Building effective formal models to prove time properties of networked automation systems

Silvain Ruel, Olivier de Smet

and Jean-Marc Faure

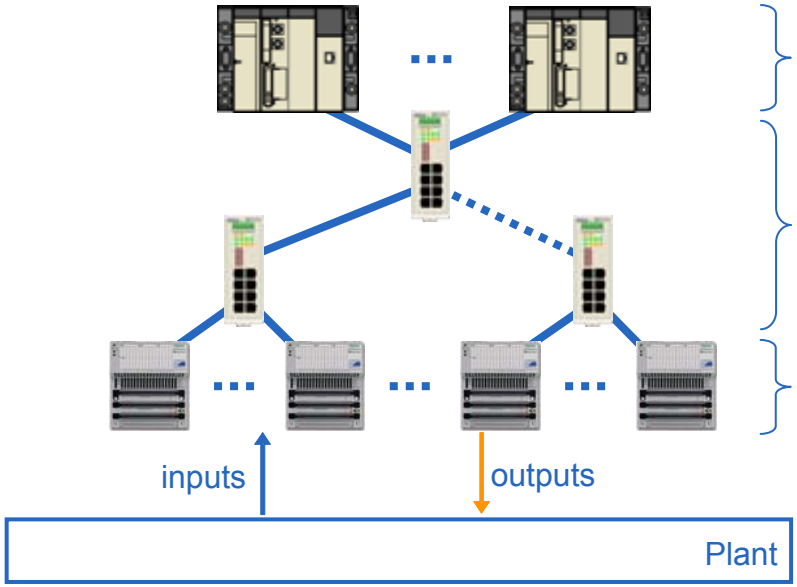
LURPA, ENS de Cachan

France

Outline

- Time performances of networked automation systems
- Timed model-checking for time performances evaluation
- Building effective formal models
- Case study
- Conclusions and outlooks

Considered class of NAS

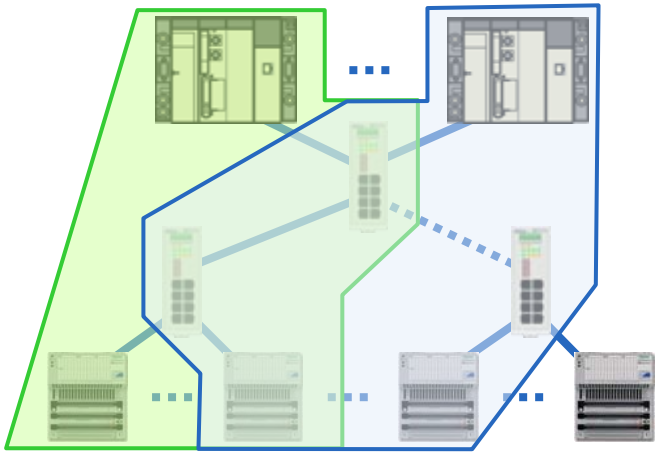


Controllers (PLCs) → clients

Switched industrial Ethernet network (Modbus-TCP)

Remote I/O modules → data servers

Considered class of NAS



Controllers (PLCs) → clients

Switched industrial Ethernet network (Modbus-TCP)

Remote I/O modules → data servers

Main features

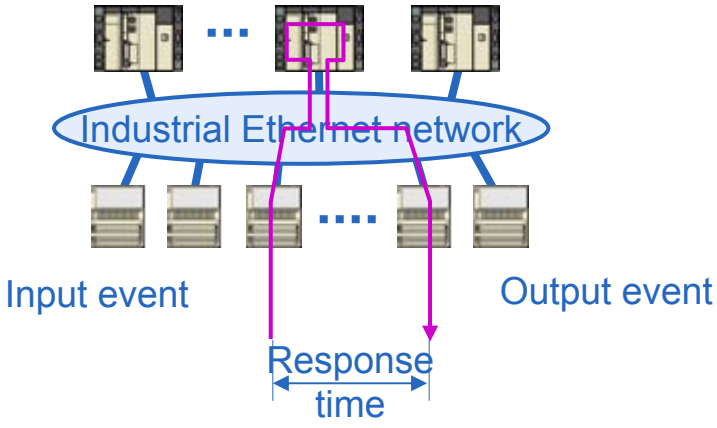
- Each PLC scans cyclically several RIOMs; PLCs scans are not synchronized.
- One RIOM may be scanned by several PLCs ⇒ **PLCs scans are concurrent processes.**

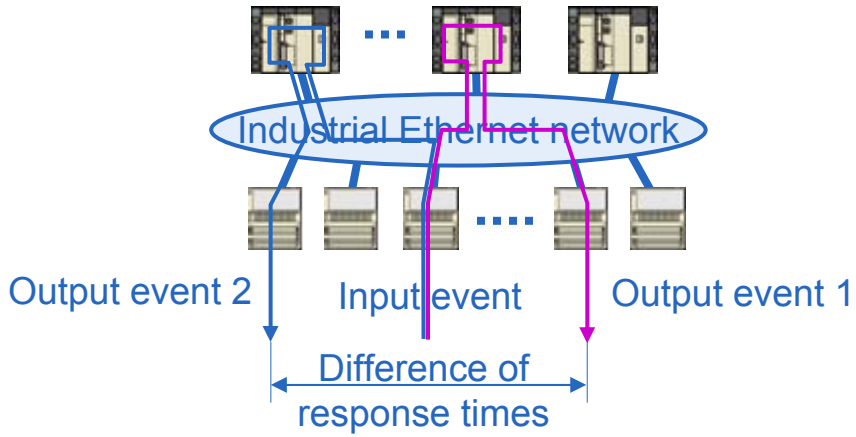
Assumption

- No frame loss (full duplex switched Ethernet, large enough switches buffers, no perturbations due to electromagnetic fields, …)

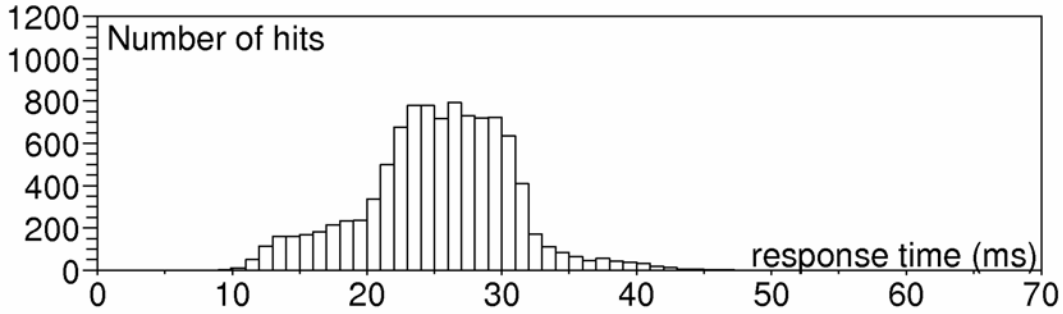
Definitions and measurement

- **Response time and difference of response times**

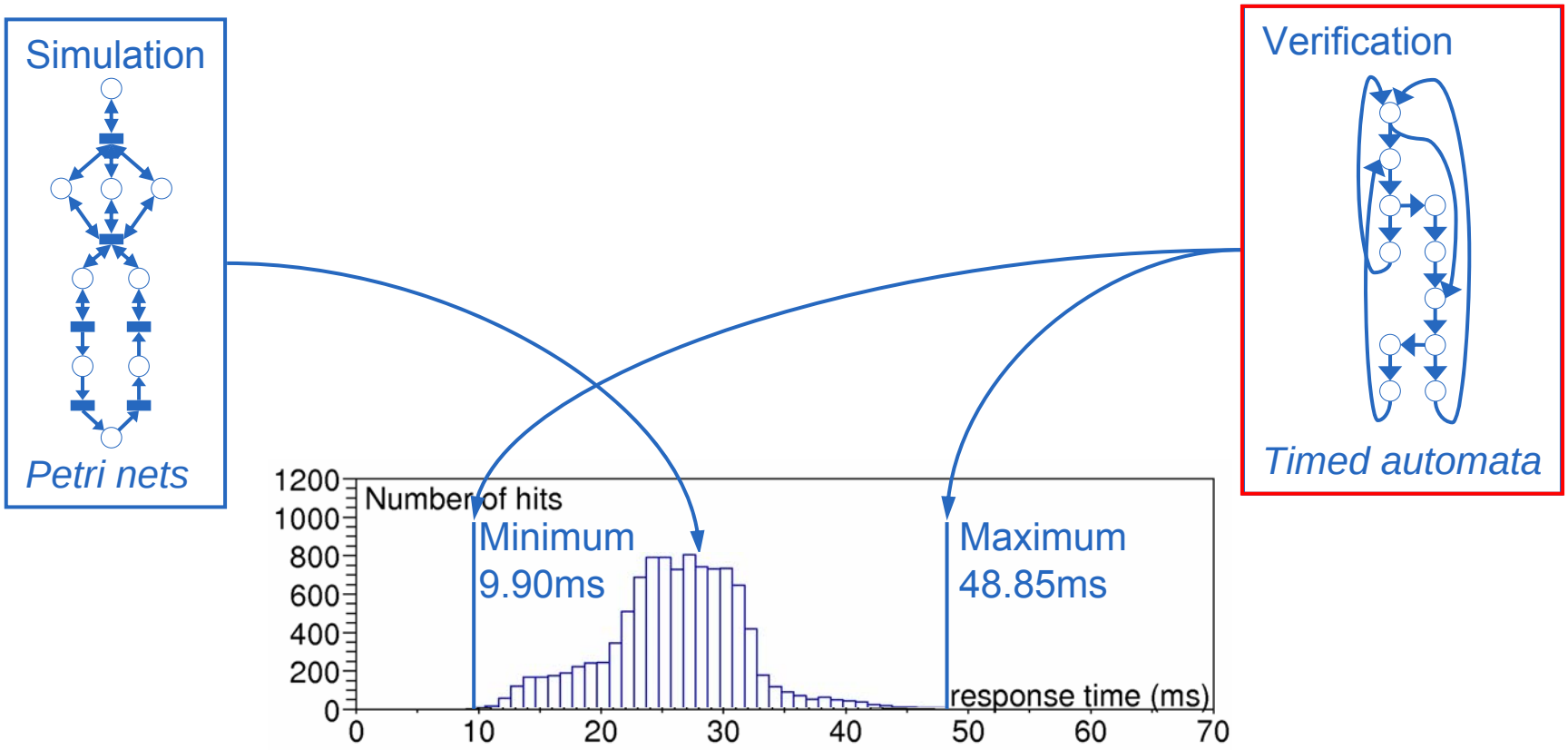




- **Experimental results: distribution of values**

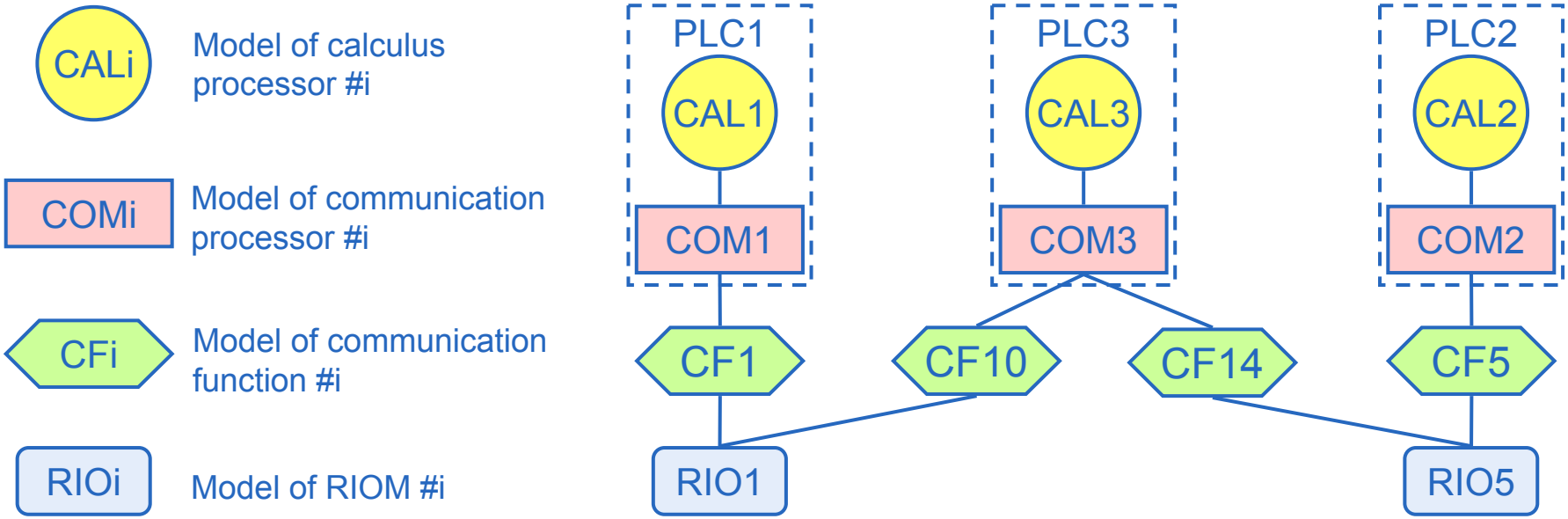


Off-line time performances evaluation using DES models



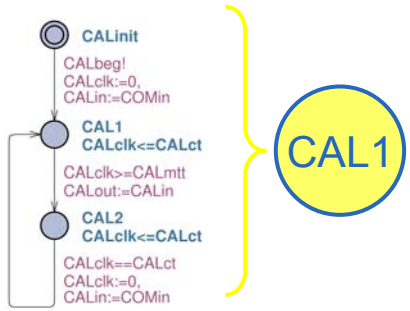
Construction of the model to check

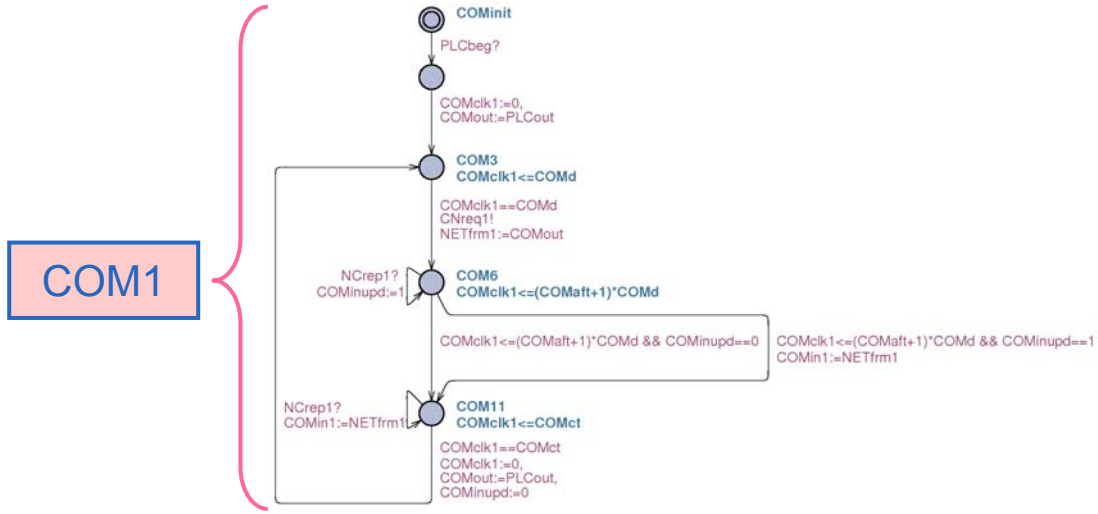
- **Structure of the NAS model: graph**
  - Nodes → components models
  - Edges → communications between components models

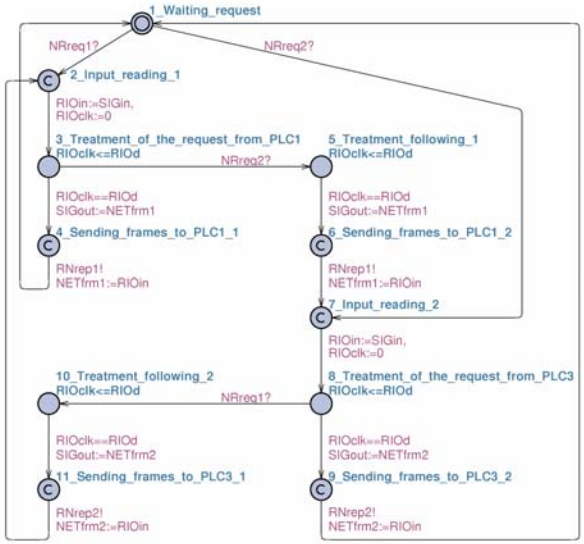


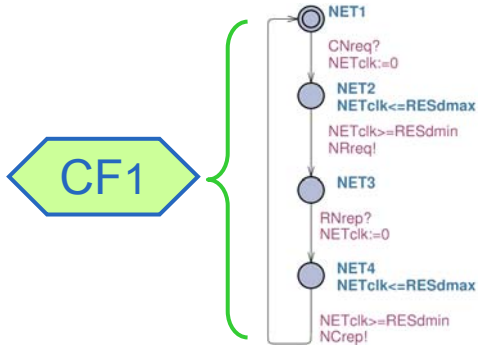
Construction of the NAS model to check (continued)

- **Structure of the NAS model: graph**
- **Components models: timed automata**





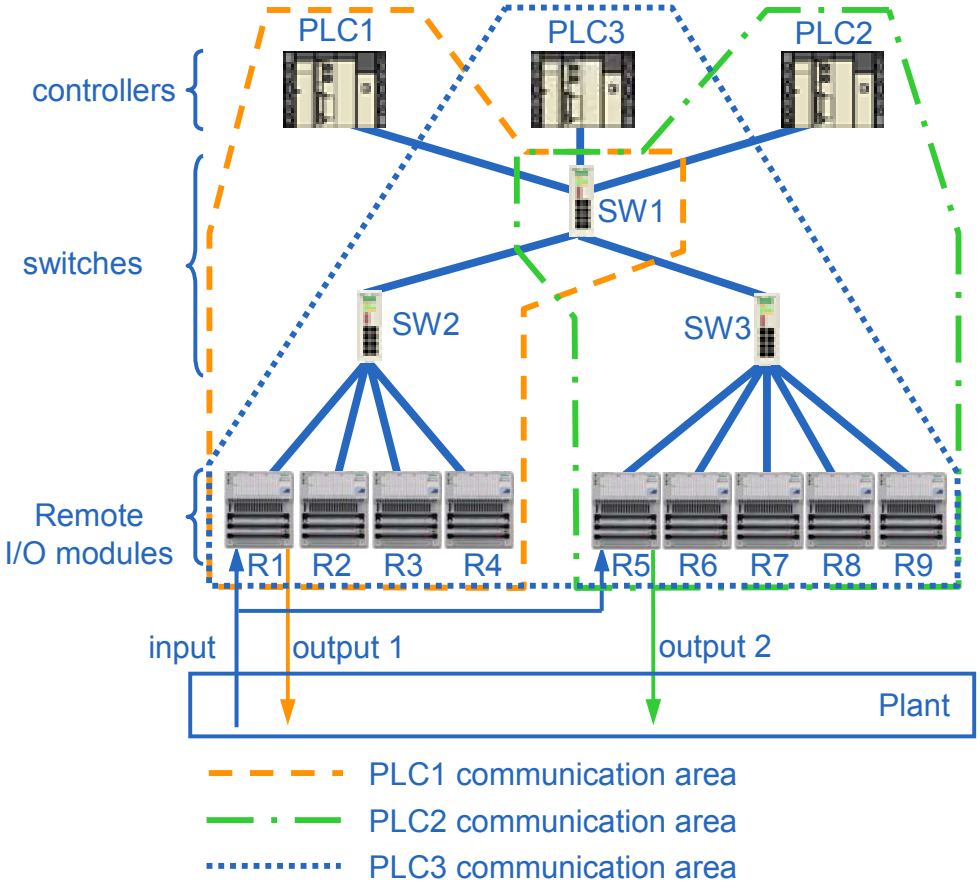




Number of clocks = Nplc*2 + Ncf + Nrio

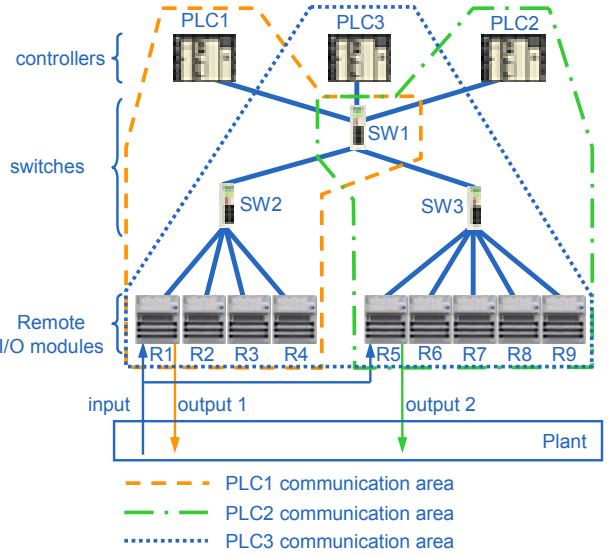
Testing scalability of the approach

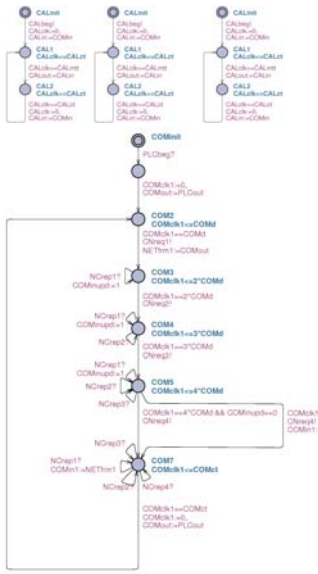
- **Small example**

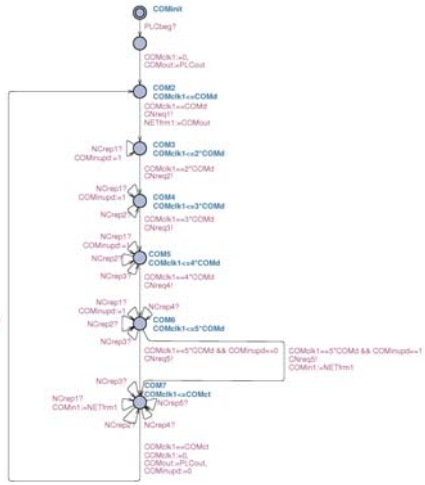


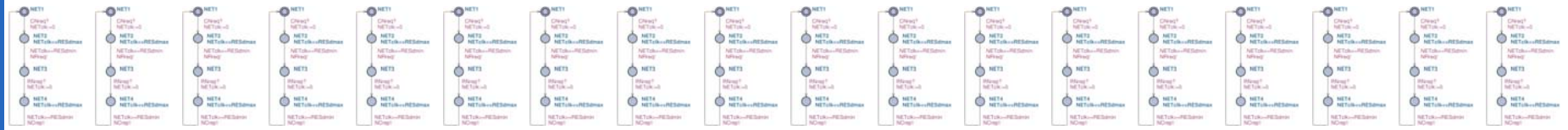
Testing scalability of the approach

- **Small example** → **Out of memory in few minutes**







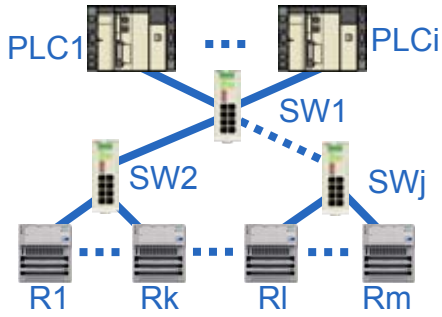


A method to build effective formal models is required

- **This method must yield abstract models that are tractable by existing model-checkers.**

- **Proof results on these models must be trustworthy.**

Method overview



Informal description of the system

Construction of the detailed model of the system

Detailed model of the system

M components models
Detailed components models

Simplification of the structure

Property to prove

Simplified model of the system

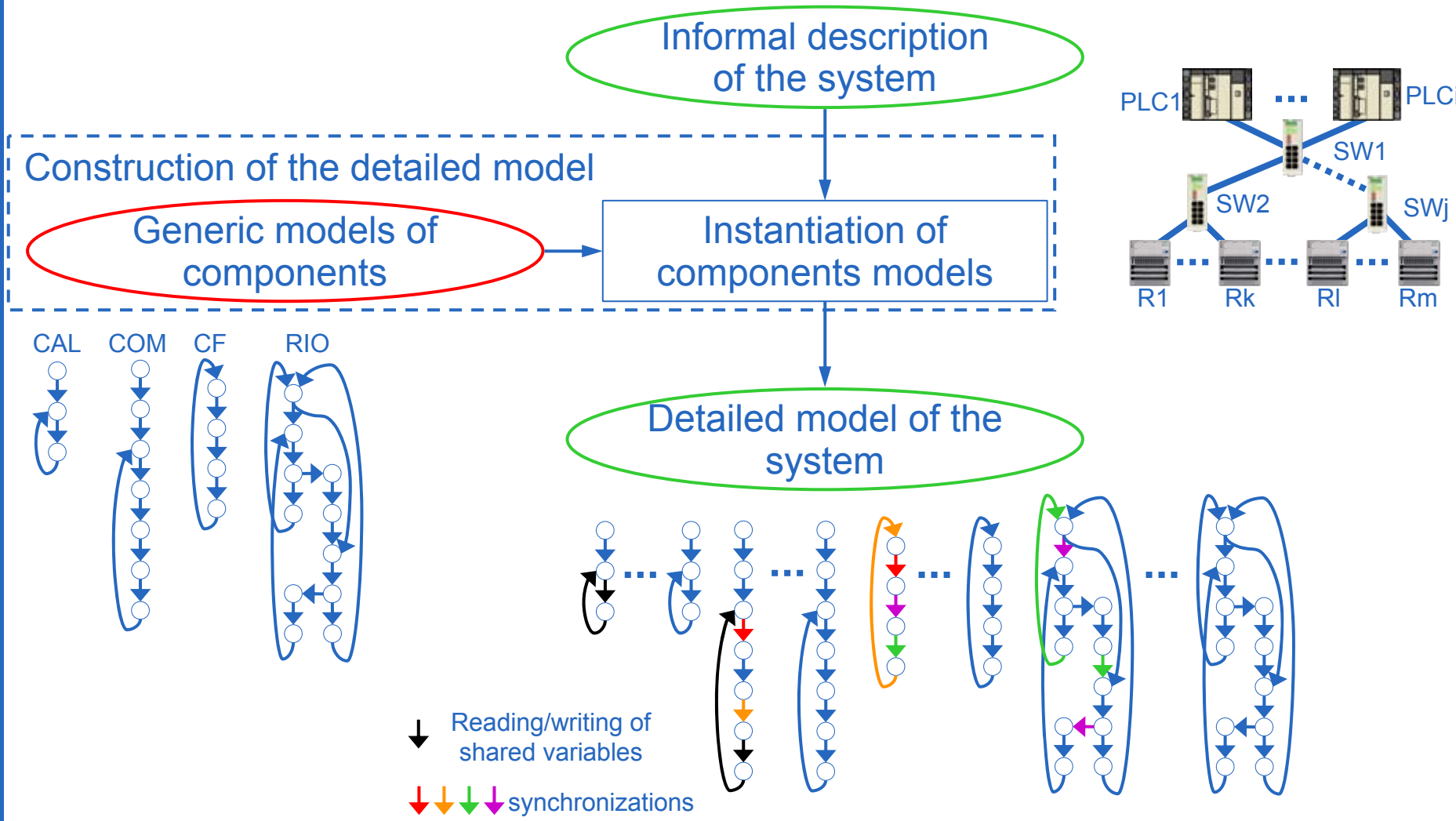
Only the N necessary components models (N≤M)
Detailed components models

Modification of the components models

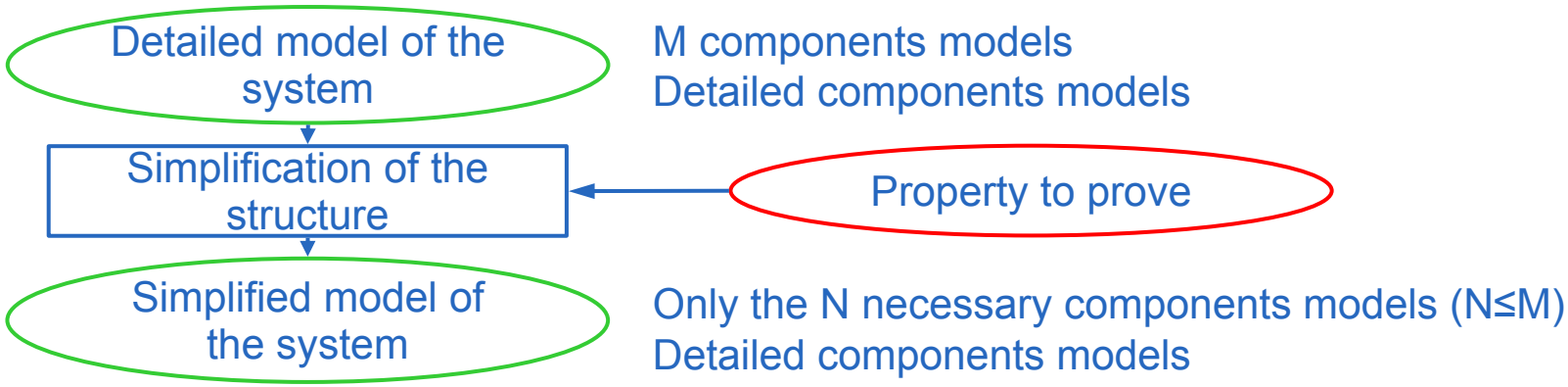
Reduced model of the system

N components models
Modified components models

Step 1: construction of the detailed model

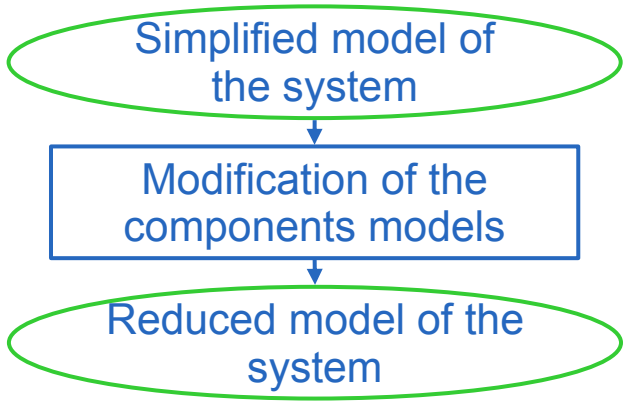


Step 2: simplification of the structure



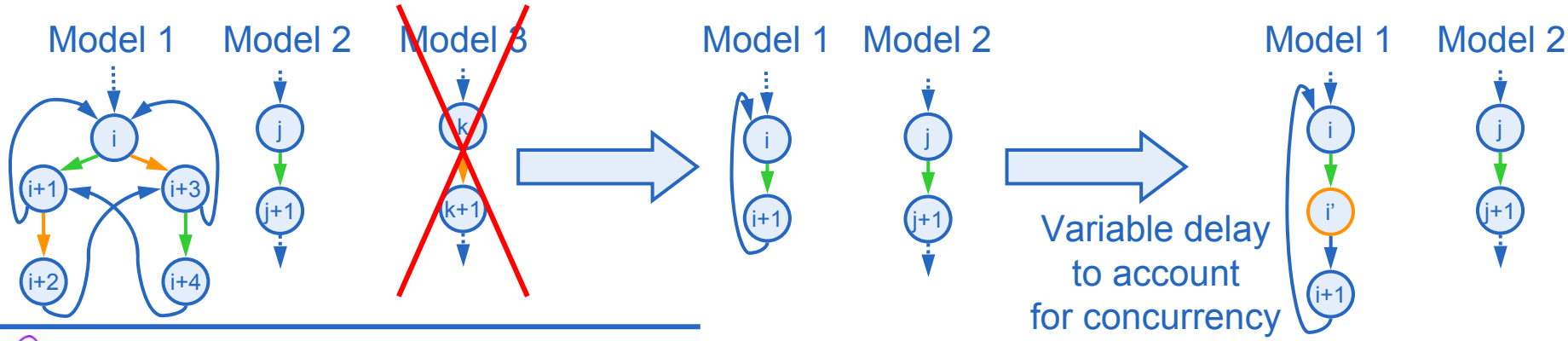
- **Principle**
  - Keep only the components models which generate, modify or propagate data that depend on the input or output events
  - **Interpretation abstraction** similar to 'cone of influence' in symbolic model-checking, or 'localization reduction' for integrated circuits verification
- **Step automation: analysis of the structure of the model (graph)**
  - Search of the **shortest path** from the considered input to the considered output that goes through the calculus processor that computes the value of the output event

Step 3: modification of the components models



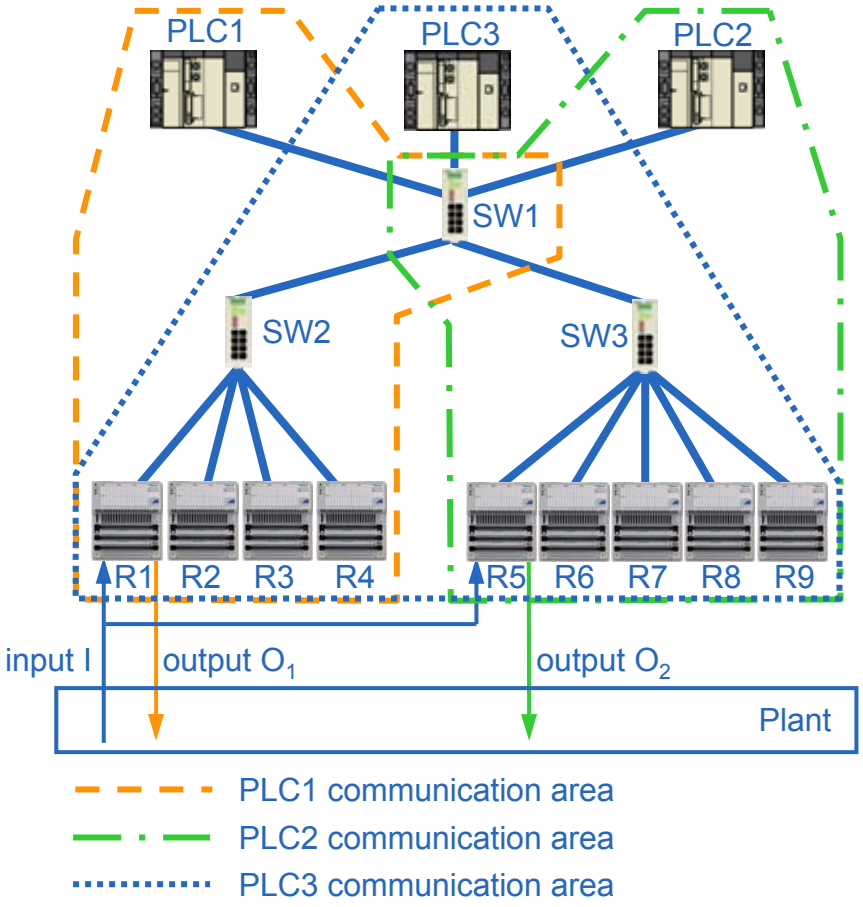
- **Consequence of the previous step**
  - Loss of possible behaviors

  in remaining components models
- **In all remaining components models**
  - Remove useless transitions and locations
  - Insert variable duration (from 0 to Worst Case Waiting Time) locations to account for concurrency

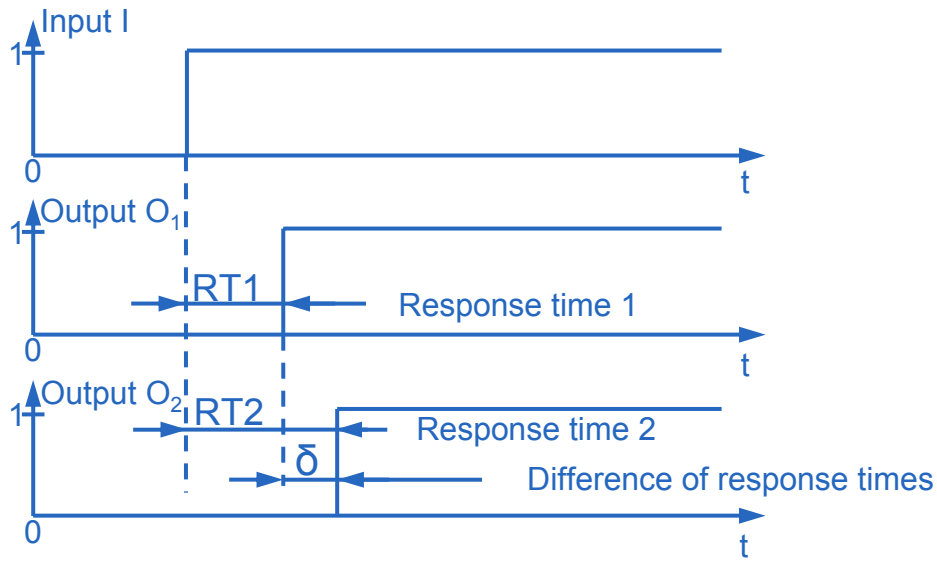


Objective of the study

- **Studied system**



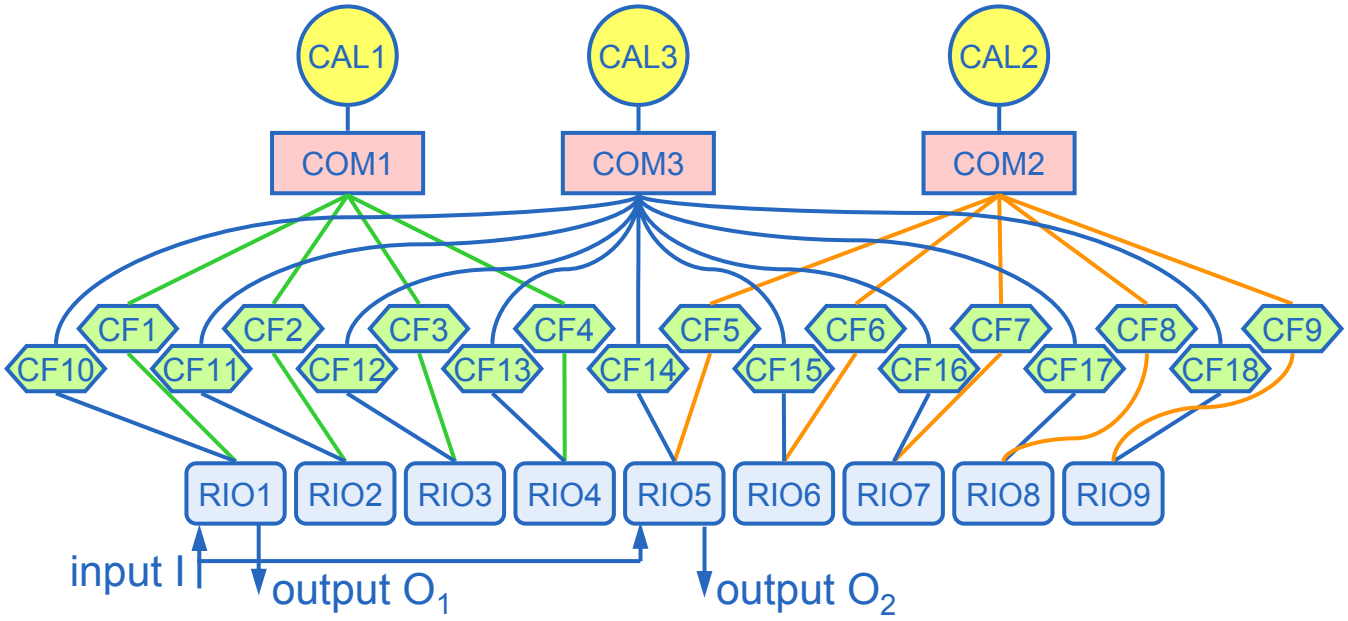
- **Studied time performance**



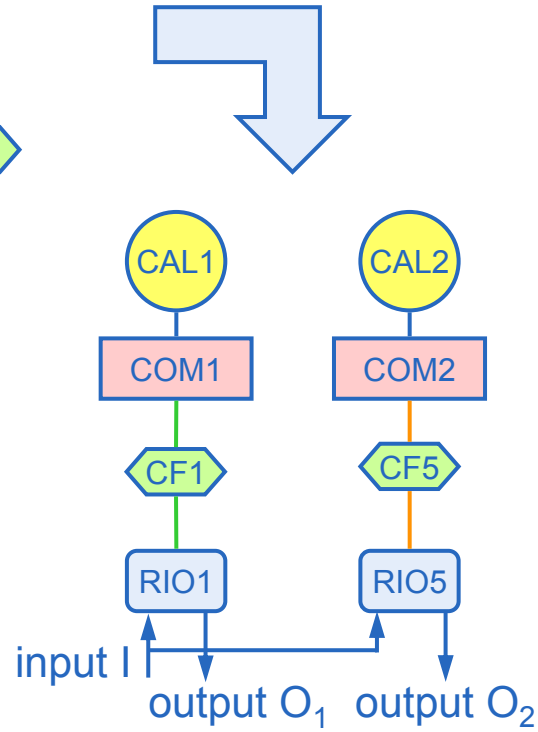
Upper bound of δ (Max(δ))?

Simplification of the NAS model structure

- **Initial and final system models**



Graph analysis

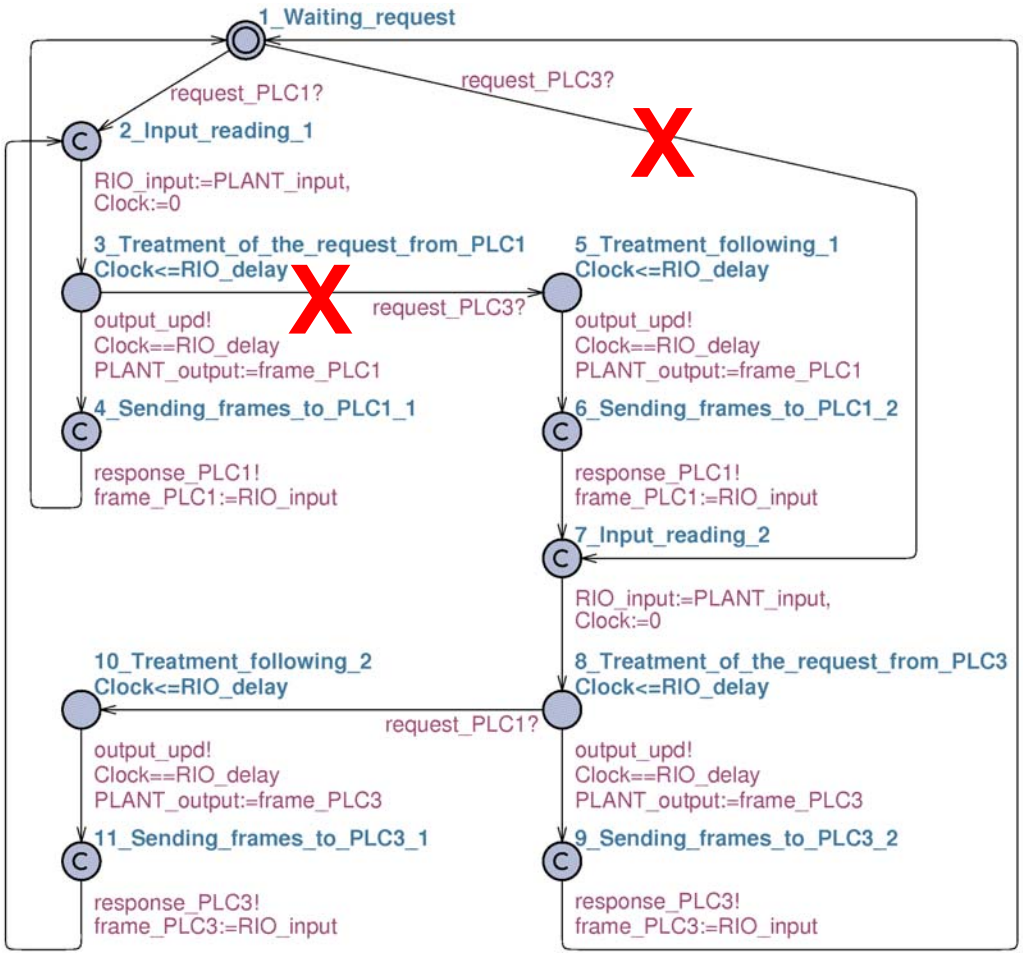


Possibility to determine separately the lower and upper bounds of RT1 and RT2 (RT1m, RT1M, RT2m, RT2M)

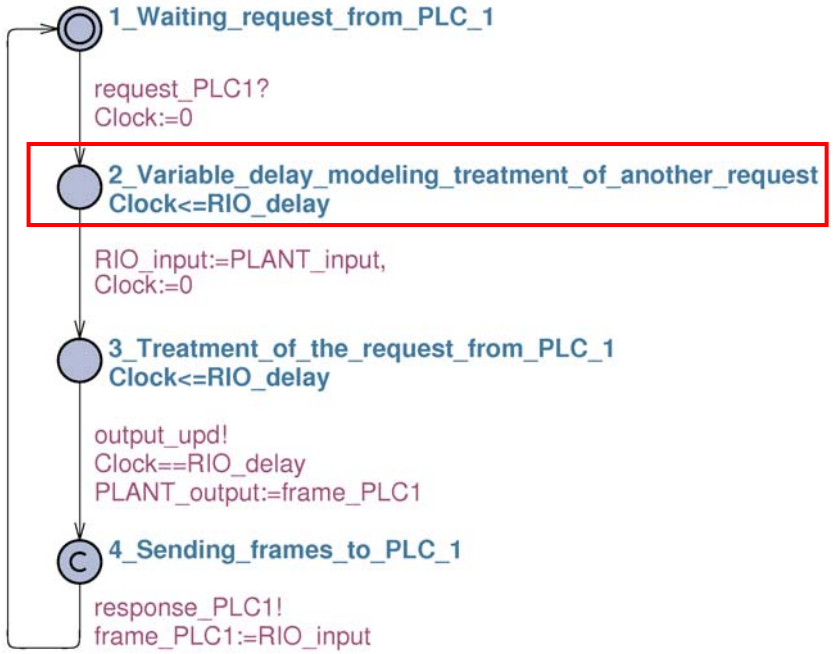
→ $Max(\delta) \leq MdRT = Max((RT1M-RT2m);(RT2M-RT1m))$

Modification of the components models (RIO model)

- **Detailed model**



- **Modified model**
  - Locations 1, 3 and 4 unchanged
  - New location (variable duration) added to account for PLC3 requests



Obtained results

	Experiment 1	Experiment 2	Experiment 3
Model	Detailed	Reduced	Reduced
Performance	Max(δ)	Max(δ)	MdRT
Calculus duration	Impossible (not enough memory)	28 hours	1 second
Obtained values		Max(δ) = 21.4 ms	MdRT = 21.4 ms → Max(δ) ≤ 21.4 ms

- **Comparison**
  - Experiment 3 leads to a very short calculus duration
  - Experiment 3 gives an overestimation of the upper bound of the difference of response times
  - Experiment 2 gives the upper bound of the difference of response times

Conclusions

- **Time performances evaluation of real systems using model-checking requires to “pre-process” detailed models**

- **Modeling method to build abstract formal models of networked automation systems based on:**
  - Simplification of the structure of the system model
  - Modification of the components models

- **Formal models obtained**
  - Are tractable by existing proof tools
  - Proofs on these models are meaningful and trustworthy

Outlooks

- **Technical improvements**
  - Automation of the different steps of the method
    - Set up of a components models library to automate step 1
    - Automatic modification of components models (step 3) from shortest path search results (results of step 2)
  - Larger case studies to assess the limits

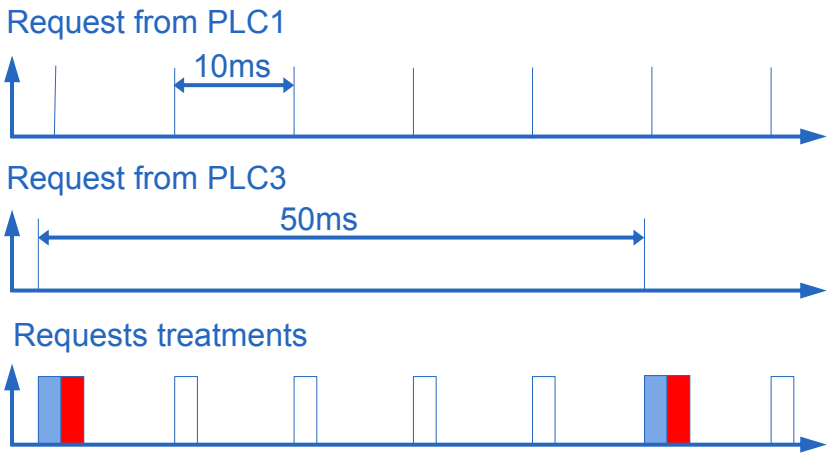
- **Further investigations**
  - More complex models that account for other communications (data exchange between PLCs, between PLCs and upper levels (SCADA, maintenance, production management systems, …))
  - Parametric model-checking so as to find sets of parameters of NAS that guarantee specified time performances bounds

Thank you for attention.

Questions ?

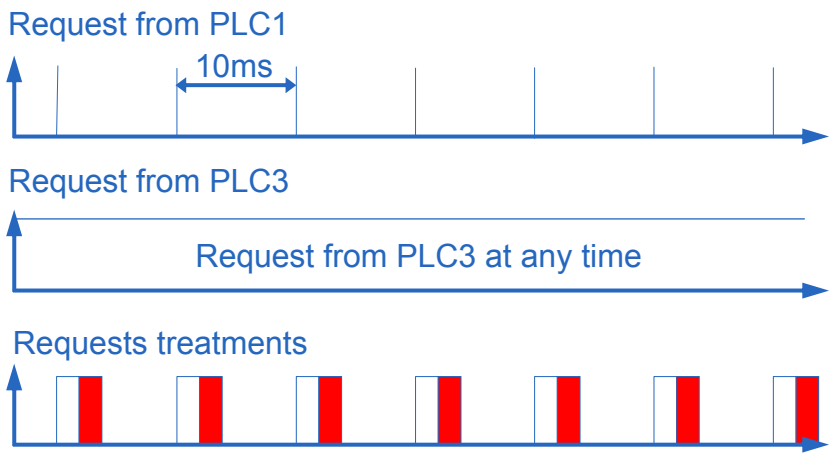
Behavior evolution

- **Initial behavior**



Treatments of requests from PLC3
Non delayed treatments of requests from PLC1
Delayed treatments of requests from PLC1

- **Abstract behavior**



Non delayed treatments of requests from PLC1 (at soonest)
Delayed treatments of requests from PLC1 (at latest)

- **With the abstract model, all requests might be delayed or not**
  - → Adding unexpected behaviors
  - → Worst case model