Logical Time & CCSL

Logical Time as/at Design Time

Charles ANDRE – project-team AOSTE
University of Nice-Sophia Antipolis and
INRIA Sophia-Méditerranée

About Time

What, then, is time? If no one ask of me, I know; if I wish to explain to him who asks, I know not.

Saint Augustine, Confessions, book XI, chapter XIV

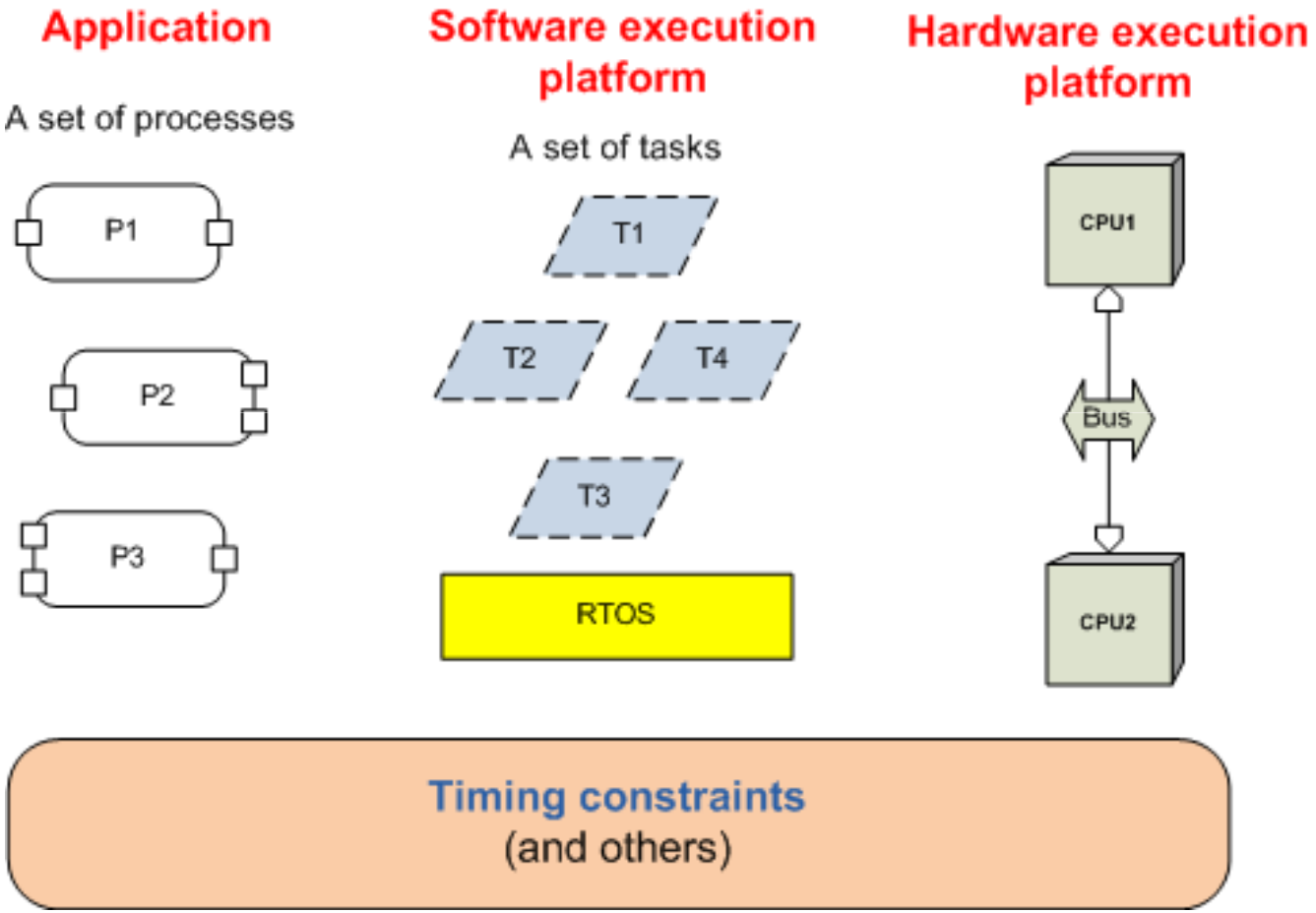
Reductionism w.r.t. Time: all talk that appears to be about time can somehow be reduced to talk about temporal relations among things and events.

Platonism w.r.t. Time: time is like an empty container into which things and events may be placed.

Causal ordering vs. temporal ordering

- A Time for (electronic) system designer

Design with time constraints



Requirements specification

A. Burns & A. Wellings. *Real-Time systems and Programming languages* (1997)

For many real-time systems, it is not sufficient for the software to be logically correct; the program must also satisfy timing constraints determined by the underlying physical system.

Ad hoc solution: a logically correct system is specified, designed, constructed and then tested to see if it meets its timing requirements. → A more systematic treatment of time is required.

One direction of development has concerned the use of formally defined language semantics and timing requirements, together with notations and logic that enable temporal properties to be represented and analyzed.

The other path has focused on the performance of real-time systems in terms of the feasibility of scheduling the required work load on the available resources.

Requirements specification (2)

A. Burns & A. Wellings. *Real-Time systems and Programming languages* (1997)

For many real-time systems, it is not sufficient for the software to be logically correct; the program must also satisfy timing constraints determined by the underlying physical system.

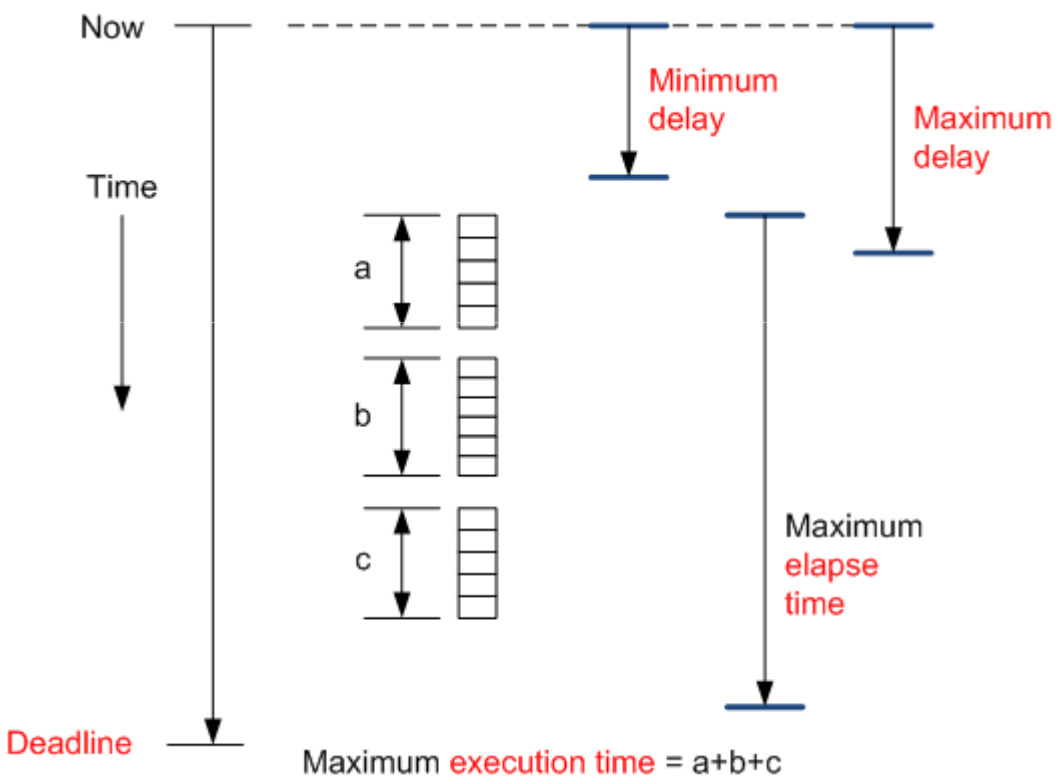
Ad hoc solution: a logically correct system is specified, designed, constructed and then tested to see if it meets its timing requirements. → A more systematic treatment of time is required.

One direction of development has concerned the use of formally defined model semantics and timing requirements, together with notations and logic that enable temporal properties to be represented and analyzed.

The other path has focused on the performance of real-time systems in terms of the feasibility of scheduling the required work load on the available resources.

Temporal scopes

Goal: facilitate the specification of the various timing constraints found in real-time applications



Some attributes

Temporal scopes can themselves be described as being either **periodic**, **aperiodic** or **sporadic**

Other properties: **Jitter**, burst

→ Often considered as Non Functional Properties (NFP) and used in later stages of the design

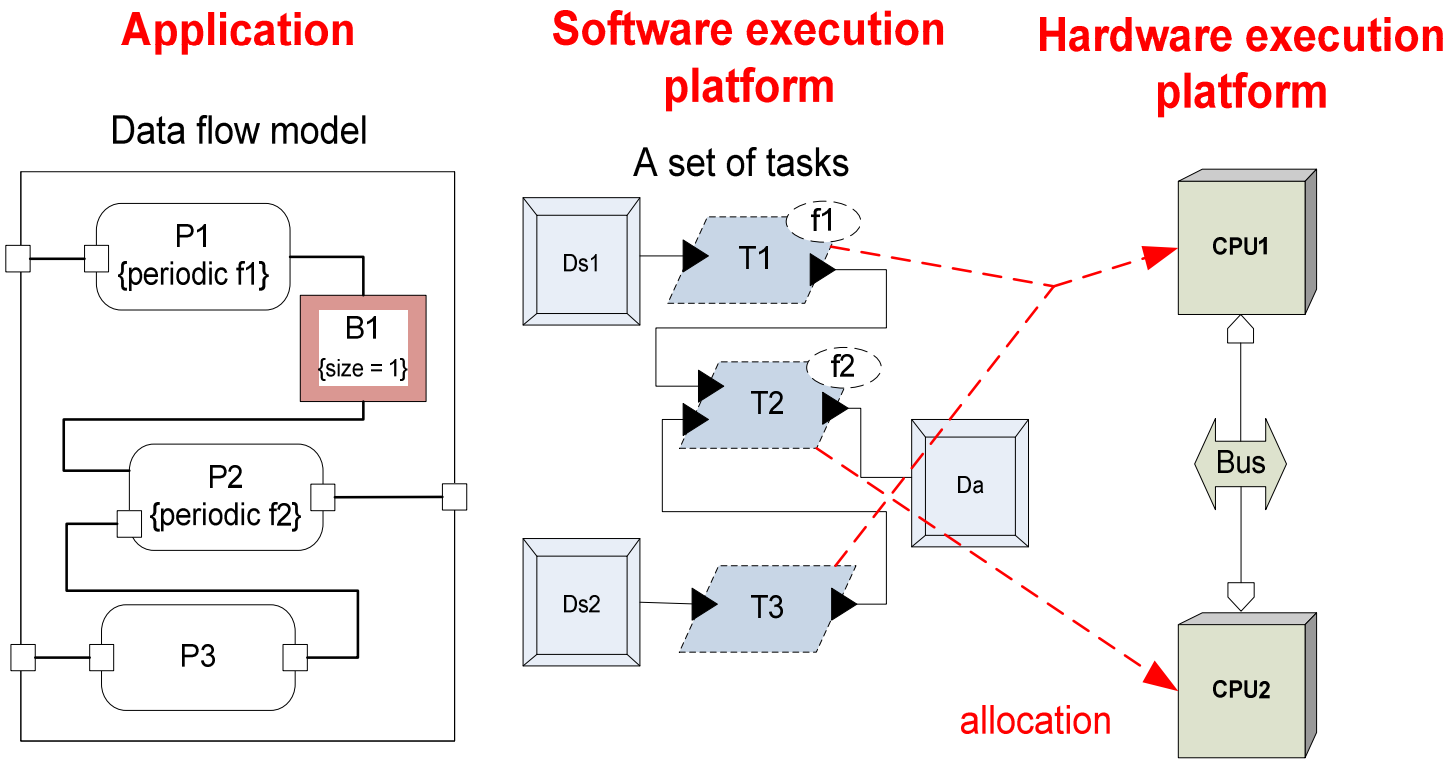
Temporal constraints as NFP

Success: Rate Monotonic scheduling
but requires
- periodic tasks
- **independent** tasks
- deadline = period
- uni-processor

What if
- mixture of periodic, sporadic, and aperiodic tasks
- shared resources → data dependency
- precedence → producer/consumer
- multi-processor → concurrency

Either complex scheduling / distribution
or considering temporal constraints not just as NFP but also
as functional → introduction of **Logical Time**

Platform-aware application



Timing constraints
(platform as a constraint on the application)

Logical time is …

- Useful at design
- **functional**
  - Should be made a first-class citizen
  - Not just a mere annotation introduced only for performance/time analysis
- **partially ordered** and progressively **refined**
  - The relative rates are **often** more important than the actual durations
  - possibly **approximate**
- **multiform**
  - Any event can be used as time reference
  - clock domains

→ The physical (real) time is just a special case ➔ chronometric time

Design Time

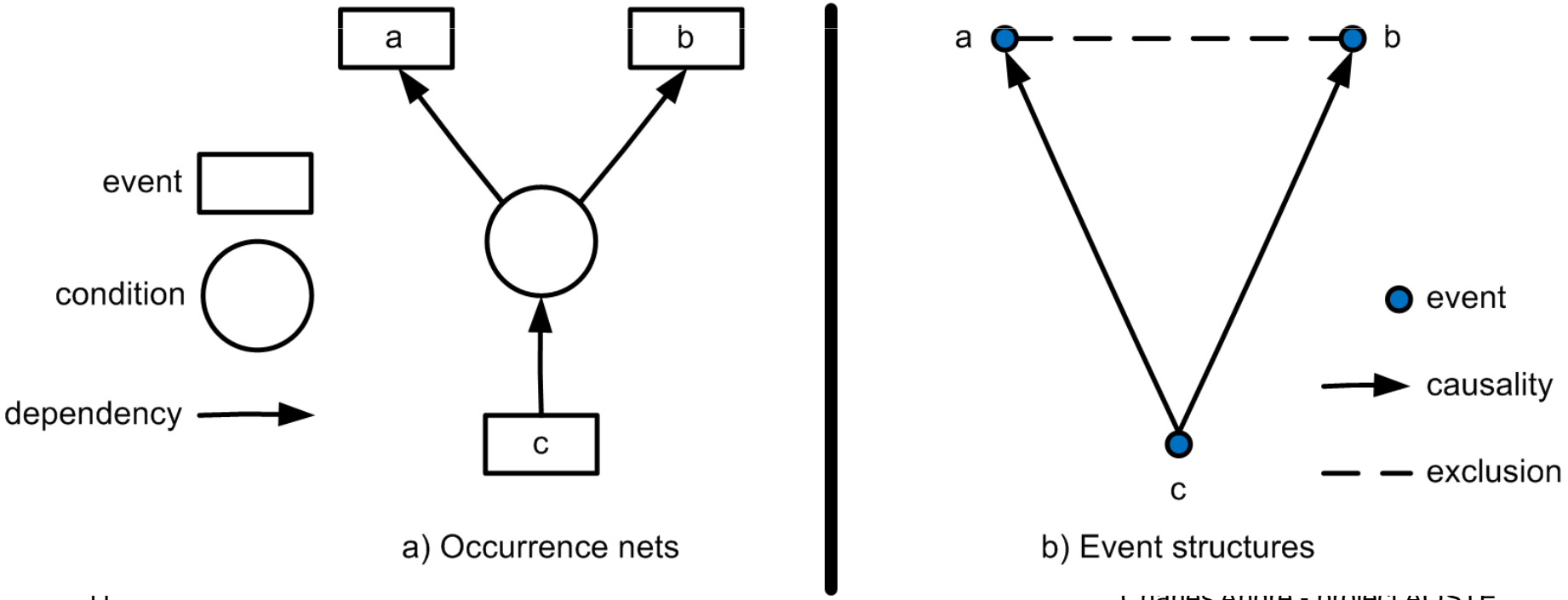
- **Untimed relationships**
  - Causality
  - Exclusion
- **Temporal relationships**
  - Before
  - After
  - At the same time (simultaneously)

Inspiration: Event structure

Untimed relationships

- From occurrence nets / event structures
  - Causality / dependency
  - Exclusion

{}, {c}, {c,a} {c,b} ~~{a,b,c}~~



Causality

Causality (untimed)

a **causes** b ➔ b cannot occur before a

becomes

- Temporal **sequentiality**

  a **precedes** b ➔ a always occurs before b

  or

- **Simultaneity**

  a **causes** b **and**

  a is **coincident** with b ➔ either a and b occur both or neither a nor b occur

Exclusion

- **Untimed exclusion**
  a **exclusiveWith** b ➔ either a or b can occur

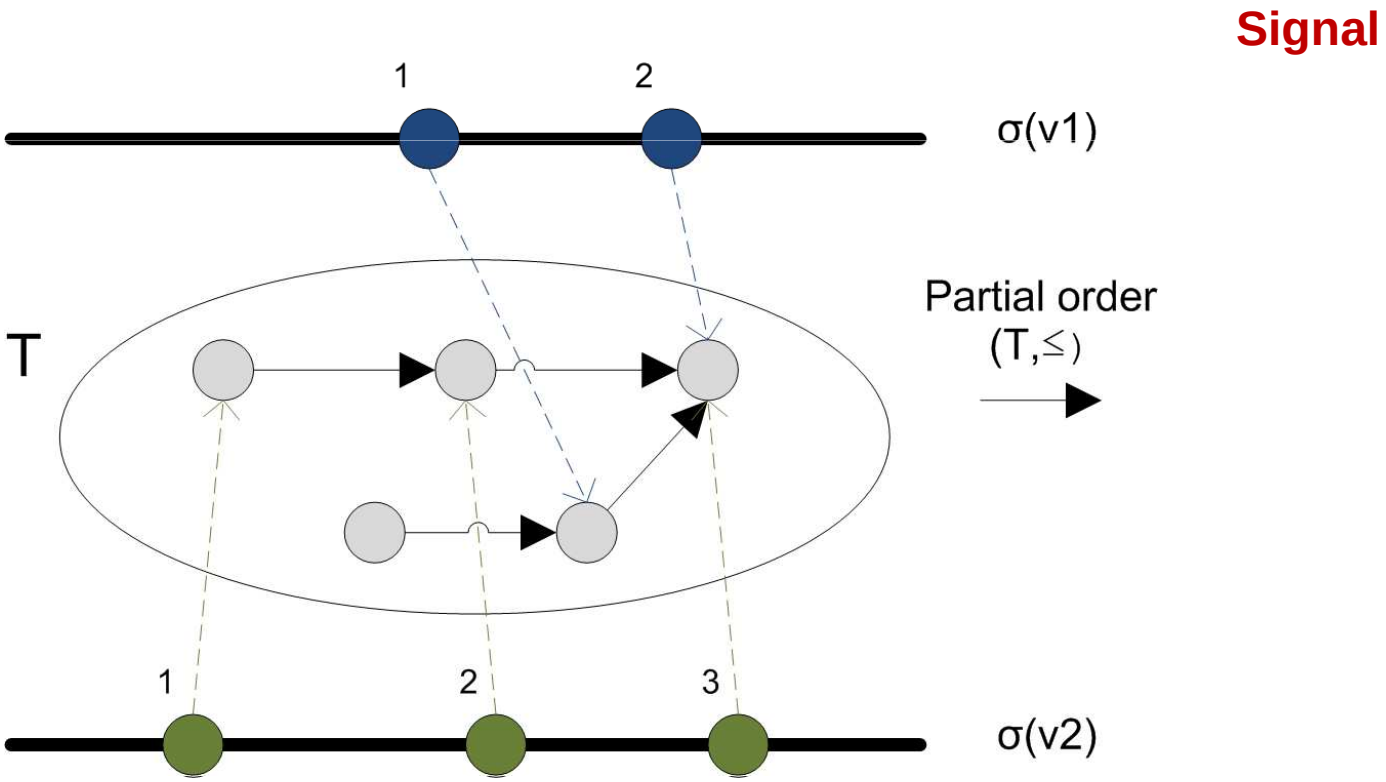
- **Timed exclusion**
  a # b ➔ a and b cannot occur simultaneously

- Asymmetry ➔ exclusion + **priority**

Inspiration: tag systems

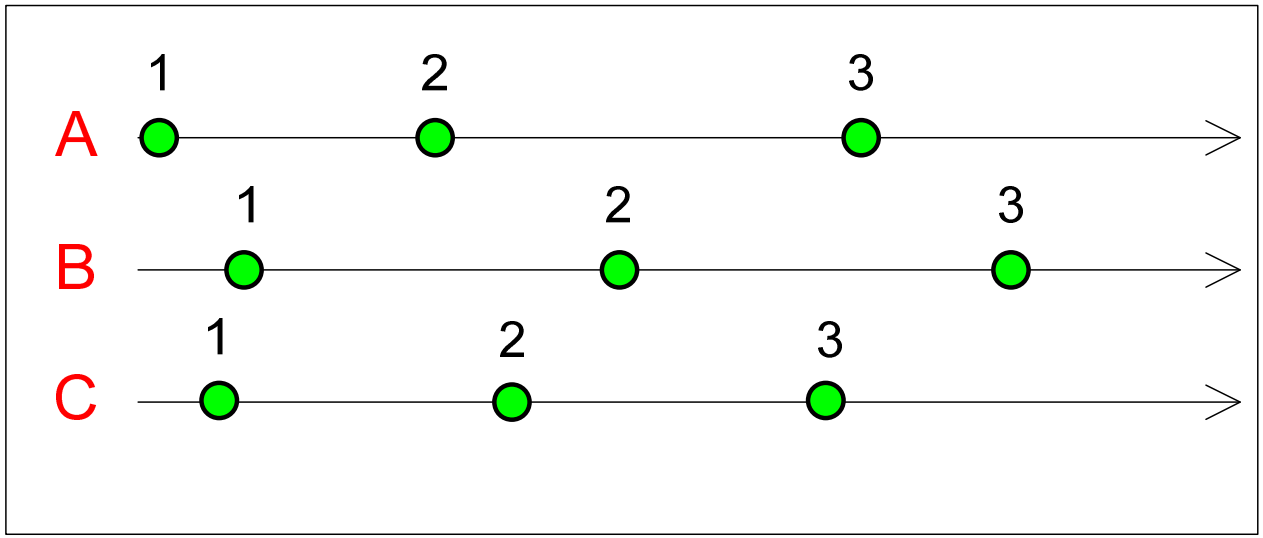
“Untimed” & “Timed” relationships

- Support for encoding several (timed or untimed) Model of Computations



Logical clock – Instant relations

- Clocks are *a priori* independent

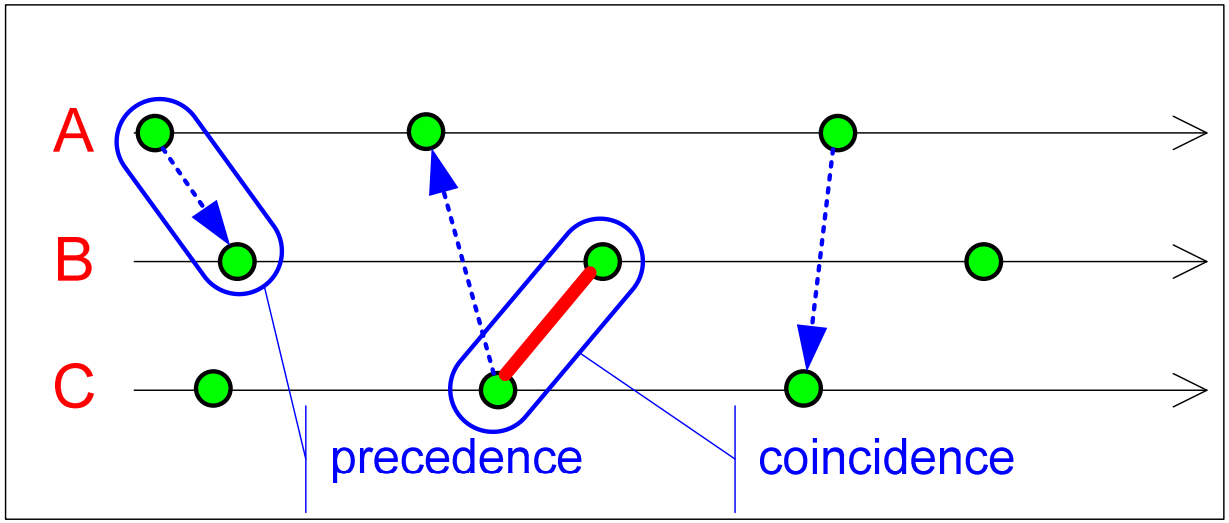


Illustrations with discrete time

Logical clock – Instant relations

- Clocks are *a priori* independent

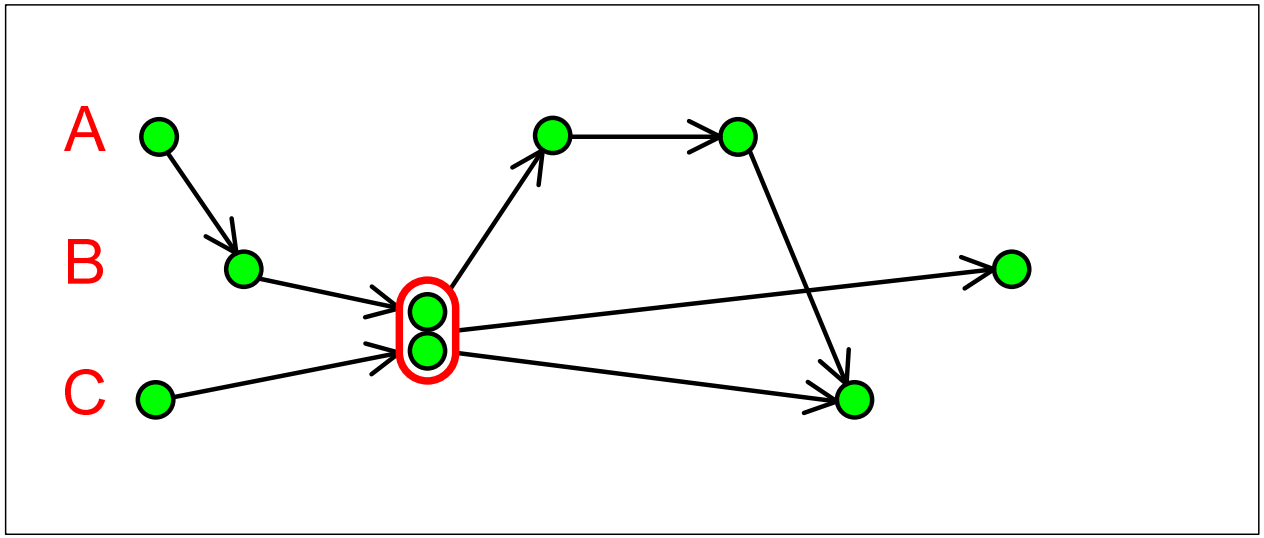
Then, we impose constraints on instants



Logical clock – Instant relations

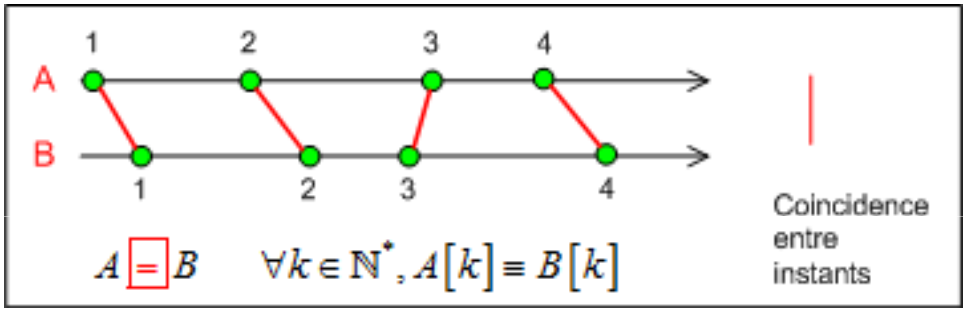
- Clocks are *a priori* independent

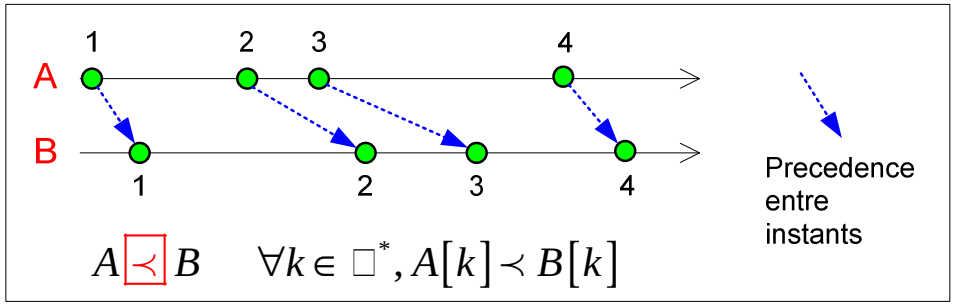
This result in a structure with partial ordering of instants



CCSL

- **Clock Constraint Specification Language**
- CCSL directly deals with **Clocks**, and
- **clock relations** (that define multiple instants relations)

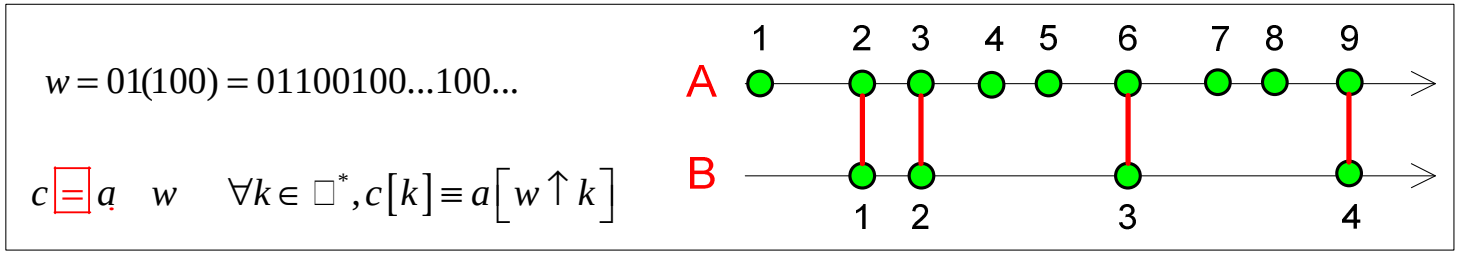




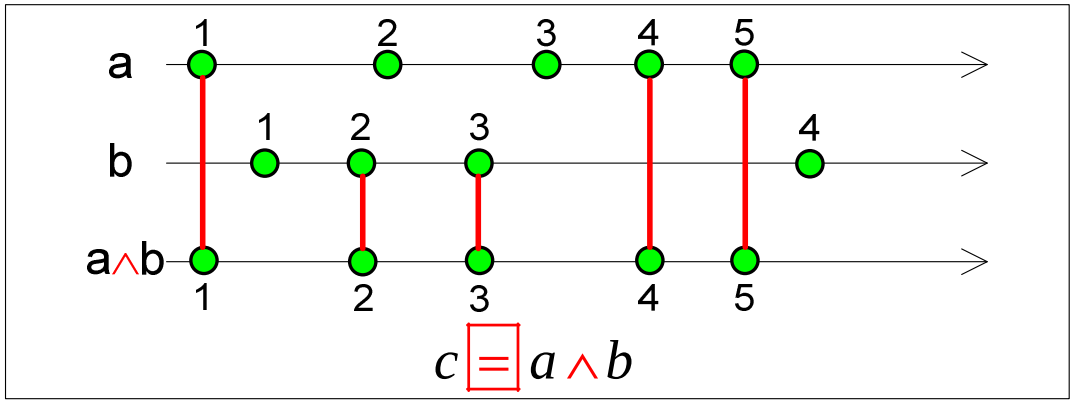
CCSL (cnt'd)

- and also Clock expressions (a way to specify new logical clocks)

Ex: **filtering**, a coincidence-based clock expression



Ex: **inf**, a clock expression considering both coincidence and precedence



Availability

- The **Time Model** is part of the OMG UML profile for MARTE
  http://www.omgmarte.org/
- **CCSL** has been introduced in a non-normative annex of MARTE

- Formal specification of CCSL
  - C. André, “Syntax and semantics of the clock constraint specification language (CCSL),” INRIA, Research Report 6925, 05 2009. [Online]. Available: http://hal.inria.fr/inria-00384077/en/

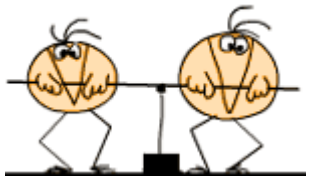
  - C. André, “Verification of clock constraints: CCSL observers in Esterel,” INRIA, Research Report 7211, 02 2010. [Online]. Available: http://hal.inria.fr/inria-00458847/en/

- TimeSquare Model Development Kit,
available at http://www-sop.inria.fr/aoste/dev/time square/

Illustration

Idea: An **event** can be represented by a (discrete) logical **clock**. The instants of this clock represent the **occurrences** of the event.

A concurrent system borrowed from “Introduction to General Net Theory”, C. A. Petri, LNCS n°84, Springer-Verlag, 1 980



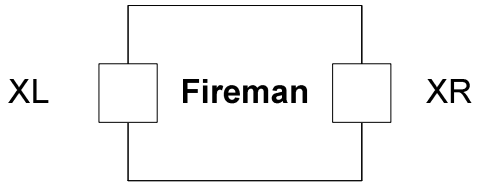
Pumping

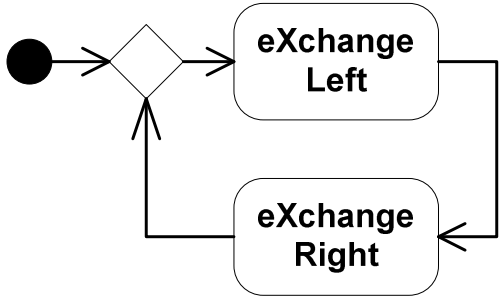
Bucket chain

Pouring water

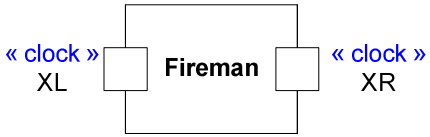
Bucket chain (1)

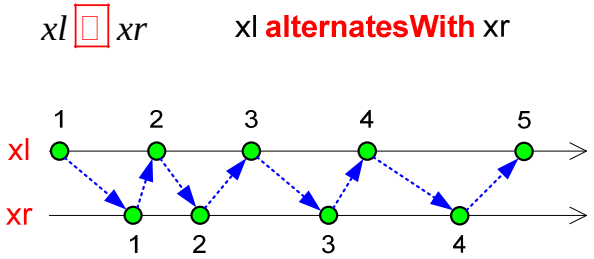
Component

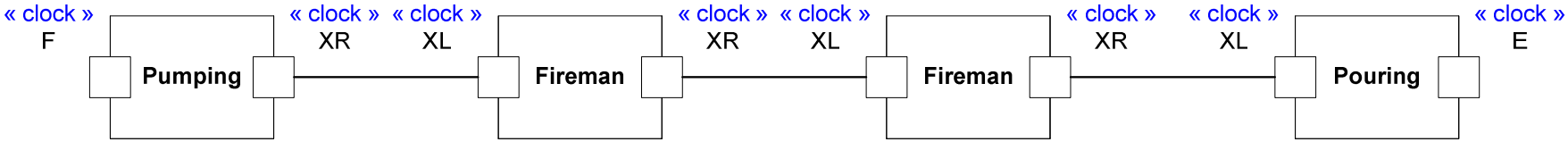




Applying the MARTE profile

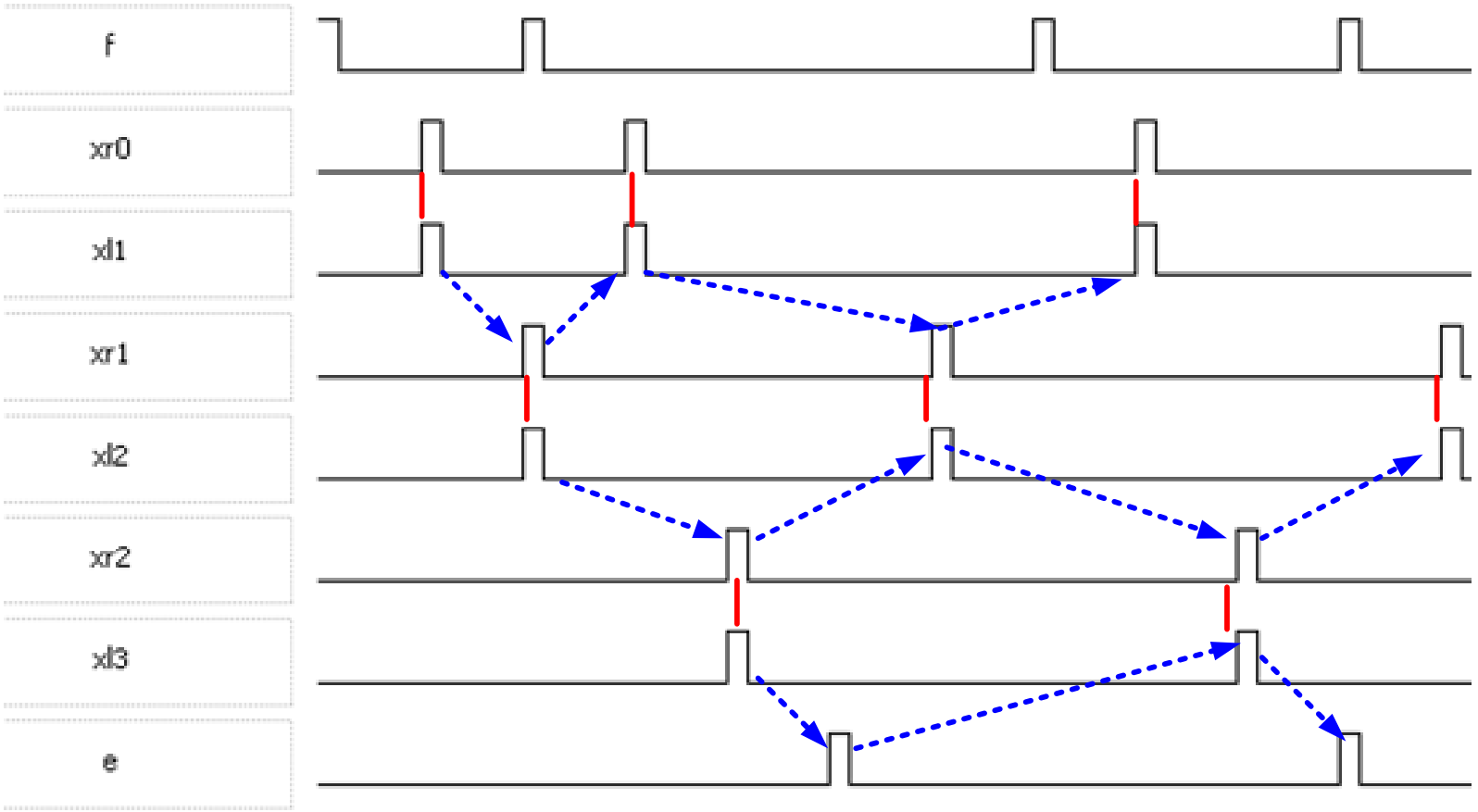






The bucket chain = a pipeline

Bucket chain (2)



Simulation with CCSL

Bucket chain (3)

Want a synchronous pipeline?

$xl \square xr$	Original constraint
$x = xl + xr$	Auxiliary clock
$b = x$	Additional constraint

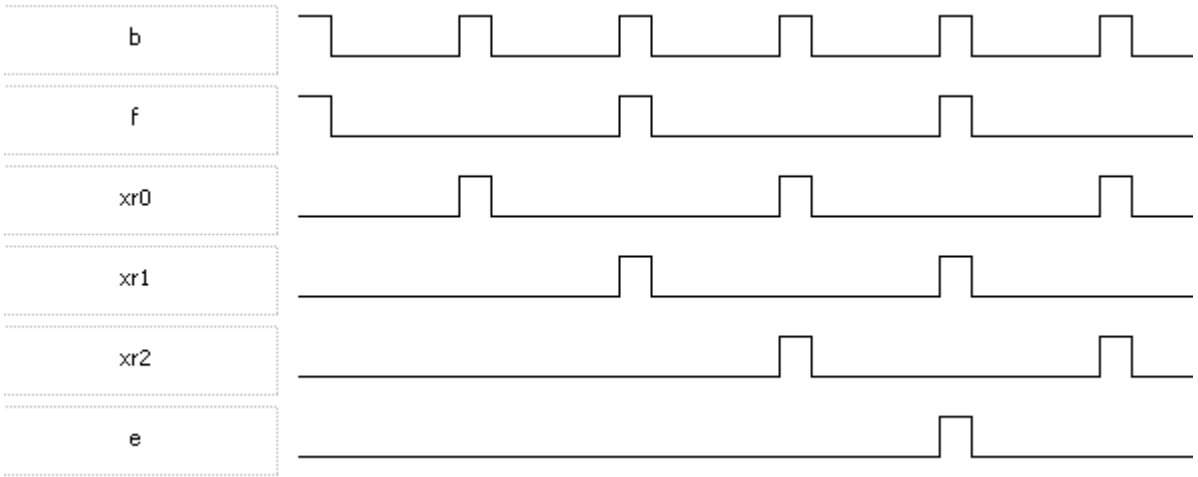
Beat clock

Bucket chain (4)

You must fill the pipeline ➔ introduce schedules

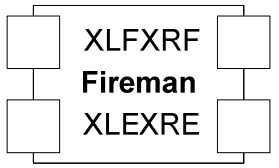
$s_k = b_{\cdot} \quad 0^k \square 1^\omega$ Auxiliary clock

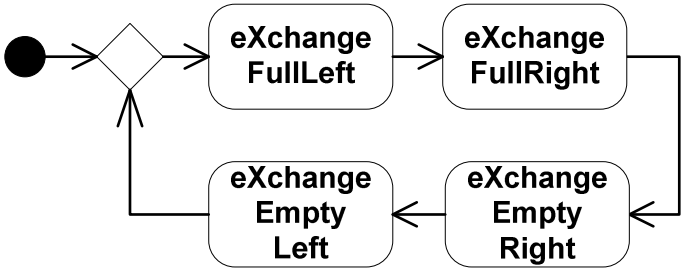
$x_k = s_k$ Additional constraint

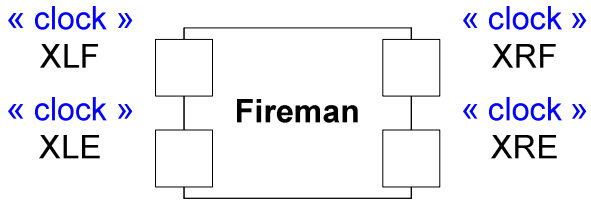


Bucket chain (5)

Refinement and Time refinement







$xl \square xr$ Original constraint

$xl = xlf + xle$

$xr = xrf + xre$ Auxiliary clocks

$xlf \square xle$

$xrf \square xre$ Additional constraints

To be continued …

- My colleagues will present several applications of CCSL
- You may have personal demonstration of CCSL at work.
- Contact me !