Composing RT Objects: A Case for Petri Nets and Girard's Linear Logic

François Taïani, Mario Paludetto (LAAS-CNRS, Toulouse, F),
Jerôme Delatour (ESEO, Angers, F)

Plan

- Introduction: Time, Objects and Composition
- Objects and Petri Nets. A good Match?
- Using a Mutant Logic to the Rescue
- Example
- Conclusion and Outlook

Time, Objects and Composition

- Object = "Divide and Conquer" Unit

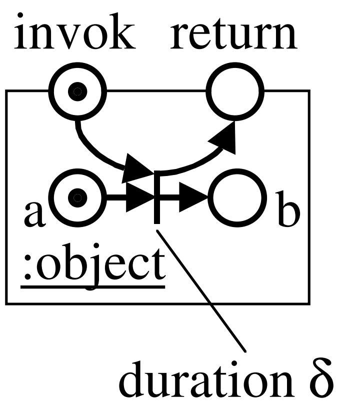
  (cross-project: reuse, cross-company: COTS)

- Integration Pitfall ⇒ Contract Based Development
- Real Time OO Systems ⇒ Time Sensitive Contracts
- (Time Sensitive) Reuse Question:
  "*Does that Object match my (Time Sensitive) Needs?*"

Checking of Temporal Interoperability
Object / Environment needed.

Objects and Petri Nets

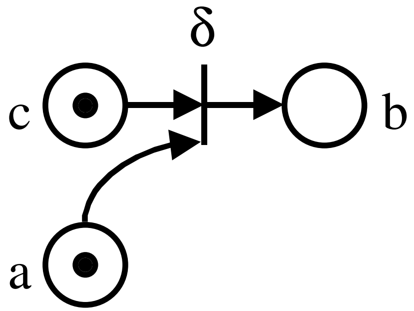
- Petri Net = Good for Control, Bad for Data
- Petri Net in Object:
  - Abstraction of Actual Behavior
  - Orders Object Needs / Services naturally (Exclusion, Precedence, …) (*Logical Time*)
  - Timed Transitions (*Quantitative Time*)
  - Token flows ≈ Threads



Using a Mutant Logic to the Rescue (I)

"... in which truths get lost once they are used."

[Girard90]



"A **at time** α **with** C **at time** γ"

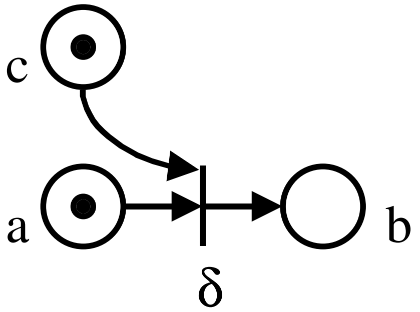
A(α), C(γ)

"**(**A **and** C**) produces** B**, in duration** δ"

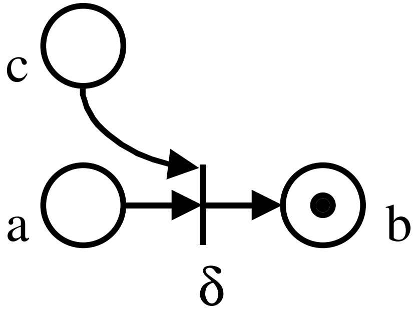
$[A \otimes C \multimap B](\delta)$

Using a Mutant Logic to the Rescue (II)

- A Rewriting Step:



$A(\alpha), C(\gamma) \; ; \; [A \otimes C \multimap B](\delta)$



$B(\mathit{max}(\alpha,\gamma) + \delta)$; <empty list>

François Taïani
(LAAS-CNRS)

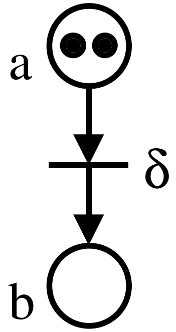
Using a Mutant Logic to the Rescue (III)

- Girard's Linear Logic ≈

  PN Unfolding with (*max*, +) Analysis
- Decisive Features :
  - Symbolic Duration Times
  - Multi-Instantiation of the same Structure (≈Threads)
  - (Finite) Loops
  - No State Explosion due to Concurrency

Using a Mutant Logic to the Rescue (IV)

- Multi-Instantiation of the same Structure

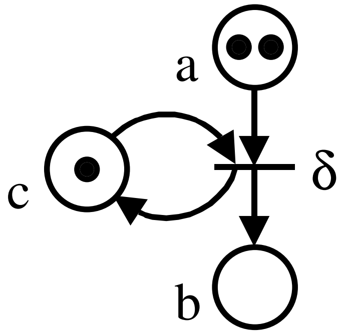


$\mathbf{A}(0)\ ,\quad A(0)\ ;\ 2\times\ [A \multimap B](\delta)$

~> $B(\delta)\ ,\quad \mathbf{A}(0)\ ;\quad [A \multimap B](\delta)$

~> $B(\delta)\ ,\quad B(\delta)\ ;\quad .$

$\Rightarrow$ **WCET = δ**



$A(0)\ ,\ \mathbf{A}(0)\ ,\ \mathbf{C}(0)\ ;\ 2\times[A\otimes C \multimap B\otimes C](\delta)$

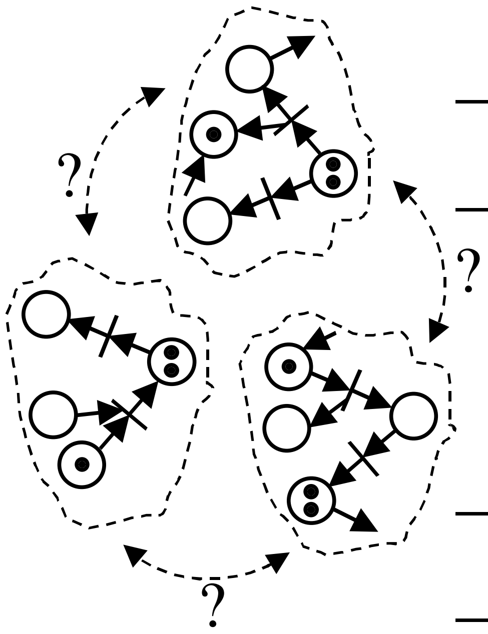
~~> $\mathbf{A}(0)\ ,\ \mathbf{C}(\delta)\ ,\ B(\delta)\ ;\quad [A\otimes C \multimap B\otimes C](\delta)$

~~> $B(2\delta)\ ,\ C(2\delta)\ ,\ B(\delta)\ ;\quad .$

$\Rightarrow$ **WCET = 2δ**

Using a Mutant Logic to the Rescue (V)

- No State Explosion due to Concurrency



—Collection of Partial States

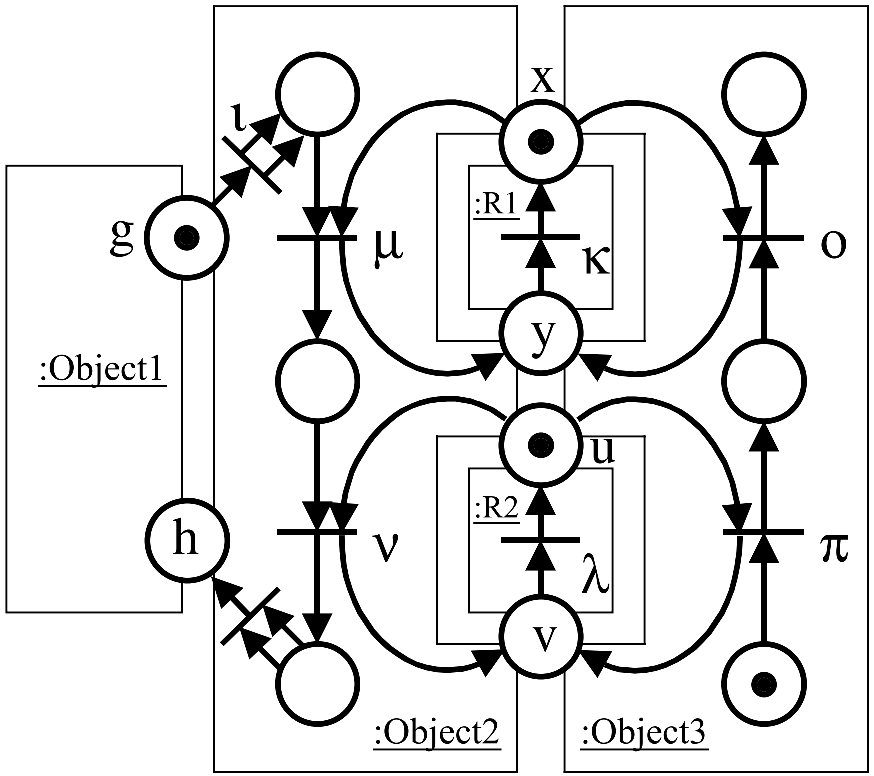
—No Assumptions about possible "Simultaneity" … until explicit Synchronization needed

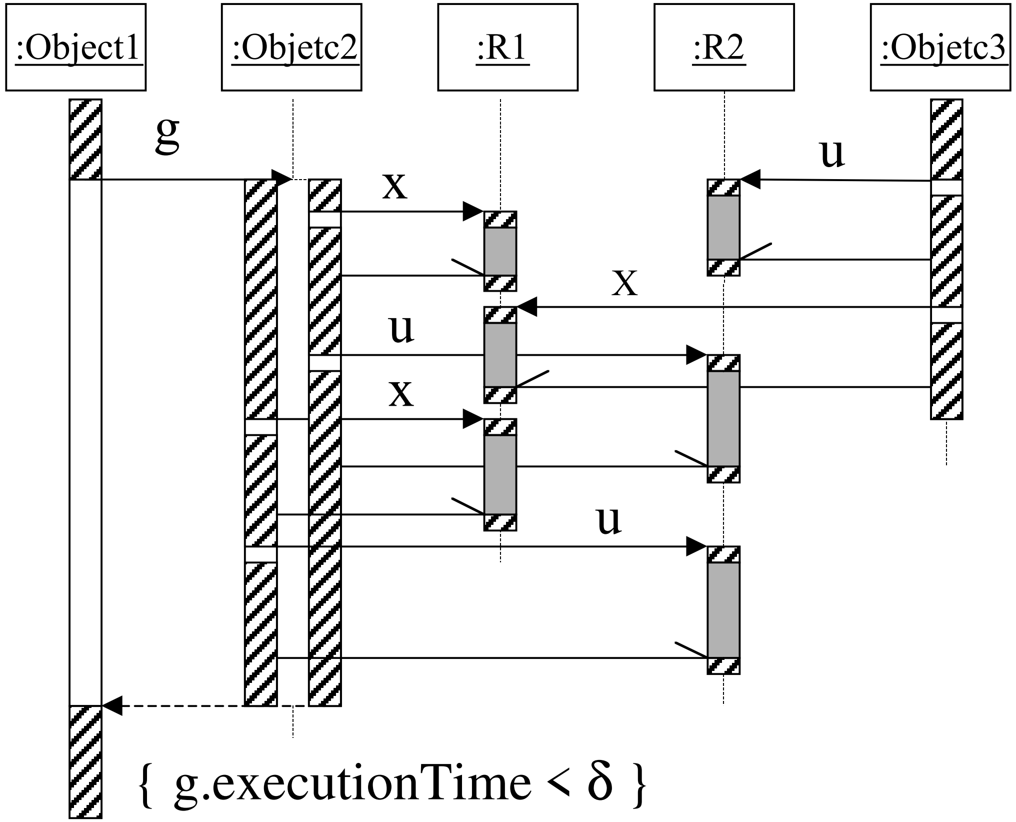
⇓

—Plain Application of Rules yields Correct Result

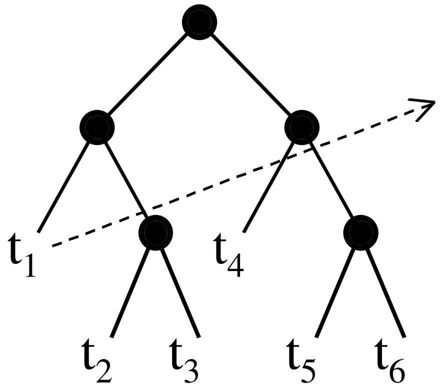
—Case Distinction only at "Conflict Points"

Example: Problem





Example: Results



$$t_1 = \nu + \mathit{max}\Big(2\mu + \kappa + \mathit{max}(\iota, \pi + o + \kappa), \nu + \lambda + \mathit{max}(\mu + \mathit{max}(\iota, \pi + o + \kappa), \pi + \lambda) \Big)$$

$\Rightarrow$

$\{$ g.executionTime $< \delta \}$ fulfilled as soon as

$$WCET_{model} = \mathit{max}(t_1, t_2, t_3, t_4, t_5, t_6) < \delta$$

● = Conflict (i.e. Choice) Point

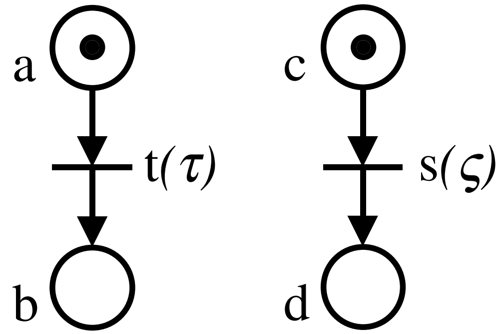
| = Plain Rewriting Sequence :

something ; something
~> something ; something
~> something ; something
~> something ; something

Conclusion and Outlook

- Temporal Constraint at one Object Interface
  $\rightarrow$ Temporal Requirements on whole System
- Further Developments:
  - Multiple Constraints? Choice of Context?
  - Scalability Study on Larger Examples
  - Tool Development (started)
  - Connection to Concrete Distributed OO Frameworks.

Annex: Partial Orders



$=_{def} [A \multimap B](\tau)$

$=_{def} [C \multimap D](\varsigma)$

- Interleaving: **t ; s** <u>and</u> **s ; t**
- Girard's LL:

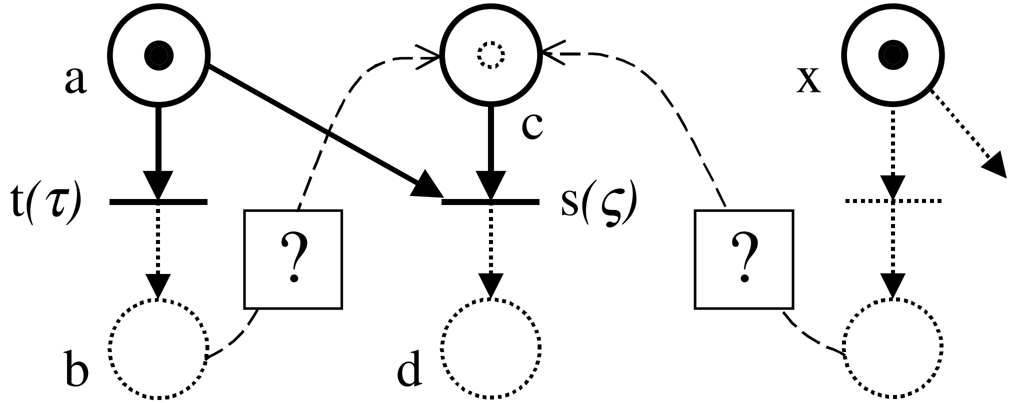
  If no Conflicts:

  → <u>All</u> rewriting Sequences are <u>equivalent</u>.

  → Only one does it all :

$$\mathbf{A}(\alpha) \quad , \; C(\gamma) \quad ; \; \mathbf{T}, S$$
$$\sim> \; B(\alpha+\tau) \; , \; \mathbf{C}(\gamma) \quad ; \; \mathbf{S}$$
$$\sim> \; B(\alpha+\tau) \; , \; D(\gamma+\varsigma) \; ; \; .$$

Annex: Potential Conflict Point



Question 1: Fire **t** ?, or wait for possible **C** ?
Question 2: Can a **C** be produced without **t** being fired?

→ Solution: Fire **t**, set flag to study causality t→C, and identify actual conflict afterwards, if any.