VERIFICATION OF C3 LINEARIZATION ALGORITHM

Miguel Flor
ADC midterm presentation

Supervisors:
Mário Pereira
António Ravara

CONTEXT

Great necessity of:

- OOP
- Multiple Inheritance
- Understand behaviour

CONTEXT

Multiple Inheritance uses C3

Why C3

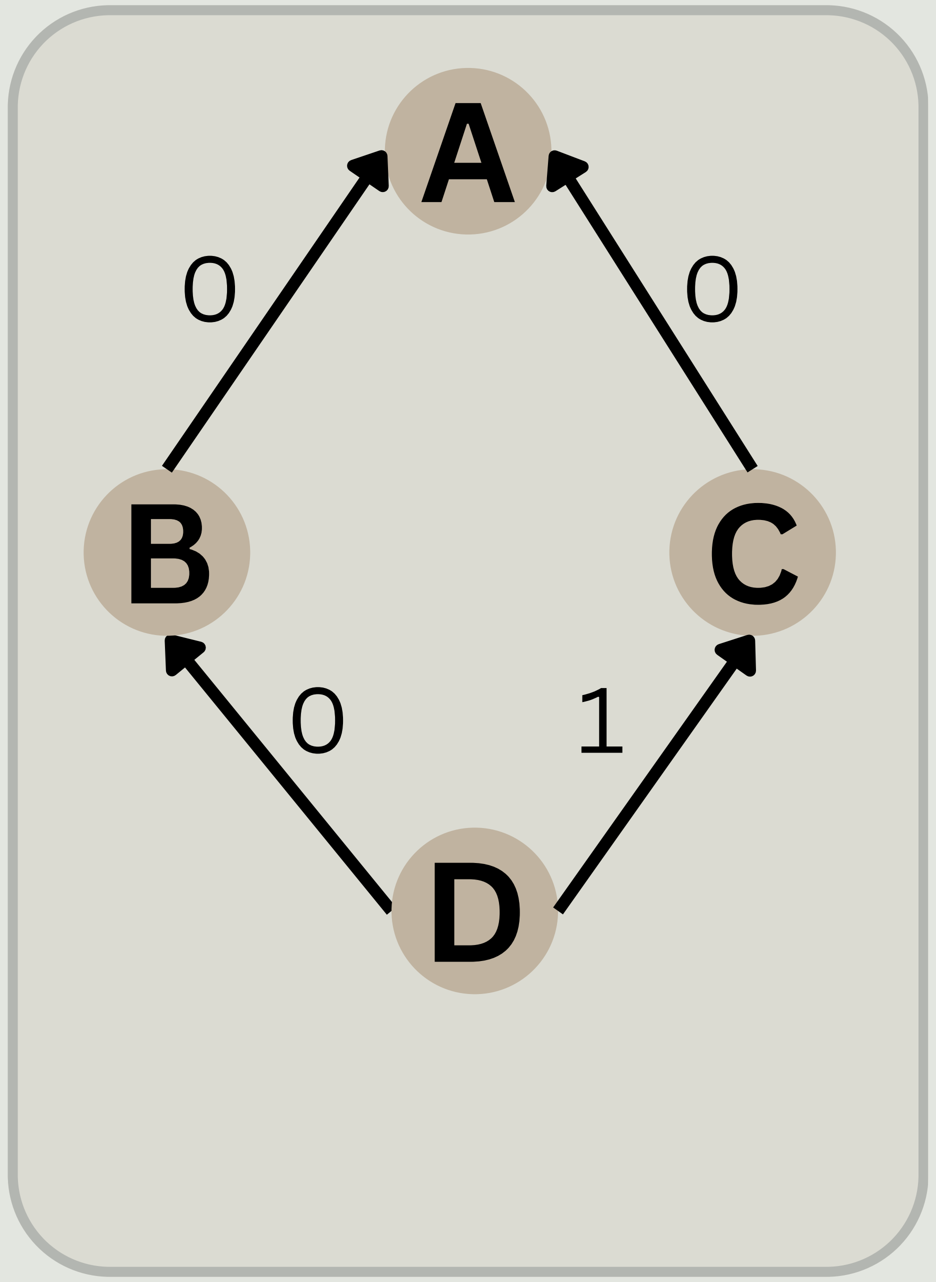
- ✓ A consistent extended precedence graph
- ✓ Preservation of Local Precedence Order
- ✓ Monotonicity Criterion

```
class B
    def m()

class C
    def m()

class D is B and C
    def m()
```

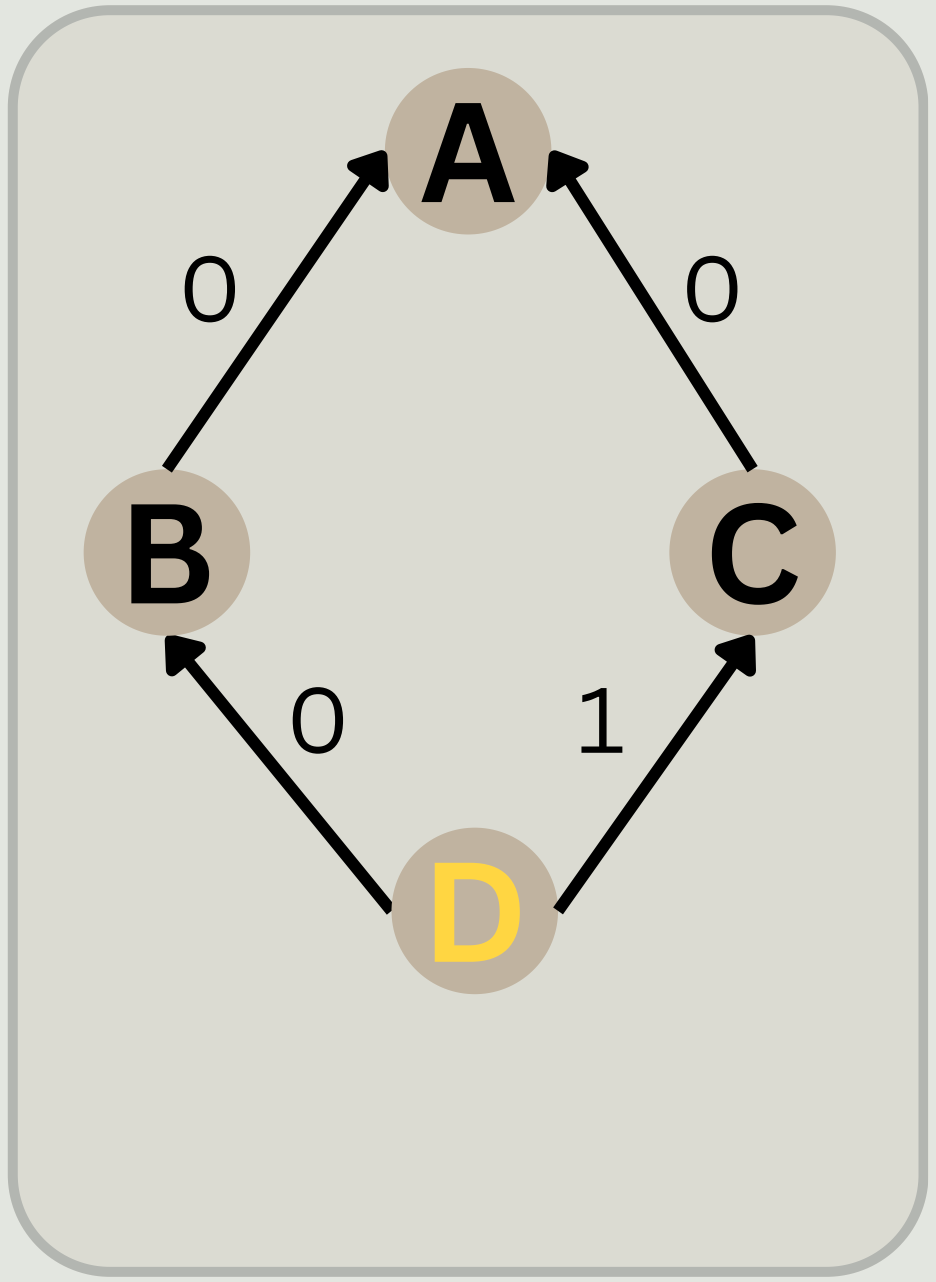
C3 LINEARIZATION



Result

D ⟶ B ⟶ C ⟶ A

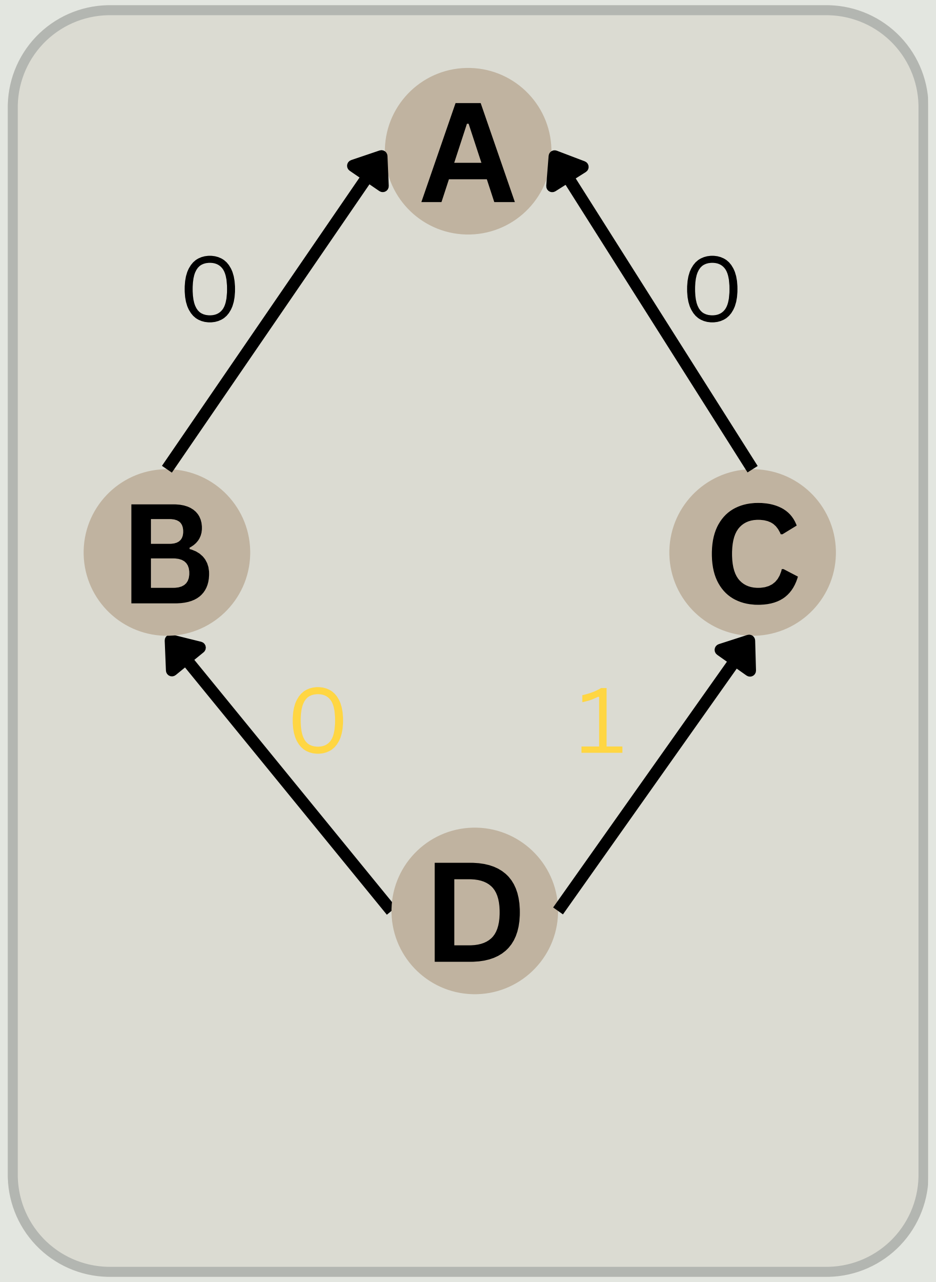
C3 LINEARIZATION



Result

$$D \rightarrow B \rightarrow C \rightarrow A$$

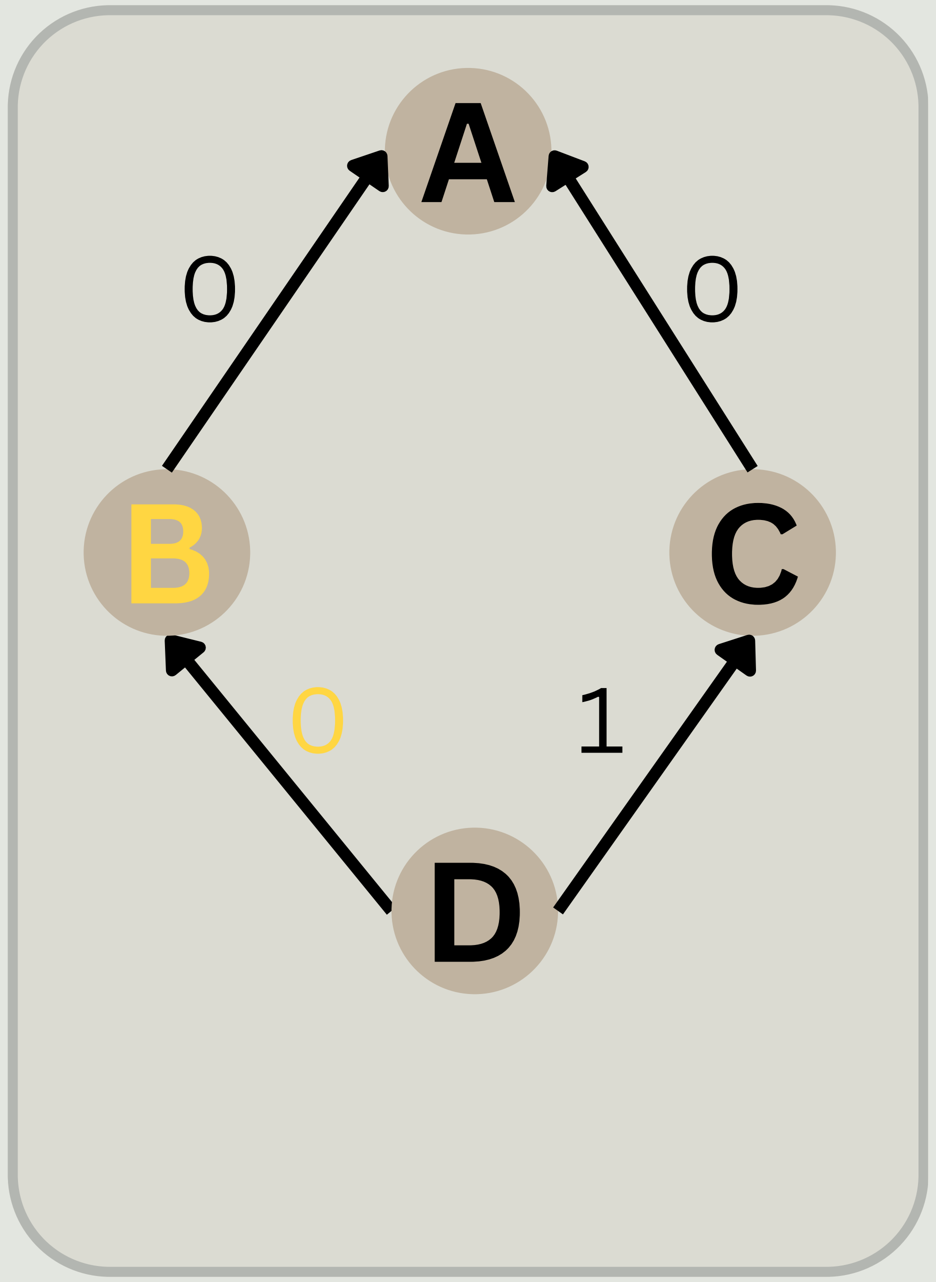
C3 LINEARIZATION



Result

D → B → C → A

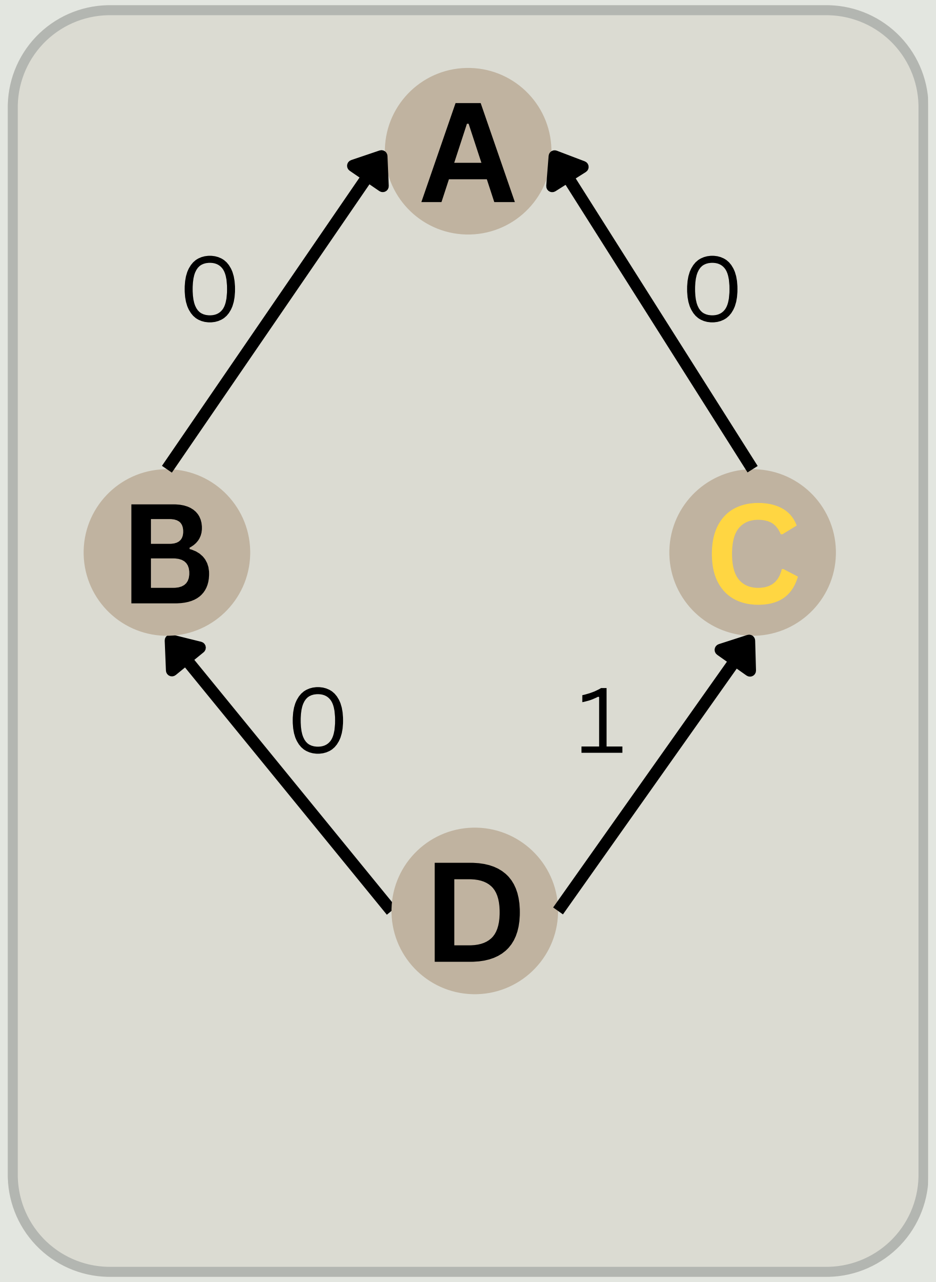
C3 LINEARIZATION



Result

$$\begin{tikzcd} D \arrow[r] & B \arrow[r] & C \arrow[r] & A \end{tikzcd}$$

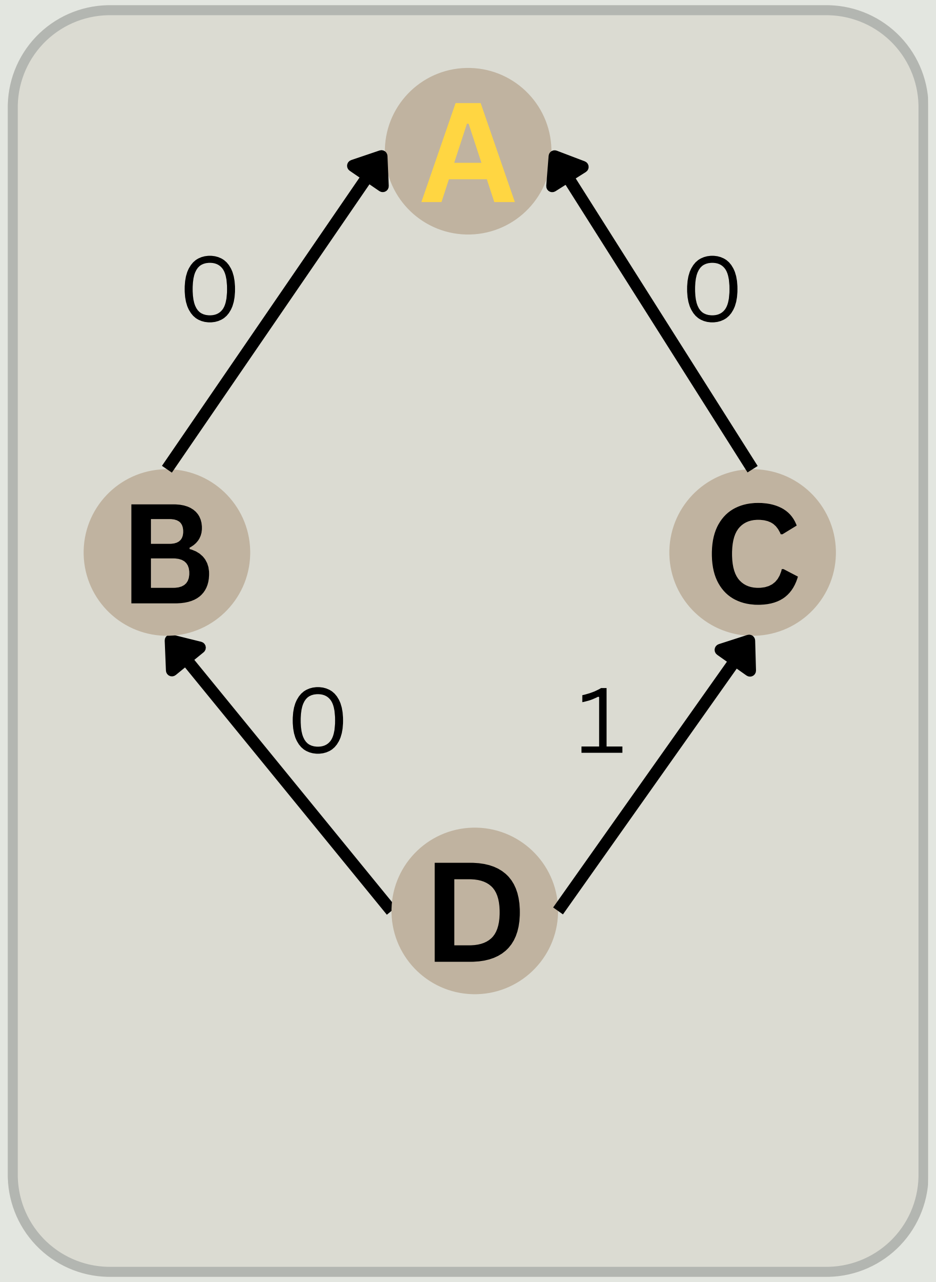
C3 LINEARIZATION



Result

$$\begin{tikzcd} D \arrow[r] & B \arrow[r] & C \arrow[r] & A \end{tikzcd}$$

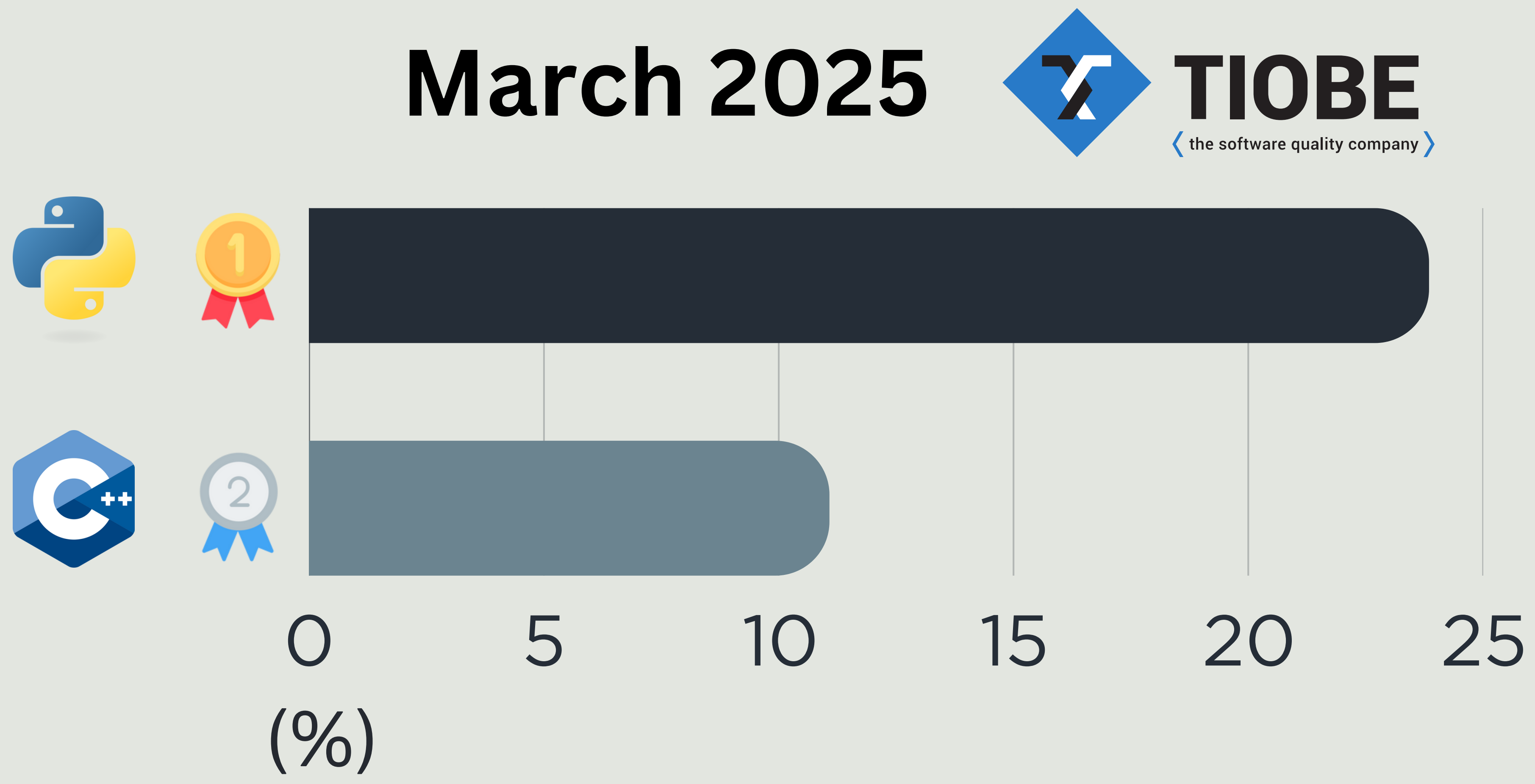
C3 LINEARIZATION



Result

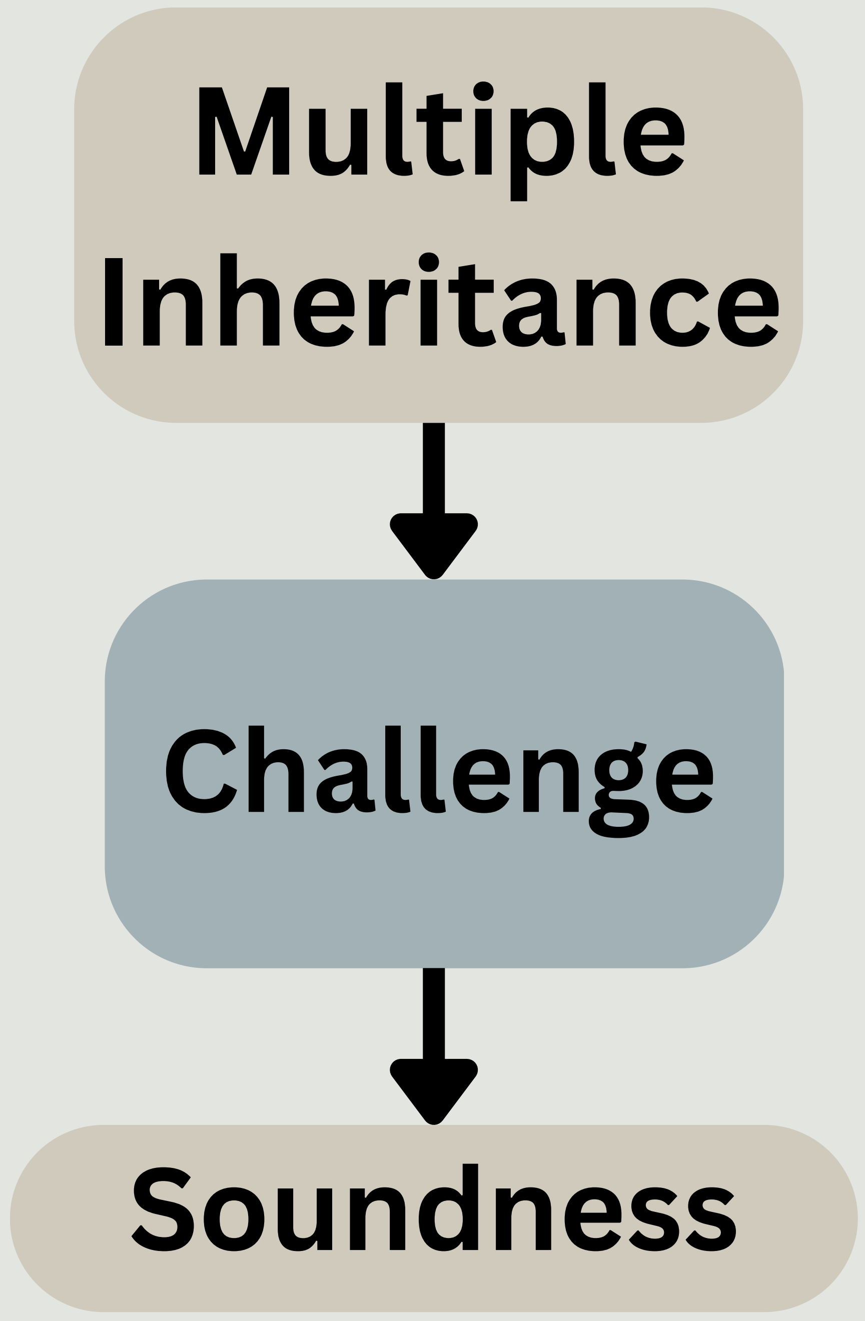
D ⟶ B ⟶ C ⟶ A

THE PROBLEM



48 272 (ETH/H)

88M (EURO/H)

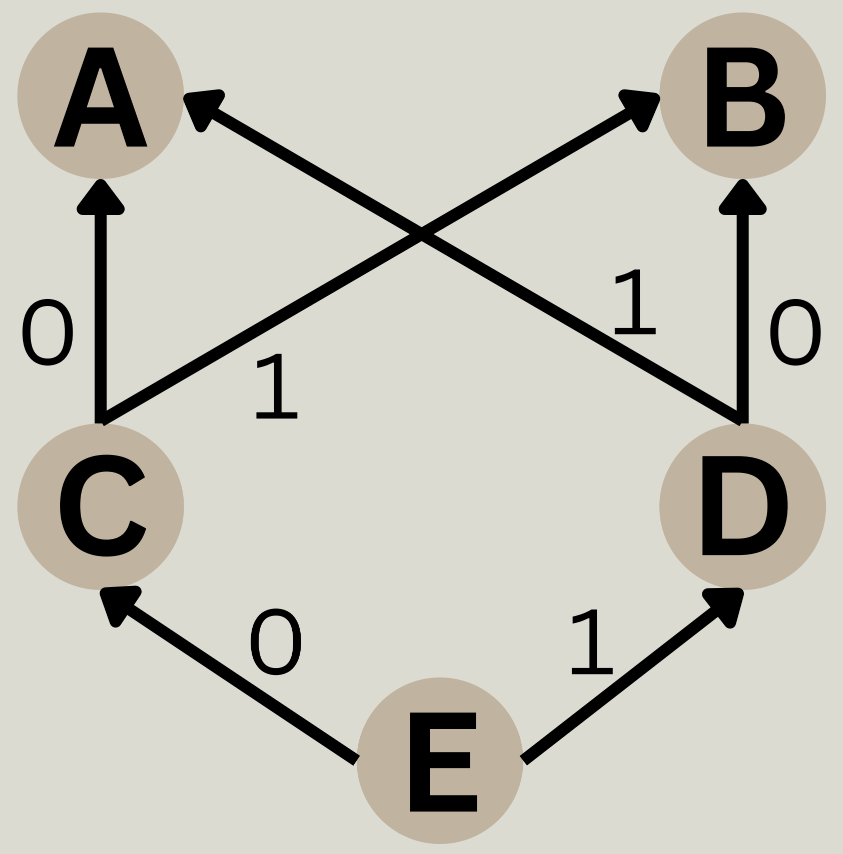


THE PROBLEM

- ✗ Rigorous Presentation
- ✗ Clear Specification

Unclear behaviour

Potential bugs



$$\begin{tikzcd}
A & & B \\
C \arrow[u, "0"] \arrow[urr, "1"'] & & D \arrow[u, "0"'] \arrow[ull, "1"] \\
& E \arrow[ul, "0"] \arrow[ur, "1"'] &
\end{tikzcd}$$

Fail

OBJECTIVES AND CONTRIBUTIONS

Define

- ✓ Rigorous Presentation
- ✓ Rigorous Specification
- ✓ Language Independent

Allows to → derive

C3: consistent with 3 properties

- ✓ A consistent extended precedence graph
- ✓ Preservation of Local Precedence Order
- ✓ Monotonicity Criterion

A CONSISTENT EXTENDED PRECEDENCE GRAPH

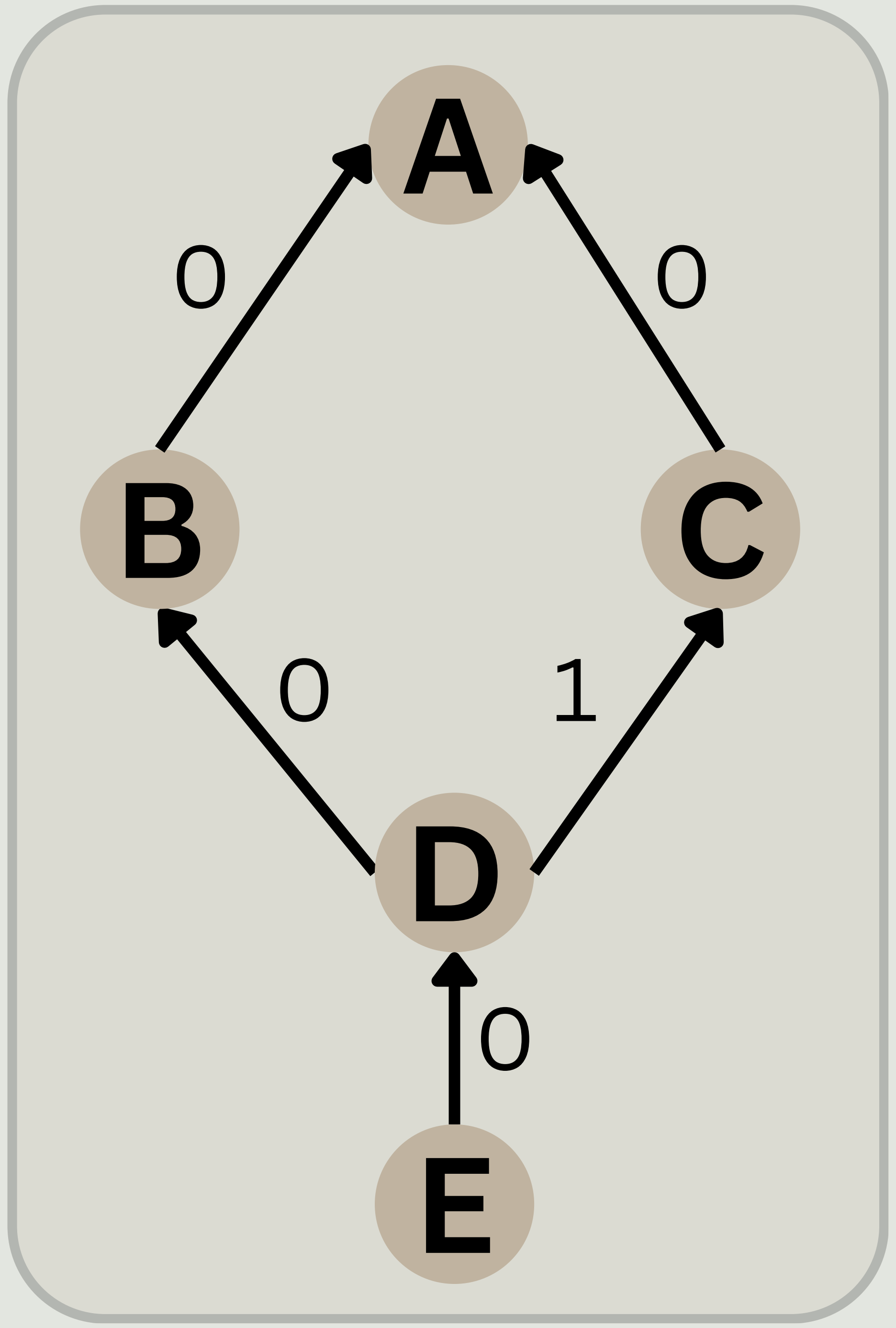
✓

$$\begin{tikzcd}
 & A & \\
B \arrow[ur, "0"] & & C \arrow[ul, "0"'] \\
 & D \arrow[ul, "0"] \arrow[ur, "1"'] &
\end{tikzcd}$$

✗

$$\begin{tikzcd}
A & & B \\
C \arrow[u, "0"] \arrow[urr, "1"'] & & D \arrow[u, "0"'] \arrow[ull, "1"] \\
 & E \arrow[ul, "0"] \arrow[ur, "1"'] &
\end{tikzcd}$$

MONOTONICITY CRITERION



✓ Result

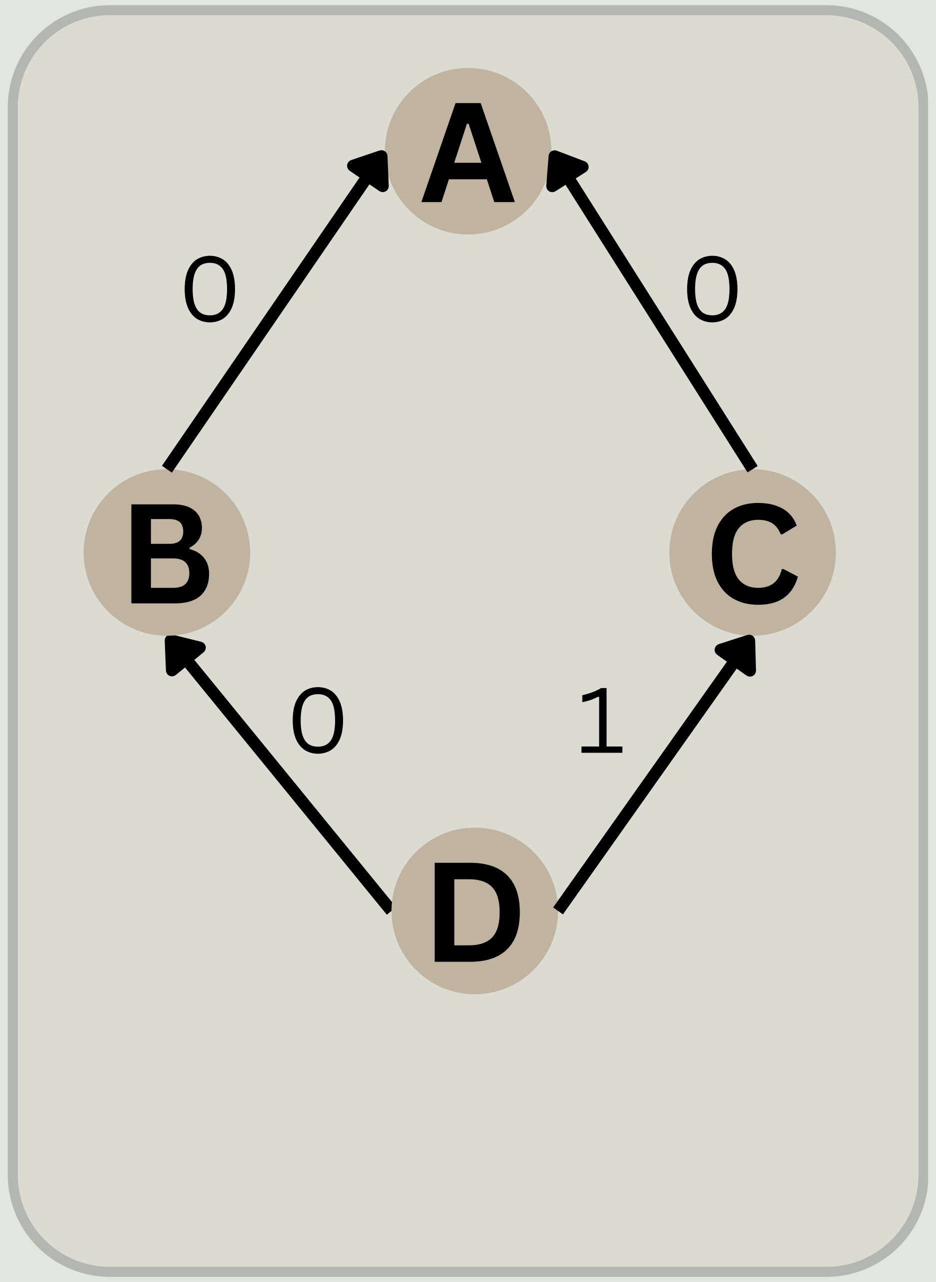
$E \rightarrow D \rightarrow B \rightarrow C \rightarrow A$

✗ Result

$E \rightarrow D \rightarrow C \rightarrow B \rightarrow A$

PRESERVATION OF LOCAL PRECEDENCE ORDER

$$\begin{tikzcd} & A & \\ B \arrow[ur, "0"] & & C \arrow[ul, "0"'] \\ & D \arrow[ul, "0"] \arrow[ur, "1"'] & \end{tikzcd}$$



✓ Result

D → B → C → A

✗ Result

D → C → B → A

RESULTS

Correction Proof

+

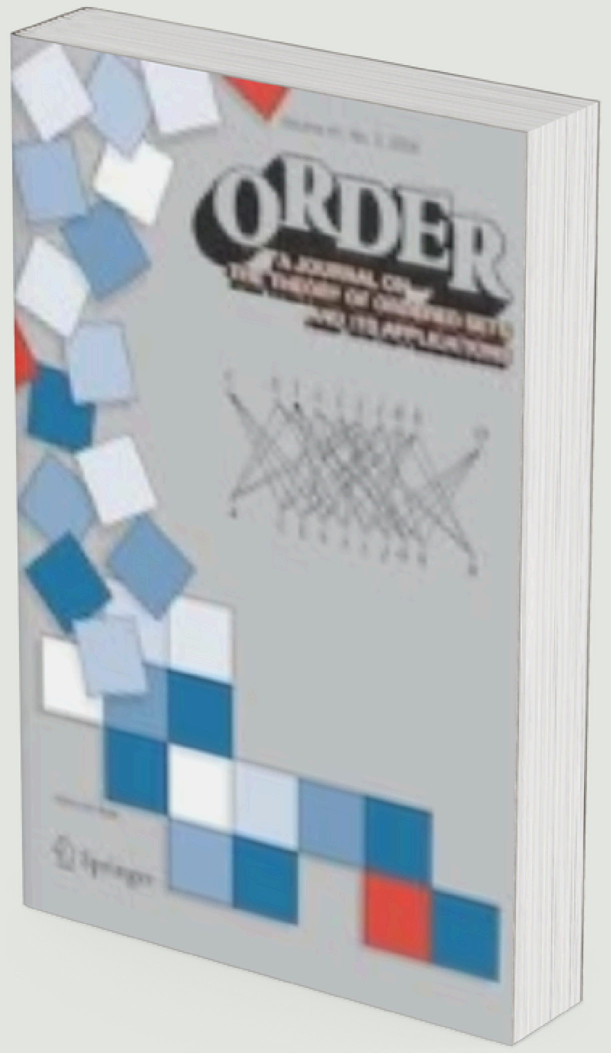
Repository with certified C3 Ocaml

+

Article

METHODOLOGY

Approach to develop certified code



Study →

"Controlling the C3 Super Class Linearization Algorithm for Large Hierarchies of Classes"

Florent Hivert & Nicolas M. Thiéry

↓

Outcome

model & properties

METHODOLOGY

“Towards a solider Solidity”

NOVA
NOVA SCHOOL OF SCIENCE & TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE

JOÃO CARLOS RAPOSO DOS REIS
Bachelor in Computer Science

TOWARDS A SOLIDER SOLIDITY

MASTER IN COMPUTER SCIENCE
NOVA University Lisbon
December, 2023

João Reis

Analyse

$\longrightarrow$

Rectify

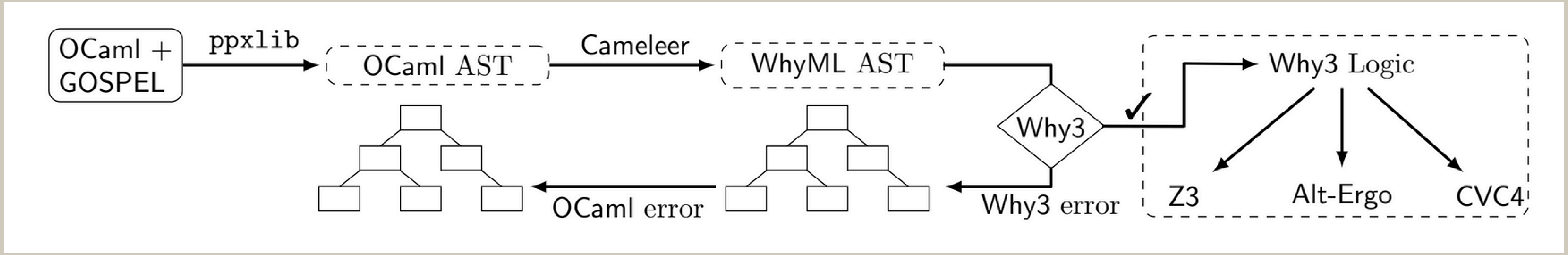
Outcome

Specification

C3 implementation

METHODOLOGY How to get certified code

Cameleer

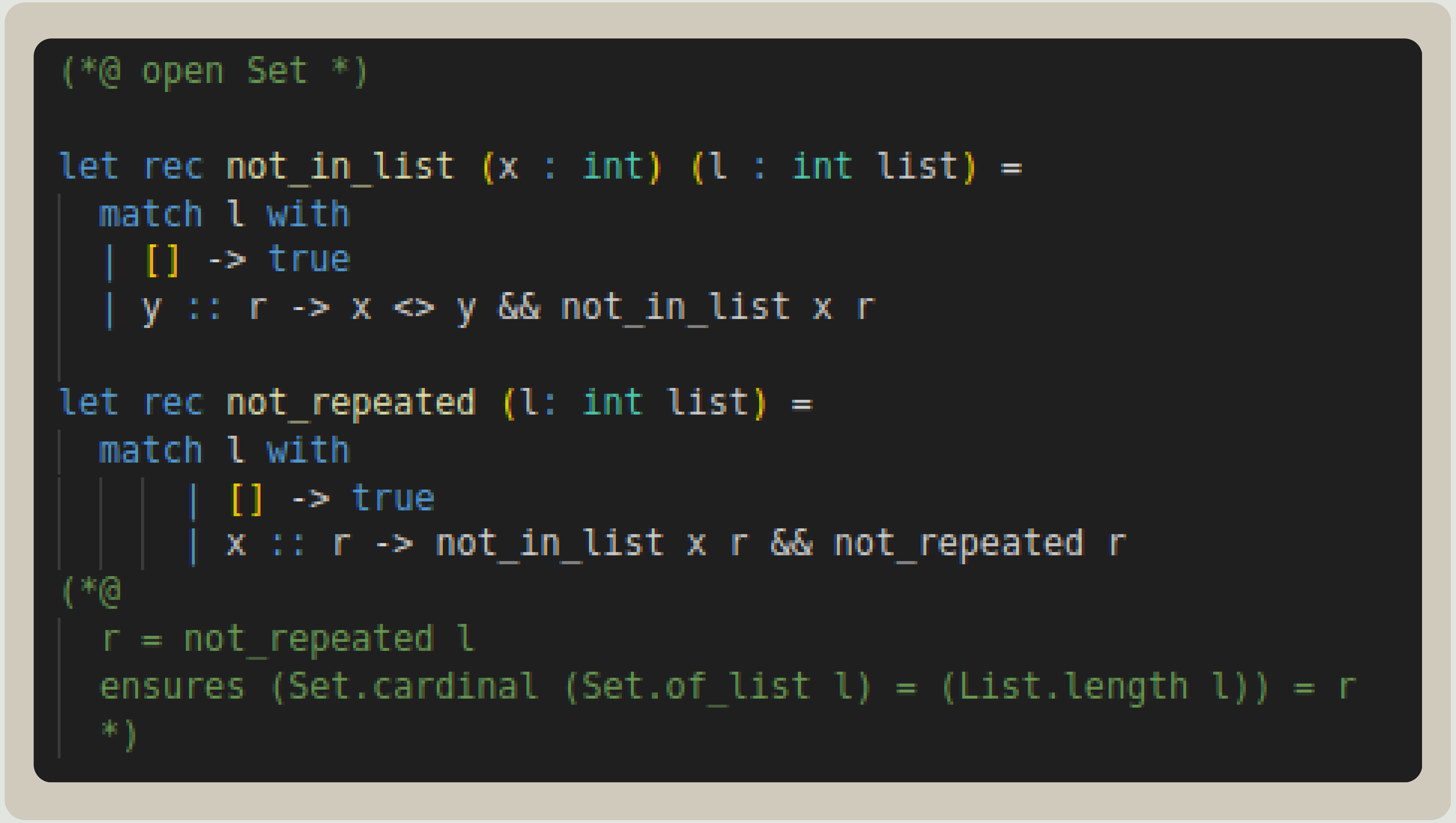


Combines

- Gospel comments
- Ocaml verification

CAMELEER - Work done

Not repeated

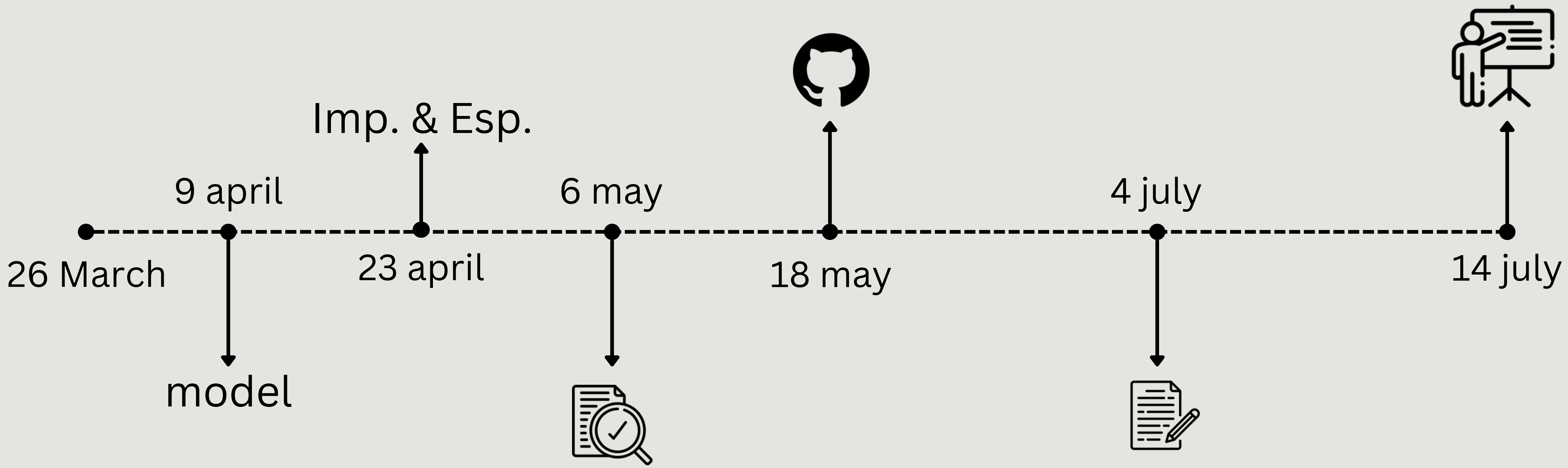


```ocaml
(*@ open Set *)

let rec not_in_list (x : int) (l : int list) =
  match l with
  | [] -> true
  | y :: r -> x <> y && not_in_list x r

let rec not_repeated (l: int list) =
  match l with
      | [] -> true
      | x :: r -> not_in_list x r && not_repeated r
(*@
  r = not_repeated l
  ensures (Set.cardinal (Set.of_list l) = (List.length l)) = r
  *)
```

PLAN


Imp. & Esp.
9 april
6 may
4 july
26 March
23 april
18 may
14 july
model

Thank You

For your attention