The SafeCap toolset for improving railway capacity while ensuring its safety

Alexander Romanovsky
Alexei Iliasov
Ilya Lopatkin
SafeCap Project

- Funded by RSSB/EPSRC
- Runs from January 2011 to August 2013
- Partners
  - Newcastle University
  - Swansea University
  - Invensys Rail UK
  - AIST (Japan)
Overview

Goal

methods and tools to improve the node and junction capacity while guaranteeing operational safety

Method

transfer formal modelling and verification techniques, developed by the computing science community
Overview

- Focus on **node** capacity
  - junctions and stations

- Verification of **signalling rules**
  - via a **formal model** of a railway schema and operation rules

- Assessment of **capacity**

- Improvement **patterns**
  - mechanised rules of schema transformation
SafeCap Domain Specific Language

- DSL describes a **data structure** defining
  - track topology
  - train detection circuits
  - routes
  - control tables
  - ...
- It is a **formal** language
  - axiomatic model semantics
Domain Specific Language
DSL Formal Semantics

- List of **axioms** describing all **valid** schemas
  - topology constraints
  - logical structuring
  - safety properties
- Operational rules defining train **movement**
  - animation
  - discovery of counter-example
  - basis of a continuous model for simulation
Capacity Assessment

- Simple formulae computed over DSL object
  - give a **first rough estimate**
  - variably sensitive to differing aspects of a schema
  - no train intertia

\[
TC_2(l) = \left( \max_{i,j} \min(V_{i\,\text{max}}, V_{j\,\text{max}}) \right) \left( \frac{1}{d_i} - \frac{1}{d_j} \right)^{-1}
\]
Capacity Assessment

Accurate simulation
- based on a hybrid formal model
- train acceleration/deceleration/point switching time, driver reaction, signal switching time
- runs a service pattern - periodic traffic schedule
- reports total execution time
- helps to identify bottlenecks (heatmap)
Capacity Assessment
Improvement Patterns

- Mechanically transformed schemas
  - written a dedicated scripting language
  - applied manually or in a batch
  - high-level abstraction: change some parts and the rest are computed automatically

- The way to capture and store knowledge
  - the typical changes attempted by a signalling engineer