

Learning the Portable Stimulus Standard

Tom Fitzpatrick

CEO Big Fish EDA Consulting

On behalf of the Accellera Systems Initiative

and the Accellera Portable Stimulus Working Group



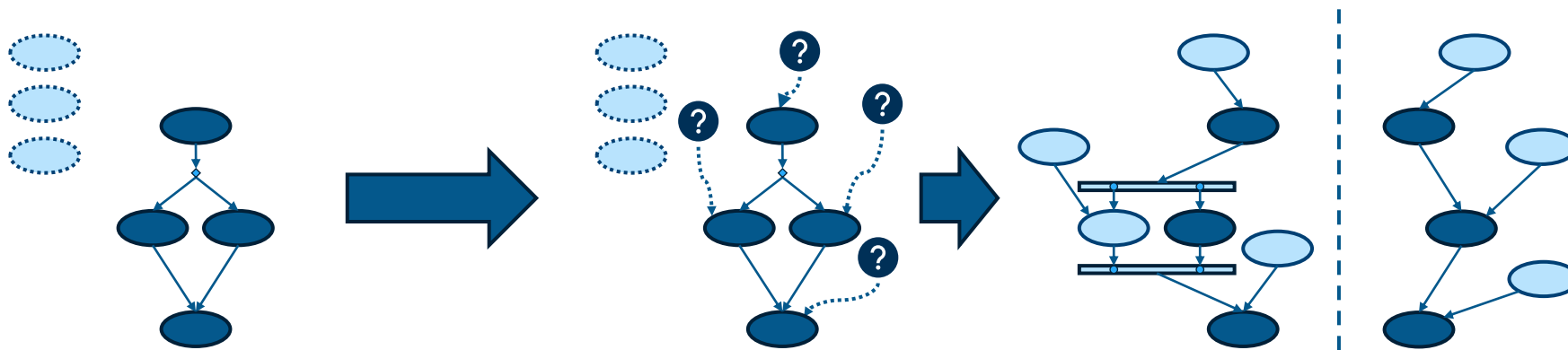
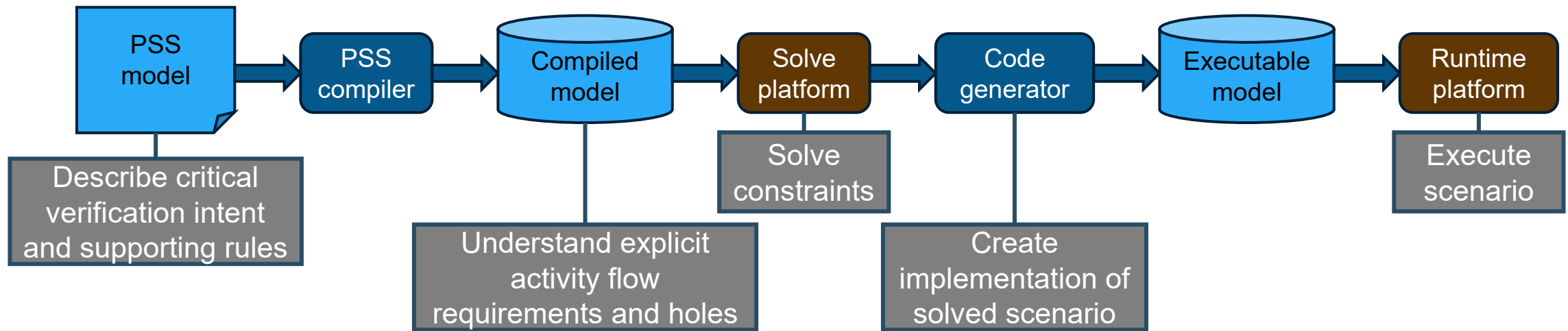
Randomization and Inferencing

▶▶▶ Session 10

⋮ What You Will Learn

- ▶ Action inferencing
- ▶ Constraint solving and lookahead
- ▶ `pre_solve` and `post_solve` blocks

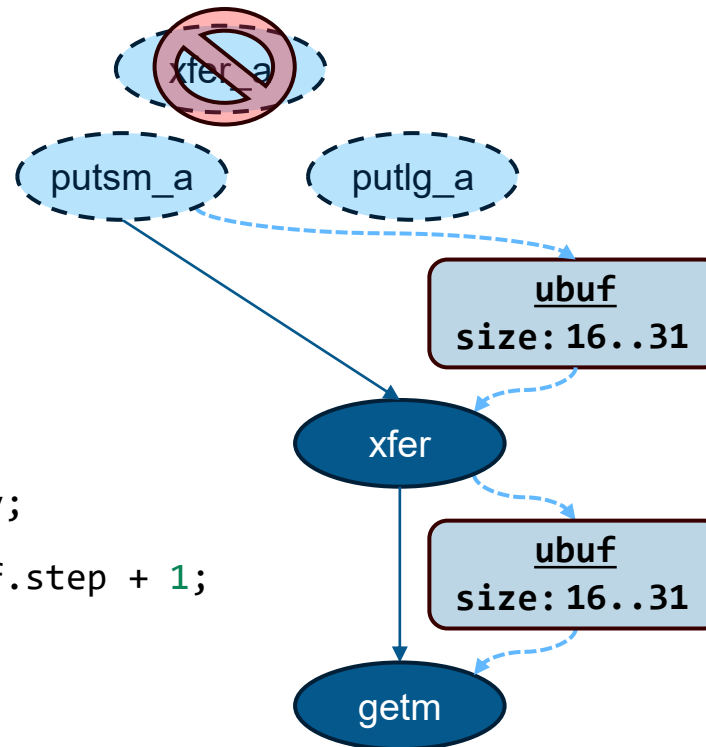
⋮ PSS Models are Declarative



::: Action Inferencing and Constraints

- ▶ The solver must infer actions as needed to make the scenario legal

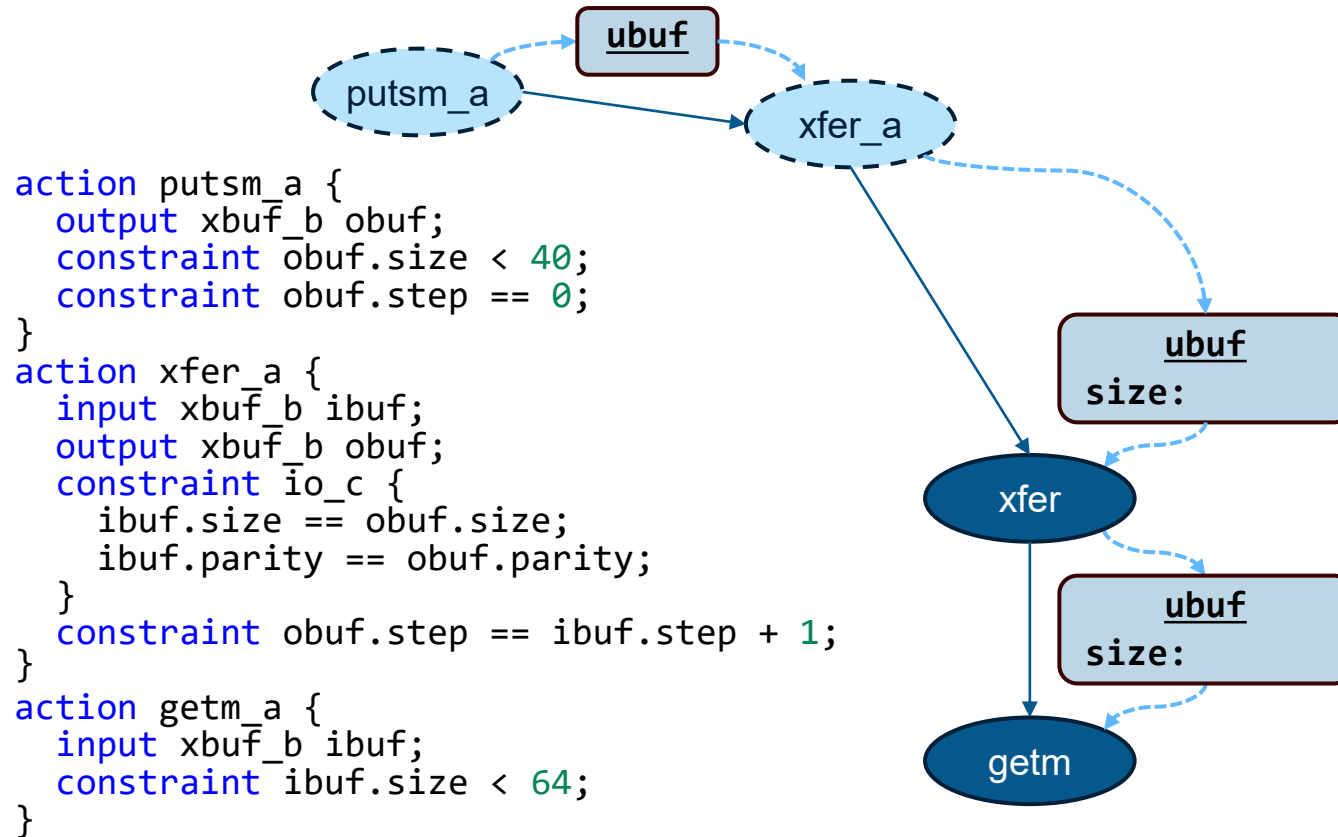
```
action putsm_a {  
  output xbuf_b obuf;  
  constraint obuf.size < 40;  
  constraint obuf.step == 0;  
}  
action xfer_a {  
  input xbuf_b ibuf;  
  output xbuf_b obuf;  
  constraint io_c {  
    ibuf.size == obuf.size;  
    ibuf.parity == obuf.parity;  
  }  
  constraint obuf.step == ibuf.step + 1;  
}  
action getm_a {  
  input xbuf_b ibuf;  
  constraint ibuf.size < 64;  
}
```



```
action putlg_a {  
  output xbuf_b obuf;  
  constraint obuf.size >= 40;  
  constraint obuf.step == 0;  
}  
action entry_a {  
  getm_a getm;  
  xfer_a xfer;  
  
  constraint getm.ibuf.size > 15;  
  constraint c2 {getm.ibuf.step == 1;}  
  activity {  
    xfer;  
    getm with {ibuf.size < 32;};  
  }  
}
```

::: Action Inferencing and Constraints

- ▶ The solver must infer actions as needed to make the scenario legal



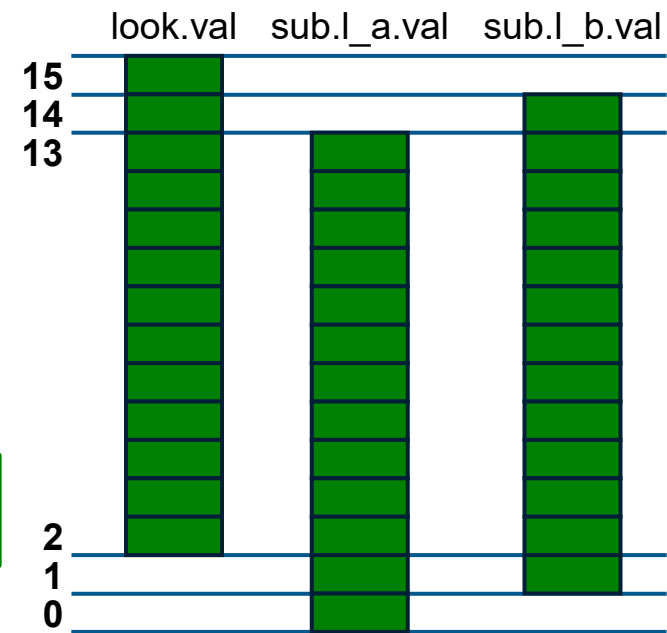
```
action putlg_a {  
  output xbuf_b obuf;  
  constraint obuf.size >= 40;  
  constraint obuf.step == 0;  
}  
action entry_a {  
  getm_a getm;  
  xfer_a xfer;  
  
  constraint getm.ibuf.size > 15;  
  constraint c2 {getm.ibuf.step == 2;}  
  activity {  
    xfer;  
    getm with {ibuf.size < 32;};  
  }  
}
```

Constraint Lookahead

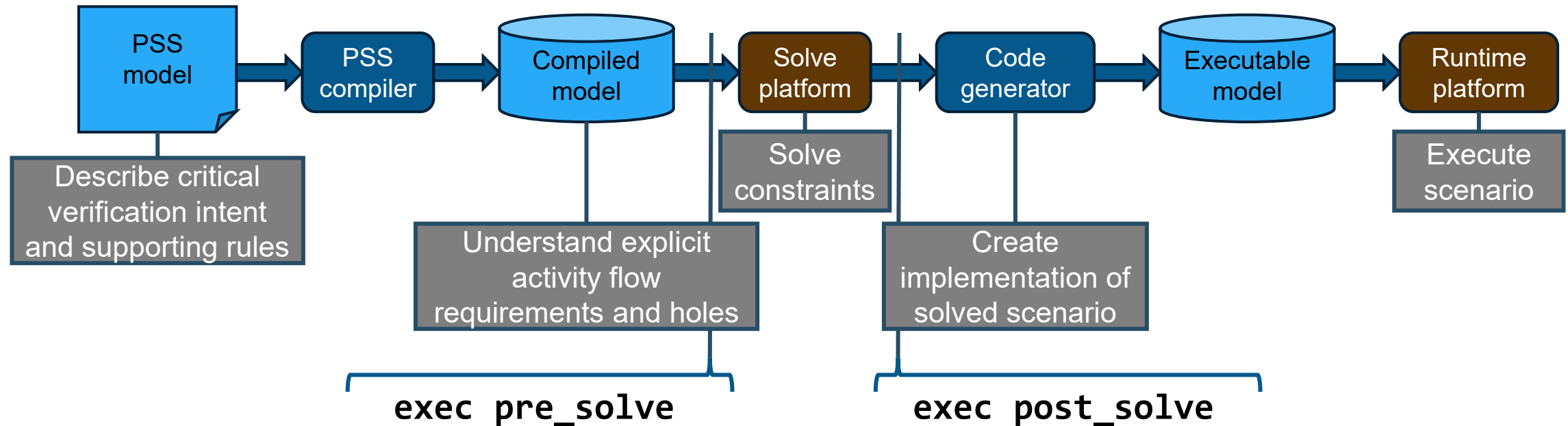
```
action look_a {  
  rand bit[4] val;  
}  
action sub_a {  
  look_a l_a, l_b;  
  constraint c1 { l_a.val < l_b.val;}  
  
  activity {  
    l_a;  
    l_b;  
  }  
}  
  
action entry_a {  
  look_c::look_a look;  
  look_c::sub_a sub;  
  constraint c2 { sub.l_b.val < look.val;}  
  
  activity {  
    look;  
    sub;  
  }  
}
```

Each randomization must consider constraints on fields across actions

Randomization happens when an action is traversed



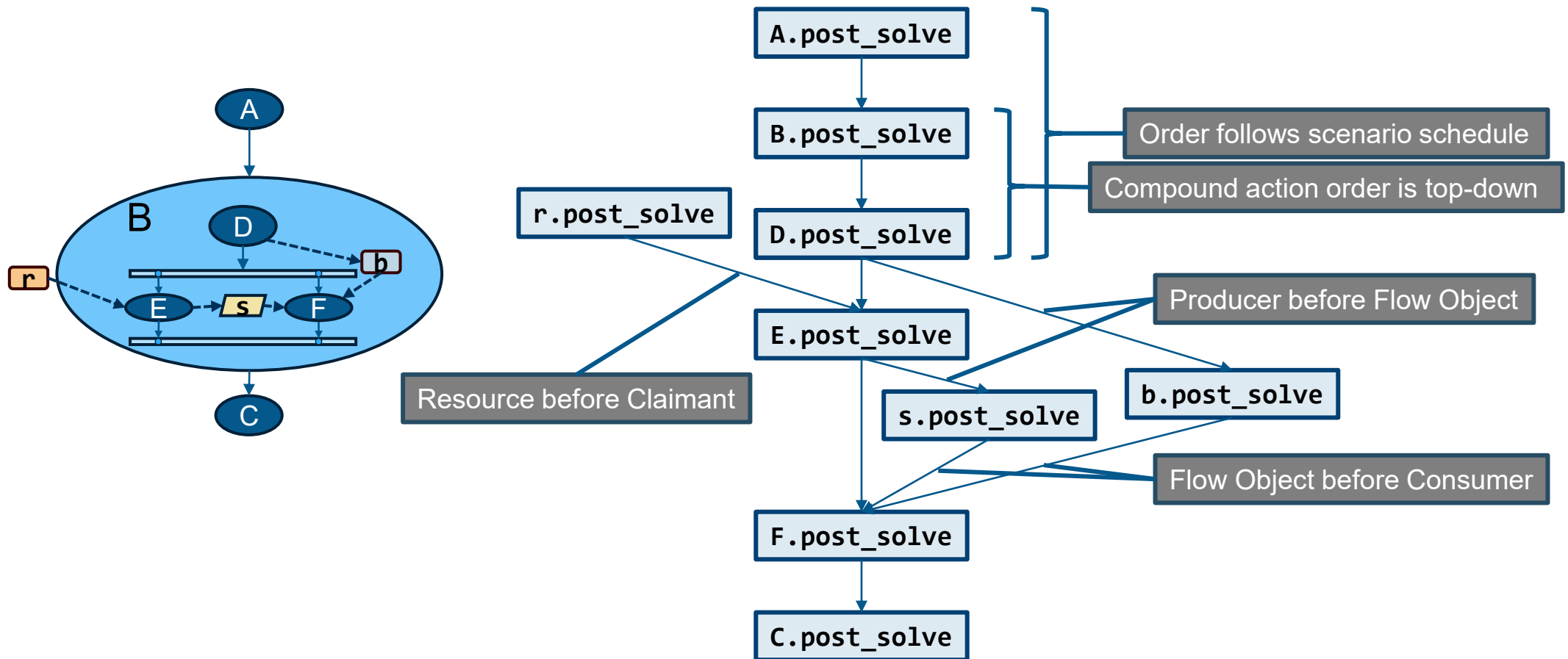
pre_ and post_solve Exec Blocks



- Set non-rand fields to be used by the solver
- Can read non-rand fields
- Cannot access handles
- Reading rand fields gives default value

- Evaluated after the solver has resolved all rand field values
- Can set non-rand field values based on randomized values

pre_solve and post_solve Blocks



Using post_solve Blocks

```
component pss_top {
```

```
  buffer xbuf_b {  
    rand bit[8] size;  
    list<bit[8]> data;  
  }
```

```
  action putm_a {
```

```
    output xbuf_b obuf;
```

```
    constraint obuf.size in [64..127];
```

```
    exec post_solve {
```

```
      repeat (i:(int)obuf.size) {
```

```
        obuf.data.push_back((((i * 37) ^ (i >> 3)) & 0xFF));
```

```
      }
```

```
    }
```

```
  }
```

```
}
```

Random field

Non-random field

Randomized at Solve time
when action is traversed

Subject to constraints

Use randomized value

Assign value to non-random field



Thank You

