

Tutorial Modelling Software-based Systems

...

Tutorial 4 : Checking annotated algorithms using Event-B

Dominique Méry

26 janvier 2026

Exercice 1 *contract-annotations*

For each case, define a contract for checking the soundness or the unsoundness of the annotation.

$$\forall x, y, , x', y'. P_\ell(x, y) \wedge cond_{\ell, \ell'}(x, y) \wedge (x', y') = f_{\ell, \ell'}(x, y) \Rightarrow P_{\ell'}(x', y')$$

You will use a context and a machine for expressing these conditions.

—

$\ell_1 : x = 10 \wedge y = z+x \wedge z = 2 \cdot x$ $y := z+x$ $\ell_2 : x = 10 \wedge y = x+2 \cdot 10$
--

— We assume that p is a prime number.

$\ell_1 : x = 2^p \wedge y = 2^{p+1} \wedge x \cdot y = 2^{2 \cdot p + 1}$ $x := y+x+2^x$ $\ell_2 : x = 5 \cdot 2^p \wedge y = 2^{p+1}$

—

$\ell_1 : x = 1 \wedge y = 12$ $x := 2 \cdot y$ $\ell_2 : x = 1 \wedge y = 24$
--

—

$\ell_1 : x = 11 \wedge y = 13$ $z := x; x := y; y := z;$ $\ell_2 : x = 26/2 \wedge y = 33/3$

Exercice 2 *(contract-simple)*

Let the following partially annotated algorithm :

precondition : $x = x_0 \wedge x_0 \in \mathbb{N}$ postcondition : $x = 0$ $\ell_0 : \{x = x_0 \wedge x_0 \in \mathbb{N}\}$ while $0 < x$ do <table border="1" style="margin-left: 20px; margin-right: 20px;"> <tr> <td style="padding: 10px;"> $\ell_1 : \{0 < x \leq x_0 \wedge x_0 \in \mathbb{N}\}$ $x := x-1;$ $\ell_2 : \{0 \leq x \leq x_0 \wedge x_0 \in \mathbb{N}\}$ </td> </tr> </table> $\ell_3 : \{x = 0\}$	$\ell_1 : \{0 < x \leq x_0 \wedge x_0 \in \mathbb{N}\}$ $x := x-1;$ $\ell_2 : \{0 \leq x \leq x_0 \wedge x_0 \in \mathbb{N}\}$
$\ell_1 : \{0 < x \leq x_0 \wedge x_0 \in \mathbb{N}\}$ $x := x-1;$ $\ell_2 : \{0 \leq x \leq x_0 \wedge x_0 \in \mathbb{N}\}$	

Algorithm 1: Exercice 2

Question 2.1 Translate each transition ℓ, ℓ' into an event modifying the variables according to the statements.

Question 2.2 Define an invariant attaching to each label an assertion satisfied at the control point.

Question 2.3 Verify proof obligations and deduce that the algorithm is partially correct.

Question 2.4 Prove that the algorithm has no runtime error.

Exercice 3 (*contract-sqrareroot*)

Let the following annotated invariant.

```

precondition :  $x \in \mathbb{N}$ 
postcondition :  $z^2 \leq x \wedge x < (z+1)^2$ 
local variables :  $y_1, y_2, y_3 \in \mathbb{N}$ 

pre :  $\{x \in \mathbb{N}\}$ 
post :  $\{z \cdot z \leq x \wedge x < (z+1) \cdot (z+1)\}$ 
 $\ell_0 : \{x \in \mathbb{N} \wedge z \in \mathbb{Z} \wedge y_1 \in \mathbb{Z} \wedge y_2 \in \mathbb{Z} \wedge y_3 \in \mathbb{Z}\}$ 
 $(y_1, y_2, y_3) := (0, 1, 1);$ 
 $\ell_1 : \{y_2 = (y_1+1) \cdot (y_1+1) \wedge y_3 = 2 \cdot y_1 + 1 \wedge y_1 \cdot y_1 \leq x\}$ 
while  $y_2 \leq x$  do
     $\ell_2 : \{y_2 = (y_1+1) \cdot (y_1+1) \wedge y_3 = 2 \cdot y_1 + 1 \wedge y_2 \leq x\}$ 
     $(y_1, y_2, y_3) := (y_1+1, y_2+y_3+2, y_3+2);$ 
     $\ell_3 : \{y_2 = (y_1+1) \cdot (y_1+1) \wedge y_3 = 2 \cdot y_1 + 1 \wedge y_1 \cdot y_1 \leq x\}$ 
;
 $\ell_4 : \{y_2 = (y_1+1) \cdot (y_1+1) \wedge y_3 = 2 \cdot y_1 + 1 \wedge y_1 \cdot y_1 \leq x \wedge x < y_2\}$ 
 $z := y_1;$ 
 $\ell_5 : \{y_2 = (y_1+1) \cdot (y_1+1) \wedge y_3 = 2 \cdot y_1 + 1 \wedge y_1 \cdot y_1 \leq x \wedge x < y_2 \wedge z = y_1 \wedge z \cdot z \leq x \wedge x < (z+1) \cdot (z+1)\}$ 

```

Algorithm 2: *sqrareroot* annotée Exercice ??

Question 3.1 Translate each transition ℓ, ℓ' into an event modifying the variables according to the statements.

Question 3.2 Define an invariant attaching to each label an assertion satisfied at the control point.

Question 3.3 Verify proof obligations and deduce that the algorithm is partially correct.

Question 3.4 Prove that the algorithm has no runtime error.

Exercice 4 (*contract-maximum*)

Soit l'algorithme suivant annoté partiellement :

Question 4.1 Translate each transition ℓ, ℓ' into an event modifying the variables according to the statements.

Question 4.2 Define an invariant attaching to each label an assertion satisfied at the control point.

Question 4.3 Verify proof obligations and deduce that the algorithm is partially correct.

Question 4.4 Prove that the algorithm has no runtime error.

```

/* algorithme de calcul du maximum avec une boucle while de l'exercice ?? */
precondition :  $\left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right)$ 
postcondition :  $\left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f) \wedge \\ (\forall j \cdot j \in 0..n-1 \Rightarrow f(j) \leq m) \end{array} \right)$ 
local variables :  $i \in \mathbb{Z}$ 

 $\ell_0 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in \mathbb{Z} \wedge m \in \mathbb{Z} \right\}$ 
 $m := f(0);$ 
 $\ell_1 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in \mathbb{Z} \wedge m = f(0) \right\}$ 
 $i := 1;$ 
 $\ell_2 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i = 1 \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f[0..i-1]) \wedge \\ (\forall j \cdot j \in 0..i-1 \Rightarrow f(j) \leq m) \end{array} \right) \right\}$ 
while  $i < n$  do
   $\ell_3 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in 1..n-1 \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f[0..i-1]) \wedge \\ (\forall j \cdot j \in 0..i-1 \Rightarrow f(j) \leq m) \end{array} \right) \right\}$ 
  if  $f(i) > m$  then
     $\ell_4 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in 1..n-1 \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f[0..i-1]) \wedge \\ (\forall j \cdot j \in 0..i-1 \Rightarrow f(j) \leq m) \end{array} \right) \wedge \right.$ 
     $f(i) > m \}$ 
     $m := f(i);$ 
     $\ell_5 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in 1..n-1 \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f[0..i]) \wedge \\ (\forall j \cdot j \in 0..i \Rightarrow f(j) \leq m) \end{array} \right) \right\}$ 
  ;
   $\ell_6 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in \mathbb{Z} \wedge i \in 1..n-1 \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f[0..i]) \wedge \\ (\forall j \cdot j \in 0..i \Rightarrow f(j) \leq m) \end{array} \right) \right\}$ 
   $i++;$ 
   $\ell_7 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i \in 2..n \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f[0..i-1]) \wedge \\ (\forall j \cdot j \in 0..i-1 \Rightarrow f(j) \leq m) \end{array} \right) \right\}$ 
;
 $\ell_8 : \left\{ \left( \begin{array}{l} n \in \mathbb{N} \wedge \\ n \neq 0 \wedge \\ f \in 0..n-1 \rightarrow \mathbb{N} \end{array} \right) \wedge i = n \wedge \left( \begin{array}{l} m \in \mathbb{N} \wedge \\ m \in \text{ran}(f) \wedge \\ (\forall j \cdot j \in 0..n-1 \Rightarrow f(j) \leq m) \end{array} \right) \right\}$ 

```

Algorithme 3: Algorithme du manimum d'une liste annoté Exercice 4