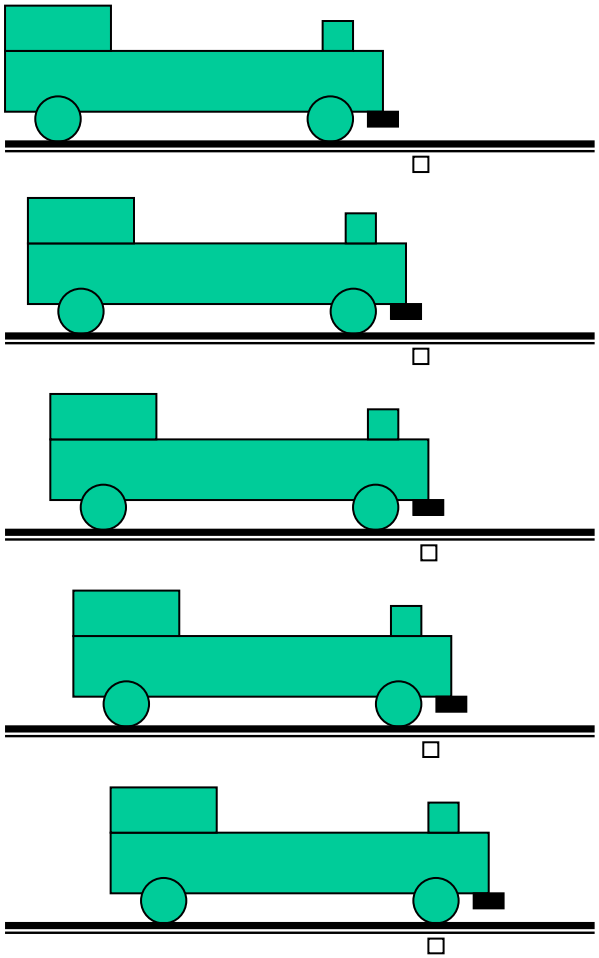
HIT3047 Real-Time Programming

Topolog2 Notes

Supplement to comments in Topolog2.ads

(See also sim_topolog2.exe)

Sensor Event Generation



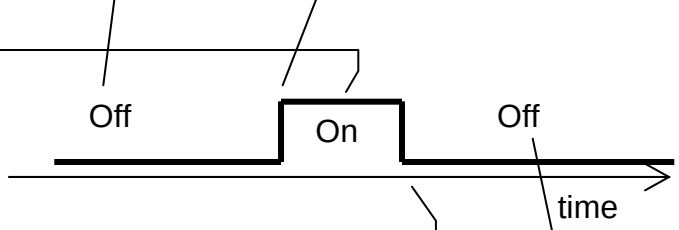
Front on plain track

Sensor On event

Front over sensor

Sensor Off event

Front over crossing



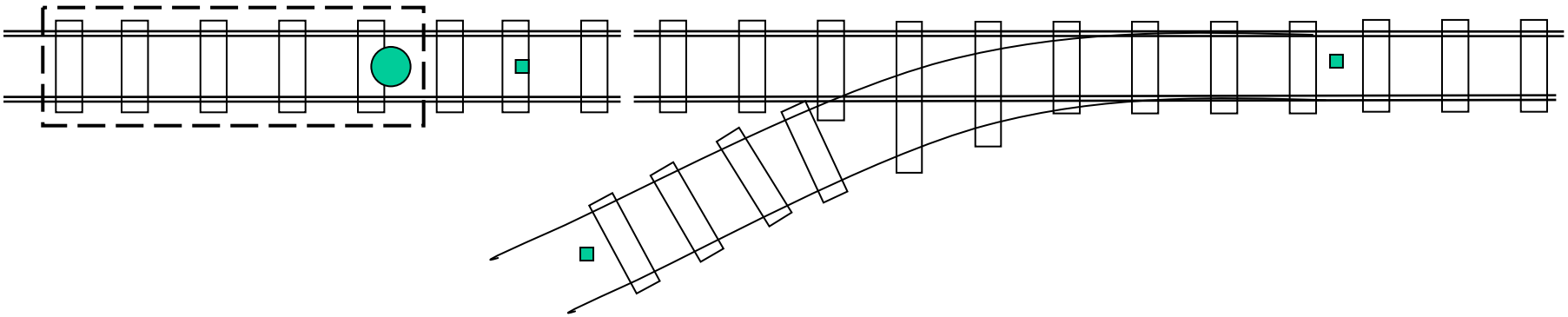
State of sensor 15

It all happens again at the rear magnet (modelled in code as the rear instance of Train_Position). (Trains always have two magnets.)

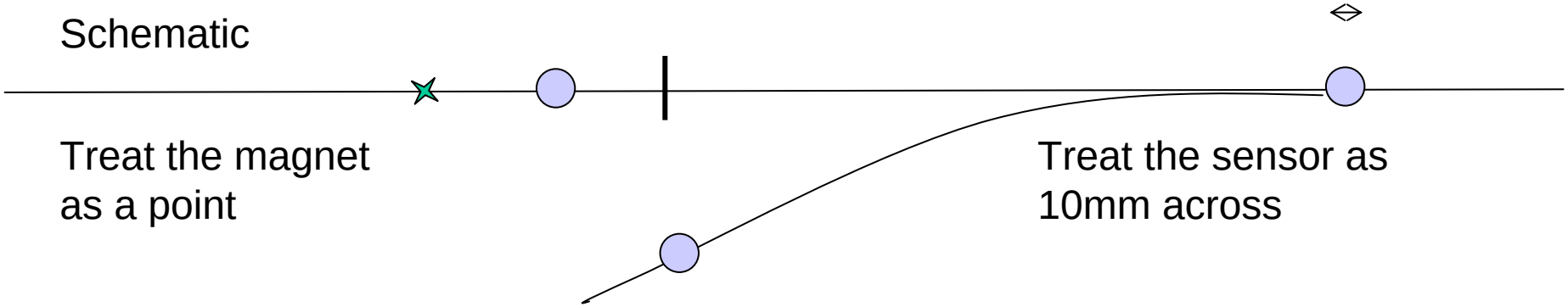
The same time diagram occurs if the train reverses (changes polarity) from the position shown, but the state model copes….

Loco, magnet, sensor, gap, turnout

sensor < 1mm diam



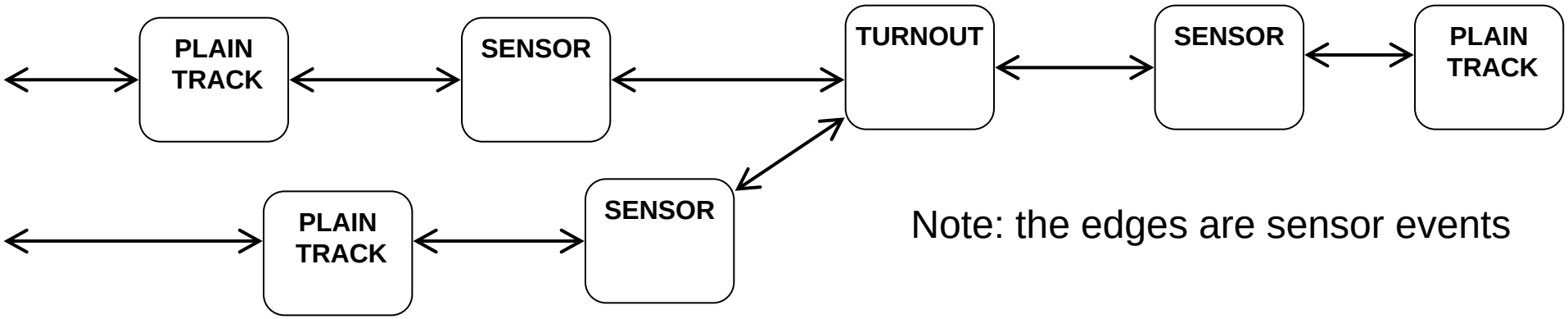
Schematic



Treat the magnet as a point

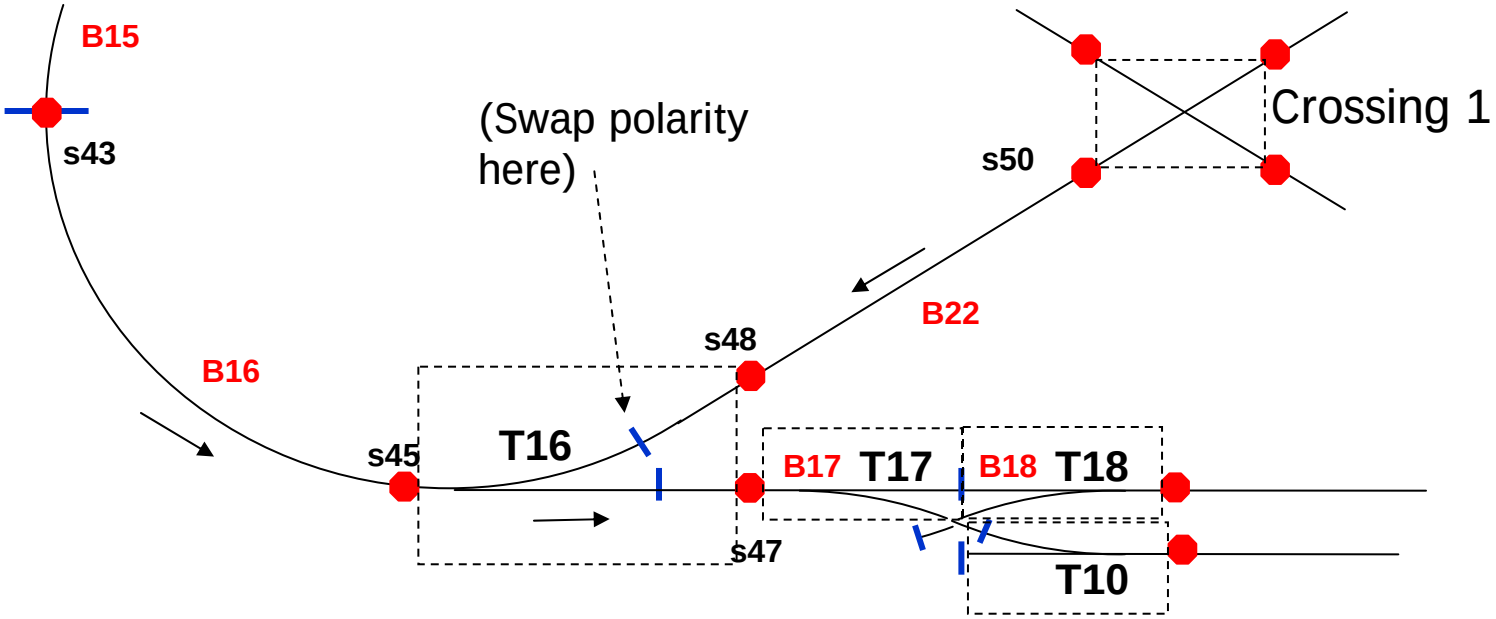
Treat the sensor as 10mm across

Graph model

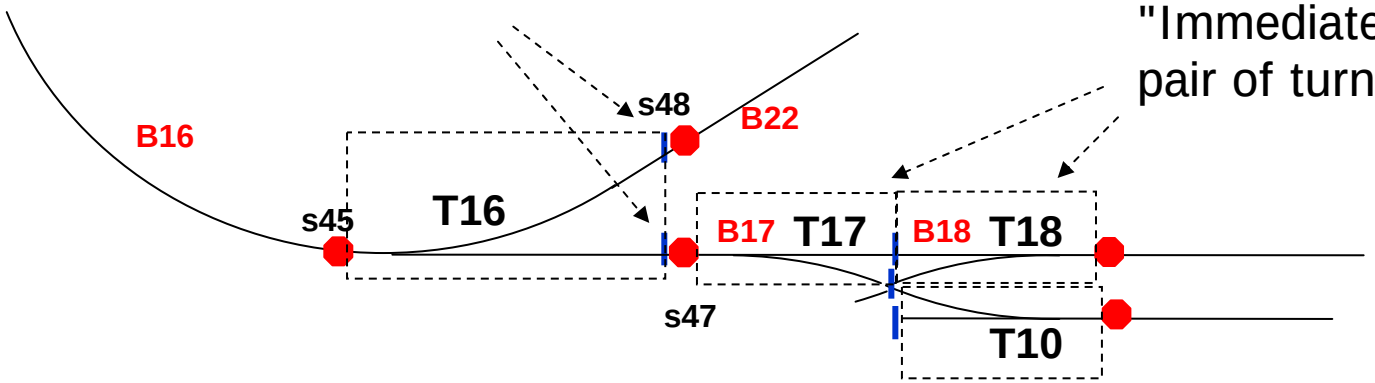


Note: the edges are sensor events

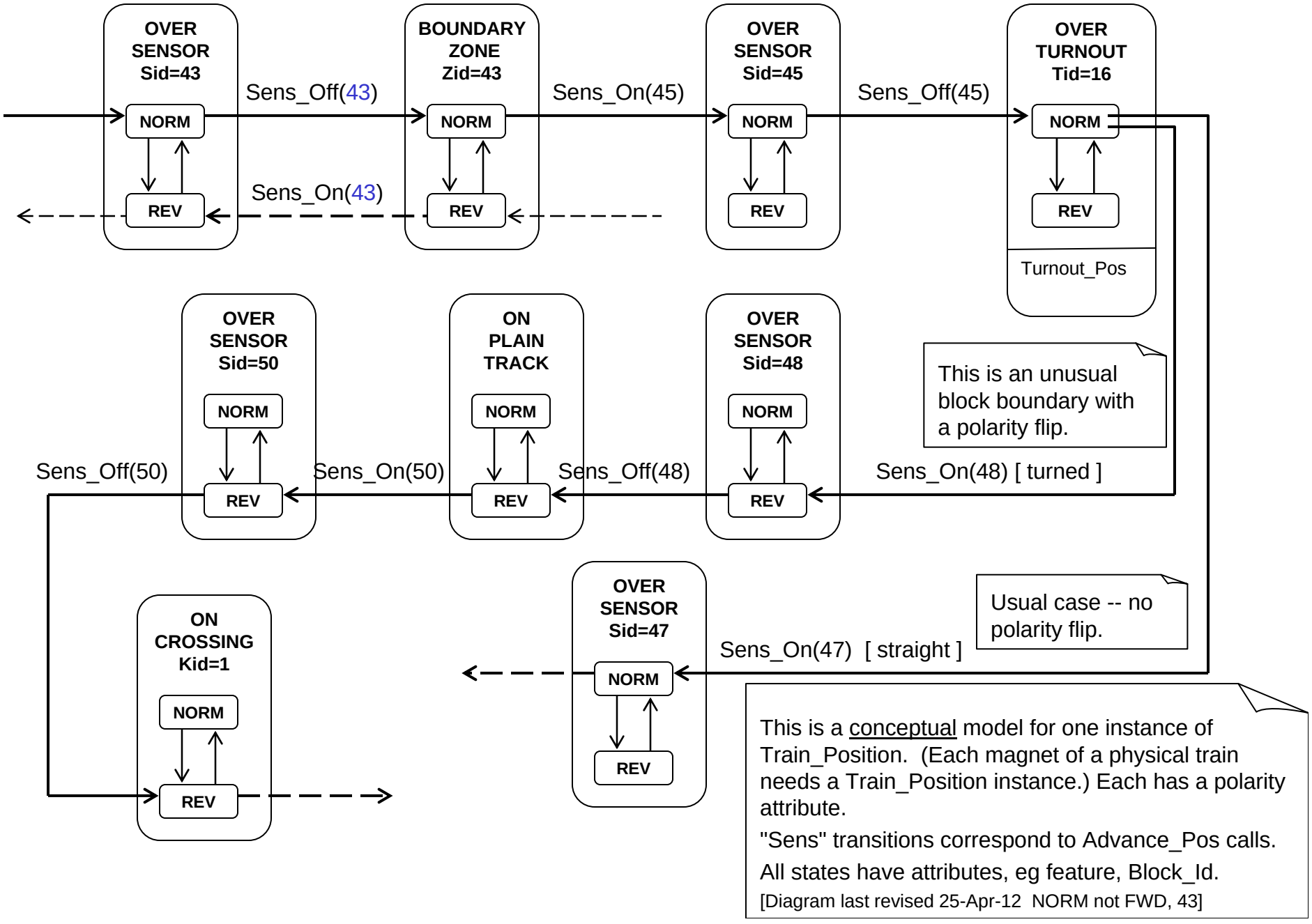
Topolog2 features



Simplification: Block gaps in turnouts are assumed at sensors

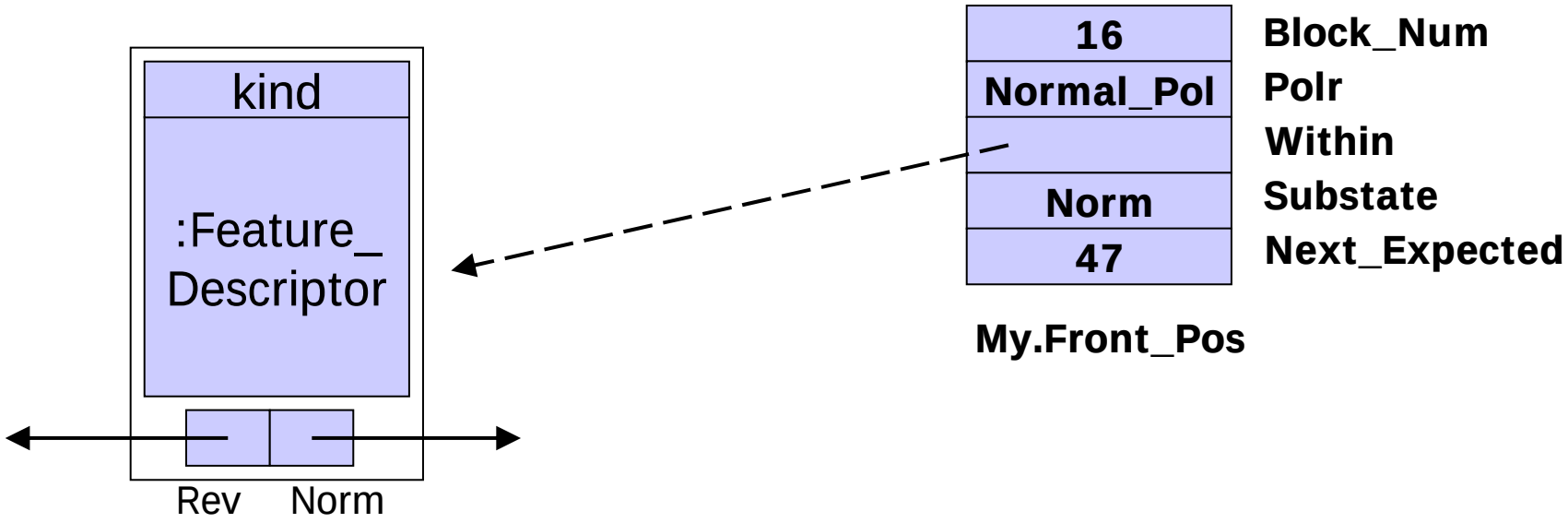


Partial UML State Model for Topolog2 package



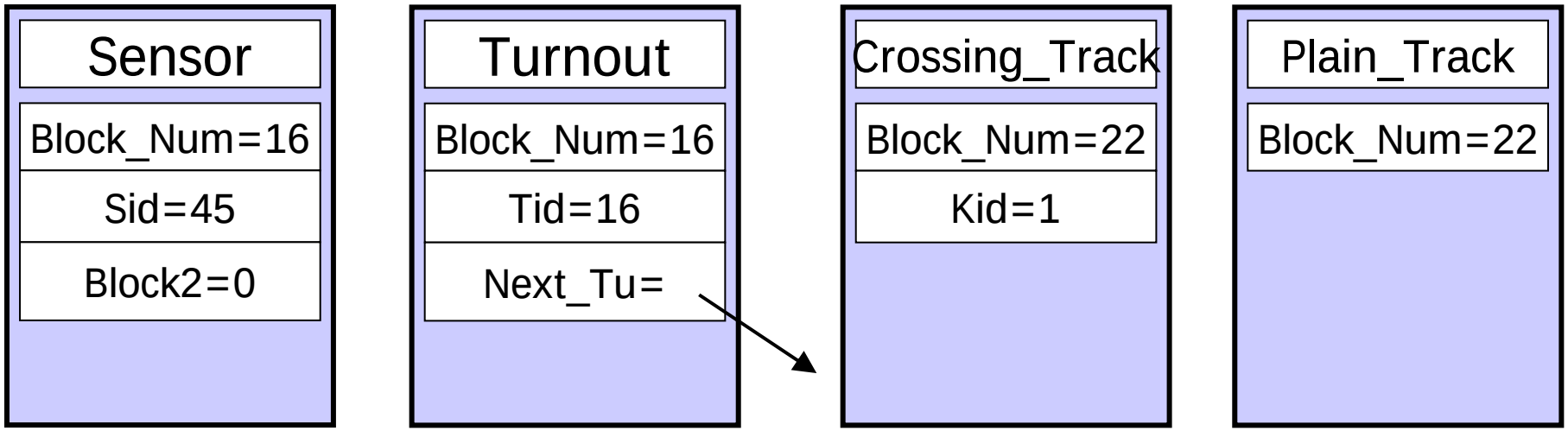
Topolog2 Data Structures - 1

- Constant array Data inside package body, defines the graph
- Features store fixed attributes of each node
- User variables Pos : Train_Position contain varying data



Topolog2 Data Structures - 2

- Features are polymorphic using a discriminated record type (ref Lecture 6)



Topolog2 Operations - 1

- Advance_Pos
  - Call in response to a sensor event provided this Pos is expecting it
  - Topologically moves to next node
  - Two versions:
    - Advance_Pos(Pos : in out Train_Position;
      Stopping : in Boolean;
      Need_Setting : out Boolean;
      For_Turnout : out Turnout_Id);

    - Advance_Pos(Pos : in out Train_Position;
      Stopping : in Boolean;
      Setting : in Turnout_Pos);

Topolog2 Operations - 2

- Check_Entering_...., ..._Leaving_...
  - Call when Pos is on a Sensor
  - Tells the significance of the sensor
  - Examples:
    - Check_Leaving_Crossing(Pos : in Train_Position;
      Leaving : out Boolean;
      Which : out Crossing_Id);

    - Check_Entering_Turnout (Pos : in Train_Position;
      Entering : out Boolean; Which : out Turnout_Id;
      Converging : out Boolean;
      Required_Setting : out Turnout_Pos;
      Chained : out Chain_Type);

Topolog2 Operations - 3

– Can find what block to acquire, release
  - Use arrays in package spec
– Example:

```
Check_Entering_Turnout ( My.Front_Pos, Entering,
   Which, Converging, Required_Setting, Chained );
if Entering then
  if Converging then
        Next_Block :=
          Turnout_Data(Which).Block_Num;
  ...
  else
        Want_Straight := ... -- calc using Straight_Is_Left
        if Want_Straight then
            Next_Block :=
              Turnout_Data(Which).Block_St;
```

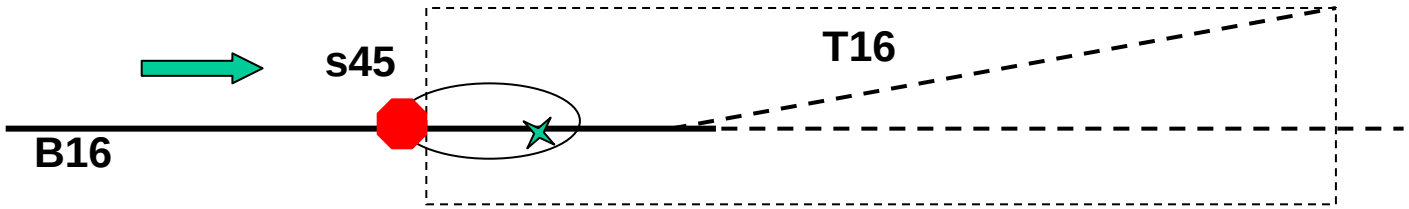
Topolog2 Operations - 4

- Turn_Around (Pos : in out Train_Position)
  - Call with front and rear Pos vars after swapping them
  - NB: physical train should be stationary (or both ends clear of sensors) else pending sensor events may be misinterpreted – beware!

- Resume (Pos : in out Train_Position;
  Setting : in Turnout_Pos))
  - Call after a train has skidded past a sensor and before train starts moving again
  - Ideally trains would stop dead on top of sensors, but ...

Skidding Scenario - 1

- Front magnet hits sensor 45 (ON event)
  - Expected by Front_Pos so call Advance_Pos on it
    - Front_Pos now (16,Normal_Pol,(s45),Norm,45)
  - Check_Entering_Turnout says entering turnout 16, diverging, but it isnt in correct position, so set DAC voltage to zero
- Front magnet sensor 45 OFF event
  - Turnout 16 still not ready so call Advance_Pos with Stopping=True
    - Front_Pos now (16,Normal_Pol,(s45),Just_After,45)



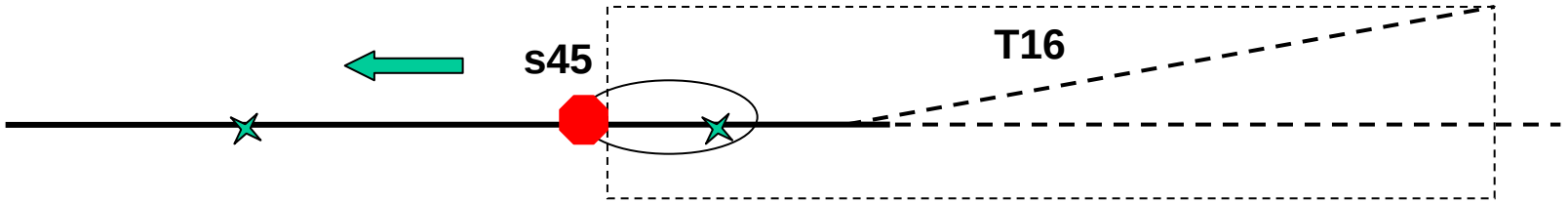
Skidding Scenario - 2

- Effectively the real 2nd Advance_Pos has been deferred
- All queries will behave as if the train had stopped on the sensor
  - ie they ignore substate value
- When turnout 16 is in correct position, call Resume
    - Front_Pos now (16,Normal_Pol,(T16),Norm,47)

Skidding Scenario - 3

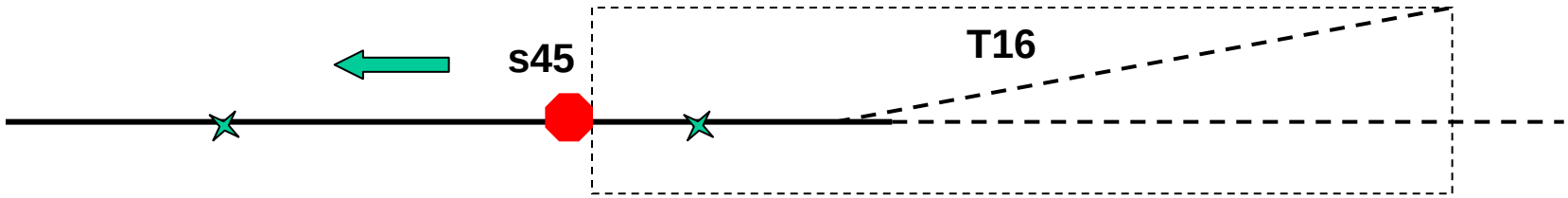
- The substate Just_Before exists in case you turn around the train while stopped
  - Maybe several times!
  - Swap pos vars and call Turn_Around

    Rear_Pos now (16,Reverse_Pol,(s45),Just_Before,45)



  - Call Resume (Rear_Pos, ...)

    Rear_Pos now (16,Reverse_Pol,(T16),Norm,45)



Questions

- The guard sensors are far enough back that trains don't skid into danger.
  - Is that true?
- What time interval to stop?
- What distance to stop?
- At what speed do trains stop within 10mm?
- What time interval between sensor edges at full speed?
- Will our software meet that deadline?