

Modelling MolNet Systems with Timed Distributed π -calculus

Cristian Prisacariu

cprisacariu@iit.tuiasi.ro

joint work with Gabriel Ciobanu

Institute of Computer Science,
Romanian Academy, Iași

14th February 2006

Dynamics and structure of biological networks,
Geometry of Computation 2006

Outline¹

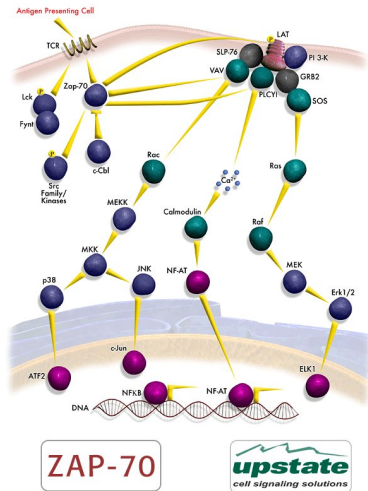
- 1 MolNet simulation tool
 - Dynamic Structure Discrete Event Systems Specification
 - The ZAP70 – LAT Interaction Example
- 2 Timed distributed π -calculus ($tD\pi$)
 - Syntax and Semantics
 - Theoretical Results
- 3 Translating MolNet into $tD\pi$
 - The ZAP70 Example Again
- 4 Comparisons with related approaches
 - Process Algebra Related
 - Modelling Approaches and Simulation Tools for Molecular Systems

¹<http://iit.iit.tuiasi.ro/~cprisacariu/geocal06.pdf>

Molecular Networks and Process Algebras

Positive aspects of a process algebra

- **behaviour properties** of the modeled system can be well understood
- **model checking** techniques help in analysis of various properties
- a **sound mathematical model** with theoretical results ensures a *strong ground* for the simulation software



Molecular Networks Simulation Tool (MolNet)^{2,3}

Molecular networks simulations require

- tremendous computing power
- component-based framework
- simulation of various stimuli and environment constraints
- simulation of explicit time

A prototype software

- based on the client/server paradigm and message transmission
- based on a mathematical formalism
- uses an improved Gillespie-like algorithm [Gibson 2000] to establish the interactions
- implemented in C with BSD-sockets, GTK 2.0 and under GPL

²For more details: <http://www.comp.nus.edu.sg/~gabriel/molnet/>

³G.Ciobanu, D.Huzum, CMSB, LNCS 2602, 2003.

Molecular Networks Simulation Tool (MolNet)^{2,3}

Molecular networks simulations require

- tremendous computing power
- component-based framework
- simulation of various stimuli and environment constraints
- simulation of explicit time

A prototype software

- based on the client/server paradigm and message transmission
- based on a mathematical formalism
- uses an improved Gillespie-like algorithm [Gibson 2000] to establish the interactions
- implemented in C with BSD-sockets, GTK 2.0 and under GPL

²For more details: <http://www.comp.nus.edu.sg/~gabriel/molnet/>

³G.Ciobanu, D.Huzum, CMSB, LNCS 2602, 2003.

Dynamic Structure Discrete Event Systems (DSDEVS)

The mathematical model

The specification of discrete event systems with dynamic structure⁴:

- is based on
 - ▶ components which form a system model
 - ▶ communication between the components is made through messages
- is defined by
 - ▶ basic models (DEVS models)
 - ▶ network models (composed of basic models)
 - ▶ the *executive* which is a modified basic model which keeps the information about the network composition and coupling and defines changes in the network.

All the molecules of the same type are considered as a basic DEVS.

⁴[Barros 1997]

Dynamic Structure Discrete Event Systems (DSDEVS)

The mathematical model

The specification of discrete event systems with dynamic structure⁴:

- is based on
 - ▶ components which form a system model
 - ▶ communication between the components is made through messages
- is defined by
 - ▶ basic models (DEVS models)
 - ▶ network models (composed of basic models)
 - ▶ the *executive* which is a modified basic model which keeps the information about the network composition and coupling and defines changes in the network.

All the molecules of the same type are considered as a basic DEVS.

⁴[Barros 1997]

Dynamic Structure Discrete Event Systems (DSDEVS)

The mathematical model

The specification of discrete event systems with dynamic structure⁴:

- is based on
 - ▶ components which form a system model
 - ▶ communication between the components is made through messages
- is defined by
 - ▶ basic models (DEVS models)
 - ▶ network models (composed of basic models)
 - ▶ the *executive* which is a modified basic model which keeps the information about the network composition and coupling and defines changes in the network.

All the **molecules** of the same type are considered as a basic DEVS.

⁴[Barros 1997]

DEVS Specification

For a molecule type M_i the associated DEVS consists of:

- $X_i = \{(p, v) \mid p \in IPorts, v \in X_{i,p}\}$; a set of pairs of *input ports*, *input values*. $X_{i,p}$ is the set of possible input values for port p ;
- $Y_i = \{(p, v) \mid p \in OPorts, v \in Y_{i,p}\}$; a set of pairs of *output ports*, *output values*. $Y_{i,p}$ is the set of possible output values for port p ;
- S_i is the set of states;
- $\delta_i^{int} : S_i \rightarrow S_i$ is the *internal transition function*;
- $\delta_i^{ext} : (Q_i \times X_i) \rightarrow S_i$ is the *external transition function* and $Q_i = \{(s_i, e) \mid s_i \in S_i, 0 \leq e \leq \tau_i(s_i)\}$; e represents the elapsed time since the last transition.
- $\lambda_i : S_i \rightarrow Y_i$ is the *output function*;
- $\tau_i : S_i \rightarrow \mathbb{R}_+$ is the *time advance function*.

DEVS Specification

For a molecule type M_i the associated DEVS consists of:

- $X_i = \{(p, v) \mid p \in IPorts, v \in X_{i,p}\}$; a set of pairs of *input ports*, *input values*. $X_{i,p}$ is the set of possible input values for port p ;
- $Y_i = \{(p, v) \mid p \in OPorts, v \in Y_{i,p}\}$; a set of pairs of *output ports*, *output values*. $Y_{i,p}$ is the set of possible output values for port p ;
- S_i is the set of states;
- $\delta_i^{int} : S_i \rightarrow S_i$ is the *internal transition function*;
- $\delta_i^{ext} : (Q_i \times X_i) \rightarrow S_i$ is the *external transition function* and $Q_i = \{(s_i, e) \mid s_i \in S_i, 0 \leq e \leq \tau_i(s_i)\}$; e represents the elapsed time since the last transition.
- $\lambda_i : S_i \rightarrow Y_i$ is the *output function*;
- $\tau_i : S_i \rightarrow \mathbb{R}_+$ is the *time advance function*.

DEVS Specification

For a molecule type M_i the associated DEVS consists of:

- $X_i = \{(p, v) \mid p \in IPorts, v \in X_{i,p}\}$; a set of pairs of *input ports*, *input values*. $X_{i,p}$ is the set of possible input values for port p ;
- $Y_i = \{(p, v) \mid p \in OPorts, v \in Y_{i,p}\}$; a set of pairs of *output ports*, *output values*. $Y_{i,p}$ is the set of possible output values for port p ;
- S_i is the set of states;
- $\delta_i^{int} : S_i \rightarrow S_i$ is the *internal transition function*;
- $\delta_i^{ext} : (Q_i \times X_i) \rightarrow S_i$ is the *external transition function* and $Q_i = \{(s_i, e) \mid s_i \in S_i, 0 \leq e \leq \tau_i(s_i)\}$; e represents the elapsed time since the last transition.
- $\lambda_i : S_i \rightarrow Y_i$ is the *output function*;
- $\tau_i : S_i \rightarrow \mathbb{R}_+$ is the *time advance function*.

DEVS Specification

For a molecule type M_i the associated DEVS consists of:

- $X_i = \{(p, v) \mid p \in IPorts, v \in X_{i,p}\}$; a set of pairs of *input ports*, *input values*. $X_{i,p}$ is the set of possible input values for port p ;
- $Y_i = \{(p, v) \mid p \in OPorts, v \in Y_{i,p}\}$; a set of pairs of *output ports*, *output values*. $Y_{i,p}$ is the set of possible output values for port p ;
- S_i is the set of states;
- $\delta_i^{int} : S_i \rightarrow S_i$ is the *internal transition function*;
- $\delta_i^{ext} : (Q_i \times X_i) \rightarrow S_i$ is the *external transition function* and $Q_i = \{(s_i, e) \mid s_i \in S_i, 0 \leq e \leq \tau_i(s_i)\}$; e represents the elapsed time since the last transition.
- $\lambda_i : S_i \rightarrow Y_i$ is the *output function*;
- $\tau_i : S_i \rightarrow \mathbb{R}_+$ is the *time advance function*.

DSDEVN Specification

The network model

The structure of a network based on a set of molecules D is given by a tuple $(D, \{M_i\}_{i \in D}, \{I_i\}_{i \in D}, \{Z_i\}_{i \in D})$ where:

- M_i is a DEVS;
- I_i is the set of possible interaction partners for molecule type i ;
- Z_i is the input function $Z_i : \times_{j \in I_i} Y_j \rightarrow X_i$.

The model of the *executive* χ

- $M_\chi = (X_\chi, S_\chi, Y_\chi, \delta_\chi^{int}, \delta_\chi^{ext}, \lambda_\chi, \tau_\chi)$
- the structure of the network is kept in the state $s_\chi \in S_\chi$ and is a tuple $s = (X_s, Y_s, D_s, \{M_i\}_{i \in D_s}, \{I_i\}_{i \in D_s}, \{Z_i\}_{i \in D_s}, \Xi)$

DSDEVN Specification

The network model

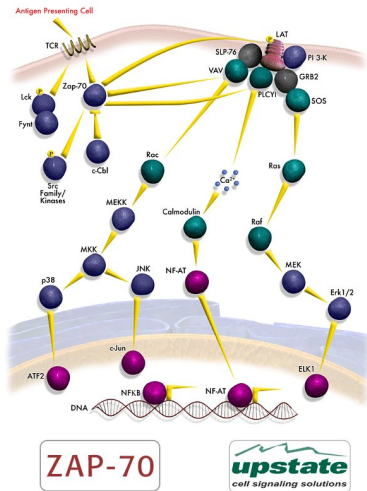
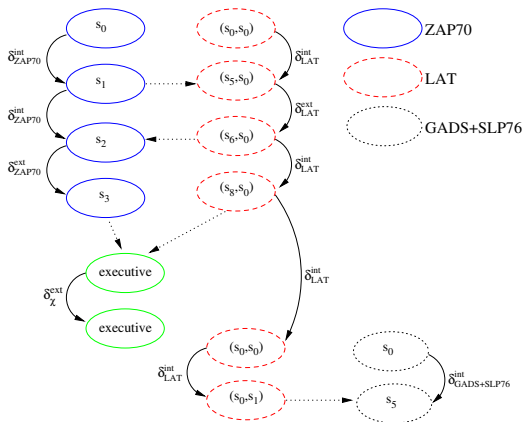
The structure of a network based on a set of molecules D is given by a tuple $(D, \{M_i\}_{i \in D}, \{I_i\}_{i \in D}, \{Z_i\}_{i \in D})$ where:

- M_i is a DEVS;
- I_i is the set of possible interaction partners for molecule type i ;
- Z_i is the input function $Z_i : \times_{j \in I_i} Y_j \rightarrow X_i$.

The model of the *executive* χ

- $M_\chi = (X_\chi, S_\chi, Y_\chi, \delta_\chi^{int}, \delta_\chi^{ext}, \lambda_\chi, \tau_\chi)$
- the structure of the network is kept in the state $s_\chi \in S_\chi$ and is a tuple $s = (X_s, Y_s, D_s, \{M_i\}_{i \in D_s}, \{I_i\}_{i \in D_s}, \{Z_i\}_{i \in D_s}, \Xi)$

ZAP70 and LAT Signalling Pathway



Timed Distributed π -calculus ($tD\pi$)

What for?

Timers, Types and Locations to tackle the simulation problem:

- to have the needed computing power we model a **distributed system**
- to model components and the communications among them we choose a **process algebra**
- to restrict the communications and the access to resources (permissions) we give a **typing system**
- to model timeouts for communications and model temporary permissions we use **timers**

$tD\pi$ features

- designed using locations, local communication and code migration to model distributed systems
- *explicit time* (given through timers)
- main action is the *communication* between processes

Timed Distributed π -calculus ($tD\pi$)

What for?

Timers, Types and Locations to tackle the simulation problem:

- to have the needed computing power we model a distributed system
- to model components and the communications among them we choose a process algebra
- to restrict the communications and the access to resources (permissions) we give a typing system
- to model timeouts for communications and model temporary permissions we use timers

$tD\pi$ features

- designed using **locations**, **local communication** and **code migration** to model distributed systems
- *explicit time* (given through **timers**)
- **main action** is the *communication* between processes

$tD\pi$ Syntax⁵

Process expression

- Timed channel names
- A located process
- An input process with restricted values

$u^{\Delta t}?$ and $u^{\Delta t}'!$;
 $k[[P]]$;
 $u^{\Delta t}?(X:T).(P, Q).$

Type system

- A channel type with timer
- Channel capabilities
- Location capabilities for location types

$\text{res}\{\tilde{\alpha}\}\Delta t$;
 $r\langle T \rangle \mid w\langle T' \rangle$;
 $u : A \mid \text{go} \mid \text{new } c.$

Subtyping relation $<$:

Location types are sets of channel types. For location types the subtyping relation is the inverse of the subset relation for sets; i.e., $A < B \Leftrightarrow A \supset B$.

⁵Details in technical report [Prisacariu & Ciobanu 2005]

$tD\pi$ Syntax⁵

Process expression

- Timed channel names $u^{\Delta t?}$ and $u^{\Delta t'!}$;
- A located process $k[[P]]$;
- An input process with restricted values $u^{\Delta t?}(X:T).(P, Q)$.

Type system

- A channel type with timer $\text{res}\{\tilde{\alpha}\}\Delta t$;
- Channel capabilities $r\langle T \rangle \mid w\langle T' \rangle$;
- Location capabilities for location types $u : A \mid go \mid new c$.

Subtyping relation $<$:

Location types are sets of channel types. For location types the subtyping relation is the inverse of the subset relation for sets; i.e., $A < B \Leftrightarrow A \supset B$.

⁵Details in technical report [Prisacariu & Ciobanu 2005]

$tD\pi$ Syntax⁵

Process expression

- Timed channel names $u^{\Delta t?}$ and $u^{\Delta t'!}$;
- A located process $k[[P]]$;
- An input process with restricted values $u^{\Delta t?}(X:T).(P, Q)$.

Type system

- A channel type with timer $\text{res}\{\tilde{\alpha}\}\Delta t$;
- Channel capabilities $r\langle T \rangle \mid w\langle T' \rangle$;
- Location capabilities for location types $u : A \mid \text{go} \mid \text{new } c$.

Subtyping relation $<$:

Location types are sets of channel types. For location types the subtyping relation is the inverse of the subset relation for sets; i.e., $A < B \Leftrightarrow A \supset B$.

⁵Details in technical report [Prisacariu & Ciobanu 2005]

$tD\pi$ Semantics

Time-stepping function $\phi_\Delta : \mathcal{P}_\Delta \rightarrow \mathcal{P}_\Delta$

$$\phi_\Delta(k[[P]]_\Gamma) = k[[u^{\Delta(t-1)}.(Q, R)]]_{\Gamma'}, \text{ if } P = u^{\Delta t}.(Q, R), t > 1 \text{ and } t \neq \infty$$

where Γ is the type environment (a set of location types).

Γ' is obtained from Γ by decreasing the timers on channel types and removing types with expired timers.

Reduction rules

$$(R_\Gamma\text{-IDLE}) \frac{k[[P]]_\Gamma \not\rightarrow}{k[[P]]_\Gamma \rightarrow \phi_\Delta(k[[P]]_\Gamma)}$$

$tD\pi$ Semantics

Time-stepping function $\phi_\Delta : \mathcal{P}_\Delta \rightarrow \mathcal{P}_\Delta$

$$\phi_\Delta(k[[P]]_\Gamma) = k[[u^{\Delta(t-1)}.(Q, R)]]_{\Gamma'}, \text{ if } P = u^{\Delta t}.(Q, R), t > 1 \text{ and } t \neq \infty$$

where Γ is the type environment (a set of location types).

Γ' is obtained from Γ by decreasing the timers on channel types and removing types with expired timers.

Reduction rules

$$(R_\Gamma\text{-IDLE}) \frac{k[[P]]_\Gamma \not\rightarrow}{k[[P]]_\Gamma \rightarrow \phi_\Delta(k[[P]]_\Gamma)}$$

Theoretical Results

Soundness

Soundness of the typing system with respect to reduction relation \rightarrow and syntactic equivalence \equiv .

Theorem (Subject reduction)

For all tagged processes

- (a) *If $N \equiv N'$ then $\Gamma \Vdash N$ if and only if $\Gamma \Vdash N'$.*
- (b) *If $N \rightarrow N'$ then $\Gamma \Vdash N$ if and only if $\Gamma \Vdash N'$.*

Theorem (Type safety)

For all tagged located processes N and all type environments Γ such that $\Gamma \Vdash N$ we have $N \not\overset{err}{\rightarrow}$.

Theoretical Results

Soundness

Soundness of the typing system with respect to reduction relation \rightarrow and syntactic equivalence \equiv .

Theorem (Subject reduction)

For all tagged processes

- (a) *If $N \equiv N'$ then $\Gamma \Vdash N$ if and only if $\Gamma \Vdash N'$.*
- (b) *If $N \rightarrow N'$ then $\Gamma \Vdash N$ if and only if $\Gamma \Vdash N'$.*

Theorem (Type safety)

For all tagged located processes N and all type environments Γ such that $\Gamma \Vdash N$ we have $N \not\stackrel{err}{\rightarrow}$.

Theoretical Results

Timed barbed bisimulation and other time related results

Definition (timed barb)

A timed barb predicate $\downarrow_{u@k}^t$ is defined inductively by the rules:

$$\frac{}{u^{\Delta t}!\langle v \rangle.(P, Q) \downarrow_{u@k}^t} \quad \frac{}{u^{\Delta t}?(X : T).(P, Q) \downarrow_{u@k}^t}$$

We define a weak timed barb $\Downarrow_{u@k}^t$ as $P \Downarrow_{u@k}^t$ if and only if there is a sequence of transitions from P to Q denoted by $P \twoheadrightarrow Q$, and $Q \downarrow_{u@k}^t$.

Lemma (The passing of time does not interfere with the type system)

If $\Gamma \Vdash k[[P]]_{\Gamma}$ then $\Gamma \Vdash \phi_{\Delta}(k[[P]]_{\Gamma})$.

Theorem ($D\pi \leftrightarrow tD\pi$)

The new calculus $tD\pi$ can express any process described in $D\pi$.

Theoretical Results

Timed barbed bisimulation and other time related results

Definition (timed barb)

A timed barb predicate $\downarrow_{u@k}^t$ is defined inductively by the rules:

$$\frac{}{u^{\Delta t}!\langle v \rangle.(P, Q) \downarrow_{u@k}^t} \quad \frac{}{u^{\Delta t}?(X : T).(P, Q) \downarrow_{u@k}^t}$$

We define a weak timed barb $\Downarrow_{u@k}^t$ as $P \Downarrow_{u@k}^t$ if and only if there is a sequence of transitions from P to Q denoted by $P \twoheadrightarrow Q$, and $Q \downarrow_{u@k}^t$.

Lemma (The passing of time does not interfere with the type system)

If $\Gamma \Vdash k[[P]]_{\Gamma}$ then $\Gamma \Vdash \phi_{\Delta}(k[[P]]_{\Gamma})$.

Theorem ($D\pi \leftrightarrow tD\pi$)

The new calculus $tD\pi$ can express any process described in $D\pi$.

Theoretical Results

Timed barbed bisimulation and other time related results

Definition (timed barb)

A timed barb predicate $\downarrow_{u@k}^t$ is defined inductively by the rules:

$$\frac{}{u^{\Delta t}!\langle v \rangle.(P, Q) \downarrow_{u@k}^t} \quad \frac{}{u^{\Delta t}?(X : T).(P, Q) \downarrow_{u@k}^t}$$

We define a weak timed barb $\Downarrow_{u@k}^t$ as $P \Downarrow_{u@k}^t$ if and only if there is a sequence of transitions from P to Q denoted by $P \twoheadrightarrow Q$, and $Q \downarrow_{u@k}^t$.

Lemma (The passing of time does not interfere with the type system)

If $\Gamma \Vdash k[[P]]_{\Gamma}$ then $\Gamma \Vdash \phi_{\Delta}(k[[P]]_{\Gamma})$.

Theorem ($D\pi \leftrightarrow tD\pi$)

The new calculus $tD\pi$ can express any process described in $D\pi$.

Translating DEVS into $tD\pi$ (I)

DEVS:

- the set D of molecules
- a molecule M_A of type A
- S_A the set of states
$$S_A = \{s_1, s_2, \dots, s_n\}$$
- input and output ports
- the set $X_{A,p}$ of input values for an input port p

$tD\pi$:

- the set of locations
- a process at a location k_A
$$k_A[[P]]_{\Gamma_A}$$
- process expressions
$$P_1 = u^{\Delta t}?(X : T).(P_i, P_j)$$
- input and output communication channels
- a type $p : res\{r\langle T \rangle\}$ to restrict the read permissions on channel p to messages conforming with type T

Translating DEVS into $tD\pi$ (I)

DEVS:

- the set D of molecules
- a molecule M_A of type A
- S_A the set of states
$$S_A = \{s_1, s_2, \dots, s_n\}$$
- input and output ports
- the set $X_{A,p}$ of input values for an input port p

$tD\pi$:

- the set of locations
- a process at a location k_A
$$k_A[[P]]_{\Gamma_A}$$
- process expressions
$$P_1 = u^{\Delta t}?(X : T).(P_i, P_j)$$
- input and output communication channels
- a type $p : res\{r\langle T \rangle\}$ to restrict the read permissions on channel p to messages conforming with type T

Translating DEVS into $tD\pi$ (I)

DEVS:

- the set D of molecules
- a molecule M_A of type A
- S_A the set of states
$$S_A = \{s_1, s_2, \dots, s_n\}$$
- input and output ports
- the set $X_{A,p}$ of input values for an input port p

$tD\pi$:

- the set of locations
- a process at a location k_A
$$k_A[[P]]_{\Gamma_A}$$
- process expressions
$$P_1 = u^{\Delta t}?(X : T).(P_i, P_j)$$
- input and output communication channels
- a type $p : res\{r\langle T \rangle\}$ to restrict the read permissions on channel p to messages conforming with type T

Translating DEVS into $tD\pi$ (I)

DEVS:

- the set D of molecules
- a molecule M_A of type A
- S_A the set of states
$$S_A = \{s_1, s_2, \dots, s_n\}$$
- **input and output ports**
- the set $X_{A,p}$ of input values for an input port p

$tD\pi$:

- the set of locations
- a process at a location k_A
$$k_A[[P]]_{\Gamma_A}$$
- process expressions
$$P_1 = u^{\Delta t}?(X : T).(P_i, P_j)$$
- **input and output communication channels**
- a type $p : res\{r\langle T \rangle\}$ to restrict the read permissions on channel p to messages conforming with type T

Translating DEVS into $tD\pi$ (I)

DEVS:

- the set D of molecules
- a molecule M_A of type A
- S_A the set of states
$$S_A = \{s_1, s_2, \dots, s_n\}$$
- input and output ports
- the set $X_{A,p}$ of input values for an input port p

$tD\pi$:

- the set of locations
- a process at a location k_A
$$k_A[[P]]_{\Gamma_A}$$
- process expressions
$$P_1 = u^{\Delta t}?(X : T).(P_i, P_j)$$
- input and output communication channels
- a type $p : res\{r\langle T \rangle\}$ to restrict the read permissions on channel p to messages conforming with type T

Translating DEVS into $tD\pi$ (II)

DEVS:

- τ_A time advance function
for $s_i \in S_A$ we have $\tau_A(s_i) \in \mathbb{R}_+$

- δ_A^{int} internal transition function

$$\delta_A^{int}(s_i) = s_{j_2}$$

- δ_A^{ext} external transition function

$$\delta_A^{ext}(s_i, e, v_p) \text{ with } e < \tau(s_i) \text{ and } v_p \in X_{i,p}$$

$tD\pi$:

- the timers

for $P_i = u^{\Delta t?}(X : T).(P_{j_1}, P_{j_2})$
with channel type $u : res\{T\}\Delta t'$

- time-stepping function ϕ_Δ

$$\phi_\Delta(k_A[[P_i]]) = k_A[[P_{j_2}]] \\ \text{if } t < 1 \text{ or if } t' < 1$$

- transition when receiving a message on an input channel

$$k_A[[P_i]] \xrightarrow{COMM} k_A[[P_{j_1}\{v/X\}]] \\ \text{if } t > 1 \text{ and } t' > 1$$

Translating DEVS into $tD\pi$ (II)

DEVS:

- τ_A time advance function
for $s_i \in S_A$ we have $\tau_A(s_i) \in \mathbb{R}_+$

- δ_A^{int} internal transition function

$$\delta_A^{int}(s_i) = s_{j_2}$$

- δ_A^{ext} external transition function

$$\delta_A^{ext}(s_i, e, v_p) \text{ with } e < \tau(s_i) \text{ and } v_p \in X_{i,p}$$

$tD\pi$:

- the timers

for $P_i = u^{\Delta t?}(X : T).(P_{j_1}, P_{j_2})$
with channel type $u : res\{T\}\Delta t'$

- time-stepping function ϕ_Δ

$$\phi_\Delta(k_A[[P_i]]) = k_A[[P_{j_2}]] \\ \text{if } t < 1 \text{ or if } t' < 1$$

- transition when receiving a message on an input channel

$$k_A[[P_i]] \xrightarrow{COMM} k_A[[P_{j_1}\{v/X\}]] \\ \text{if } t > 1 \text{ and } t' > 1$$

Translating DEVS into $tD\pi$ (II)

DEVS:

- τ_A time advance function
for $s_i \in S_A$ we have $\tau_A(s_i) \in \mathbb{R}_+$
- δ_A^{int} internal transition function
$$\delta_A^{int}(s_i) = s_{j_2}$$
- δ_A^{ext} external transition function
 $\delta_A^{ext}(s_i, e, v_p)$ with
 $e < \tau(s_i)$ and $v_p \in X_{i,p}$

$tD\pi$:

- the timers
for $P_i = u^{\Delta t?}(X : T).(P_{j_1}, P_{j_2})$
with channel type $u : res\{T\}\Delta t'$
- time-stepping function ϕ_Δ
$$\phi_\Delta(k_A[[P_i]]) = k_A[[P_{j_2}]]$$

if $t < 1$ or if $t' < 1$
- transition when receiving a message on an input channel
$$k_A[[P_i]] \xrightarrow{COMM} k_A[[P_{j_1}\{v/X\}]]$$

if $t > 1$ and $t' > 1$

Modelling a DSDEVN

- We model DEVS-like molecules as **tagged located processes** $k[[P]]_{\Gamma}$
- Each molecule M_A of type A carries the information about the set I_A and models the input function Z_A
- The **type environment** Γ of each tagged located process represents the set of interaction partners I_A of molecule M_A in each state
- The input function is a syntactic construct in $tD\pi$ which gets inputs from all the process at the locations available in the type environment and gives an output value to the input channel of the molecule.
- The **executive** is a coordinator able to manage
 - ▶ time coordination given through timer assignments
 - ▶ coordination through messages of communications among processes
 - ▶ resource access constraints given by types

Modelling a DSDEVN

- We model DEVS-like molecules as **tagged located processes** $k[[P]]_{\Gamma}$
- Each molecule M_A of type A carries the information about the set I_A and models the input function Z_A
- The **type environment** Γ of each tagged located process represents the set of interaction partners I_A of molecule M_A in each state
- The input function is a syntactic construct in $tD\pi$ which gets inputs from all the process at the locations available in the type environment and gives an output value to the input channel of the molecule.
- The **executive** is a coordinator able to manage
 - ▶ time coordination given through timer assignments
 - ▶ coordination through messages of communications among processes
 - ▶ resource access constraints given by types

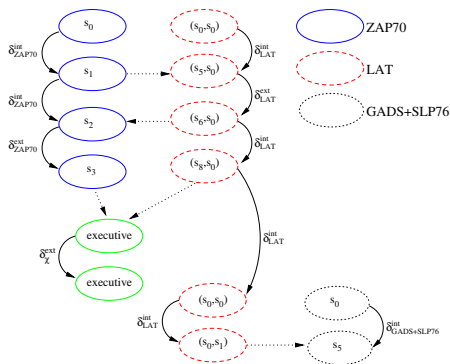
Modelling a DSDEVN

- We model DEVS-like molecules as **tagged located processes** $k[[P]]_{\Gamma}$
- Each molecule M_A of type A carries the information about the set I_A and models the input function Z_A
- The **type environment** Γ of each tagged located process represents the set of interaction partners I_A of molecule M_A in each state
- The input function is a syntactic construct in $tD\pi$ which gets inputs from all the process at the locations available in the type environment and gives an output value to the input channel of the molecule.
- The **executive** is a coordinator able to manage
 - ▶ time coordination given through timer assignments
 - ▶ coordination through messages of communications among processes
 - ▶ resource access constraints given by types

Modelling a DSDEVN

- We model DEVS-like molecules as **tagged located processes** $k[[P]]_{\Gamma}$
- Each molecule M_A of type A carries the information about the set I_A and models the input function Z_A
- The **type environment** Γ of each tagged located process represents the set of interaction partners I_A of molecule M_A in each state
- The input function is a syntactic construct in $tD\pi$ which gets inputs from all the process at the locations available in the type environment and gives an output value to the input channel of the molecule.
- The **executive** is a coordinator able to manage
 - ▶ time coordination given through timer assignments
 - ▶ coordination through messages of communications among processes
 - ▶ resource access constraints given by types

ZAP70 and LAT Signalling Pathway (II)



$$I_Z[[Zap]]_{\Gamma_Z} \mid I_L[[Lat]]_{\Gamma_L} \mid I_X[[Exec]]_{\Gamma_X}$$

$$Lat = u^{\Delta t_1}?(X : T).(P', Q'_L) \mid E$$

$$Q'_L = in^{\Delta t_2}?(X : T).(Q''_L, Q)$$

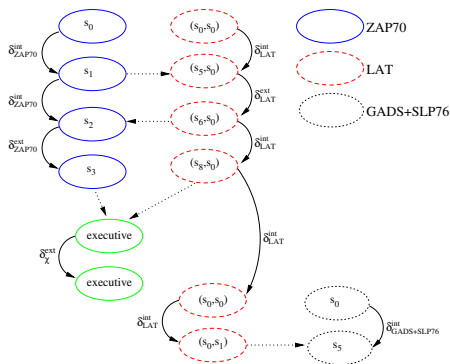
$$Q''_L = go I_Z.out^{\Delta t_3}!\langle v \rangle.(0, Q) \mid P$$

$$P = go I_X.u_X^{\Delta t_4}!\langle res \rangle.(Lat, ER)$$

$$\Gamma_L = \{I_Z : loc \{out : res\{T\}, \dots\}, \dots\},$$

$$I_X : loc \{u_X : res\{T'\}, \dots\} \dots$$

ZAP70 and LAT Signalling Pathway (II)



$$I_Z[[Zap]]_{\Gamma_Z} \mid I_L[[Lat]]_{\Gamma_L} \mid I_X[[Exec]]_{\Gamma_X}$$

$$Lat = u^{\Delta t_1}?(X : T).(P', Q'_L) \mid E$$

$$Q'_L = in^{\Delta t_2}?(X : T).(Q''_L, Q)$$

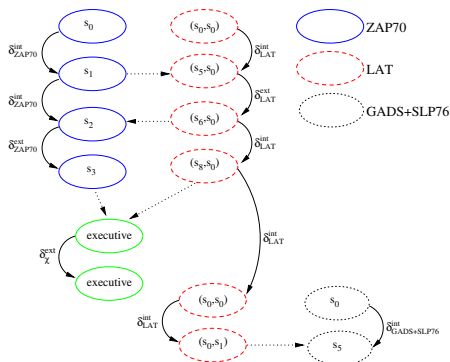
$$Q''_L = go I_Z.out^{\Delta t_3}!\langle v \rangle.(0, Q) \mid P$$

$$P = go I_X.u_X^{\Delta t_4}!\langle res \rangle.(Lat, ER)$$

$$\Gamma_L = \{I_Z : loc \{out : res\{T\}, \dots\}, \dots\},$$

$$I_X : loc \{u_X : res\{T'\}, \dots\} \dots$$

ZAP70 and LAT Signalling Pathway (II)



$$I_Z[[Zap]]\Gamma_Z \mid I_L[[Lat]]\Gamma_L \mid I_X[[Exec]]\Gamma_X$$

$$Lat = u^{\Delta t_1}?(X : T).(P', Q'_L) \mid E$$

$$Q'_L = in^{\Delta t_2}?(X : T).(Q''_L, Q)$$

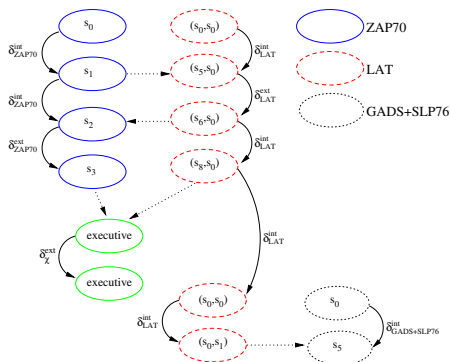
$$Q''_L = go I_Z.out^{\Delta t_3}!\langle v \rangle.(0, Q) \mid P$$

$$P = go I_X.u_X^{\Delta t_4}!\langle res \rangle.(Lat, ER)$$

$$\Gamma_L = \{I_Z : loc \{out : res\{T\}, \dots\}, \dots\},$$

$$I_X : loc \{u_X : res\{T'\}, \dots\} \dots$$

ZAP70 and LAT Signalling Pathway (II)



$$I_Z[[Zap]]\Gamma_Z \mid I_L[[Lat]]\Gamma_L \mid I_X[[Exec]]\Gamma_X$$

$$Lat = u^{\Delta t_1}?(X : T).(P', Q'_L) \mid E$$

$$Q'_L = in^{\Delta t_2}?(X : T).(Q''_L, Q)$$

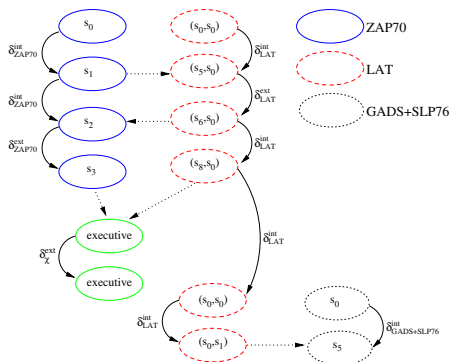
$$Q''_L = go I_Z.out^{\Delta t_3}!\langle v \rangle.(0, Q) \mid P$$

$$P = go I_X.u_X^{\Delta t_4}!\langle res \rangle.(Lat, ER)$$

$$\Gamma_L = \{I_Z : loc \{out : res\{T\}, \dots\}, \dots\},$$

$$I_X : loc \{u_X : res\{T'\}, \dots\} \dots$$

ZAP70 and LAT Signalling Pathway (II)



$$I_Z[[Zap]]\Gamma_Z \mid I_L[[Lat]]\Gamma_L \mid I_X[[Exec]]\Gamma_X$$

$$Lat = u^{\Delta t_1}?(X : T).(P', Q'_L) \mid E$$

$$Q'_L = in^{\Delta t_2}?(X : T).(Q''_L, Q)$$

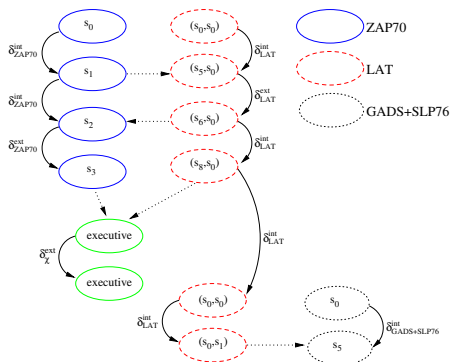
$$Q''_L = go I_Z.out^{\Delta t_3}!\langle v \rangle.(0, Q) \mid P$$

$$P = go I_X.u_X^{\Delta t_4}!\langle res \rangle.(Lat, ER)$$

$$\Gamma_L = \{I_Z : loc \{out : res\{T\}, \dots\}, \dots\},$$

$$I_X : loc \{u_X : res\{T'\}, \dots\} \dots$$

ZAP70 and LAT Signalling Pathway (II)



$$I_Z[[Zap]]\Gamma_Z \mid I_L[[Lat]]\Gamma_L \mid I_X[[Exec]]\Gamma_X$$

$$Lat = u^{\Delta t_1}?(X : T).(P', Q'_L) \mid E$$

$$Q'_L = in^{\Delta t_2}?(X : T).(Q''_L, Q)$$

$$Q''_L = go I_Z.out^{\Delta t_3}!\langle v \rangle.(0, Q) \mid P$$

$$P = go I_X.u_X^{\Delta t_4}!\langle res \rangle.(Lat, ER)$$

$$\Gamma_L = \{ I_Z : loc \{ out : res \{ T \}, \dots \},$$

$$I_X : loc \{ u_X : res \{ T' \}, \dots \} \dots \}$$

Other process algebra (or related) approaches

molecule as computation

- π -calculus representation [Regev & Shapiro & Silverman 2001, Regev & Shapiro 2004]
interactions between the molecules
- stochastic π -calculus [Priami & Regev & Shapiro & Silverman 2001]
quantitative aspects of biology
- BioAmbients [Cardelli & Panina & Regev & Shapiro & Silverman 2004]
compartments are also important
- Bio-calculus [Nagasaki & Onami & Miyano & Kitano 1999]
- κ -calculus (core formal molecular biology) [Danos & Laneve 2003]
complexation and activation

Other process algebra (or related) approaches

molecule as computation

- π -calculus representation [Regev & Shapiro & Silverman 2001, Regev & Shapiro 2004]
interactions between the molecules
- stochastic π -calculus [Priami & Regev & Shapiro & Silverman 2001]
quantitative aspects of biology
- BioAmbients [Cardelli & Panina & Regev & Shapiro & Silverman 2004]
compartments are also important
- Bio-calculus [Nagasaki & Onami & Miyano & Kitano 1999]
- κ -calculus (core formal molecular biology) [Danos & Laneve 2003]
complexation and activation

Other process algebra (or related) approaches

molecule as computation

- π -calculus representation [Regev & Shapiro & Silverman 2001, Regev & Shapiro 2004]
interactions between the molecules
- stochastic π -calculus [Priami & Regev & Shapiro & Silverman 2001]
quantitative aspects of biology
- BioAmbients [Cardelli & Panina & Regev & Shapiro & Silverman 2004]
compartments are also important
- Bio-calculus [Nagasaki & Onami & Miyano & Kitano 1999]
- κ -calculus (core formal molecular biology) [Danos & Laneve 2003]
complexation and activation

Other process algebra (or related) approaches

molecule as computation

- π -calculus representation [Regev & Shapiro & Silverman 2001, Regev & Shapiro 2004]
interactions between the molecules
- stochastic π -calculus [Priami & Regev & Shapiro & Silverman 2001]
quantitative aspects of biology
- BioAmbients [Cardelli & Panina & Regev & Shapiro & Silverman 2004]
compartments are also important
- Bio-calculus [Nagasaki & Onami & Miyano & Kitano 1999]
- κ -calculus (core formal molecular biology) [Danos & Laneve 2003]
complexation and activation

Modelling Approaches for Molecular Systems

and Simulation Tools

Models

- Pathway Modelling Language (PML)
- Boolean networks
- Petri nets

Simulation tools

- AMBER
- NAMD with the VMD visualiser
- CHARMM
- X-PLOR and CHIMERA

Modelling Approaches for Molecular Systems

and Simulation Tools

Models

- Pathway Modelling Language (PML)
- Boolean networks
- Petri nets

Simulation tools

- AMBER
- NAMD with the VMD visualiser
- CHARMM
- X-PLOR and CHIMERA

References



F. J. Barros.

Modeling formalisms for dynamic structure systems.

ACM Trans. Model. Comput. Simul., 7(4):501–515, 1997.



G. Ciobanu and D. Huzum.

Discrete event systems and client-server model for signaling mechanisms.

In C.Priami, editor, *CMSB*. LNCS 2602, 175–177, 2003.



M.A. Gibson.

Computational Methods for Stochastic Biological Systems.

PhD thesis, California Institute of Technology, 2000.



C. Prisacariu and G. Ciobanu.

Timed distributed π -calculus.

Technical Report IIT, Romanian Academy, 2005.

<http://iit.iit.tuiasi.ro/TR/>.



C. Priami, A. Regev, E.Y. Shapiro, and W. Silverman.

Application of a stochastic name-passing calculus to representation and simulation of molecular processes.

Inf. Process. Lett., 80(1):25–31, 2001.



A. Regev, E.M. Panina, W. Silverman, L. Cardelli, and E.Y. Shapiro.

Bioambients: an abstraction for biological compartments.

Theor. Comput. Sci., 325(1):141–167, 2004.



A. Regev and E. Shapiro.

The π -calculus as an abstraction for biomolecular systems.

In G. Rozenberg G. Ciobanu, editor, *Modelling in Molecular Biology*, Natural Computing Series, pages 219–266. Springer, 2004.



A. Regev, W. Silverman, and E.Y. Shapiro.

Representation and simulation of biochemical processes using the pi-calculus process algebra.

In *Pacific Symposium on Biocomputing*, pages 459–470, 2001.