CLEARSY
Safety Solutions Designer

Hackathon
JUL2025

AIX
LYON
PARIS
STRASBOURG

WWW.CLEARSY.COM

Airlock
Functional Controller

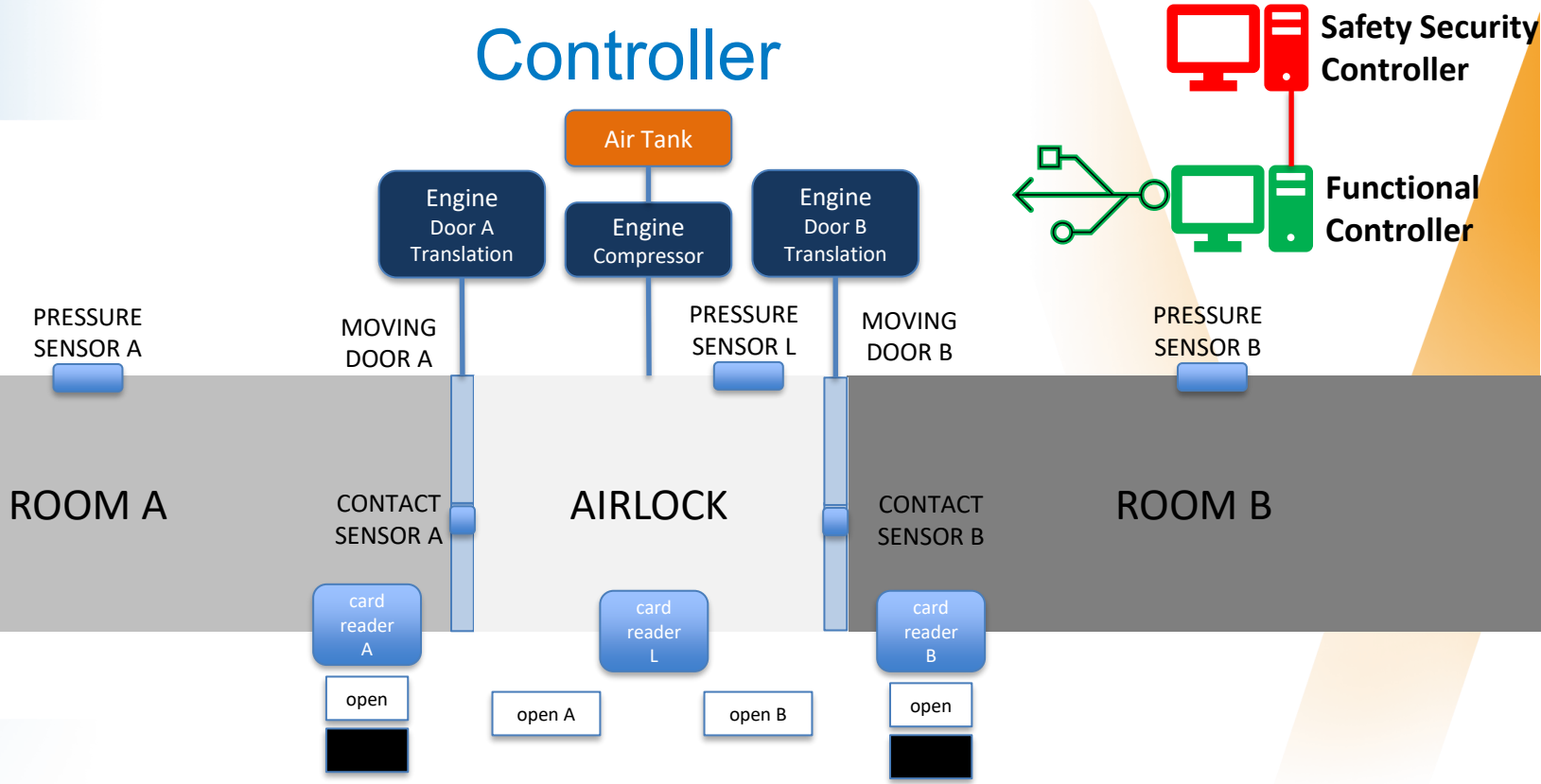
Thierry Lecomte
R&D Director

PART III

One Solution

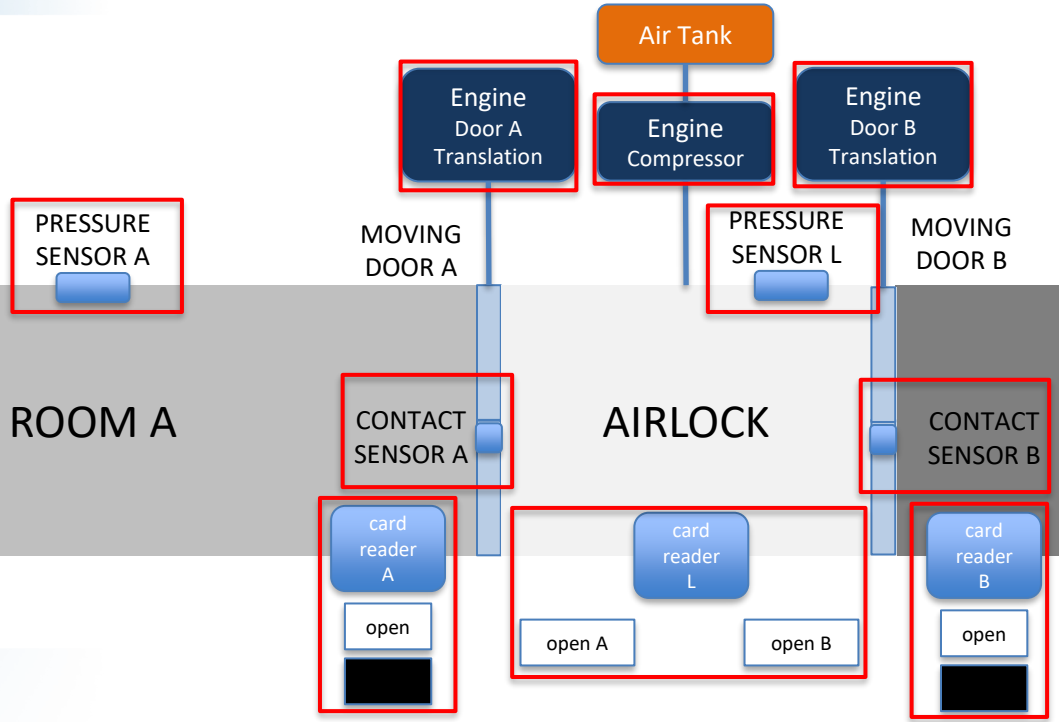
THIERRY.LECOMTE@CLEARSY.COM

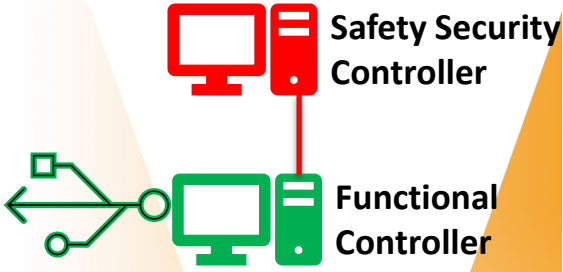
Controller



design decision: we remove the **close** buttons to save money

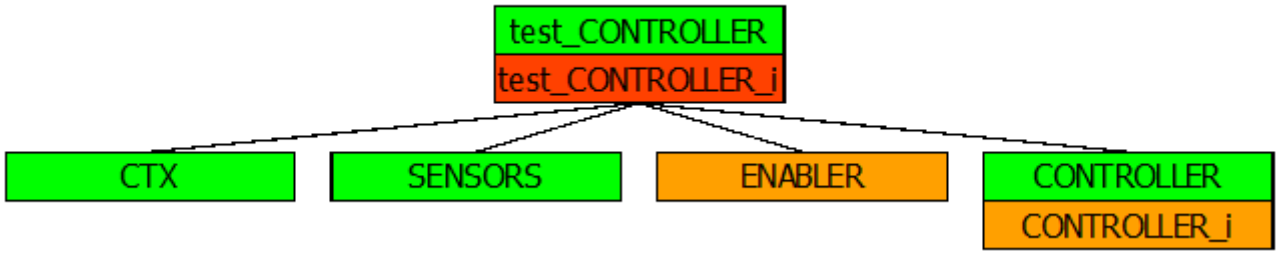
Controller





Your turn

- *Develop a B model of this control function [8 pt]*
  - *many sensors to read*
  - *Definition of objective (based on card reader activated and button pushed)*
  - *Definition of actions (based on objective and current sensors, this is the next action to perform in this order: close door, pressure, open door)*
  - ***safety properties** – they are going to be linked in a combining component where the enabler will be linked with the controller*



Your turn

- Complete the postcondition of the function control
  - Keep focused on the main aspects (safety first !)

Your turn

- It is important to have a mental model of the system and the way it is going scheduled
- Many aspects to consider but do not put the algorithm in the specification
- What are the minimum safety properties to consider when dealing with actions ?
  - Opening doors, modifying the pressure
  - Closing doors is safe (except if someone/something is in the path but by hypothesis we don't care as we cannot sense people)

```
control =
BEGIN
    current_action,
    current_objective :(
        current_action: ACTIONS &
        current_objective : OBJECTIVES &
        (current_action = TRANSLATE_OPEN_DOOR_A =>
            pressure_sensor_l = PRESSURE_A & contact_sensor_b = TRUE
        ) &
        (current_action = TRANSLATE_OPEN_DOOR_B =>
            pressure_sensor_l = PRESSURE_B & contact_sensor_a = TRUE
        ) &
        (current_action = ADAPT_PRESSURE_L_TO_A =>
            contact_sensor_b = TRUE & contact_sensor_b = TRUE
        ) &
        (current_action = ADAPT_PRESSURE_L_TO_B =>
            contact_sensor_a = TRUE & contact_sensor_b = TRUE
        )
    )
END
```

Airlock pressure adequate
Opposite door closed

Both doors closed

Your turn

- It is important to have a mental model of the system and the way it is going scheduled
- Many aspects to consider but do not put the algorithm in the specification
- What are the minimum safety properties to consider when dealing with actions ?
  - Opening doors, modifying the pressure
  - Closing doors is safe (except if someone/something is in the path but by hypothesis we don't care as we cannot sense people)
- Other properties can be added at will to add coherency

Other properties

```
(current_action = TRANSLATE_CLOSE_DOOR_A =>
        pressure_sensor_l = PRESSURE_A &
        contact_sensor_a = FALSE
  ) &


(current_action = NONE =>
        current_objective = OBJ_NONE
) &
```

Mistakes

```
(current_action = TRANSLATE_OPEN_DOOR_A =>
        pressure_sensor_l = PRESSURE_A &
        contact_sensor_b = TRUE &
        contact_sensor_a = TRUE &
        current_objective = OBJ_OPEN_DOOR_A
 ) &
```

Valid only to initiate

Some mistakes

```
(current_objective = OBJ_NONE =>
        button_room_a_open_a = FALSE &
        button_room_l_open_a = FALSE &
        button_room_l_open_b = FALSE &
        button_room_b_open_b = FALSE
)
```

User can push buttons without being authenticated

Some mistakes

```
(current_objective = OBJ_OPEN_DOOR_A =>
      contact_sensor_a = TRUE &
      (
      button_room_a_open_a = TRUE or
      button_room_l_open_a = TRUE
      ) &
      (
      current_authentication = AUTHENTICATED_A or
      current_authentication = AUTHENTICATED_L
      )
) &
```

Valid only to initiate

Your turn

```
MACHINE
    test_CONTROLLER
OPERATIONS
    test_control = skip
END
```

- Execute 10 steps with ProB [2 pts]
  - Save the probtrace file

```
IMPLEMENTATION test_CONTROLLER_i
REFINES test_CONTROLLER

IMPORTS CTX, SENSORS, ENABLER, CONTROLLER
OPERATIONS
    test_control =
    BEGIN
        update_sensors_states;
        process_readers;
        control;
        compute_enabling
    END
END
```

Your turn

- ***Optional**: do you see a more efficient way to manage the sequence of actions to perform, to complete a scenario? [3 pt]*

The modelling is simplistic because of the configuration.
We could have modelled actions as ordered B sequences and removed elements when an action is completed.
We could have used a different interface (not read nor buttons in the airlock) and only buttons “go to A” or “go to B” (people forced to go through before being able to go back).