Practical and Dependable Control Synthesis for Programmable Logic Controllers

Janan Zaytoon
Bernard Riera

CReSTIC
University of Reims Champagne-Ardenne

Université de Reims Champagne-Ardenne

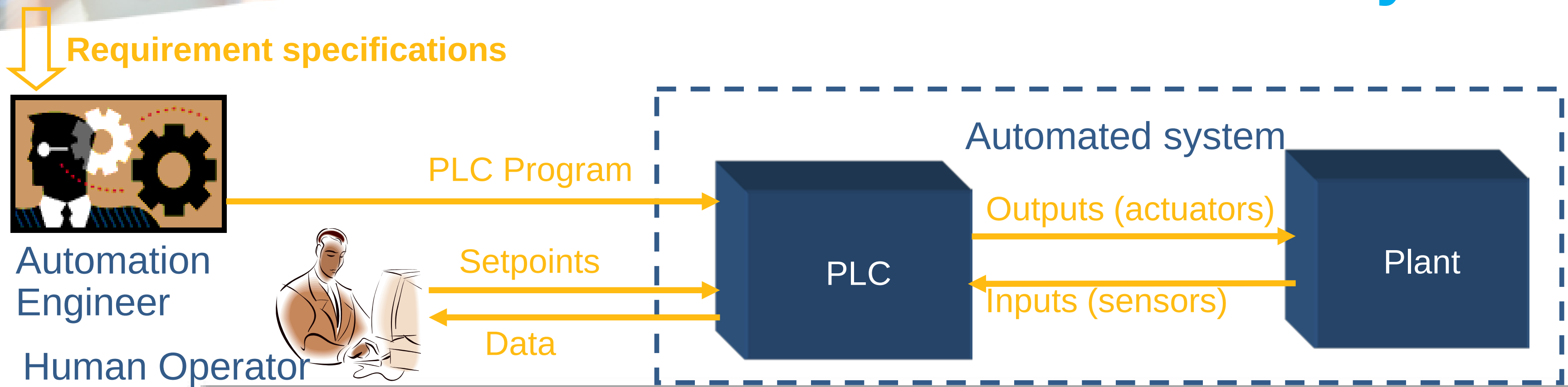
Outline

- Introduction and Context
- Models and Activities for Logic Control Design and Implementation
- Supervisory control theory (SCT)
- SCT based approaches
- Non SCT based approaches
- Conclusion

Introduction and Context

- Programmable Logic Controllers (PLCs) are widely used in a very large number of systems since the 1970s
- Control engineers classically interpret the informal specifications to implement the control tasks with the help of standardized tools for programming of PLCs
- Growing complexity, demand for reduced development time & criticality of control problems → formal verification & design methods to guarantee & reinforce the requirement specifications
- Synthesis approaches aim at generating a controller that satisfy the required specifications by construction, with very little involvement of the designer

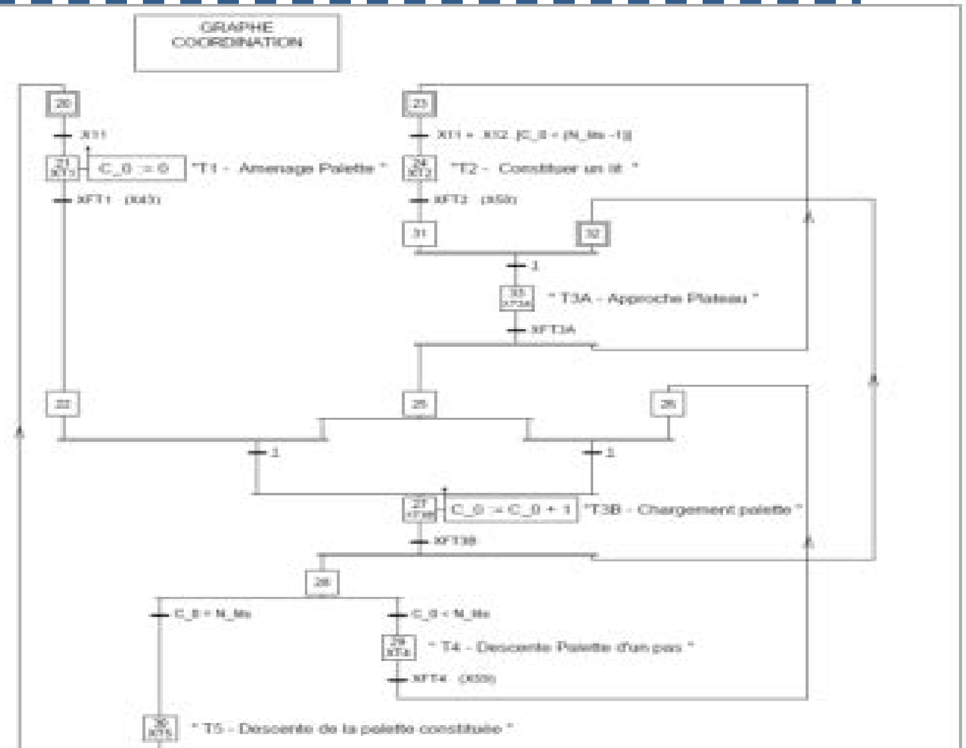
Introduction and Context: Automation Systems



Control is classically specified using GRAFCET/SFC (IEC 60848)

High expressive power but Safety is not always formally checked

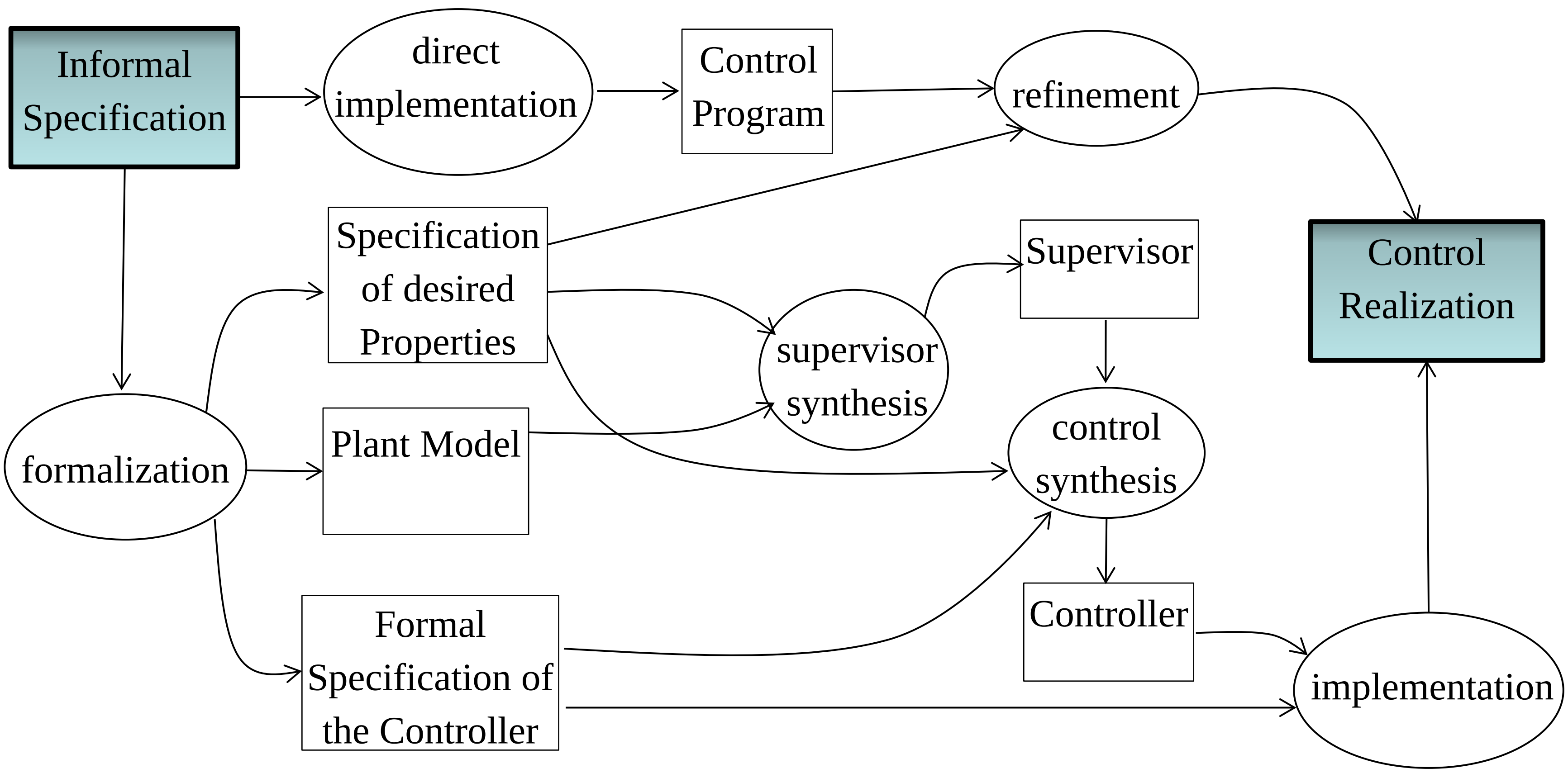
Resistance to change the control design methods used in industry



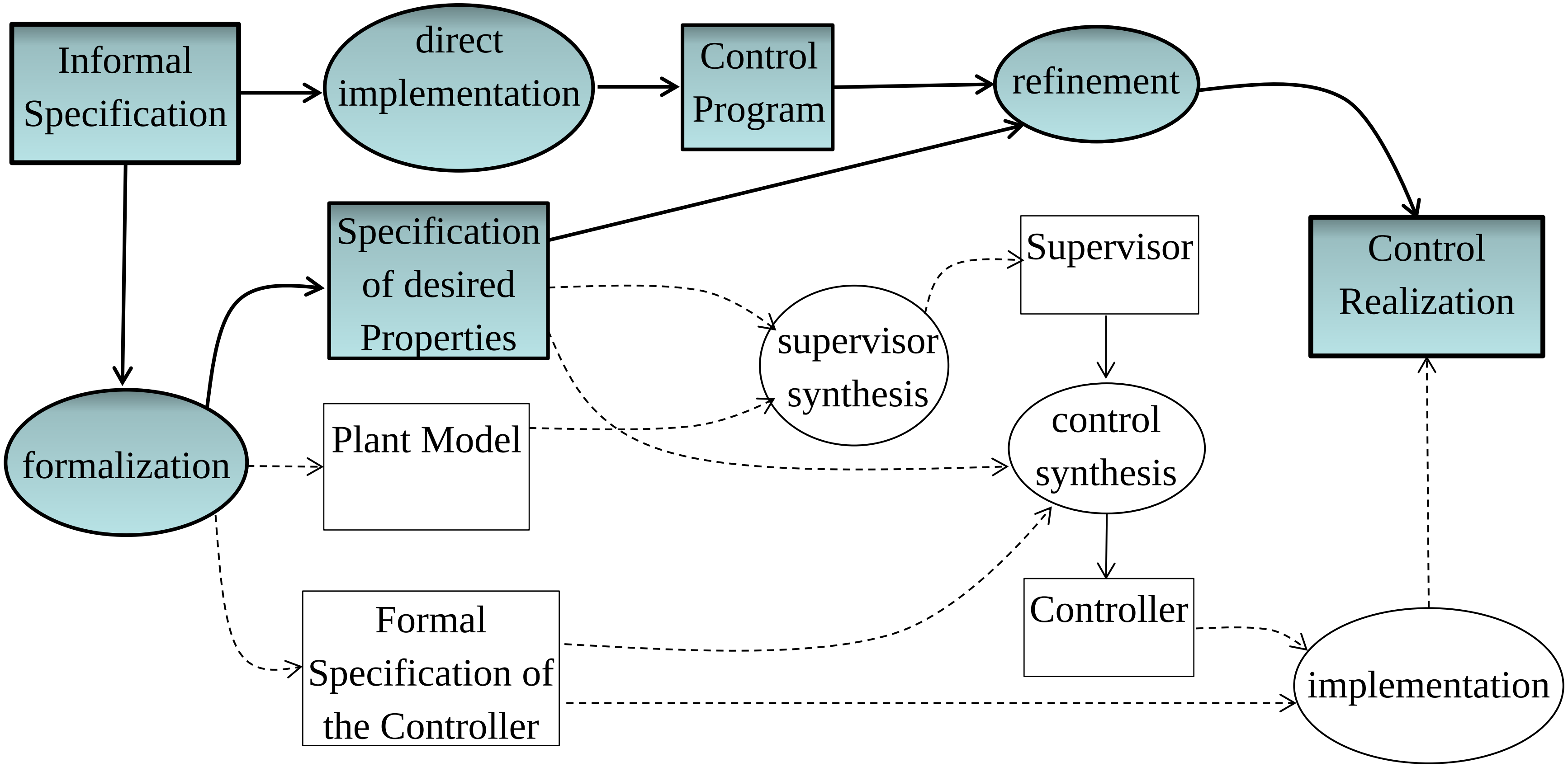
Outline

- Introduction and Context
- Models and Activities for Logic Control Design and Implementation
- Supervisory control theory (SCT)
- SCT based approaches
- Non SCT based approaches
- Conclusion

Models and Activities for Control Design and Implementation

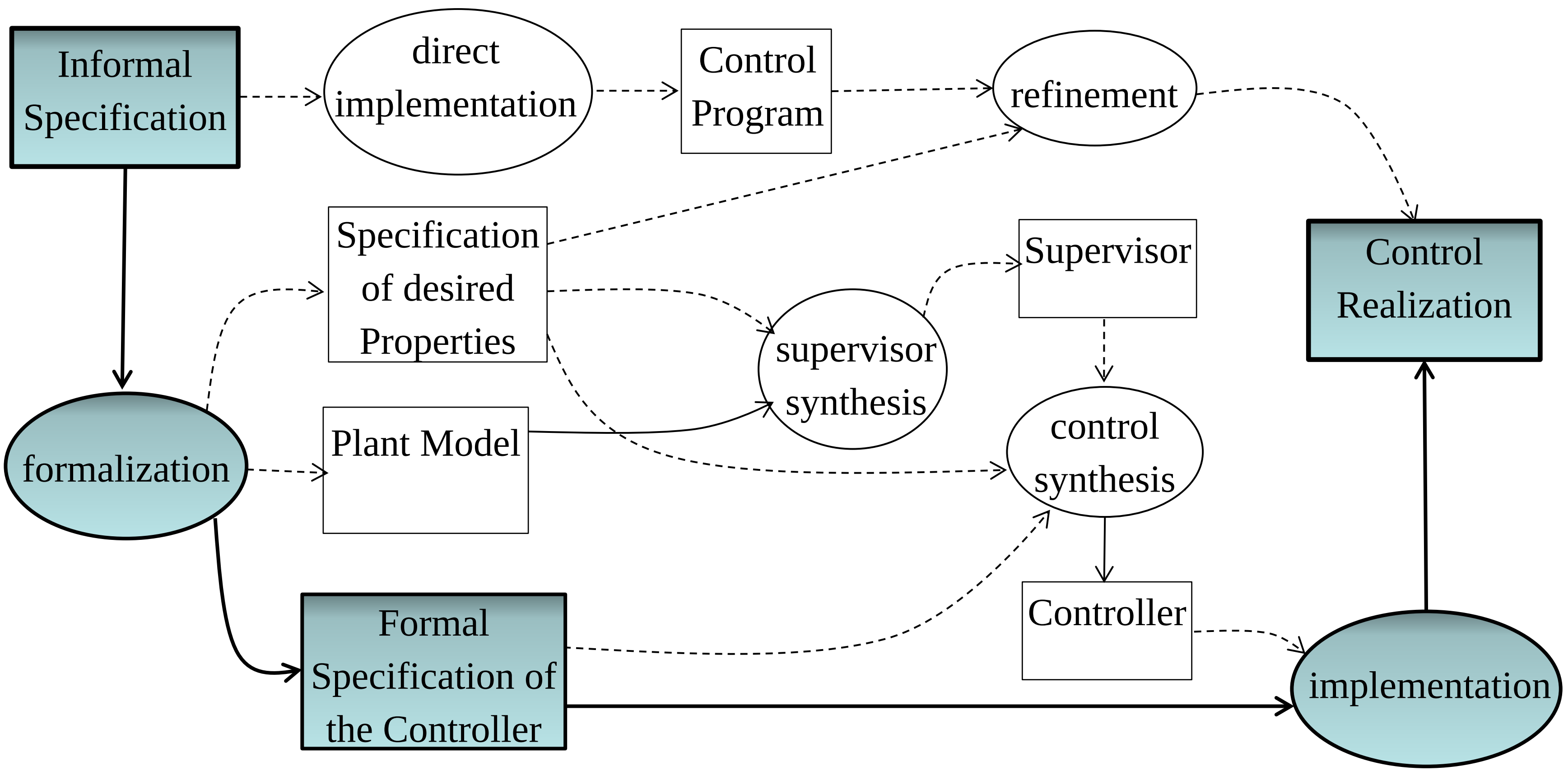


Models and Activities for Control Design and Implementation

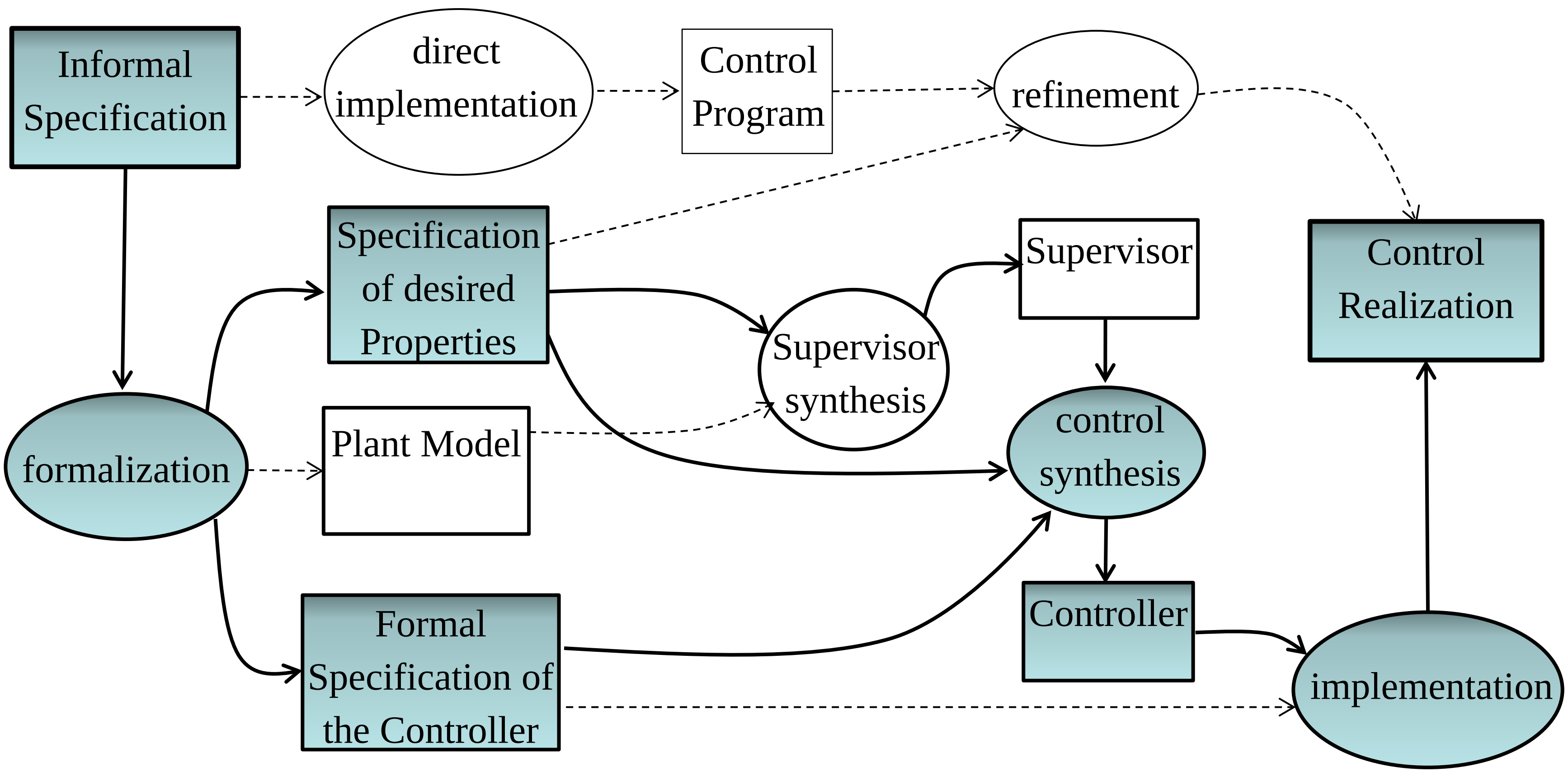


Standard industrial approach

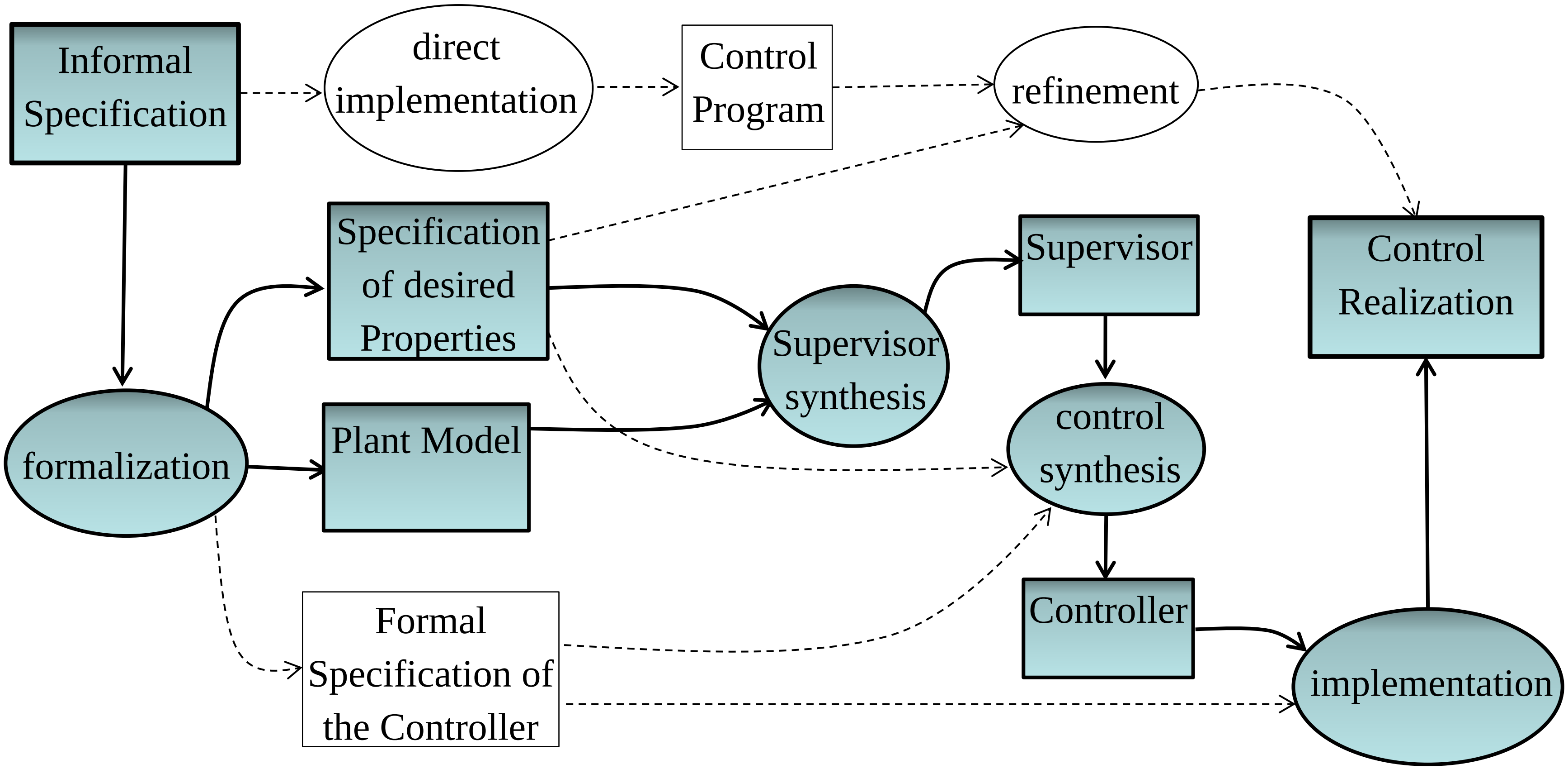
Models and Activities for Control Design and Implementation



Models and Activities for Control Design and Implementation



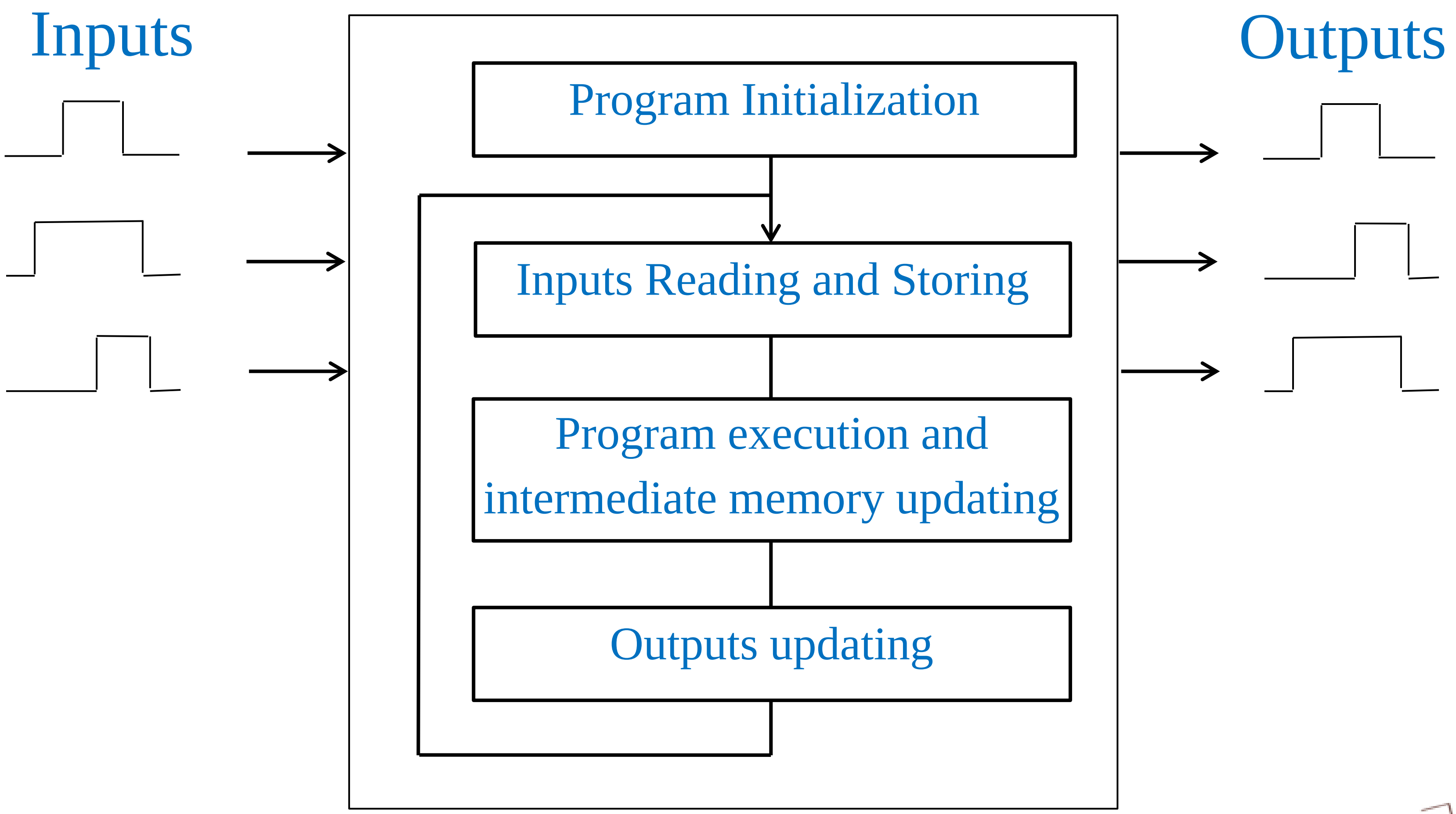
Models and Activities for Control Design and Implementation



Problems and Challenges

- Formalization: unsuitability of classical control-based specifications to identify plant reactions ⇒ how to obtain meaningful models of the plant (abstraction level, complexity, modularity, genericity) and the desired properties?
- Synthesis: Complexity, readability of the result!
- Implementation: code is generated in a standardized PLC Language/architecture, semantically incompatible with the used models

Control Realization: PLC Cycle

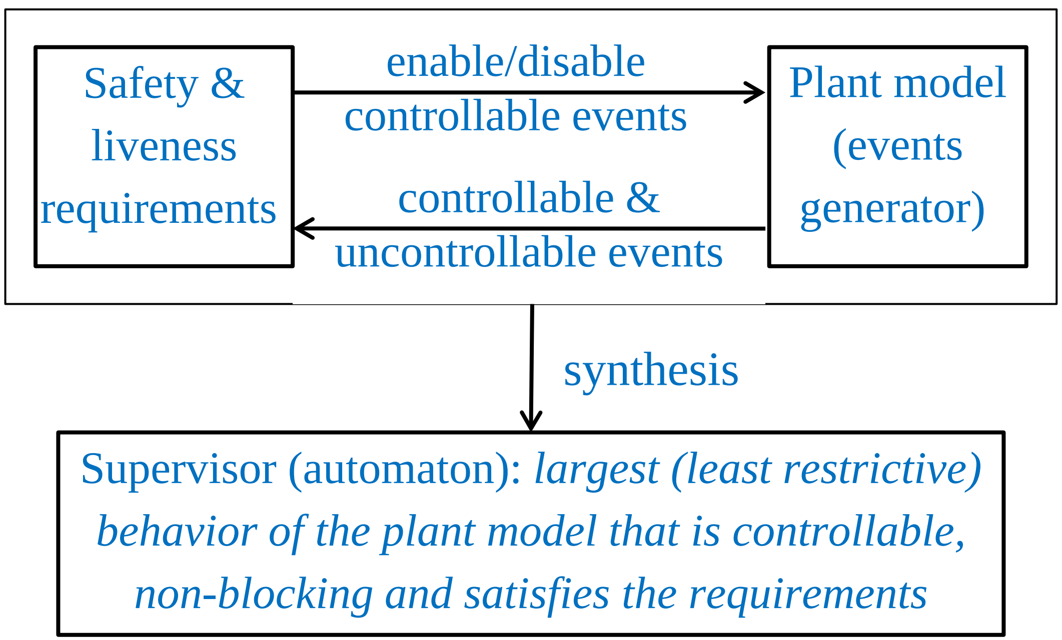


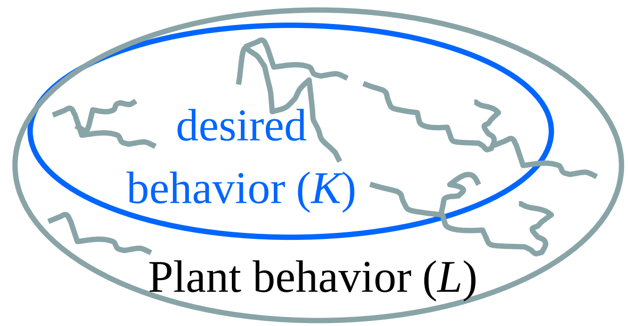
Outline

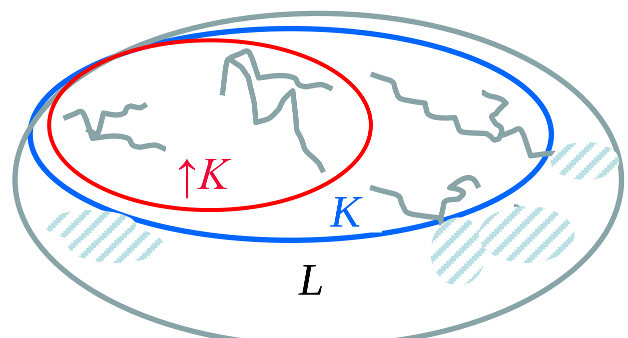
- Introduction and Context
- Models and Activities for Logic Control Design and Implementation
- Supervisory control theory (SCT)
- SCT based approaches
- Non SCT based approaches
- Conclusion

Supervisory control theory (Ramadge et Wonham, 87, 89)

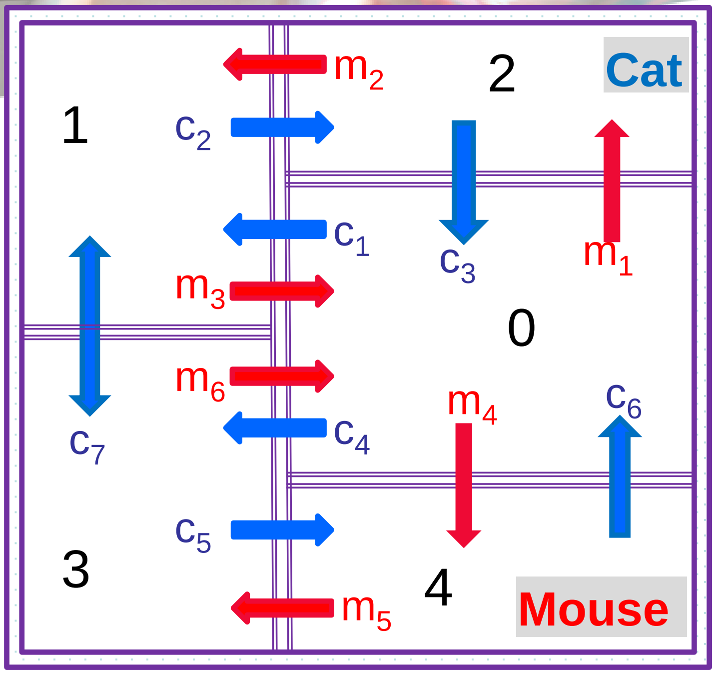
Apply control theoretic concepts to Discrete-Event Systems (separate open loop dynamics from f/b control, controllability, observability, …) to provide solutions for a variety of control synthesis problems using automata and formal languages

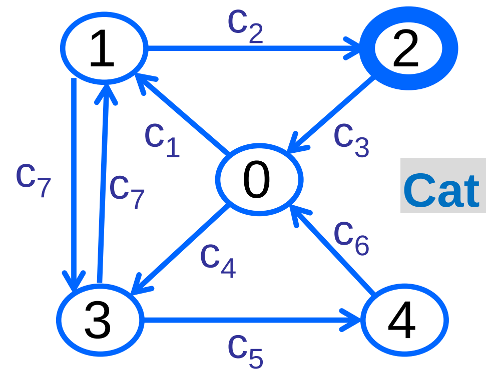


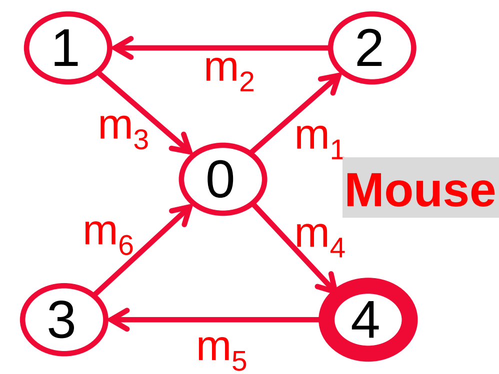




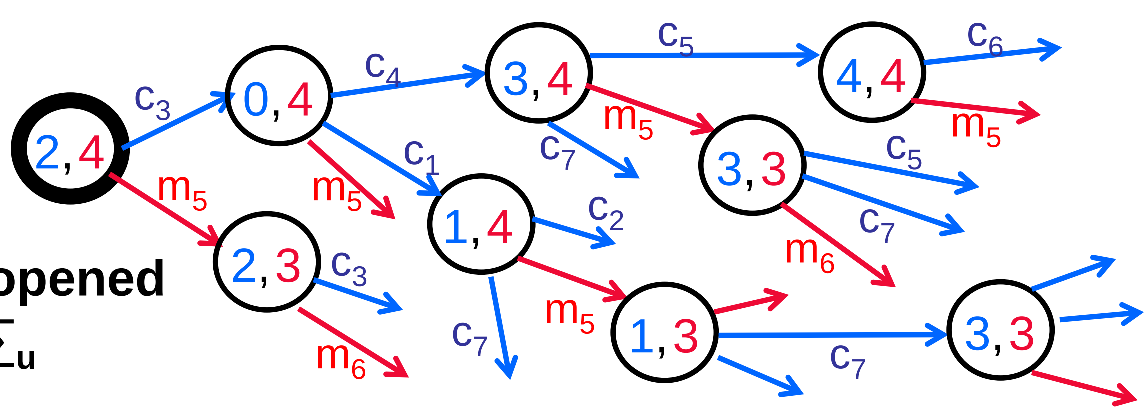
Supervisory control theory
Example: Cat & Mouse Maze







G = Cat || Mouse ; 25 states and 70 transitions



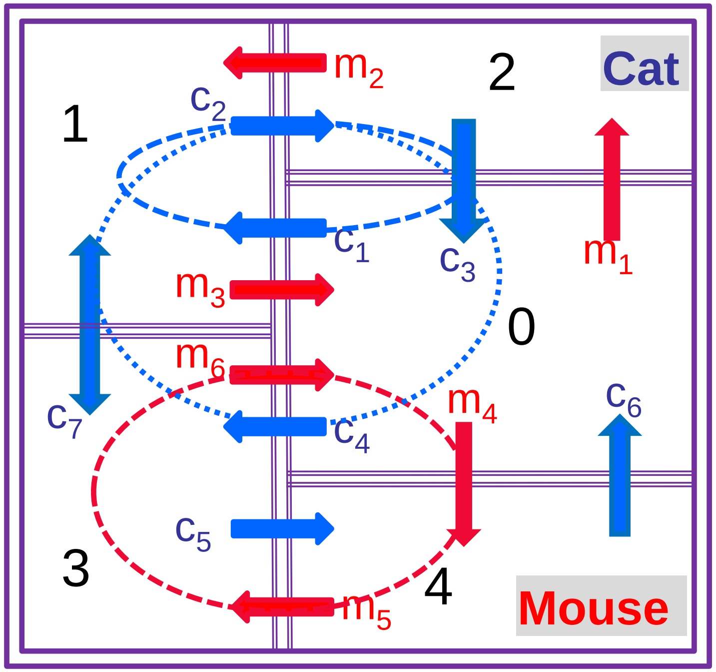
Each door, except c_7, can be opened or closed: $\sum_u = \{c_7\}$, $\sum_c = \sum - \sum_u$

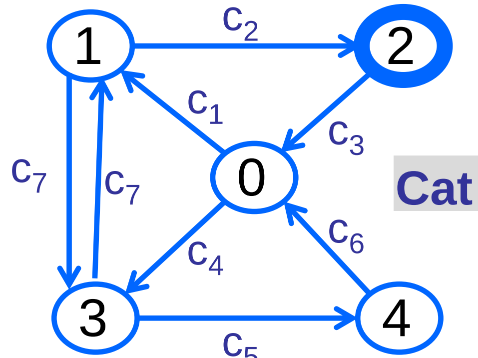
design the controller providing the greatest possible freedom of movement and guarantying that:

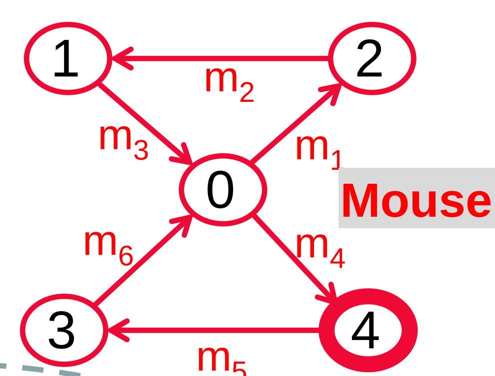
i) the cat & the mouse never occupy the same room simultaneously;

ii) each of the cat & the mouse can return to its initial room

Supervisory control theory
Example: Cat & Mouse Maze



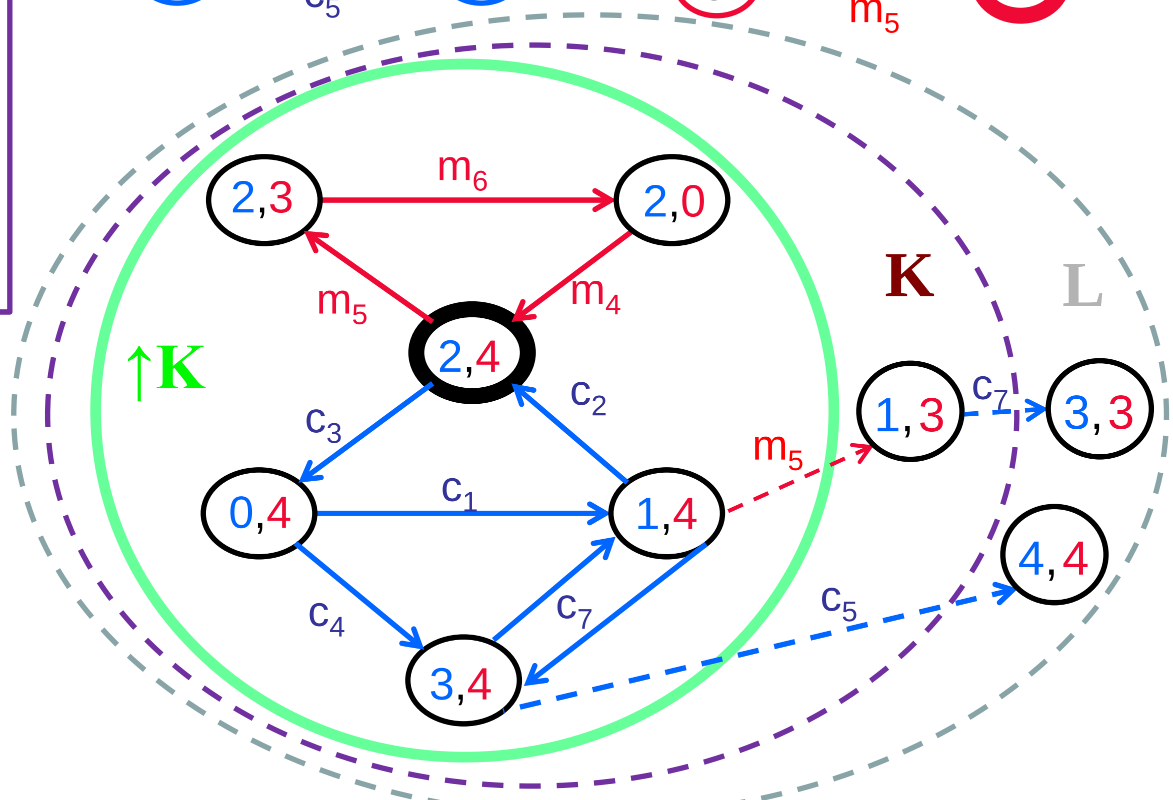




L: 25 states & 70 transitions

K: 16 states & 38 transitions

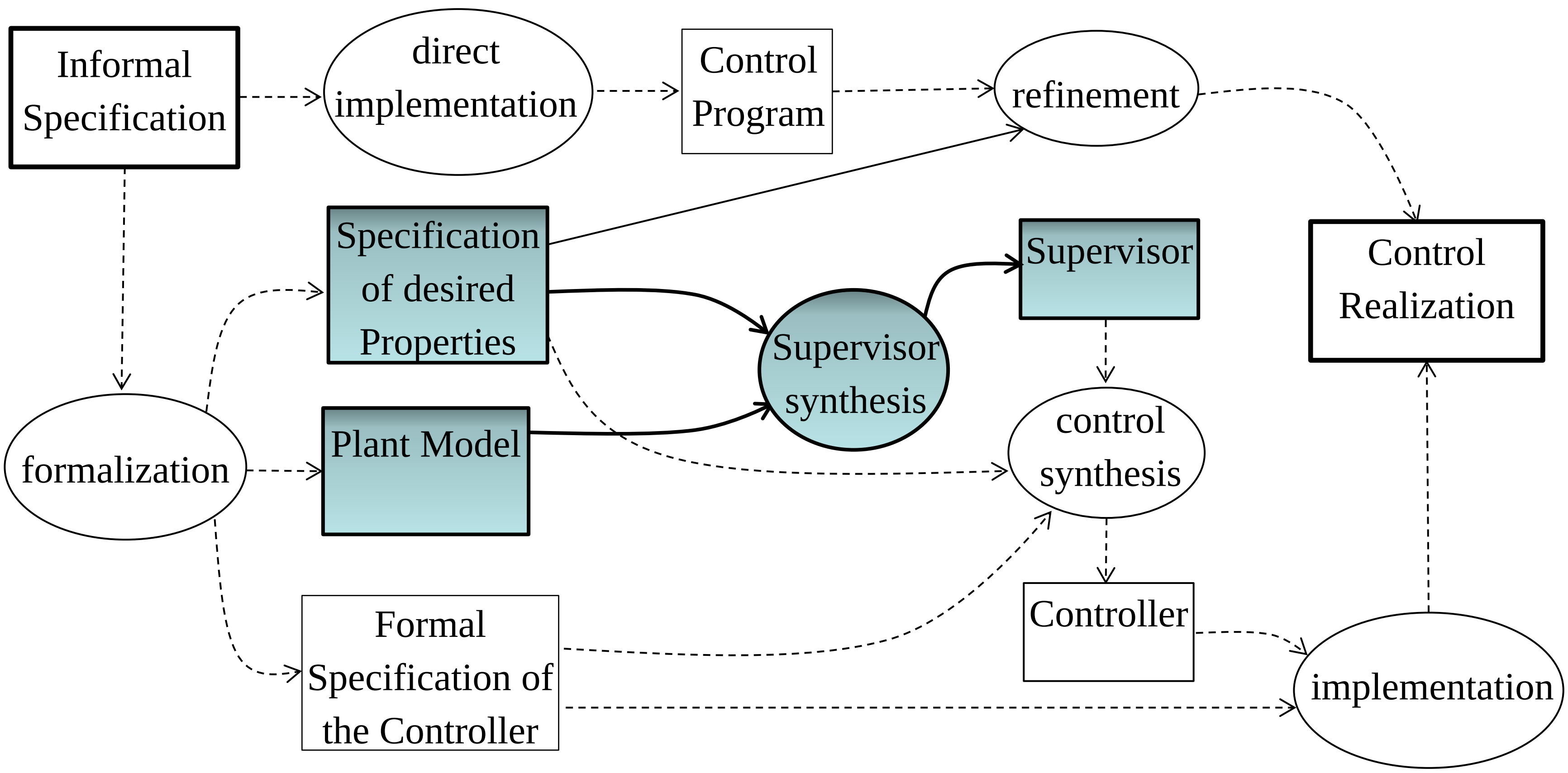
↑K: 6 states & 9 transitions



Supervisory control theory: Advantages

- Solid formal background, with well-established theoretic results dealing with different problems & settings
- Some few successful applications of the resulting supervisors, but it is not clear how to generalize the results
- Provides a systematic approach to supervisory control design, but does not cover the overall design process to obtain the control realization

Supervisory control theory (Ramadge et Wonham, 87, 89)



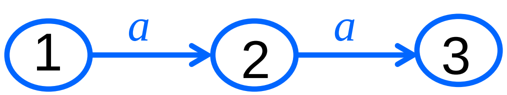
Supervisory control theory: Control implementation problems

- How to obtain suitable plant & requirements models, knowing that industrial practice is primarily concerned with control-based rather than plant-based specifications?
  → Proposal of high-level specification models & suitable methods, but how to adapt them to SCT semantics?
- Calculation complexity due to combinatorial explosion
  → Modularity, decentralization & hierarchy, but these structures may be incompatible with the natural modularity or hierarchy of the control system

Supervisory control theory: Control implementation problems

- behavioral discrepancy between SCT supervisors and the resulting PLC implementation:

  1- Asynchronous instantaneous events vs. persisting synchronously updated Binary signals ⇒ mapping events to signals, missing events, and avalanche effects



  2- Successive events occurring between 2 PLC cycles are considered by the PLC as simultaneous ⇒ loss of order of events occurring close together

Supervisory control theory: Control implementation problems

3- Causality and nature of control mechanism: enable/disable controllable events (what should not be done) vs. set/reset output signals (what to be done)

4- Determinism: How to choose the control action among the alternative paths provided by the supervisor

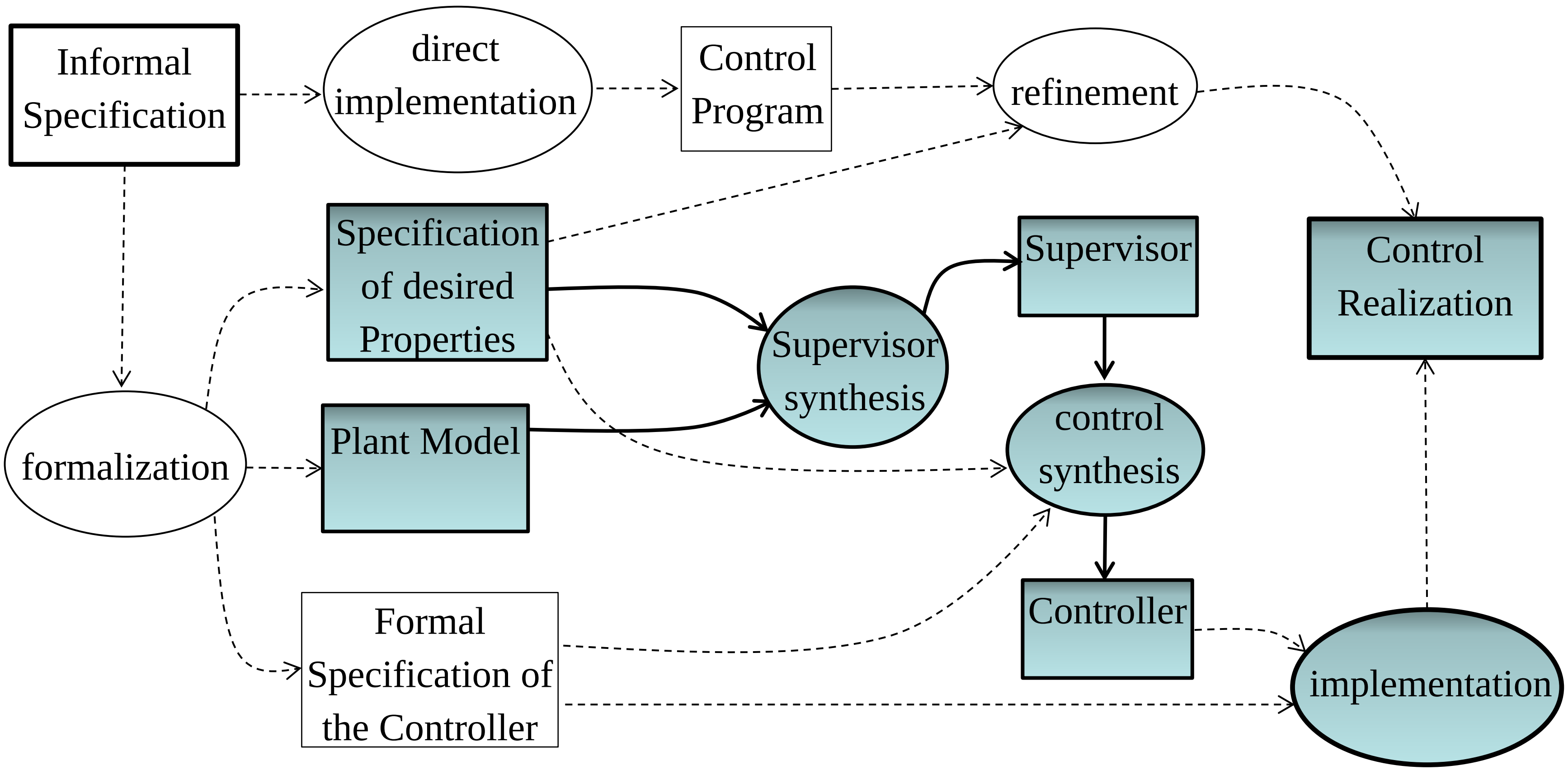
5- Inexact Synchronization: due to PLC scan cycle delay, the control logic is always performed on old frozen data

⇒ The proposed approaches either deal with only part of the above problems or are not adapted to modular & hierarchical supervision

Outline

- Introduction and Context
- Models and Activities for Logic Control Design and Implementation
- Supervisory control theory (SCT)
- SCT based approaches
  - refinement of the supervisor
  - introducing a specification model for the controller
- Non SCT based approaches
- Conclusion

SCT based approaches: refinement of the supervisor



SCT-based approaches: refinement of the supervisor

- Extracting a deterministic controller from the supervisor by selecting only one among the enabled controllable events:

  - a non-blocking supervisor may yield a blocking controller → Computation of a nonblocking safe controller

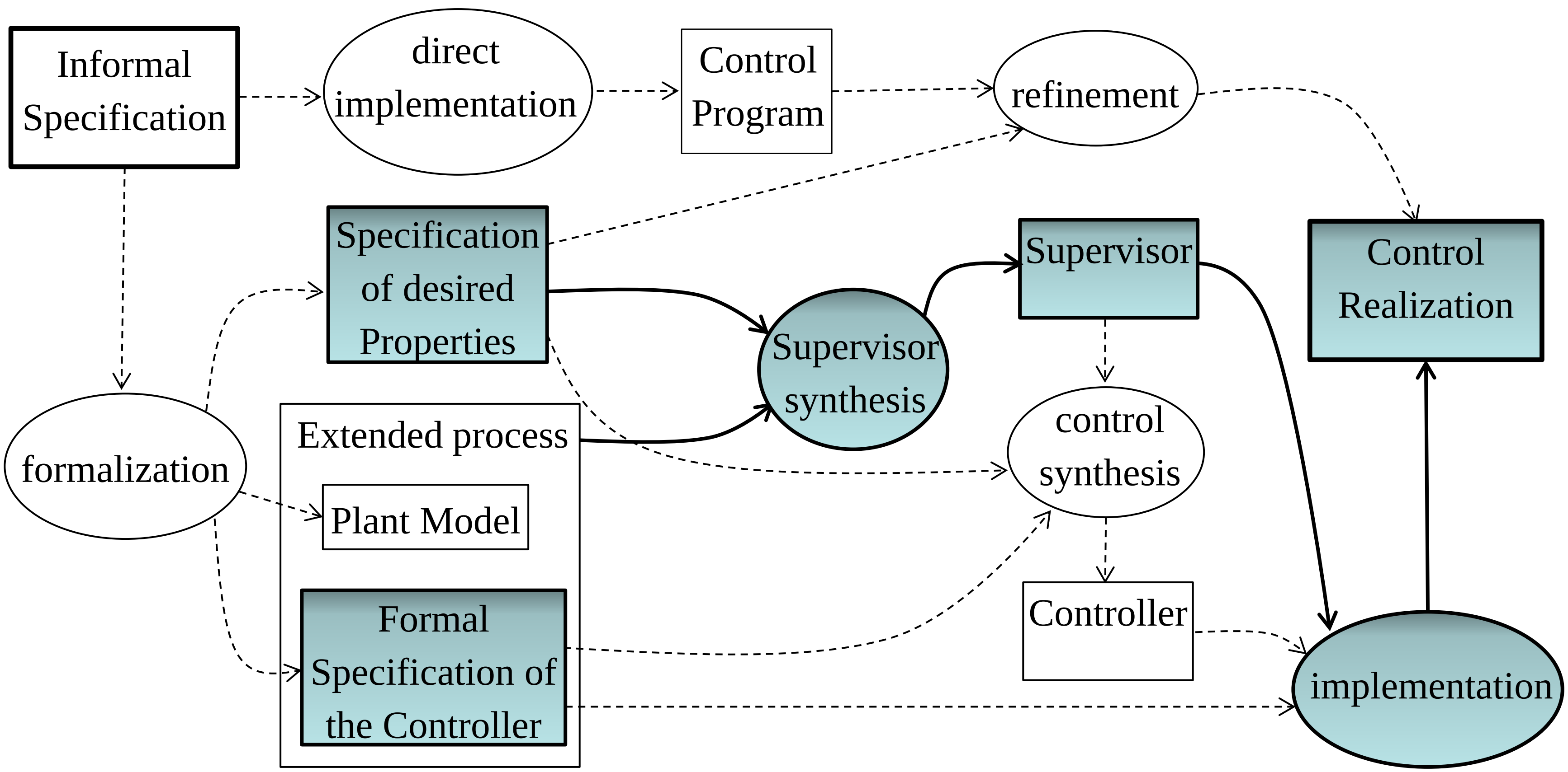
  - How to guarantee that the selected event correspond to the best choice available for each case? → Proposal of criteria for choice

  - How to provide meaningful models? → Enrich plant model with plant/controller interaction features: I/O, PLC-cycle based interpretation

SCT-based approaches: refinement of the supervisor

- PLC implementation of local modular supervisory control (Leal et al., 09, 12; Queiroz & Cury, 02): Model decomposition is driven by the controllability of the events
- Directed Control (Chandra et al. 03; Huang & Kumar, 08)
- Use of a generic PLC cycle and I/O models (Cantarelli & Roussel, 08; Roussel & Giua, 05)
- Timed DES supervisors and sampled-data controllers (Brandin & Wonham, 94; Leduc et al., 14)

SCT based approaches: Introducing a specification model of the controller

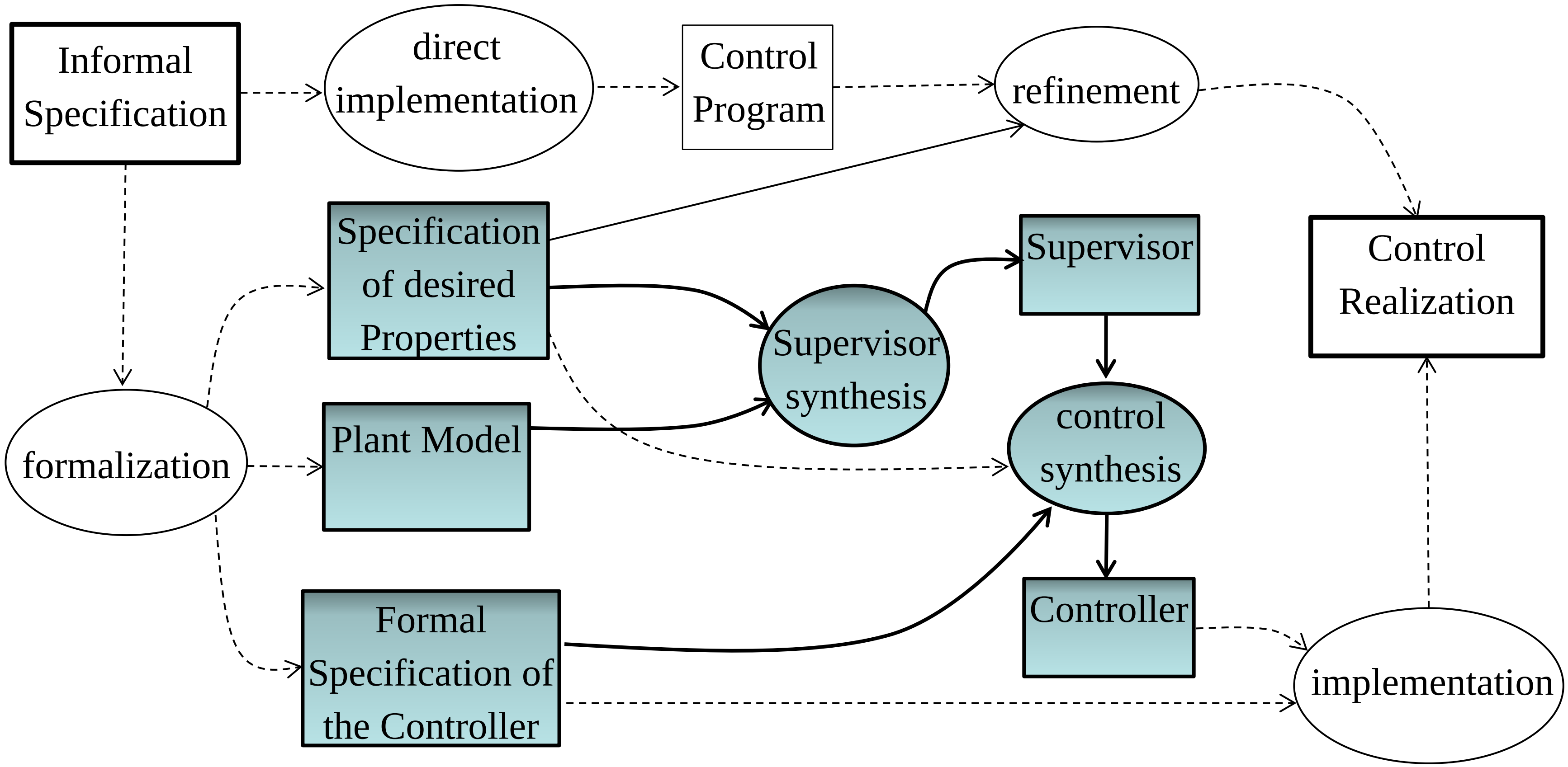


Supervised Control without a plant model

SCT based approaches: Introducing a specification model of the controller

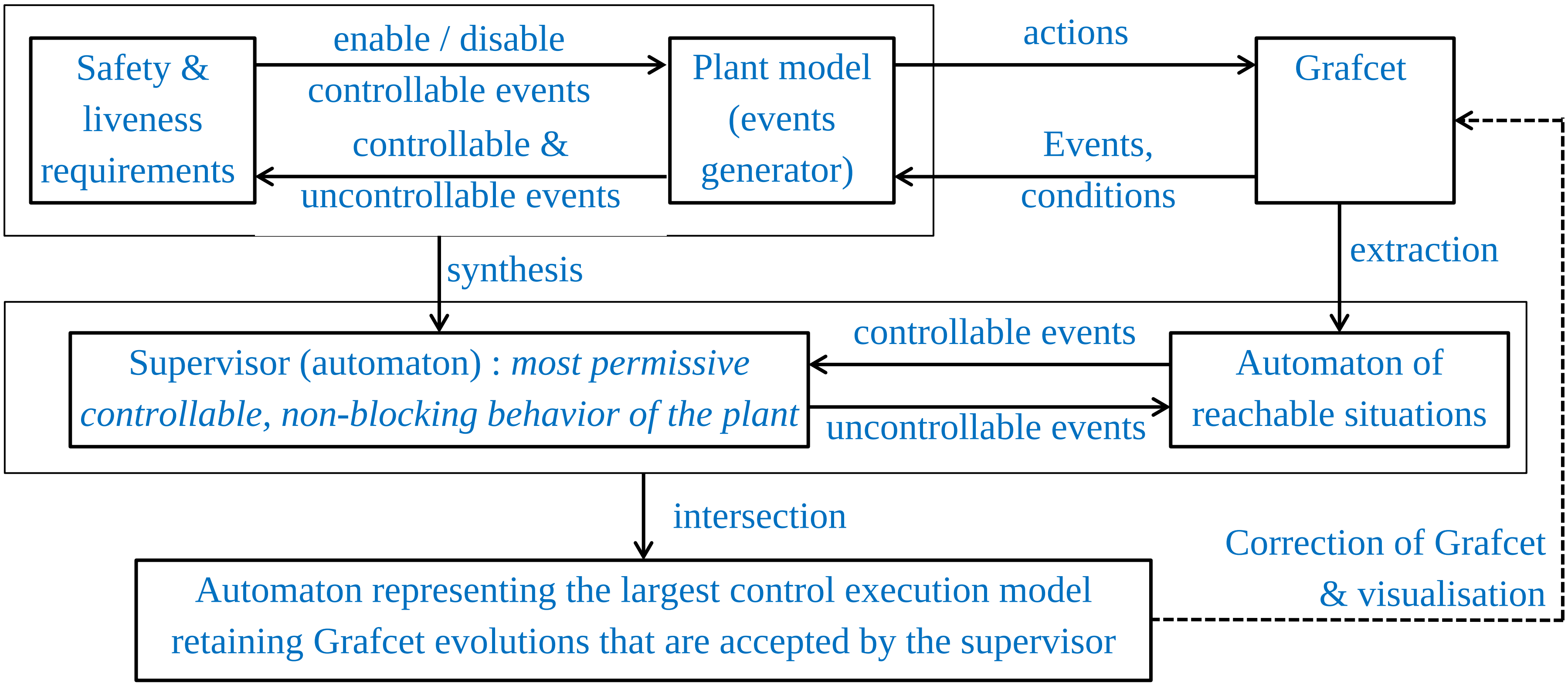
- Translate Grafcet control specifications into automata, the supervisor acts as a coordinator (Charbonnier et al., 1991):
  - limited to a subset of Grafcet structure with restrictive assumptions on plant-controller interactions
- Extension to service-based architecture (Basile et al., 13):
  - IEC 61131 is used to code the basic control sequences in Function Blocks (FB) providing their functionalities as services whose execution is forced by a Petri Net controller (forced events)
  - improved reusability: FB programming only deals with functional aspects, logical constraints are enforced using a formal method
  - well-adapted to the coordination level, does not guarantee deadlock-freeness at the control implementation level

SCT based approaches: Introducing a specification model of the controller



Supervised Control with a plant model

SCT based approaches Zaytoon et al. (1999, 2001)



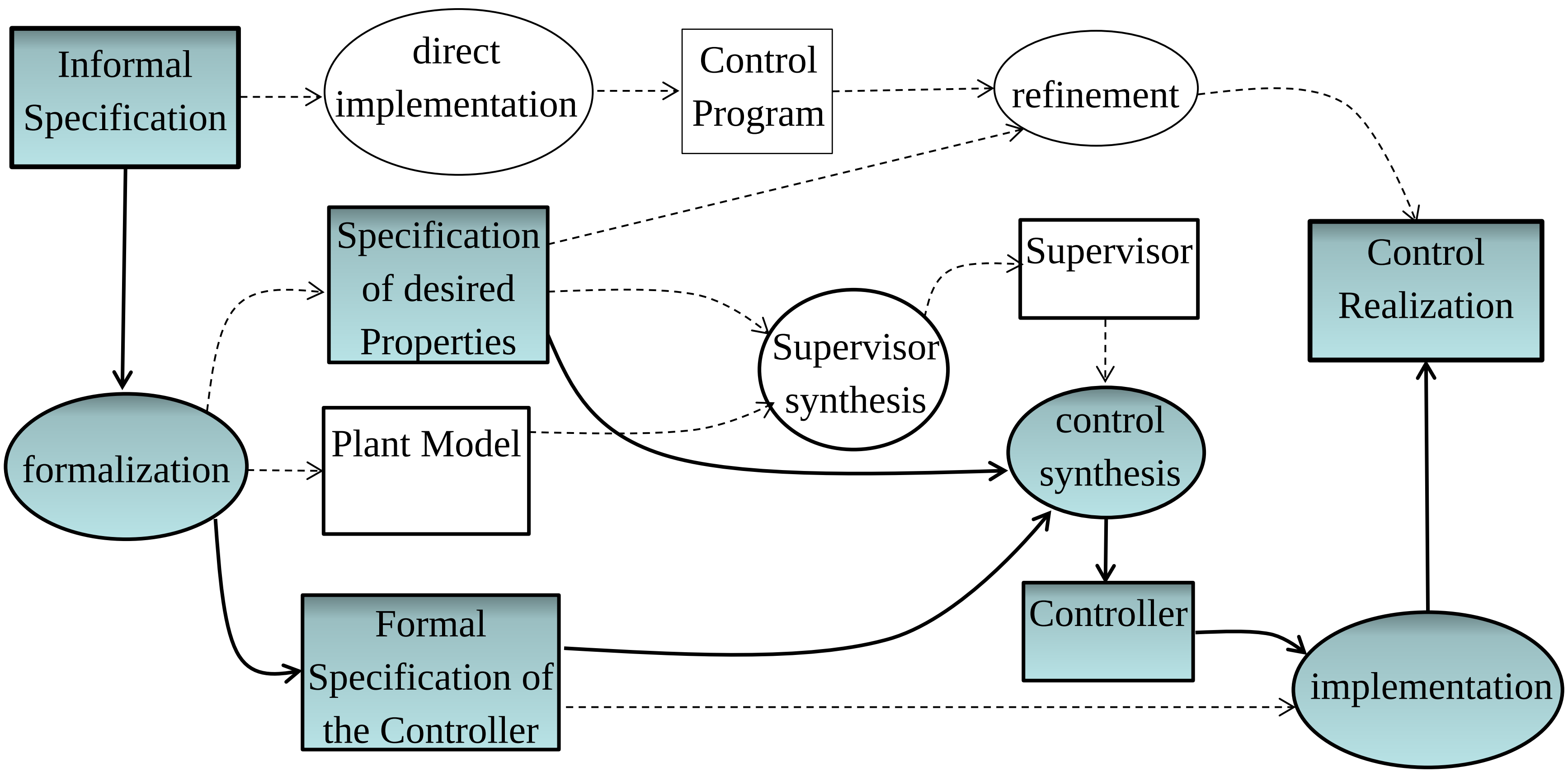
SCT based approaches Zaytoon et al. (1999, 2001)

- The supervisor enforces the safety & liveness specifications, the controller directs the system toward the desired goal, to accomplish a specific set of tasks
- Problem: semantic distance separating Grafcet (*a commonly used control model in practice based on conditions, events, logic operations, synchronism, reactivity, parallelism*) and the SCT model (*asynchronous, interpretation of events & controller-plant interactions*) → How to obtain meaningful models for the plant and the required properties?

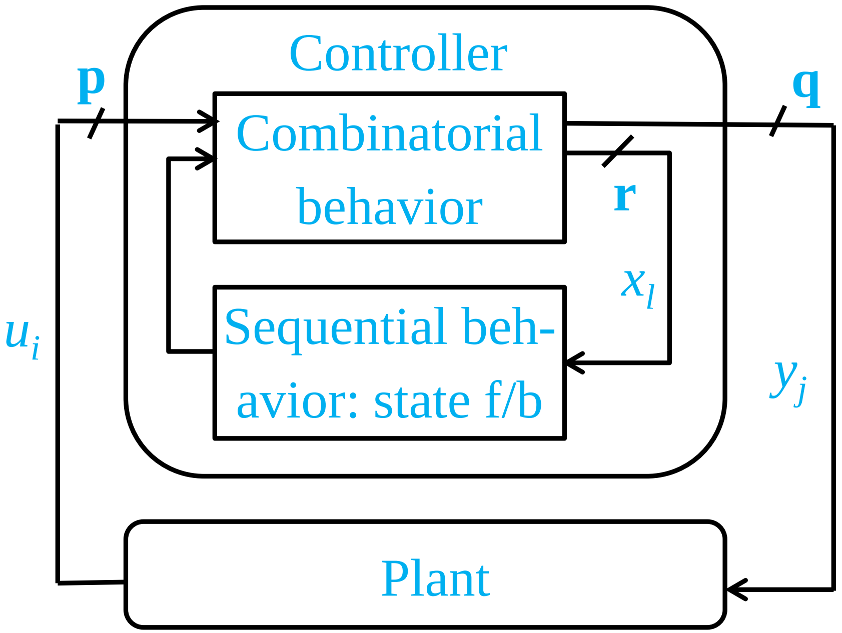
Outline

- Introduction and Context
- Models and Activities for Logic Control Design and Implementation
- Supervisory control theory (SCT)
- SCT-based approaches
- Non SCT based approaches
  - Algebraic approaches
  - Logic Filters
- Conclusion

Non SCT based approaches: Algebraic approaches



Non SCT based approaches: Algebraic approaches (Roussel & Lesage, 14)



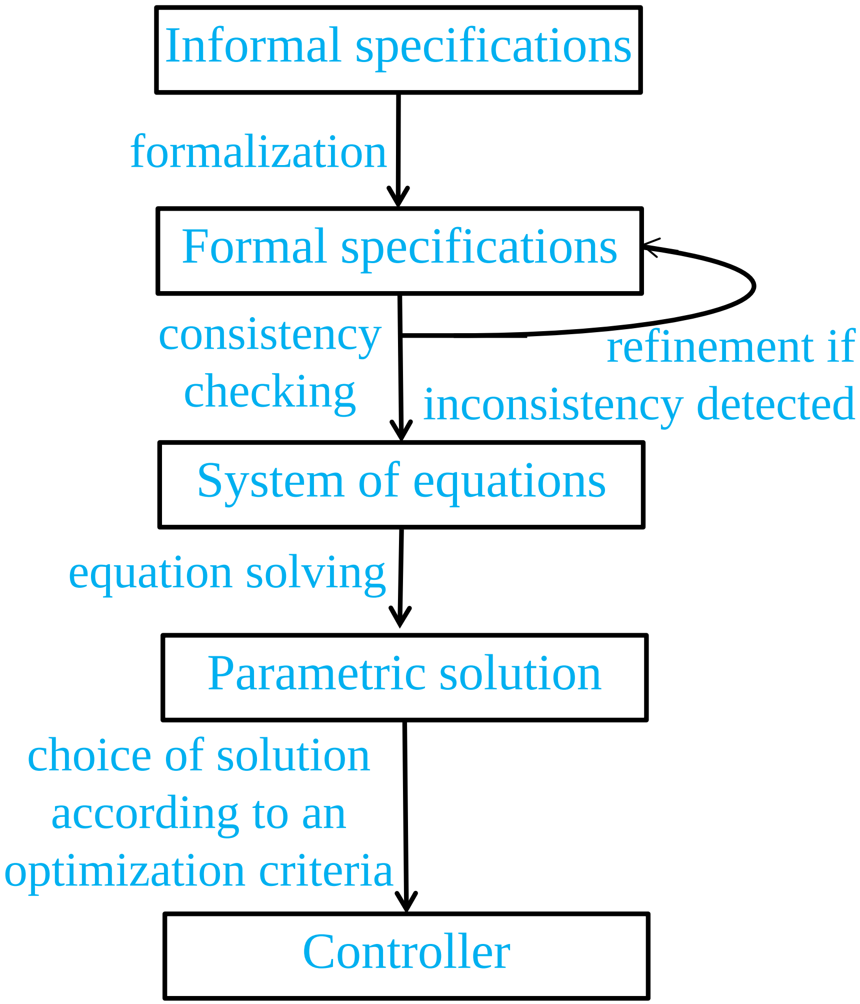
$y_j[k] = F_j\,(u_1[k], \ldots, u_p[k], x_1[k\text{-}1], \ldots, x_r[k\text{-}1])$

$x_l[k] = F_{q+1}\,(u_1[k], \ldots, u_p[k], x_1[k\text{-}1], \ldots, x_r[k\text{-}1])$

Use symbolic representation of Recurrent (switching) Boolean equations + theorem proving techniques to deal with combinatorial explosion,

Synthesis problem: deduce the switching functions by searching a solutions for the parametric equation: $(\forall U_i)(\forall X_l^*)(\exists Y_j)(\exists X_l)\varphi(U_i, X_l^*, Y_j, X_l)$

Non SCT based approaches: Algebraic approaches (Roussel & Lesage, 14)



Advantages:

- using a unique formal framework for:

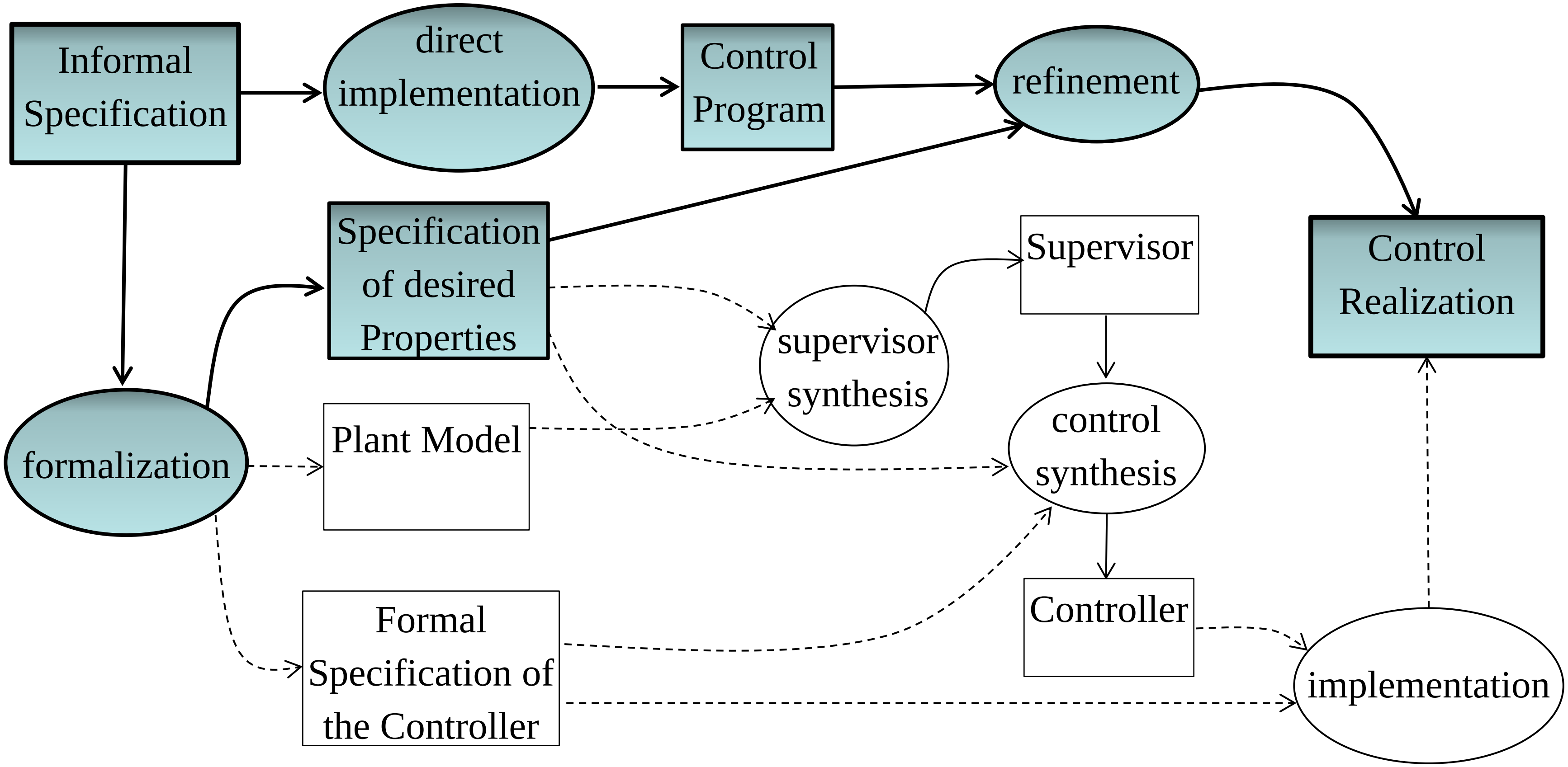
i) modelling, verification & control synthesis,
ii) assisting the designer in formalizing & refining the requirements as PLC design process is iterative, not linear

- recurrent Boolean equations are well adapted to express safety requirements and for implementation

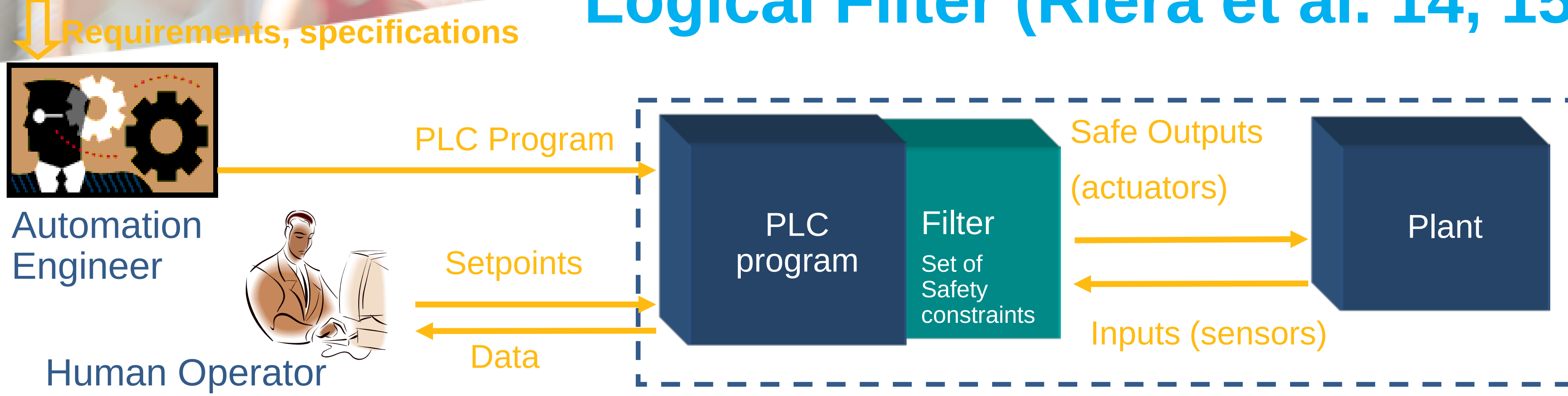
But:

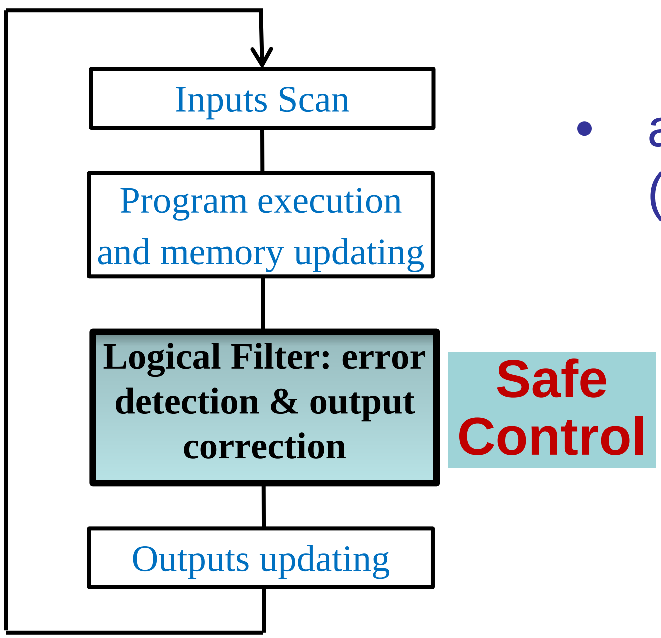
- algebraic approaches are not very popular in engineering practice;
- difficulty of maintenance of the PLC realization, based on a different formalism;
- how to deal with modular design?

Non SCT based approaches: Logical Filter (Riera et al. 14, 15)



Non SCT based approaches: Logical Filter (Riera et al. 14, 15)

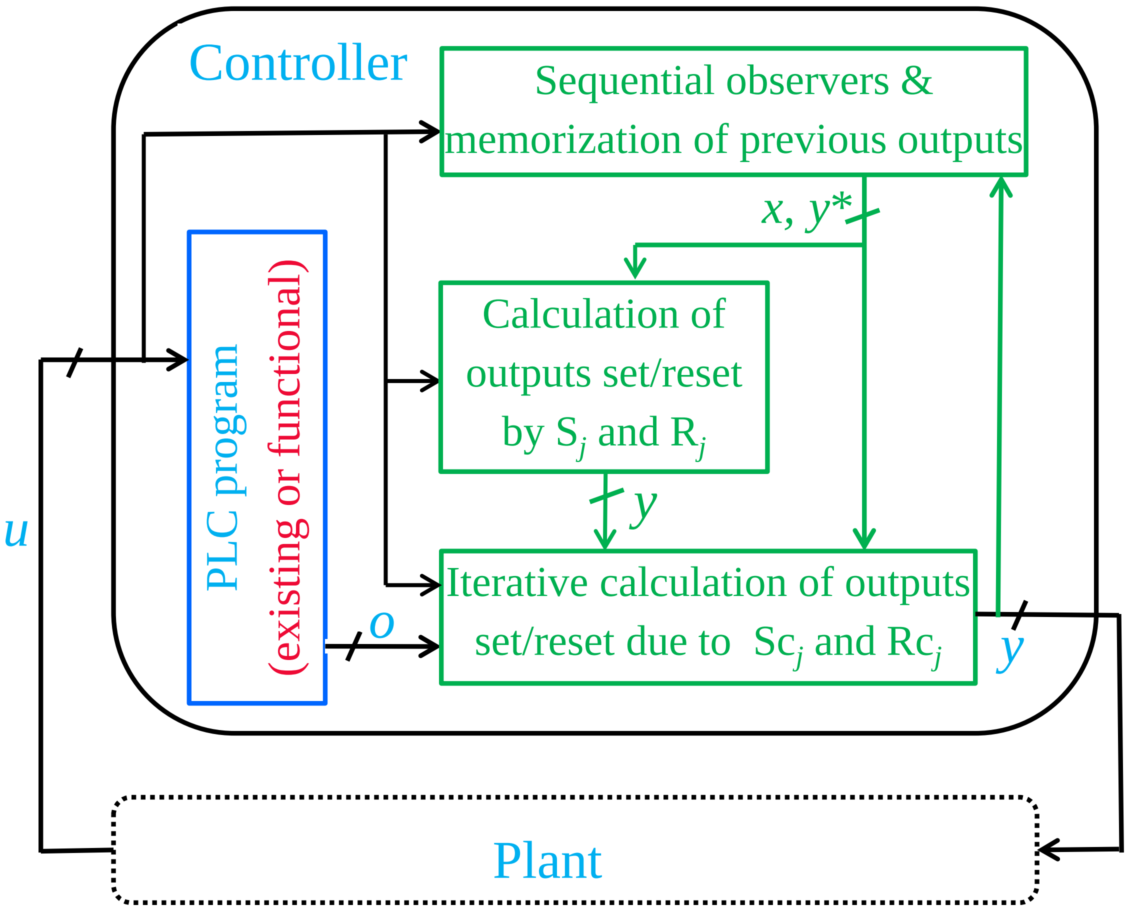




- add a piece of code into PLC program to correct (force) outputs according to safety requirements
  - Simplify maintenance & provide explanations
  - Suitable for distributed system logic control design
  - Does not guarantee deadlock-freeness

Non SCT based approaches: Logical Filter (Riera et al. 14, 15)

Logical Filter design



$\forall y_j$, 4 logical (safety) functions can be defined to set/reset y_j if the function = True , otherwise $y_j = o_j$

S_j : f (u, x, y^*) , sets y_j

R_j : f (u, x, y^*) , resets y_j

⇒ $S_j.R_j = 0$

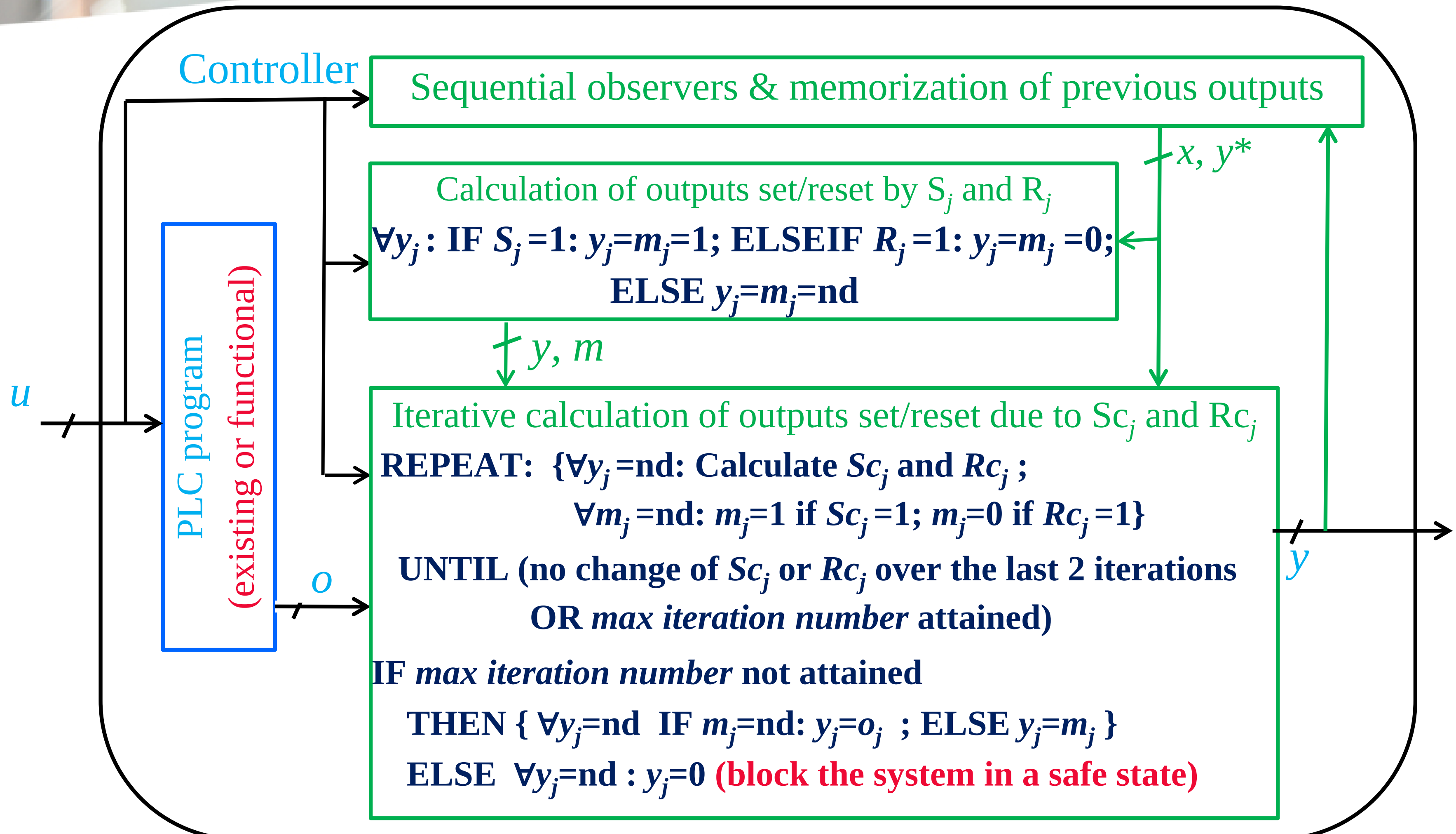
Sc_j : f (u, x, y^*, o) , sets y_j

Rc_j : f (u, x, y^*, o) , resets y_j

⇒ $Sc_j.Rc_j = 0$

S_j, R_j have priority over Sc_j, Rc_j

Non SCT based approaches: Logical Filter (Riera et al. 14, 15)



Converges if number of iterations ≤ twice the number of outputs with combined functions, otherwise the system is blocked in a safe state if constraints are badly defined or if a malfunction occurs

Non SCT based approaches: Logical Filter (Riera et al. 14, 15)

- The designer needs not alter his/her traditional engineering method → *Guarantee safe execution of existing PLC programs without modifying them*

- safety & functional requirements separately defined:

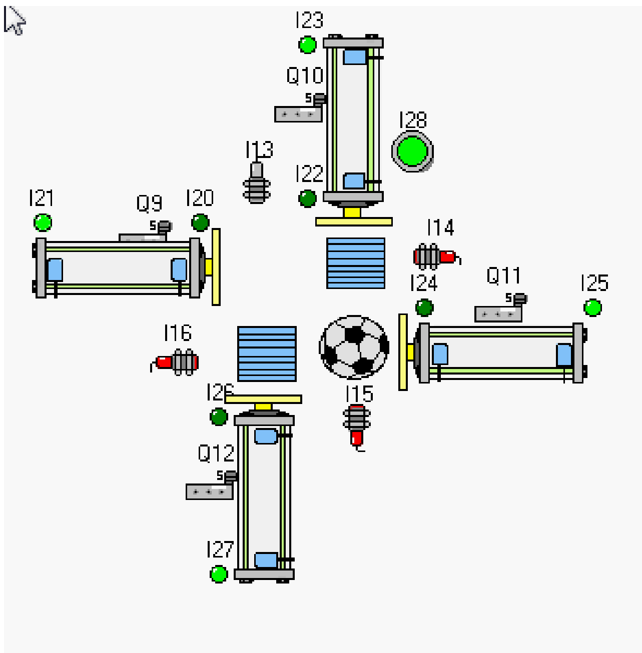
  - Intuitive and natural way to represent safety constraints as output set/reset functions that can be formally checked (offline) to verify pertinence and consistency

  - Simplify the definition of functional aspects: no need to use a complete GRAFCET specification

  - The controller is safe even if functional requirements are badly defined (if PLC program is wrong)

Example: 4 cylinders

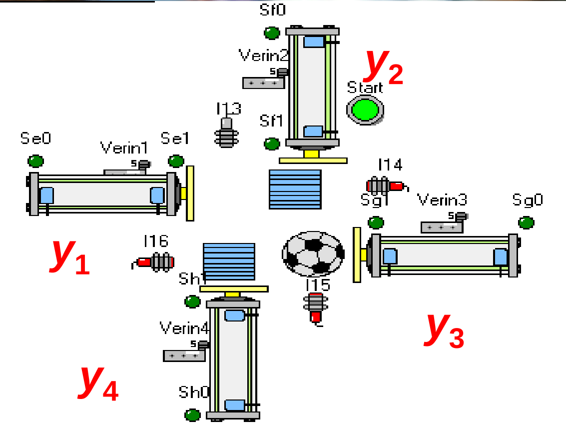
Requirement:
Parts have to turn in the clockwise motion avoiding collisions



ELEMENTS	Sensors	Actuators
Cylinder 1	Se_0, Se_1	y_1
Cylinder 2	Sf_0, Sf_1	y_2
Cylinder 3	Sg_0, Sg_1	y_3
Cylinder 4	Sh_0, Sh_1	y_4
Presence	I_{13}, I_{14}, I_{15}, I_{16}	

- functional aspect: when a part is in front of a cylinder, the cylinder extends & retracts

- safety: avoid collisions between parts and cylinders



Boolean constraints for safe PLC control

For each *y*, 3 functions (simple reset, simple set, combined reset) + 1 observer

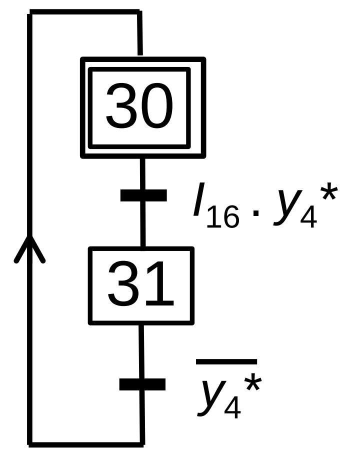
Simple Reset function: *Forbid extending cylinder 1 if:*

It not retracted	*parts exist in front of cylinders 1 & 2*	*there is a part in front of it and cylinder 2 not retracted*	*cylinder 4 is moving with a part*	*there is a part in front of it and if cylinder 2 is extending*

$$R_1 = \overline{y_1^*} \,.\, (\overline{Se_0} + I_{13} \,.\, I_{14} + I_{13} \,.\, \overline{Sf_0} + X_{31} + y_2^* \,.\, I_{13})$$

$S_1 = y_1^* \,.\, \overline{Se_1}$ *continue extending cylinder 1 if it is not completely extended (forbid stopping extension)*

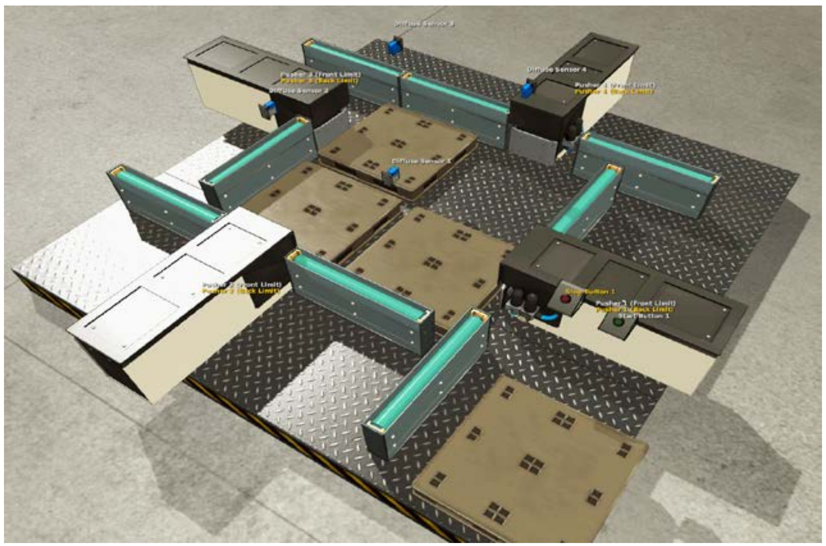
$Rc_1 = I_{16} \,.\, y_4$ *forbid extending 2 adjacent cylinders simultaneously when there is a part*

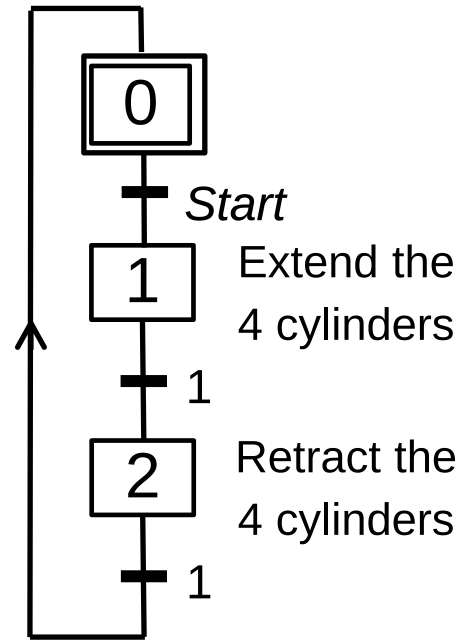


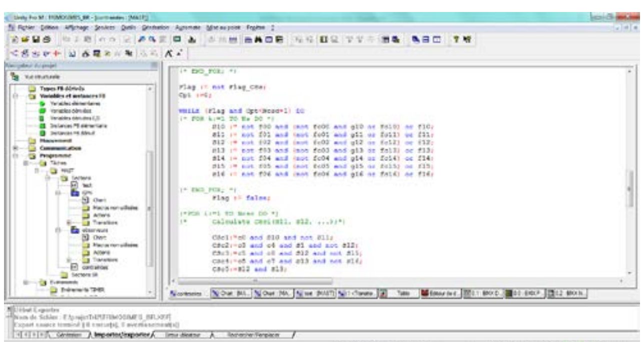
Similar for: y_2 ; y_3 ; y_4

Application to a 4 cylinders system

The method has been implemented and tested with a M340 soft PLC







FACTORY I/O

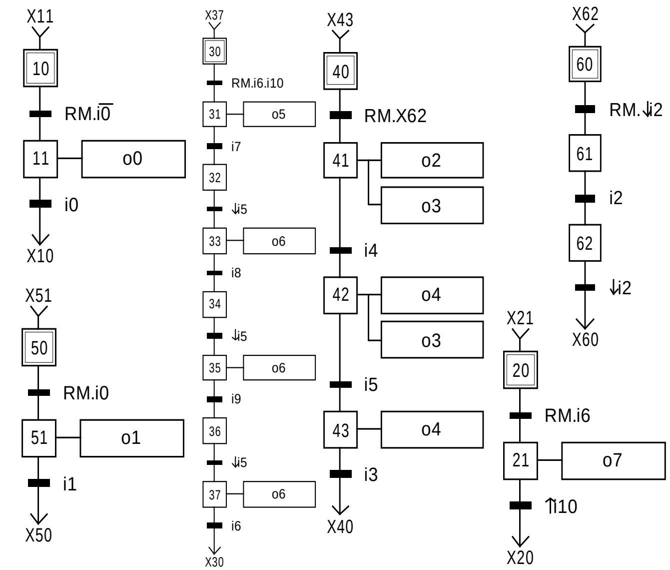
Real Games

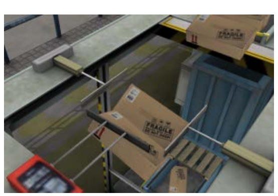
Schneider Electric

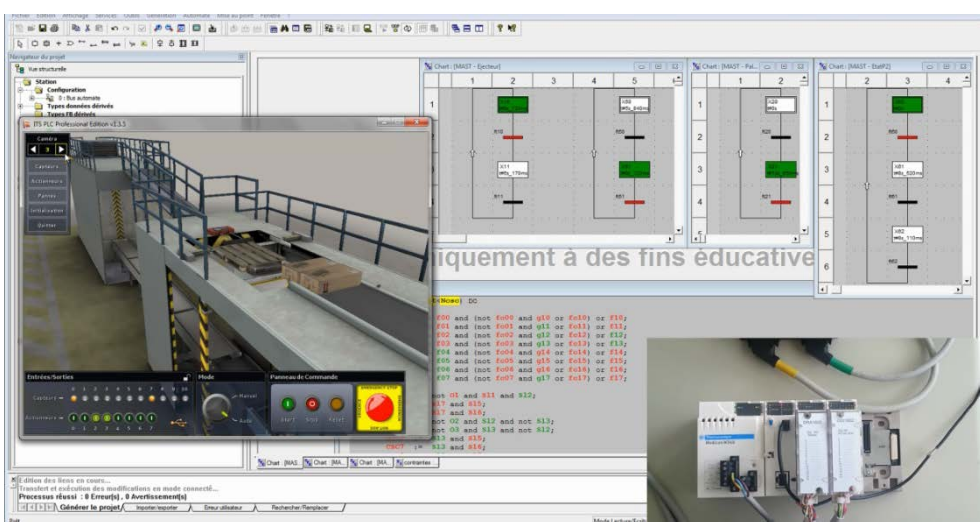
Université de Reims Champagne-Ardenne

Example: virtual palettizer

- 2 case elevators, conveyor belt, exit bay
- 11 sensors, 8 actuators
- Safety analysis (based on FMEA & offline model checking) ⇨ 30 SLC, 9 CLC, and 4 observers
- Objective: palletize cases up to three levels
- Simple functional specification & safe operation, whereas a complete Grafcet is difficult to elaborate because safety & functional aspects mst be mixed







Conclusion

- Powerful theoretical & engineering practice tools/methods have been developed to synthesis & implement logic controllers → Need meaningful models & adapted methods to match the gap between theory & practice

- Adaptation to Industry 4.0, modern factories, networks:

  - Importance and impact of Education and Simulation

  - Exploit/adapt the event-driven, distributed, service-oriented Function Block structure of the new IEC 61499 standard: combine SCT, algebraic approaches & filter-based approaches, each at the right level

  - Adapt to / Make best use of RFID & high-power sensor technology