Railway Modelling in CSP||B

Steve Schneider
Helen Treharne

Faron Moller
Nga Hoang Nguyen
Markus Roggenbach

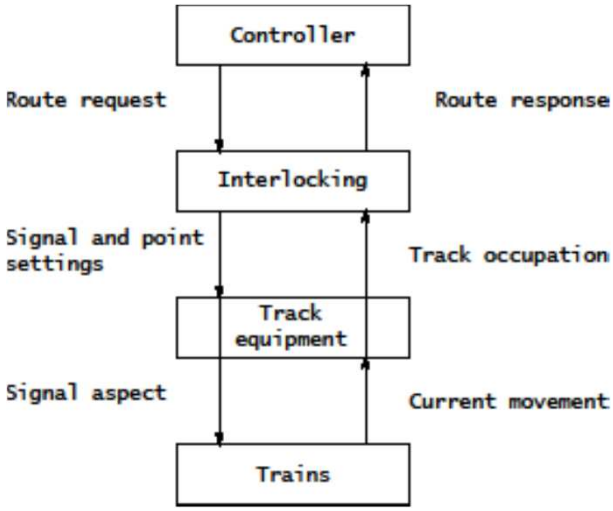
UNIVERSITY OF SURREY

Swansea University
Prifysgol Abertawe

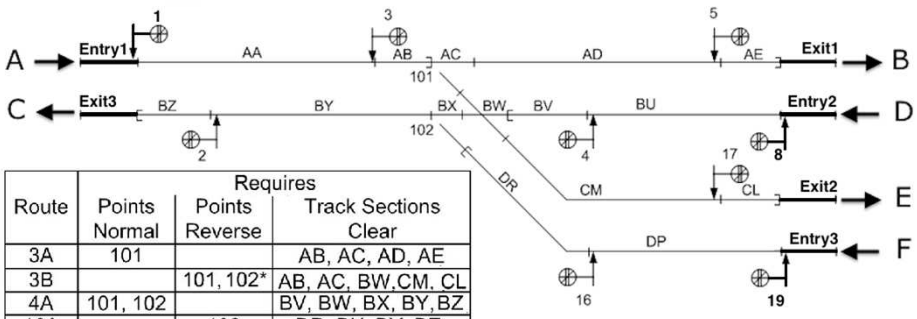
Aims of our approach

- "Natural modelling" to provide accessible and traceable formal specifications
- Tool support for verification with clear feedback to engineers
- Work done in close collaboration with Invensys Rail
  - Responsible for one of Britain's largest resignalling schemes (Thameslink) at London Bridge Station

Railway Interlocking



Example Track Scheme



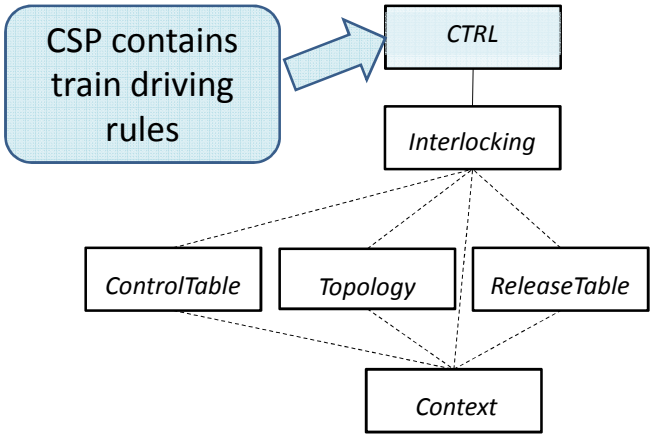
Route	Requires		
	Points Normal	Points Reverse	Track Sections Clear
3A	101		AB, AC, AD, AE
3B		101, 102*	AB, AC, BW,CM, CL
4A	101, 102		BV, BW, BX, BY,BZ
16A		102	DR, BX, BY, BZ

Railway's duality

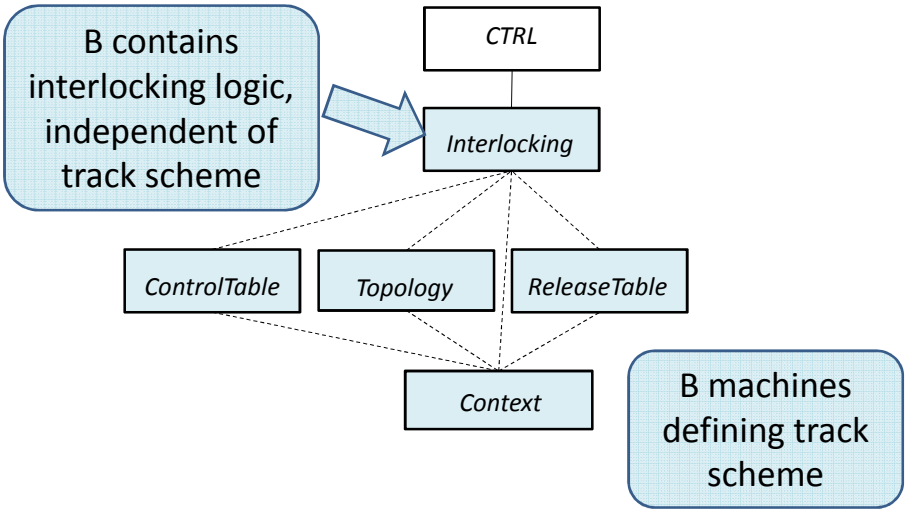
- Event-based: "train moved from track AA to track AB"
- State-based: "signal S3 shows proceed if tracks AB, AC, AD and AE are clear"

  then "route 3A is granted"
- (Same notion of state and event as Michael discussed this morning)

CSP||B Architecture



CSP||B Architecture



Example of CSP

$$
\begin{aligned}
&RW\_CTRL = \\
&\quad \sqcap_{r \in ROUTE}\ (request!r?b \rightarrow RW\_CTRL) \\
&\quad \sqcap \\
&\quad \sqcap_{r \in ROUTE}\ (release!r?b \rightarrow RW\_CTRL)
\end{aligned}
$$

$$TRAIN\_OFF(t) = enter!t?newp \rightarrow TRAIN\_CTRL(t, newp)$$

$$
\begin{aligned}
&TRAIN\_CTRL(t, pos) = \\
&\quad pos \notin EXIT \wedge pos \in SIGNALHOMES\ \&\ nextSignal!t?aspect \rightarrow \\
&\qquad \textbf{if}\ aspect == green \\
&\qquad \textbf{then} \\
&\qquad\quad move!t.pos?newp \rightarrow TRAIN\_CTRL(t, newp) \\
&\qquad\quad \sqcap \\
&\qquad\quad stay!t.pos \rightarrow TRAIN\_CTRL(t, pos) \\
&\qquad \textbf{else} \\
&\qquad\quad stay!t.pos \rightarrow TRAIN\_CTRL(t, pos) \\
&\quad \Box \\
&\quad pos \notin EXIT \wedge pos \notin SIGNALHOMES\ \& \\
&\qquad move!t.pos?newp \rightarrow TRAIN\_CTRL(t, newp) \\
&\qquad \sqcap \\
&\qquad stay!t.pos \rightarrow TRAIN\_CTRL(t, pos) \\
&\quad \Box \ \ldots
\end{aligned}
$$

$$ALL\_TRAINS = |||_{t \in TRAIN}\ TRAIN\_OFF(t)$$

Example of B Interlocking operation

```
currp, newp <-- move(t) =
PRE t : TRAIN & t : dom(pos)
THEN
   IF (pos(t) /: dom(nextd)) THEN
              LET track BE
                track = nullTrack
              IN
      currp := pos(t) ||
                pos(t) := track ||
                newp := track
              END
   ELSE
              LET track BE
                track = nextd(pos(t))
              IN
                currp := pos(t) ||
                pos(t) := track ||
                newp := track ||
                currentLocks := currentLocks - releaseTable[{track}] ||
                IF (pos(t) : ran(homeSignal))
                THEN
                   signalStatus( homeSignal~(pos(t))) := red
                END
              END
   END
END;
```

Example of B static information

clearTable : ROUTE --> POW(TRACK) &

clearTable = { A1 |-> {AA,AB},
A2 |-> {BZ},
A3 |-> {AB, AC, AD, AE},
B3 |-> {AB, AC, BW1, BW2, CM, CL},
A4 |-> {BV, BW1, BW2, BX, BY, BZ},
A5 |-> {AE},
A8 |-> {BU, BV},
A16 |-> {DR, BX, BY, BZ},
A17 |-> {CL},
A19 |-> {DP, DR} }

Verification Aims

- In AVOCS 2012
  - "correctness" of control tables and track plans
- In AVOCS 2010
  - correct implementation of control tables in Ladder Logic (James and Roggenbach)

Properties of Interest

- Safety expresses as temporal logic statements
  - Collision (AG (not (e(collision)))
  - Derailment (AG (not (e(derailment)))
- Liveness
  - Progress of movement

Collision Detection Encoding

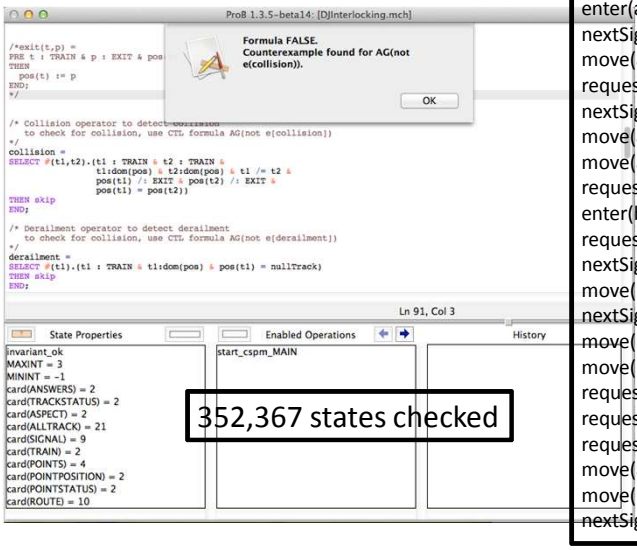
```
collision =
SELECT #(t1,t2).(t1 : TRAIN & t2 : TRAIN &
            t1:dom(pos) & t2:dom(pos) & t1 /= t2
             &
            pos(t1) /: EXIT & pos(t2) /: EXIT &
            pos(t1) = pos(t2))
THEN skip
END
```

Scenario - Faulty Tracks in ClearTable

```
clearTable :   ROUTE --> POW(TRACK) &

clearTable = { A1  |-> {AA,AB},
               A2  |-> {BZ},
               A3  |-> {AB, AC, AD, AE},
               B3  |-> {AB, AC, BW1, BW2, CM, CL},
               A4  |-> {BV, BW1, BW2, BX, BY, BZ},
               A5  |-> {AE},
               A8  |-> {BU, BV},
               A16 |-> {DR, BX, BY, BZ},
               A17 |-> {CL},
               A19 |-> {DP, DR}  }
```

Scenario - Faulty Tracks in ClearTable

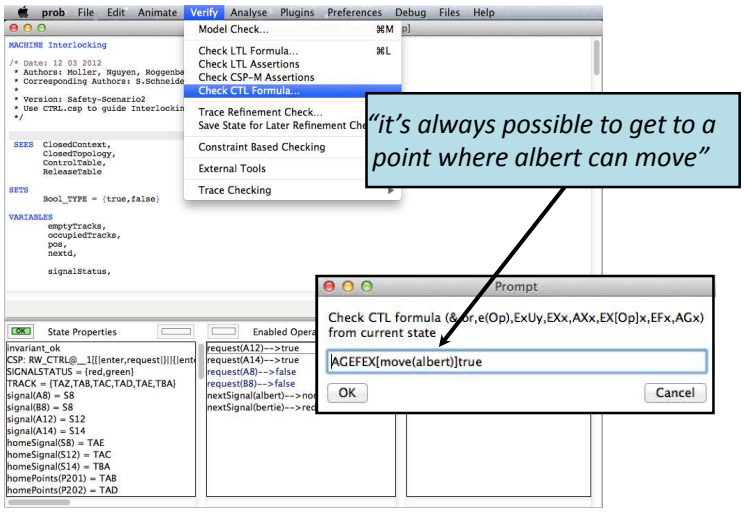


request(A1)-->yes
enter(albert,Entry1)
nextSignal(albert)-->green
move(albert)-->Entry1,AA
request(A3)-->yes
nextSignal(albert)-->green
move(albert)-->AA,AB
move(albert)-->AB,AC
request(A1)-->yes
enter(bertie,Entry1)
request(A3)-->yes
nextSignal(bertie)-->green
move(bertie)-->Entry1,AA
nextSignal(bertie)-->green
move(bertie)-->AA,AB
move(bertie)-->AB,AC
request(A16)-->yes
request(A5)-->yes
request(A2)-->yes
move(albert)-->AC,AD
move(bertie)-->AC,AD
nextSignal(albert)-->green

Summary of Scenarios

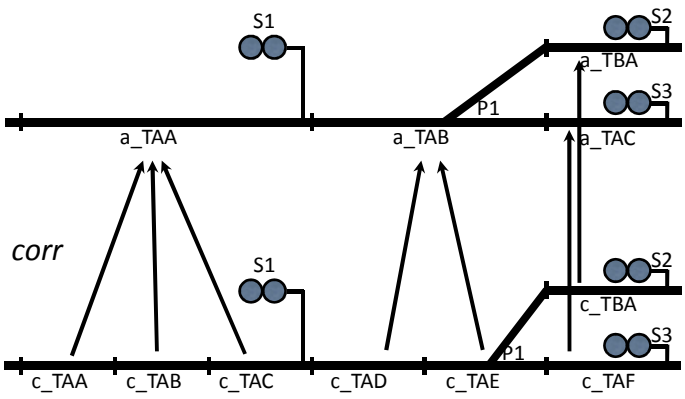
	States Checked	Results
B machines alone	X	Simple Violation of Collision
B machines and Restricted CTRL	240,655	No violation
Faulty clear tracks for a route in control table	352,367	Violation trace with 22 events leading to collision
Faulty points in control table	20,109	Violation with 10 events leading to derailment
Alteration to Release Table	85,052	Violation with 9 events leading to derailment

liveness checking:
albert never gets stuck



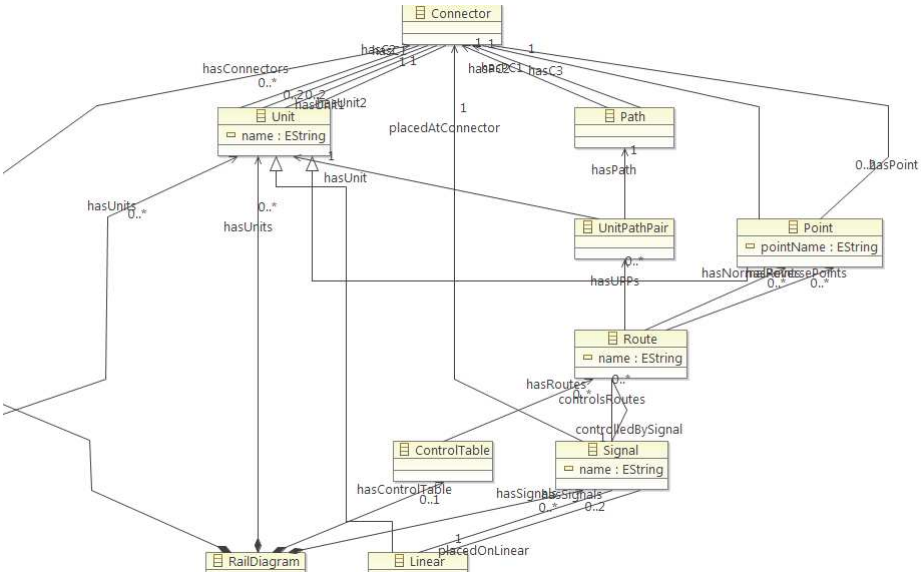
17

Ongoing work: defining abstractions



$corr : C\_ALLTRACK \twoheadrightarrow A\_ALLTRACK$

Railway DSL



Conclusion

- CSP contained driving rules
- B contained signalling rules, track plans and datatypes
- CSP||B models
  - Readable by railway engineers (validation)
- Ongoing work
  - Abstractions (HVC 2012)
  - Automatic generation from railway DSL using Epsilon (MDE framework)
  - Incorporating time

Single Lead Junction