Automated verification of rules and regulations in CAD models of railway signalling and interlocking

Bjørnar Luteberget and Claus Feyling

15th International Conference on
Railway Engineering Design and Operation

July 21, 2016

RailCOMPLETE

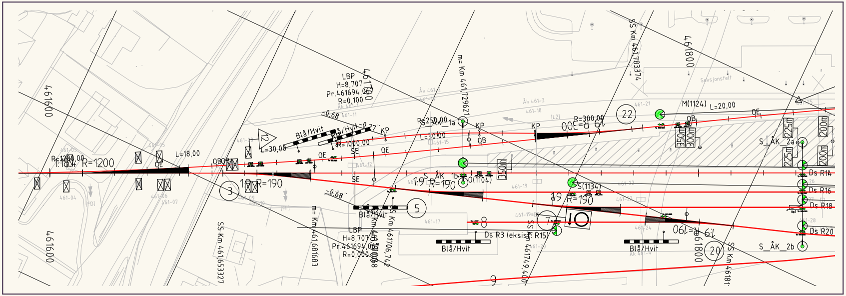
UiO : University of Oslo

Talk outline

1. Background and motivation
2. Semantic CAD using railML
3. Knowledge base design for verification
4. Prototype tool integrating this verification into existing engineering tools (RailCOMPLETE)

Designing signalling and interlocking

- Constructing new railway lines or improving existing ones requires through planning to meet quality demands
- Computer-aided design (CAD) tools are widely used for producing documentation
- Creating a good design takes much skill and effort

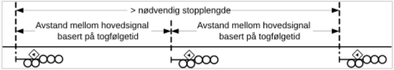


Technical regulations

- In our case study: Norwegian regulations from infrastructure manager Jernbaneverket

https://trv.jbv.no/wiki/Signal/Prosjektering/Lyssignal

e) Dersom nødvendig stopplengde er lengre enn avstanden mellom to etterfølgende hovedsignal, skal det benyttes gjennomsignalering ved hjelp av ATC (Signal/Prosjektering/ATC), se Figur 7.



Figur 7: Forsignalplassering og gjennomsignalering

f) Et forsignal skal plasseres på foregående hovedsignals mast dersom avstanden mellom det tilhørende hovedsignalet og det foregående hovedsignalet er ≤ 2200 meter.

g) Mellom et forsignal og det tilhørende hovedsignalet skal det ikke plasseres andre hoved- eller forsignal.

h) Et forsignal skal plasseres slik at siktavstanden oppfyller kravene til enten “brutt sikt” eller til “ubrutt sikt” i Tabell 4:

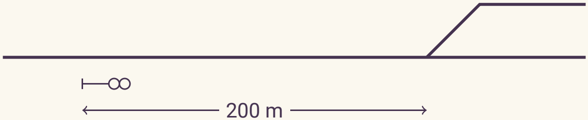
Tabell 4: Siktkrav til forsignal

Sikt	Strekningens høyeste tillatte kjørehastighet [km/h]																		
	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120	125	≥130
	Siktavstand [m]																		
Ubrutt	78	88	97	107	117	126	136	146	156	165	175	185	194	204	214	224	233	243	250

Technical regulations

Example from regulations:

- A *home main signal* shall be placed at least 200 m in front of the first controlled, facing switch in the entry train path.



- Many regulations fall into one or more of the following categories:
  - Object properties
  - Topological layout properties
  - Geometrical layout properties
  - Interlocking specification properties

Objective

Given a railway signalling and interlocking design,
verify that it complies with regulations.

Secondary objectives:

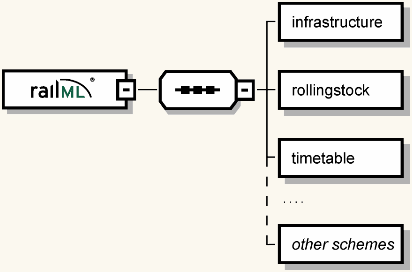
- Integrate with engineering/design tools
  - On-the-fly verification (“lightweight”)
  - Usable for engineers who are not formal methods experts
- Find suitable language for expressing regulations

Talk outline

1. Background and motivation
2. Semantic CAD using railML
3. Knowledge base design for verification
4. Prototype tool integrating this verification into existing engineering tools (RailCOMPLETE)

The railML XML standard data exchange format

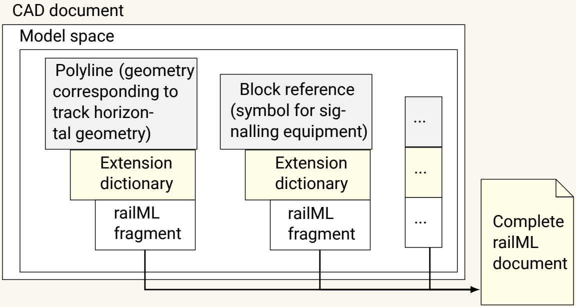
- Thoroughly modelled infrastructure schema
- First presented by Nash et al. at COMPRAIL 2004
- Development by international standard committee




```
<tracks>
  <track id="tr0" name="01">
    <trackTopology>
      <trackBegin id="x399" pos="0.000000" absPos="34
        <connection id="co399" ref="co397"/>
      </trackBegin>
      <trackEnd id="y151" pos="80.000000" absPos="346
        <connection id="co151_2" ref="co151_1"/>
      </trackEnd>
    </trackTopology>
    <trackElements>
      <speedChanges>
        <speedChange id="spu399" pos="0.000000" absPo
        <speedChange id="spd403" pos="30.000000" absP
        <speedChange id="spu405" pos="30.000000" absP
        <speedChange id="spd151" pos="80.000000" absP
      </speedChanges>
      <gradientChanges>
        <gradientChange id="gr399" pos="0.000000" abs
      </gradientChanges>
      <radiusChanges>
        <radiusChange id="ra399" pos="0.000000" absPo
      </radiusChanges>
      <platformEdges>
        <platformEdge id="pe399" pos="0.000000" absPo
      </platformEdges>
    </trackElements>
    <ocsElements>
      <signals>
        <signal id="si399" pos="0.000000" absPos="346
" code="6"/>
```

Embedding railML in CAD: “semantic CAD”

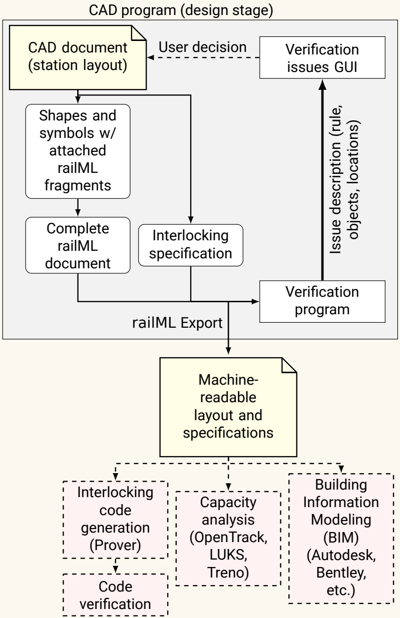
- Extending CAD objects with additional information gives railway-technical meaning to the symbols



CAD verification tool and tool chain

- Also, the structured data can be re-used for many other purposes, notably data exchange with other tools:
  - Interlocking code generation and verification
  - Capacity simulation
  - 3D view, Building Information Modeling
- This leads us to the tool chain overview...

Tool chain overview



- Static verification can discover violations of technical regulations early, as the user is building the model
- Dotted boxes indicate external programs
- More heavy-weight verification, simulation, testing, etc. benefits from machine-readable data exhcange

Talk outline

1. Background and motivation
2. Semantic CAD using railML
3. Knowledge base design for verification
4. Prototype tool integrating this verification into existing engineering tools (RailCOMPLETE)

Formalization of regulations

- Formalize the following information
  - The CAD design (extensional information, or facts)
  - The regulations (intensional information, or rules)
- Use a solver which:
  - Is capable of expressing and verifying the regulations
  - Runs fast enough for on-the-fly verification

Datalog

- Basic Datalog: conjunctive queries with fixed-point operators ("SQL with recursion")
  - Guaranteed termination
  - Polynomial running time (in the number of facts)
- Expressed as logic programs in a Prolog-like syntax:

$$a(X, Y) \mathrel{:\!-} b(X, Z), c(Z, Y)$$

$$\Updownarrow$$

$$\forall x, y : ((\exists z : (b(x, z) \wedge c(z, y))) \rightarrow a(x, y))$$

- We also use:
  - Stratified negation (negation-as-failure semantics)
  - Arithmetic (which is "unsafe")

Encoding facts and rules in Datalog

- The process of formalizing the railway data and rules to Datalog format is divided into three stages:
  1. Railway designs (station data) – facts
  2. Derived concepts (used in several rules) – rules
  3. Technical regulations to be verified – rules
- Now, more details about each stage...

Derived concepts

- Derived concepts are defined through intermediate rules
- Railway concepts defined independently of the design
- Example:

$$\textit{directlyConnected}(a,b) \leftarrow \exists t : \textit{track}(t) \wedge \textit{belongsTo}(a,t) \wedge \textit{belongsTo}(b,t),$$

$$\begin{aligned} \textit{connected}(a,b) \leftarrow \textit{directlyConnected}(a,b) \vee (\exists c_1, c_2 : \textit{connection}(c_1,c_2) \wedge \\ \textit{directlyConnected}(a,c_1) \wedge \textit{connected}(c_2,b)). \end{aligned}$$

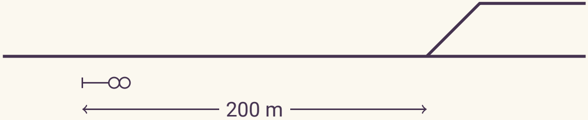
- A library of concepts allows concise expression of technical regulations

Technical regulations as Datalog rules

- Detecting errors in the design corresponds to finding objects involved in a regulation violation
- To *validate* the rules in a given design, we show that there are no satisfiable instances of the *negation* of the rule
- An example:
  - Home signal placement: topological and geometrical layout property for placement of a home signal

Rule example

- A *home main signal* shall be placed at least 200 m in front of the first controlled, facing switch in the entry train path.
- Uses arithmetic and negation



$$\mathit{isFirstFacingSwitch}(b, s) \leftarrow \mathit{stationBoundary}(b) \wedge \mathit{facingSwitch}(s) \wedge \neg(\exists x : \mathit{facingSwitch}(x) \wedge \mathit{between}(b, x, s)),$$

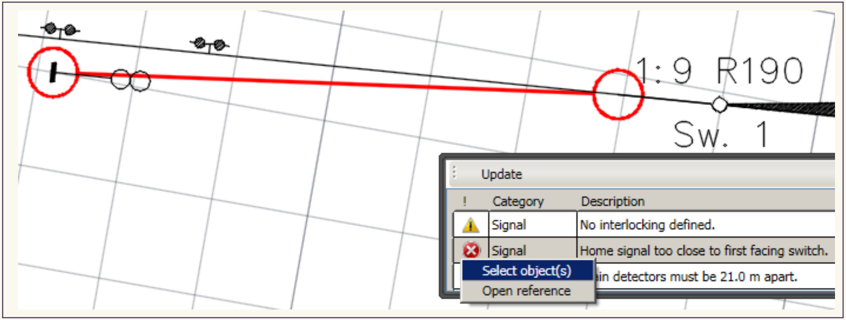
$$\mathit{ruleViolation}(b, s) \leftarrow \mathit{isFirstFacingSwitch}(b, s) \wedge (\neg(\exists x : \mathit{signalFunction}(x, \text{home}) \wedge \mathit{between}(b, x, s)) \vee (\exists x, d, l : \mathit{signalFunction}(x, \text{home}) \wedge \wedge \mathit{distance}(x, s, d, l) \wedge l < 200).$$

Talk outline

1. Background and motivation
2. Semantic CAD using railML
3. Knowledge base design for verification
4. Prototype tool integrating this verification into existing engineering tools (RailCOMPLETE)

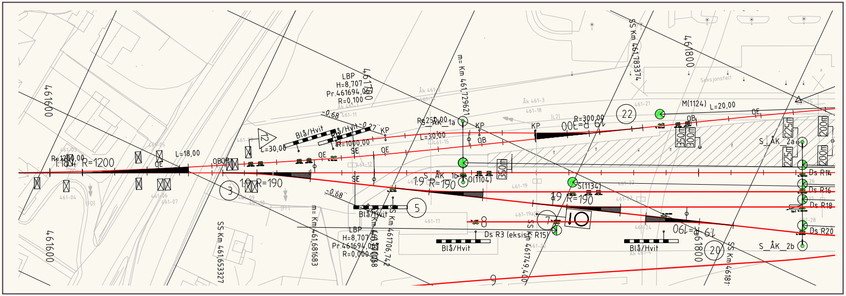
Prototype tool implementation

- Verification integrated in the RailCOMPLETE tool, based on Autodesk AutoCAD and XSB Prolog



Case study

- Railway engineers working on CAD model of Arna station (Norconsult AS / RailComplete AS), have thoroughly modeled using railML attributes



- Challenge: engineers want to understand and modify rules to better cover regulations, add edge cases, etc. Programming in Datalog is still outside railway engineer's competence.

Running time

	Testing station	Arna phase A	Arna phase B
Relevant components	15	152	231
Interlocking routes	2	23	42
Datalog facts	85	8283	9159
Running time (s)	0.1	4.4	9.4

- Running time for verification of a few properties: $\approx$1 – 10 s
  - More optimization needed for truly on-the-fly verification
- Challenge: Compute the verification so fast that the engineering/design process benefits from immediate feedback on changes.

Summary

- We have demonstrated a way to automate checking of regulations compliance for railway signalling and interlocking designs
- Our tools have been integrated in an existing CAD design environment
- Datalog allowed us to express technical regulations concisely and perform efficient verification
- Advantages:
  - eliminate tedious tasks, like filling out check-lists
  - get instant feedback on design quality while editing
  - make use of railML, a standard for describing railway designs

Future work

- Immediate feedback: use incremental evalulation of Datalog programs for efficiency
  - DRed algorithm, FBF algorithm
  - Tools such as XSB Prolog and RDFox support incr. eval.
- Involve engineers in knowledge base design: find user-friendly input language
  - DSL for expressing railway regulations
  - Controlled Natural Language, à la Attempto.