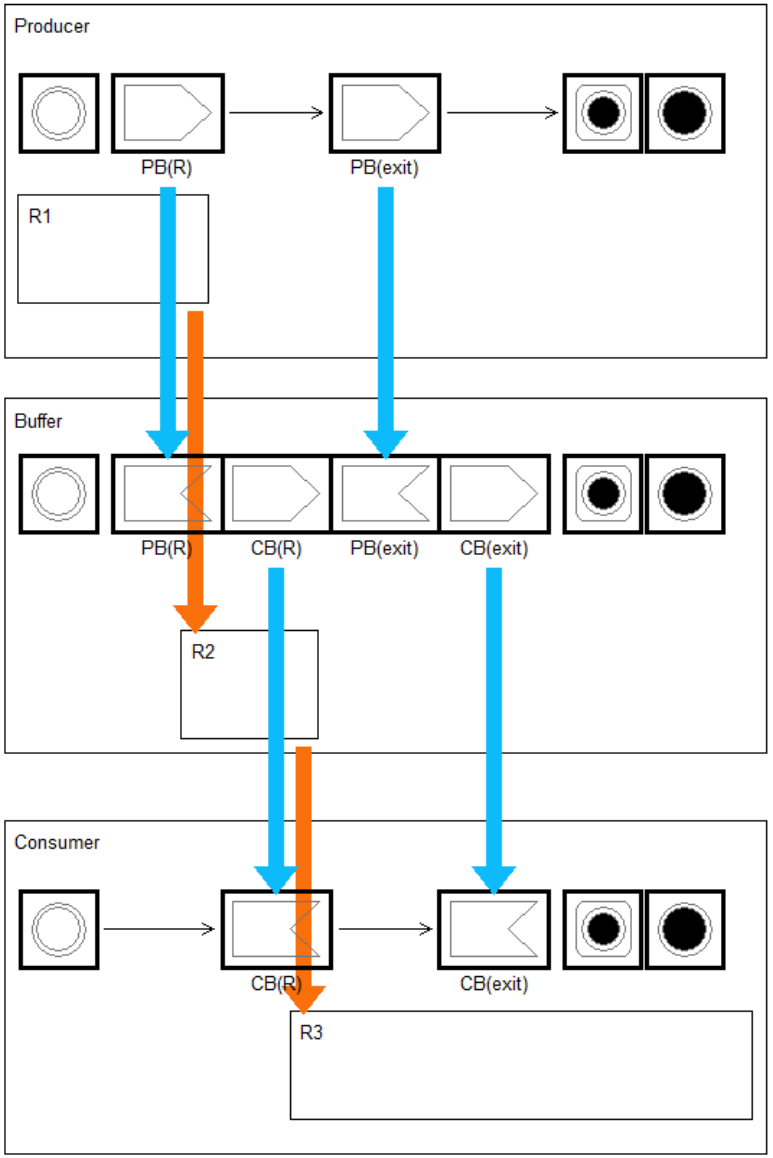
Blocks

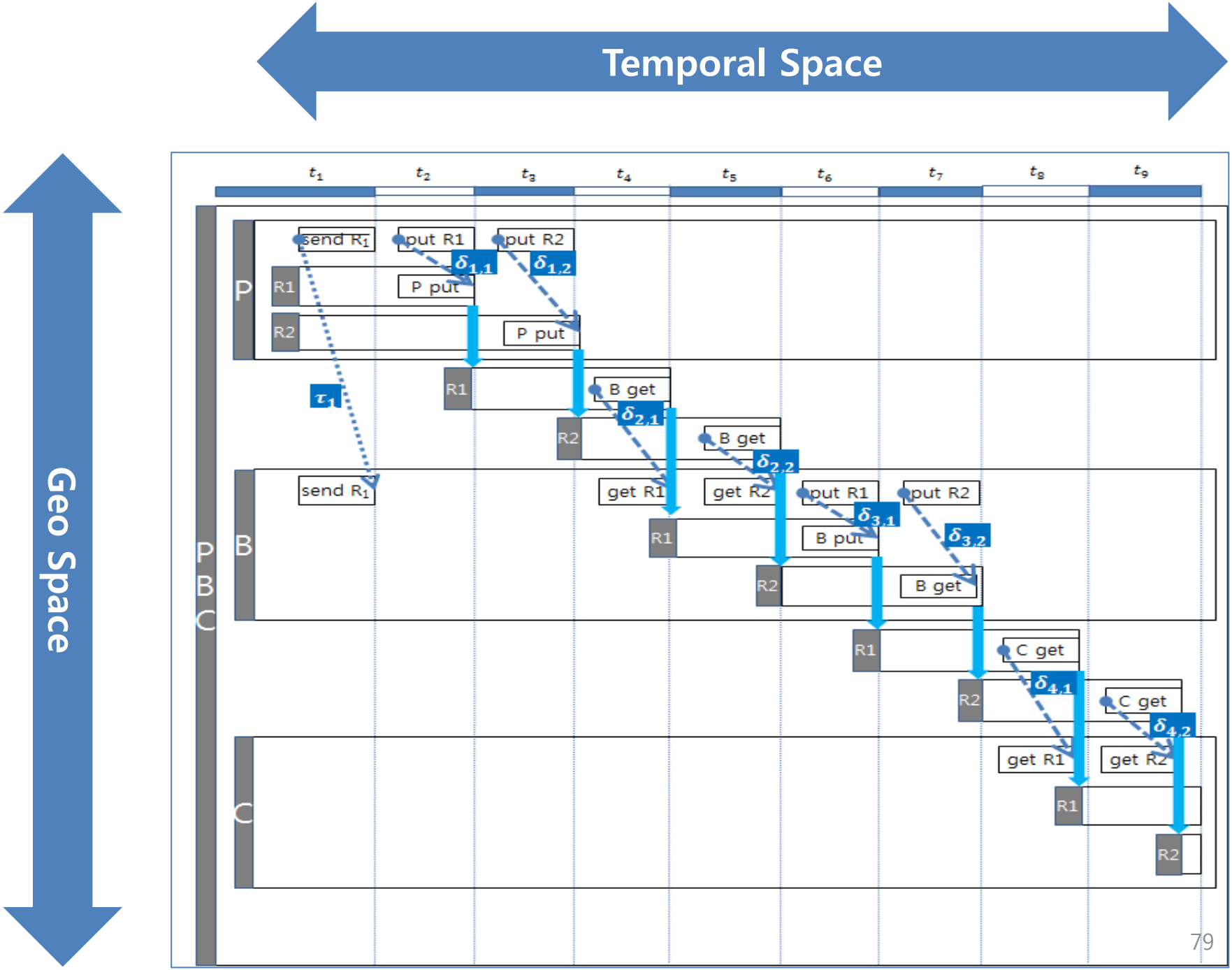
- Block: Space x Time
  - System
  - Processes
  - Actions
  - Interactions
- Interactions
  - Communication
  - Movements
  - Control
- Properties
  - Channel: c_i
  - Message: m_i
  - Priority:n_i

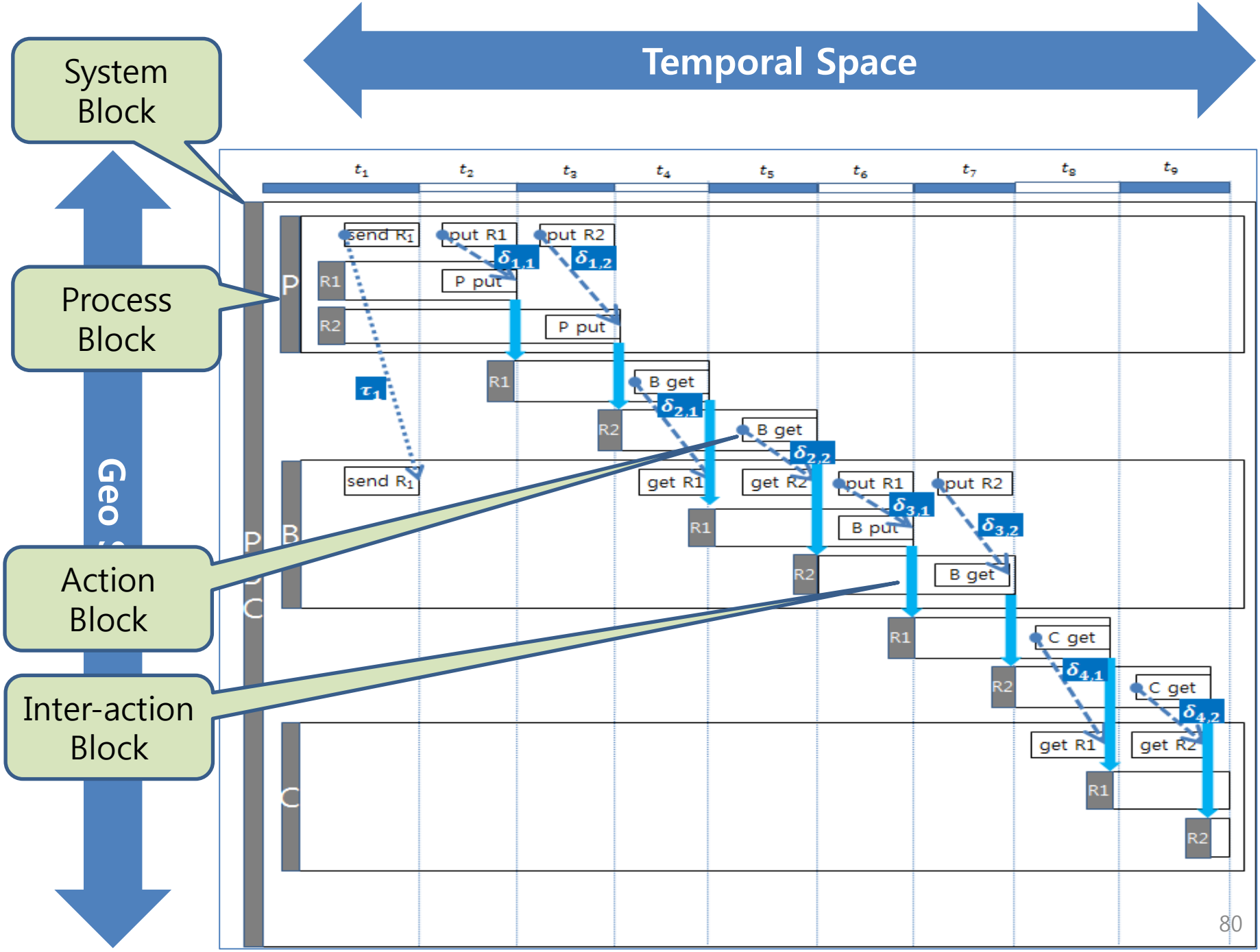


Blocks

- Block: Space x Time
  - System
  - Processes
  - Actions
  - Interactions
- Interactions
  - Communication
  - Movements
  - Control
- Properties
  - Channel: c_i
  - Message: m_i
  - Priority:n_i




Temporal Space
Geo Space
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9
P
B
C
R1
R2
send R1
put R1
put R2
P put
B get
get R1
get R2
B put
C get
$\delta_{1,1}$
$\delta_{1,2}$
$\delta_{2,1}$
$\delta_{2,2}$
$\delta_{3,1}$
$\delta_{3,2}$
$\delta_{4,1}$
$\delta_{4,2}$
τ_1


Temporal Space
System Block
Process Block
Action Block
Inter-action Block
Geo
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9
P
R1
R2
send R1
put R1
put R2
$\delta_{1,1}$
$\delta_{1,2}$
P put
P put
τ_1
B get
$\delta_{2,1}$
B get
$\delta_{2,2}$
P B C
send R1
get R1
get R2
put R1
put R2
$\delta_{3,1}$
$\delta_{3,2}$
B put
B get
C get
$\delta_{4,1}$
C get
$\delta_{4,2}$
get R1
get R2
C

Properties: Restrictions

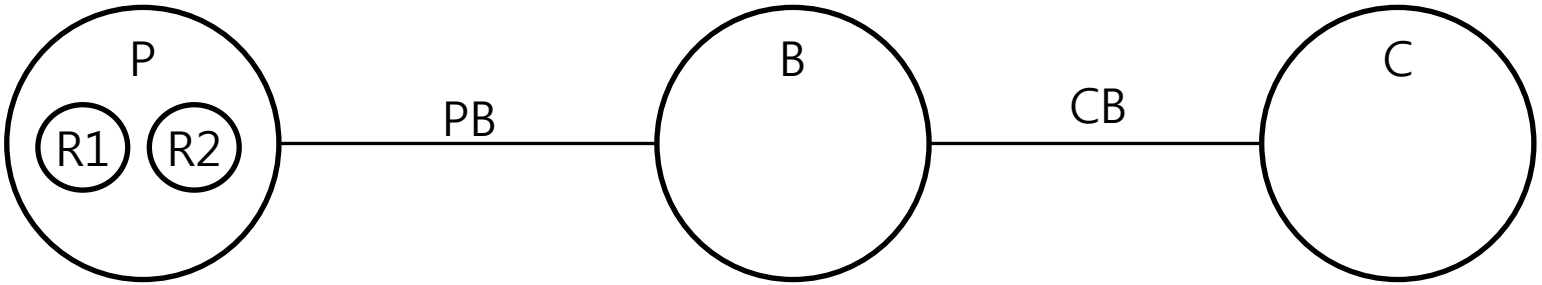
- Syntax
  - Actions in P cannot be overlapped in any space in any time.
  - If $P \parallel Q$, there is no overlap in space, but in time.
  - If $P[Q]$, there is an overlap of Q over P in space during a period of time.
- Semantics
  - τ for communication:
    - Q and P must be in the same time period.
  - δ for movements:
    - For active *in* movement, Q and P must be in the same space at the same time period.
    - For active *out* movement, Q must be in P or P must be in Q at the same time period.
    - For passive *in* movement, Q and P must be in the same space at the same time period.
    - For passive *out* movement, Q must be in P or P must be in Q at the same time period.
  - ρ for process control:
    - For *terminate*, *suspend* and *wake*, Q must be in P or P must be in Q at the same time period.
- Requirements: Dependencies among blocks
  - Inclusions/Precedencies/Duration
  - Conditionals/And/Or/Not

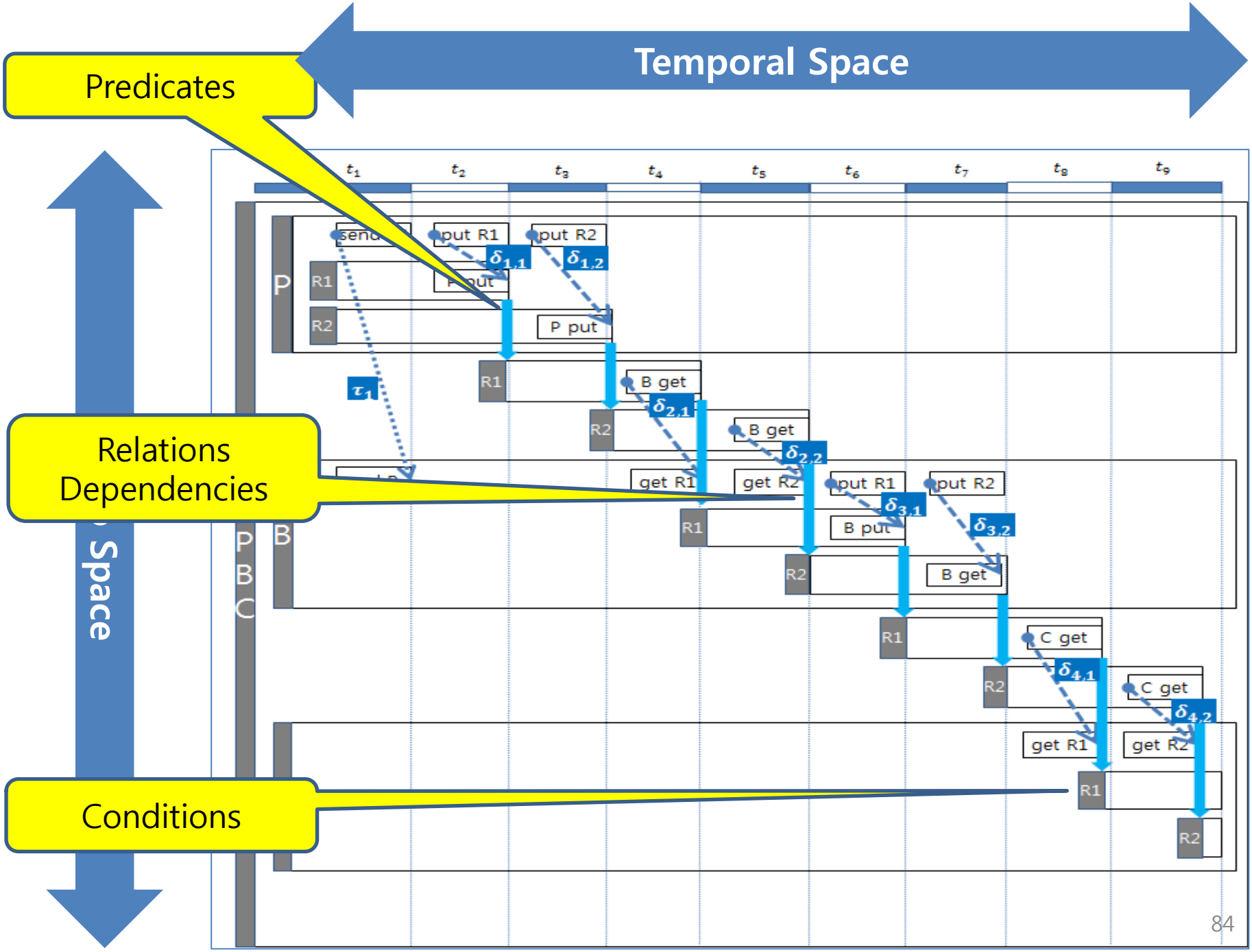
1. Motivation
2. Modeling
   1) Specification: δ-Calculus
   2) Verification: GTS Logic
   3) EMS Example
3. SAVE Tool
4. Cyber-Physical Systems Application
5. Summary
6. Discussion

2.2 GTS Logic: Secure Specification & Verification

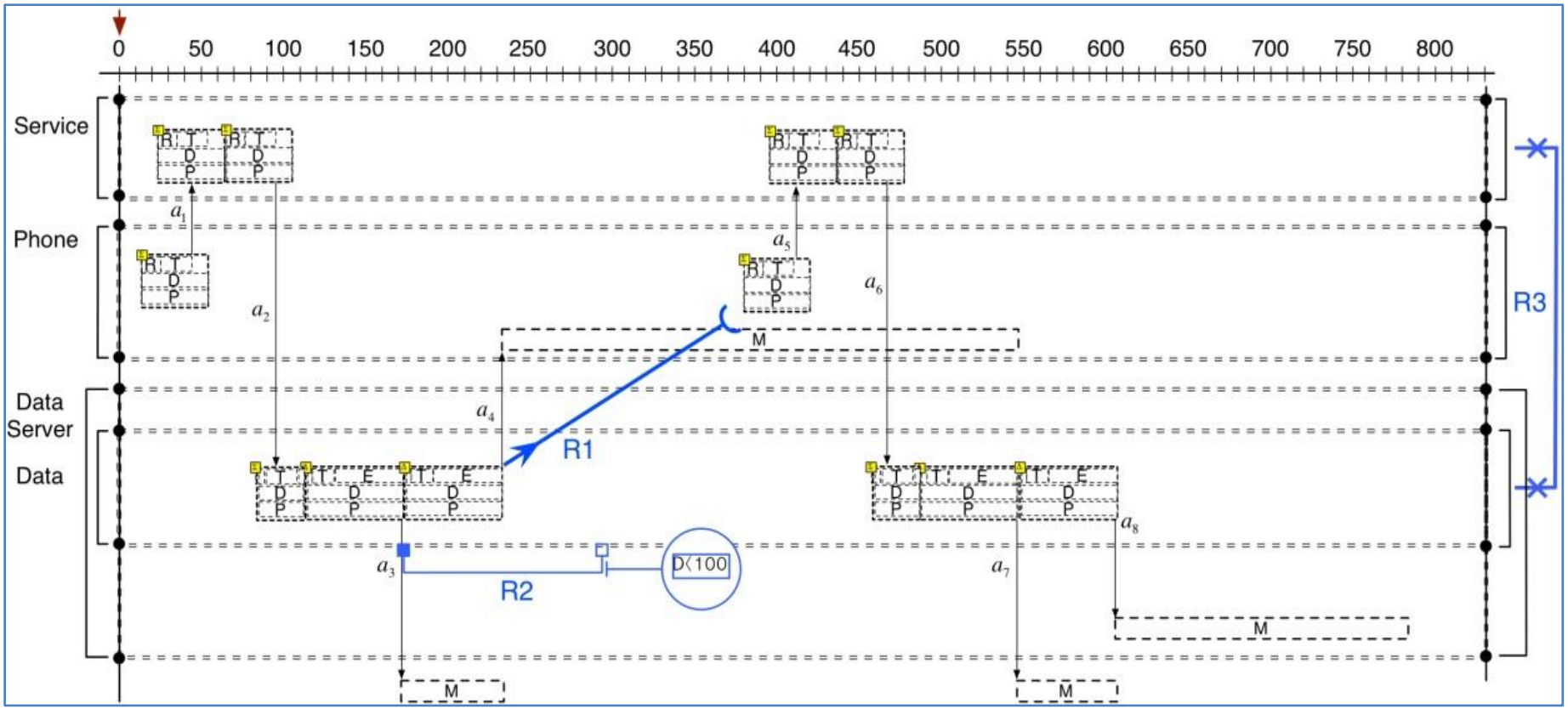
Requirements: PBC Example

- Operational Requirements
  - *Producer* produces two resources, R1 and R2.
  - *Producer* stores the resources in *Buffer* in order.
  - *Producer* informs *Buffer* of the order of R1 and R2, or R2 and R1.
  - *Consumer* consumes the resources from *Buffer* in order.
  - The order of the consumption is formed to *Buffer* by *Consumer*.
- **Secure Requirements**
  - Security:
    - The order should not be violated, since the first resource contains security information to decode the second resource.
    - The propagation between the first and the second should be less than 30 seconds.
  - Safety
    - The resources produced by *Producer* should be consumed by *Consumer* less than 5 minutes.

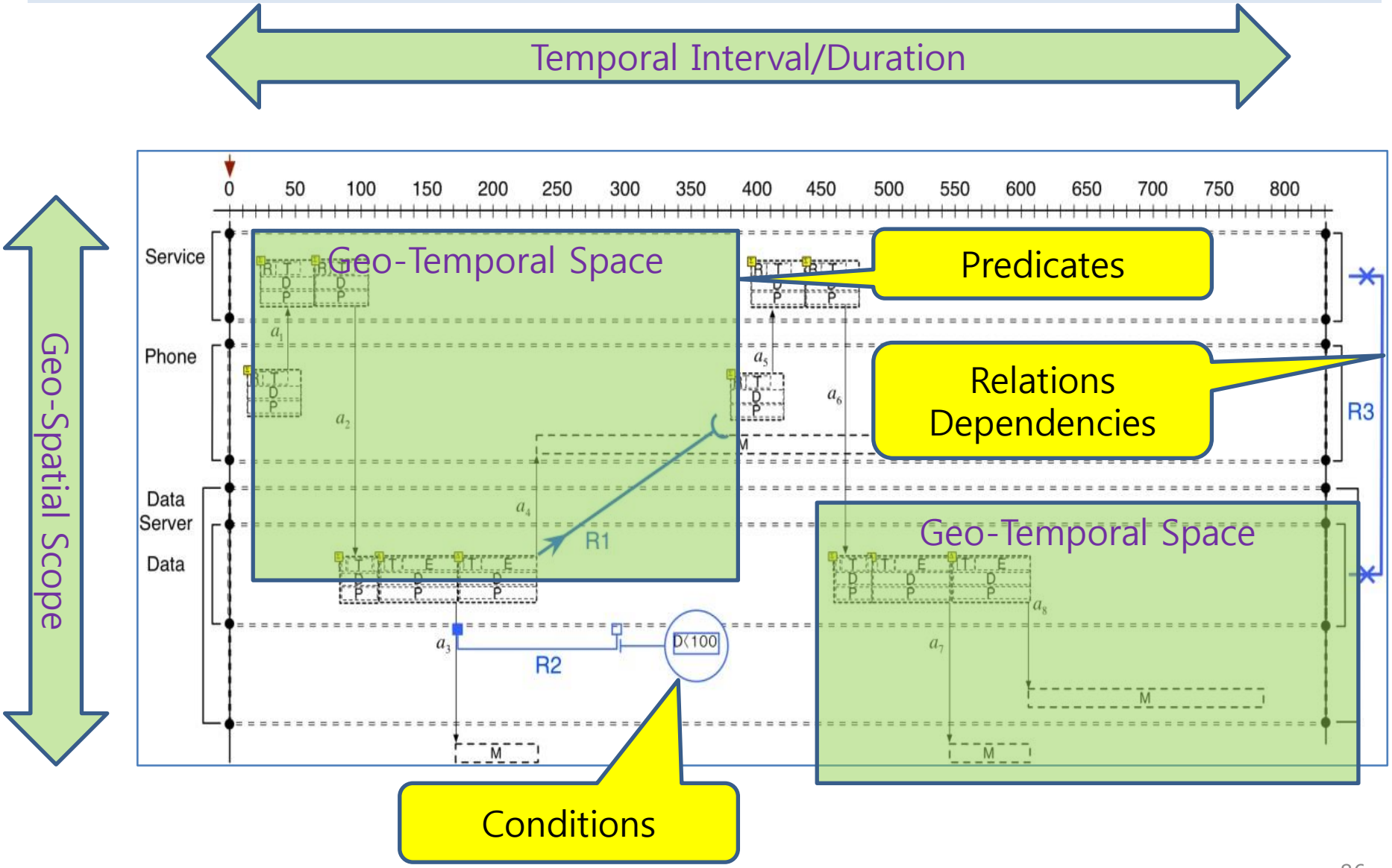



Temporal Space
Predicates
Relations
Dependencies
Conditions
Space
P
B
C
R1
R2
send
put R1
put R2
P put
B get
get R1
get R2
B put
C get
t_1
t_2
t_3
t_4
t_5
t_6
t_7
t_8
t_9
τ_1
$\delta_{1,1}$
$\delta_{1,2}$
$\delta_{2,1}$
$\delta_{2,2}$
$\delta_{3,1}$
$\delta_{3,2}$
$\delta_{4,1}$
$\delta_{4,2}$

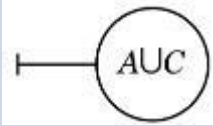
Conceptual Example



Conceptual Example



GTS Visual Logic

	Syntax	Semantics	Syntax	Semantics
Geographical requirements		A can be in B. B can be in A.		A cannot be in B. B cannot be in A.
		A can be in B. B cannot be in A.		B should be in A, always.
		B can be in A. A cannot be in B.		A should be in B, always.
Temporal requirements		Action a should be performed before b in time.		Action a and b can be performed concurrently in the same time period.
		Action b should be performed after a.		Action a and b cannot be performed concurrently in the same time period.
Interval		Open Interval		Close Interval
Conditions		Condition for requirements: time, frequency, behavior, etc.		

Visual Logic: Geo (1)

- 

  - A can be in B. B can be in A
    - $\left(\left(A \sqcap_g B\right) \sqcap_g A = (A \vee \emptyset)\right) \wedge$ $\left(\left(A \sqcap_g B\right) \sqcap_g B = (B \vee \emptyset)\right)$
- 

  - A can be in B. B cannot be in A
    - $\left(\left(A \sqcap_g B\right) \sqcap_g A = \emptyset\right) \wedge \left(\left(A \sqcap_g B\right) \sqcap_g B = (B \vee \emptyset)\right)$

Visual Logic: : Geo (2)

- 

  - A cannot be in B. B can be in A
  - $\left(\left(A \sqcap_g B\right) \sqcap_g A = (A \vee \emptyset)\right) \wedge \left(\left(A \sqcap_g B\right) \sqcap_g B = \emptyset\right)$
- 

  - A cannot be in B. B cannot be in A
  - $\left(\left(A \sqcap_g B\right) \sqcap_g A = \emptyset\right) \wedge \left(\left(A \sqcap_g B\right) \sqcap_g B = \emptyset\right)$

Visual Logic : Geo (3)

- 

  - B should be in A, always.
  - $B \subset A$
- 

  - A should be in B, always.
  - $A \subset B$

Visual Logic : Temporal (1)

- 

  - Action a should be performed before b in time.
  - $(a \prec b) \vee (a \preceq b)$
- 

  - Action a should be performed after b in time.
  - $(b \prec a) \vee (b \preceq a)$

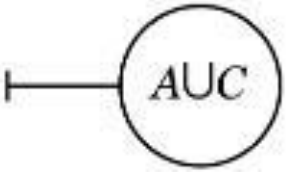
Visual Logic : Temporal (2)

- 

  - Action a and b can be performed concurrently in the same time period.
  - $a \mathbin{\hat{=}} b$
- 

  - Action a and b cannot be performed concurrently in the same time period.
  - $a \sqcap_t b = \emptyset$

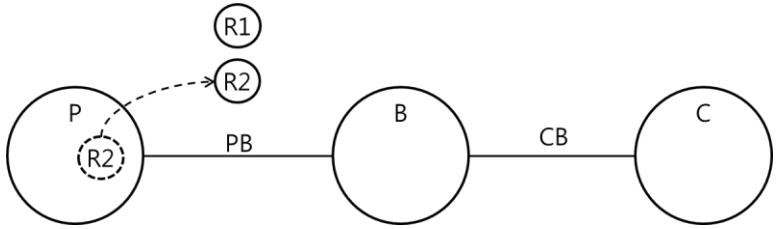
Visual Logic: Interval & Conditions

- 
  - Open interval
  - (t_1, t_2)
- 
  - Close interval
  - $[t_1, t_2]$
- 

  - Conditions
  - Depends on conditions.

Interactions

- τ: Communication
  - $\tau_1 = \left(P\colon PB, \left(\overline{Send\ R1}\right), B\colon PB(Send\ R1)\right)$
  - $\tau_2 = \left(P\colon PB, \left(\overline{Send\ R2}\right), B\colon PB(Send\ R2)\right)$
- δ: Movements
  - $\delta_{1,1} = (P\colon put\ R1, R1\colon P\ put)$
  - $\delta_{1,2} = (P\colon put\ R2, R2\colon P\ put)$
  - $\delta_{2,1} = (B\colon get\ R1, R1\colon B\ get)$
  - $\delta_{2,2} = (B\colon get\ R2, R2\colon P\ get)$
  - $\delta_{3,1} = (B\colon put\ R1, R1\colon B\ put)$
  - $\delta_{3,2} = (B\colon put\ R2, R2\colon B\ put)$
  - $\delta_{4,1} = (C\colon get\ R1, R1\colon C\ get)$
  - $\delta_{4,2} = (C\colon get\ R2, R2\colon C\ get)$



Requirements: GTS Logic

$$Rq1 = \tau_1 \rightarrow \forall i: \left(\delta_{i,1} < \delta_{i,2}\right)\ [i = 1,2]$$

$$Rq2 = \tau_2 \rightarrow \forall i: \left(\delta_{i,2} < \delta_{i,2}\right)\ [i = 1,2]$$

$$Rq3 = \tau_1 \vee \tau_2 < \delta_{1,1} \wedge \delta_{1,2}$$

$$Rq4 = \delta_{1,1} < B: (get\ R1)$$

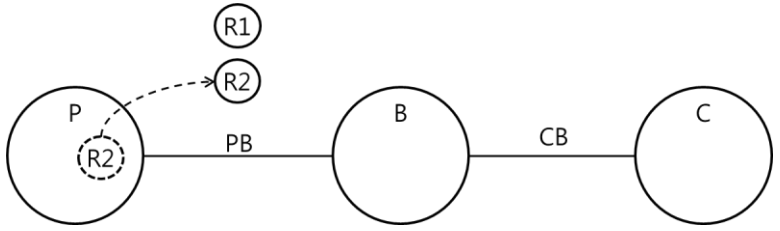
$$Rq5 = \delta_{1,2} < B: (get\ R2)$$

$$Rq6 = \delta_{2,1} < B: (put\ R1)$$

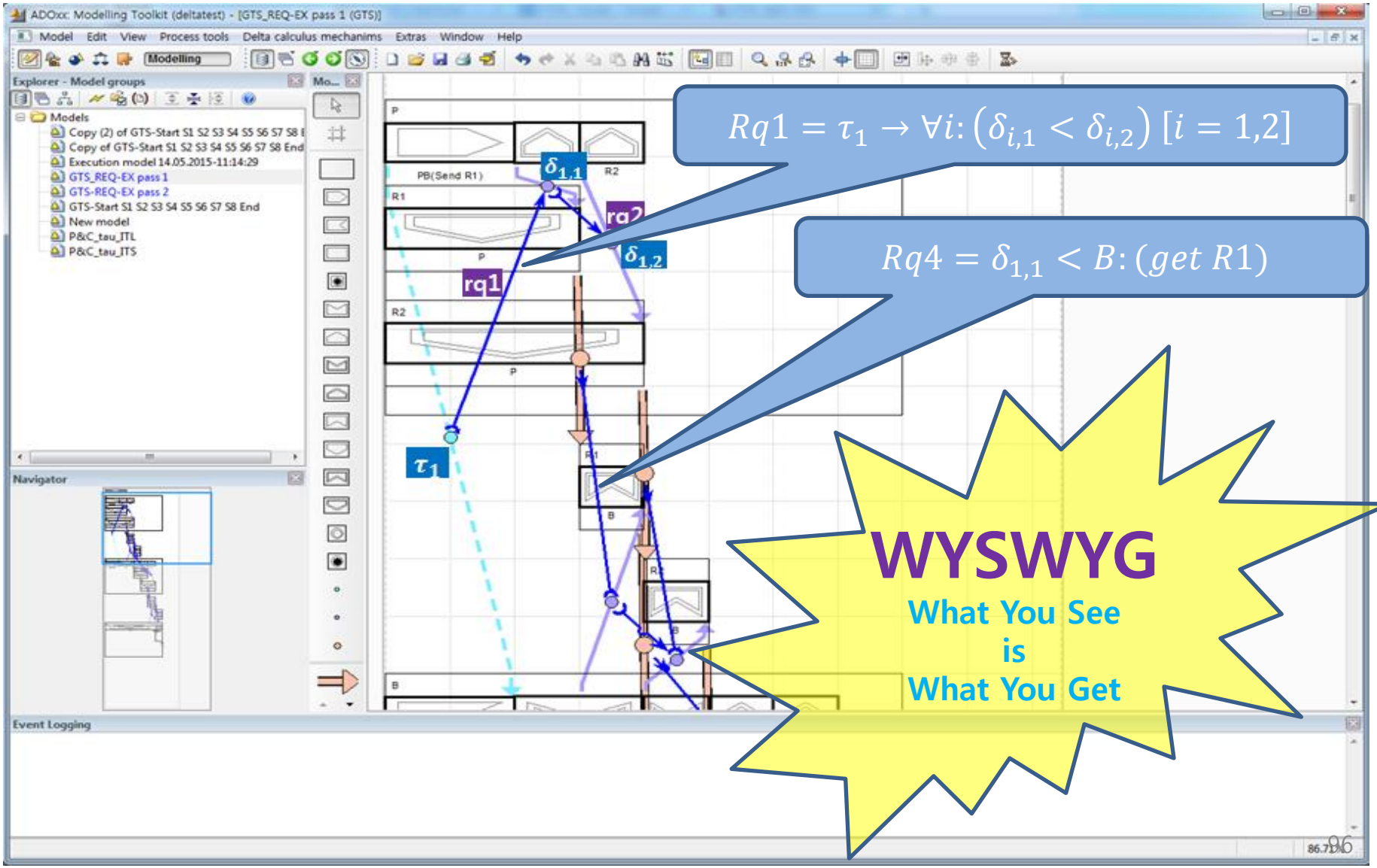
$$Rq7 = \delta_{2,2} < B: (put\ R2)$$

$$Rq8 = \delta_{3,1} < C: (get\ R1)$$

$$Rq9 = \delta_{3,2} < C: (get\ R2)$$

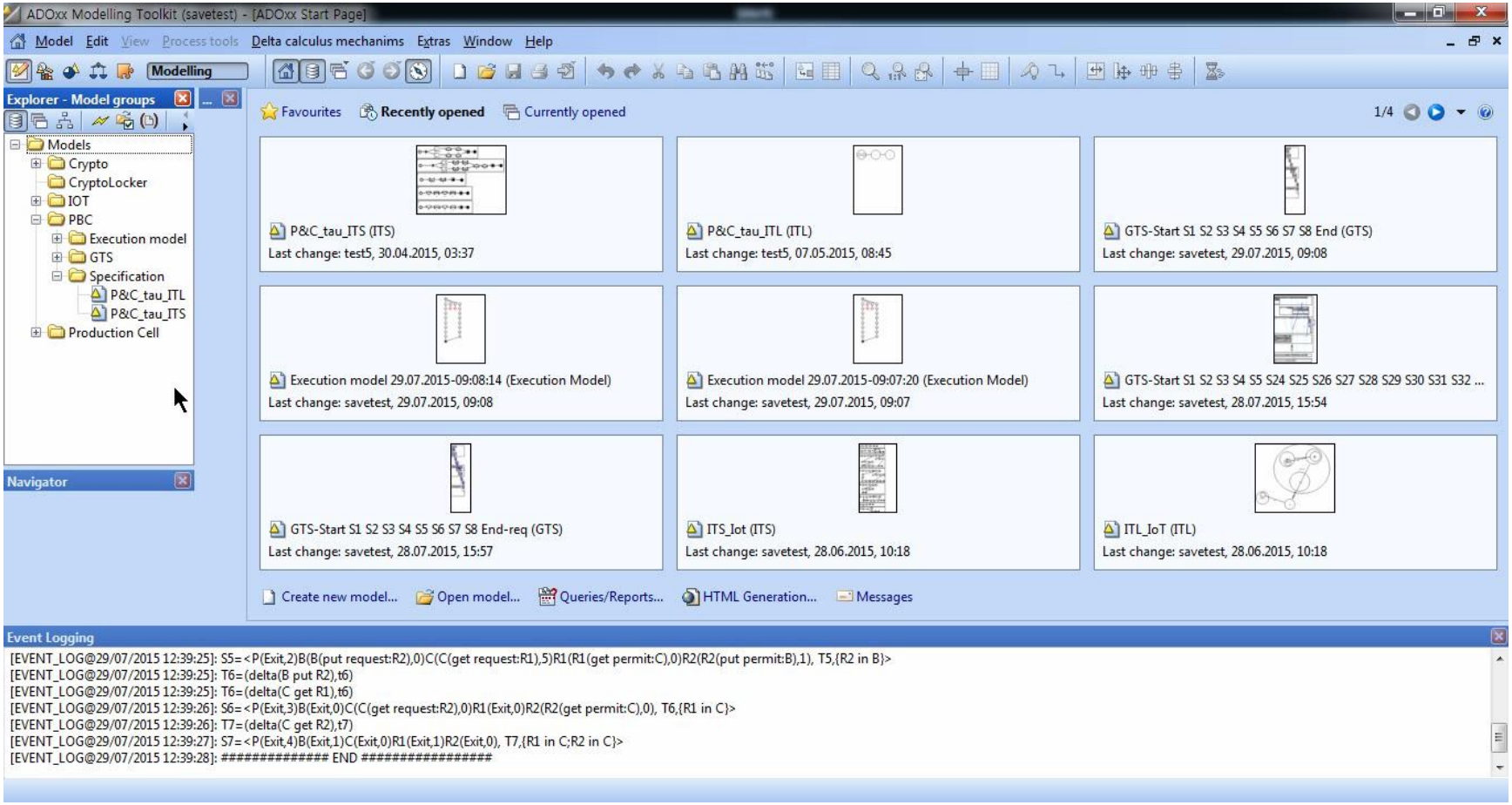


Requirements Verification



SAVE

DEMO: PBC

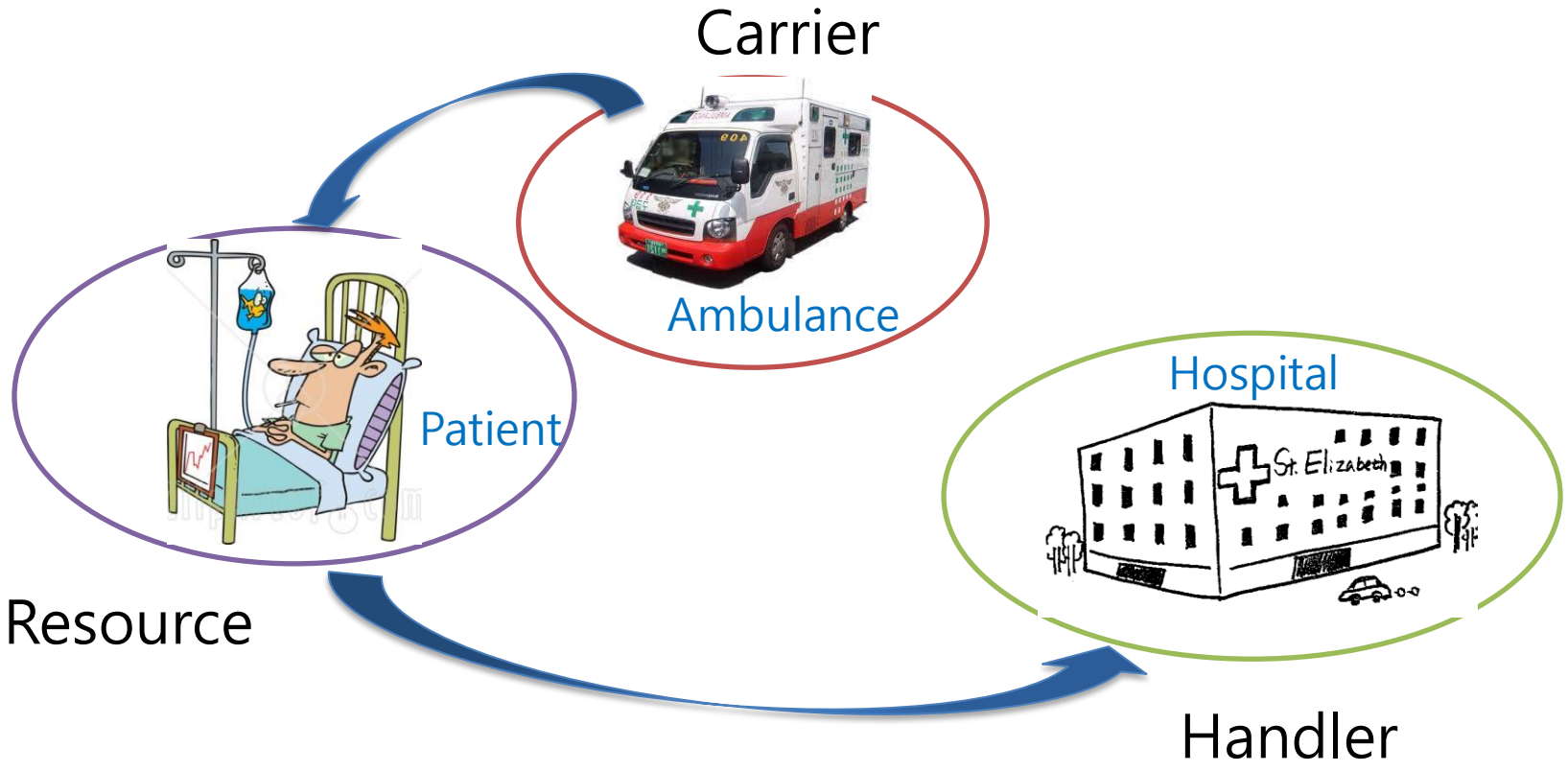

ADOxx Modelling Toolkit (savetest) - [ADOxx Start Page]
Model Edit View Process tools Delta calculus mechanims Extras Window Help
Modelling
Explorer - Model groups
Models
Crypto
CryptoLocker
IOT
PBC
Execution model
GTS
Specification
P&C_tau_ITL
P&C_tau_ITS
Production Cell
Navigator
Favourites
Recently opened
Currently opened
1/4
P&C_tau_ITS (ITS)
Last change: test5, 30.04.2015, 03:37
P&C_tau_ITL (ITL)
Last change: test5, 07.05.2015, 08:45
GTS-Start S1 S2 S3 S4 S5 S6 S7 S8 End (GTS)
Last change: savetest, 29.07.2015, 09:08
Execution model 29.07.2015-09:08:14 (Execution Model)
Last change: savetest, 29.07.2015, 09:08
Execution model 29.07.2015-09:07:20 (Execution Model)
Last change: savetest, 29.07.2015, 09:07
GTS-Start S1 S2 S3 S4 S5 S24 S25 S26 S27 S28 S29 S30 S31 S32 ...
Last change: savetest, 28.07.2015, 15:54
GTS-Start S1 S2 S3 S4 S5 S6 S7 S8 End-req (GTS)
Last change: savetest, 28.07.2015, 15:57
ITS_Iot (ITS)
Last change: savetest, 28.06.2015, 10:18
ITL_IoT (ITL)
Last change: savetest, 28.06.2015, 10:18
Create new model...
Open model...
Queries/Reports...
HTML Generation...
Messages
Event Logging
[EVENT_LOG@29/07/2015 12:39:25]: S5=<P(Exit,2)B(B(put request:R2),0)C(C(get request:R1),5)R1(R1(get permit:C),0)R2(R2(put permit:B),1), T5,{R2 in B}>
[EVENT_LOG@29/07/2015 12:39:25]: T6=(delta(B put R2),t6)
[EVENT_LOG@29/07/2015 12:39:25]: T6=(delta(C get R1),t6)
[EVENT_LOG@29/07/2015 12:39:26]: S6=<P(Exit,3)B(Exit,0)C(C(get request:R2),0)R1(Exit,0)R2(R2(get permit:C),0), T6,{R1 in C}>
[EVENT_LOG@29/07/2015 12:39:26]: T7=(delta(C get R2),t7)
[EVENT_LOG@29/07/2015 12:39:27]: S7=<P(Exit,4)B(Exit,1)C(Exit,0)R1(Exit,1)R2(Exit,0), T7,{R1 in C;R2 in C}>
[EVENT_LOG@29/07/2015 12:39:28]: ############### END #################

1. Motivation
2. Modeling
   1) Specification: δ-Calculus
   2) Verification: GTS Logic
   3) EMS Example
3. SAVE Tool
4. Cyber-Physical Systems Application
5. Summary
6. Discussion

2.3 Example: Emergency Medical System (EMS)

Emergency Medical Systems

- Main theme
  - "Ambulances transport patients to hospitals."

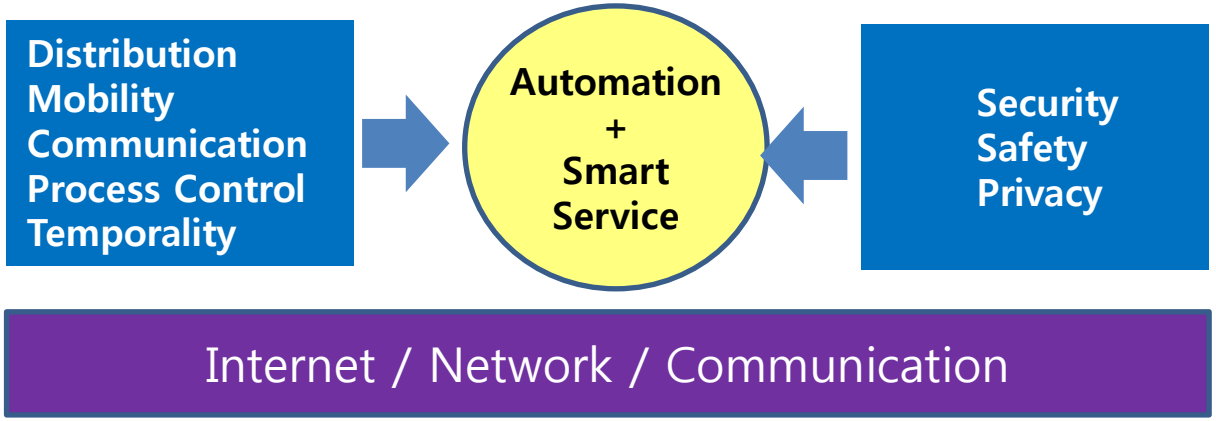


Operational Requirements

- Patients should call Control Center of their situations.
- The center orders 911 or Hospitals to handle the patients with Ambulances.
- Ambulances go to the locations where the patients are, and transport them to Hospitals.
- Ambulances follow orders from 911 or Hospitals.
- Ambulances call Hospitals to let Doctors to be prepared to treat Patients.

Smart EMS / IoT

- Operational Requirements
  - Patients should call Control Center of their situations.
  - The center orders 911 or Hospitals to handle the patients with Ambulances.
  - Ambulances go to the locations where the patients are, and transport them to Hospitals.
  - Ambulances follow orders from 911 or Hospitals.
  - Ambulances call Hospitals to let Doctors to be prepared to treat Patients.



Configuration: 35 Processes

- Main theme: System (1)
  - Control Center: 1
  - Carrier: 3 Ambulances
    - A in Hospital A
    - B, C in 911 Center
  - Sources: 5 Places
    - 4 Houses: A, B, C, D
    - 1 School: E
  - Resources: 8 Patients
    - 1 Patient w/ heart disease in House A
    - 2 Patient w/ high blood pressure in House B, C.
    - 1 Patient w/ diabetes in House D
    - 1 Patient w/ high blood pressure in School E
    - 2 Patient w/ food poisoning in School E
  - Destination: 3 Hospitals
    - A: 1 Lounge; 3 Doctors; 1 Surgery
    - B: 1 Lounge; 3 Doctors ; 1 Surgery
    - C: 1 Lounge; 2 Doctors ; 1 Surgery

Specification (1): 5

$$CS = ((CALL(HAHBP).ORDER(\overline{HAHBP}).CALL(HDHD).ORDER(\overline{HDHD}))$$
$$+(CALL(HDHD).ORDER(\overline{HDHD}).CALL(HAHBP).ORDER(\overline{HAHBP})).$$
$$CALL(SCFP1).CALL(SCFP2).CALL(SCHD).CALL(SCHBP).ORDER(\overline{ALL1}).ORDER(\overline{ALL2}).$$
$$((CALL(HBHBP).ORDER(\overline{HBHBP}).CALL(HCHD).ORDER(\overline{HCHD}))$$
$$+(CALL(HCHD).ORDER(\overline{HCHD}).CALL(HBHBP).ORDER(\overline{HBHBP})).\emptyset^{\infty}$$

$$911 = ORDER(HDHD).((AmbA(\overline{HD}).AmbA\ out.ORDER(ALL1).AmbB(\overline{SC}).AmbB\ out.ORDER(HCHD).AmbA(\overline{HC}))$$
$$\oplus^{1}(AmbB(\overline{HD}).AmbB\ out.ORDER(ALL1).AmbA(\overline{SC}).AmbA\ out.ORDER(HCHD).AmBA(\overline{HC})).AmbA\ in.AmbB\ in.\emptyset^{\infty}$$

$$HospitalA = ORDER(HAHBP).AmbC(\overline{HA}).AmbC\ out.CALL(Arrive1).HA(\overline{Ready1}).AmbC\ in.ORDER(ALL2).AmbC(\overline{SC}).$$
$$AmbC\ out.ORDER(HBHBP).AmbC(\overline{HB}).CALL(Arrive2).HA(\overline{Ready2}).HA(\overline{Ready3}).AmbC\ in.\emptyset^{\infty}$$

$$Hospital\ B = CALL(Arrive3).HB(\overline{Ready1}).((AmbA\ in.AmbA\ out)\oplus^{1}(AmbB\ in.AmbB\ out)).$$
$$CALL(Arrive4).HB(\overline{Ready2}).((AmbB\ in.AmbB\ out)\oplus^{1}(AmbA\ in.AmbA\ out)).$$
$$CALL(Arrive5).HB(\overline{Ready3}).((AmbA\ in.AmbA\ out)\oplus^{1}(AmbB\ in.AmbB\ out)).\emptyset^{\infty}$$

$$HospitalC = CALL(Arrive6).HC(\overline{Ready1}).HC(\overline{Ready2}).((AmbB\ in.AmbB\ out)\oplus^{1}(AmbA\ in.AmbA\ out)).\emptyset^{\infty}$$

Specification (2): 3

$$
\begin{aligned}
AmbA = ((&AmbA(HD).out\ 911.in\ HouseD.get\ PHD2.out\ HouseD.CALL(\overline{Arrive3}).in\ HospitalB.put\ PHD2.\\
&out\ HospitalB.AmbA(HC).in\ HouseC.get\ PHD1.out\ HouseC.CALL(\overline{Arrive3}).in\ HospitalB.put\ PHD1\\
&out\ HospitalB)\oplus^{1}(AmbA(SC).out\ 911.in\ School.get\ PFP1.get\ PFP2.get\ PHD3.out\ School.\\
&((CALL(\overline{Arrive4}).in\ HospitalB.put\ PHD3.out\ HospitalB.CALL(\overline{Arrive6}).in\ HospitalC.put\ PFP1.put\ PFP2.\\
&out\ HospitalC) + (CALL(\overline{Arrive6}).in\ HospitalC.put\ PFP1.put\ PFP2.out\ HospitalC.CALL(\overline{Arrive4}).\\
&in\ HospitalB.put\ PHD3.out\ HospitalB))).in\ 911.\emptyset^{\infty}\\
AmbB = ((&AmbB(SC).out\ 911.in\ School.get\ PFP1.get\ PFP2.get\ PHD3.out\ School.((CALL(\overline{Arrive4}).in\ HospitalB.\\
&put\ PHD3.out\ HospitalB.CALL(\overline{Arrive6}).in\ HospitalC.put\ PFP1.put\ PFP2.out\ HospitalC) + (CALL(\overline{Arrive6}).\\
&in\ HospitalC.put\ PFP1.put\ PFP2.out\ HospitalC.CALL(\overline{Arrive4}).in\ HospitalB.put\ PHD3.out\ HospitalB)))\\
&\oplus^{1}(AmbB(HD).out\ 911.in\ HouseD.get\ PHD2.out\ HouseD.CALL(\overline{Arrive3}).in\ HospitalB.put\ PHD2.\\
&out\ HospitalB.AmbB(HC).in\ HouseC.get\ PHD1.out\ HouseC.CALL(\overline{Arrive3}).in\ HospitalB.put\ PHD1\\
&out\ HospitalB)).in\ 911.\emptyset^{\infty}\\
AmbC = &AmbC(HA).out\ HospitlA.in\ HouseA.get\ PHBP1.out\ HouseA.CALL(\overline{Arrive1}).in\ HospitalA.put\ PHBP1.\\
&AmbC(SC).out\ HospitalA.in\ School.get\ PHBP3.out\ School\ .AmbC(HB).in\ HouseB.get\ PHBP2.out\ HouseB.\\
&CALL(\overline{Arrive2}).in\ HospitalA.put\ PHBP2.put\ PHBP3.\emptyset^{\infty}
\end{aligned}
$$

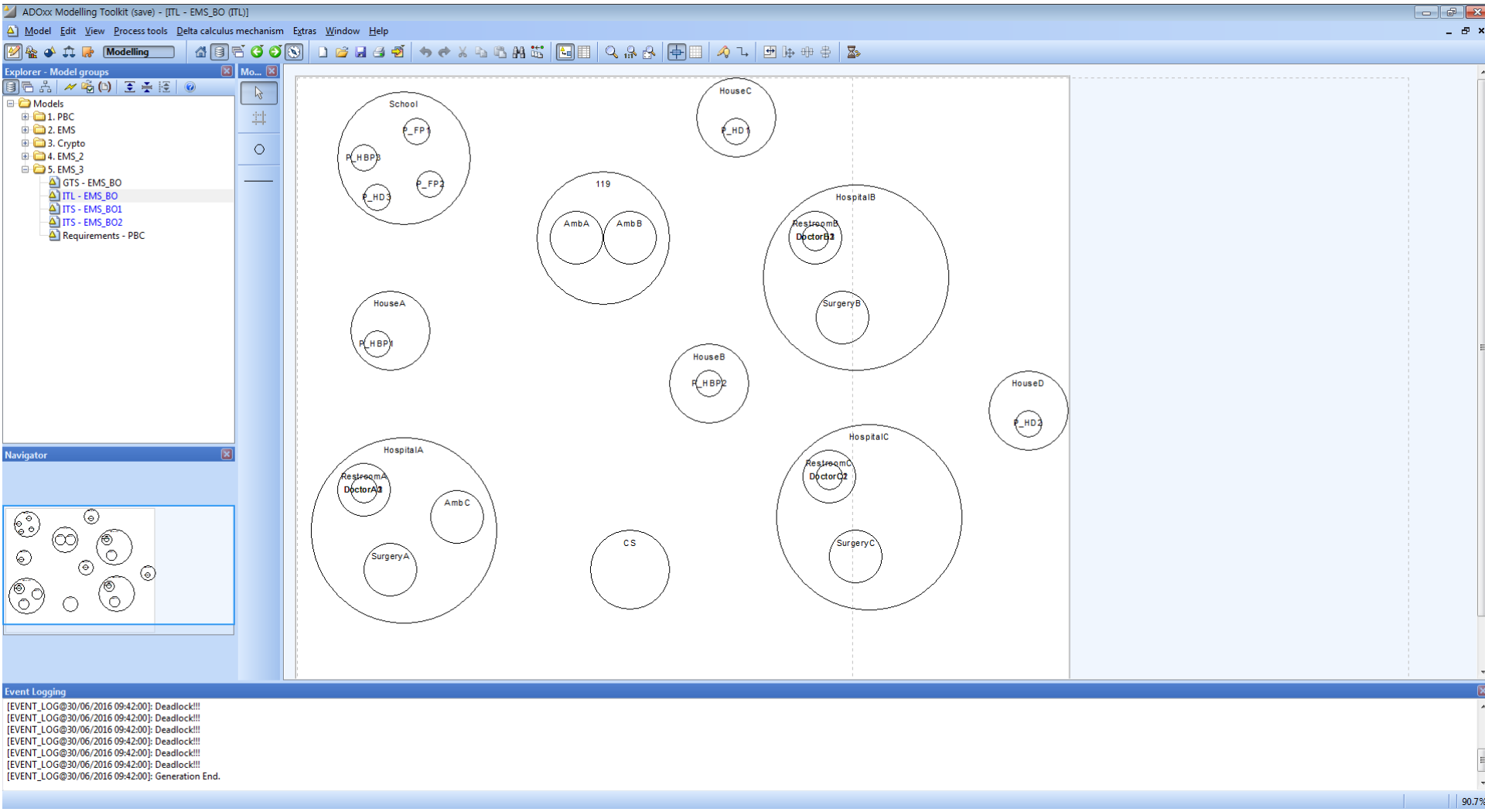
Specification (3): 16

$$HouseA = AmbC\ in.AmbC\ out.\emptyset^{\infty}$$
$$HouseB = AmbC\ in.AmbC\ out.\emptyset^{\infty}$$
$$HouseC = \big((AmbA\ in.AmbA\ out)\oplus^{1}(AmbB\ in.AmbB\ out)\big).\emptyset^{\infty}$$
$$HouseD = \big((AmbA\ in.AmbA\ out)\oplus^{1}(AmbB\ in.AmbB\ out)\big).\emptyset^{\infty}$$
$$School = \big((AmbB\ in.AmbB\ out)\oplus^{1}(AmbA\ in.AmbA\ out)\big).AmbC\ in.AmbC\ out.\emptyset^{\infty}$$
$$PHBP1 = CALL(\overline{HAHBP}).AmbC\ get.AmbC\ put.SurgeryA\ get.DoctorA1(Surgery).\emptyset^{\infty}$$
$$PHBP2 = CALL(\overline{HBHBP}).AmbC\ get.AmbC\ put.SurgeryA\ get.DoctorA2(Surgery).\emptyset^{\infty}$$
$$PHBP3 = CALL(\overline{SCHBP}).AmbC\ get.AmbC\ put.SurgeryA\ get.DoctorA3(Surgery).\emptyset^{\infty}$$
$$PHD1 = CALL(\overline{HCHD}).\big((AmbA\ get.AmbA\ put)\oplus^{1}(AmbB\ get.AmbB\ put)\big).SurgeryB\ get.DoctorB1(Surgery).\emptyset^{\infty}$$
$$PHD2 = CALL(\overline{HDHD}).\big((AmbA\ get.AmbA\ put)\oplus^{1}(AmbB\ get.AmbB\ put)\big).SurgeryB\ get.DoctorB2(Surgery).\emptyset^{\infty}$$
$$PHD3 = CALL(\overline{SCHD}).\big((AmbB\ get.AmbB\ put)\oplus^{1}(AmbA\ get.AmbA\ put)\big).SurgeryB\ get.DoctorB3(Surgery).\emptyset^{\infty}$$
$$PFP1 = CALL(\overline{SCHD}).\big((AmbB\ get.AmbB\ put)\oplus^{1}(AmbA\ get.AmbA\ put)\big).SurgeryC\ get.DoctorC1(Surgery).\emptyset^{\infty}$$
$$PFP2 = CALL(\overline{SCHD}).\big((AmbB\ get.AmbB\ put)\oplus^{1}(AmbA\ get.AmbA\ put)\big).SurgeryC\ get.DoctorC2(Surgery).\emptyset^{\infty}$$
$$SurgeryA = DoctorA1\ in.PHBP1\ get.DoctorA2\ in.PHBP2\ get.DoctorA3\ in.PHBP3\ get.\emptyset^{\infty}$$
$$SurgeryB = DoctorB1\ in.PHD1\ get.DoctorB2\ in.PHD2\ get.DoctorB3\ in.PHD3\ get.\emptyset^{\infty}$$
$$SurgeryC = DoctorC1\ in.PFP1\ get.DoctorC2\ in.PFP2\ get.\emptyset^{\infty}$$

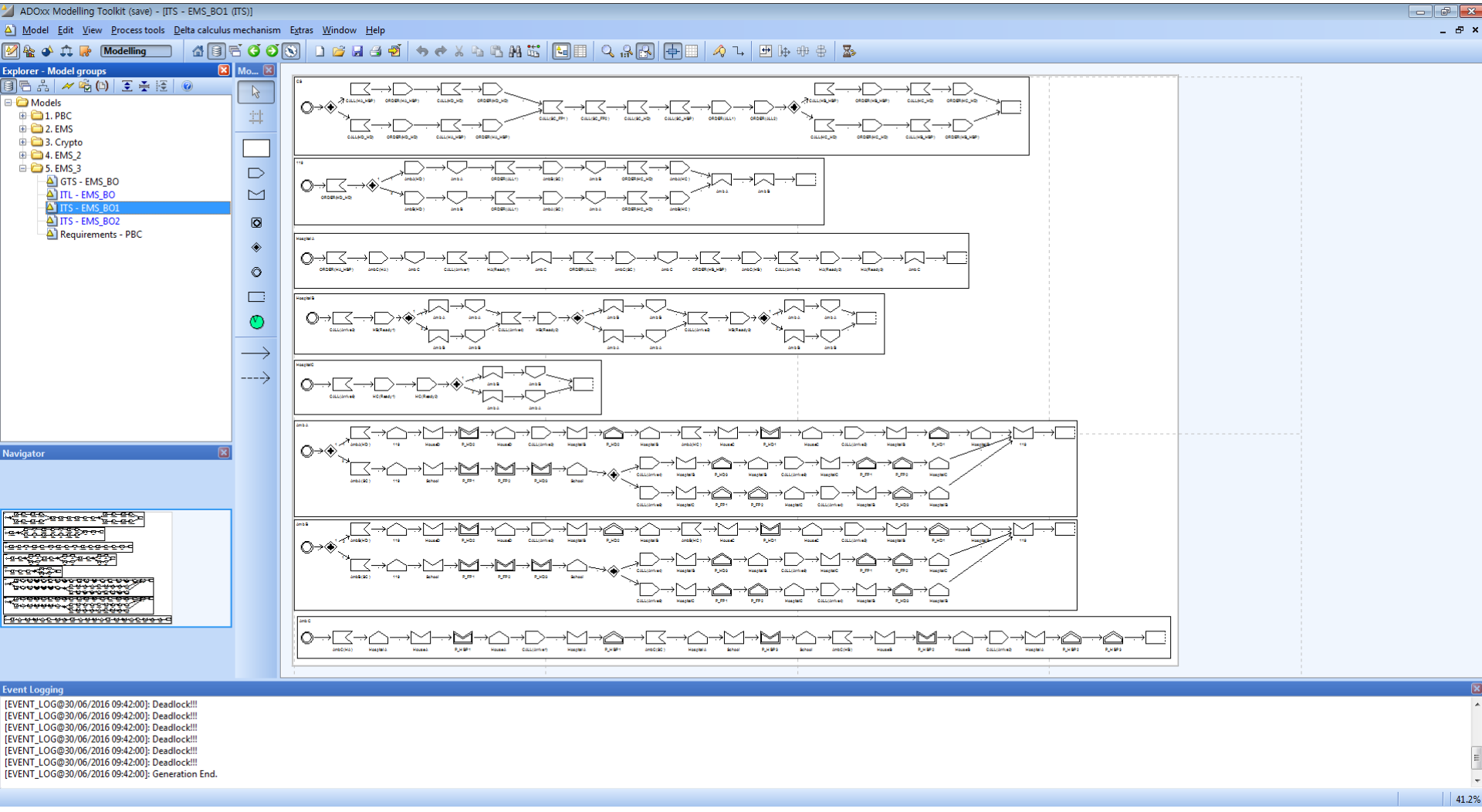
Specification (4): 11

$DoctorA1 = HA(Ready1).out\ RestroomA.in\ SurgeryA.DoctorA1(\overline{Surgery}).\emptyset^{\infty}$
$DoctorA2 = HA(Ready2).out\ RestroomA.in\ SurgeryA.DoctorA2(\overline{Surgery}).\emptyset^{\infty}$
$DoctorA3 = HA(Ready3).out\ RestroomA.in\ SurgeryA.DoctorA3(\overline{Surgery}).\emptyset^{\infty}$
$DoctorB1 = HB(Ready1).out\ RestroomB.in\ SurgeryB.DoctorB1(\overline{Surgery}).\emptyset^{\infty}$
$DoctorB2 = HB(Ready2).out\ RestroomB.in\ SurgeryB.DoctorB2(\overline{Surgery}).\emptyset^{\infty}$
$DoctorB3 = HB(Ready3).out\ RestroomB.in\ SurgeryB.DoctorB3(\overline{Surgery}).\emptyset^{\infty}$
$DoctorC1 = HC(Ready1).out\ RestroomC.in\ SurgeryB.DoctorC1(\overline{Surgery}).\emptyset^{\infty}$
$DoctorC2 = HC(Ready2).out\ RestroomC.in\ SurgeryB.DoctorC2(\overline{Surgery}).\emptyset^{\infty}$
$RestroomA = DoctorA1\ out.DoctorA2\ out.DoctorA3\ out.\emptyset^{\infty}$
$RestroomB = DoctorB1\ out.DoctorB2\ out.DoctorB3\ out.\emptyset^{\infty}$
$RestroomC = DoctorC1\ out.DoctorC2\ out.\emptyset^{\infty}$

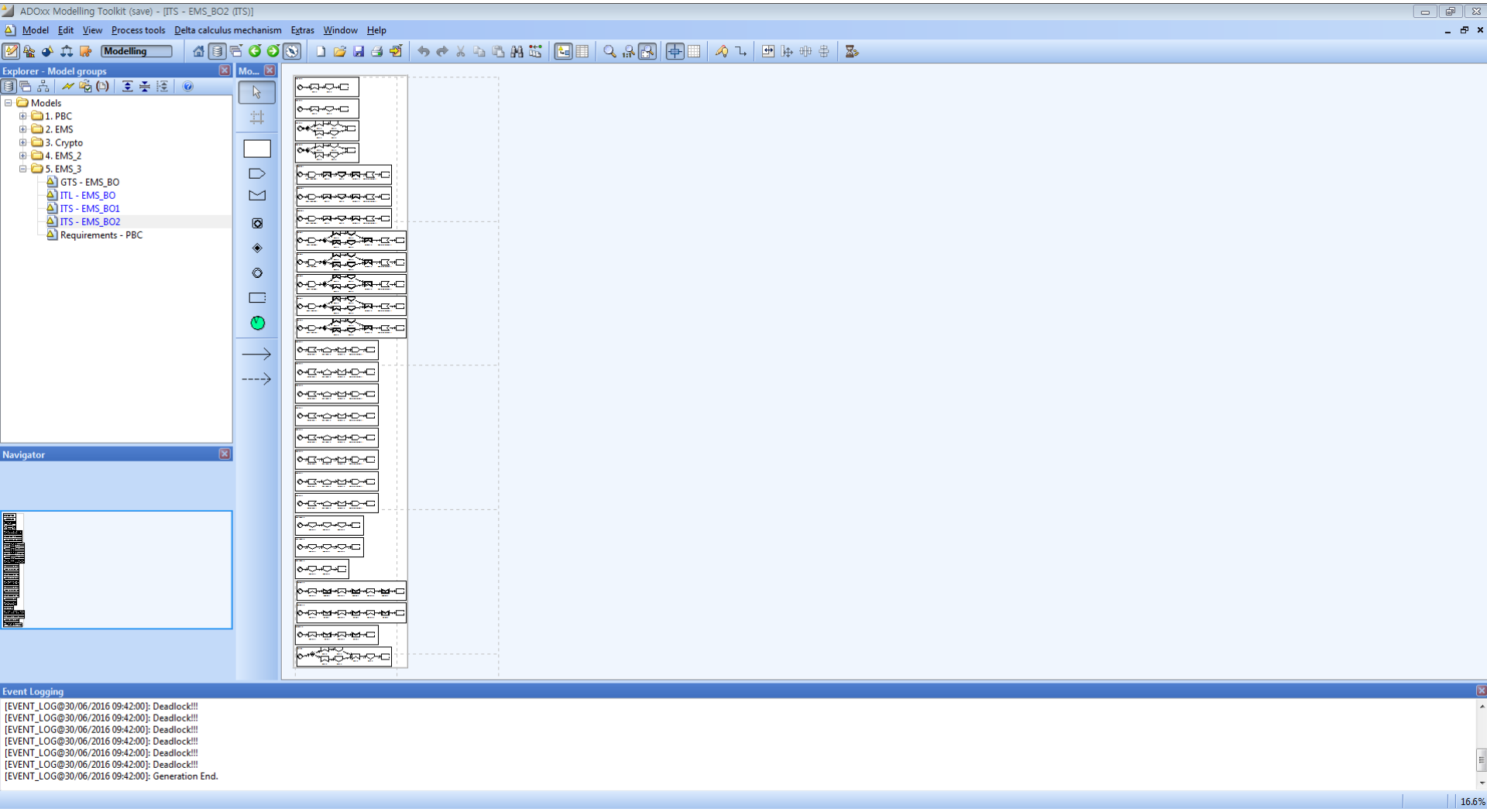
ITL


ADOxx Modelling Toolkit (save) - [ITL - EMS_BO (ITL)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
1. PBC
2. EMS
3. Crypto
4. EMS_2
5. EMS_3
GTS - EMS_BO
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Requirements - PBC
Navigator
School
P_FP1
P_HBP3
P_FP3
P_HD3
HouseC
P_HD1
119
AmbA
AmbB
HospitalB
RestroomB
DoctorB1
SurgeryB
HouseA
P_HBP1
HouseB
P_HBP2
HouseD
P_HD2
HospitalA
RestroomA
DoctorA1
AmbC
SurgeryA
CS
HospitalC
RestroomC
DoctorC1
SurgeryC
Event Logging
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Generation End.
90.7%

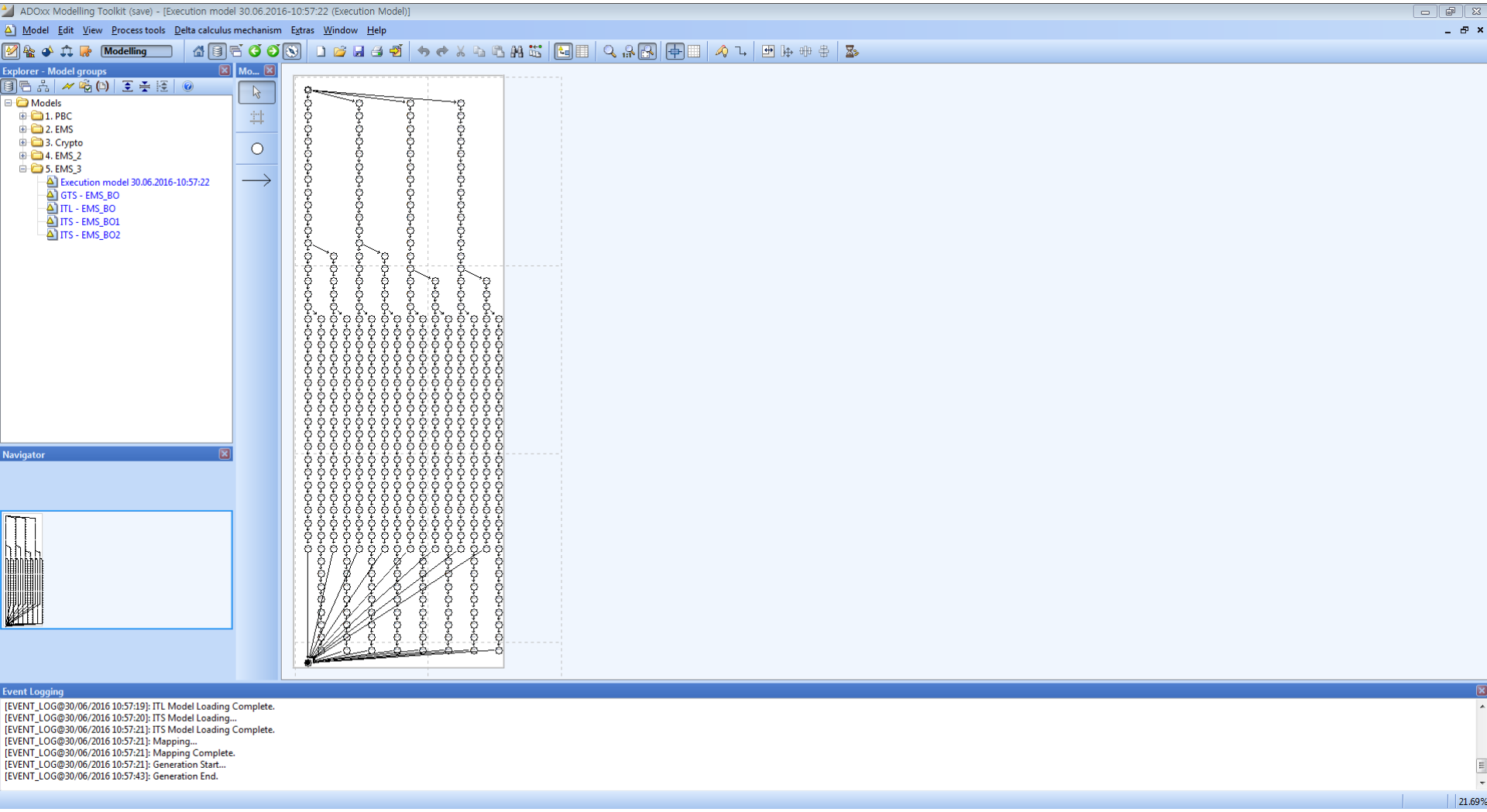
ITS (1)


ADOxx Modelling Toolkit (save) - [ITS - EMS_BO1 (ITS)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
1. PBC
2. EMS
3. Crypto
4. EMS_2
5. EMS_3
GTS - EMS_BO
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Requirements - PBC
Navigator
Event Logging
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Generation End.
41.2%

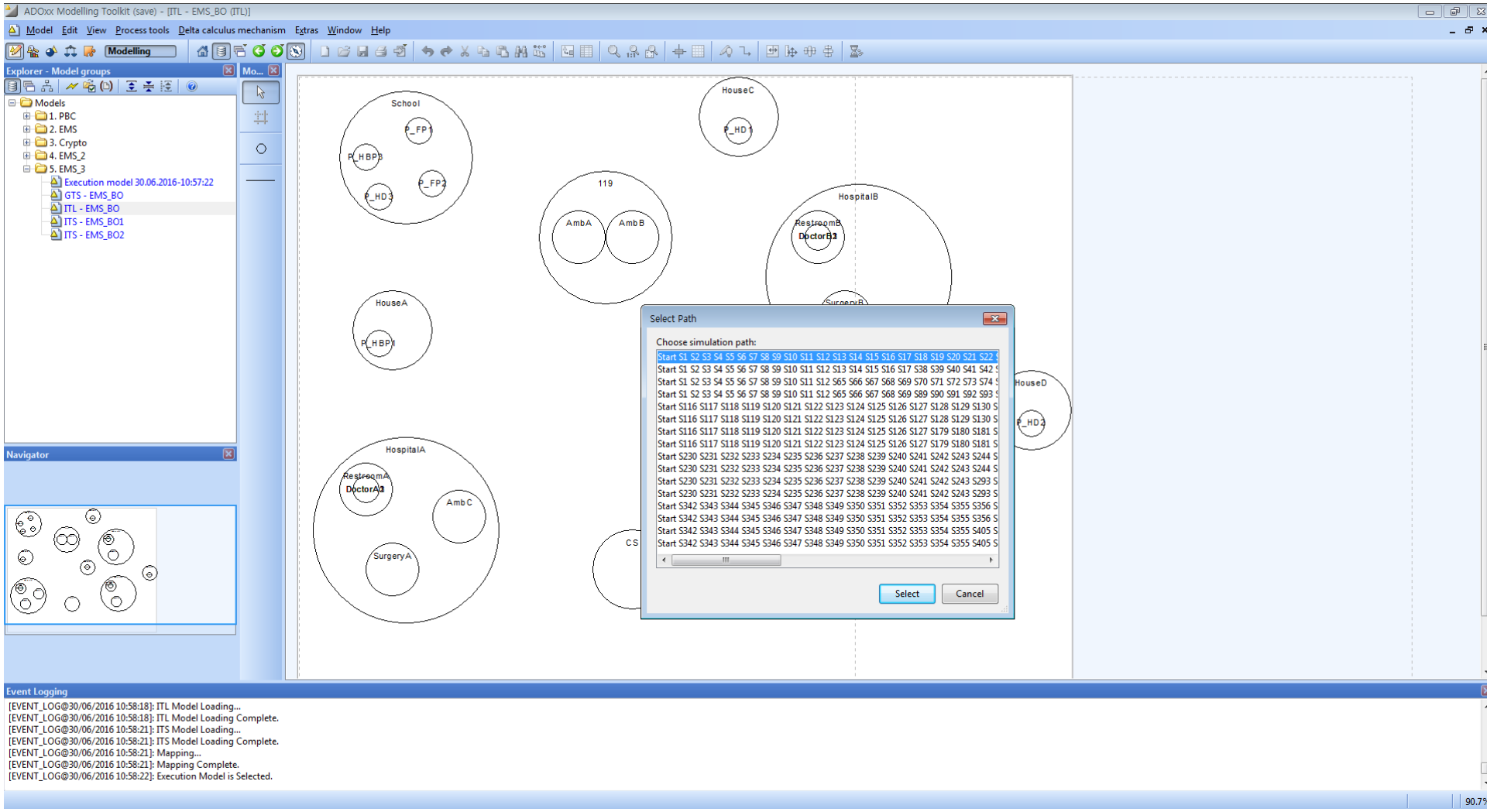
ITS (2)


ADOxx Modelling Toolkit (save) - [ITS - EMS_BO2 (ITS)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
1. PBC
2. EMS
3. Crypto
4. EMS_2
5. EMS_3
GTS - EMS_BO
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Requirements - PBC
Navigator
Event Logging
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Deadlock!!!
[EVENT_LOG@30/06/2016 09:42:00]: Generation End.
16.6%

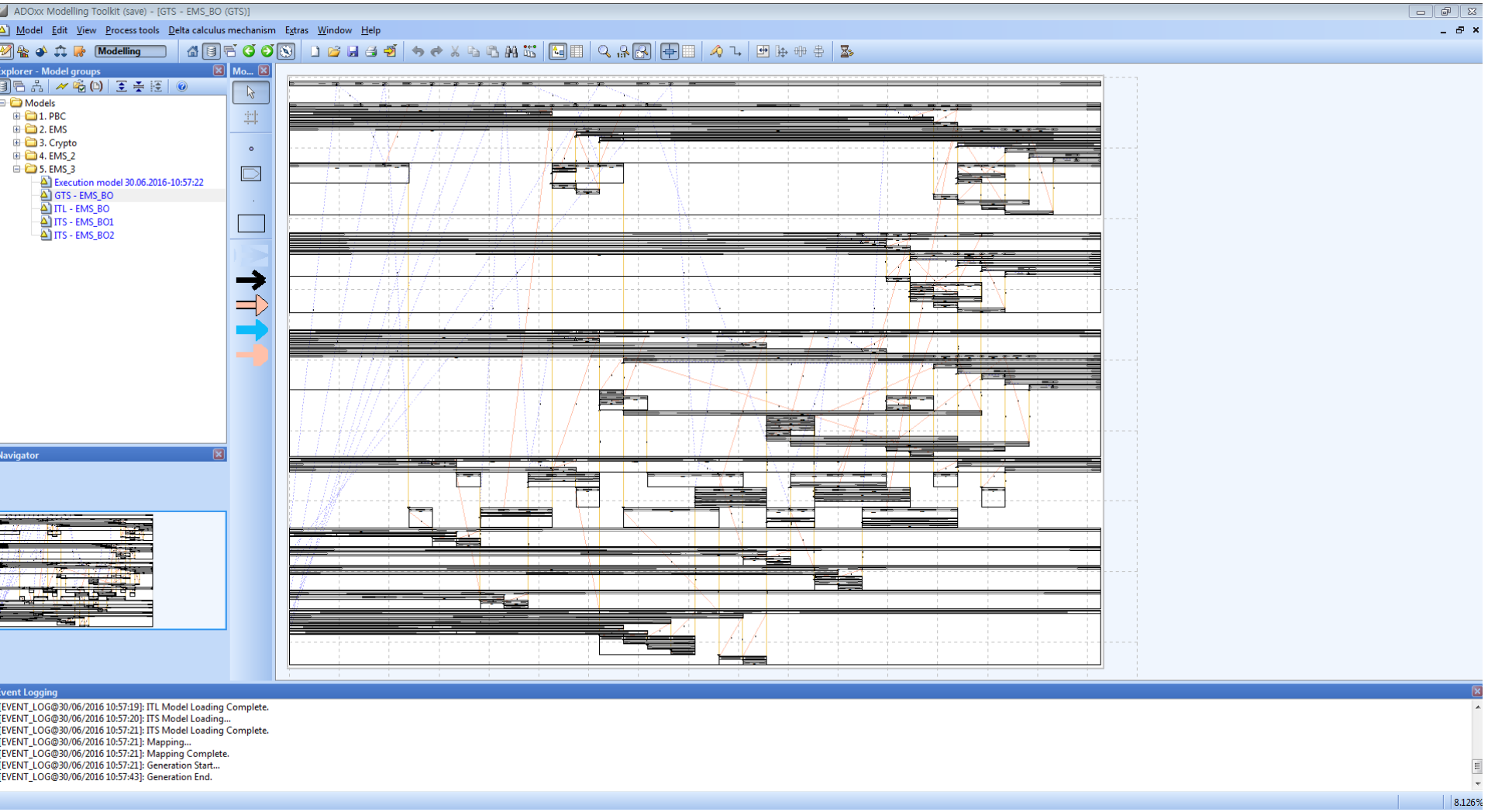
Execution Model



Simulation


ADOxx Modelling Toolkit (save) - [ITL - EMS_BO (ITL)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
1. PBC
2. EMS
3. Crypto
4. EMS_2
5. EMS_3
Execution model 30.06.2016-10:57:22
GTS - EMS_BO
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Navigator
School
P_FP1
P_HBP3
P_FP2
P_HD3
HouseC
P_HD1
119
AmbA
AmbB
HospitalB
RestroomB
DoctorB1
SurgeryB
HouseA
P_HBP1
HouseD
P_HD2
HospitalA
RestroomA
DoctorA1
AmbC
SurgeryA
CS
Select Path
Choose simulation path:
Start S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13 S14 S15 S16 S17 S18 S19 S20 S21 S22
Start S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13 S14 S15 S16 S17 S38 S39 S40 S41 S42
Start S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S65 S66 S67 S68 S69 S70 S71 S72 S73 S74
Start S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S65 S66 S67 S68 S69 S89 S90 S91 S92 S93
Start S116 S117 S118 S119 S120 S121 S122 S123 S124 S125 S126 S127 S128 S129 S130
Start S116 S117 S118 S119 S120 S121 S122 S123 S124 S125 S126 S127 S128 S129 S130
Start S116 S117 S118 S119 S120 S121 S122 S123 S124 S125 S126 S127 S179 S180 S181
Start S116 S117 S118 S119 S120 S121 S122 S123 S124 S125 S126 S127 S179 S180 S181
Start S230 S231 S232 S233 S234 S235 S236 S237 S238 S239 S240 S241 S242 S243 S244
Start S230 S231 S232 S233 S234 S235 S236 S237 S238 S239 S240 S241 S242 S243 S244
Start S230 S231 S232 S233 S234 S235 S236 S237 S238 S239 S240 S241 S242 S243 S293
Start S230 S231 S232 S233 S234 S235 S236 S237 S238 S239 S240 S241 S242 S243 S293
Start S342 S343 S344 S345 S346 S347 S348 S349 S350 S351 S352 S353 S354 S355 S356
Start S342 S343 S344 S345 S346 S347 S348 S349 S350 S351 S352 S353 S354 S355 S356
Start S342 S343 S344 S345 S346 S347 S348 S349 S350 S351 S352 S353 S354 S355 S405
Start S342 S343 S344 S345 S346 S347 S348 S349 S350 S351 S352 S353 S354 S355 S405
Select
Cancel
Event Logging
[EVENT_LOG@30/06/2016 10:58:18]: ITL Model Loading...
[EVENT_LOG@30/06/2016 10:58:18]: ITL Model Loading Complete.
[EVENT_LOG@30/06/2016 10:58:21]: ITS Model Loading...
[EVENT_LOG@30/06/2016 10:58:21]: ITS Model Loading Complete.
[EVENT_LOG@30/06/2016 10:58:21]: Mapping...
[EVENT_LOG@30/06/2016 10:58:21]: Mapping Complete.
[EVENT_LOG@30/06/2016 10:58:22]: Execution Model is Selected.

GTS


ADOxx Modelling Toolkit (save) - [GTS - EMS_BO (GTS)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
1. PBC
2. EMS
3. Crypto
4. EMS_2
5. EMS_3
Execution model 30.06.2016-10:57:22
GTS - EMS_BO
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Navigator
Event Logging
[EVENT_LOG@30/06/2016 10:57:19]: ITL Model Loading Complete.
[EVENT_LOG@30/06/2016 10:57:20]: ITS Model Loading...
[EVENT_LOG@30/06/2016 10:57:21]: ITS Model Loading Complete.
[EVENT_LOG@30/06/2016 10:57:21]: Mapping...
[EVENT_LOG@30/06/2016 10:57:21]: Mapping Complete.
[EVENT_LOG@30/06/2016 10:57:21]: Generation Start...
[EVENT_LOG@30/06/2016 10:57:43]: Generation End.
8.126%

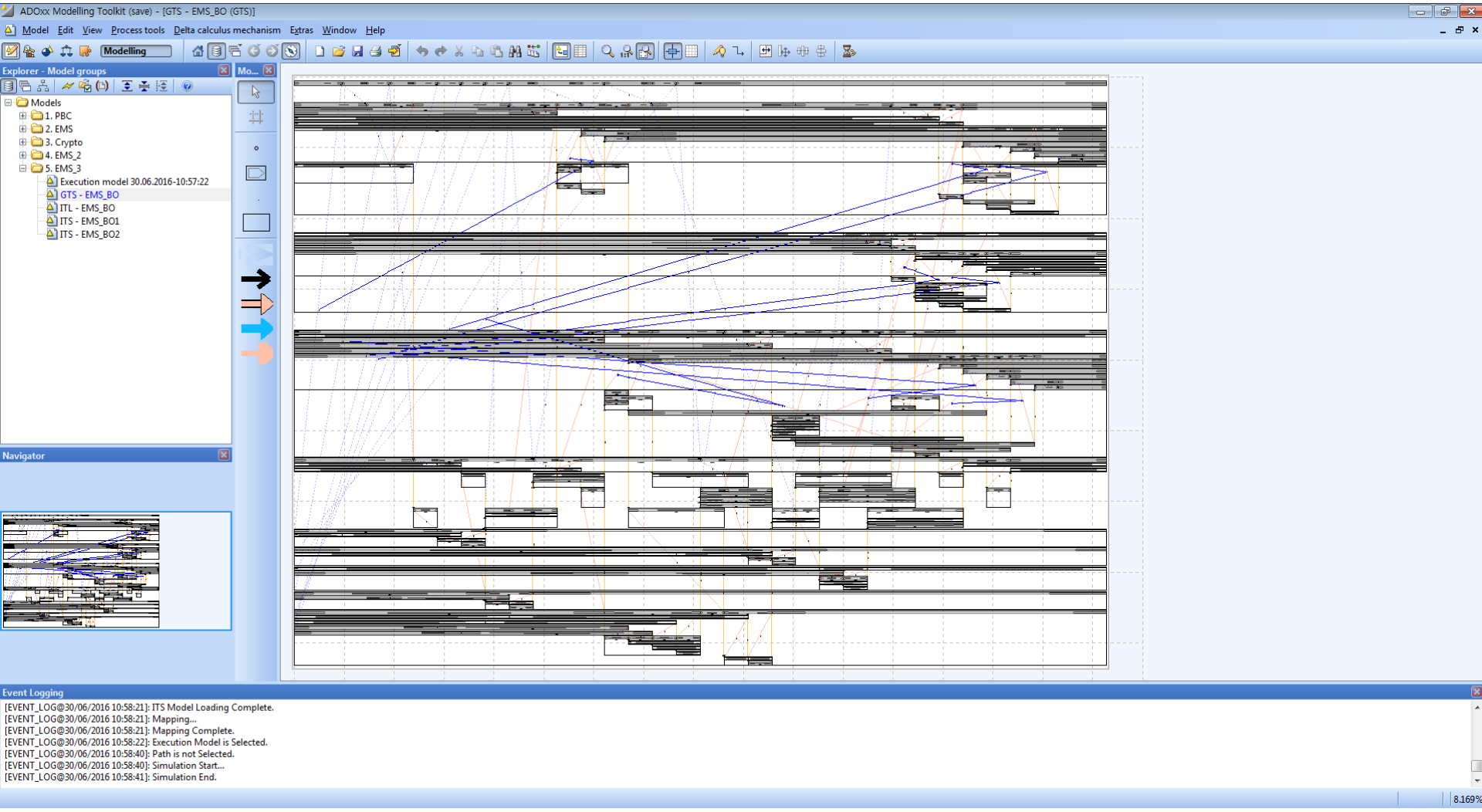
Secure Requirements (1)

- Patient HBP should be transported to Hospital A:
  - R1: $(P\_HBP1 : CALL(\overline{HA\_HBP}) \rightarrow (SurgeryA: get\ P\_HBP1)$
  - R2: $(P\_HBP2 : CALL(\overline{HB\_HBP}) \rightarrow (SurgeryA: get\ P\_HBP2)$
  - R3: $(P\_HBP3 : CALL(\overline{SC\_HBP}) \rightarrow (SurgeryA: get\ P\_HBP3)$
- Patient HD should be transported to Hospital B:
  - R4: $(P\_HD1 : CALL(\overline{HC\_HD}) \rightarrow (SurgeryB: get\ P\_HD1)$
  - R5: $(P\_HD2 : CALL(\overline{HD\_HD}) \rightarrow (SurgeryB: get\ P\_HD2)$
  - R6: $(P\_HD3 : CALL(\overline{SC\_HD}) \rightarrow (SurgeryB: get\ P\_HD3)$
- Patient FP should be transported to Hospital C:
  - R7: $(P\_FP1 : CALL(\overline{SC\_FP1}) \rightarrow (SurgeryC: get\ P\_FP1)$
  - R8: $(P\_FP2 : CALL(\overline{SC\_FP2}) \rightarrow (SurgeryC: get\ P\_FP2)$

Secure Requirements (2)

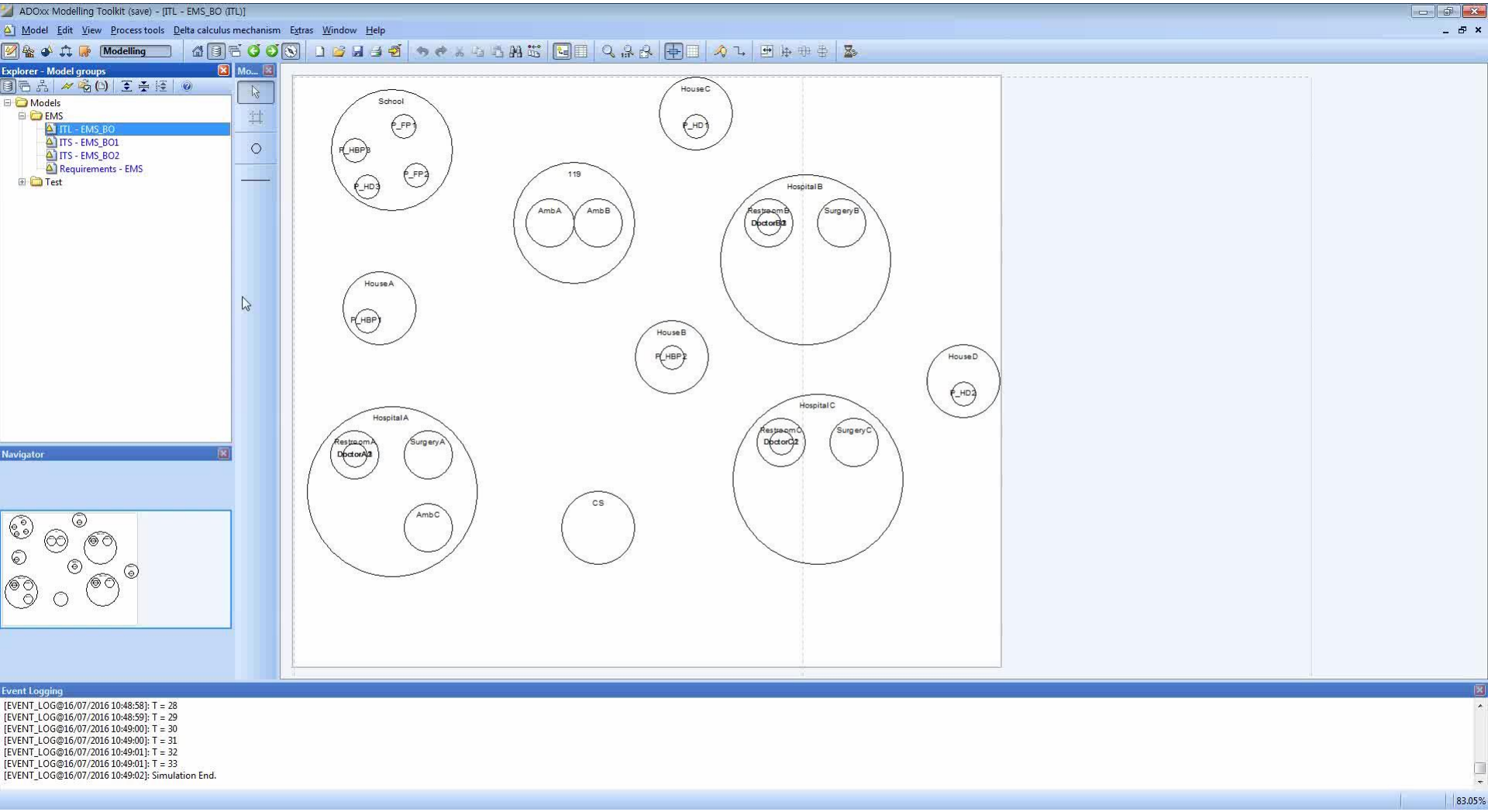
- Doctor should be ready before Patient's arrival for treatment:
  - R9: $(DoctorA1: in\ SurgeryA) < (SurgeryA: get\ P\_HBP1)$
  - R10: $(DoctorA2: in\ SurgeryA) < (SurgeryA: get\ P\_HBP2)$
  - R11: $(DoctorA3: in\ SurgeryA) < (SurgeryA: get\ P\_HBP3)$
  - R12: $(DoctorB1: in\ SurgeryB) < (SurgeryA: get\ P\_HD1)$
  - R13: $(DoctorB2: in\ SurgeryB) < (SurgeryA: get\ P\_HD2)$
  - R14: $(DoctorB3: in\ SurgeryB) < (SurgeryA: get\ P\_HD3)$
  - R15: $(DoctorC1: in\ SurgeryC) < (SurgeryA: get\ P\_FP1)$
  - R16: $(DoctorC2: in\ SurgeryC) < (SurgeryA: get\ P\_FP2)$

GTS Logic


ADOxx Modelling Toolkit (save) - [GTS - EMS_BO (GTS)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
1. PBC
2. EMS
3. Crypto
4. EMS_2
5. EMS_3
Execution model 30.06.2016-10:57:22
GTS - EMS_BO
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Navigator
Event Logging
[EVENT_LOG@30/06/2016 10:58:21]: ITS Model Loading Complete.
[EVENT_LOG@30/06/2016 10:58:21]: Mapping...
[EVENT_LOG@30/06/2016 10:58:21]: Mapping Complete.
[EVENT_LOG@30/06/2016 10:58:22]: Execution Model is Selected.
[EVENT_LOG@30/06/2016 10:58:40]: Path is not Selected.
[EVENT_LOG@30/06/2016 10:58:40]: Simulation Start...
[EVENT_LOG@30/06/2016 10:58:41]: Simulation End.

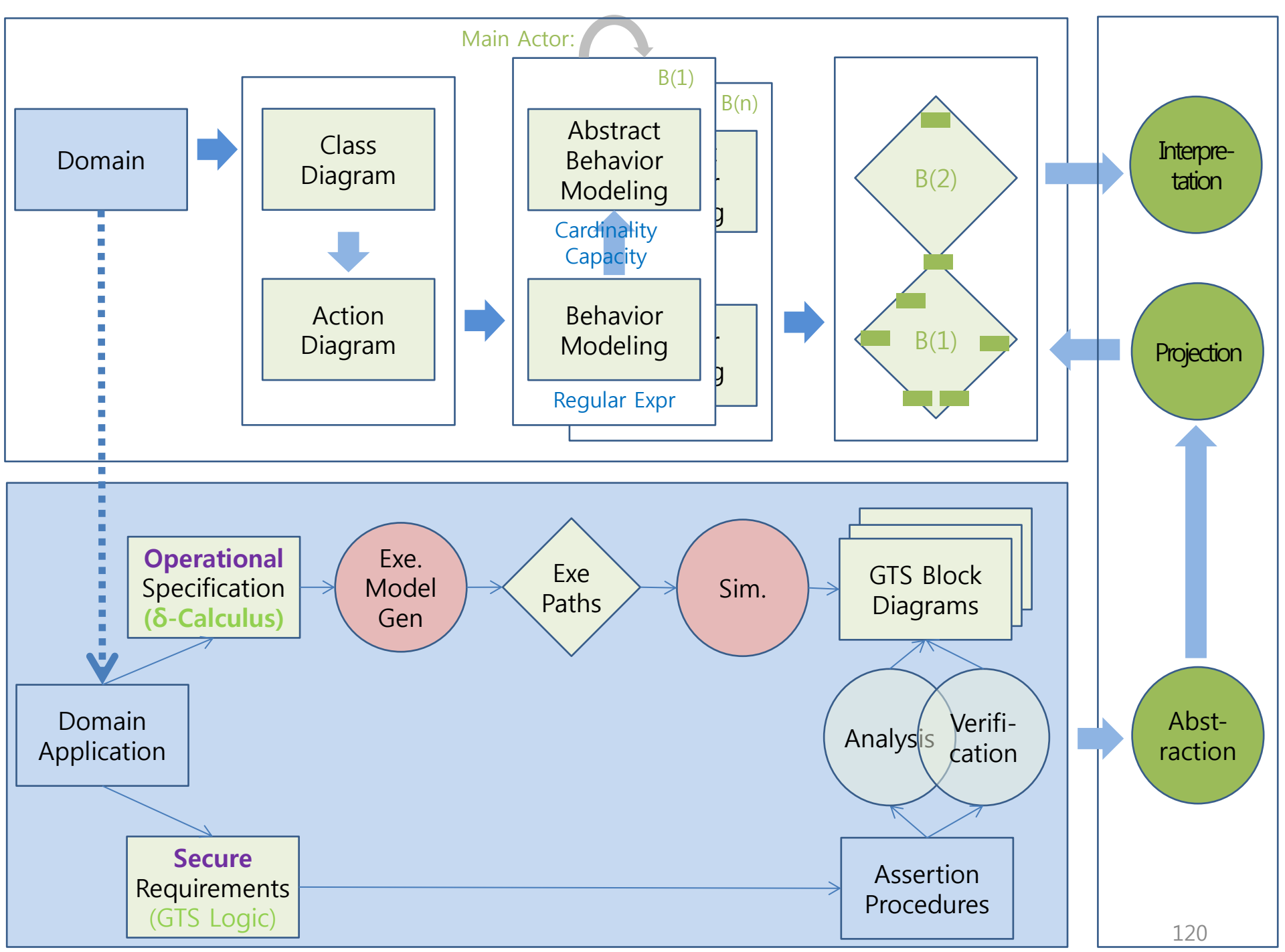
SAVE

DEMO: SAVE EMS

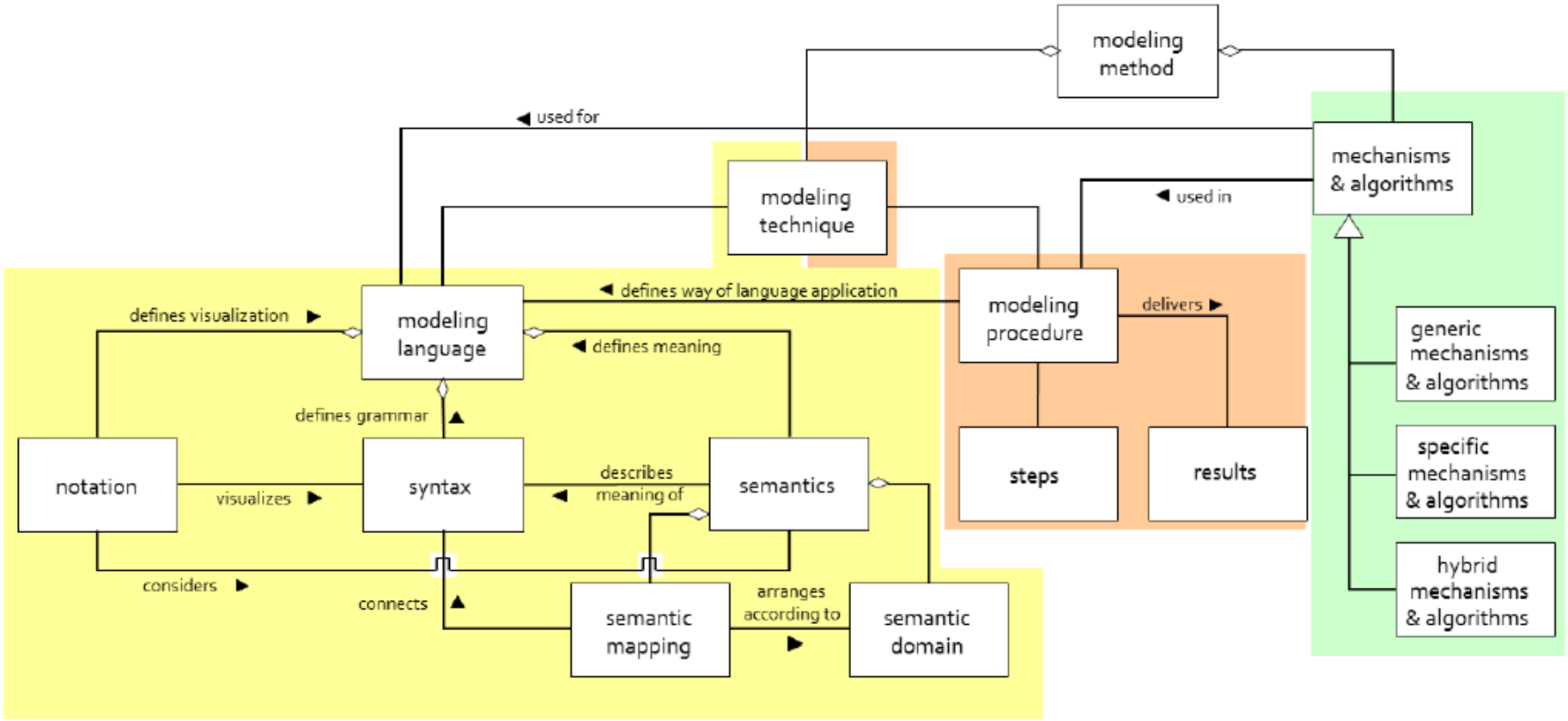


1. Motivation
2. Modeling
    1) Specification: δ-Calculus
    2) Verification: GTS Logic
    3) EMS Example
3. **SAVE Tool**
4. Cyber-Physical Systems Application
5. Summary
6. Discussion

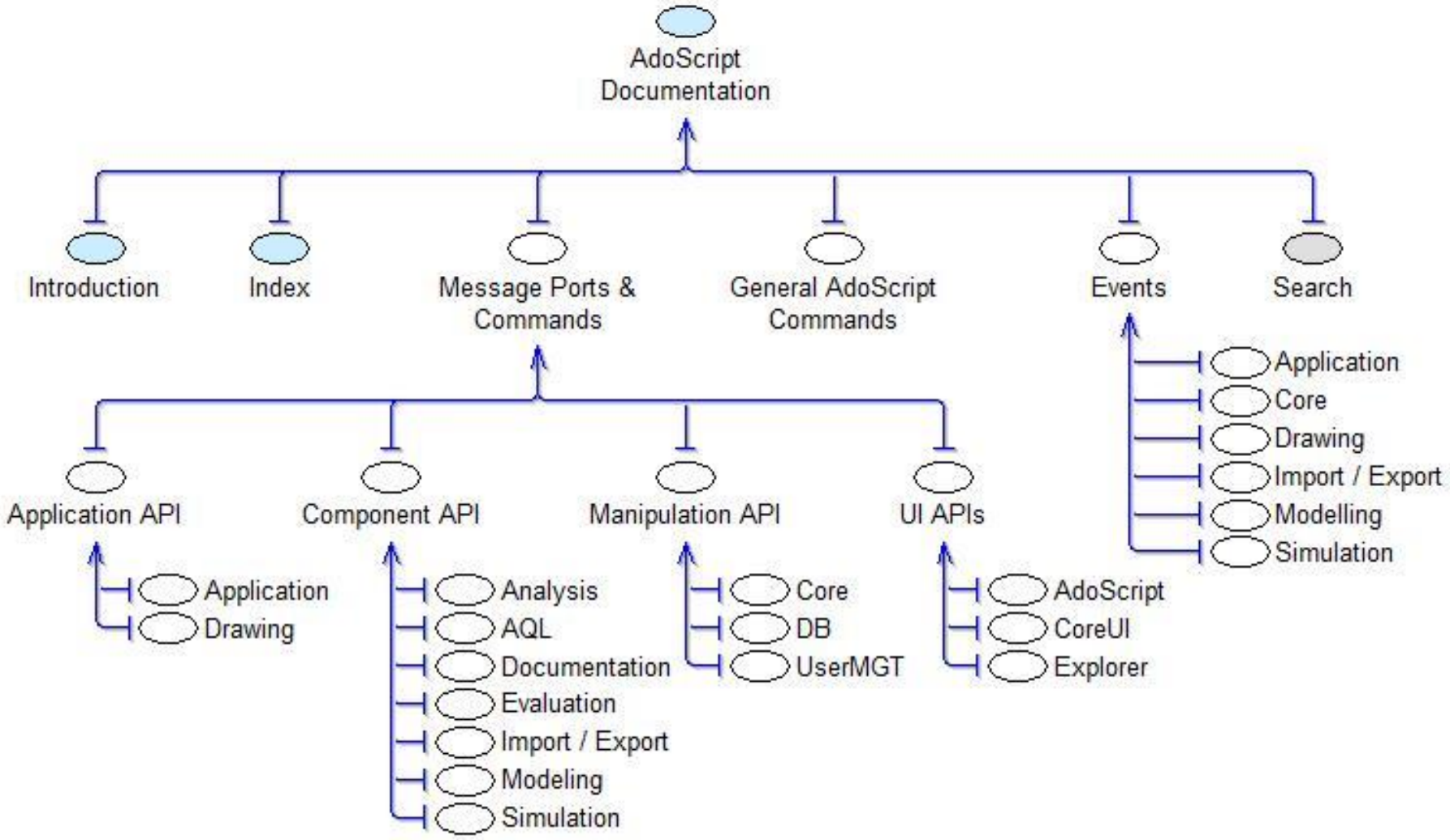
3. SAVE TOOLS


Main Actor:
Domain
Class Diagram
Action Diagram
B(1)
B(n)
Abstract Behavior Modeling
Cardinality Capacity
Behavior Modeling
Regular Expr
B(2)
B(1)
Interpre-tation
Projection
Abst-raction
Operational Specification (δ-Calculus)
Exe. Model Gen
Exe Paths
Sim.
GTS Block Diagrams
Domain Application
Analysis
Verifi-cation
Secure Requirements (GTS Logic)
Assertion Procedures

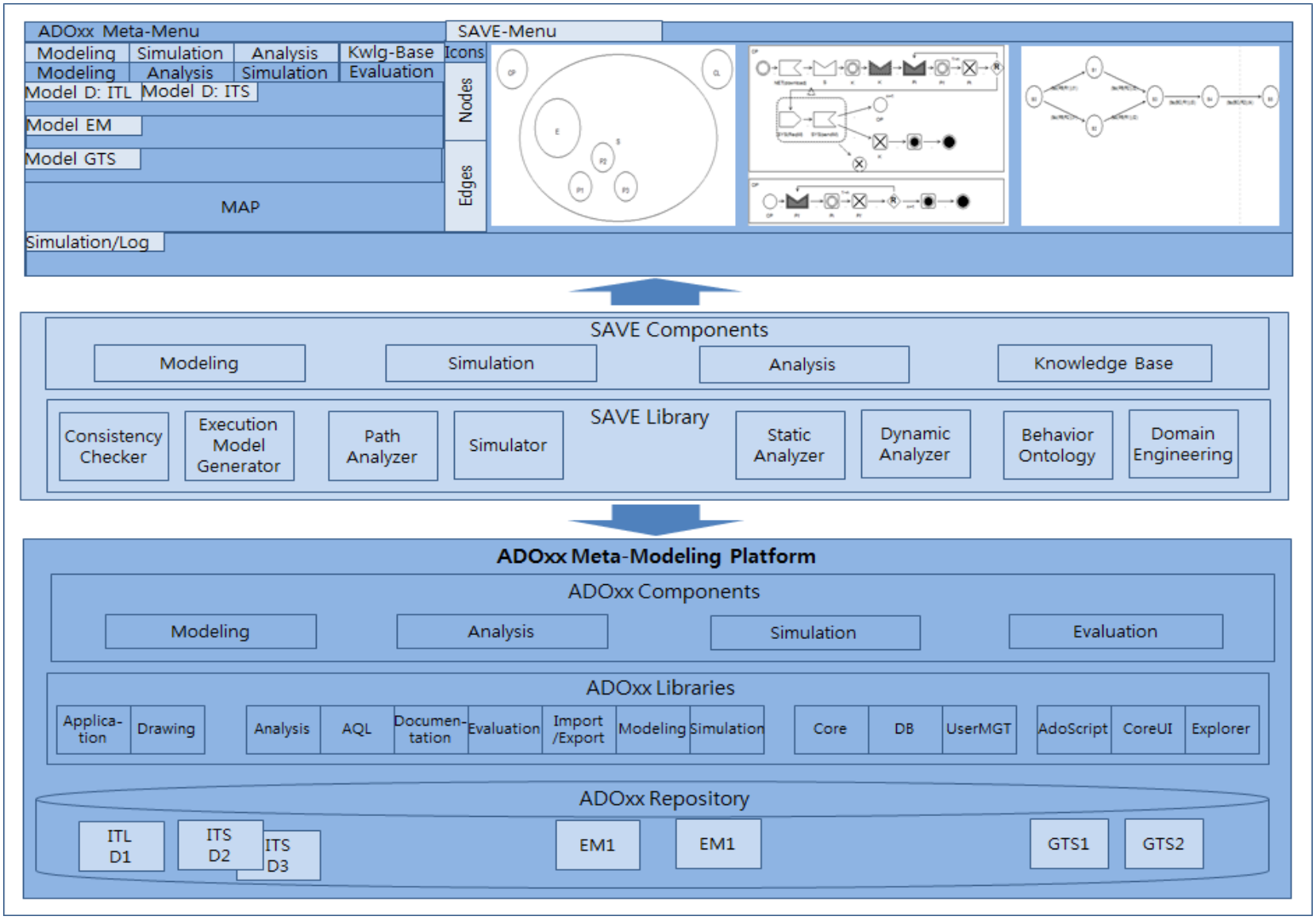
Generic Modelling Method Framework



ADOxx Library



Platform


ADOxx Meta-Menu
Modeling
Simulation
Analysis
Kwlg-Base
Modeling
Analysis
Simulation
Evaluation
Model D: ITL
Model D: ITS
Model EM
Model GTS
MAP
Simulation/Log
SAVE-Menu
Icons
Nodes
Edges
SAVE Components
Modeling
Simulation
Analysis
Knowledge Base
SAVE Library
Consistency Checker
Execution Model Generator
Path Analyzer
Simulator
Static Analyzer
Dynamic Analyzer
Behavior Ontology
Domain Engineering
ADOxx Meta-Modeling Platform
ADOxx Components
Modeling
Analysis
Simulation
Evaluation
ADOxx Libraries
Applica-tion
Drawing
Analysis
AQL
Documen-tation
Evaluation
Import /Export
Modeling
Simulation
Core
DB
UserMGT
AdoScript
CoreUI
Explorer
ADOxx Repository
ITL D1
ITS D2
ITS D3
EM1
EM1
GTS1
GTS2

Execution Model Generator

```
CC "CoreUI" MODEL_SELECT_BOX modeltype:(exModel) title:("Select Execution Model") boxtext:("Please select execution model:")
IF (endbutton = "ok")
{
    SET sExModelIDs:(modelids)
    SET exModelID:(VAL sExModelIDs)
    CC "Core" LOAD_MODEL modelid:(exModelID)
    EVENT_LOG msgType:"EVENT_LOG" message: ("Execution Model is Selected.")

    CC "Core" GET_CLASS_ID classname:"State_EX"
    SET nStateClassID:(classid)
    CC "Core" GET_OBJ_ID modelid:(exModelID) classid:(nStateClassID) objn
    SET startState:(objid)
    SET totalPath:(0)
    SETL pathString:("")
    SET totalPathString:("")

    FIND_PATH start:(STR startState) path:(pathString)
    CC "AdoScript" LISTBOX entries:(totalPathString) toksep:";" title:"Se

    IF (endbutton = "ok")
    {
        SET select:(selection)
        #CC "AdoScript" VIEWBOX text:(select)
        EVENT_LOG msgType:"EVENT_LOG" message: ("Path is Selected.")
    }
    ELSE
    {
        EVENT_LOG msgType:"EVENT_LOG" message: ("Path is not Selected.")
        SET exModelID:(0)
    }
}
ELSE
{
    EVENT_LOG msgType:"EVENT_LOG" message: ("Execution Model is not Selec
    SET exModelID:(0)
}

#Simulation Start
EVENT_LOG msgType:"EVENT_LOG" message: ("Simulation Start...")

#Simulating...
SETG statecount:(0)
SETG pathcount:(0)
SETG choicemap:(map())
SETG choicemap["all_list"]:("")
FOR pname in:(loadITS["list"])
{
    SETG choicemap[loadITS[pname+"_name"]+"_list"]:("")
}
SIM_INIT mode:("sim") lcurrent:lcurrent
SETL current:(lcurrent)
SETL tempsync:(map())
SETL tempsync["list"]:("")
SIM current:(current) mode:("sim") globaltime:(0) sync:(STR tempsync) syn

EVENT_LOG msgType:"EVENT_LOG" message: ("Simulation End.")
```

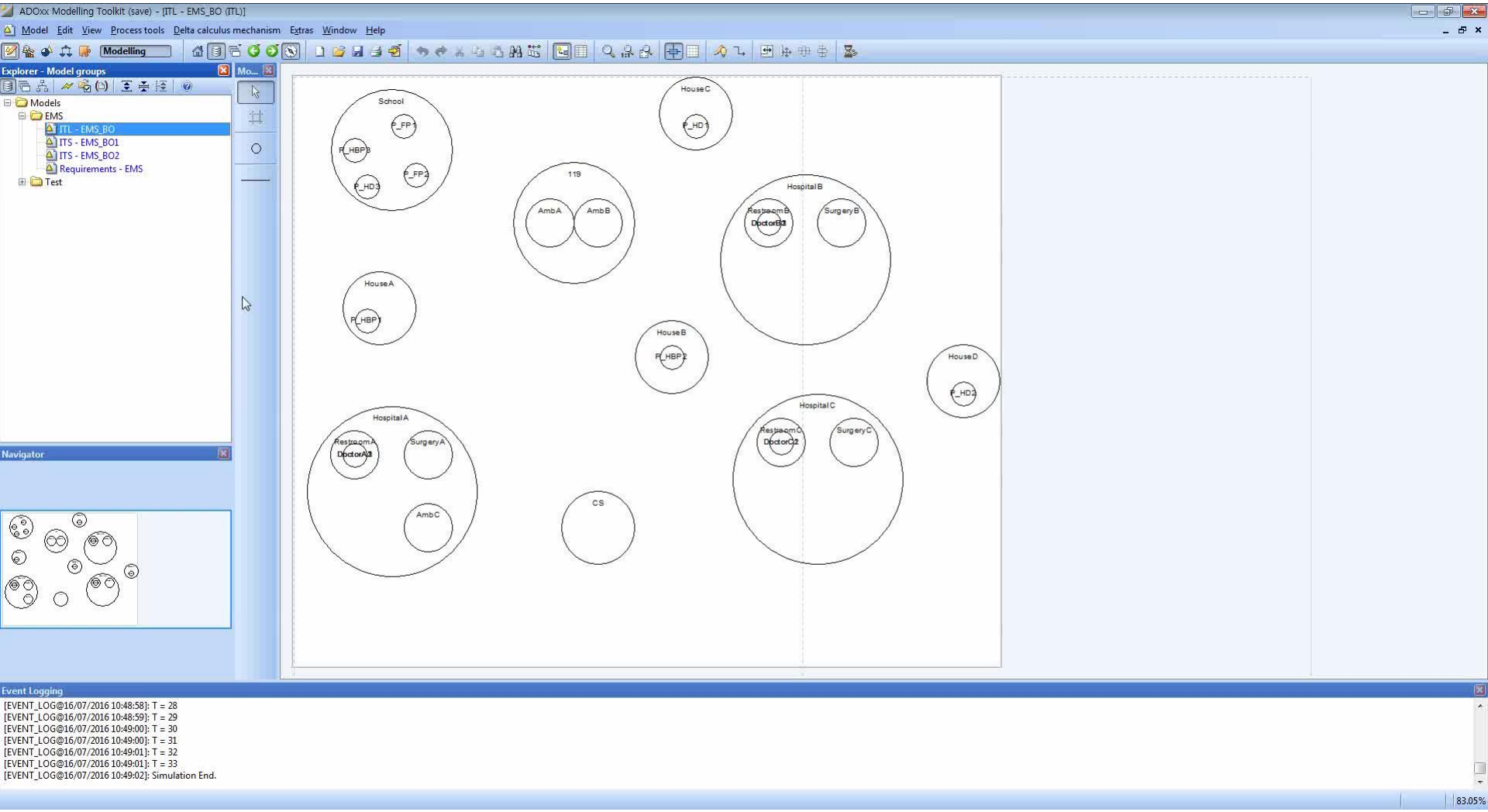
```
        }
    }
    ELSIF (acttype = "In")
    {
        SETL state:(loadITS[actid+"_state"])
        SETL ctime:(loadITS[actid+"_time"])
        SETL gtime:(loadITS[actid+"_gtime"])
        SETL dtime:(loadITS[actid+"_deadline"])
        #SETL ctime:(ctime + 1)
        #SETL gtime:(gtime + 1)

        #state and time check
        IF (state = "Waiting")
        {
            SETL type:(loadITS[actid+"_attr1"])
            SETL process:(loadITS[actid+"_attr2"])
            SETL myprocess:(loadITS[actid+"_process"])

            IF (type = "Request")
            {
                CHECK_MOVE current:(currentlist) move:("In") process:(process) myprocess:(myprocess) find:("Permission") moveresult:result
                SETL findid:(result)

                IF (findid != 0)
                {
                    SETL change["list"]:(tokunion(change["list"], actid))
                    SETL change[actid+"_mode"]:("Executing")
                    SETG loadITS[actid+"_time"]:(-1)
                    SETL synclist:(tokunion(synclist, actid))
                }
            }
            ELSE
            {
                CHECK_MOVE current:(currentlist) move:("In") process:(process) myprocess:(myprocess) find:("Request") moveresult:result
                SETL findid:(result)

                IF (findid != 0)
                {
                    SETL change["list"]:(tokunion(change["list"], actid))
                    SETL change[actid+"_mode"]:("Executing")
                    SETG loadITS[actid+"_time"]:(-1)
                    SETL synclist:(tokunion(synclist, actid))
                }
            }
        }
    }
    ELSIF (acttype = "Out")
    {
        SETL state:(loadITS[actid+"_state"])
        SETL ctime:(loadITS[actid+"_time"])
        SETL gtime:(loadITS[actid+"_gtime"])
        SETL dtime:(loadITS[actid+"_deadline"])
```

GTS Analyzer

```
FOR i from:1 to:(sEndOfStateNum)
{
   SETL state_num:(STR i)
   SETL t:(0)
   IF (nDebug = 1) {SET data10:(data10+state_num+": \t sEndOfStateNum \n")}
   #7.5
   FOR process in:(mD_Data[state_num+"_"+"list"])
   {
      IF (type(mD_Data[state_num+"_"+process+"_"+"interactiontype"]) != "undefined")
      {
         IF (nDebug = 1) {SET data14:(data14+state_num+": "+process+": "+ "STATE: type:\t"+mD_De
         SETL a:(mD_Data[state_num+"_"+process+"_"+"interactiontype"])
      }
      IF (type(mD_Data[state_num+"_"+process+"_"+"interactionnum"]) != "undefined")
      {
      #  IF (nDebug = 1) {SET data14:(data14+state_num+": "+process+": "+ "STATE: num:\t"+mD_D
         SETL b:(mD_Data[state_num+"_"+process+"_"+"interactionnum"])
      }
      IF (type(mD_Data[state_num+"_"+process+"_"+"interactiontype"]) != "undefined" AND type(mD_
      {
         IF (type(mD_Data[process+"_"+a+b+"_AtrOrAte"]) != "undefined")
         {
            IF (nDebug = 1) {SET data14:(data14+state_num+": "+process+": "+ "STATE: AtrOrAte:\t
         }
      }
      # SET mD_Data[state_num+"_"+acteeprocess+"_"+"interactiontype"]:(sInterActionType) #버그수정
      # SET mD_Data[state_num+"_"+acteeprocess+"_"+"interactionnum"]:(sInterActionNum) #버그수정을
      # IF (type(mD_Data[state_num+"_"+acteeprocess+"_"+"interactiontype"])!="undefined")
      # {
         # IF (nDebug = 1) {SET data14:(data14+state_num+": "+process+": "+ "STATE: AtrOrAte:\
      # }

   }
   IF (nDebug = 1) {SET data14:(data14+"EndOfState\n")}
}
IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_De


SET data15:("")
FOR i from:1 to:(mSingleActionData["num"])
{
   IF (type(mSingleActionData[STR i+"_AtrOrAte"]) != "undefined")
   {
      SETL t:(mSingleActionData[STR i+"_process"])
      SETL a:(mSingleActionData[STR i+"_interactiontype"])
      SETL b:(mSingleActionData[STR i+"_interactionnum"])
      IF (nDebug = 1) {SET data15:(data15+STR i+": "+t+": "+ "STATE: AtrOrAte:\t"+"\t type: "+a+
   }
}
IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_De



FOR process in:(mAll_Of_Process["list"])
{
   SETL prev_blockaddress:("")
   IF (type(mD_Data[process+"_InterActionList"]) = "undefined")
   {
      IF (nDebug = 1) {SET data07:(data07+"GET_ERROR: ")}
      IF (nDebug = 1) {SET data07:(data07+process+"\n")}
   }
   ELSE
   {
      IF (nDebug = 1) {SET data07:(data07+"NO_ERROR: ")}
```

```
IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_Debug_Text/dat
IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_Debug_Text/int



IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_Debug_Text/dat

########################################################################
# 7. 모든 프로세스 Obj의 Y축 크기 구하기
# 1) 모든 상태의 프로세스 포함관계를 모두 구함. 모든 상태의 포함관계(block address)를 구함
# 2) 프로세스 포함관계 결과를 통해 프로세스의 Y축 크기 계산함
# 3) 프로세스 및 액션의 X축 크기를 계산함. 동시에 Y축 크기도 계산함
# 4) 인터액션의 x1 y1 x2 y2 구하기 (태스팅 후 진행
# 5) 프로세스 포함관계 결과를 통해 프로세스 액션블록의 X축 크기 계산함
# 6) 프로세스 포함관계 결과를 통해 상글 액션블록의 X축 크기 계산함
# nlist: 모든 프로세스 정보
# slist: 수행된 프로세스 정보
########################################################################

SET data05:("")
SET data06:("")
SET data07:("")

SET sAll_Of_Children:(map())
SET sAll_Of_Children["list"]:("")
###7.1 모든 상태의 프로세스 포함관계를 모두 구함

FOR i from:1 to:(sEndOfStateNum)
{
   SETL state_num:(STR i)
   SETL t:(0)
   IF (nDebug = 1) {SET data05:(data05+state_num+": \t sEndOfStateNum \n")}
   #7.1
   FOR process in:(mD_Data[state_num+"_"+"list"])
   {
      IF (mD_Data[state_num+"_"+process+"_"+"parent"] = "")
      {
         IF (nDebug = 1) {SET data05:(data05+state_num+": "+"root_"+process+": \t GET_Children \n")}
         SET_ALL_OF_CHILDREN current_address:("root") next_address:(process) state_num:(state_num) sAll_Of_C
      }
   }
}
IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_Debug_Text/dat



###7.2 프로세스 포함관계 결과를 통해 7.1의 결과에 대한 프로세스의 Y축 크기 계산함
SET init_y:(1.0)
SET_ALL_OF_PROCESS_BLOCK_Y_POSITION current_address:("root") init_y:init_y sAll_Of_Children:sAll_Of_Childr
IF (nDebug = 1) {CC "AdoScript" FWRITE file:("C:/Users/default.DESKTOP-AH94BHT/Desktop/ADOxx_Debug_Text/dat

###7.5 프로세스 포함관계 결과를 통해 프로세스 액션블록의 X축 크기 계산함
###7.6 프로세스 포함관계 결과를 통해 상글 액션블록의 X축 크기 계산함
SET data10:("")
SET data11:("")
SET data13:("")
SET prev_blockaddress:(map()) #프로세스 블록 계산
SET prev_sblockaddress:(map()) #상글블록 계산
FOR i from:1 to:(sEndOfStateNum)
{
   SETL state_num:(STR i)
   SETL t:(0)
   IF (nDebug = 1) {SET data10:(data10+state_num+": \t sEndOfStateNum \n")}
   #7.5
   FOR process in:(mD_Data[state_num+"_"+"list"])
```

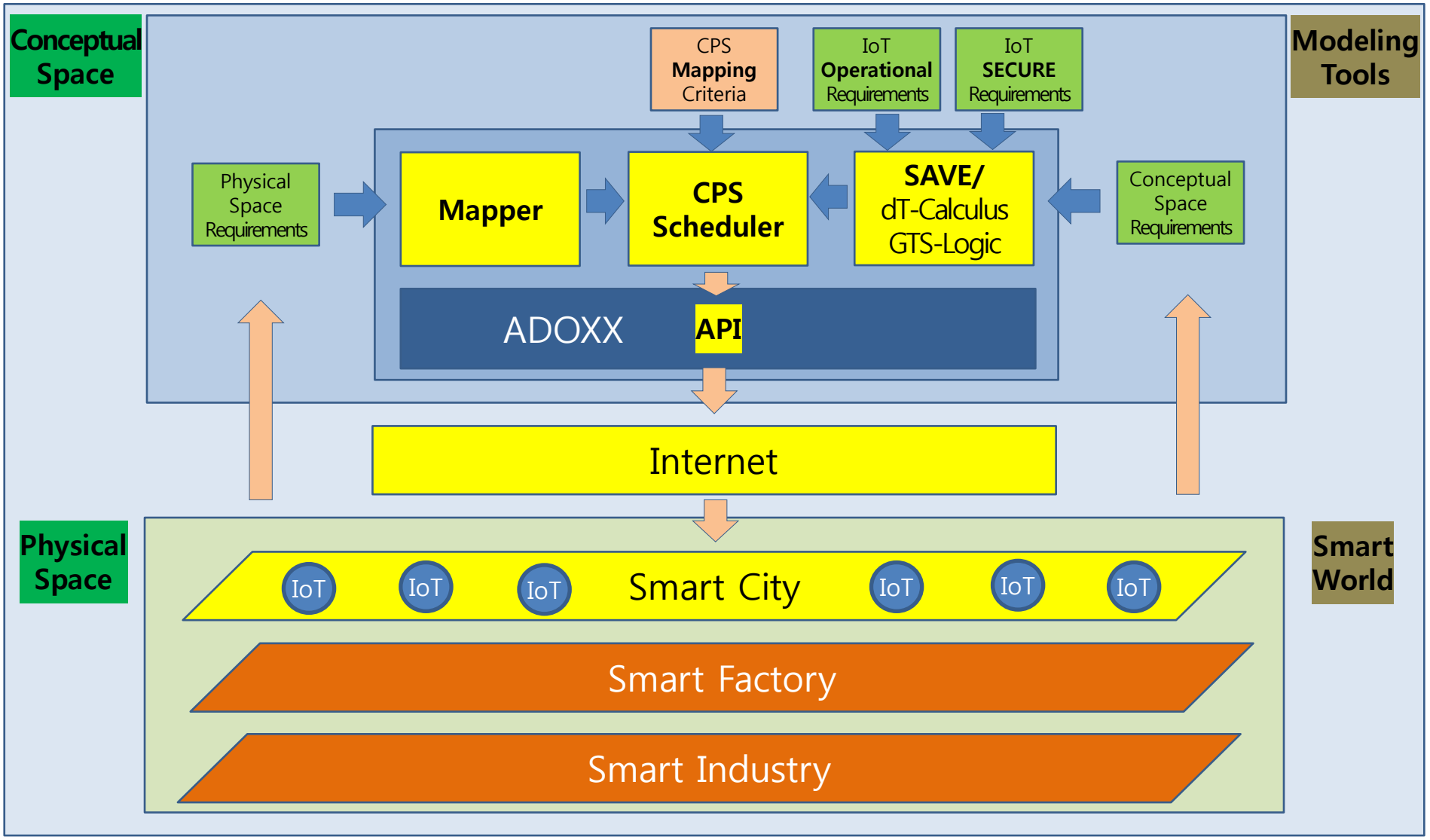
DEMO: SAVE EMS



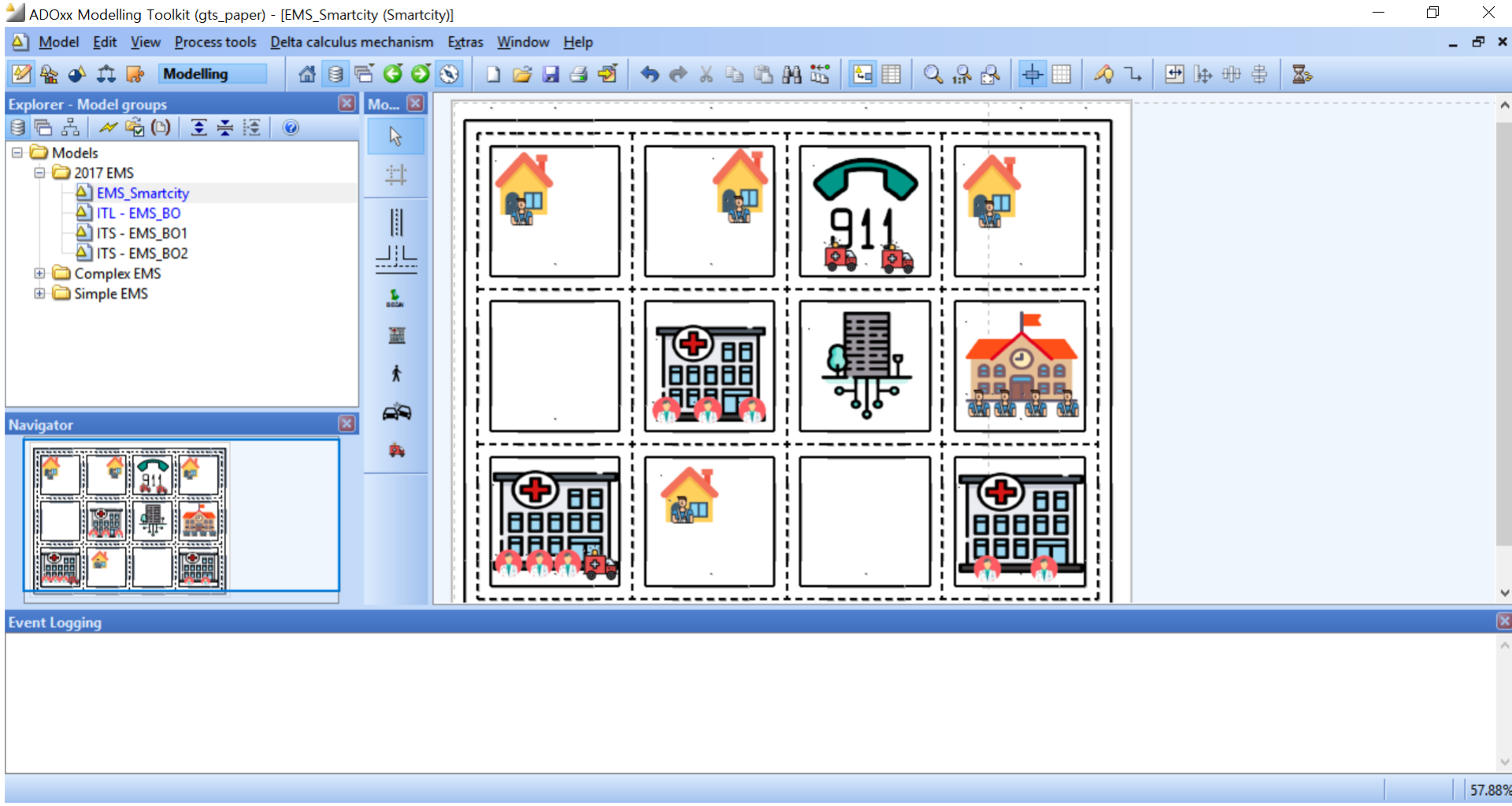
1. Motivation
2. Modeling
    1) Specification: δ-Calculus
    2) Verification: GTS Logic
    3) EMS Example
3. SAVE Tool
4. **Cyber-Physical Systems Application**
5. Summary
6. Discussion

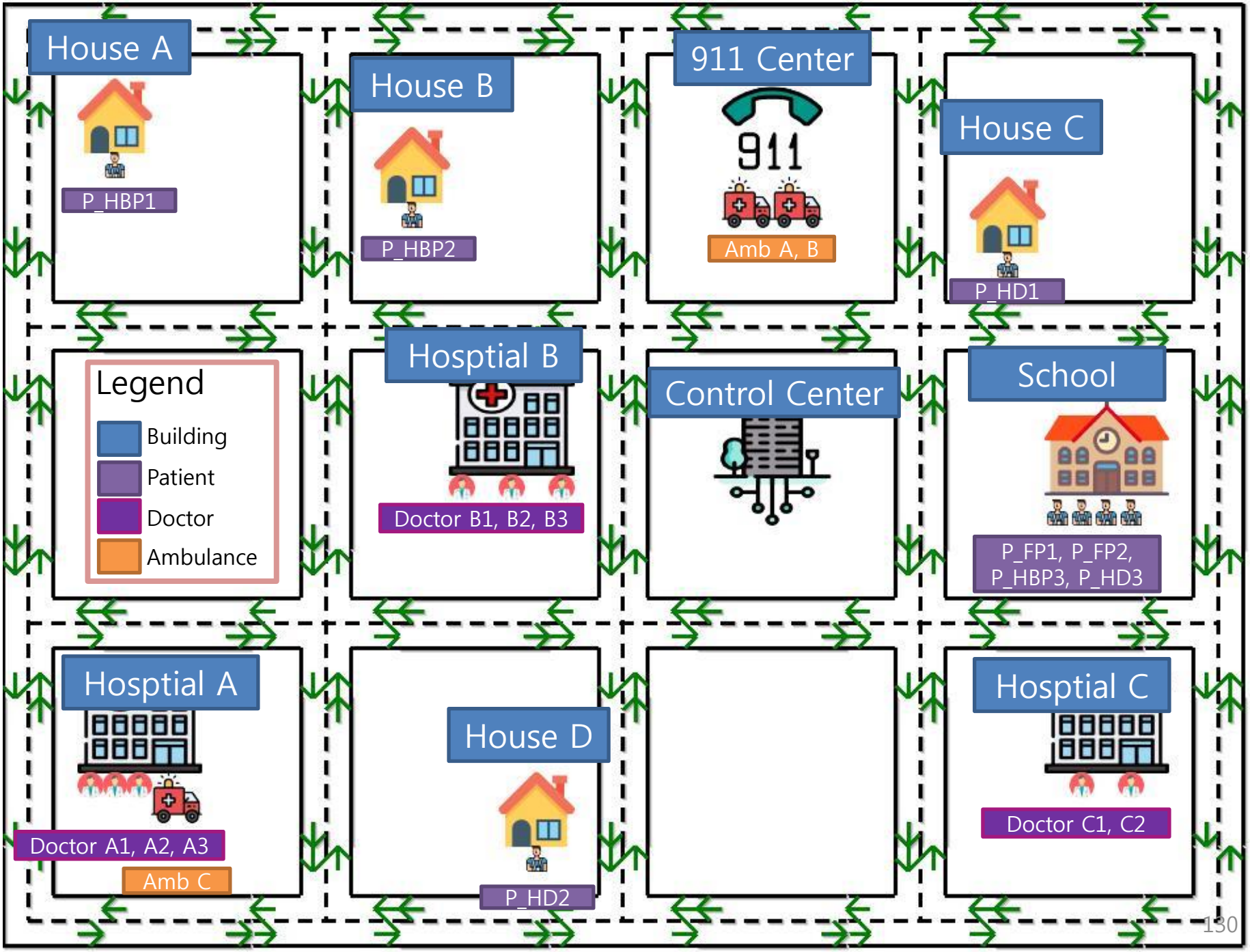
4. CYBER PHYSICAL SYSTEMS ON MODELING

Application: CPS

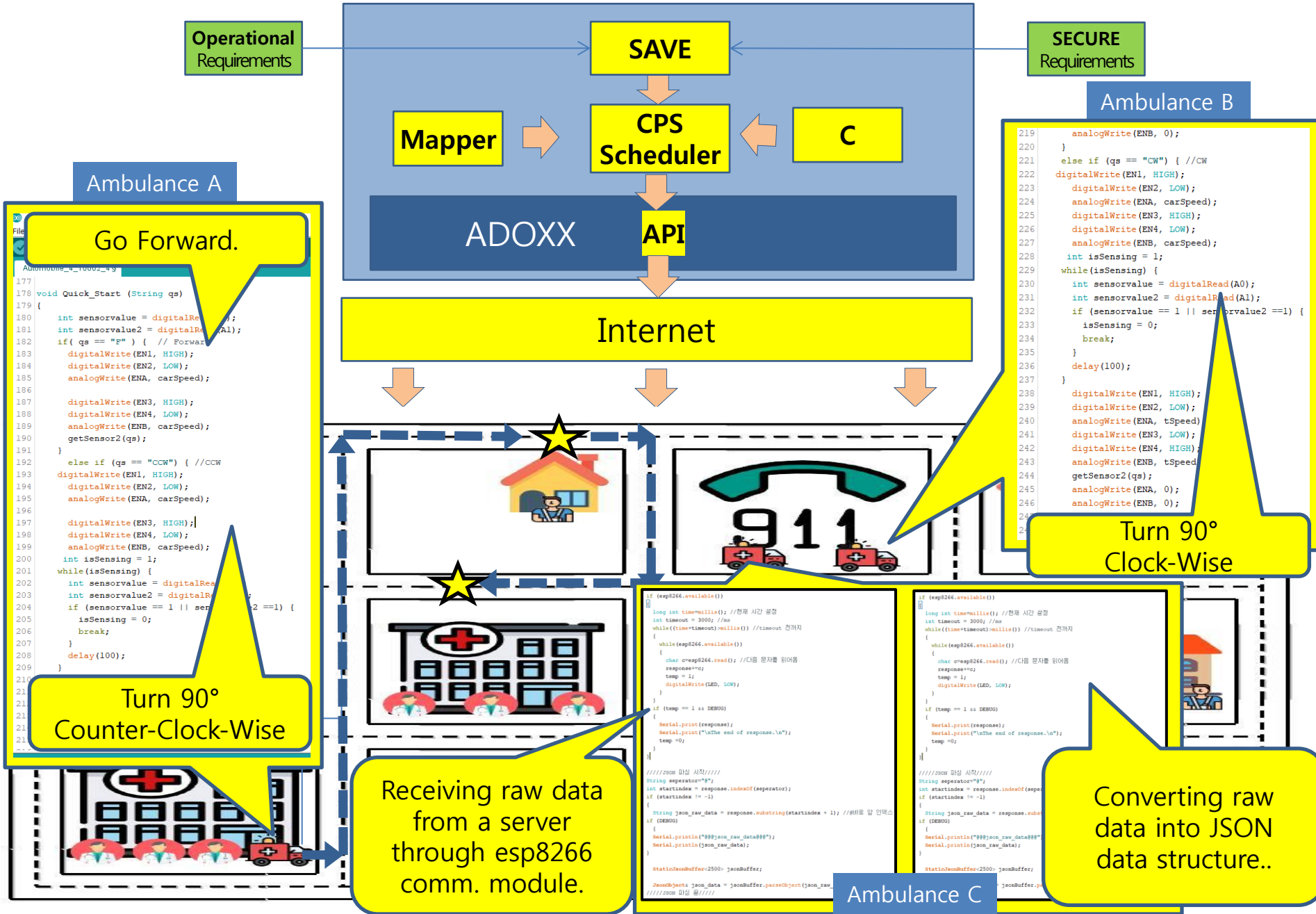


SAVE/Map

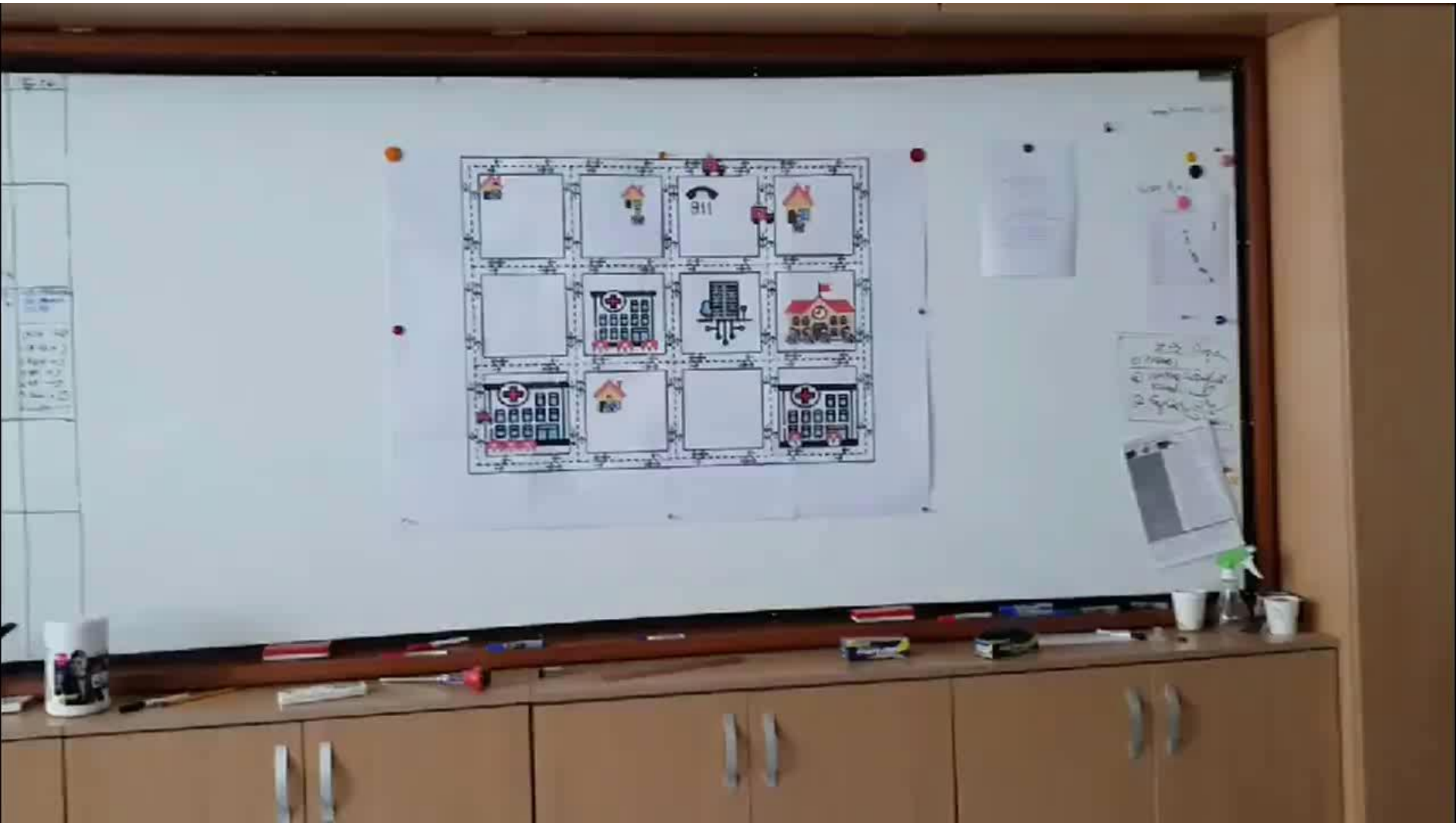

ADOxx Modelling Toolkit (gts_paper) - [EMS_Smartcity (Smartcity)]
Model Edit View Process tools Delta calculus mechanism Extras Window Help
Modelling
Explorer - Model groups
Models
2017 EMS
EMS_Smartcity
ITL - EMS_BO
ITS - EMS_BO1
ITS - EMS_BO2
Complex EMS
Simple EMS
Navigator
Event Logging
911
57.88%


House A
P_HBP1
House B
P_HBP2
911 Center
911
Amb A, B
House C
P_HD1
Legend
Building
Patient
Doctor
Ambulance
Hosptial B
Doctor B1, B2, B3
Control Center
School
P_FP1, P_FP2,
P_HBP3, P_HD3
Hosptial A
Doctor A1, A2, A3
Amb C
House D
P_HD2
Hosptial C
Doctor C1, C2

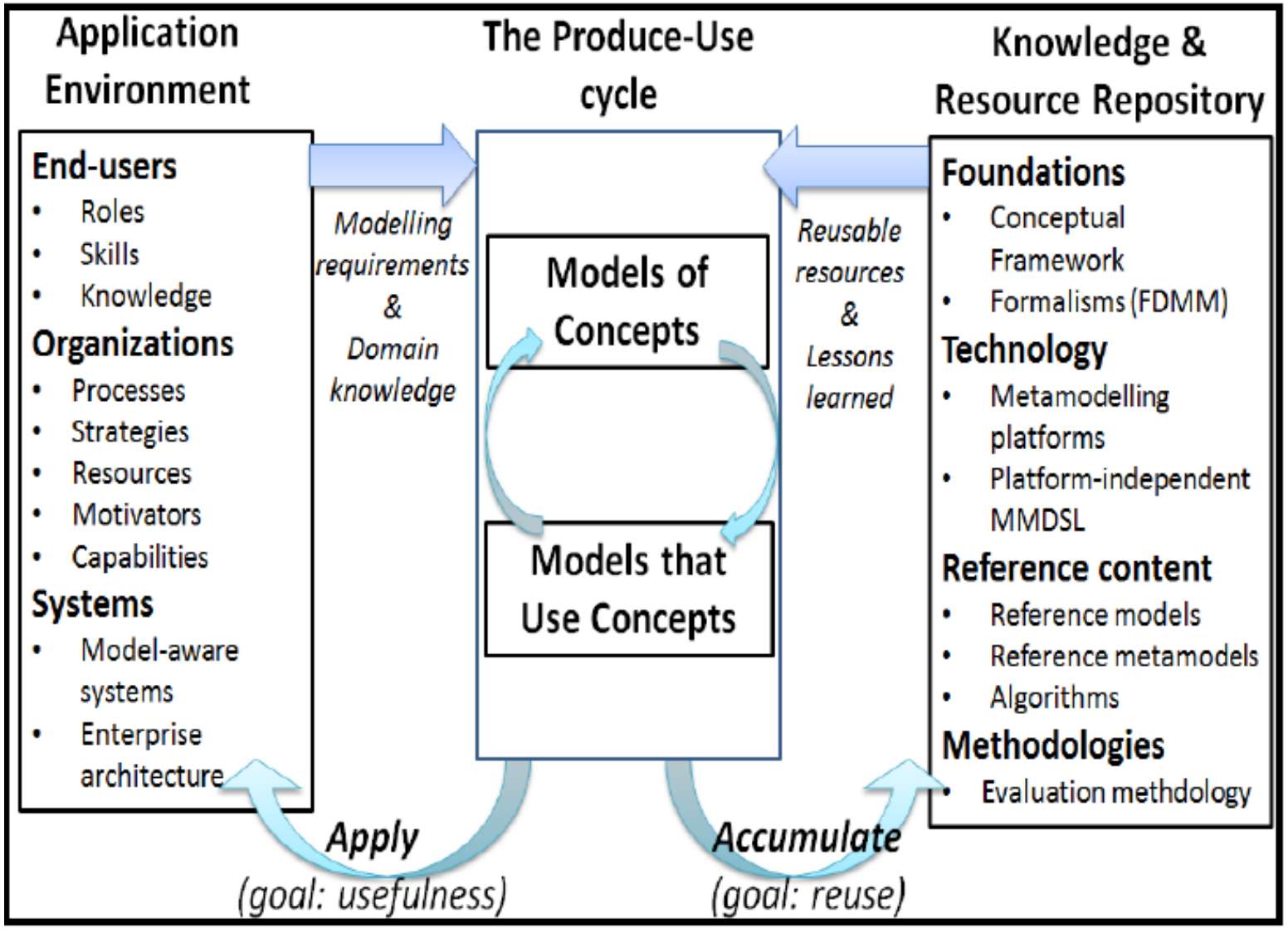
Distributed Real-time Mobile Systems: Scheduling



SAVE/EMS/CPS Example

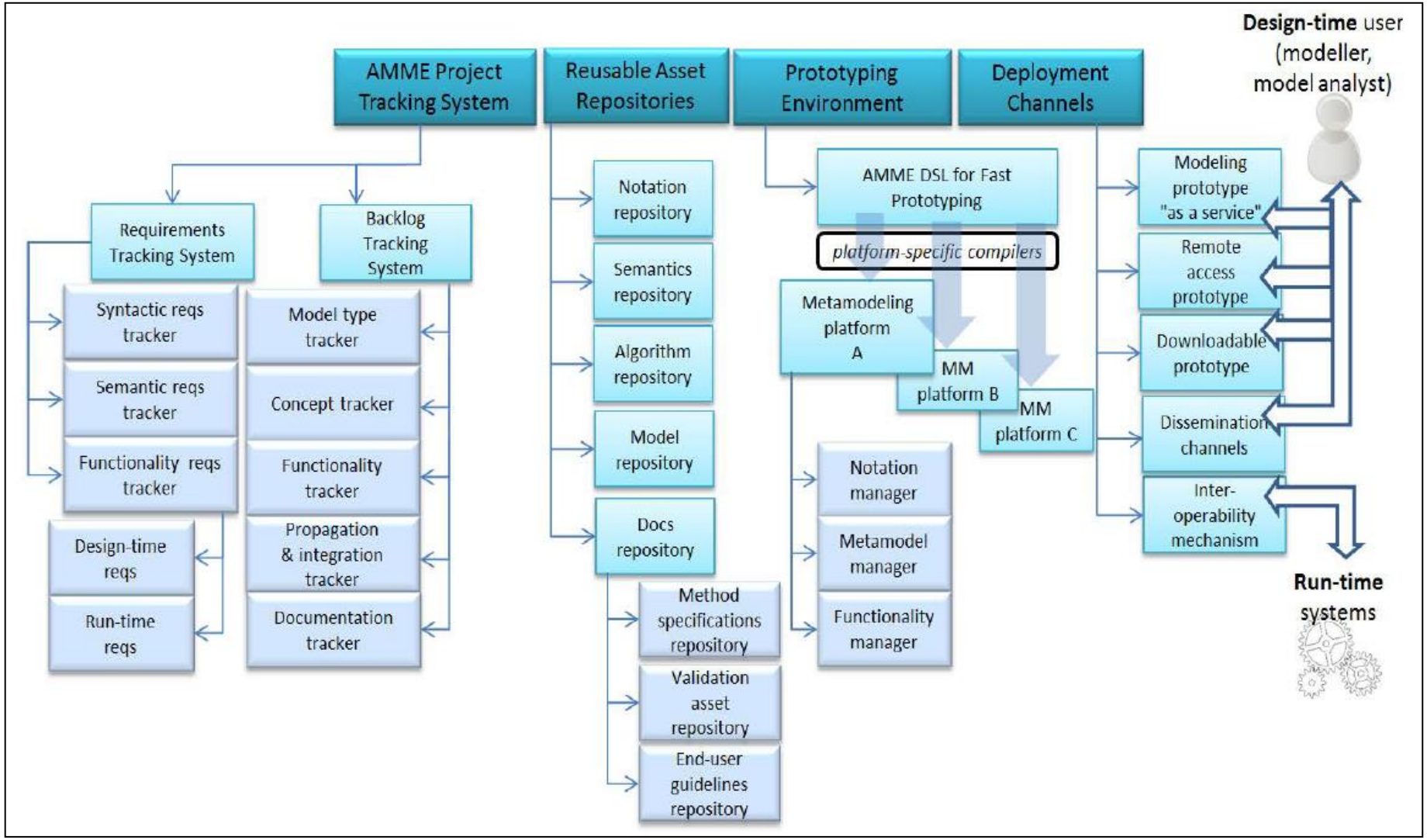


The AMME Framework



Dimitris Karagiannis, Agile Modeling Method Engineering, Proceedings of the 19th Panhellenic Conference on Informatics, Pages 5-10 , Athens, Greece — October 01 - 03, 2015.

AMME Architecture



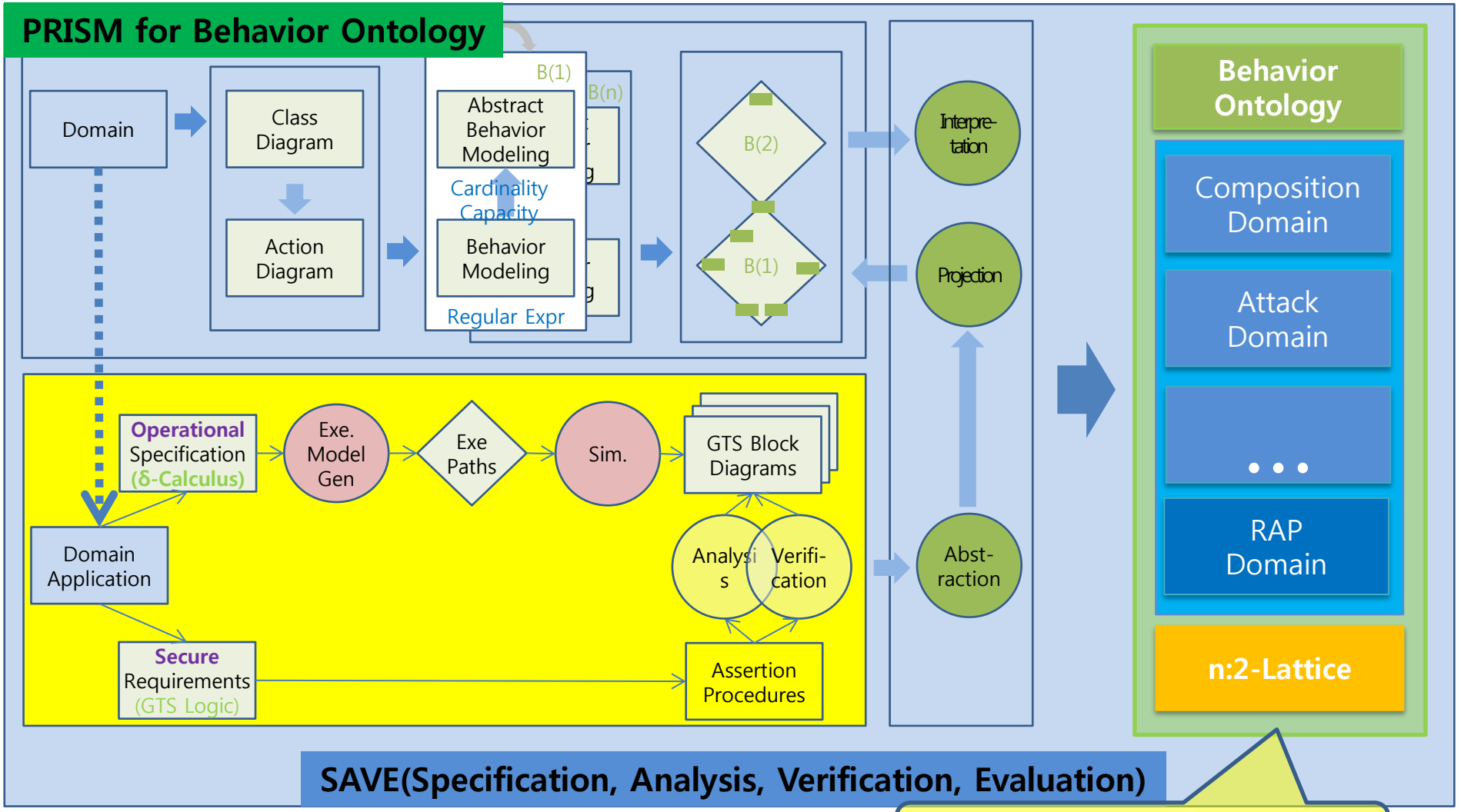
Implementation Issues

- Main Characteristics
  - Distribution
  - Real-time
  - Mobility
  - Secure: Safety, Security
  - Exceptional handling
  - Fault-tolerance
- Scheduling
  - Real-time
  - Monitoring
  - Autonomy
  - Security
  - Sensors

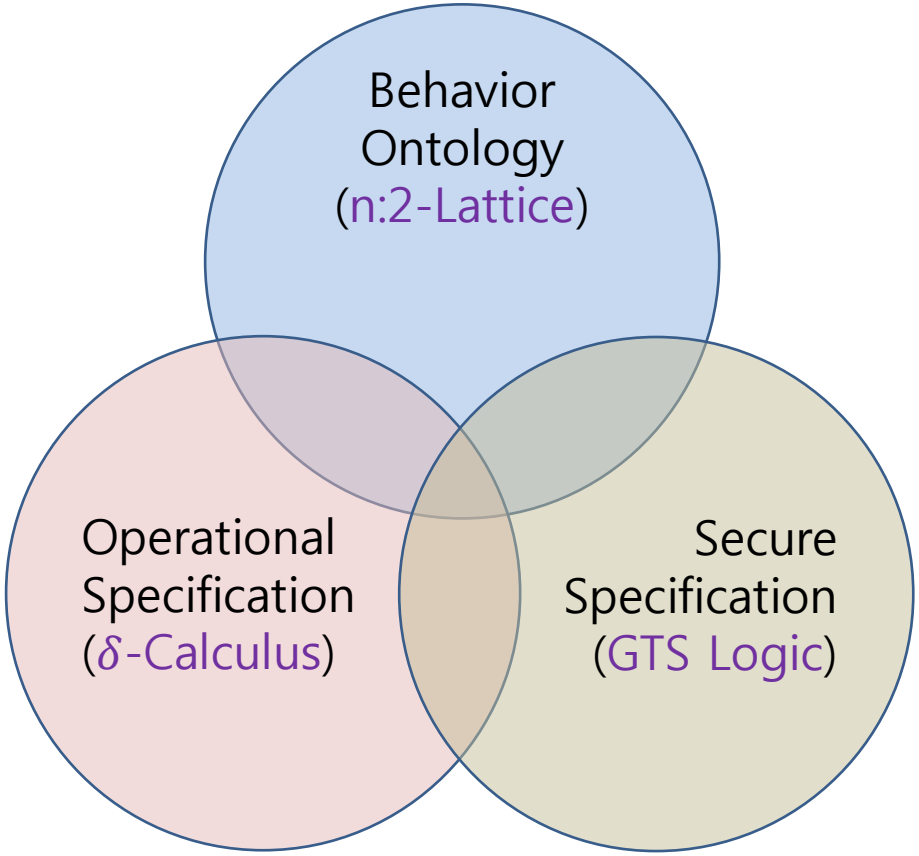
1. Motivation
2. Modeling
    1) Specification: δ-Calculus
    2) Verification: GTS Logic
    3) EMS Example
3. SAVE Tool
4. Cyber-Physical Systems Application
5. **Summary**
6. Discussion

5. SUMMARY

Approach: SAVE



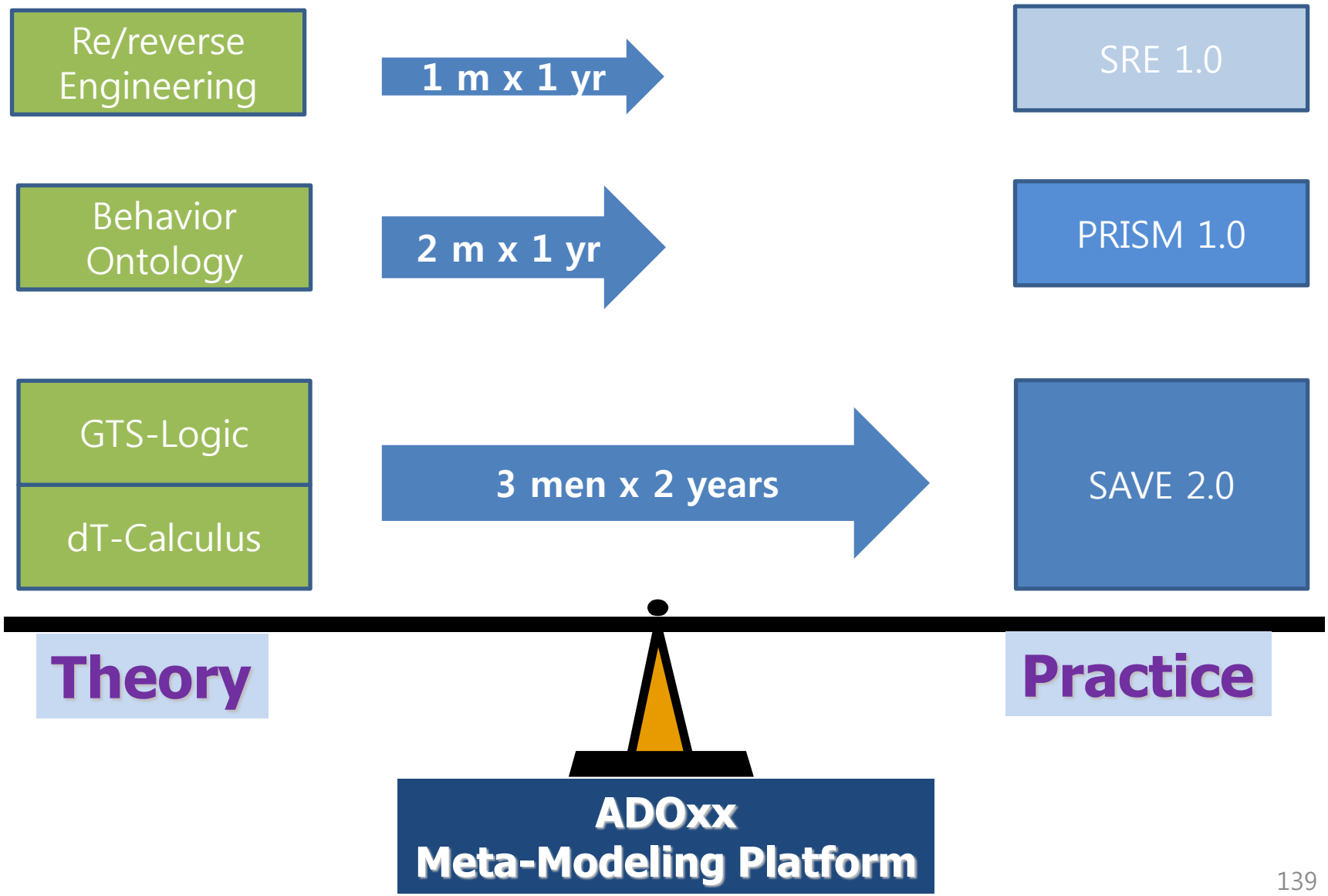
Theories

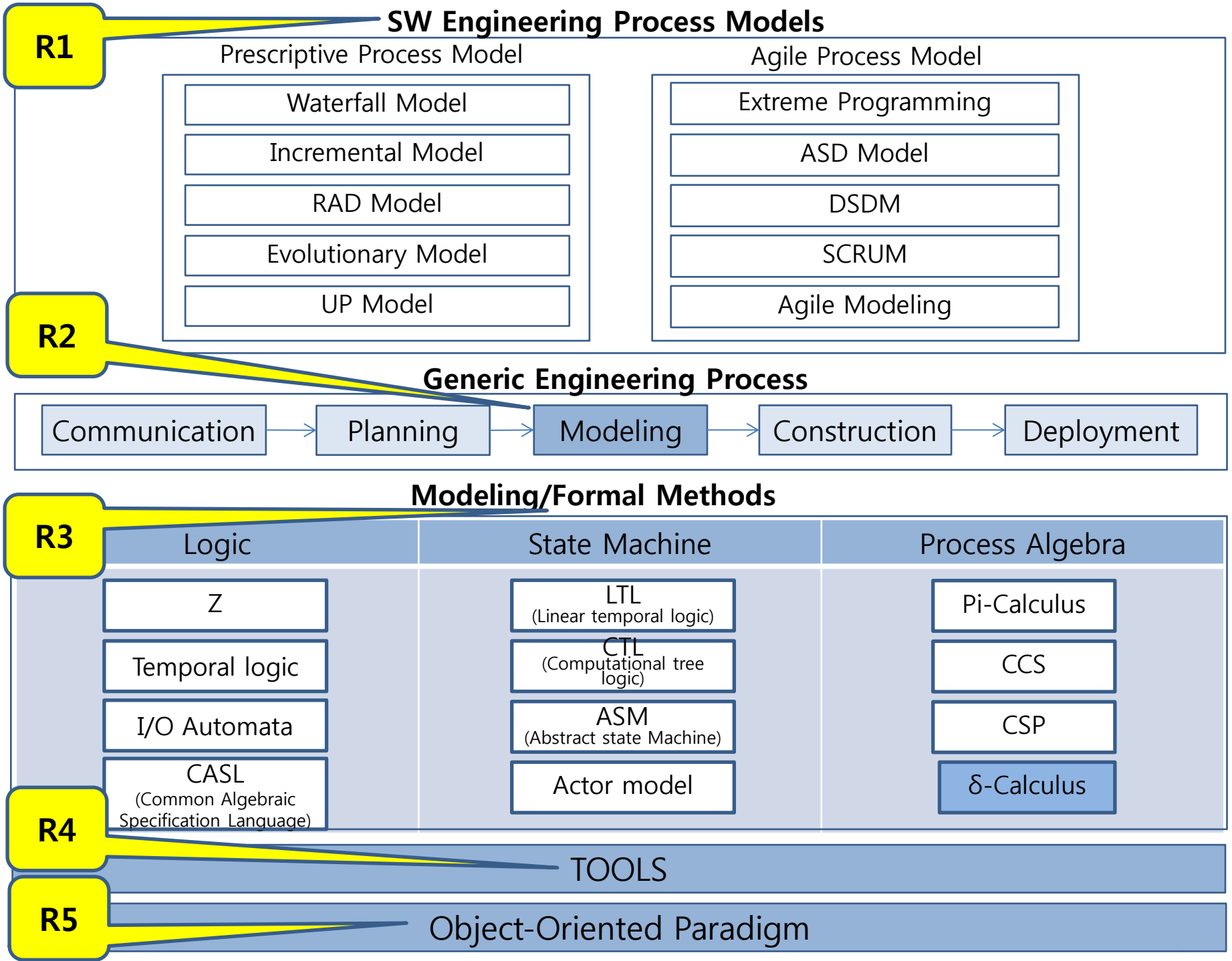


Meta-Modeling

Modeling

SW Engineering




R1
SW Engineering Process Models
Prescriptive Process Model
Waterfall Model
Incremental Model
RAD Model
Evolutionary Model
UP Model
Agile Process Model
Extreme Programming
ASD Model
DSDM
SCRUM
Agile Modeling
R2
Generic Engineering Process
Communication
Planning
Modeling
Construction
Deployment
Modeling/Formal Methods
R3
Logic
State Machine
Process Algebra
Z
Temporal logic
I/O Automata
CASL
(Common Algebraic Specification Language)
LTL
(Linear temporal logic)
CTL
(Computational tree logic)
ASM
(Abstract state Machine)
Actor model
Pi-Calculus
CCS
CSP
δ-Calculus
R4
TOOLS
R5
Object-Oriented Paradigm

R1 & 2: FMM: Formal Method Model

SW Engineering Process Model

Modeling : Design/Specification

CPS Implementation

Requirements Extraction/Analysis

Specificati & Verificati

Implementation & Testing

추출

분석

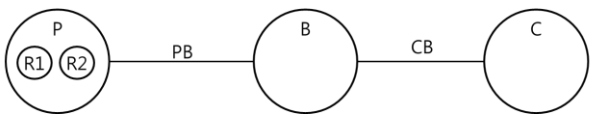
명세

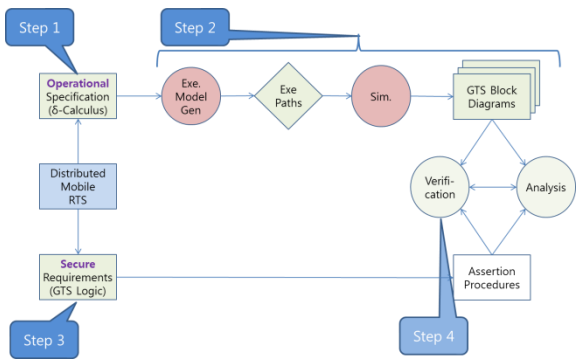
검증

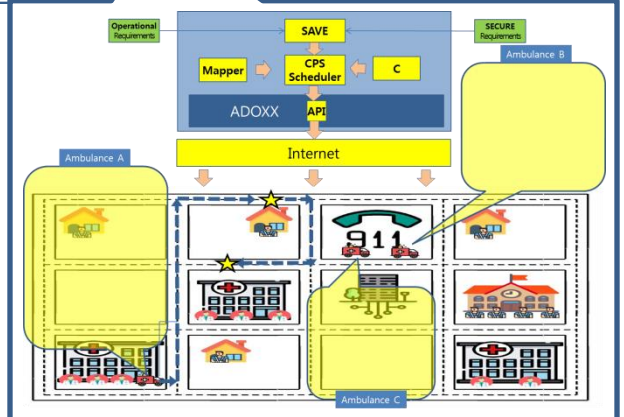
구현

검사

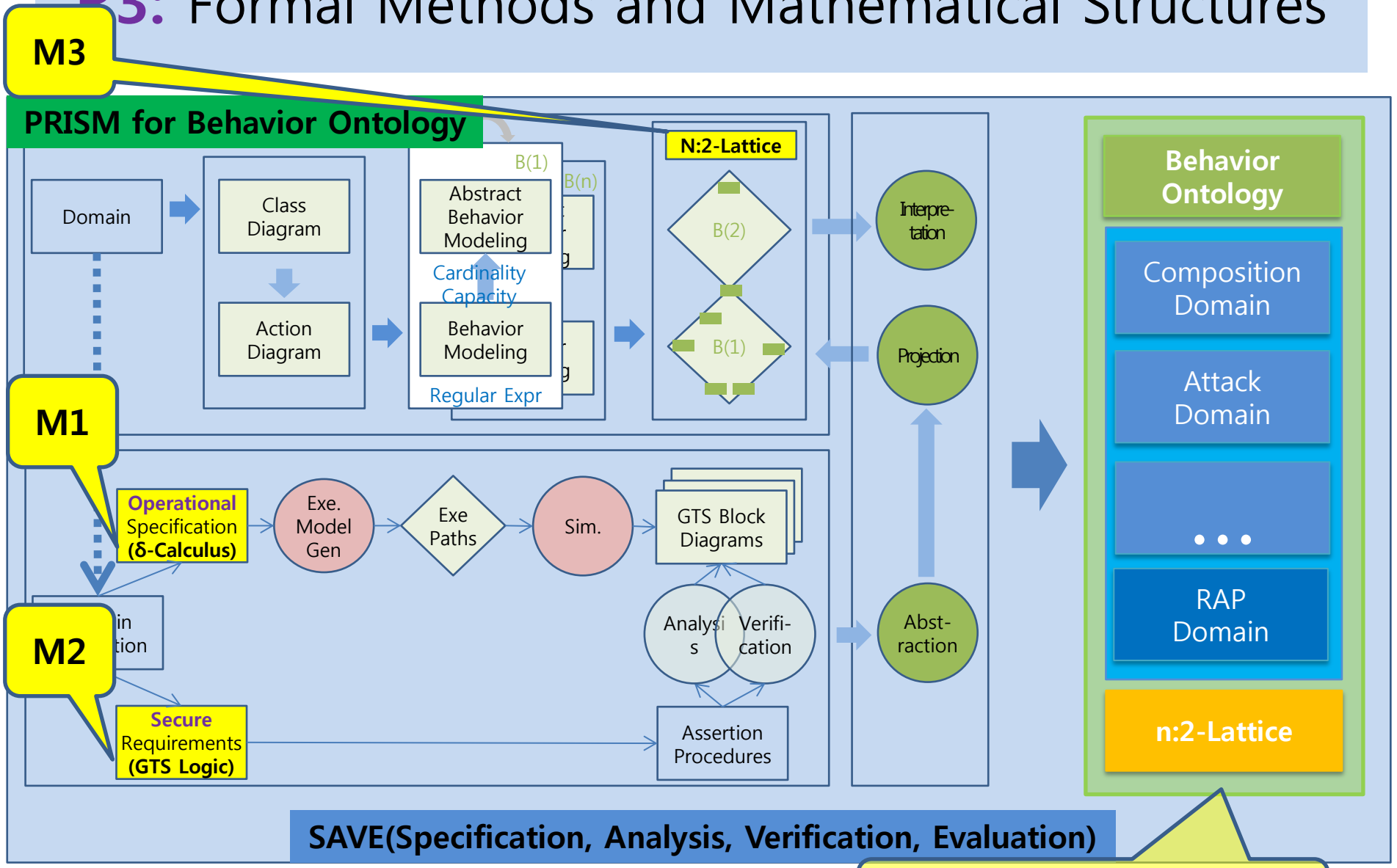
- Operational Requirements
  - *Producer* produces two resources, R1 and R2.
  - *Producer* stores the resources in *Buffer* in order.
  - *Producer* informs *Buffer* of the order of R1 and R2, or R2 and R1.
  - *Consumer* consumes the resources from *Buffer* in order.
  - The order of the consumption is formed to *Buffer* by *Consumer*.
- Secure Requirements
  - Security:
    - The order should not be violated, since the first resource contains security information to decode the second resource.
    - The propagation between the first and the second should be less than 30 seconds.
  - Safety
    - The resources produced by *Producer* should be consumed by *Consumer* less than 5 minutes.



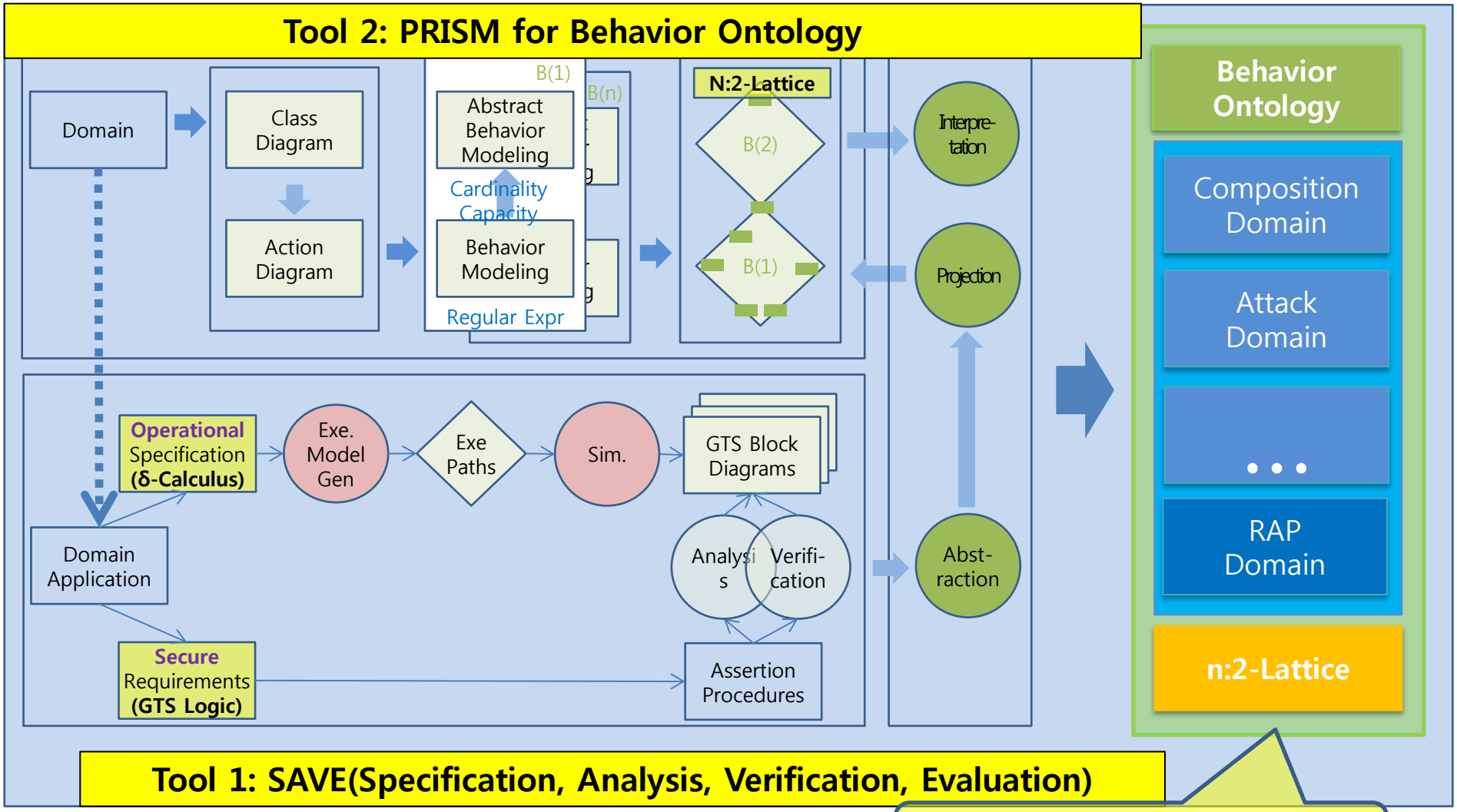




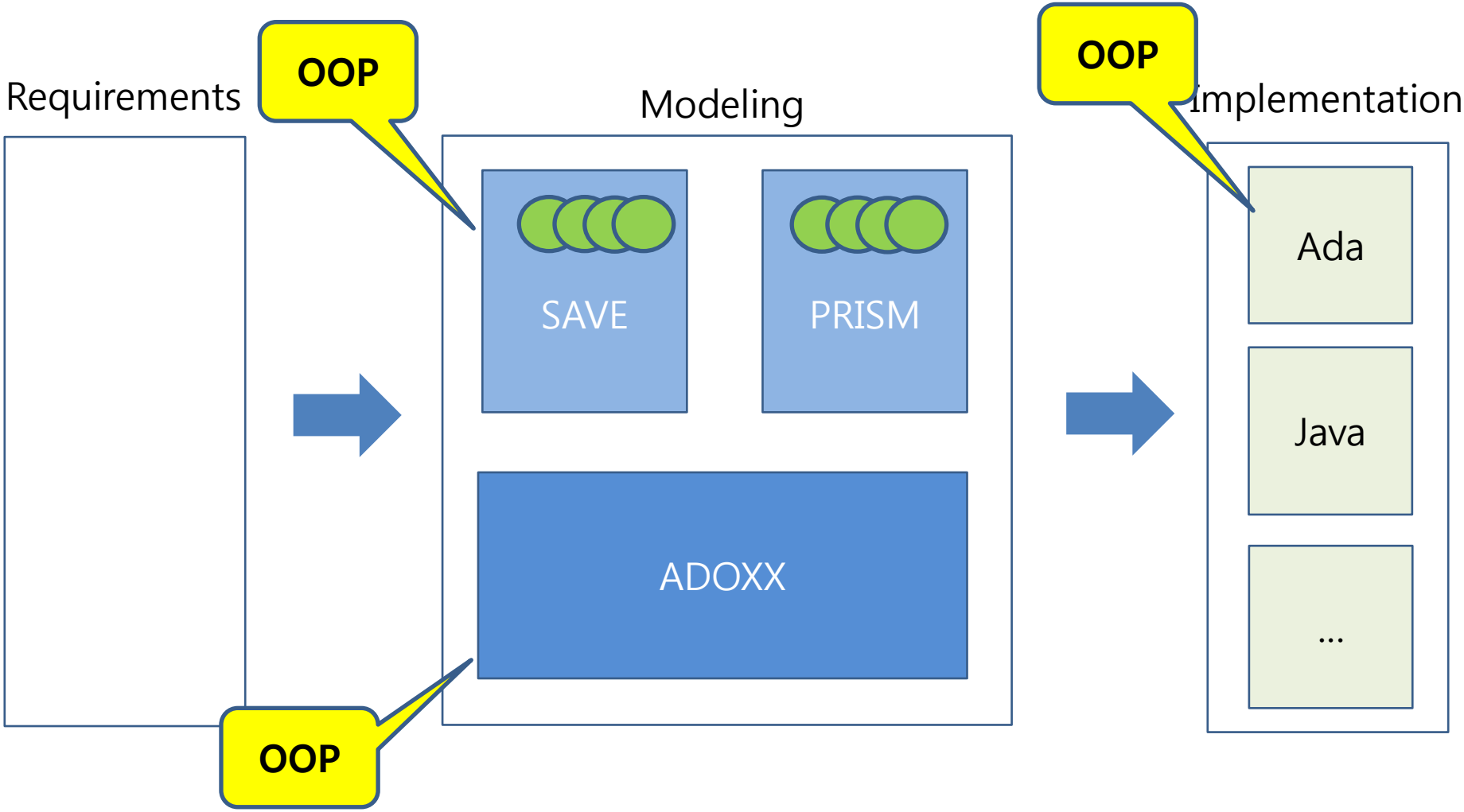
R3: Formal Methods and Mathematical Structures



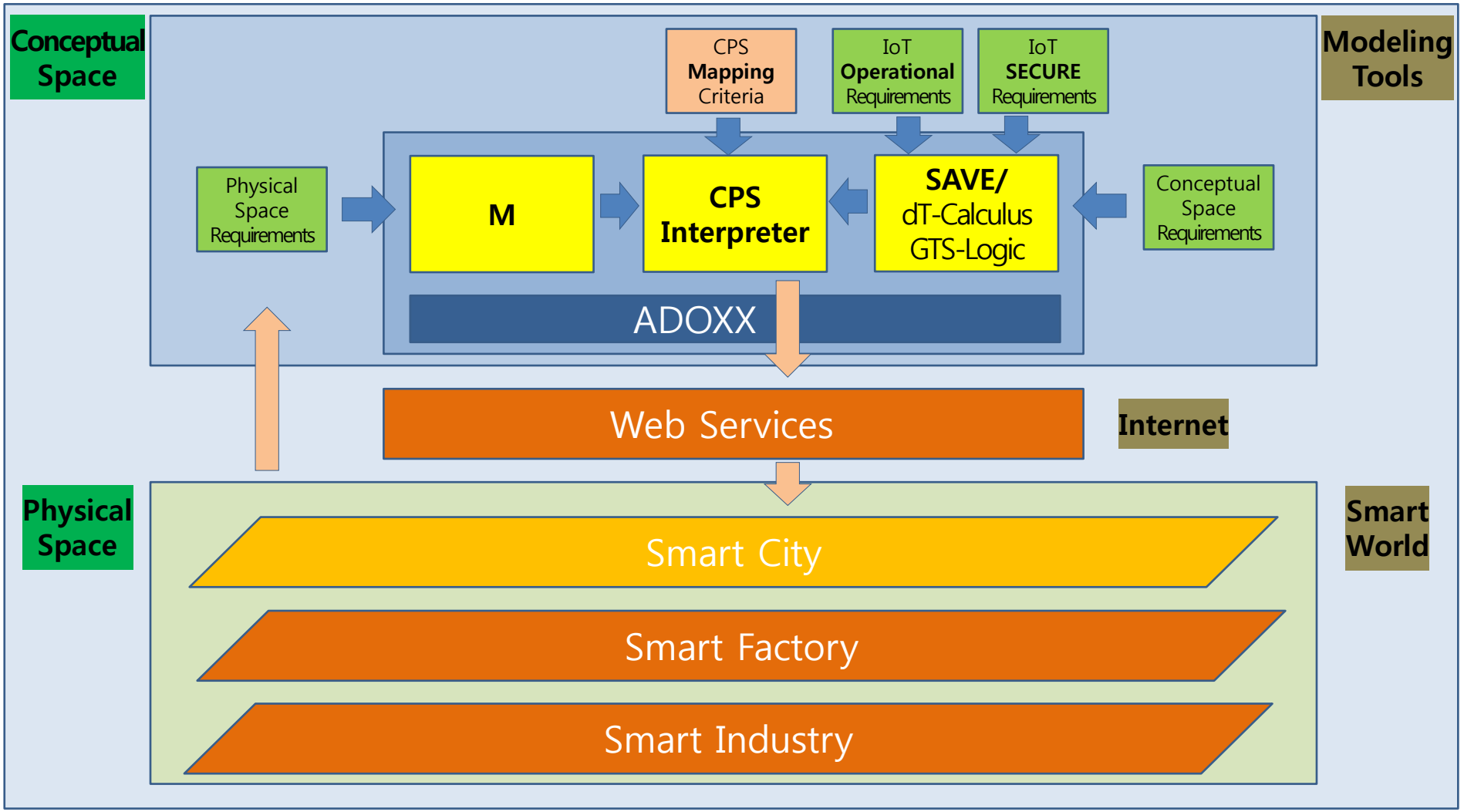
R4: SAVE & PRISM



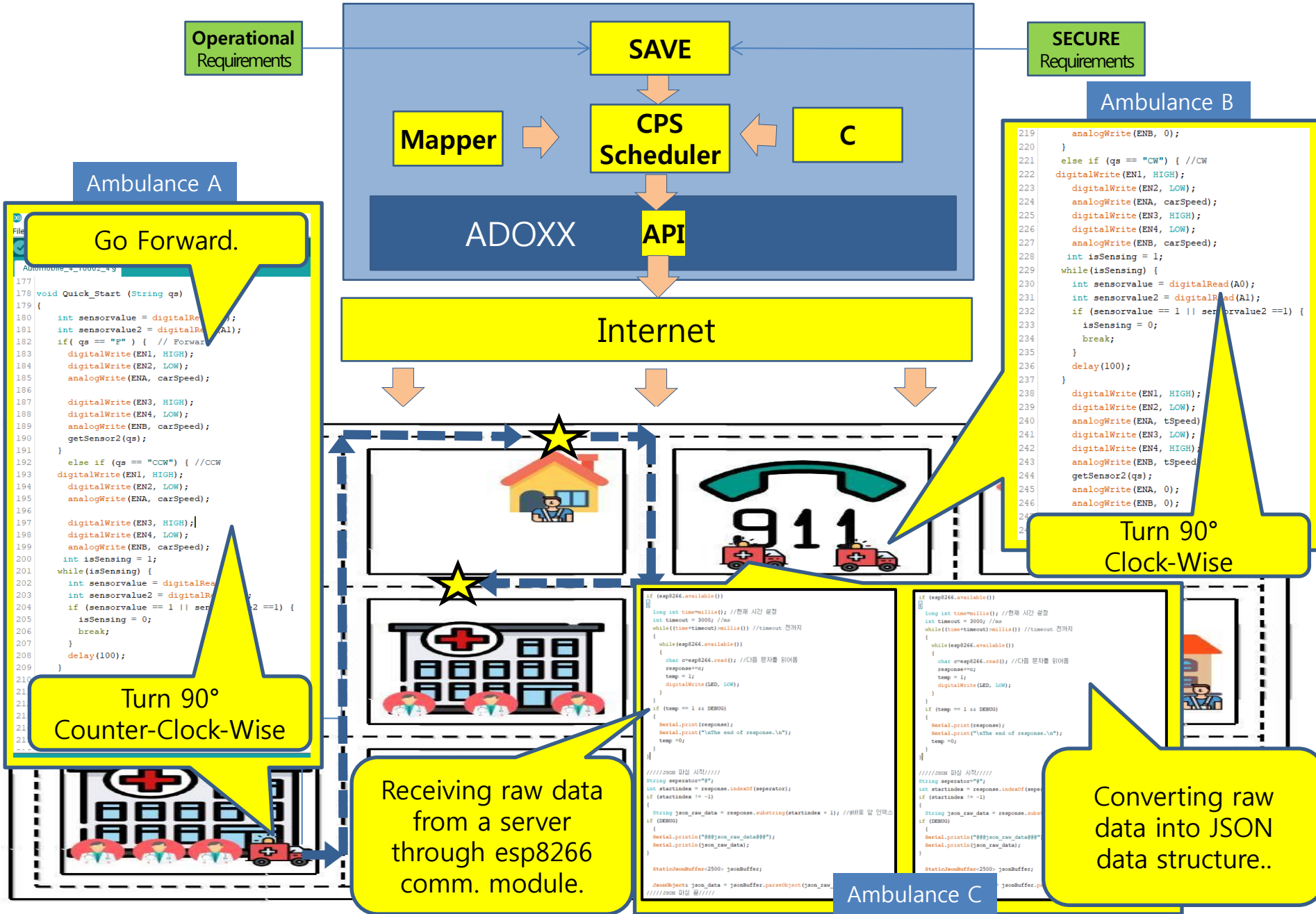
R5: Object-Oriented Paradigm



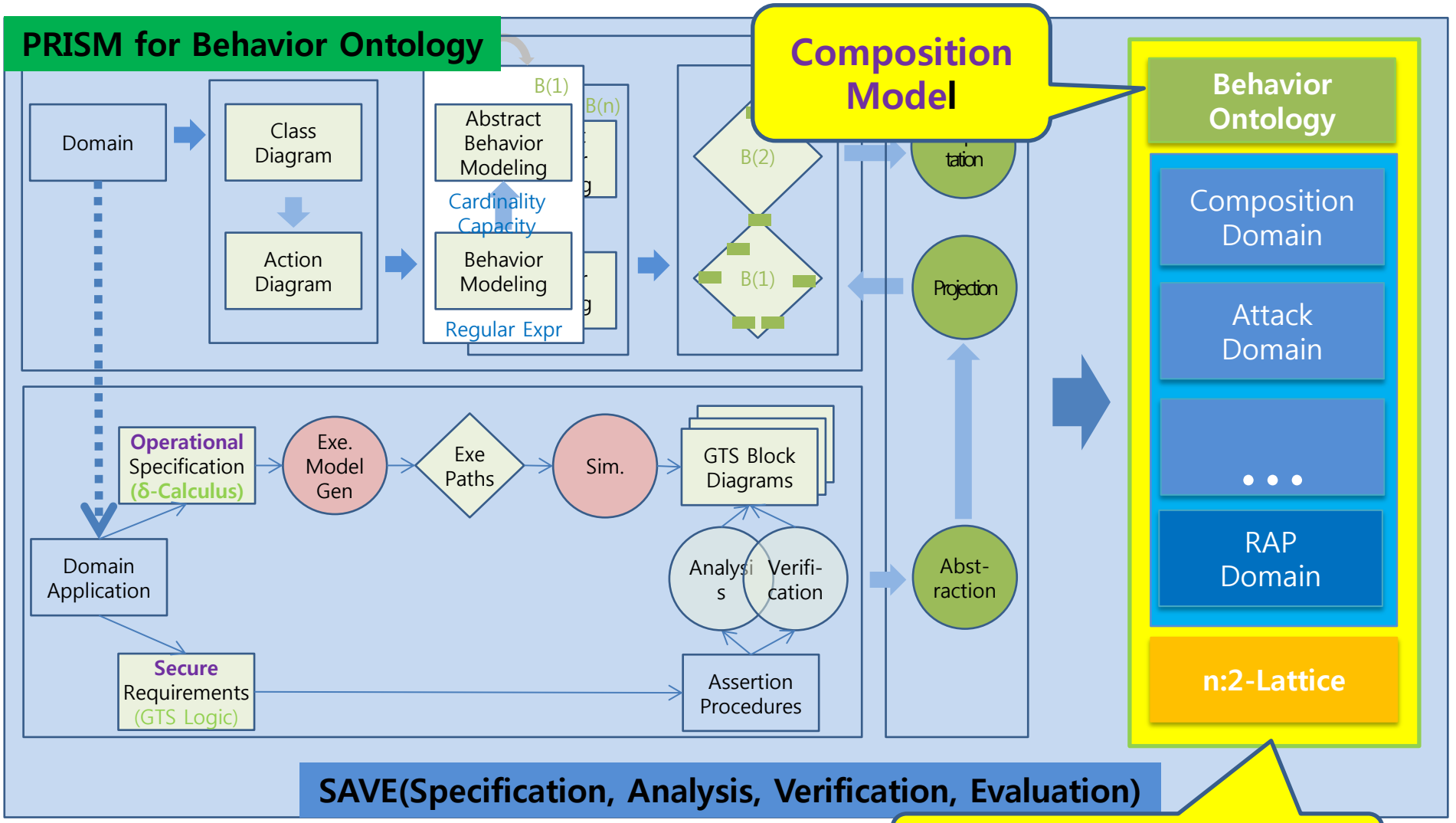
Application: CPS



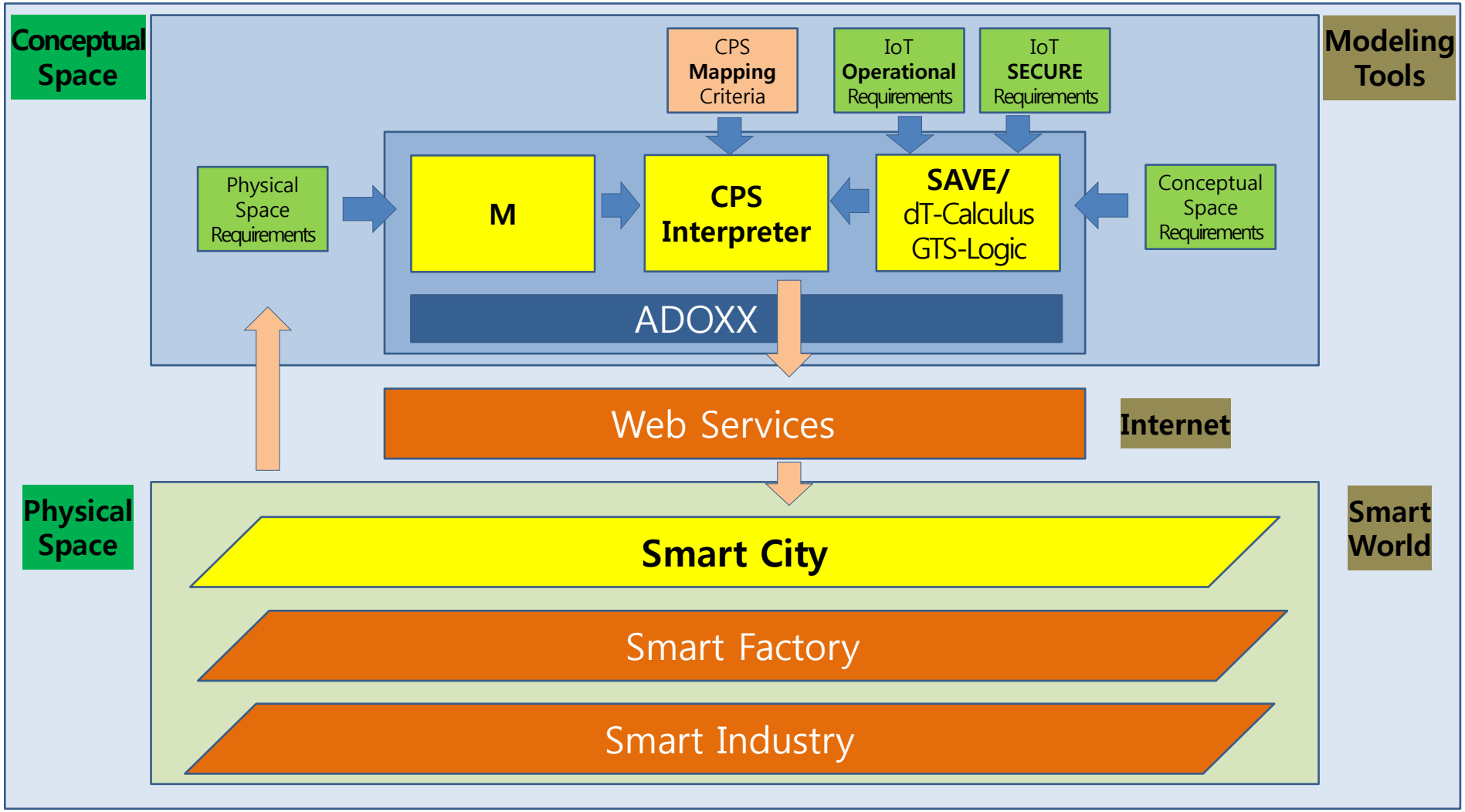
Distributed Real-time Mobile Systems: Scheduling



NEMO'19-A: Knowledge Engineering



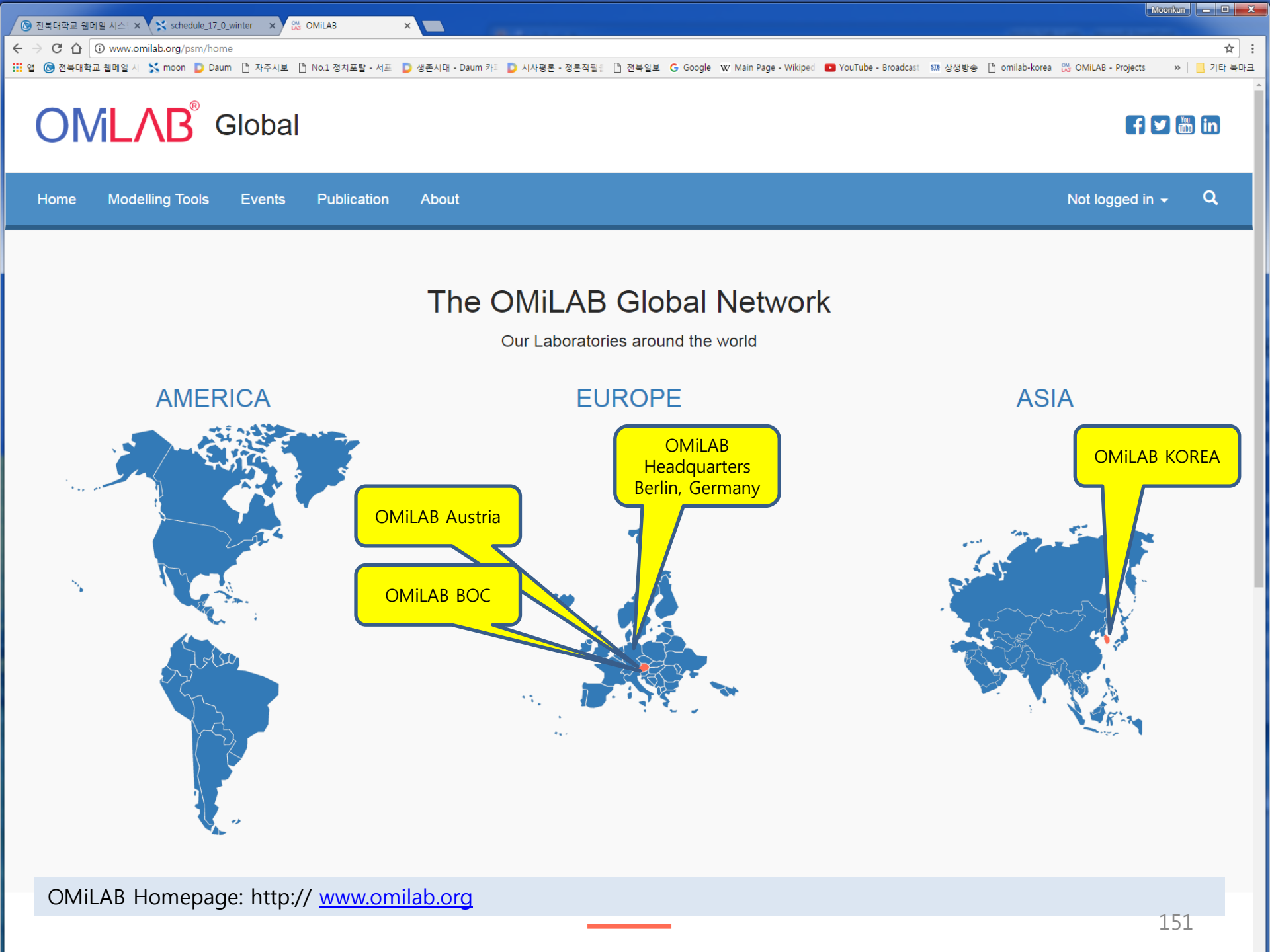
NEMO'19-B: CPS/Smart City



1. Motivation
2. Modeling
    1) Specification: δ-Calculus
    2) Verification: GTS Logic
    3) EMS Example
3. SAVE Tool
4. Cyber-Physical Systems Application
5. Summary
6. **Discussion**

6. DISCUSSION


info @ omilab.org
Call Us Now: 0049 3026 36 78 63
OMiLAB
HOME
OMILAB NODES
ACTIVITIES
ABOUT US
CONTACT
Design Thinking
Conceptual Modelling
Digital Innovation
OMiLAB Nodes
Education & Training
Tools
Projects & Events



OMiLAB Homepage: http:// www.omilab.org

OMiLAB – Open Innovation for Digital Transformation

FoF EU Project and OMiLAB Nodes develop Digital Services for Cyber Physical Systems.

3 OMiLABs existing:
- 2 x Vienna, Chonbuck

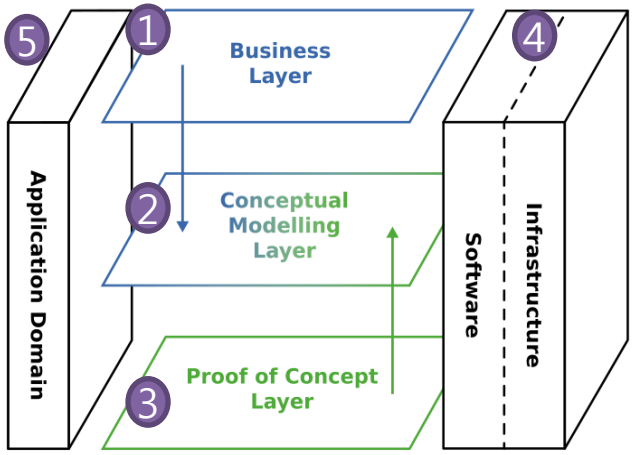
http://go0dman-project.eu/

5 OMiLABs in preparation:
- Sibiu, Bergamo, Warsaw, St. Etienne, City of Oulo

https://digifof.eu/

3 OMiLABs under Investigation
- 2 x Germany, 1 x South Korea 1 x Poland, 1 x Slovenia

http://omilab.org



Innovate, Develop and Evaluate Digital Services considering:

1. **Scenario Layer** – using Scene2Model Environment
2. **Conceptual Model Layer** – using the Bee-Up Tool
3. **Run-time Layer** – using Dobot Magician, Makeblock mbot
4. **Software** – using ADOxx, OLIVE, …
5. **Application Domain:** i.e. Factory of the Future

OMiLAB Nodes

Home / OMiLAB Nodes

All network nodes share common tools and processes, while focusing on their individual core topic. Together we build a worldwide distributed laboratory with multi-disciplinary competences.

Build Your Node!

Best Service

OMiLAB Nodes

The Vienna Node - Austria

Here in Vienna our Focus is on Knowledge-based Methods and Technologies for Digitalisation.

Read More

The Chonbuk Node - Korea

Here in Chonbuk our focus is on formal methods and the Internet of Things.

Read More

Your Node - Next

You can launch your own OMiLAB and interact with our global community.

Read More

OMiLAB Book Series

Domain-specific Conceptual Modelling

The book draws new attention to domain-specific conceptual modelling by presenting the work of thought leaders who have designed and deployed modelling methods. It provides hands-on guidance on how to build models in specific application domains. In addition to these results, it also puts forward ideas for future developments. All this is enriched with exercises, case studies, detailed references and further related information.

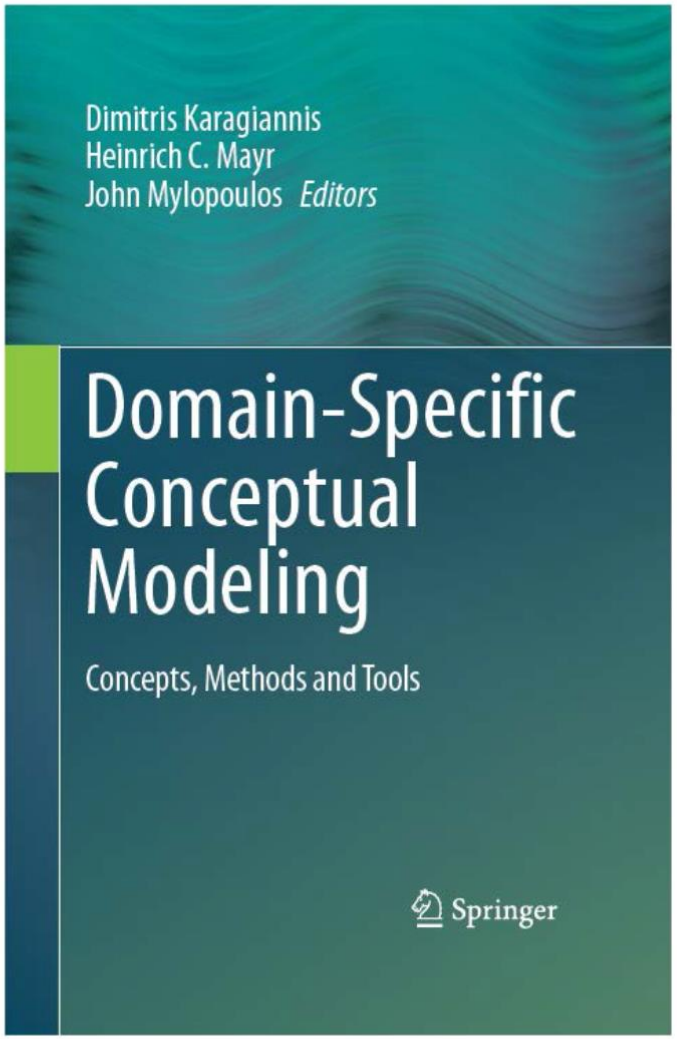
All domain-specific methods described in this book also have a tool implementation in the OMiLAB Tool section [Link catre Tools], which has been made available by the OMiLAB Node Vienna – a dedicated research and experimentation space for modelling method engineering.

The domains addressed by the modelling methods and tools contained in this book are:

- Modelling Method Conceptualization
- Big Data
- Business Process Management
- Business and Process Transformation
- Enterprise Information Systems
- Enterprise Strategic Management
- Internet of Things
- Knowledge Engineering
- Production Management Systems
- Requirements Engineering
- Service Science: Social Implications
- Technology Enhanced Learning

The collection of works presented here will benefit experts and practitioners from academia and industry alike, including members of the conceptual modelling community as well as lecturers and students.

To get access to the book and its chapters, please visit "Springer Publishing"



Dates

Start: July 6, 2020 at 9 a.m.

End: July 17, 2020 after 5 p.m.

Location

University of Vienna
Faculty of Computer Science
Währinger Str. 29 Vienna, Austria

NEMO Brochure
Download pdf

REGISTER
Registration form

NEMO Website
nemo.omilab.org

NEMO Summer School

Today's students will work in and for digitized organisations where smart devices, digital artefacts, intelligent machines and robots, data streams and connectivity are ubiquitous. In their work the will face human/machine interaction challenges, lifecycle challenges (e.g. ICT embedded in the lifecycle of consumer products), disruptive business models and increased questions about privacy and security. Additionally, a higher level of automated processing of digital information as well as the "end-to-end" integration of processes across multiple organizations and customers is required by the users.

The NEMO Summer School Series focuses on addressing these challenges through modelling, both in theory and practice. How to define modelling with the 'right' level of abstraction and how to engineer suitable modelling tools is at the heart of the summer school.

NEMO 2019 SUMMER SCHOOL

Participants

Austria	4	Italy	2
Belgium	1	Japan	2
Chile	3	Korea	6
Croatia	2	Luxembourg	1
Denmark	1	Poland	1
Estonia	2	Romania	5
France	4	South Africa	1
Germany	4	Sweden	1
Greece	2	Switzerland	6
India	3	Tunisia	1
Ireland	2		

Total:
54 Participants
21 Countries

NEMO 2019 Summer School - Programme Overview

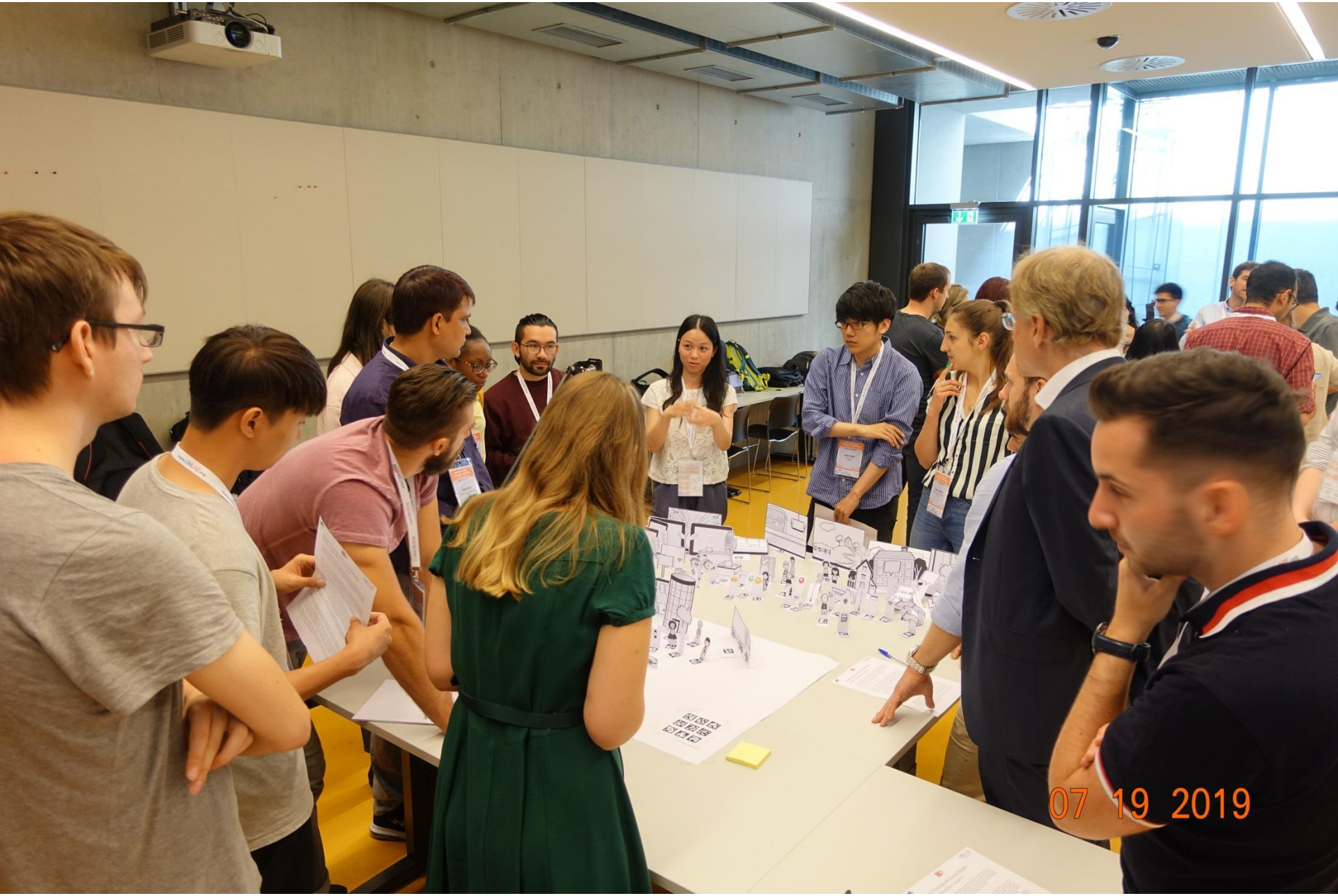
ADOXX TRAINING DAYS

	10 - 12 July	15 July	16 July	17 July	18 July	19 July
09:00 - 10:00		Opening Ceremony W. Gansterer, Dean Fac. of Comp. Science	Modern Approaches in Data Engineering and Information System Design I. Lukovic	Modelling Knowledge Action and Time: Action Theories and Their Application in Dynamic Domains D. Plexousakis, T. Patkos	Digital Transformation: Better Guided than Chaotic J. Ralyte	OMiLAB NPO: An Introduction D. Karagiannis, M.K. Lee
10:00 - 11:00		The Power of Model-Centering H.C. Mayr	N.N. N. Choi	Space of Services (SoS) - a method of design and improvement of services V. Strahonja, M. Tomicic Furjan	Foundations and Applications of Business Process Compliance S. Rinderle-Ma	OMiLAB@work A Smart City Case - Design Thinking Workshop The OMiLAB-Team
11:00 - 11:30		B R E A K		B R E A K	B R	E A K
11:30 - 12:30		How can Conceptual Modelling Support Digitalization? D. Karagiannis	Security Assessment Using SAPnet in the Internet of Things (IoT) Ecosystem C. Douligeris	Service Engineering models for the design and development of Product-Service Systems S. Cavalieri	Japanese Creative Service as a Next Generation Enterprise Modelling Y. Hara, H. Masuda	OMiLAB@work A Smart City Case - Parallel Working Groups The OMiLAB-Team
12:30 - 14:00		L U N C H		L U N C H	L U	N C H
14:00 - 15:00		Bee-UP The ADOxx-Team	Value Modelling: from current practice to future applications B. Roelens	Capability-oriented Enterprise Modelling for Mastering Dynamic Business Context J. Zdravkovic	Integrated Decision and Process Modelling J. Vanthienen	OMiLAB@work A Smart City Case - Parallel Working Groups The OMiLAB-Team
15:00 - 16:00		Practice Session - DSMM H.C. Mayr, M. Paczona	Collaborative and well-behaved outdoor robots in harsh environment J. Röning	Participatory Enterprise Modeling with the 4EM Method J. Stirna, B. Lantow	Modelling Knowledge Work: Integrating Decision-aware Business Processes and Case Management K. Hinkelmann	OMiLAB@work A Smart City Case - Parallel Working Groups The OMiLAB-Team
16:00 - 16:30		B R E A K		B R E A K	B R	E A K
16:30 - 17:30		FUJITSU Rebuilding Trust in the Digital Age Y. Takashige	HILTI Intelligent Customer Interactions Require an Intelligent System Setup M. Petry	Practice Session - 4EM J. Stirna, B. Lantow	Practice Session K. Hinkelmann	OMiLAB@work A Smart City Case - Parallel Working Groups The OMiLAB-Team
		Get Together | Open End				

LEISURE DAYS

	20 - 21 July	22 July	23 July	24 July	25 July	26 July
09:00 - 10:00		Multi-Perspective Enterprise Modelling as a Foundation of IT-Business Alignment U. Frank	Hybrid Knowledge Bases: the Interplay between Domain-specific Models and Knowledge Graphs R. Buchmann, A.M. Ghiran	Process Algebra to Model Probabilistic Behavior of Smart IoT M. Lee	Enterprise Modelling and Business Intelligence W. Grossmann, C. Moser	STUDENT PRESENTATIONS
10:00 - 11:00		Domain Storytelling: A Modelling Approach for Business Processes H. Züllighoven, S. Hofer	The industrial transition towards Product-Service-Systems: articulating enterprise modelling and economic model balancing X. Boucher	Supporting Business Process Improvement through a Modeling Tool F. Johannsen	Agent-oriented Cyber-physical Systems Modelling C. Cares	STUDENT PRESENTATIONS
11:00 - 11:30		B R E A K		B R E A K	B R	E A K
11:30 - 12:30		Parallel Practice Session U. Frank/ H. Züllighoven	Parallel Practice Session R. Buchmann/ X. Boucher	Parallel Practice Session M. Lee/ F. Johannsen	Parallel Practice Session W. Grossmann/ C. Cares	STUDENT PRESENTATIONS
12:30 - 14:00		L U N C H		L U N C H	L U	N C H
14:00 - 15:00		A User-Centric Platform PRINTEPS to Develop Intelligent Robot Applications T. Yamaguchi	Enterprise Modeling for Continuous Requirements Engineering M. Kirikova	Capability Oriented Requirements Engineering E. Kavakli	Grounded Enterprise Modelling E. Proper	STUDENT PRESENTATIONS
15:00 - 16:00		Joining Distributed Ledger Technologies and Enterprise Models: The Concept of Knowledge Blockchains H.G. Fill	Systematic development of web information systems B. Thalheim	Business Processes for Business Communities A. Oberweis	Fractal Enterprise Model and its Usage for Business Transformation I.Bider, E. Perjons	STUDENT PRESENTATIONS
16:00 - 16:30		B R E A K		B R E A K	B R	E A K
16:30 - 17:30		ATOS	How to model your eco-system? J. Gordijn	Quality Assurance for BPMN Models A. Polini	Trials & tribulations of PhD research and beyond P. Loucopoulos	Closing Ceremony


NEMO
Next-Generation Enterprise:
Modelling in the Digital Age
SUMMER SCHOOL SERIES
Opening Ceremony 2019
nemo.omilab.org


07 19 2019

Know About

ADOxx Platform Trainings

OMiLAB community members benefit from free ADOxx Platform Trainings, which are provided by our partner the ADOxx.org team. Typically held four times a year in Vienna or upon request at a community member's location the three day training programme teaches method- and software engineers in the basics of coding on ADOxx.

The interactive training sessions enable participants to develop individual notation, syntax and semantic without programming effort.

The training curriculum includes:

- Setting up the implementation environment
- Modelling language implementation (including classes, relations, attributes and attribute facets, model types)
- Mechanisms & algorithms implementation (including core functions for model manipulation, configuration of ADOxx components, external coupling of ADOxx functionality and add-on implementation)
- Software packaging and deployment

At the end of the training each participant is able to take home an individual installable and distributable software package.

Click HERE for more information on ADOxx Trainings.

Overview

Project Name:	DigiFoF
Funding:	European Union
Program: Alliance	Erasmus+ Knowledge
Start:	January 2019
End:	December 2021

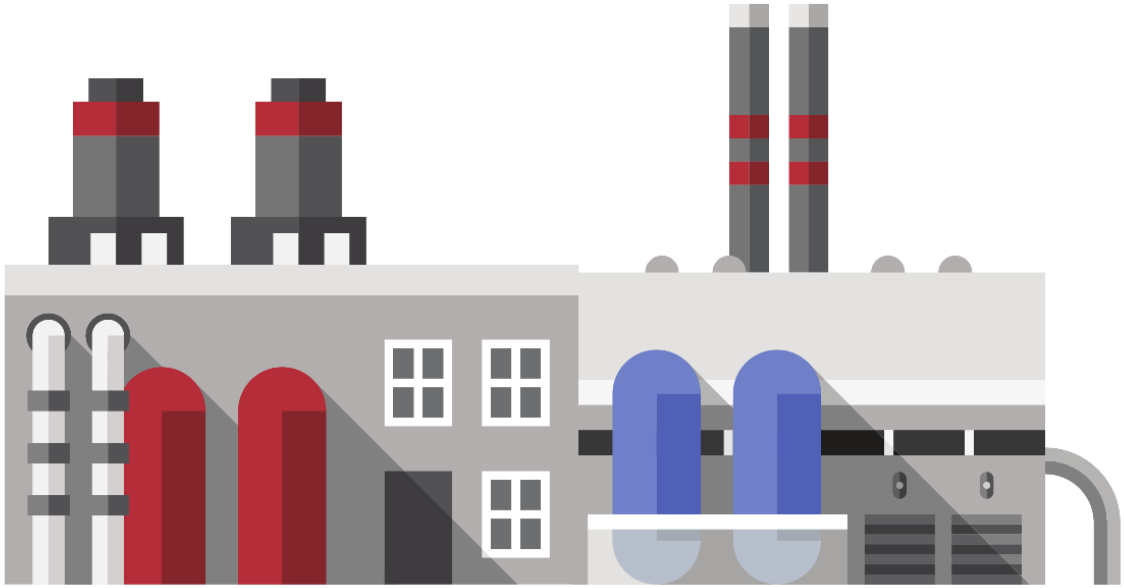
Co-funded by the Erasmus+ Programme of the European Union

Request Free Consultation

Name

Email Address

Subject



“DigiFoF: Digital Design Skills for Factories of the Future

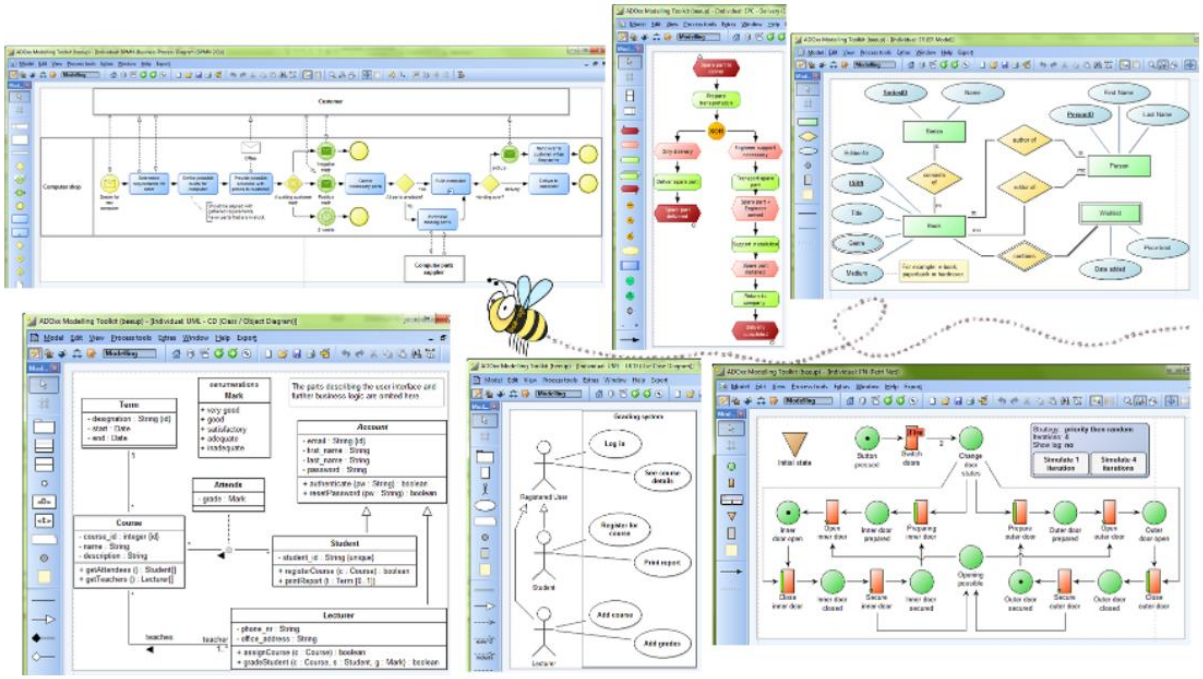
The DigiFoF project proposes a network of training environments where HEIs, enterprises, and training institutions come together to develop skill profiles, trainings concepts as well as materials for design aspects of the Factory of the Future (FoF). It aims to contribute to the transformation of the manufacturing sector, which is disrupted by digitalization. The economic potential of the digital technologies is significant: the Factory of the Future (FoF) is expected to yield a market of about USD 67 billion globally by 2020. 87% of European manufacturing enterprises estimate that digital transformation is a competitive opportunity. Yet industry needs new types of digital skills, which 90% of European enterprises indicate they lack, as 30-90 million manufacturing employees could lose their (semi)-manual jobs.

Modelling methods

- BPMN
- EPC
- ER
- UML
- Petri Nets

Bee-Up Case Study
Download pdf

Bee-Up Tool
Download tool

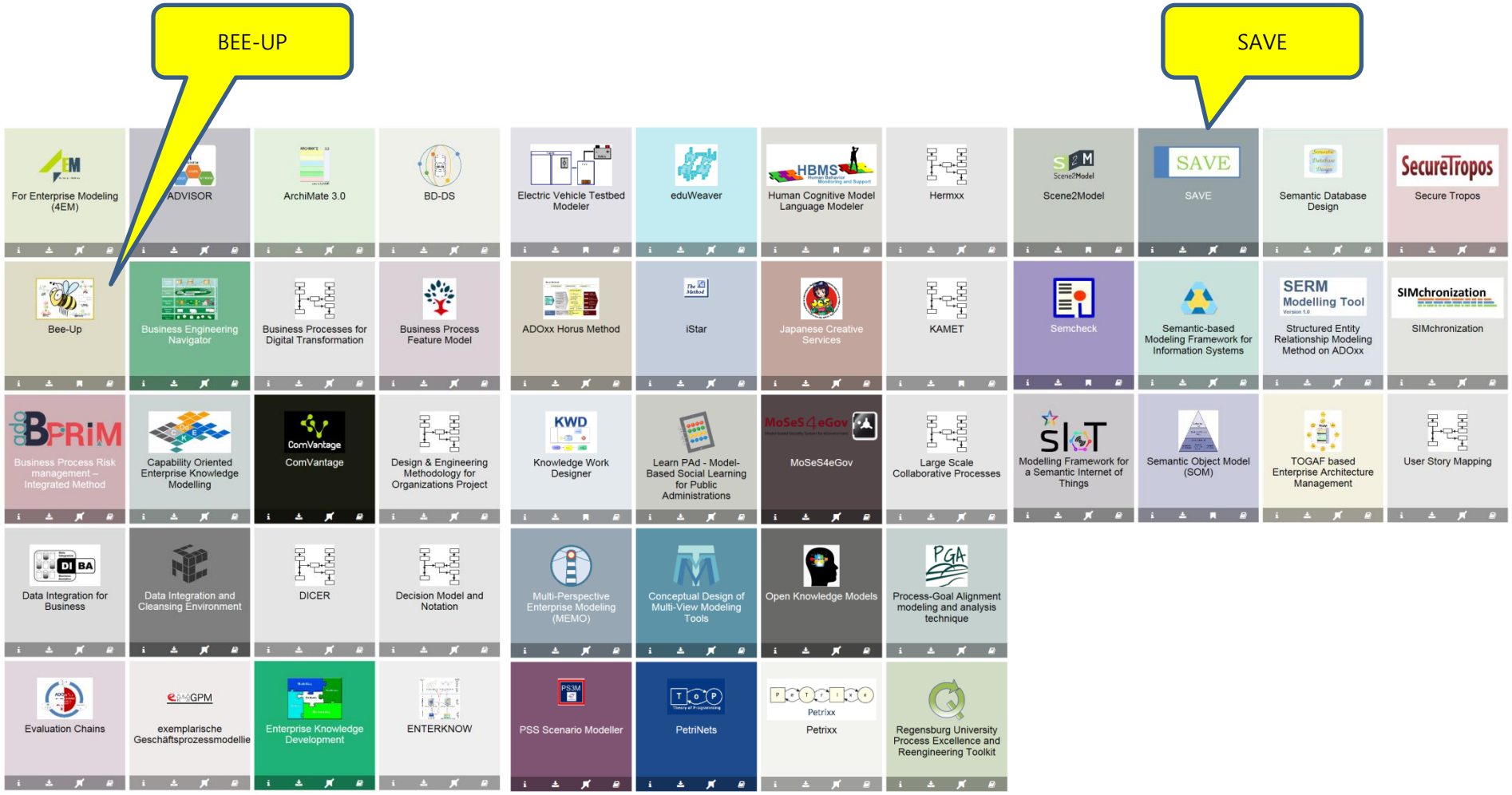


Teaching Conceptual Modelling with the Bee-up Tool

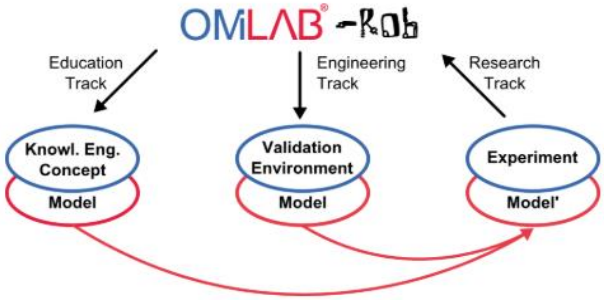
Bee-Up is an implementation of a hybrid modelling method incorporating and extending several modelling languages that gained wide popularity, namely

- the Business Process Model and Notation (BPMN)
- Event-driven Process Chains (EPC),
- Entity-Relationship models (ER),
- the Unified Modelling Language (UML) and
- Petri Nets.

52 Open Modeling Tools



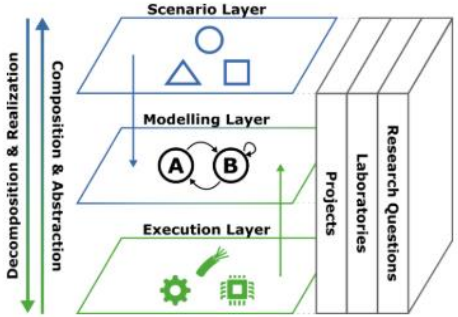
https://austria.omilab.org/psm/exploreprojects?param=explore



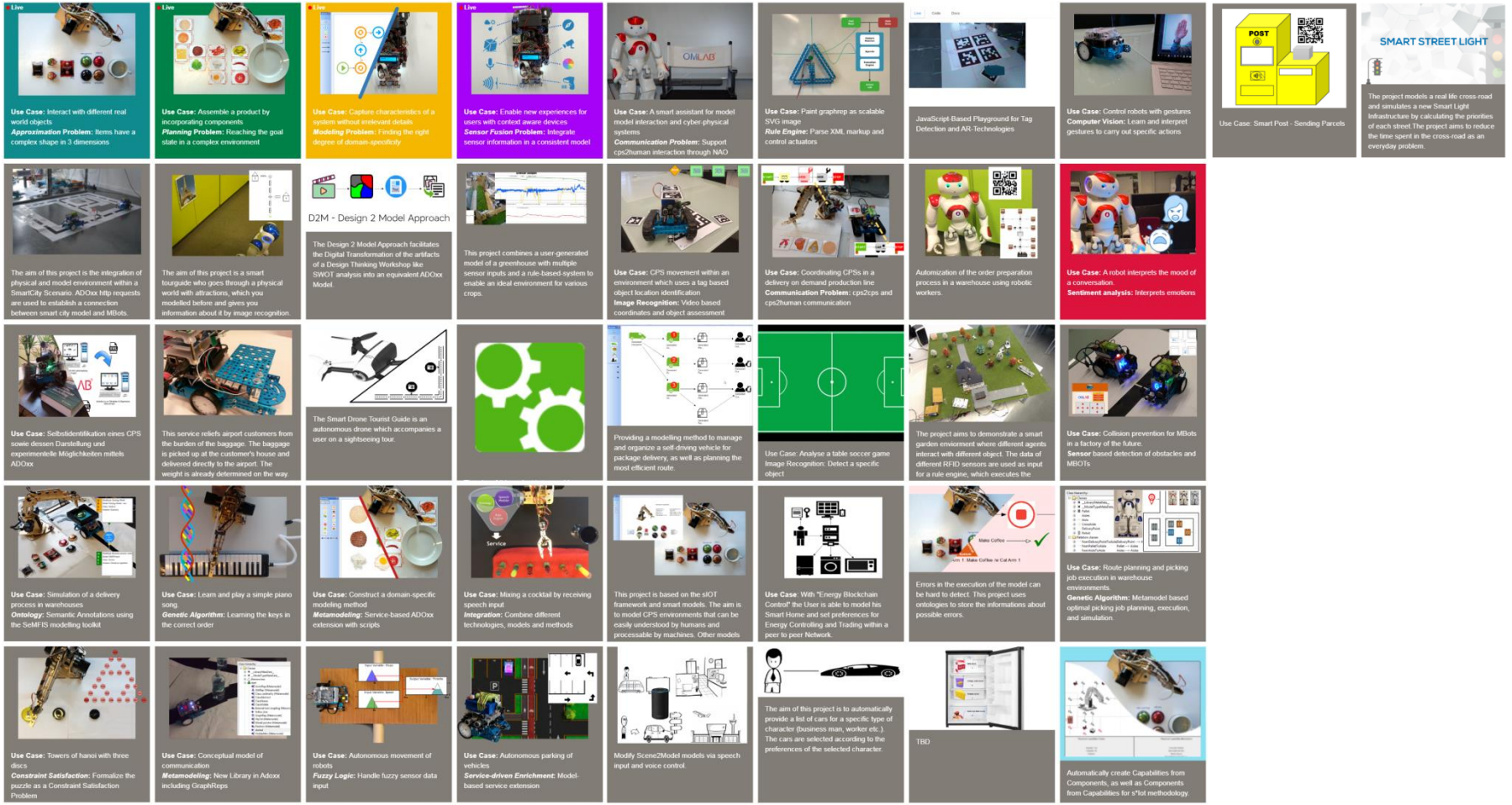
Driven by the development plan of the Univerity of Vienna, and inspired by the Open Models (OMi) Laboratory, the OMiLAB-Rob project provides a dedicated research and experimentation space to understand, model, and engineer the knowledge-intensive systems of the future. Both a physical and virtual place, OMiLab-Rob links together

- a research methodology,
- a dedicated community and
- a technical environment.

The **objective of the OMiLAB-Rob project** is to integrate conceptual models (which are constructed by humans for humans) with cyber-physical systems. Thereby, the focus is to enable a knowledge-based understanding of "smart" models by machines. To facilitate the engineering of these systems, the OMiLAB-Rob project provices a physical and virtual environment, where experiments validate the integration of conceptual models and cyber-physical systems. In a continuous cycle, the experiments' results are used to refine the necessary methods.

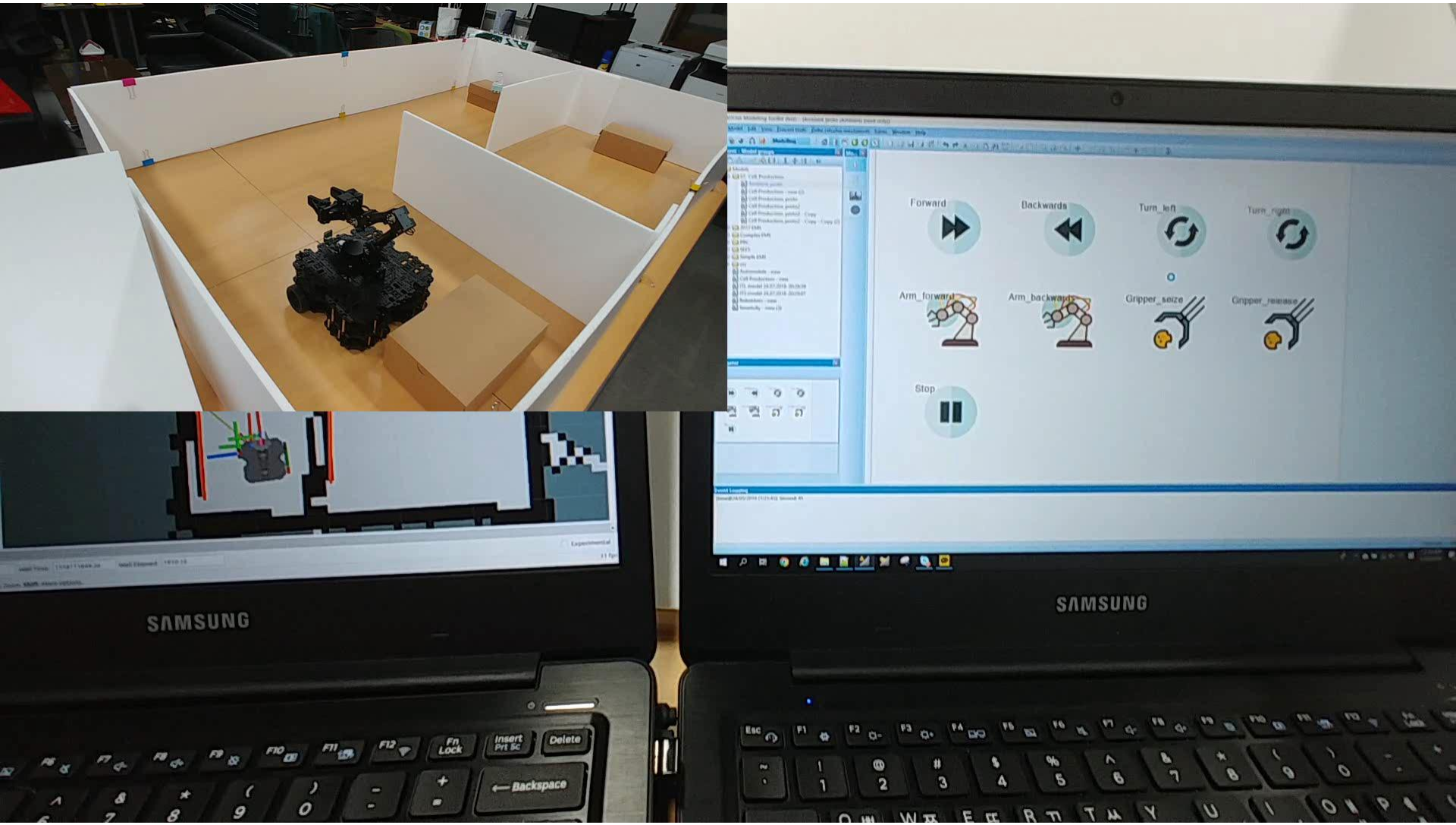


42 OMiLAB-Rob Experiments

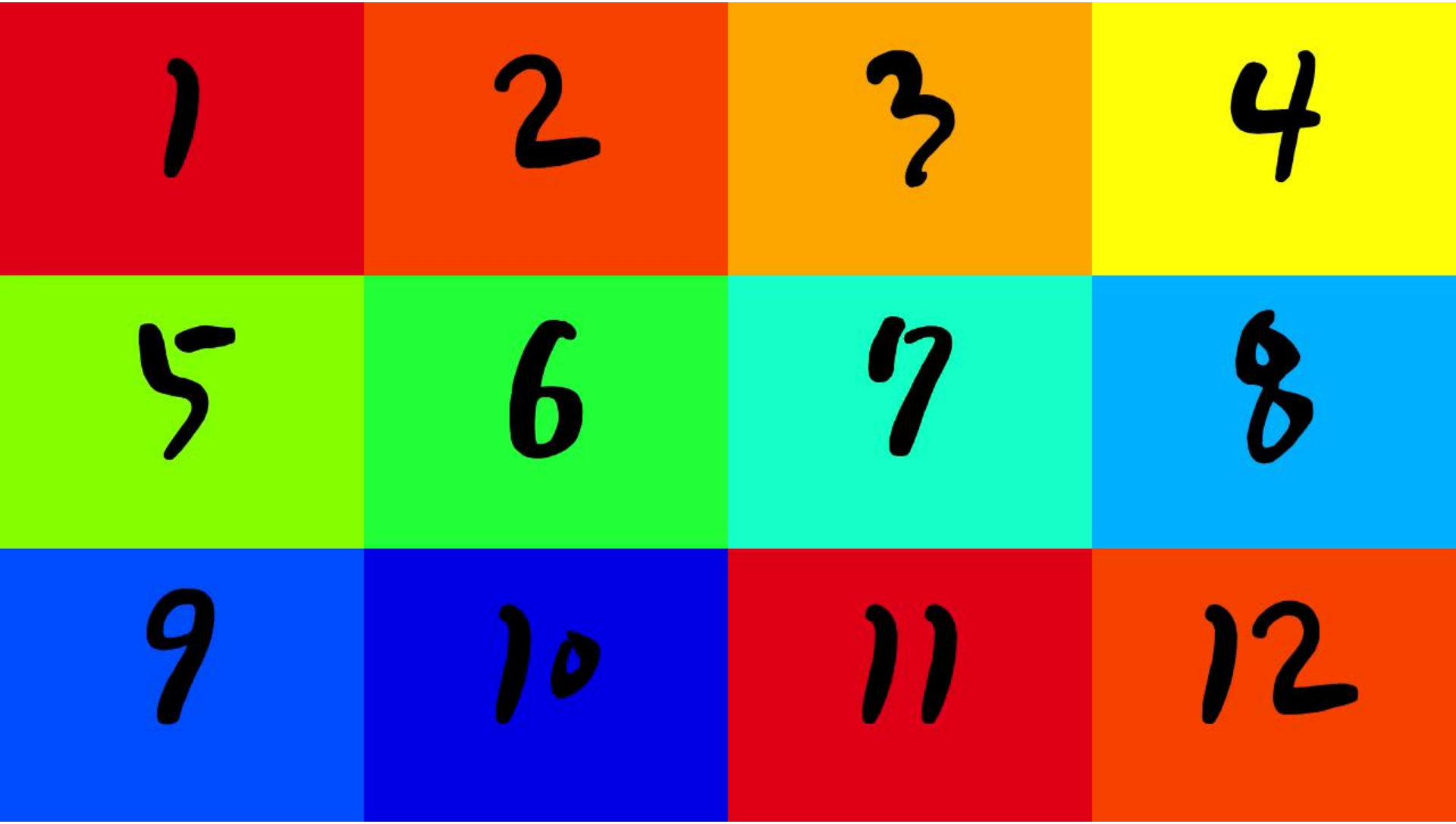


https://austria.omilab.org/psm/omirob

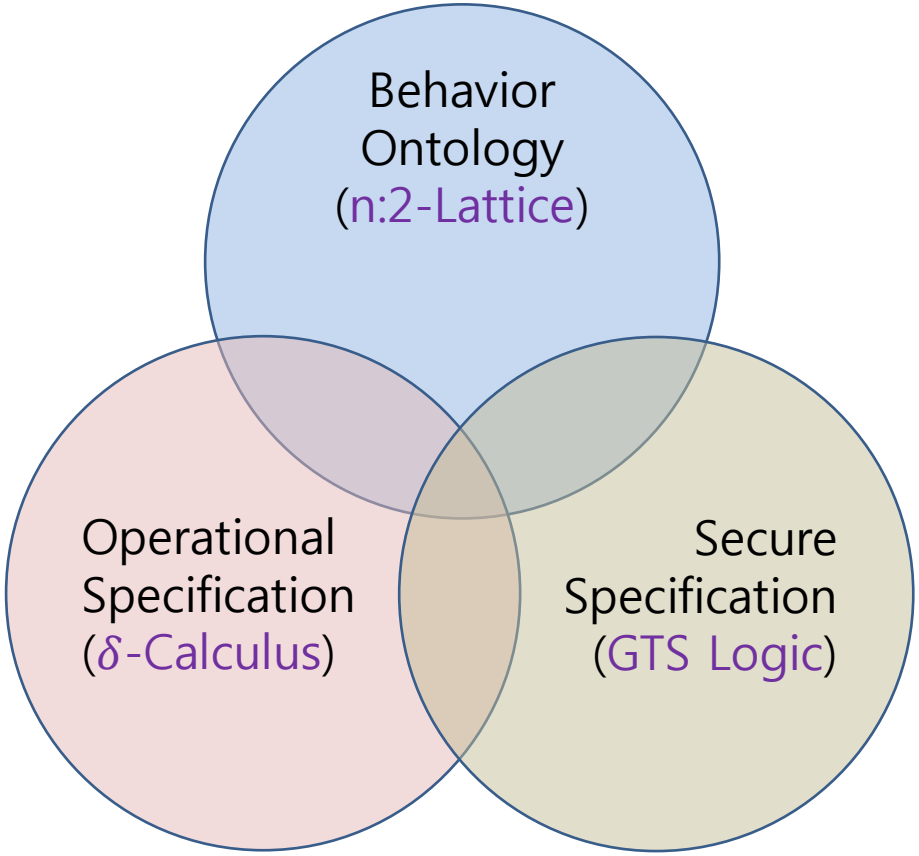
Smart Factory: NGV [Industry]



CPS: SAVE/NGV/Probability



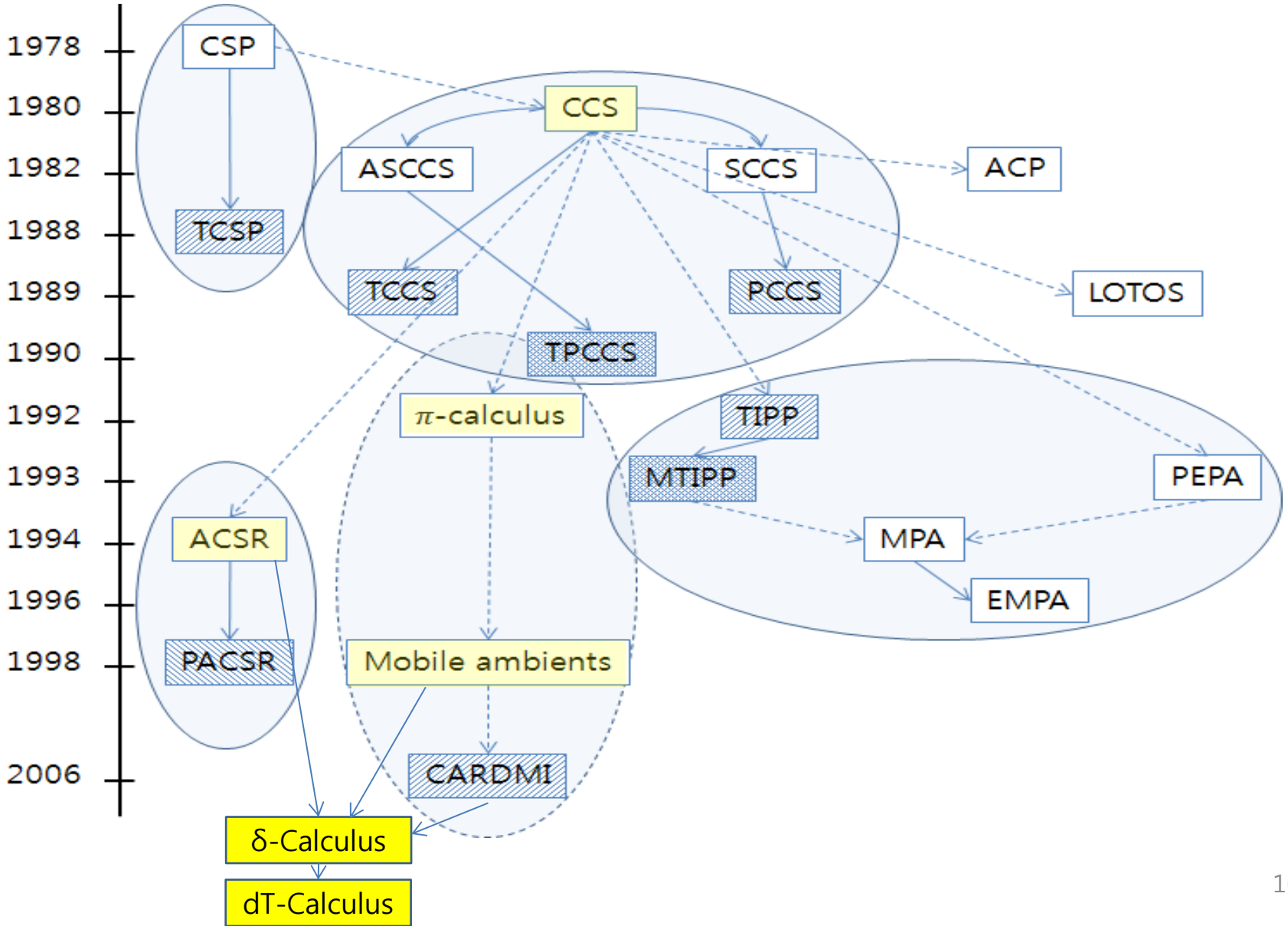
Theories



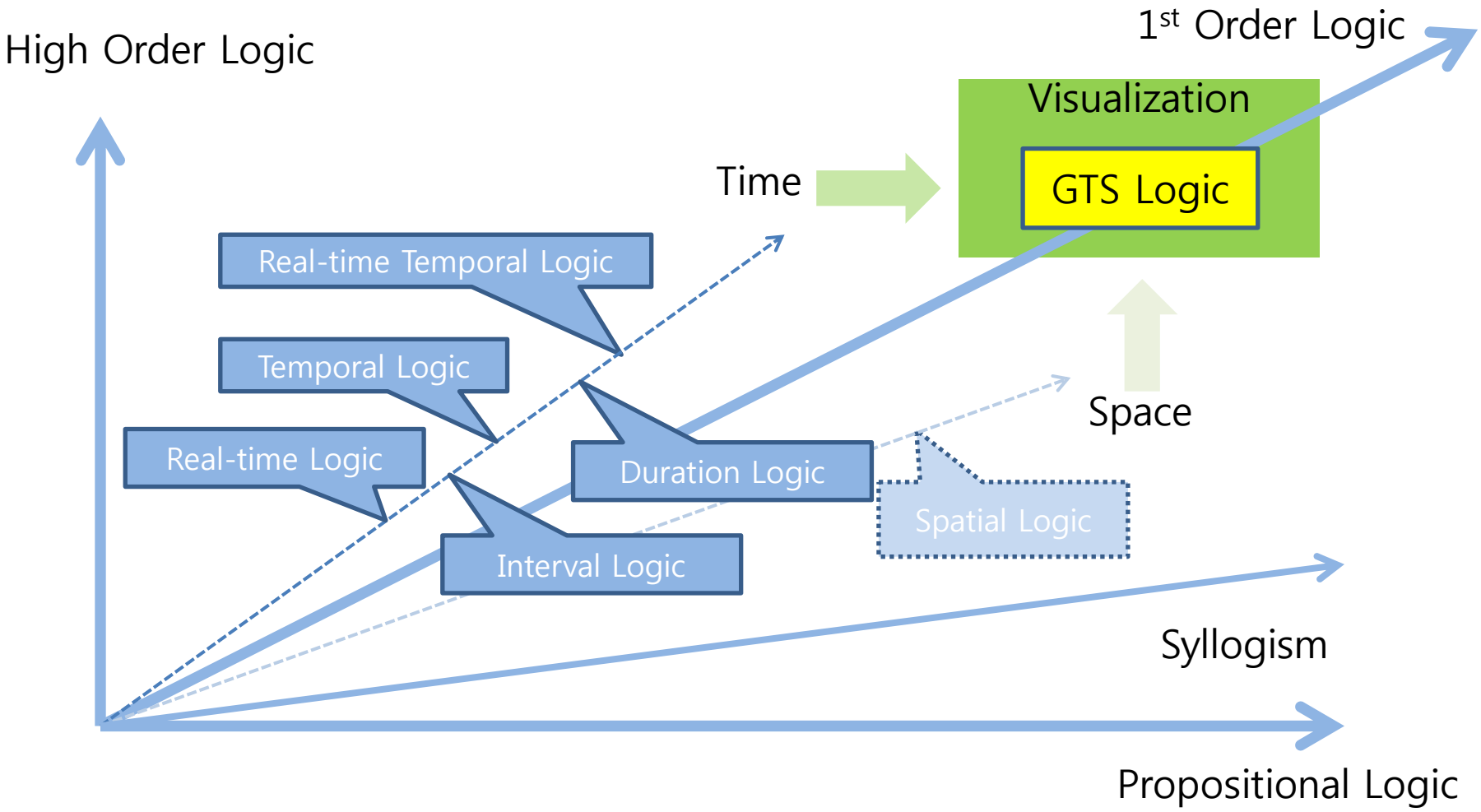
Meta-Modeling

Modeling

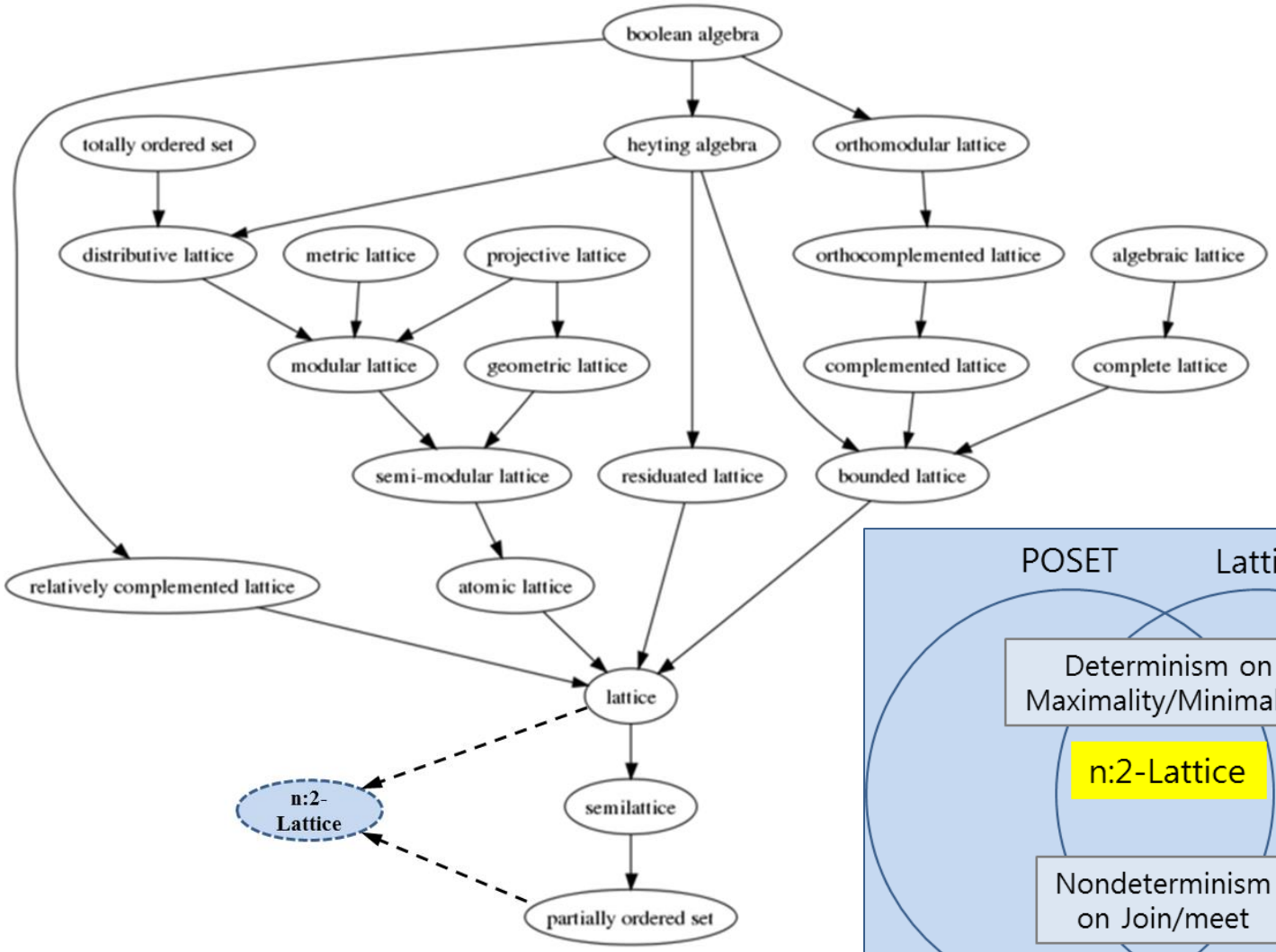
Map of Process Algebra



Dimension of Logic



Map of Lattices



Recent Publication [Since 2012]

- **Books [Korean]**
  - M. Lee, *Formal Methods*, Chonbuk National University Press, Korea, Aug. 2018.
  - M. Lee, *Process ALgebra*, Eowha Press, Korea, Aug. 2015.
  - M. Lee, *Behavior Ontology*, Eowha Press, Korea, Aug. 2013.
- **Chapters:**
  - Y. Choe, M. Lee, Algebraic Method to Model Secure IoT. , Edited D. Karagiannis, et. al., *Domain-Specific Conceptual Modeling*, Springer, 2016, Chapter.
  - S. Lee, Y. Choe, M. Lee, dT-Calculus: A Formal Method To Specify Distributed Mobile Real-Time IoT Systems. Edited . Jaydip Sen, *The Internet of Things*, ISBN 978-953-51-5945-2, inTechOpen, 2018.
- **Journals**
  - S. Lee, J. Song, Y. Choe, M. Lee, ADONIS: A Service Design and Certification Tool for Certification of Software Development Process in International Standard Organization, Journal of SOSS, 2018.
  - J. Song, M. Ramahni, M. Lee, Journal of KIISE, Modeling and Composition Method of Collective Behavior of Interactive Systems for Knowledge Engineering, Vol. 44, No. 11, pp. 1178-1193, Nov. 2017. Y. Choe, S. Lee, M. Lee, Process Algebra to Model Timed Movements of Processes in Distributed Real-Time Systems, Transactions on Information Science and Applications, Volume 14, 2017, pp. 89-101.
  - Y. Choe, S. Lee, M. Lee, dT-Calculus: A Process Algebra to Model Timed Movements of Processes The IARAS International Journal of Computers, April, 2017.
  - S. Lee, Y. Choe, M. Lee, "A Dual Method to Model IoT Systems", International Journal of Mathematical Models and Methods in Applied Sciences, 2016.05.
  - M. Lee, "A Reengineering Method from Procedural SW to Object-Oriented SW for SaaS in Cloud Computing", International Journal on Recent an Innovation Trends in Computing and Communication, 2016.03.
  - Y. Choe, M. Lee, A Process Algebra Construct Method for Reduction of States in Reachability Graph: Conjunctive and Complement Choices, Journal of KIISE, Vol. 43, No. 5, pp. 541-552, 2016. 5.
  - Y. Choe, M. Lee, A Process Algebra for Modeling Secure Movements of Distributed Mobile Processes, Journal of KIISE, Vol. 43, No. 3, pp. 314-326, 2016. 3.
  - J. On, Y. Choe, M. Lee, A Meta-Modeling for Security to Detect Attach Behavior, Journal of KIISE, Vol. 41, No. 12, Dec 2014, pp. 1035-1049.
  - J. H. Choi, J. On, S. Woo, G. Park, M. Lee, A Method for Personal Font Construction Using Template, Journal of KIISE: Practice and Reality, Vol. 29, No. 3, April 2012, pp. 336-340.
- **Conferences**
  - M. Rahmani, J. Song, M. Lee, PRISM: A Knowledge Engineering Tool to Model Collective Behavior of Real-Time IoT Systems, PoEM2017/PrOs2017, Lueven, Belgium, Nov. 2017.
  - J. Song. M. Rahmani, M. Lee, Behavior Ontology to Model Collective Behavior of Emergency Medical Systems, ER2017/AHA2017, Valencia, Spain, Nov. 2017.
  - Y, Choe, M. Lee, A Modeling Method for Smart Mobile Service with IoT Environment, ICServ2017, Vienna, Austria, July 2017.
  - Y. Choe, S. Lee, M. Lee, "dT-Calculus: A Process Algebra to Model Timed Movements of Processes", 19th International Conference on MATHEMATICAL and COMPUTATIONAL METHODS in SCIENCE and ENGINEERING, March 2017.
  - Y. Choe, S. Lee, M. Lee, SAVE: An Environment for Visual Specification and Verification of IoT, EDOC2016/ModTools2016, Vienna, Austria, Sept. 2016.
  - M. Lee, Composition Model for Cloud Services with Behavior Ontology, ICServ2016, Tokyo, Japan, Sept. 2016.
  - S. Lee, Y. Choe, M. Lee, A Dual Method to Model IoT Systems, INASE AMCME 2016, May 2016, Riga, Latvia.
  - Y. Choe, M. Lee, δ-CALCULUS: PROCESS ALGEBRA TO MODEL SECURE MOVEMENTS OF DISTRIBUTED MOBILE PROCESSES IN REAL-TIME BUSINESS APPLICATION, 23rd European Conference on Information Systems, 2015.05. Muenster, Germany.
  - Y. Choi, M. Lee, et. al., A Tool for Visual Specification and Verification for Secure process Movements, eChallenge e-2015, Vilius, Lithuania, Nov. 2016. The Best Paper.
  - W. Choi, Y. Choe, M. Lee, A Reduction Method for Process and System Complexity with Conjunctive and Complement Choices in a Process Algebra, 39th COMPSAC/MVDA, July 2015. Taiwan.
  - Y. Choe, M. Lee, A Lattice Model to Verify Behavioral Equivalences, Proceedings of UKSim-AMSS 8th European Modelling Symposium, Oct 2014.
  - J. Choi, M. Lee, A Calculus for Transportation Systems, Proceedings of 38th Annual IEEE Computer Software and Applications Conference Workshops, July 2014.
  - M. Lee, Y. Choe, Modeling and Analysis of Equivalences in Behavior Ontology, Proceedings of 4th European Conference of Computer Science, Oct 2013.
  - J. On, Y. Choe, M. Lee, An Abstraction Method of Behaviors for Process Algebra, Proceedings of IEEE 37th Annual Computer Software and Applications Conference Workshops, July 2013.
  - S. Woo, J. On, M. Lee, Behavior Ontology: A Framework to Detect Attack Patterns for Security, Proceedings of 7thInternational Symposium on Security and Multimodality in Pervasive Environment (SMPE-2013), March 2013.
  - J. On, S. Woo, M. Lee, Onion: a Visual Formal Method for Workflow Design in Cloud Computing, Proceedings of MSV12, July 2012.
  - J. On, S. Woo, M. Lee, A Graphical Language to Integrate Process Algebra and State Machine Views for Specification and Verification of Distributed Real-Time Systems, Proceedings of 36th IEEE International Conference on Computer Software and Applications Workshops, July 2012, pp. 218-223.

References (1)

[ALUR] ALUR, Rajeev; HENZINGER, Thomas A. Real-time logics: Complexity and expressiveness. *Information and Computation*, 1993, 104.1: 35-77.

[ARV08] Arvidsson, F.; Flycht-Eriksson, A. "Ontologies I" (PDF). Retrieved 26 November 2008.

[BEN] ABDALLAH, Hanene Ben. *Graphical Communicating Shared Resources: A Language for the Specification, Refinement and Analysis of Real-Time Systems*. 1996. PhD Thesis. University of Pennsylvania.

[BROS] BROSGOL, Ben; COMAR, Cyrille. DO-178C: A New Standard for Software Safety Certification. 2010.

[BERG] BERGSTRA, J. A.; KLOP, Jan Willem. Algebra of communicating processes.*Mathematics and Computer Science, CWI Monograph*, 1986, 1: 89-138.

[CES816] .M. Clarke, E.A. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. ACM Trans. On Programming Languages and Systems, 8(2):244-263, 1986.

[DILL] DILLER, Antoni. *Z: An introduction to formal methods*. John Wiley &Sons, Inc., 1990.

[DOWS] DOWSON, Mark. The Ariane 5 software failure. *ACM SIGSOFT Software Engineering Notes*, 1997, 22.2: 84.

[GHEZ] GHEZZI, Carlo; KEMMERER, Richard A. ASTRAL: an assertion language for specifying realtime systems. In: *ESEC'91*. Springer Berlin Heidelberg, 1991. p. 122-146.

[GRU93] Gruber, Thomas R. (June 1993). "A translation approach to portable ontology specifications" (PDF). Knowledge Acquisition **5** (2): 199–220. doi:10.1006/knac.1993.1008.

[HARE] HAREL, David. Statecharts: A visual formalism for complex systems. *Science of computer programming*, 1987, 8.3: 231-274.

[HOAR] HOARE, Charles Antony Richard. Communicating sequential processes. *Communications of the ACM*, 1978, 21.8: 666-677.

[ISL] LEE, Insup; DAVIDSON, Susan; GERBER, Richard. Communicating Shared Resources: A Paradigm for Integrating Real-Time Specification and Implementation. *Foundations of Real-Time Computing: Formal Specifications and Methods*, 1991, 87-109.

[ISO] LOTOS, I. S. O. A formal description technique based on the temporal ordering of observational behaviour. *ISO8807, 1XS989*, 1989.

[JAHA1] JAHANIAN, Farnam; MOK, Aloysius K.. . Modechart: A specification language for real-time systems. *Software Engineering, IEEE Transactions on*, 1994, 20.12: 933-947.

[JAHA2] JAHANIAN, Farnam; MOK, Aloysius K.. . Safety analysis of timing properties in real-time systems. *Software Engineering, IEEE Transactions on*, 1986, 9: 890-904.

[JM86] F. Jahanian and A.K. Mok. Safety analysis of timing properties in real-time systems. IEEE Trans. Software Engineering, SE-12(9):890-904, Sept. 1986.

[JONE] JONES, M., et al. Introducing ECSS Software-Engineering Standards within ESA. *ESA bulletin*, 2002, 132-139.

References (2)

[KAR] Dimitris Karagiannis, Agile Modeling Method Engineering, Proceedings of the 19th Panhellenic Conference on Informatics, Pages 5-10 , Athens, Greece — October 01 - 03, 2015.

[KEHO] KEHOE, Raymond. *ISO 9000-3.: A tool for Software Product and Process Improvement. Edition en anglais*. Springer, 1996.

[KUHN] KÜHN, H. The ADOxx® Metamodelling Platform. In: *Workshop on Methods as Plug-Ins for Meta-Modelling, Klagenfurt, Austria*. 2010.

[NASA] NASA TECHNICAL STANDARD, "SOFTWARE ASSURANCE STANDARD", National Aeronautics and Space Administration, 2004

[MILN] MILNER, Robin. *Communication and concurrency*. Prentice-Hall, Inc., 1989.

[OVER] OVERMYER, S. P. DoD-Std-2167A and methodologies. *ACM SIGSOFT Software Engineering Notes*, 1990, 15.5: 50-59.

[LEE1] Lee, M., *An Environment for Understanding of Real-time Systems*, Ph.D. Thesis, The University of Pennsylvania, 1995.

[LEE2] Lee, M., Prywes, N., Lee I., "Automation of Analysis, Simulation and Understanding of Real-Time Large Ada Software," The First IEEE International Conference on Engineering of Complex Computer Systems, IEEE Press, 1995.

[LEE3] CLARKE, Duncan; LEE, Insup; XIE, Hong-Liang. VERSA: A tool for the specification and analysis of resource-bound real-time systems. *Technical Reports (CIS)*, 1993, 510.

[PETE] PETERSON, James L. Petri nets. *ACM Computing Surveys (CSUR)*, 1977, 9.3: 223-252.

[RADA] RADATZ, Jane; OLSON, Myrna; CAMPBELL, Stuart. Mil-std-498. *Crosstalk, the Journal of Defense Software Engineering*, 1995, 8.2: 2-5.

[[SHOC] SHOCK, Sticker. Estimating the real cost of modern fighter aircraft. *Defense-aerospace, 2006.(2006-7)[2011-4]. Http://www. defense-aerospace, com*, 2006.

[SING] SINGH, Raghu. International Standard ISO/IEC 12207 software life cycle processes. *Software Process Improvement and Practice*, 1996, 2.1: 35-50.

[SPC] SOFTWARE PRODUCTIVITY CONSORTIUM SERVICES CORPORATION. *Reuse-driven Software Processes Guidebook: SPC-92019-CMC, Version 02.00. 03*. Software Productivity Consortium Services Corporation, 1993..

[SPIV] SPIVEY, J. Michael; ABRIAL, J. R. *The Z notation*. Hemel Hempstead: Prentice Hall, 1992.

[WANG] WANG, Jiacun. *Timed Petri nets: Theory and application*. Dordrecht: Kluwer Academic Publishers, 1998.