Automated Generation of Verified Railway Interlocking Systems

Karim Kanso

Swansea University, Wales, UK

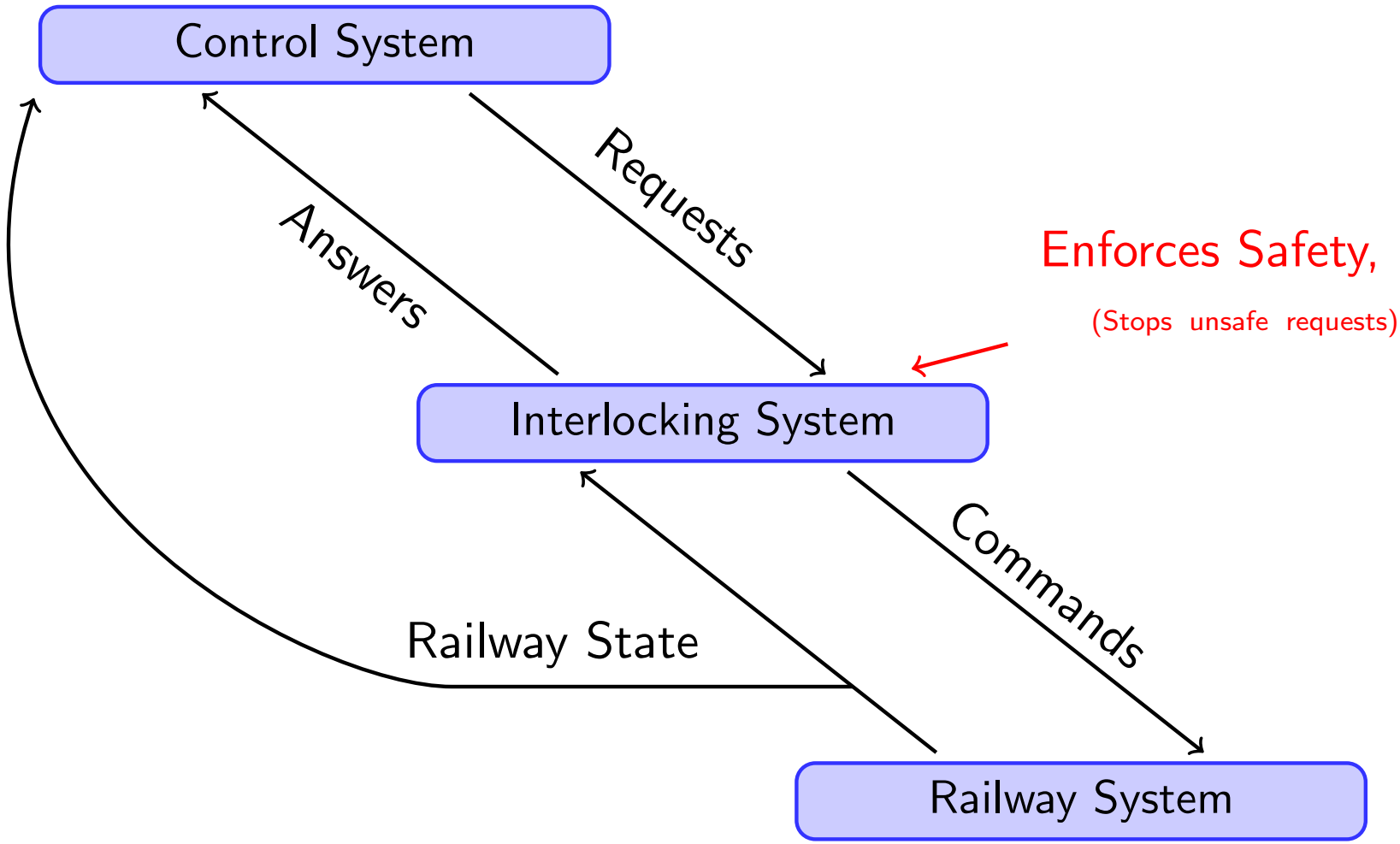
25th British Colloquium for Theoretical Computer Science
Warwick University 2009

This work is funded by: Westinghouse Rail Systems LTD, Chippenham, UK

Talk Outline

- Project Background,
- Project Outline,
- RDM: its views and generator,
- RDM narrative, time permitting *and*
- Current Progress.

Interlocking Systems



Background

Much work has been done with railway interlocking systems with respect to verification of safety properties.

- Swedish National Rail Administration – Lars Eriksson
- Danish & Chinese Railways (UNU-IIST) – Dines Bjørner
- UK Railways
  - Mathew Morley – 1996 Edinburgh
  - Wan Fokkink – 1998 Swansea & Amsterdam
  - Phillip James & Karim Kanso – 2008 (to current) Swansea

Also many more not listed.

Current Drawbacks in Industry

- Formal methods are not applied for development or testing, *and*
- Testing is done in the traditional way, i.e. books of signalling principles are converted into test cases. These test cases are manually tested.

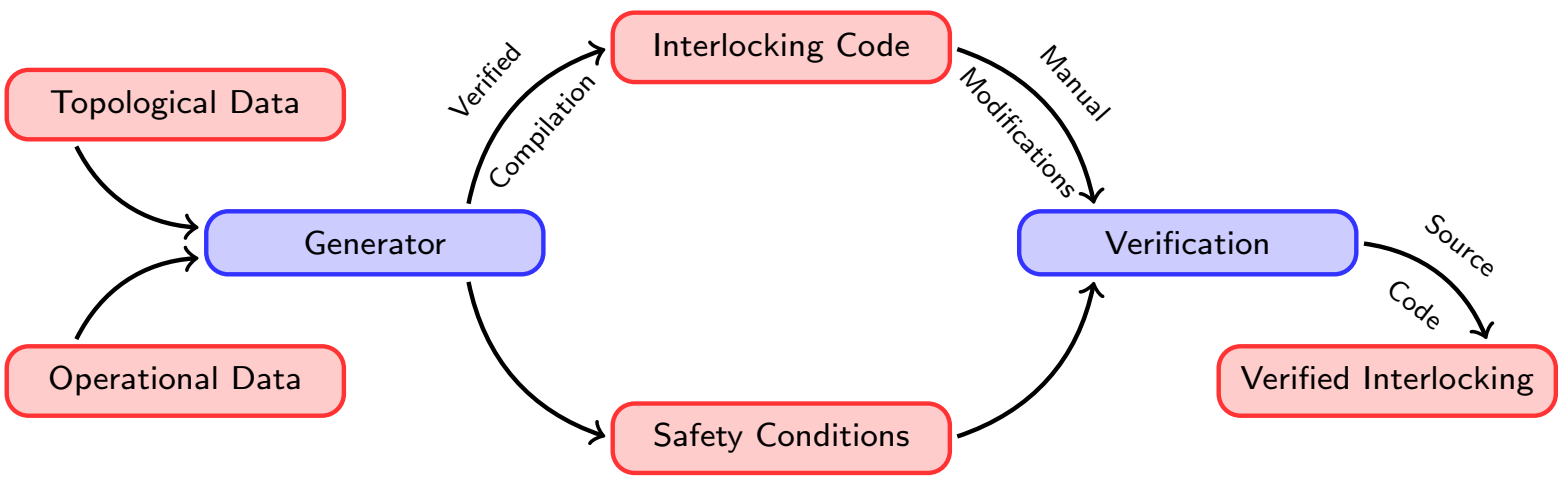
We wish to change this by

- automating the development process (as much as possible), *and*
- using verification to reduce resources for testing.

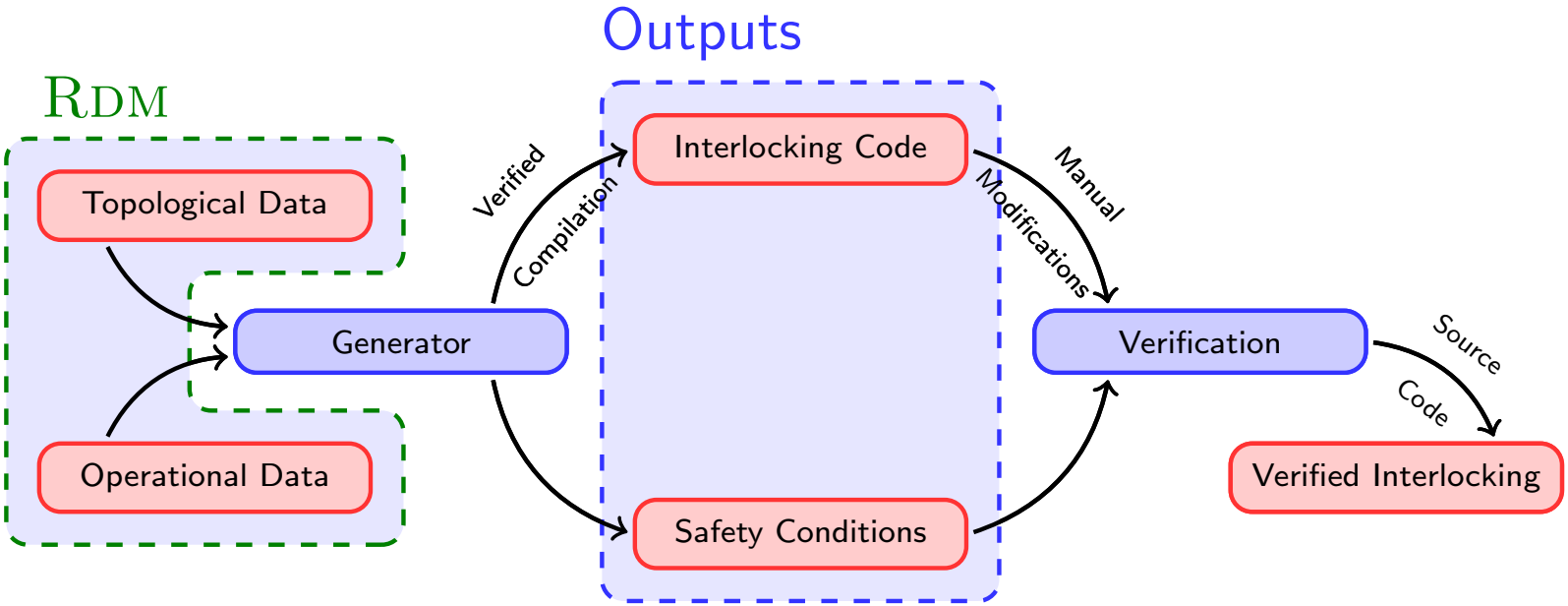
Verification is covered in a subsequent talk.

Practical Objectives

- *Generate* Interlocking Code from a Track Plan and Control Table; *and*
- Allow for the code to be modified after generation and still be verifiable.



Practical Objectives



- Verification has been researched within the railway domain many times:
  - SAT Technology
    - Lars-Henrik Eriksson
    - Prover (NP-Tools)
    - Phillip James and Karim Kanso
  - Model Checking
    - Mathew Morley (μ-calculus)

Project Outline

Here is a brief outline of this project, what we have done and want to accomplish.

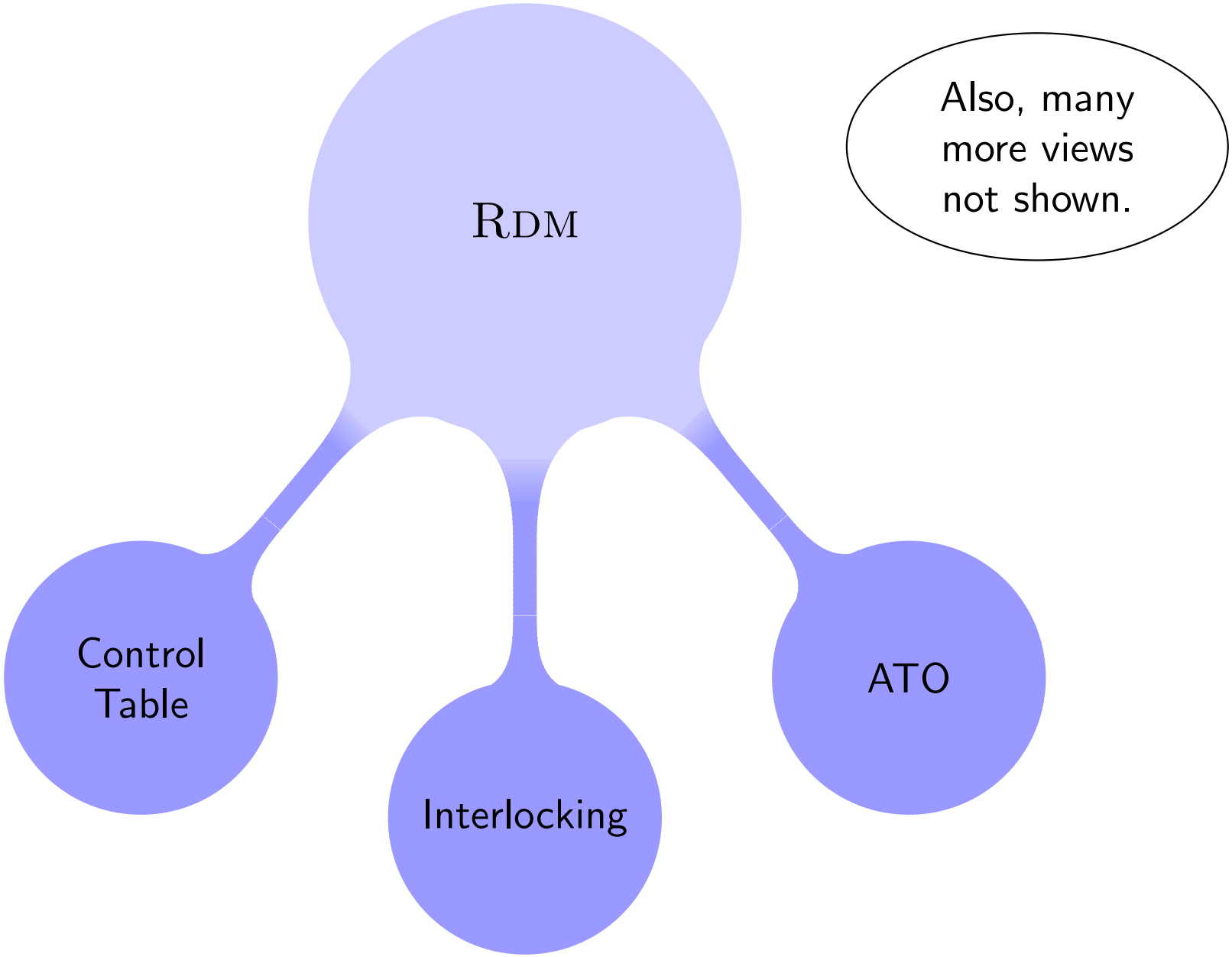
1. Define the RDM within a specification language

2. Define a view of this specification for the interlocking
   - Views can be taken for other purposes such as:
     - Control Centres
     - ATO / ATP
     - Future Research

3. Prove various safety properties about the specification

4. Refine the view to an implementation of the *generator*

RDM and it's Views



The Interlocking View (and its Generator)

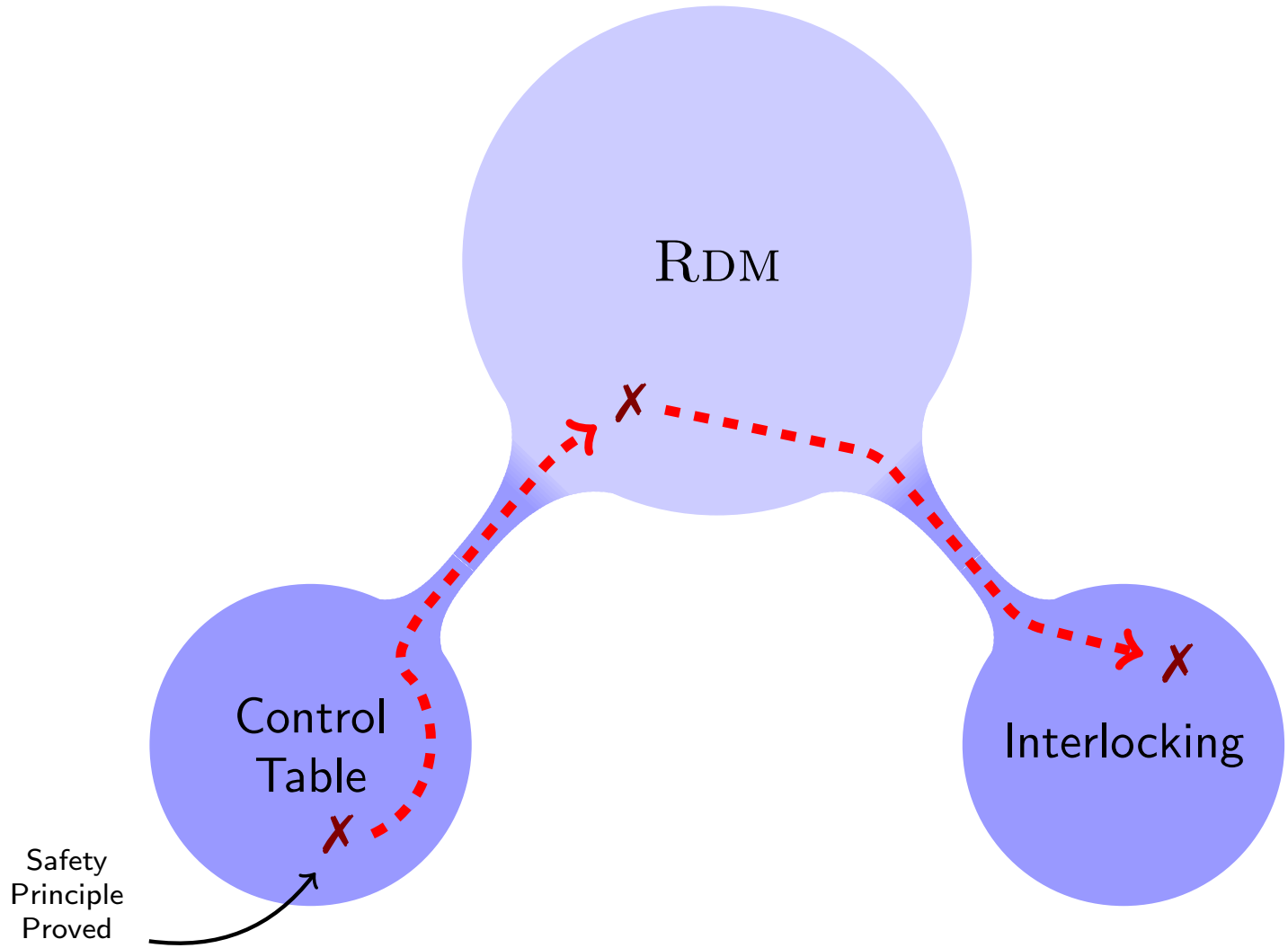
- The Interlocking View of the RDM can be seen as a relationship between the RDM and an interlocking systems logic.
  - X 'is an interlocking implementation for' RDM
  - i.e. There are different types of interlocking, with very different internal architectures programmed using the different languages. But implementing the same application logic.

- Given an RDM model MOD. The *generator* creates a concrete interlocking program PROG so that:
  - PROG 'is an interlocking implementation for' MOD

  holds.

Proving Properties

We wish to explore proving properties in one view, thus proving a property in the RDM and other views. (cf. Data Refinement, Roever)



Specifying the Railway Domain Model

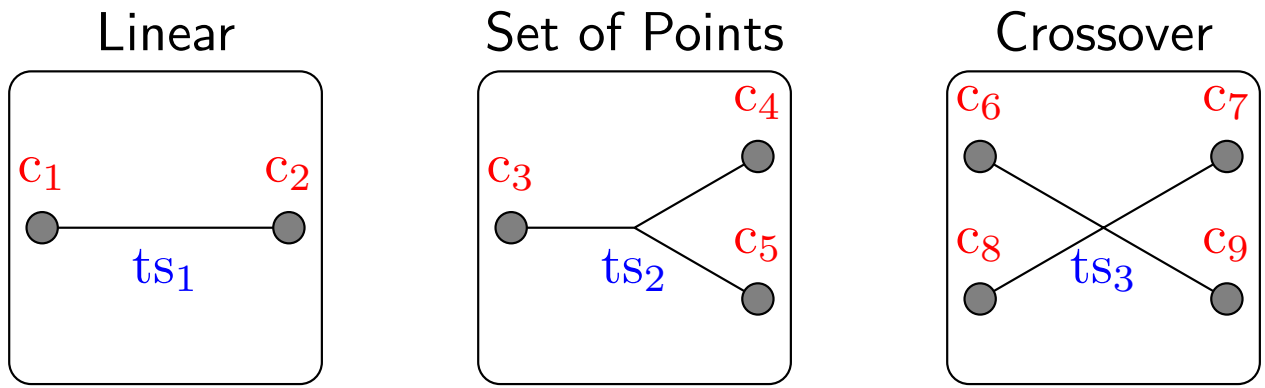
- Requires expert knowledge of the UK railways
- Subsequent processes depend upon the chosen representations
- Dines Bjørner
  - Many years Experience with Transport Domain
  - PRaCoSy Project at UNU/IIST using RSL
- Languages Considered:
  - CASL
    - Primary choice
  - RSL
    - PRaCoSy
  - Agda2
    - Proposed use for proving some theorems

A Narrative of the RDM

– Track Segments & Connections –

Track segments are the basic building blocks of the railway network, each track segment has

- ≥ 2 connectors (i.e. no terminal track segments), and
- track between connectors, and
- a unique identification.



These are example track segments, we purposely do not fix what they are for future compatibility. This is defined by a refinement.

A Narrative of the RDM (CASL)
– Track Segments & Connections –

spec TRACKWORK =

sorts *TrackSegment*, *Connector*

then ops *TrackSegmentConnections*,
: $TrackSegment \rightarrow Set[Connector]$

%% TrackSegmentConnections is not defined here, it
%% is refined at a latter stage.

pred *is_linear_tracksegment* : *TrackSegment*
$\forall$ *tracksegment* : *TrackSegment*
• *is_linear_tracksegment*(*tracksegment*)
$\Leftrightarrow \sharp$ *TrackSegmentConnections*(*tracksegment*) = 2

end

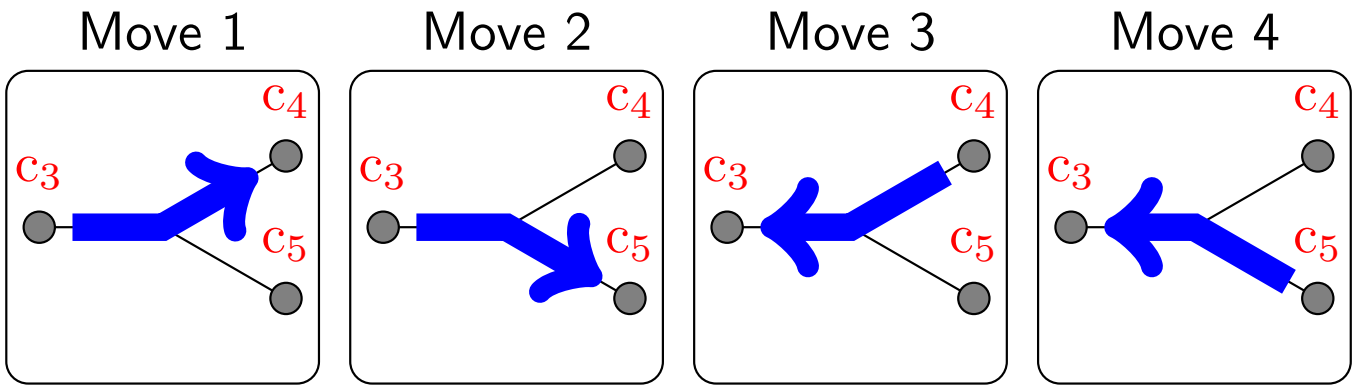
A Narrative of the RDM

– Moves –

A given track segment can have many different moves through it. Moves are:

- a representation of the abstract state of the track segment; *and*
- represented by pairs of connectors.

Example: Set of Points



Typically a set of points has two physical states but, each is bidirectional: yielding four moves.

A Narrative of the RDM (CASL)
– Moves –

spec TRACKSEGMENTMOVE =

TRACKWORK

then sort $Move =$
$\{p : Pair[Connector, Connector] \bullet \neg\ first(p) = second(p)\}$

op *tracksegment_moves*
$: TrackSegment \rightarrow Set[Move]$

op $tracksegment\_from\_move : Move \rightarrow ?\ TrackSegment$

sort $MoveList = List[Move]$

end

Current Progress

- This project started in January 2009 so is in its infancy;
- **Currently we have a specification written in CASL,**
  - **this is a modified and extended specification from PRaCoSy project;** *and*
  - **needs to be ratified by domain experts.**
- Also, an Agda2 implementation has been started.
- Future Work
  - Exploring relationships between different views, *and*
  - Defining the interlocking view of the RDM will soon start.