Laboratoire Méthodes Formelles

CentraleSupélec | université PARIS-SACLAY

DÉVELOPPEMENT DE SYSTÈMES CRITIQUES AVEC LA MÉTHODE EVENT-B

LES CONCEPTS DE BASE DE LA MODÉLISATION ET DE LA PREUVE

🎓 3A cursus ingénieurs - Mention Sciences du Logiciel
🏛 CentraleSupelec - Université Paris-Saclay - 2025/2026

Idir AIT SADOUNE
idir.aitsadoune@centralesupelec.fr

OUTLINE

Back to the outline - Back to the begin

CentraleSupélec
université PARIS-SACLAY

OUTLINE

Back to the outline - Back to the begin

CentraleSupélec
université PARIS-SACLAY

LES TYPES DE MÉTHODES FORMELLES

- **Une méthode formelle** est une approche systématique utilisée pour déterminer si **un programme respecte des propriétés** spécifiques.
- Différents types de **méthodes formelles** (selon cette définition) :
  - **Type checking** ou la vérification de type
  - **Static analysis** ou l'interprétation abstraite
  - **Model checking** ou la vérification de modèles
  - **Theorem proving** ou la preuve de théorèmes

CentraleSupélec
université PARIS-SACLAY

TYPE CHECKING

- Contrôler les **propriétés de bas niveau** des variables dans un programme.
- Un **type** définit :
  - un **ensemble de valeurs** à affecter à une **variable**
  - les **opérations** que l'on peut effectuer sur une variable
  - la façon dont une variable sera **stockée dans la mémoire**
- **Type checking** contrôle si :
  - **les affectations** de valeur à une variable sont **correctes**
  - la variable n'est utilisée que dans **les opérations autorisées**
- Cela se fait **automatiquement** dans les **compilateurs**.

CentraleSupélec
université PARIS-SACLAY

STATIC ANALYSIS

- C'est une **technique automatique** utilisée pour vérifier qu'un programme n'aura pas certaines **erreurs d'exécution**.

- **Erreurs d'exécution** typiques détectées :
  - division par zéro
  - débordement lié au tableau
  - débordement arithmétique (virgule flottante)
  - ...

- L'analyse est effectuée en **abstrayant les variables du programme** et en exécutant l'**abstraction résultante**
  - ➠ l'**interprétation abstraite** peut conduire à une **fausse alerte**

CentraleSupélec
université PARIS-SACLAY

MODEL CHECKING

- Les propriétés à vérifier **ne sont pas des propriétés du programmes**
  - ce sont des propriétés **du modèles du programme**
- Habituellement, ces modèles désignent des **machines à états finis** (états et transitions)
- Les **propriétés à vérifier** sont souvent des **propriétés temporelles** (**accessibilité** d'un état)
- Les **Model checkers** fonctionnent **automatiquement**

CentraleSupélec
université PARIS-SACLAY

THEOREM PROVING

- C'est l'approche que je vais **développer dans ce cours**.
  - ➠ la méthode **Event-B**
- Les propriétés à prouver font partie des modèles :
  - ➠ **théorèmes** et **invariants**.
- La méthode **Event-B** se base sur la construction de modèles par **raffinements successifs**.
- Le modèle le plus concrêt (le **dernier niveau de raffinement**) est traduit **automatiquement en programme**.

CentraleSupélec
université PARIS-SACLAY

OUTLINE

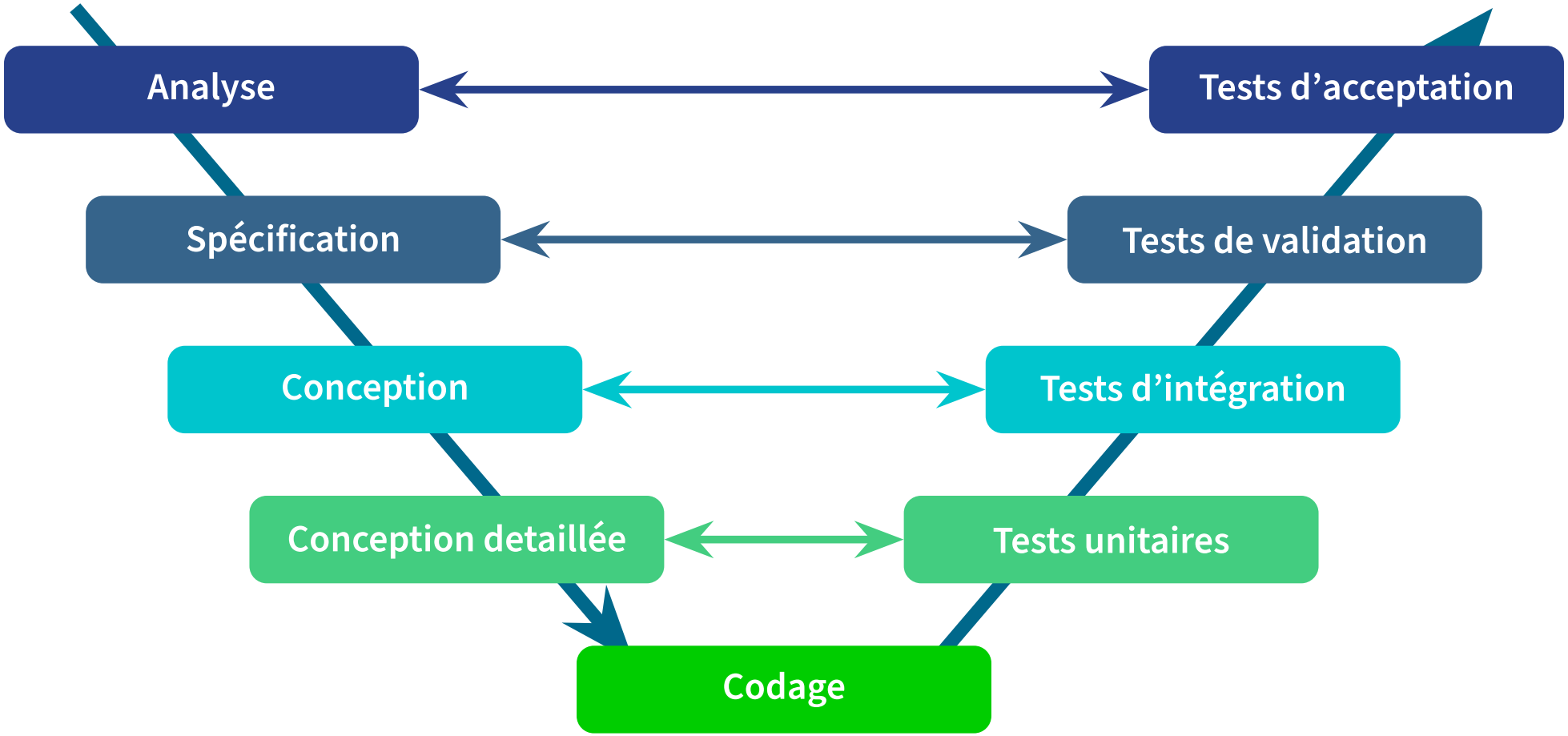
Back to the outline - Back to the begin

CentraleSupélec
université PARIS-SACLAY

AVANT LA MODÉLISATION

- Définir les principaux **objectifs** du futur système
- Définir les **exigences**
- Étude de **faisabilité**

CYCLE DE DÉVELOPPEMENT



Partie la plus faible → **Analyse** (le cahier des charges)

LE CAHIER DES CHARGES

- **Importance** de ce document (sa position dans le cycle de vie)
- Les cahiers des charges sont généralement **difficiles à exploiter**
- D'où très souvent la nécessité de **le réécrire**

CentraleSupélec
université PARIS-SACLAY

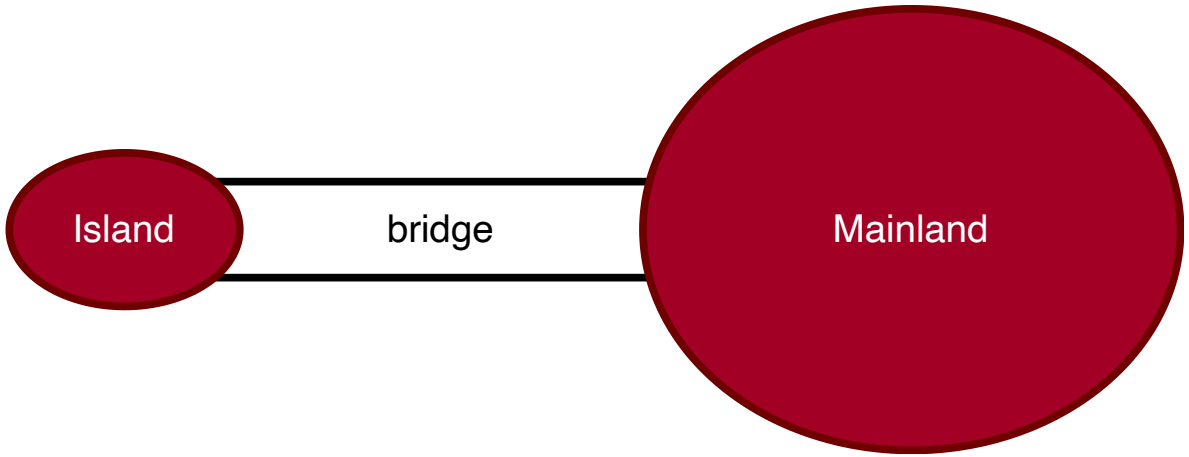
EXEMPLE D'UN CAHIER DES CHARGES

- Le système que nous allons construire est **un logiciel connecté à certains équipements**.
- Il existe deux types d'exigences :
  1. les exigences concernées par **le matériel**, labellisées **EQP**,
  2. les exigences concernés par **la fonction du système**, étiquetés **FUN**.
- La fonction de ce système est de **contrôler les voitures sur un pont étroit**. Ce pont est censé relier **le continent** à **une petite île**.

CentraleSupélec
université PARIS-SACLAY

EXEMPLE D'UN CAHIER DES CHARGES

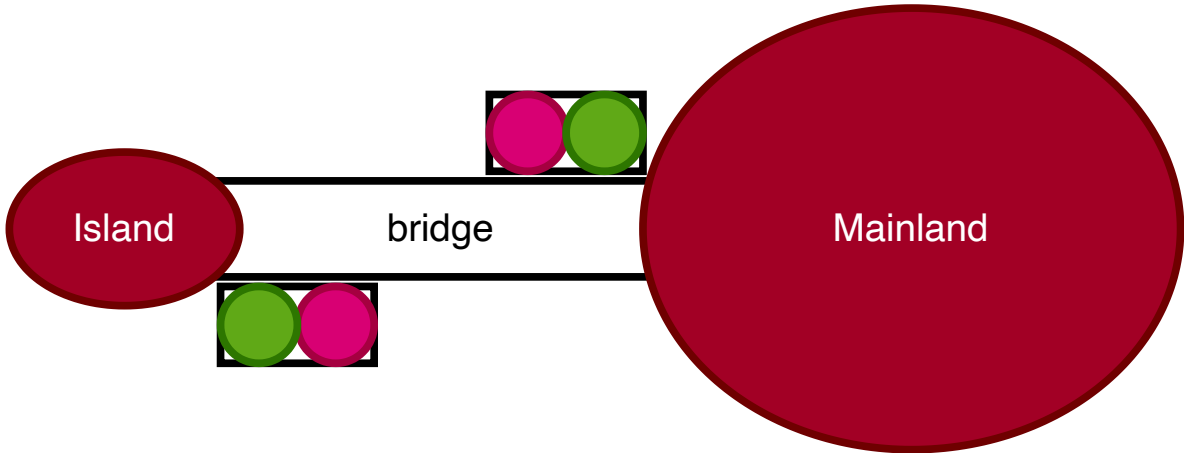
- **FUN-1** $\longrightarrow$ the system is controlling cars on a bridge between the mainland and an island.



CentraleSupélec
université PARIS-SACLAY

EXEMPLE D'UN CAHIER DES CHARGES

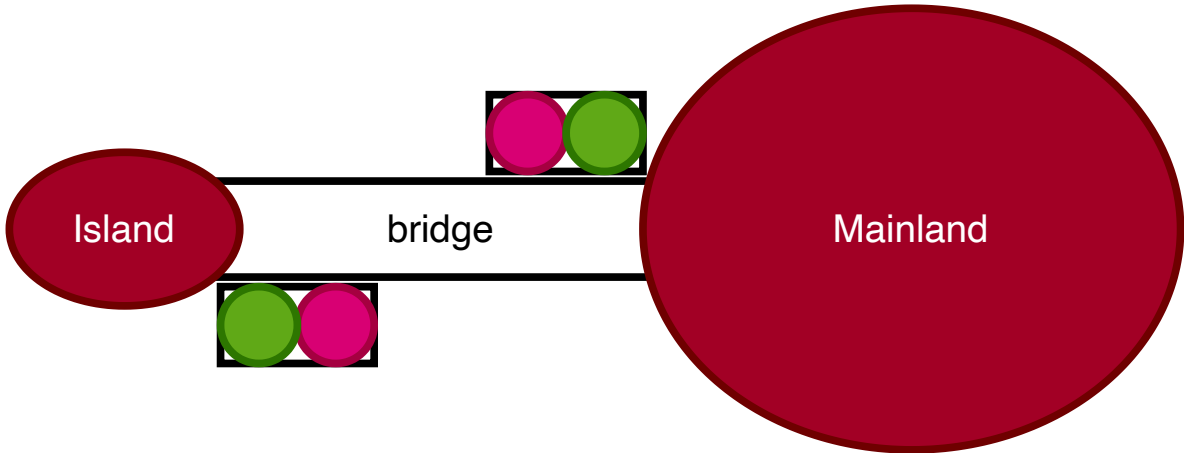
- **EQP-1** $\rightarrow$ the system has two traffic lights with two colors: green and red, one of the traffic lights is situated on the mainland and the other one on the island Both are close to the bridge.



CentraleSupélec
université PARIS-SACLAY

EXEMPLE D'UN CAHIER DES CHARGES

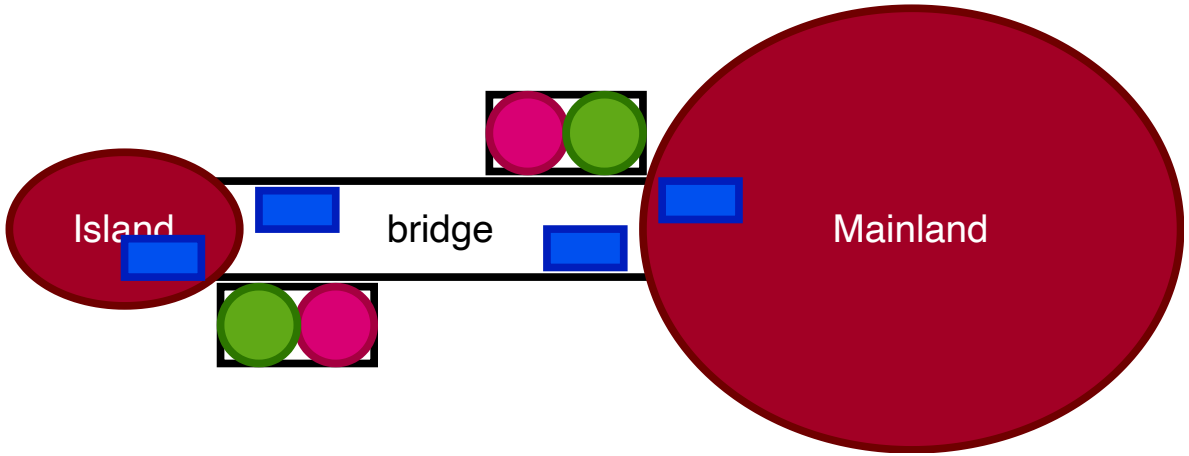
- **EQP-2** $\longrightarrow$ the traffic lights control the entrance to the bridge at both ends of it.
- **EQP-3** $\longrightarrow$ cars are not supposed to pass on a red traffic light, only on a green one.



CentraleSupélec
université PARIS-SACLAY

EXEMPLE D'UN CAHIER DES CHARGES

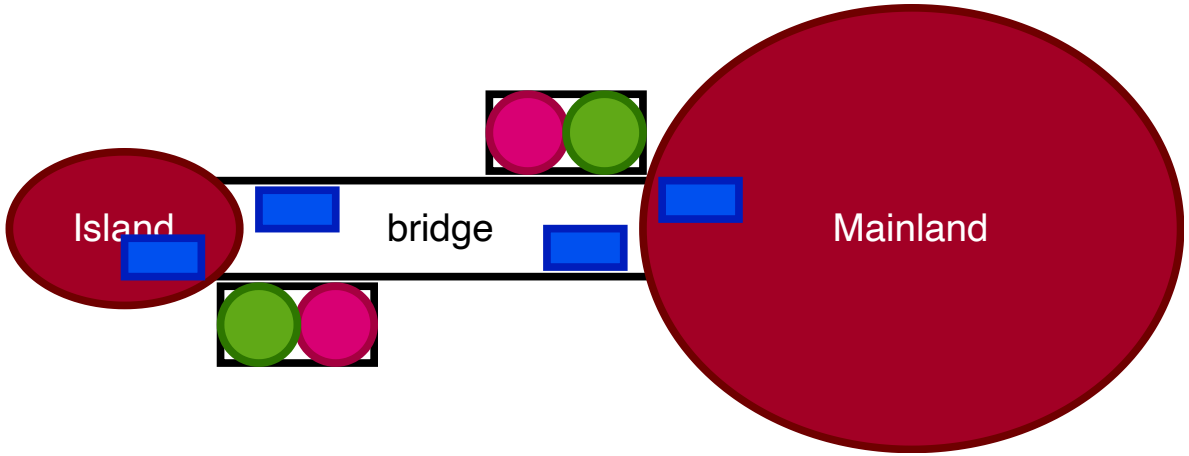
- **EQP-4** $\longrightarrow$ the system is equipped with four car sensors each with two states: on or off.
- **EQP-5** $\longrightarrow$ the sensors are used to detect the presence of cars entering or leaving the bridge.



CentraleSupélec
université PARIS-SACLAY

EXEMPLE D'UN CAHIER DES CHARGES

- **FUN-2** $\rightarrow$ the number of cars on the bridge and the island is limited.
- **FUN-3** $\rightarrow$ the bridge is one way or the other, not both at the same time.



CentraleSupélec
université PARIS-SACLAY

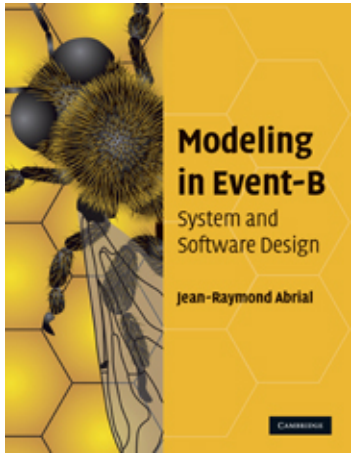
OUTLINE

Back to the outline - Back to the begin

CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

- The **Event-B method** is an evolution of the classical **B method**.
  - modeling a system by a set of events instead of operations.
- The **Event-B method** is a formal method based on first-order logic and set theory.
- The **Event-B method** is based on :
  - the notions of pre-conditions and post-conditions (**Hoare**),
  - the weakest pre-condition (**Dijkstra**),
  - and the calculus of substitution (**Abrial**).
- The **Event-B method** is adapted to analyse discrete systems.
  - offers the possibility of modelling **discrete behaviors**.



CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

THE STATE OF A MODEL

- A discrete model is first made of a state
- The state is represented by some constants and variables
- Constants are linked by some properties
- Variables are linked by some invariants
- Properties and invariants are written using set-theoretic expressions

CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

THE EVENTS OF A MODEL (TRANSITIONS)

- A discrete model is also made of a number of events
- An event is made of a guard and an action
- The guard denotes the enabling condition of the event
- The action denotes the way the state is modified by the event
- Guards and actions are written using set-theoretic expressions

CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

A MODEL SCHEMATIC VIEW

CONTEXT ctx_1
EXTENDS ctx_2

SETS s
CONSTANTS c
AXIOMS
$A(s, c)$
THEOREMS
$T(s, c)$
END

MACHINE mch_1
REFINES mch_2
SEES ctx_i

VARIABLES v
INVARIANTS
$I(s, c, v)$
THEOREMS
$T(s, c, v)$
EVENTS
$[events\_list]$
END

$event \mathrel{\widehat{=}}$
any x
where
$G(s, c, v, x)$
then
$BA(s, c, v, x, v')$
end

CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

OPERATIONAL INTERPRETATION

```
Initialize;
while (some events have true guards) {
  Choose one such event;
  Modify the state accordingly
}
```

- An event execution is supposed to take no time
- Thus, no two events can occur simultaneously
- When all events have false guards, the discrete system stops
- When some events have true guards, one of them is chosen non-deterministically and its action modifies the state
- The previous phase is repeated (if possible)

CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

COMMENTS ON THE OPERATIONAL INTERPRETATION

- Stopping is not necessary: a discrete system may run for ever
- This interpretation is just given here for informal understanding
- The meaning of such a discrete system will be given by the proofs which can be performed on it

CentraleSupélec
université PARIS-SACLAY

BUILDING LARGE COMPUTERIZED SYSTEMS

REFINEMENT

- Refinement allows us to build model gradually
- We shall build an ordered sequence of more precise models
- Each model is a refinement of the one preceding it
- A useful analogy: looking through a microscope
- Spatial as well as temporal extensions
- Data refinement

CentraleSupélec
université PARIS-SACLAY

PURPOSE OF THIS LECTURE

- To present an **example of system development**
- Our approach $\longrightarrow$ a series of **more and more accurate models**
- This approach is called **refinement**
- The models formalize the view of an **external observer**
- With each refinement **observer** *"zooms in"* to see more details

CentraleSupélec
université PARIS-SACLAY

PURPOSE OF THIS LECTURE

- Each model will be analyzed and **proved to be correct**
- The **aim** is to obtain a system that will be **correct by construction**
- The **correctness criteria** are formulated as **proof obligations**
- **Proofs** will be performed by using the **sequent calculus**
- **Inference rules** used in the sequent calculus will be **reviewed**

CentraleSupélec
université PARIS-SACLAY

THE EVENT-B METHOD

MODELS AND PROOF OBLIGATIONS

CONTEXT ctx_1
EXTENDS ctx_2

SETS s
CONSTANTS c
AXIOMS
$A(s, c)$
THEOREMS
$T(s, c)$
END

MACHINE mch_1
REFINES mch_2
SEES ctx_i

VARIABLES v
INVARIANTS
$I(s, c, v)$
THEOREMS
$T(s, c, v)$
EVENTS
$[events\_list]$
END

$event \mathrel{\widehat{=}}$
any x
where
$G(s, c, v, x)$
then
$BA(s, c, v, x, v')$
end

$$
\begin{aligned}
&A(s,c) \quad \vdash \quad T(s,c)\\
&A(s,c) \wedge I(s,c,v) \quad \vdash \quad T(s,c,v)\\
&A(s,c) \wedge I(s,c,v) \wedge G(s,c,v,x) \quad \vdash \quad \exists v'.\, BA(s,c,v,x,v')\\
&A(s,c) \wedge I(s,c,v) \wedge G(s,c,v,x) \wedge BA(s,c,v,x,v') \quad \vdash \quad I(s,c,v')\\
&\ldots
\end{aligned}
$$

CentraleSupélec
université PARIS-SACLAY

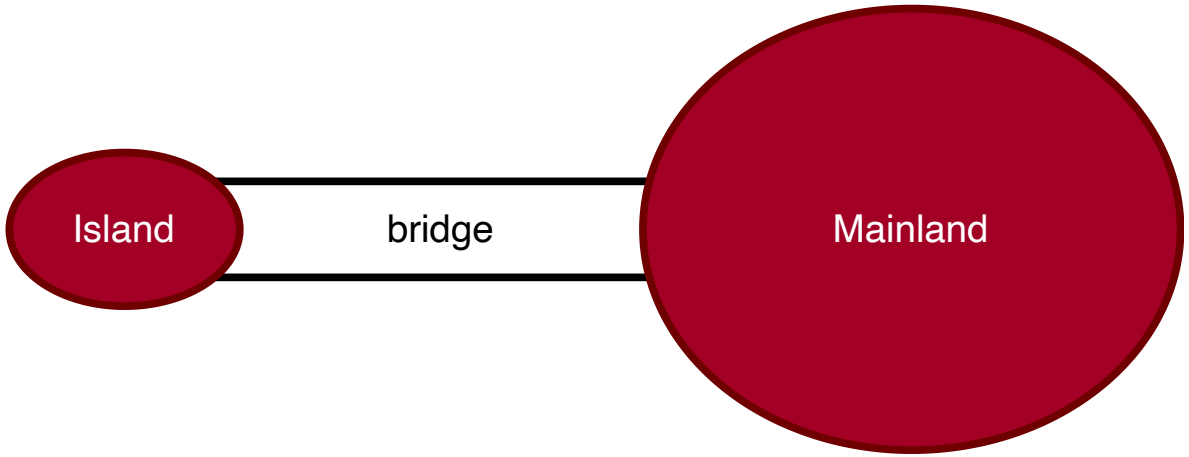
OUTLINE

Back to the outline - Back to the begin

CentraleSupélec
université PARIS-SACLAY

A REQUIREMENTS DOCUMENT

- The function of this system is to **control cars** on a **narrow bridge**.
- This bridge is supposed to link the **mainland** to a small **island**.
- **FUN-1** $\longrightarrow$ controlling cars on a bridge between the mainland and an island.
- **FUN-2** $\longrightarrow$ the number of cars on the bridge and the island is limited.
- **FUN-3** $\longrightarrow$ the bridge is one way or the other, not both at the same time.



CentraleSupélec
université PARIS-SACLAY

OUR REFINEMENT STRATEGY

- **Initial model** $\longrightarrow$ Limiting the number of cars (**FUN-2**)
- **First refinement** $\longrightarrow$ Introducing the one way bridge (**FUN-1**, **FUN-3**)

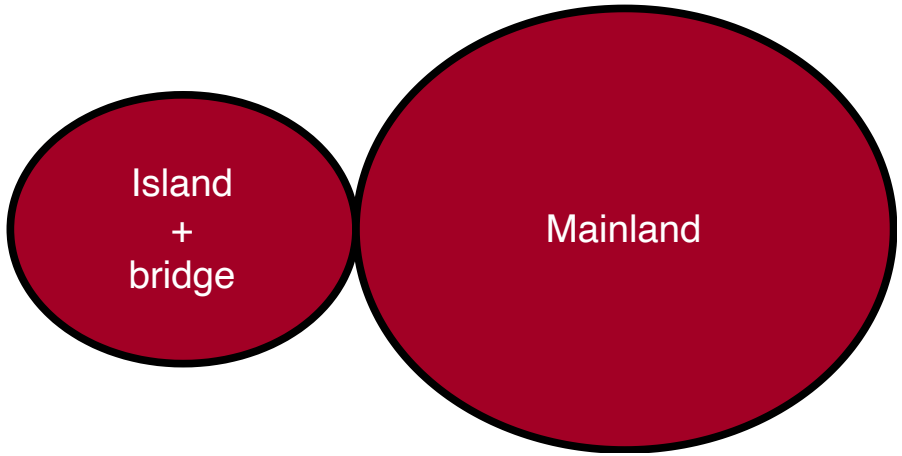
CentraleSupélec
université PARIS-SACLAY

OUR REFINEMENT STRATEGY

- **Initial model** $\longrightarrow$ Limiting the number of cars (**FUN-2**)
  - It is **very simple**
  - We do not even consider the bridge
  - We are just interested in the **pair "island-bridge"**
  - We are focusing **FUN-2** $\longrightarrow$ limited number of cars on island-bridge
- **First refinement** $\longrightarrow$ Introducing the one way bridge (**FUN-1**, **FUN-3**)

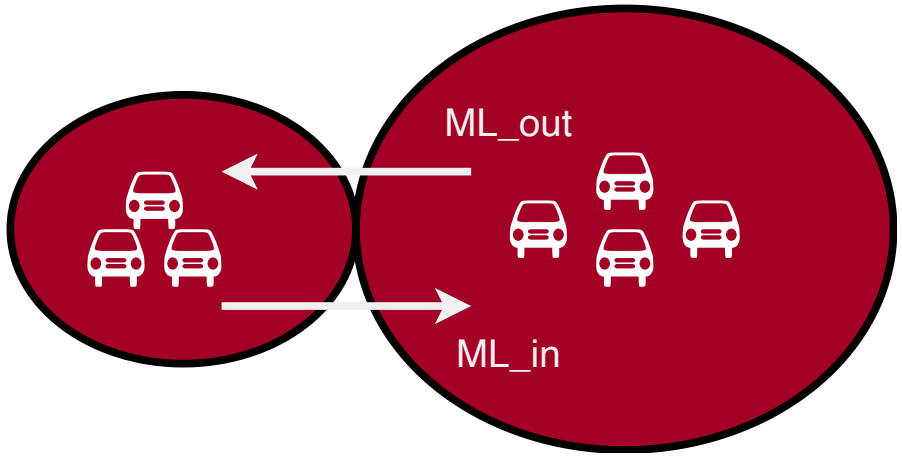
CentraleSupélec
université PARIS-SACLAY

A SITUATION AS SEEN FROM THE SKY



CentraleSupélec
université PARIS-SACLAY

TWO EVENTS THAT MAY BE OBSERVED



CentraleSupélec
université PARIS-SACLAY

FORMALIZING THE STATE

- **STATIC PART** of the state $\longrightarrow$ **constant** d with **axiom** axm0_1

CONSTANTS
d
AXIOMS
axm0_1: $d \in \mathbb{N}$

- d is the maximum number of cars allowed on the Island-Bridge
- axm0_1 states that d is a natural number
- Constant d is a member of the set $\mathbb{N} = \{0, 1, 2, \dots\}$

CentraleSupélec
université PARIS-SACLAY

FORMALIZING THE STATE

- **DYNAMIC PART** of the state $\longrightarrow$ **variable** n with **invariants** `inv0_1` and `inv0_2`

VARIABLES

n

INVARIANTS

`inv0_1`: $n \in \mathbb{N}$

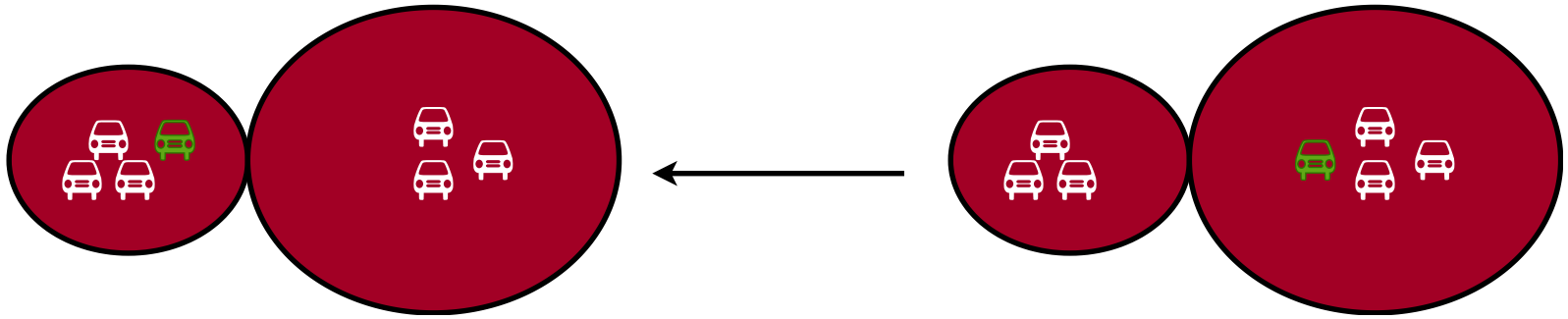
`inv0_2`: $n \leq d$

- n is the **effective number of cars** on the Island-Bridge
- n is a natural number (`inv0_1`)
- n is always smaller than or equal to d (`inv0_2`) $\longrightarrow$ this is **FUN 2**

CentraleSupélec
université PARIS-SACLAY

EVENT ML_out

- This is the **first transition** (or event) that can be **observed**
- A car is leaving the mainland and entering the Island-Bridge

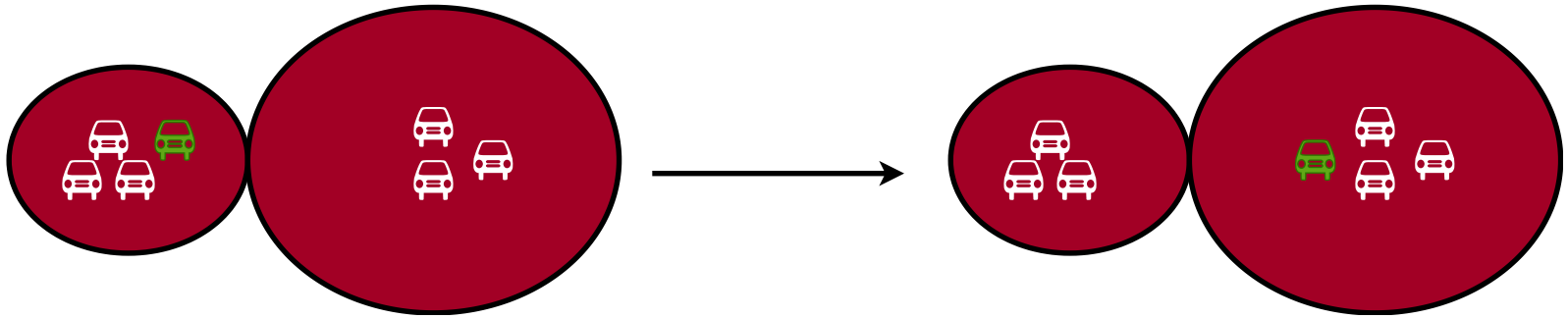


- The **number of cars** in the Island-Bridge is **incremented**

CentraleSupélec
université PARIS-SACLAY

EVENT ML_in

- We can also observe a **second transition** (or event)
- A car leaving the Island-Bridge and re-entering the mainland



- The **number of cars** in the Island-Bridge is **decremented**

CentraleSupélec
université PARIS-SACLAY

FORMALIZING THE TWO EVENTS (APPROXIMATION)

- An event is denoted by its **name** and its **action** (an assignment)
- Event `ML_out` **increments** the number of cars

$$\texttt{ML\_out} \mathrel{\widehat{=}}$$
then
$$\texttt{act0\_1: } n := n + 1$$
end

- Event `ML_in` **decrements** the number of cars

$$\texttt{ML\_in} \mathrel{\widehat{=}}$$
then
$$\texttt{act0\_1: } n := n - 1$$
end

CentraleSupélec
université PARIS-SACLAY

WHY AN APPROXIMATION?

- These events are approximations for **two reasons**:
  1. They might be **insufficient** at this stage because **not consistent with the invariant**
  2. They might be **refined** (made more precise) later
- We have to perform a **proof** in order to **verify this consistency**.

CentraleSupélec
université PARIS-SACLAY

INVARIANTS

- An invariant is a **constraint** on the allowed values of the variables
- An invariant **must hold on all reachable states** of a model
- To verify that this holds we must show that
  1. the invariant holds for **initial states**, and
  2. the invariant is **preserved by all events**
- We will formalize these two statements as **proof obligations (POs)**
- We need a **rigorous proof** showing that these POs indeed hold

CentraleSupélec
université PARIS-SACLAY

BEFORE-AFTER PREDICATES

- To each event can be associated a **before-after predicate**
- It describes the **relation** between the **values** of the variable(s) **just before** and **just after** the event occurrence
- The **before-value** is denoted by the **variable name**, say n
- The **after-value** is denoted by the **primed variable name**, say n'

CentraleSupélec
université PARIS-SACLAY

BEFORE-AFTER PREDICATES

EXAMPLE

- The **events**

ML_out $\widehat{=}$
  then
    act0_1: $n := n + 1$
  end

ML_in $\widehat{=}$
  then
    act0_1: $n := n - 1$
  end

- The corresponding **before-after predicates**

$$n' = n + 1 \qquad\qquad n' = n - 1$$

These representations are equivalent.

CentraleSupélec
université PARIS-SACLAY

ABOUT THE SHAPE OF THE BEFORE-AFTER PREDICATES

- The before-after predicates we have shown are **very simple**

$$n' = n + 1 \qquad\qquad n' = n - 1$$

- The after-value n' is defined as a **function** of the before-value n
- This is because the corresponding events are **deterministic**
- We shall also consider some **non-deterministic** events

$$n' \in \{n + 1, n + 2\}$$

CentraleSupélec
université PARIS-SACLAY

INTUITION
ABOUT INVARIANT PRESERVATION

- Let us consider invariant `inv0_1`

$$n \in \mathbb{N}$$

- And let us consider event `ML_out` with before-after predicate

$$n' = n + 1$$

- **Preservation of** `inv0_1` means that we have (just after `ML_out`):

$$n' \in \mathbb{N} \quad \text{that is} \quad n + 1 \in \mathbb{N}$$

CentraleSupélec
université PARIS-SACLAY

BEING MORE PRECISE

- Under hypothesis $n \in \mathbb{N}$ the conclusion $n + 1 \in \mathbb{N}$ holds
- This can be written as follows

$$n \in \mathbb{N} \quad \vdash \quad n + 1 \in \mathbb{N}$$

- This type of statement is called a **sequent**
- Sequent above $\longrightarrow$ invariant preservation proof obligation for `inv0_1`

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION

INVARIANT PRESERVATION

- We are given an event with before-after predicate $v' = E(c, v)$
- The following sequent expresses preservation of invariant $I_i(c, v)$

$$INV \ : \ A(c), I(c, v) \quad \vdash \quad I_i(c, E(c, v))$$

- It says $\longrightarrow$ $I_i(c, E(c, v))$ provable under hypotheses $A(c)$ and $I(c, v)$
- We have given the name INV to this proof obligation

CentraleSupélec
université PARIS-SACLAY

VERTICAL LAYOUT OF PROOF OBLIGATIONS

- The proof obligation

$$INV \ : \ A(c), I(c,v) \quad \vdash \quad I_i(c, E(c,v))$$

- can be re-written vertically as follows

Axioms	$A(c)$
Invariants	$I(c,v)$
$\vdash$	$\vdash$
Modified Invariant	$I_i(c, E(c,v))$

CentraleSupélec
université PARIS-SACLAY

BACK TO OUR EXAMPLE

- We have two events

ML_out $\widehat{=}$
  then
    act0_1: $n := n + 1$
  end

ML_in $\widehat{=}$
  then
    act0_1: $n := n - 1$
  end

- ... and two invariants

inv0_1: $n \in \mathbb{N}$

inv0_2: $n \leq d$

- Thus, we need to prove four proof obligations

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION FOR ML_out AND inv0_1

ML_out $\widehat{=}$
  then
    act0_1: $n := n + 1$ // $n' = n + 1$
  end

Axioms axm0_1	$d \in \mathbb{N}$
Invariant inv0_1	$n \in \mathbb{N}$
Invariant inv0_2	$n \leq d$
$\vdash$	$\vdash$
Modified Invariant inv0_1	$n + 1 \in \mathbb{N}$

This proof obligation is named **ML_out/inv0_1/INV**

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION FOR ML_out AND inv0_2

ML_out $\widehat{=}$
  then
    act0_1: $n := n + 1$ // $n' = n + 1$
  end

Axioms axm0_1	$d \in \mathbb{N}$
Invariant inv0_1	$n \in \mathbb{N}$
Invariant inv0_2	$n \leq d$
$\vdash$	$\vdash$
Modified Invariant inv0_2	$n + 1 \leq d$

This proof obligation is named **ML_out/inv0_2/INV**

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION FOR ML_in AND inv0_1

ML_in $\widehat{=}$
  then
    act0_1: $n := n - 1$ // $n' = n - 1$
  end

Axioms axm0_1	$d \in \mathbb{N}$
Invariant inv0_1	$n \in \mathbb{N}$
Invariant inv0_2	$n \leq d$
$\vdash$	$\vdash$
Modified Invariant inv0_1	$n - 1 \in \mathbb{N}$

This proof obligation is named **ML_in/inv0_1/INV**

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION FOR ML_in AND inv0_2

ML_in $\widehat{=}$
  then
    act0_1: $n := n - 1$ // $n' = n - 1$
  end

Axioms axm0_1	$d \in \mathbb{N}$
Invariant inv0_1	$n \in \mathbb{N}$
Invariant inv0_2	$n \leq d$
$\vdash$	$\vdash$
Modified Invariant inv0_2	$n - 1 \leq d$

This proof obligation is named: **ML_in/inv0_2/INV**

CentraleSupélec
université PARIS-SACLAY

SUMMARY OF PROOF OBLIGATIONS

ML_out/inv0_1/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$\vdash$

$n + 1 \in \mathbb{N}$

ML_out/inv0_2/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$\vdash$

$n + 1 \leq d$

ML_in/inv0_1/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$\vdash$

$n - 1 \in \mathbb{N}$

ML_in/inv0_2/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

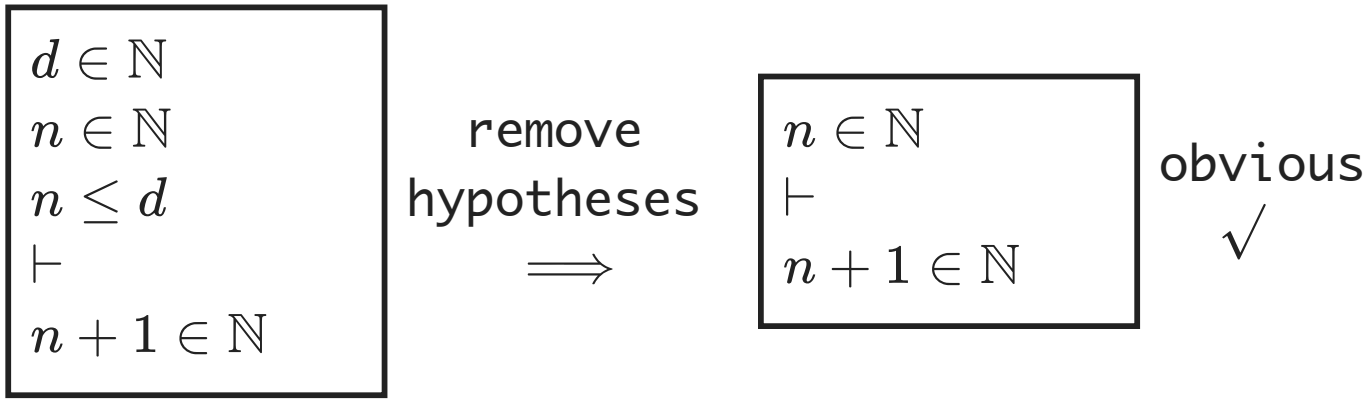
$\vdash$

$n - 1 \leq d$

CentraleSupélec

université PARIS-SACLAY

INFORMAL PROOF OF ML_out/inv0_1/INV



- In the first step, we remove some irrelevant hypotheses
- In the second and final step, we accept the sequent as it is
- We have implicitly applied inference rules
- For rigorous reasoning we will make these rules explicit

CentraleSupélec
université PARIS-SACLAY

INFERENCE RULES

MONOTONICITY OF HYPOTHESES

- The rule that removes hypotheses can be stated as follows:

$$\frac{H \vdash G}{H, H' \vdash G} \quad \text{MON}$$

- It expresses the monotonicity of the hypotheses

CentraleSupélec
université PARIS-SACLAY

SOME ARITHMETIC INFERENCE RULES

THE SECOND PEANO AXIOM

$$\frac{}{n \in \mathbb{N} \quad \vdash \quad n+1 \in \mathbb{N}} \quad \text{P2}$$

$$\frac{}{0 < n \quad \vdash \quad n-1 \in \mathbb{N}} \quad \text{P2'}$$

CentraleSupélec
université PARIS-SACLAY

MORE ARITHMETIC INFERENCE RULES

AXIOMS ABOUT ORDERING RELATIONS ON THE INTEGERS

$$\frac{}{n < m \quad \vdash \quad n + 1 \leq m} \quad \text{INC}$$

$$\frac{}{n \leq m \quad \vdash \quad n - 1 \leq m} \quad \text{DEC}$$

All inference rules implemented in Rodin are available here

CentraleSupélec
université PARIS-SACLAY

PROOFS

- A **proof** is a tree of sequents with axioms at the leaves.
- The rules applied to the leaves are axioms.
- Each sequent is labeled with (name of) proof rule applied to it.
- The sequent at the root of the tree is called the root sequent.
- The purpose of a proof is to establish the truth of its root sequent.

CentraleSupélec
université PARIS-SACLAY

A FORMAL PROOF OF ML_out/inv0_1/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n+1 \in \mathbb{N} \end{array}} \quad \text{MON} \implies \quad \boxed{\begin{array}{l} n \in \mathbb{N} \\ \vdash \\ n+1 \in \mathbb{N} \end{array}} \quad \text{P2} \ \surd$$

Proof requires only application of two rules $\longrightarrow$ MON and P2

CentraleSupélec
université PARIS-SACLAY

A FAILED PROOF ATTEMPT

ML_out/inv0_2/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n + 1 \leq d \end{array}} \quad \overset{\text{MON}}{\Longrightarrow} \quad \boxed{\begin{array}{l} n \leq d \\ \vdash \\ n + 1 \leq d \end{array}} \quad ?$$

- We put a **?** to indicate that we have no rule to apply
- **The proof fails** $\rightarrow$ we cannot conclude with rule INC ($n < d$ needed)

$$\frac{}{n < m \quad \vdash \quad n + 1 \leq m}\ \mathrm{INC}$$

CentraleSupélec
université PARIS-SACLAY

A FAILED PROOF ATTEMPT

ML_in/inv0_1/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n - 1 \in \mathbb{N} \end{array}} \quad \text{MON} \implies \quad \boxed{\begin{array}{l} n \in \mathbb{N} \\ \vdash \\ n - 1 \in \mathbb{N} \end{array}} \quad ?$$

- **The proof fails** $\rightarrow$ we cannot conclude with rule $\mathrm{P2'}$ ($0 < n$ needed)

$$\frac{}{0 < n \quad \vdash \quad n - 1 \in \mathbb{N}} \; \mathrm{P2'}$$

CentraleSupélec
université PARIS-SACLAY

A FORMAL PROOF OF ML_in/inv0_2/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n - 1 \leq d \end{array}} \quad \overset{\text{MON}}{\Longrightarrow} \quad \boxed{\begin{array}{l} n \leq d \\ \vdash \\ n - 1 \leq d \end{array}} \quad \begin{array}{c} \text{DEC} \\ \surd \end{array}$$

$$\frac{}{n \leq m \quad \vdash \quad n - 1 \leq m} \ \text{DEC}$$

REASONS FOR PROOF FAILURE

- We needed hypothesis $n < d$ to prove ML_out/inv0_2/INV
- We needed hypothesis $0 < n$ to prove ML_in/inv0_1/INV

ML_out $\widehat{=}$
  then
    act0_1: $n := n + 1$
  end

ML_in $\widehat{=}$
  then
    act0_1: $n := n - 1$
  end

- We are going to add $n < d$ as a guard to event ML_out
- We are going to add $0 < n$ as a guard to event ML_in

CentraleSupélec
université PARIS-SACLAY

IMPROVING THE EVENTS
INTRODUCING GUARDS

ML_out $\widehat{=}$
  when
    grd0_1: $n < d$
  then
    act0_1: $n := n + 1$
  end

ML_in $\widehat{=}$
  when
    grd0_1: $0 < n$
  then
    act0_1: $n := n - 1$
  end

- We are adding guards to the events
- The guard is the necessary condition for an event to *occur*

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION

GENERAL INVARIANT PRESERVATION

- Given c with axioms $A(c)$ and v with invariants $I(c, v)$
- Given an event with guard $G(c, v)$ and b-a predicate $v' = E(c, v)$
- We modify the **Invariant Preservation PO** as follows:

Axioms	$A(c)$
Invariants	$I(c, v)$
Guard of the event	$G(c, v)$
$\vdash$	$\vdash$
Modified Invariant	$I_i(c, E(c, v))$

CentraleSupélec
université PARIS-SACLAY

A FORMAL PROOF OF ML_out/inv0_1/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n + 1 \in \mathbb{N} \end{array}} \quad \begin{array}{c} \text{MON} \\ \Longrightarrow \end{array} \quad \boxed{\begin{array}{l} n \in \mathbb{N} \\ \vdash \\ n + 1 \in \mathbb{N} \end{array}} \quad \begin{array}{c} \text{P2} \\ \surd \end{array}$$

Adding new assumptions to a sequent does not affect its provability

CentraleSupélec
université PARIS-SACLAY

A FORMAL PROOF OF ML_out/inv0_2/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n + 1 \leq d \end{array}} \quad \text{MON} \implies \quad \boxed{\begin{array}{l} n < d \\ \vdash \\ n + 1 \leq d \end{array}} \quad \text{INC} \ \surd$$

- Now we can conclude the proof using rule INC

$$\frac{}{n < m \quad \vdash \quad n + 1 \leq m} \ \mathrm{INC}$$

CentraleSupélec
université PARIS-SACLAY

A FORMAL PROOF OF ML_in/inv0_1/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ 0 < n \\ \vdash \\ n - 1 \in \mathbb{N} \end{array}} \quad \text{MON} \implies \boxed{\begin{array}{l} 0 < n \\ \vdash \\ n - 1 \in \mathbb{N} \end{array}} \quad \text{P2'} \surd$$

- Now we can conclude the proof using rule $\mathrm{P2'}$

$$\frac{}{0 < n \quad \vdash \quad n - 1 \in \mathbb{N}} \ \mathrm{P2'}$$

CentraleSupélec
université PARIS-SACLAY

A FORMAL PROOF OF ML_in/inv0_2/INV

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ 0 < n \\ \vdash \\ n - 1 \leq d \end{array}} \quad \text{MON} \implies \boxed{\begin{array}{l} n \leq d \\ \vdash \\ n - 1 \leq d \end{array}} \quad \text{DEC} \ \surd$$

Again, the proof still works after the addition of a new assumption

CentraleSupélec
université PARIS-SACLAY

RE-PROVING THE EVENTS NO PROOFS FAIL

ML_out/inv0_1/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$n < d$

$\vdash$

$n + 1 \in \mathbb{N}$

ML_out/inv0_2/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$n < d$

$\vdash$

$n + 1 \leq d$

ML_in/inv0_1/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$0 < n$

$\vdash$

$n - 1 \in \mathbb{N}$

ML_in/inv0_2/INV

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

$0 < n$

$\vdash$

$n - 1 \leq d$

CentraleSupélec

université PARIS-SACLAY

INITIALISATION

- Our system must be initialized (with no car in the island-bridge)
- The initialisation event is never guarded
- It does not mention any variable on the right hand side of $:=$
- Its before-after predicate is just an after predicate

```
init ≙
  begin
    init0_1: n := 0
  end
```

After predicate $\Longrightarrow$ $n' = 0$

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION
INVARIANT ESTABLISHMENT

- Given c with axioms $A(c)$ and v with invariants $I(c, v)$
- Given an init event with after predicate $v' = K(c)$
- The Invariant Establishment PO is the following:

$$\begin{array}{ll} \text{Axioms} & A(c) \\ \vdash & \vdash \\ \text{Modified Invariant} & I_i(c, K(c)) \end{array}$$

CentraleSupélec
université PARIS-SACLAY

APPLYING THE INVARIANT ESTABLISHMENT PO

axm0_1	$d \in \mathbb{N}$	
$\vdash$	$\vdash$	inv0_1/INV
Modified inv0_1	$0 \in \mathbb{N}$	

axm0_1	$d \in \mathbb{N}$	
$\vdash$	$\vdash$	inv0_2/INV
Modified inv0_2	$0 \leq d$	

CentraleSupélec
université PARIS-SACLAY

MORE ARITHMETIC INFERENCE RULES

- First Peano Axiom

$$\frac{}{\vdash \quad 0 \in \mathbb{N}} \ \mathrm{P1}$$

- Third Peano Axiom (slightly modified)

$$\frac{}{n \in \mathbb{N} \quad \vdash \quad 0 \leq n} \ \mathrm{P3}$$

CentraleSupélec
université PARIS-SACLAY

PROOFS OF INVARIANT ESTABLISHMENT

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ \vdash \\ 0 \in \mathbb{N} \end{array}} \quad \begin{array}{c} \text{MON} \\ \Longrightarrow \end{array} \quad \boxed{\begin{array}{l} \vdash \\ 0 \in \mathbb{N} \end{array}} \quad \begin{array}{c} \text{P1} \\ \surd \end{array}$$

$$\boxed{\begin{array}{l} d \in \mathbb{N} \\ \vdash \\ 0 \leq d \end{array}} \quad \begin{array}{c} \text{P3} \\ \surd \end{array}$$

CentraleSupélec
université PARIS-SACLAY

A MISSING REQUIREMENT

- It is possible for the system to be blocked if both guards are false
- We do not want this to happen
- We figure out that one important requirement was missing
- **FUN-4** $\longrightarrow$ Once started, the system should work for ever (Deadlock Freedom)

CentraleSupélec
université PARIS-SACLAY

PROOF OBLIGATION

THE THEOREM PO RULE

- Given c with axioms $A(c)$ and v with invariants $I(c, v)$
- Given the theorem $Th(c, v)$
- Given the guards $G_1(c, v), \ldots, G_m(c, v)$ of the events
- We have to prove the following:

$$A(c) \quad I(c, v) \quad \vdash \quad Th(c, v)$$

$$A(c) \quad I(c, v) \quad \vdash \quad G_1(c, v) \ \vee \ \ldots \ \vee \ G_m(c, v)$$

CentraleSupélec
université PARIS-SACLAY

APPLYING THE DEADLOCK FREEDOM PO

`axm0_1`	$d \in \mathbb{N}$
`inv0_1`	$n \in \mathbb{N}$
`inv0_2`	$n \leq d$
$\vdash$	$\vdash$
Disjunction of guards	$n < d \ \lor \ 0 < n$

- This cannot be proved with the inference rules we have so far
- $n \leq d$ can be replaced by $n = d \lor n < d$
- We continue our proof by a case analysis:
  - case 1: $n = d$
  - case 2: $n < d$

CentraleSupélec
université PARIS-SACLAY

INFERENCE RULES FOR DISJUNCTION

- Proof by case analysis

$$\frac{H, P \vdash R \qquad H, Q \vdash R}{H, P \vee Q \vdash R} \text{ OR\_L}$$

- Choice for proving a disjunctive goal

$$\frac{H \vdash P}{H \vdash P \vee Q} \text{ OR\_R1}$$

$$\frac{H \vdash Q}{H \vdash P \vee Q} \text{ OR\_R2}$$

CentraleSupélec
université PARIS-SACLAY

PROOF OF DEADLOCK FREEDOM

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n < d \ \vee \ 0 < n \end{array}$$

MON $\Longrightarrow$

$$\begin{array}{l} n \leq d \\ \vdash \\ n < d \ \vee \ 0 < n \end{array}$$

OR_L $\Longrightarrow$

$$\begin{array}{l} n < d \\ \vdash \\ n < d \ \vee \ 0 < n \end{array}$$

$$\begin{array}{l} n = d \\ \vdash \\ n < d \ \vee \ 0 < n \end{array}$$

$$\begin{array}{l} n < d \\ \vdash \\ n < d \ \vee \ 0 < n \end{array}$$

OR_R1 $\Longrightarrow$

$$\begin{array}{l} n < d \\ \vdash \\ n < d \end{array}$$

? $\Longrightarrow$ seems to be obvious

$$\begin{array}{l} n = d \\ \vdash \\ n < d \ \vee \ 0 < n \end{array}$$

? $\Longrightarrow$ can be (partially) solved by applying the equality

CentraleSupélec
université PARIS-SACLAY

MORE INFERENCE RULES

IDENTITY AND EQUALITY

- The identity axiom (conclusion holds by hypothesis)

$$\frac{}{P \vdash P}\ \text{HYP}$$

- Rewriting an equality (**EQ_LR**) and reflexivity of equality (**EQL**)

$$\frac{H(F), E = F \vdash P(F)}{H(E), E = F \vdash P(E)}\ \text{EQ\_LR}$$

$$\frac{}{\vdash E = E}\ \text{EQL}$$

CentraleSupélec
université PARIS-SACLAY

PROOF OF DEADLOCK FREEDOM

$n < d$
$\vdash$
$n < d \ \vee \ 0 < n$

OR_R1
$\Longrightarrow$

$n < d$
$\vdash$
$n < d$

HYP
$\surd$

$n = d$
$\vdash$
$n < d \ \vee \ 0 < n$

EQ_LR
$\Longrightarrow$

$\vdash$
$d < d \ \vee \ 0 < d$

OR_R2
$\Longrightarrow$

$\vdash$
$0 < d$ **?**

- We still have a problem $\longrightarrow$ d must be positive!

CentraleSupélec
université PARIS-SACLAY

ADDING THE FORGOTTEN AXIOM

- If $d = 0$, then no car can ever enter the Island-Bridge

CONSTANTS

d

AXIOMS

axm0_1: $d \in \mathbb{N}$

axm0_2: $0 < d$

CentraleSupélec

université PARIS-SACLAY

INITIAL MODEL
CONCLUSION

- Thanks to the proofs, we discovered 3 errors
- They were corrected by:
  - adding guards to both events
  - adding an axiom
- The interaction of modeling and proving is an essential element of Formal Methods with Proofs

CentraleSupélec
université PARIS-SACLAY

THANK YOU

PDF version of the slides

Back to the begin - Back to the outline

CentraleSupélec
université PARIS-SACLAY