

Modelling Software-based Systems

Lecture 2 Proof Obligation Generation

Telecom Nancy (IL et LE)

Dominique Méry
Telecom Nancy, Université de Lorraine

24 novembre 2025
dominique.mery@loria.fr

① Overview of machines, contexts and proof obligations

② Proof Obligations for Contexts and Machines

PO thm/THM (context)

PO th/THM (machine)

PO evt/inv/INV

PO evt/act/FIS

③ Proof Obligations for Refinement

PO evt/grd/GRD

PO evt/act/SIM

PO evt/NAT

PO NAT

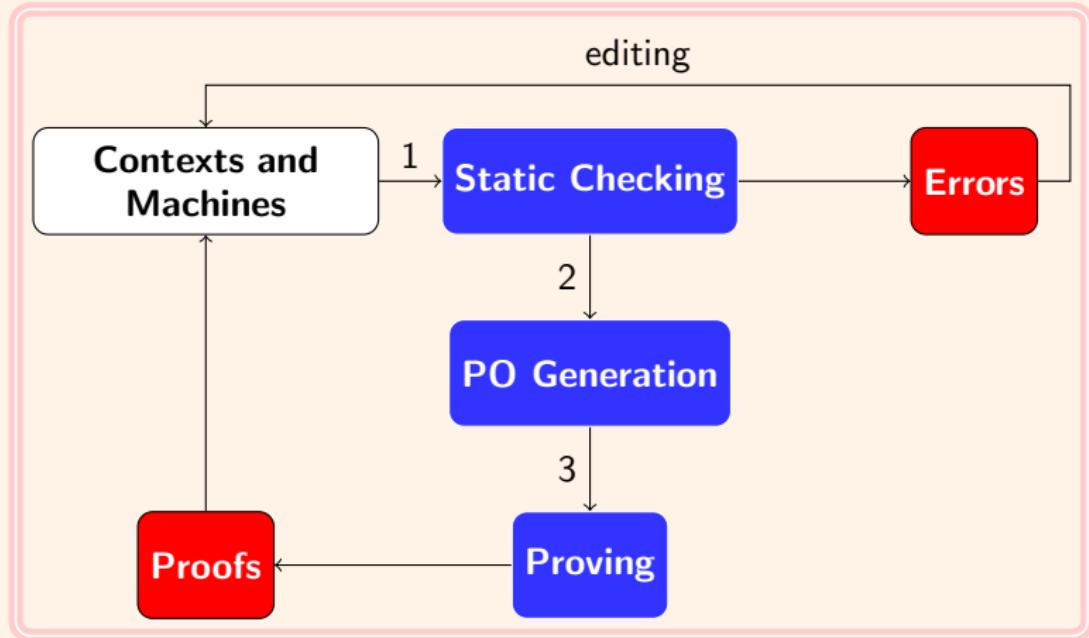
PO evt/VAR (arithmetic)

PO evt/VAR (set-theoretic)

PO evt/x/WFIS

- ① Overview of machines, contexts and proof obligations
- ② Proof Obligations for Contexts and Machines
- ③ Proof Obligations for Refinement

Analysis of the Event-B Models



Machines en Event B

MACHINE

m

REFINES

am

SEES

c

VARIABLES

u

INVARIANTS

$I(s, c, u)$

THEOREMS

$Q(s, c, u)$

VARIANT

$exp(s, c, u)$

EVENTS

INITIALIZATION

...

e

...

END

Machines en Event B

MACHINE	
<i>m</i>	
REFINES	
<i>am</i>	
SEES	
<i>c</i>	
VARIABLES	
<i>u</i>	
INVARIANTS	
<i>I(s, c, u)</i>	
THEOREMS	
<i>Q(s, c, u)</i>	
VARIANT	
<i>exp(s, c, u)</i>	
EVENTS	
<i>INITIALIZATION</i>	
...	
<i>e</i>	
...	
END	

- $\Gamma(m)$: environment for the machine m defined by the context c and it provides a list of seen axioms $Ax(s, c)$ and a list of seen theorems $Th(s, c)$ for the sets s and constants c .
- $\Gamma(m) \vdash \forall u.\text{INIT}(s, c, u) \Rightarrow I(s, c, u)$
- For each event e in E :
 $\Gamma(m) \vdash \forall u, u'.I(s, c, u) \wedge BA(e)(u, u') \Rightarrow I(u')$
- For each event e in E :
 $\Gamma(m) \vdash \forall u.I(s, c, u) \wedge GRD(e)(s, c, u) \Rightarrow \exists u'.BA(e)(u, u')$
- $\Gamma(m) \vdash \forall u.I(s, c, u) \Rightarrow Q(s, c, u)$
- Generated proof obligations are derived from those conditions.

Checking the well formation of Event-B expressions

- Event-B expressions are contexts, machines, properties, equations, set-theoretical expressions ...
- e is an Event-B expression and $\text{wd}(e)$ is a logical property expressing the well definition of e .
- $\text{wd}(1 = 2) \triangleq \text{wd}(1) \wedge \text{wd}(2)$
- $\text{wd}(a/b) \triangleq b \neq 0 \wedge \text{wd}(a) \wedge \text{wd}(b)$
- $\text{wd}(f(g)) \triangleq g \in \text{dom}(f) \wedge f \in A \rightarrow B$

- ① Overview of machines, contexts and proof obligations
- ② Proof Obligations for Contexts and Machines
- ③ Proof Obligations for Refinement

CONTEXTS c **EXTENDS** ac **SETS** s **CONSTANTS** c **AXIOMS** $Ax(s, c)$ **THEOREMS** $th_1 : P_1(s, c)$ \dots $th_n : P_n(s, c)$ $th : P(s, c)$ \dots **END**

s	<i>seen sets</i>
c	<i>seen constants</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>previous proved theorems</i>
$P(s, c)$	$PTh(s, c) = \{P_i(s, c) i \text{ } 1..n\}$ <i>property over s and c</i>

PO th/THM

$$Ax(s, c), Th(s, c) \vdash P(s, c)$$

PO th/THM (machine)

MACHINE
m
\dots
VARIABLES
u
INVARIANTS
$I(s, c, u)$
THEOREMS
$Q(s, c, u)$
$th : P(s, c, u)$
\dots
END

s	<i>seen sets</i>
c	<i>seen constants</i>
u	<i>variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>invariants</i>
$Q(s, c, u)$	<i>theorems</i>
$P(s, c, u)$	<i>property over s, c and u</i>

PO th/THM

$$Ax(s, c), Th(s, c), I(s, c, u) \vdash P(s, c, u)$$

PO evt/inv/INV

```
EVENT evt
  ANY x WHERE
    G(x, s, c, u)
  THEN
    u : |BAP(x, s, c, u, u')
  END
```

$$\begin{aligned} BA(\text{evt}) &\equiv \\ \exists x. & \left(\begin{array}{l} \wedge G(x, s, c, u) \\ \wedge BAP(x, s, c, u, u') \end{array} \right) \\ GRD(\text{evt}) &\equiv G(x, s, c, u) \\ ACT(\text{evt}) &\equiv BAP(x, s, c, u, u') \end{aligned}$$

<i>s</i>	<i>seen sets</i>
<i>c</i>	<i>seen constants</i>
<i>u</i>	<i>variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>invariants</i>
$Q(s, c, u)$	<i>theorems</i>
<i>evt</i>	<i>event name</i>
<i>x</i>	<i>event parameter</i>
$G(x, s, c, u)$	<i>event guard</i>
$BAP(x, s, c, u, u')$	<i>event before-after predicate</i>
$inv : inv(s, c, u')$	<i>specific modified invariant</i>

PO evt/inv/INV

$$Ax(s, c), Th(s, c), I(s, c, u), G(x, s, c, u), BAP(x, s, c, u, u') \vdash inv(s, c, u')$$

PO Q/THM $Ax(s, c), Th(s, c), I(s, c, u) \vdash Q(s, c, u)$

PO evt/act/FIS

```
EVENT evt
ANY x WHERE
  G(x, s, c, u)
THEN
  u : |BAP(x, s, c, u, u')
END
```

$BA(\text{evt}) \hat{=} \left(\begin{array}{l} \wedge G(x, s, c, u) \\ \wedge BAP(x, s, c, u, u') \end{array} \right)$

$GRD(\text{evt}) \hat{=} G(x, s, c, u)$

$ACT(\text{evt}) \hat{=} G(x, s, c, u)$

$BAP(x, s, c, u, u')$

s	<i>seen sets</i>
c	<i>seen constants</i>
u	<i>variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>invariants</i>
$Q(s, c, u)$	<i>theorems</i>
evt	<i>event name</i>
x	<i>event parameter</i>
$G(x, s, c, u)$	<i>event guard</i>
$BAP(x, s, c, u, u')$	<i>event before-after predicate</i>

PO evt/act/FIS

$Ax(s, c), Th(s, c), I(s, c, u), G(x, s, c, u), \vdash \exists u'. BAP(x, s, c, u, u')$

- ① Overview of machines, contexts and proof obligations
- ② Proof Obligations for Contexts and Machines
- ③ Proof Obligations for Refinement

PO evt/grd/GRD

```
EVENT ae
  ANY x WHERE
    G(x, s, c, u)
  THEN
    u : |ABAP(x, s, c, u, u')
  END
```

```
EVENT ce
  REFINES
    ae
  ANY y WHERE
    H(y, s, c, v)
  WITH
    x : W(x, y, s, c, v)
  THEN
    v : |CBAP(y, s, c, v, v')
  END
```

<i>s</i>	<i>seen sets</i>
<i>c</i>	<i>seen constants</i>
<i>u, v</i>	<i>abstract and concrete variables</i>
<i>Ax(s, c)</i>	<i>seen axioms</i>
<i>Th(s, c)</i>	<i>seen theorems</i>
<i>I(s, c, u)</i>	<i>abstract invariants</i>
<i>J(s, c, u, v)</i>	<i>concrete invariants</i>
<i>Q(s, c, u), R(s, c, u, v)</i>	<i>abstract and concrete theorems</i>
<i>ae, ce</i>	<i>abstract and concrete event names</i>
<i>x, y</i>	<i>event parameters</i>
<i>G(x, s, c, u)</i>	<i>abstract event guard</i>
<i>H(y, s, c, v)</i>	<i>concrete event guard</i>
<i>ABAP(x, s, c, u, u')</i>	<i>abstract event before-after predicate</i>
<i>CBAP(x, s, c, u, u')</i>	<i>concrete event before-after predicate</i>
<i>W(x, y, s, c, v)</i>	witness predicate

PO evt/grd/GRD

$Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v), W(x, y, s, c, v), H(y, s, c, v), \vdash$
 $G(x, s, c, u, u')$

PO evt/act/SIM

```

EVENT ae
  ANY x WHERE
     $G(x, s, c, u)$ 
  THEN
     $u : |ABAP(x, s, c, u, u')$ 
  END

EVENT ce
  REFINES
    ae
  ANY y WHERE
     $H(y, s, c, v)$ 
  WITH
     $x : WP(x, y, s, c, v)$ 
     $u' : WV(y, u', s, c, v)$ 
  THEN
     $v : |CBAP(y, s, c, v, v')$ 
  END

```

s	<i>seen sets</i>
c	<i>seen constants</i>
u, v	<i>abstract and concrete variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>abstract invariants</i>
$J(s, c, u, v)$	<i>concrete invariants</i>
$Q(s, c, u), R(s, c, u, v)$	<i>abstract and concrete theorems</i>
ae, ce	<i>abstract and concrete event names</i>
x, y	<i>event parameters</i>
$G(x, s, c, u)$	<i>abstract event guard</i>
$H(y, s, c, v)$	<i>concrete event guard</i>
$ABAP(x, s, c, u, u')$	<i>abstract event before-after predicate</i>
$CBAP(x, s, c, u, u')$	<i>concrete event before-after predicate</i>
$WP(x, y, s, c, v)$	witness parameter predicate
$WV(y, u', s, c, v)$	witness variable predicate

PO evt/act/SIM

$$\left(\begin{array}{l} Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v) \\ WP(x, y, s, c, v), WV(y, u', s, c, v) \\ H(y, s, c, v), CBAP(y, s, c, v, v') \end{array} \right) \vdash ABAP(x, s, c, u, u')$$

PO evt/act/SIM

```
EVENT ae
ANY x WHERE
  G(x, s, c, u)
THEN
  u : |BAP(x, s, c, u, u')
END
...
VARIANT
  exp(s, c, u)
```

s	<i>seen sets</i>
c	<i>seen constants</i>
u, v	<i>abstract and concrete variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>abstract invariants</i>
$J(s, c, u, v)$	<i>concrete invariants</i>
$Q(s, c, u), R(s, c, u, v)$	<i>abstract and concrete theorems</i>
evt, ce	<i>event name</i>
x	<i>event parameters</i>
$G(x, s, c, u)$	<i>abstract event guard</i>
$BAP(x, s, c, u, u')$	<i>event before-after predicate</i>
$exp(s, c, u)$	<i>arithmetric expression</i>

PO evt/NAT

$$Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v), G(x, s, c, u) \vdash exp(s, c, u) \in \mathbb{N}$$

PO evt/act/SIM

```
EVENT ae
  ANY x WHERE
    G(x, s, c, u)
  THEN
    u : |BAP(x, s, c, u, u')
  END
...
VARIANT
  exp(s, c, u)
```

s	<i>seen sets</i>
c	<i>seen constants</i>
u, v	<i>abstract and concrete variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>abstract invariants</i>
$J(s, c, u, v)$	<i>concrete invariants</i>
$Q(s, c, u), R(s, c, u, v)$	<i>abstract and concrete theorems</i>
evt, ce	<i>event name</i>
x	<i>event parameters</i>
$G(x, s, c, u)$	<i>abstract event guard</i>
$BAP(x, s, c, u, u')$	<i>event before-after predicate</i>
$setexp(s, c, u)$	<i>set expression</i>

```
PO evt/NAT  $Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v), G(x, s, c, u) \vdash$ 
finite( $setexp(s, c, u)$ )
```

```

EVENT ae
  ANY x WHERE
    G(x, s, c, u)
  THEN
    u : |BAP(x, s, c, u, u')
  END
...
VARIANT
  exp(s, c, u)

```

<i>s</i>	<i>seen sets</i>
<i>c</i>	<i>seen constants</i>
<i>u, v</i>	<i>abstract and concrete variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>abstract invariants</i>
$J(s, c, u, v)$	<i>concrete invariants</i>
$Q(s, c, u), R(s, c, u, v)$	<i>abstract and concrete theorems</i>
evt, ce	<i>event name</i>
x	<i>event parameters</i>
$G(x, s, c, u)$	<i>abstract event guard</i>
$BAP(x, s, c, u, u')$	<i>event before-after predicate</i>
$exp(s, c, u)$	<i>arithmetric expression</i>

PO evt/VAR

$Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v), G(x, s, c, u), BAP(x, s, c, u, u') \vdash$
 $exp(s, c, u') < exp(s, c, u)$

PO evt/VAR

```
EVENT ae
  ANY x WHERE
    G(x, s, c, u)
  THEN
    u : |BAP(x, s, c, u, u')
  END
  ...
VARIANT
  setexp(s, c, u)
```

s	<i>seen sets</i>
c	<i>seen constants</i>
u, v	<i>abstract and concrete variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>abstract invariants</i>
$J(s, c, u, v)$	<i>concrete invariants</i>
$Q(s, c, u), R(s, c, u, v)$	<i>abstract and concrete theorems</i>
evt, ce	<i>event name</i>
x	<i>event parameters</i>
$G(x, s, c, u)$	<i>abstract event guard</i>
$BAP(x, s, c, u, u')$	<i>event before-after predicate</i>
$setexp(s, c, u)$	<i>set-theoretic expression</i>

PO evt/VAR

$Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v), G(x, s, c, u), BAP(x, s, c, u, u') \vdash$
 $setexp(s, c, u') \subset setexp(s, c, u)$

PO evt/x/WFIS

```
EVENT ae
  ANY x WHERE
    G(x, s, c, u)
  THEN
    u : |ABAP(x, s, c, u, u')
  END
```

```
EVENT ce
  REFINES
    ae
  ANY y WHERE
    H(y, s, c, v)
  WITH
    x : WP(x, y, s, c, v)
    u' : WV(y, u', s, c, v)
  THEN
    v : |CBAP(y, s, c, v, v')
  END
```

<i>s</i>	<i>seen sets</i>
<i>c</i>	<i>seen constants</i>
<i>u, v</i>	<i>abstract and concrete variables</i>
$Ax(s, c)$	<i>seen axioms</i>
$Th(s, c)$	<i>seen theorems</i>
$I(s, c, u)$	<i>abstract invariants</i>
$J(s, c, u, v)$	<i>concrete invariants</i>
$Q(s, c, u), R(s, c, u, v)$	<i>abstract and concrete theorems</i>
<i>ae, ce</i>	<i>abstract and concrete event names</i>
<i>x, y</i>	<i>event parameters</i>
$G(x, s, c, u)$	<i>abstract event guard</i>
$H(y, s, c, v)$	<i>concrete event guard</i>
$ABAP(x, s, c, u, u')$	<i>abstract event before-after predicate</i>
$CBAP(x, s, c, u, u')$	<i>concrete event before-after predicate</i>
$WP(x, y, s, c, v)$	witness parameter predicate
$WV(y, u', s, c, v)$	witness variable predicate

PO evt/x/WFIS

$$Ax(s, c), Th(s, c), I(s, c, u), J(s, c, u, v), H(y, s, c, v) \vdash \\ \exists x. WP(x, y, s, c, v)$$