Coordination of tasks on a Real-Time OS

Guillermina Cledou & **José Proença** &
Bernhard H.C. Sputh & Eric Verhulst

Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR
Project POCI-01-0145-FEDER-029946

Slides: coord19.proenca.org

Coordination 2019 - Copenhagen, 20 june

Real-Time OS

VirtuosoNext™

(the real-time OS)

Visual Designer

(the IDE)

Coordination

Gluing tasks

Real-Time OS

VirtuosoNext™
(the real-time OS)

Visual Designer
(the IDE)

Coordination

Gluing tasks

Our contribution:
How to improve this part

Real-Time OS

Coordination

Gluing tasks

Our contribution:
How to improve this part

Altreonic

Virtu
(the

Visua

CISTER
Research Center in
Real-Time & Embedded
Computing Systems

DaVinci project

Distributed architectures: Variability and interaction in CPS

KURT modular vehicles

Outline

Programming
Real-time systems
with **VirtuosoNext™**

Understanding interactions
between tasks

Building interaction
protocols

Online prototype to
analyse protocols
http://arcatools.org/#virtuoso

Programming a RTOS
The classical way

Task 1

Initialise
(some work)
Every 10s
(more work)

Task 2

Initialise
(some work)
Every 10s
(more work)

Actuator task

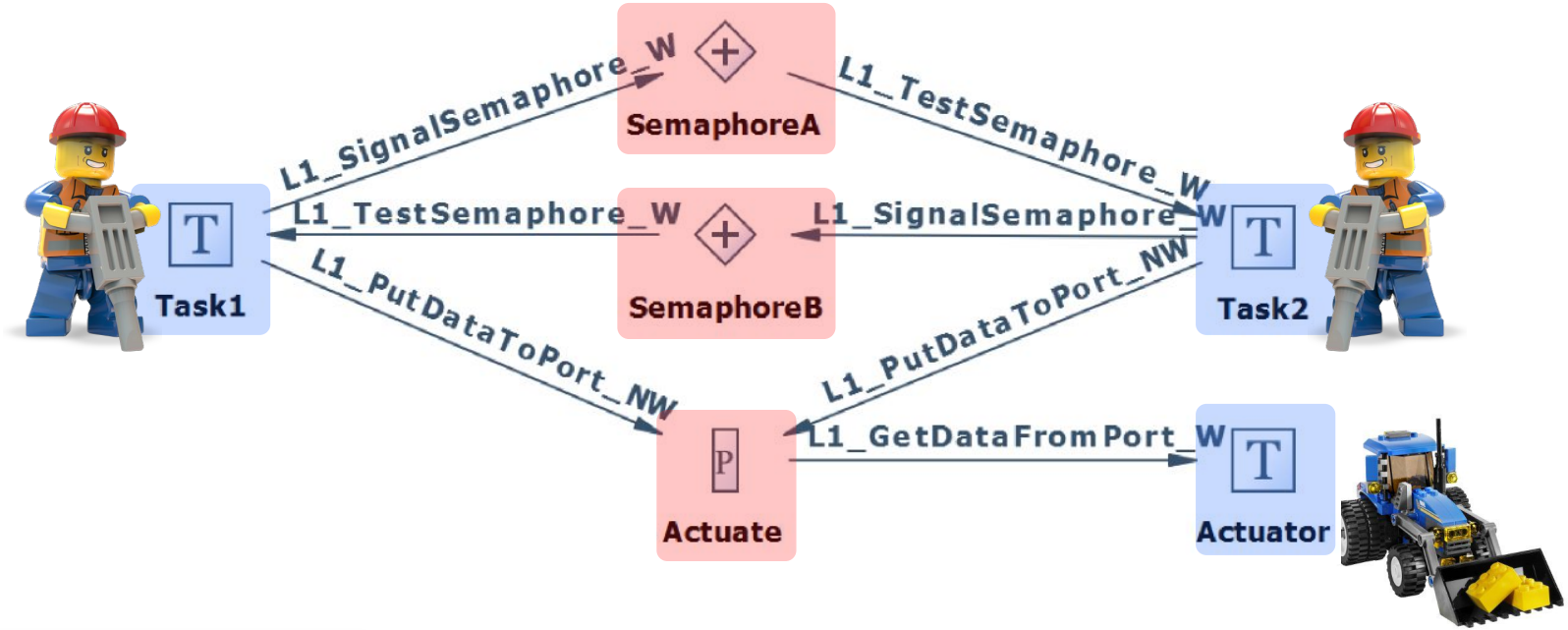
Initialise
(some work)
Every 5s
(Consume data from Task 1 & 2)

Shared data!

Scheduler

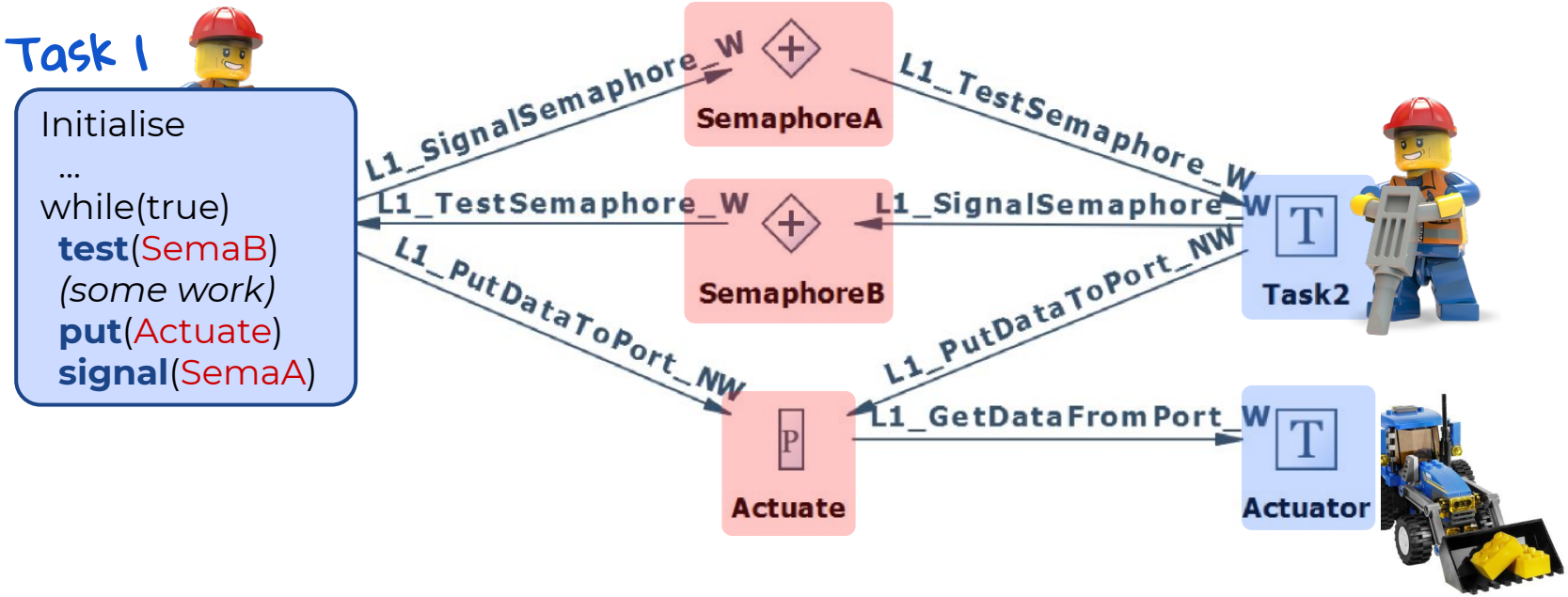
Deadline?
Computation time?
Priority?
Schedule...

Programming a RTOS
*The **VirtuosoNext™** way*



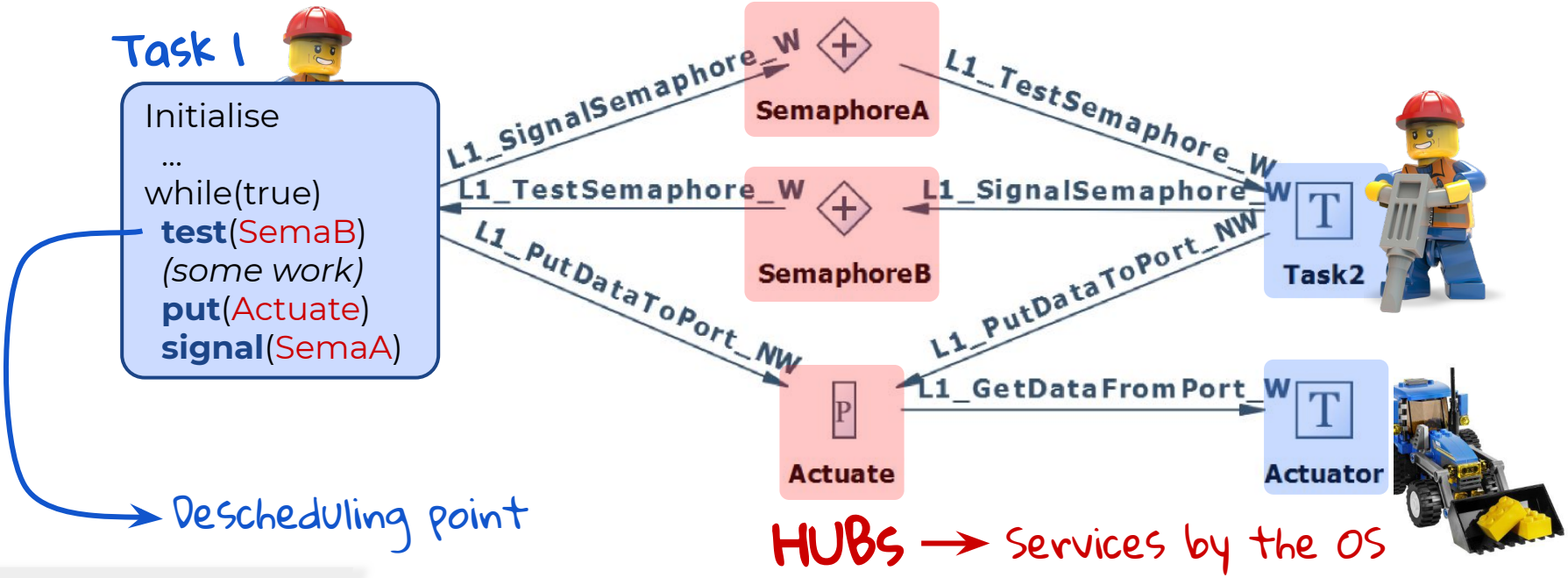
Slides: coord19.proenca.org

Programming a RTOS
*The **VirtuosoNext™** way*



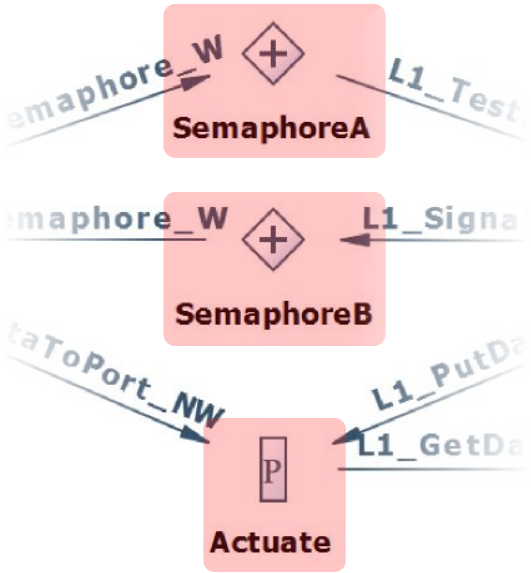
Slides: coord19.proenca.org

Programming a RTOS
*The **VirtuosoNext™** way*



Slides: coord19.proenca.org

Hubs in VirtuosoNext™

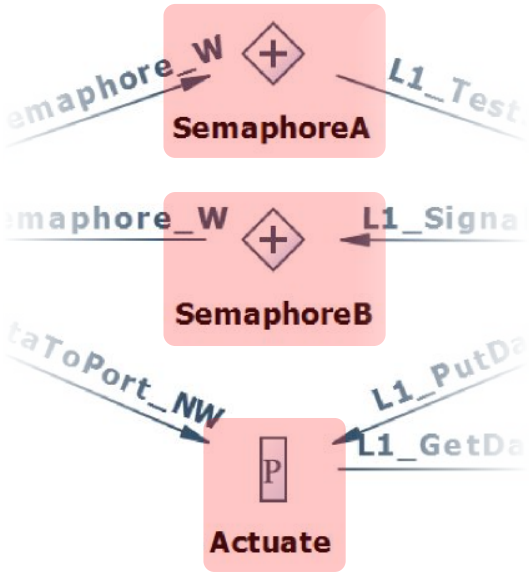


Executed by a dedicated **Kernel Task**

Decide who can be scheduled

May have state

Hubs in VirtuosoNext™

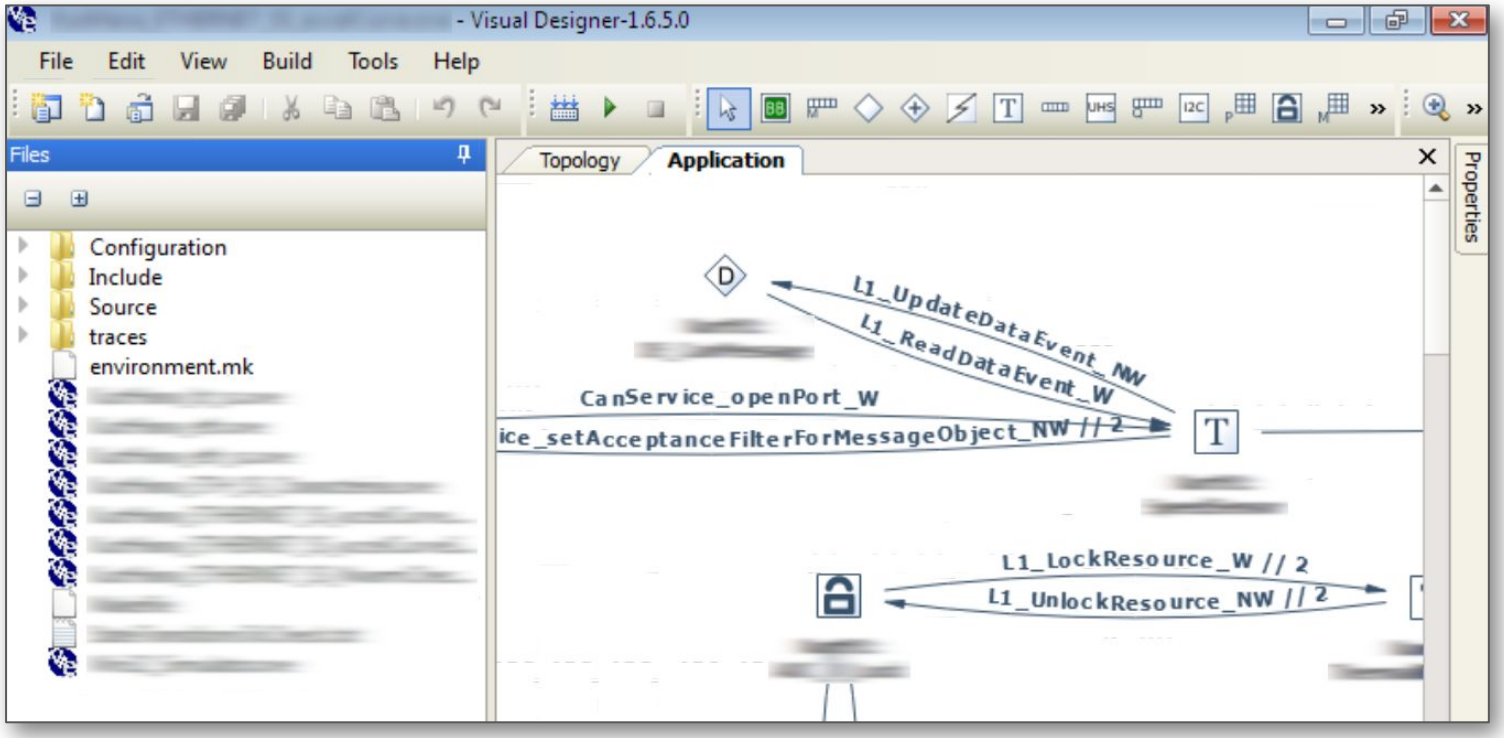


Executed by a dedicated **Kernel Task**

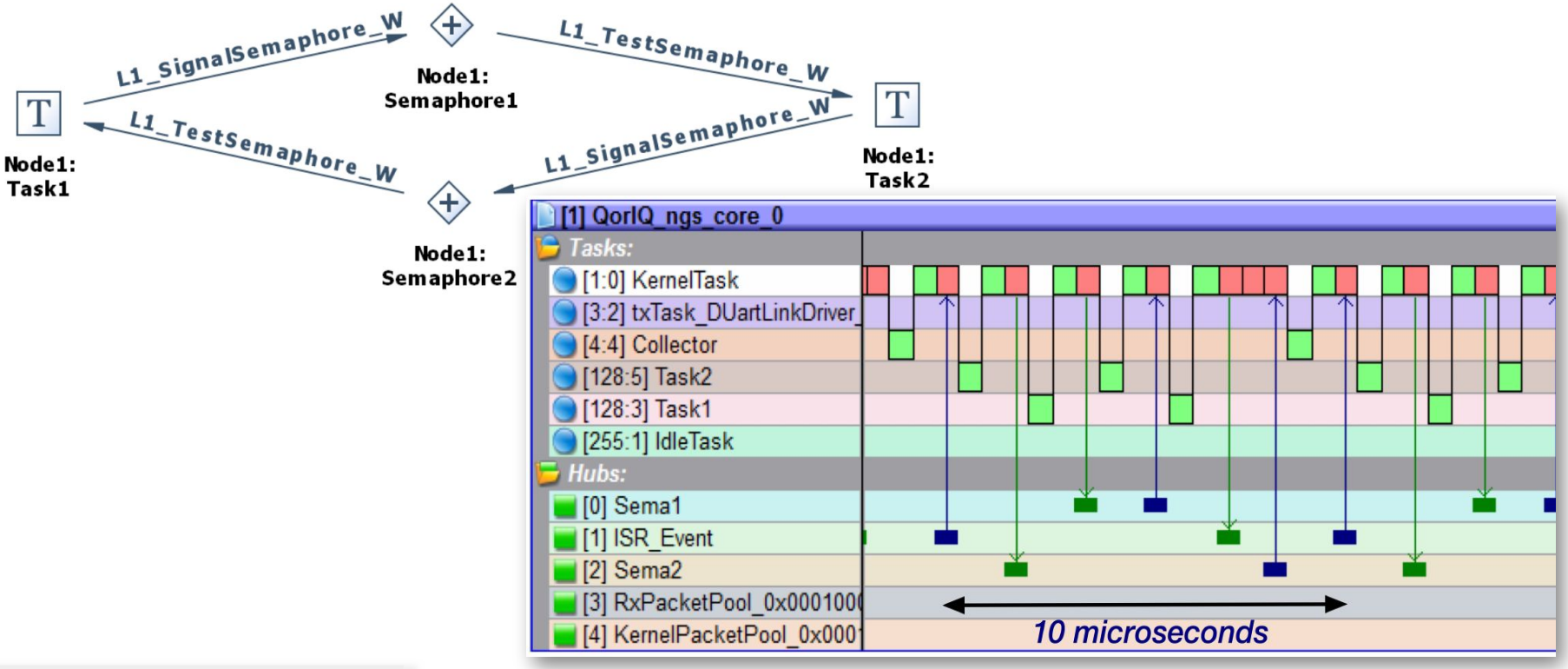
Decide who can be scheduled

May have state

Running Visual Designer



Timing executions



Hubs Semantics

Semaphore

signal – signals the semaphore, incrementing an internal counter `c`. Succeeds if `c < MAX`.

test – checks if `c > 0`, in which case succeeds, and decrements `c`.

Port

put – signals some data entering the port
get – signals some data leaving the port

Both must synchronize to succeed.

Hubs Semantics

DataEvent

update – sets an event and buffers some data, overriding any previous data. Always succeeds.

read – reads the data. Succeeds if the event is set.

clear – clears the buffer and the event.

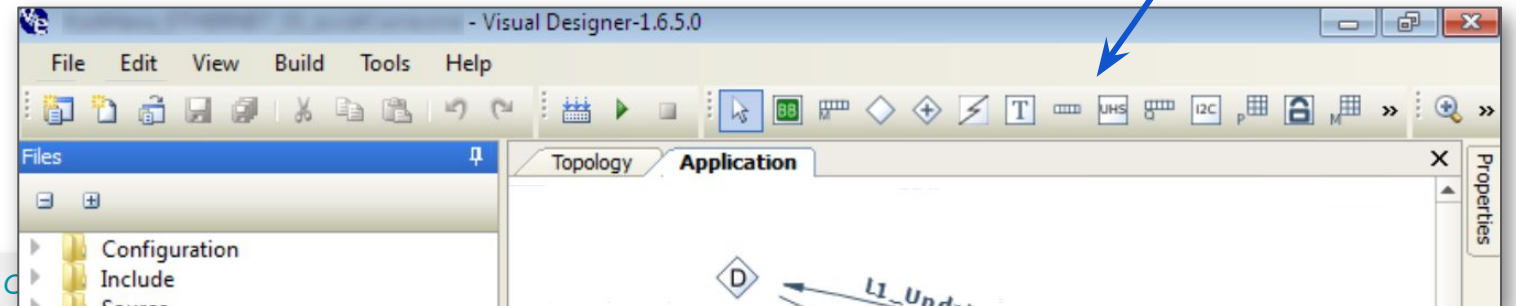
FIFO

enqueue – buffers some data in the queue. Succeeds if the queue is not full.

dequeue – reads the next data. Succeeds if the queue is not empty.

Challenge

How to **add** new Hubs?

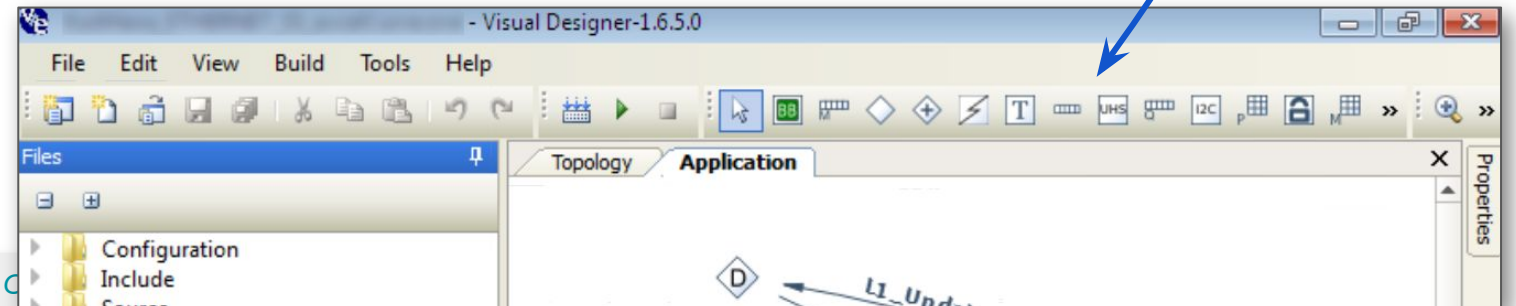


Challenge

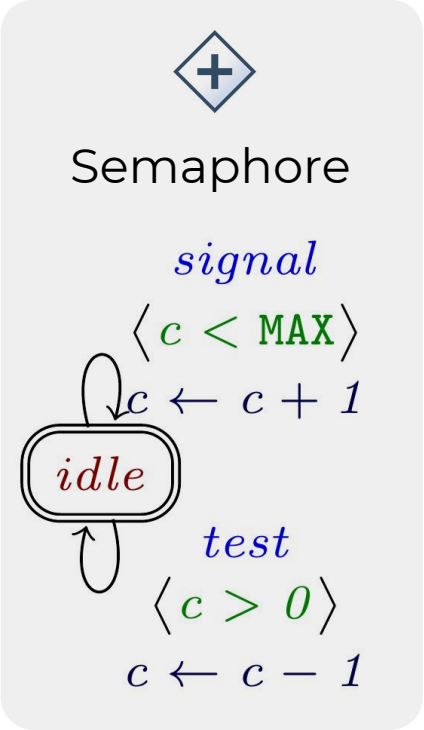
How to

- **generalise** Hubs
- **build** complex (and useful) Hubs
- **analyse** Hubs (trust Hubs)

How to **add** new Hubs?



Hubs ++



Automata semantics

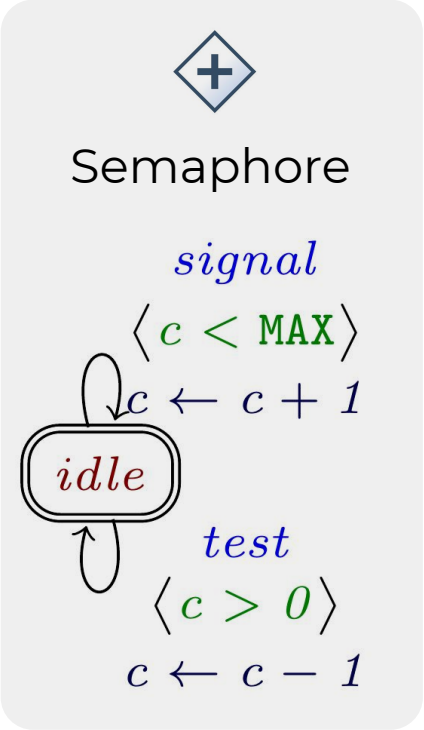
entry point

guard

update $u := x \leftarrow e \mid u; u \mid u|u$

state

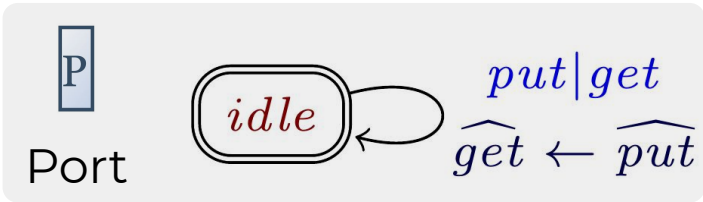
Hubs ++



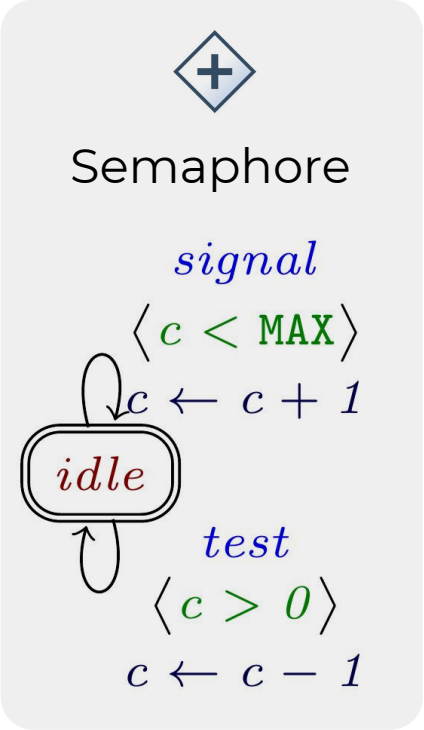
Automata semantics

with data + synchronisation

many entry points



Hubs ++

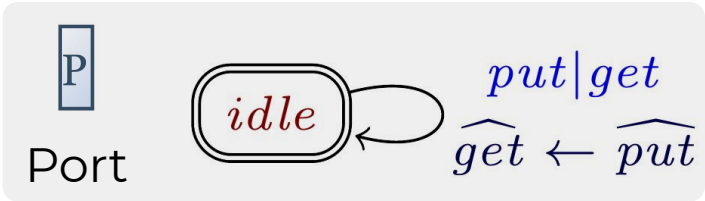


Automata semantics

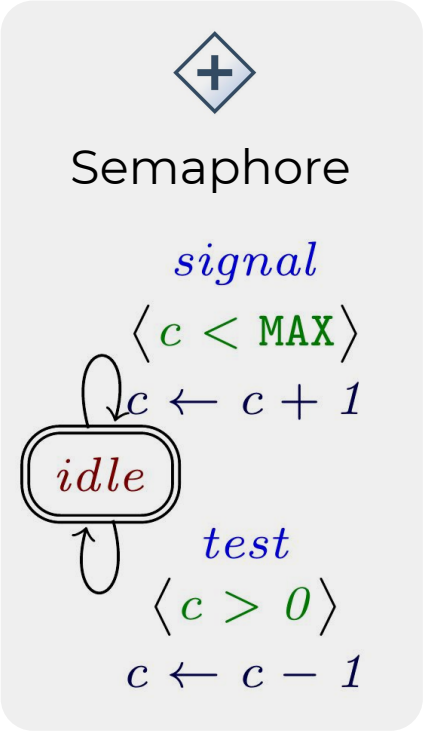
with data + synchronisation

with composition

Based on Constraint Automata



Hubs ++



Automata semantics

with data + synchronisation

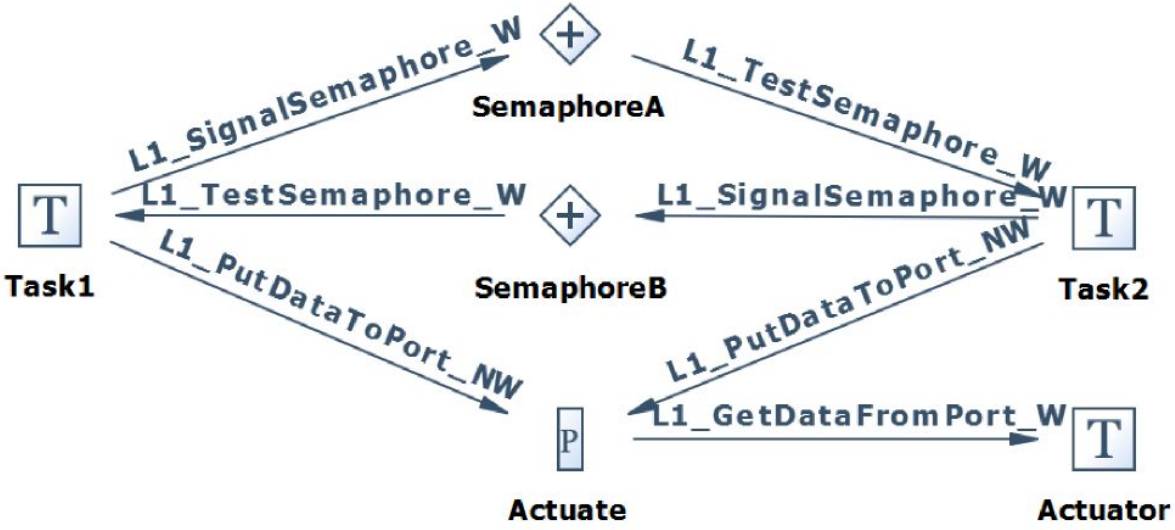
with composition

- online tool -

Based on Constraint Automata

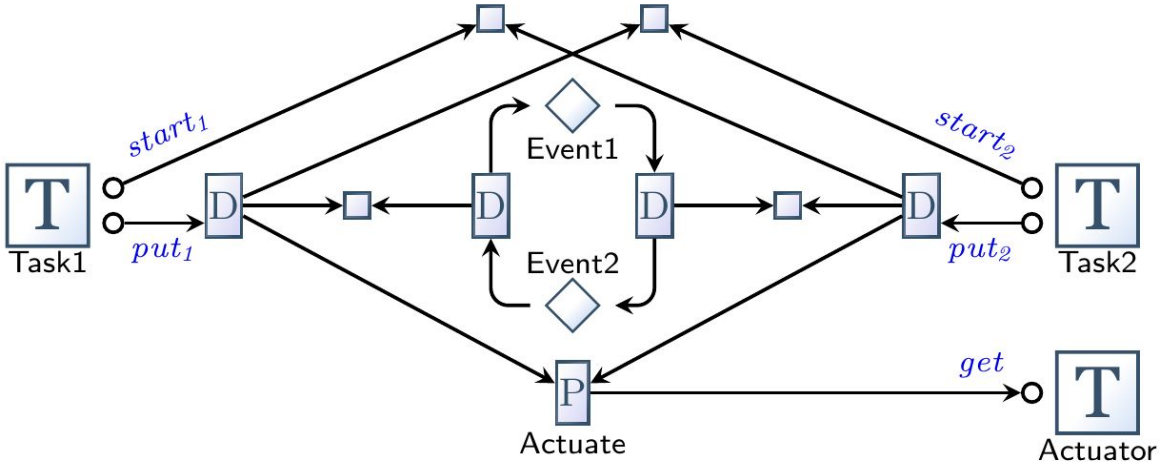
http://arcatools.org/#virtuoso

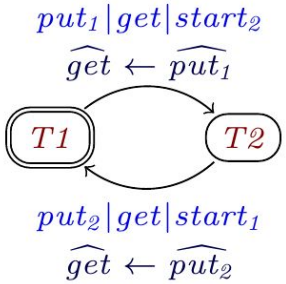
Example 1



Tasks 1 & 2 alternate between **signal**, **test**, and **put**

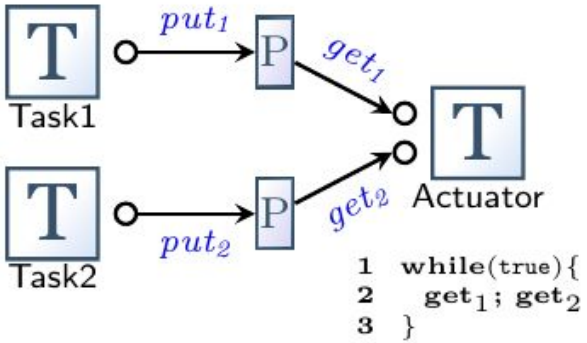
Example 2



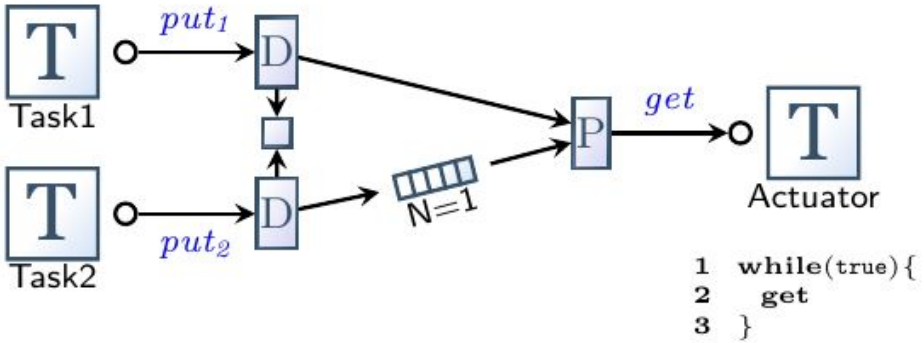


Tasks 1 & 2 can **start** and eventually **put** a value

Examples 3 & 4



Actuator alternates between tasks



Hub imposes alternation

Insights gained

Performance (1000 rounds)
Example 1 - *40ms*
Run complex example (2)
- as a user task - *60ms*
- as simpler native hub - *30ms*

Simplest (example 3) - *20ms*

Coordination burden often moved to tasks

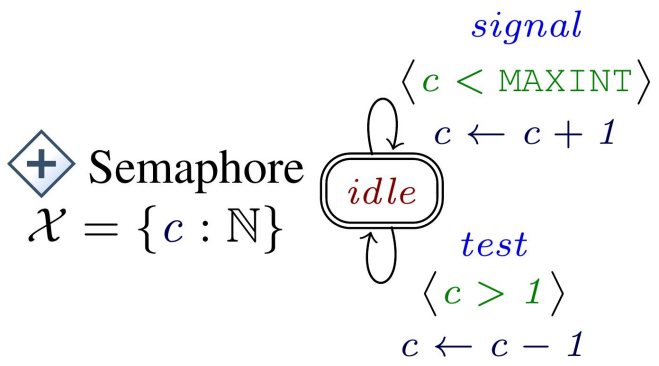
The devil is in the details

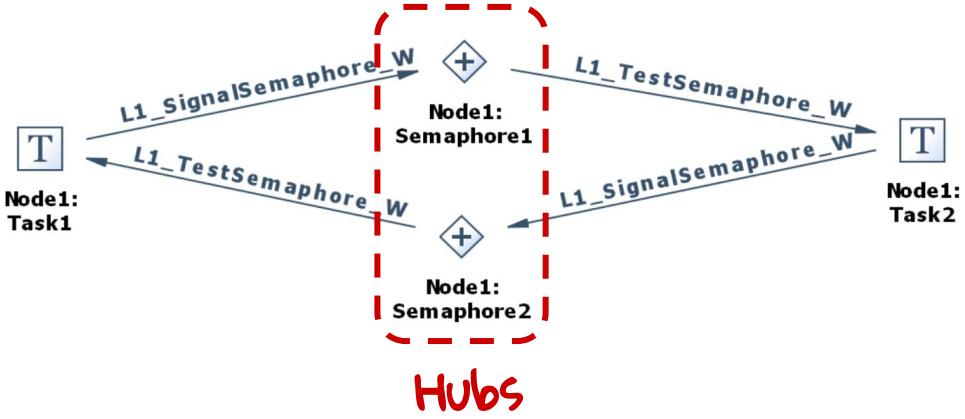
Nr. Context Switches
Ex. 1 - *17* / Ex. 2 - *13* / Ex. 3&4 - *9*

Blackbox tasks:
harder to reason

Wrap up

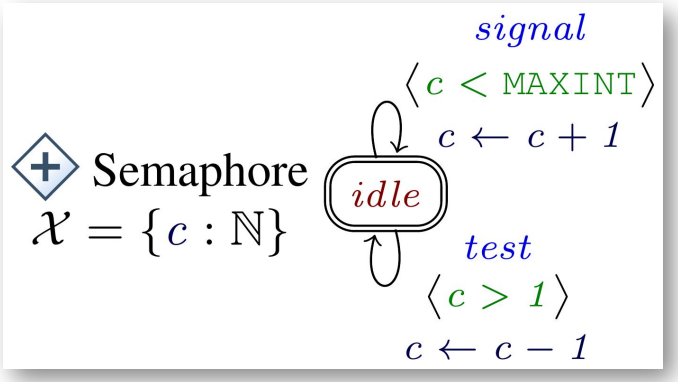
Automata semantics

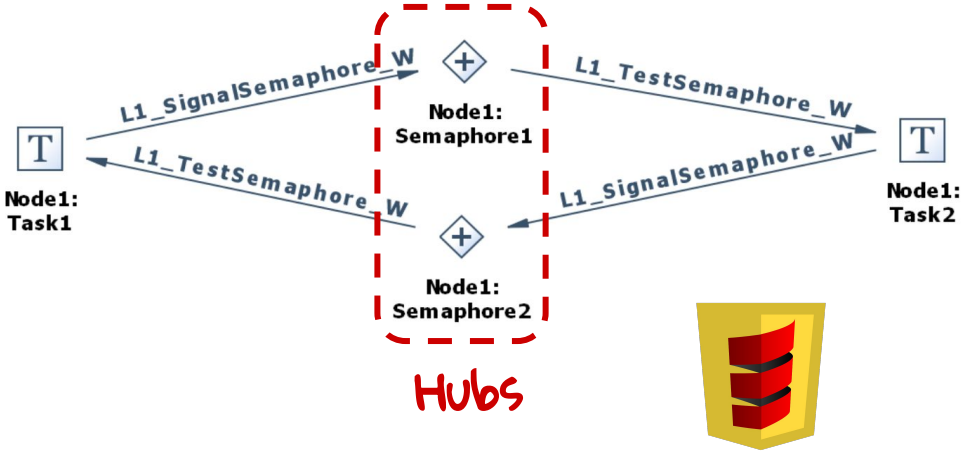




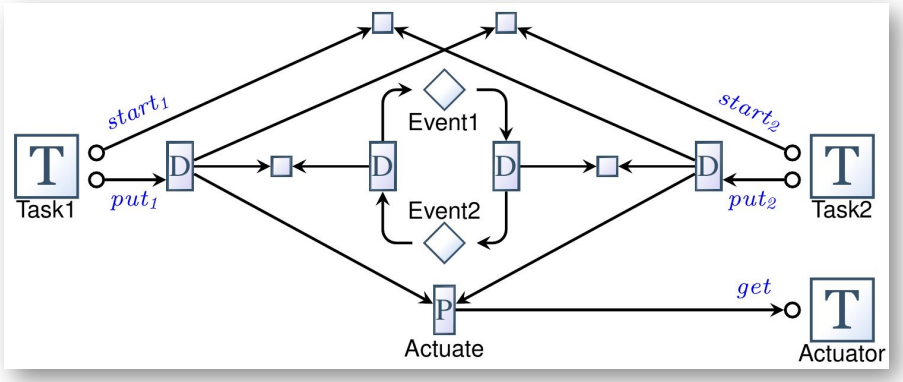
Wrap up

Automata semantics



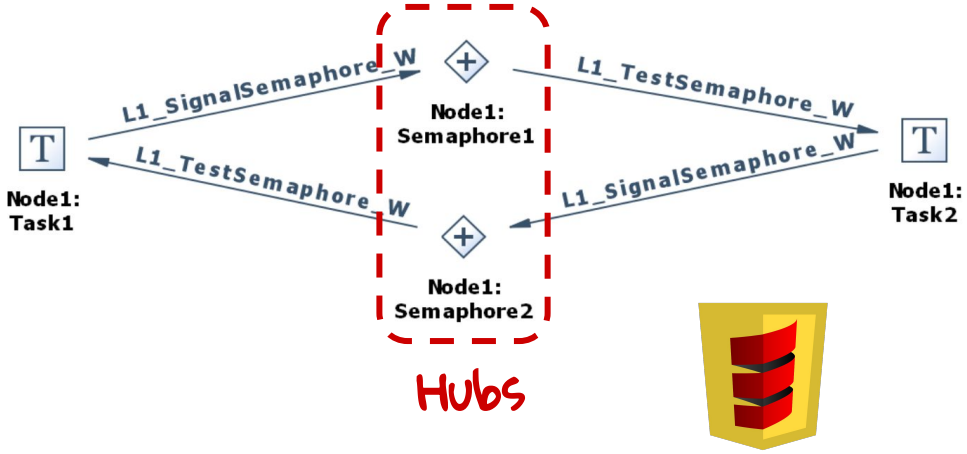


Complex hubs by composition

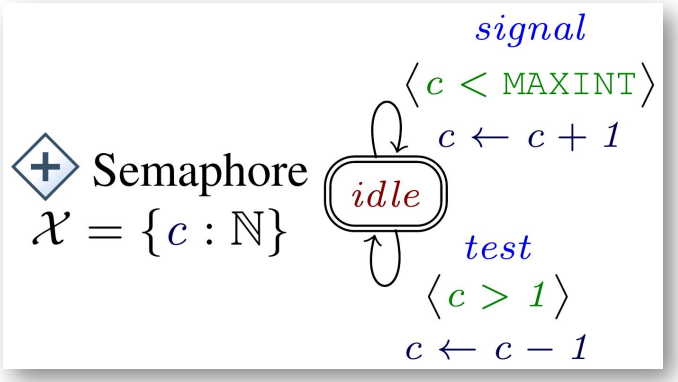


Slides: coord19.proenca.org

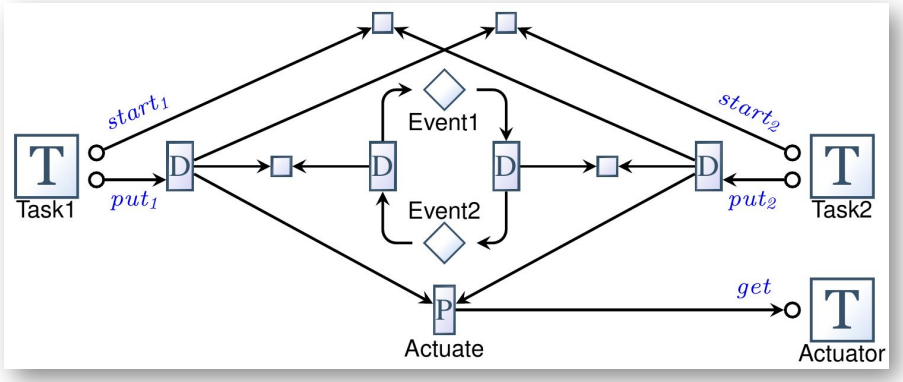
Wrap up



Automata semantics



Complex hubs by composition



Ongoing work:

- DSL for tasks
  (Funct. React. Prog + hubs)
- Commun. across nodes

Thank you!

Slides: coord19.proenca.org