UNIVERSITY OF Southampton

Machine Inclusion Mechanism for Event-B

Dana Dghaym, T.S. Hoang, Michael Butler
27 June 2017

University of Southampton

Motivation

- **Event-B top-down development**
  - Refinement
  - Decomposition
- **Pros and Cons**

✓ Details can be introduced gradually in the formal model

✗ Large models with monolithic structures

University of Southampton

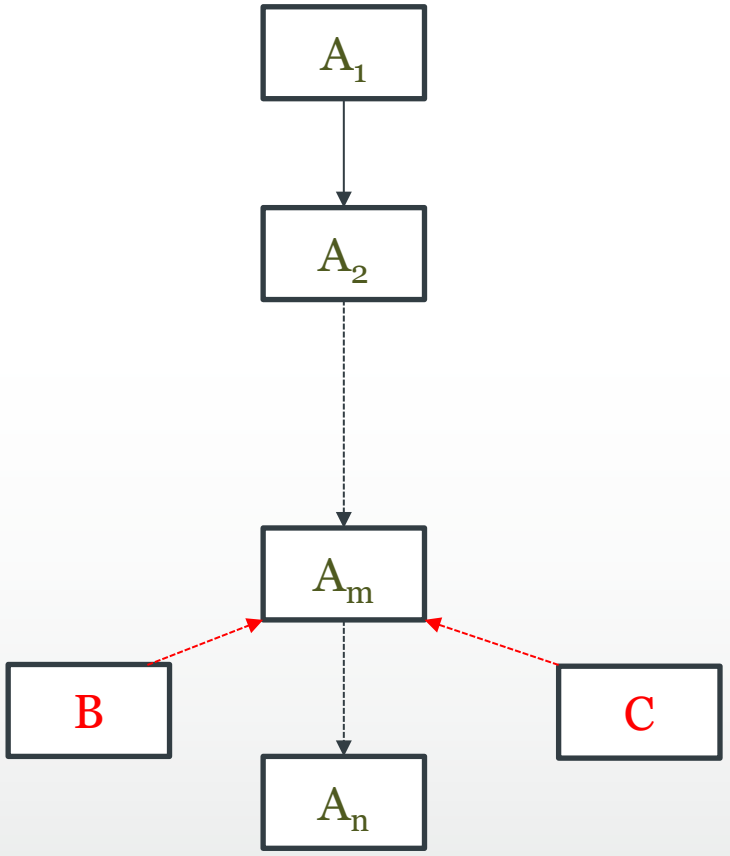
Aims

- Need for composing specifications
  - Including existing specification
  - Reusing consistently
- Smooth integration with the Event-B development process
  - Accommodate changes seamlessly

University of Southampton

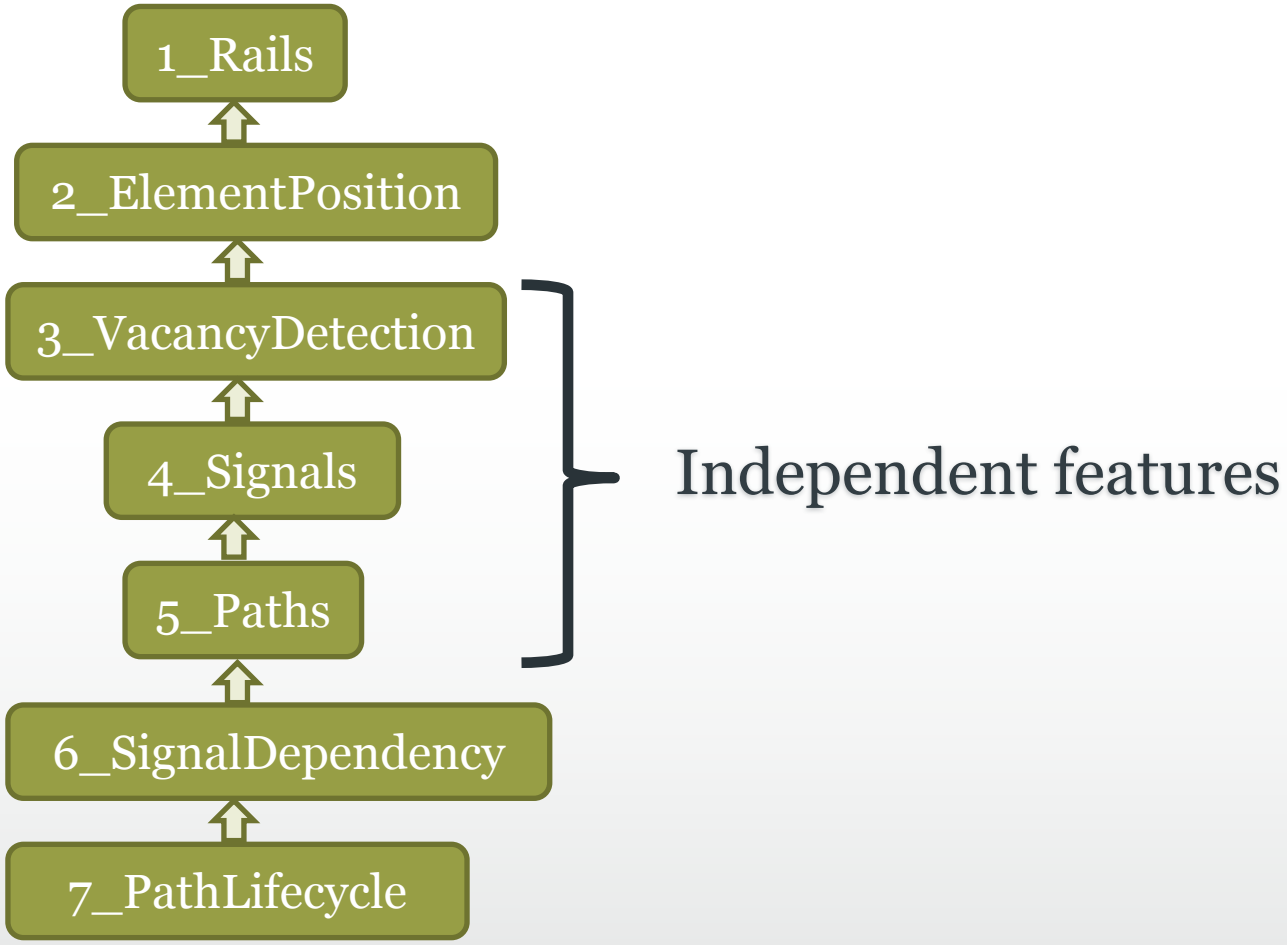
Machine Inclusion Refinement Structure

- Development: Combination of **top-down** and **bottom-up**
- Correct-by-construction
- Reusable Fragments



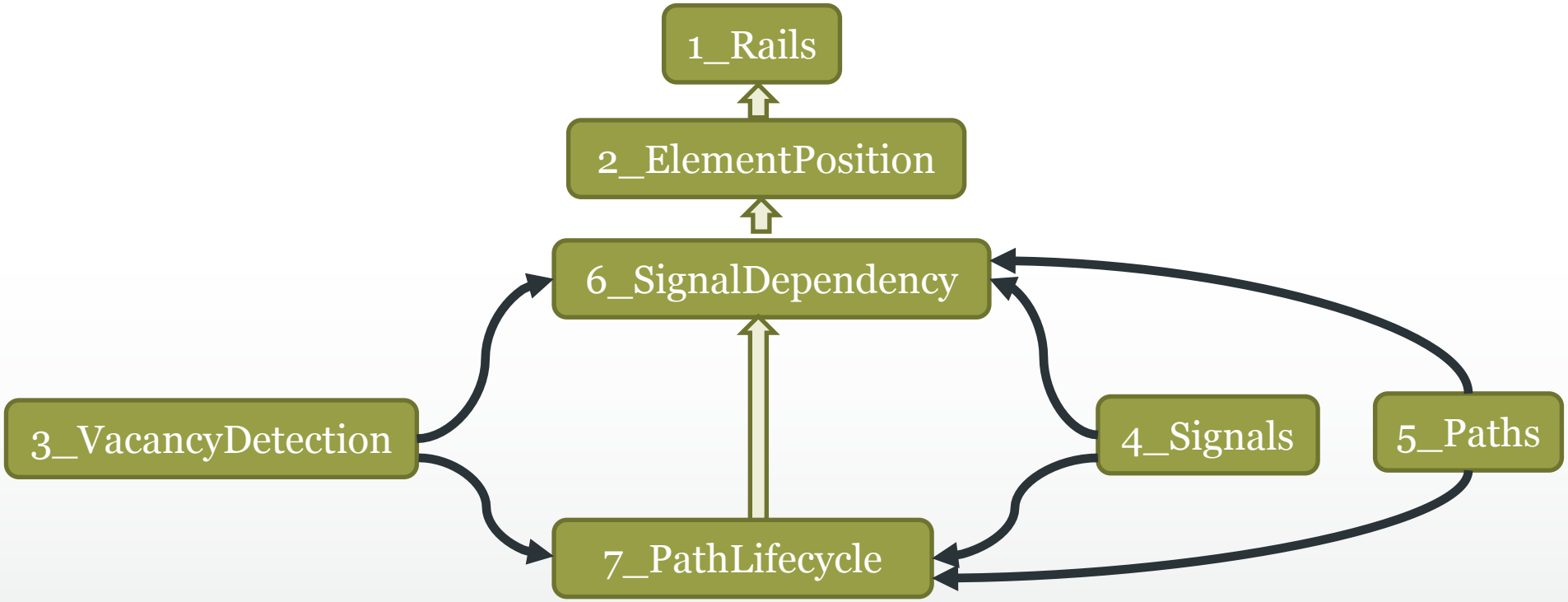
UNIVERSITY OF Southampton

RailGround Example[1]



University of Southampton

RailGround Example: Machine Inclusion



UNIVERSITY OF Southampton

Machine Inclusion - Concepts

- Machine **A** includes machine **B**
  - **A** inherits **B**'s variables
  - **A** inherits **B**'s invariants
  - **B**'s variables can only be modified via event synchronisation
- Multiple instances of **B** can be included via prefixing
  - Variables, events are renamed accordingly

UNIVERSITY OF Southampton

Machine Inclusion - Illustration

machine B

variables y
invariants
$J(y)$

events
event f
any u where
$G_B(y, u)$
then
$y :| BAP_B(y, u, y')$
end

machine A
includes p_B
variables x
invariants
$I(x, p\_y)$

events
event e
synchronises p_f
any t where
$G_A(x, t)$
$H_{AB}(x, p\_y, t, p\_u)$
then
$x :| BAP_A(x, t, x')$
end

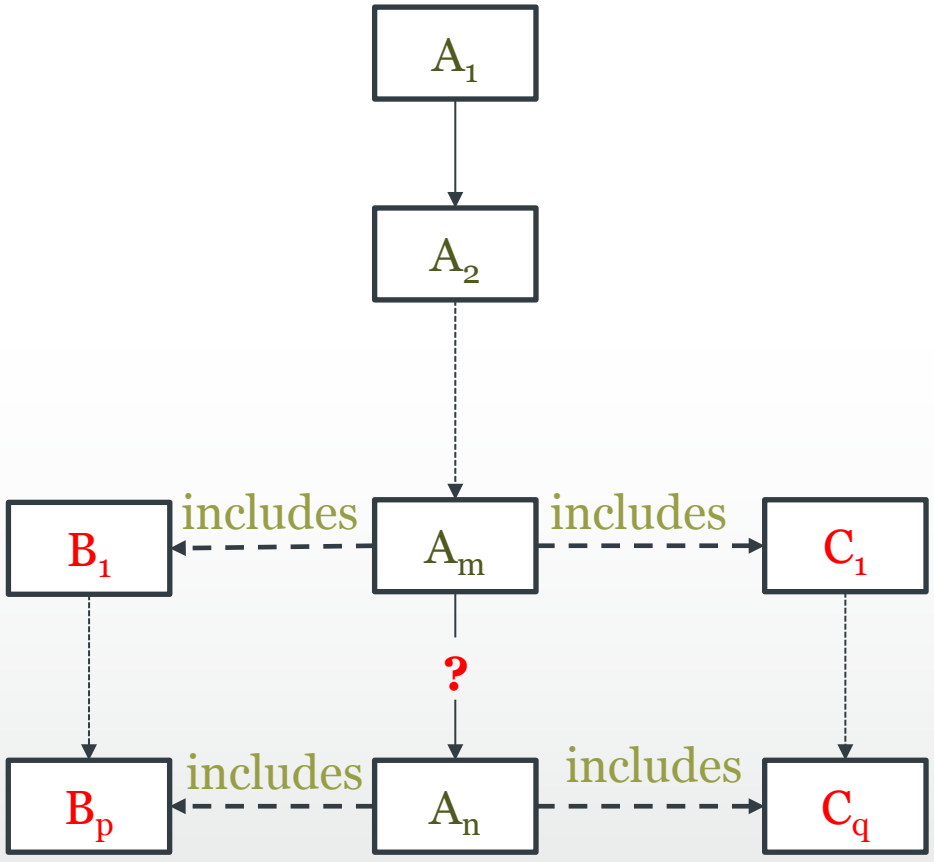
machine (flatten_)A

variables x , p_y
invariants
$I(x, p\_y)$
$J(p\_y)$

events
event e
any t, p_u where
$G_A(x, t)$
$H_{AB}(x, p\_y, t, p\_u)$
$G_B(p\_y, p\_u)$
then
$x :| BAP_A(x, t, x')$
$y :| BAP_B(p\_y, p\_u, p\_y')$
end

UNIVERSITY OF Southampton

Refinement-Chain Inclusion



UNIVERSITY OF Southampton

Refinement-Chain Inclusion

```
machine A1
includes B1

events
   e synchronises eb
   any u, v1 where
        Ga (u, x)
        Gab1 (u, v1, x, y1)
        Gb1 (v1, y1)
  then
        x := Ea (u, x)
        y1 := Eb1 (v1, y1)
    end
end
```

```
machine A2
includes B2   // B2 refines B1
refines A1

events
   e synchronises eb
   any u, v2 where
        Ga (u, x)
        Gab2 (u, v2, x, y2)
        Gb2 (v2, y2)
  then
        x := Ea (u, x)
        y2 := Eb2 (v2, y2)
    end
end
```

The refinement is "**almost**" correct-by-construction

(The only change in A2 compared to A1 is **includes** clause)

Refinement-Chain Inclusion

```
machine A1
includes B1

events
   e synchronises eb
   any u, v1 where
      Ga (u, x)
      Gab1 (u, v1, x, y1)
      Gb1 (v1, y1)
   then
      x := Ea (u, x)
      y1 := Eb1 (v1, y1)
   end
end
```

```
machine A2
includes B2   // B2 refines B1
refines A1

events
   e synchronises eb
   any u, v2 where
      Ga (u, x)
      Gab2 (u, v2, x, y2)
      Gb2 (v2, y2)
   then
      x := Ea (u, x)
      y2 := Eb2 (v2, y2)
   end
end
```

- Guard strengthening for Ga is trivial
- Action simulation for Ea is trivial

UNIVERSITY OF Southampton

Refinement-Chain Inclusion

```
machine A1
includes B1

events
   e synchronises eb
   any u, v1 where
       Ga (u, x)
       Gab1 (u, v1, x, y1)
       Gb1 (v1, y1)
   then
       x := Ea (u, x)
       y1 := Eb1 (v1, y1)
    end
end
```

```
machine A2
includes B2   // B2 refines B1
refines A1

events
   e synchronises eb
   any u, v2 where
       Ga (u, x)
       Gab2 (u, v2, x, y2)
       Gb2 (v2, y2)
   then
       x := Ea (u, x)
       y2 := Eb2 (v2, y2)
    end
end
```

- Guard strengthening: Gb1 by Gb2 is guaranteed by B2 refines B1
- Action simulation: Eb1 by Eb2 is guaranteed by B2 refines B1

UNIVERSITY OF Southampton

Refinement-Chain Inclusion

```
machine A1
includes B1

events
   e synchronises eb
   any u, v1 where
       Ga (u, x)
       Gab1 (u, v1, x, y1)
       Gb1 (v1, y1)
  then
       x := Ea (u, x)
       y1 := Eb1 (v1, y1)
    end
end
```

```
machine A2
includes B2   // B2 refines B1
refines A1

events
   e synchronises eb
   any u, v2 where
       Ga (u, x)
       Gab2 (u, v2, x, y2)
       Gb2 (v2, y2)
  then
       x := Ea (u, x)
       y2 := Eb2 (v2, y2)
    end
end
```

- Guard strengthening for Gab1 by Gab2 needs to be proved

UNIVERSITY OF Southampton

Refinement-Chain Inclusion

```
machine A1
includes B1

events
   e synchronises eb
   any u, v1 where
       Ga (u, x)
       Gab1 (u, v1, x, y1)
       Gb1 (v1, y1)
  then
       x := Ea (u, x)
       y1 := Eb1 (v1, y1)
    end
end
```

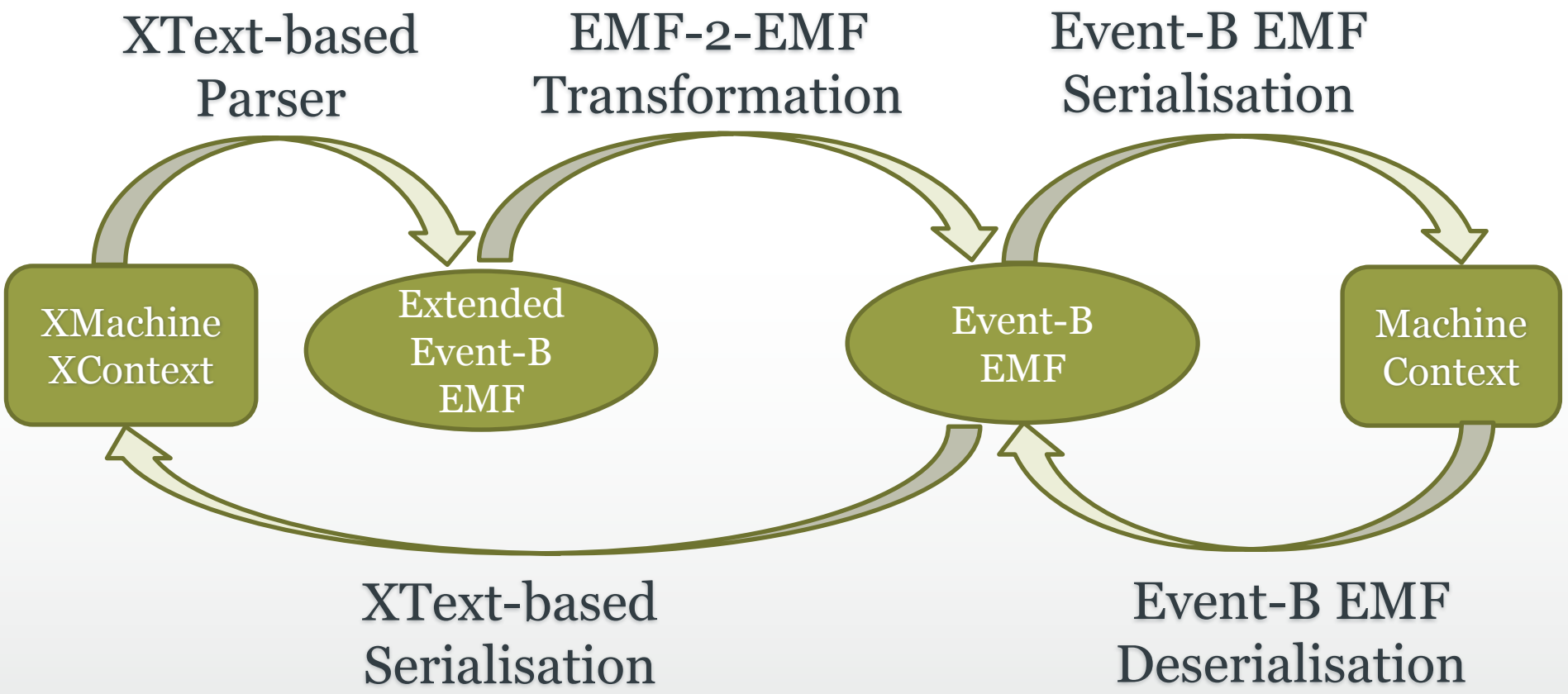
```
machine A2
includes B2   // B2 refines B1
refines A1

events
   e synchronises eb
   any u, v2 where
       Ga (u, x)
       Gab2 (u, v2, x, y2)
       Gb2 (v2, y2)
  then
       x := Ea (u, x)
       y2 := Eb2 (v2, y2)
    end
end
```

Refinement Inclusion Theorem:

B1 refined by B2 & (Gab1 ≤ Gab2) => A1 refined by A2

University of Southampton

Tool Support



UNIVERSITY OF Southampton

Summary

- Machine Inclusion supports:
  - Reuse
  - **Top-down** and **bottom-up** development
- Refinement-Chain inclusion
  - **Almost correct-by-construction**
- Tool Support
  - **XText** + Extended Event-B **EMF**