

SOFTWARE VERIFICATION RESEARCH CENTRE

SCHOOL OF INFORMATION TECHNOLOGY

THE UNIVERSITY OF QUEENSLAND

**Queensland 4072
Australia**

TECHNICAL REPORT

No. 00-39

Automatic Generation and Verification of Design Specifications

**Neil Robinson, David Barney, Peter Kearney,
George Nikandros, David Tombs**

December 2000

**Phone: +61 7 3365 1003
Fax: +61 7 3365 1533**

Note: Most SVRC technical reports are available via anonymous FTP, from [svrc.it.uq.edu.au](ftp://svrc.it.uq.edu.au) in the directory [/pub/SVRC/techreports](ftp://svrc.it.uq.edu.au/pub/SVRC/techreports). Abstracts and compressed postscript files are available via <http://svrc.it.uq.edu.au>.

Automatic Generation and Verification of Design Specifications

Neil Robinson^a, David Barney^b, Peter Kearney^a,
George Nikandros^b, David Tombs^a

^a Software Verification Research Centre, The University of Queensland, Queensland 4072, Australia
^b QR, 305 Edward Street, Brisbane, Queensland 4000, Australia

Abstract. We describe a project to develop a set of design tools for a railway signalling application, which is safety critical and has complex functional requirements. The toolset generates design specifications from functional requirements and verifies them against safety principles. The project addresses the overall engineering design process and has applicability to other domains.

INTRODUCTION

Designing software intensive systems is a challenging activity, which is well known to be prone to error. In safety-related applications, the consequence of design errors can be injury or death and thus it is extremely important to eliminate such errors where possible. Introduction of a systematic process is a good first step in reducing design error. Deployment of design tools, within a well-defined design process, can be very effective in improving the efficiency and correctness of the design.

QR, the major railway operator and owner in Queensland, Australia, recognises that the generation of signalling application designs is currently labour intensive and requires very specialised skills in railway signalling. Thus, a research project has been initiated with the Software Verification Research Centre, University of Queensland, which aims to develop a railway Signalling Design Toolset (SDT)

The QR signalling design process involves the following key stages:

- Design of the signalling layout in the form of an Arrangement Signalling Plan (*AS Plan*) which defines, for example, the positions of signals and points, and the permitted routes (paths) between signals.
- Production of a functional specification for the signalling of the layout. The specification is captured by a *Control Table*, which is developed from a set of general *Signalling Principles*.
- Implementation of the Control Tables in either (limited variability) software or electrical relays. The system that implements the functions embodied in the Control Tables is called an *Interlocking*.
- Verification of the implementation against the Signalling Principles through testing, a process

known as *Principles Testing*.

At present the bulk of design verification takes place late in the process, during Principles Testing. Principles Testing is highly reliant on the specialised skills and experience of the Principles Tester. QR wishes to reduce this reliance and increase the efficiency of the design process, without compromising design safety.

The research project is investigating the introduction of a Signalling Design Toolset (SDT) into the Control Table production phase. Three tools are envisaged: to generate a control table automatically; to allow a designer to manually edit a table; and to automatically verify an edited Control Table against the Signalling Principles. Verification of the control tables means that errors are detected earlier in the process, improving efficiency. With greater trust in the correctness of the tables, there should be less reliance on the expertise of the Principles Tester.

The SDT will be novel in the railway signalling domain. While there is some existing work on the generation of control tables (for example (Hachiga 1996) and (Cullyer 1993)) existing work on signalling design verification targets the later interlocking design phase. In contrast the SDT will enable earlier verification of designs expressed in the control tables.

Many of the issues which arise in this research project are applicable outside the railway signalling domain. In particular we expect the tools framework to be applicable wherever a generic safety-related product is configured for each installation according to generic rules – for example Safety Instrumented Systems in process plants and Road Traffic Controllers. In addition, the following key aspects of this project are relevant generally to the use of formal design tools:

- design tool dependability in safety-related applications
- modelling of the design process and the role of design tools within it
- hazard analysis of design tools in the design process
- use of formal methods
- use of state-of-the-art automatic reasoning

- techniques
- maintenance and support of large formal specifications

These issues are discussed in the following sections.

THE APPLICATION

The signalling functions controlling train movements at a particular location are together known as an *Interlocking* (because setting one movement locks out other movements). An interlocking is typically defined in terms of a set of *Routes* from one signal to another. It specifies the conditions necessary to clear a Route, including states of *Track Segments* (occupied or clear), *Points* (normal or reverse) and *Signals* (proceed or stop). A *Control Table* is a tabular presentation of this information.

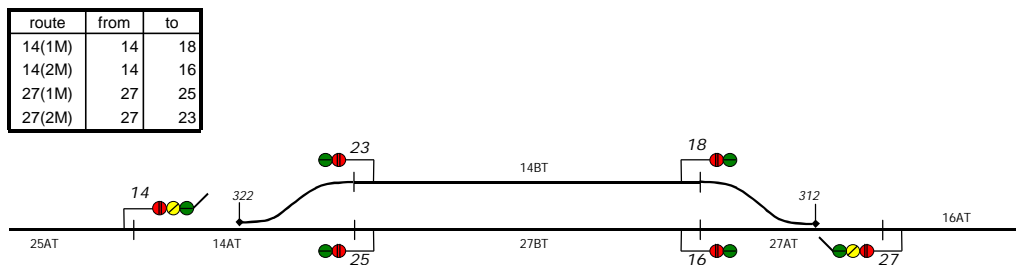


Figure 1 - Sample Layout

Figure 1 shows a layout for a simple crossing loop, comprising six Track Segments, two Points, and six Signals, all labelled. The interlocking for this layout has a number of routes, four of which are indicated in the box.

Control Tables have their own grammar and terminology. A section of the Control Table corresponding to Figure 1 is presented in Figure 2. This layout and Control Table is schematic only, and does not necessarily reflect actual QR practice. It shows some of the conditions for clearing Route 14(2M) from Signal 14 to Signal 16. It may be seen that the in-route tracks 14AT, 27BT and overlap 27AT shall be clear, and that the points 322 and 312 shall be set to allow passage along the route. Other aspects of the table concern locking out opposing routes when a train is passing along the route and when it has stopped in a route segment.

Actual control tables are considerably more complex than this example suggests. As well as in-route tracks, points and signals, an interlocking must specify overlaps (safety margins beyond signals),

shunt movements, standing times and approach-side locking of signals. There are also rules governing railway equipment such as level crossings, rules restricting the movement of trains (e.g. non-wired routes) and, in Queensland, rules for the operation of mixed-gauge track. Also, at many locations special rules apply that are not covered in the generic Signalling Principles. Local rules may be more restrictive or less restrictive than the Signalling Principles.

A complete SDT should handle all of these situations, either automatically or with manual intervention.

THE SIGNALLING DESIGN TOOLSET (SDT)

The basic concept of the Signalling Design Toolset (SDT) is shown in Figure 3. (In Figures 3 and 4, coloured boxes denote parts of the SDT which must be created once only; white boxes denote processes and objects that must be created fresh for each interlocking. Computer symbols next to a box indicate an automatic process.)

Control Tables are automatically generated based on a formalised track layout, which must be generated from the AS Plan, and a set of *Control Table Generation Algorithms*. There is some existing work in this area, for example (Hachiga 1996) and (Cullyer 1993), which demonstrate the feasibility of automatic Control Table generation. However, as suggested above, some parts of the functionality are extremely difficult to design correctly, and currently we consider that it will not be possible to automatically generate 100% of the Control Tables. Accordingly, we allow the designer to edit the automatically generated tables.

Signal	Route Number	Route to	Route Indication	Requires								
				Points Locked		Signals		Route Holding	or Until		Tracks	
				Normal	Reverse	Normal	Reverse	Maintained by tracks occ	Tracks occ	for Time secs	Clear	Occ
14	2M	16	-	322								14AT 27BT 27AT
				312				14AT 27BT	27BT	30		
						25		14AT				
						27		14AT 27BT				

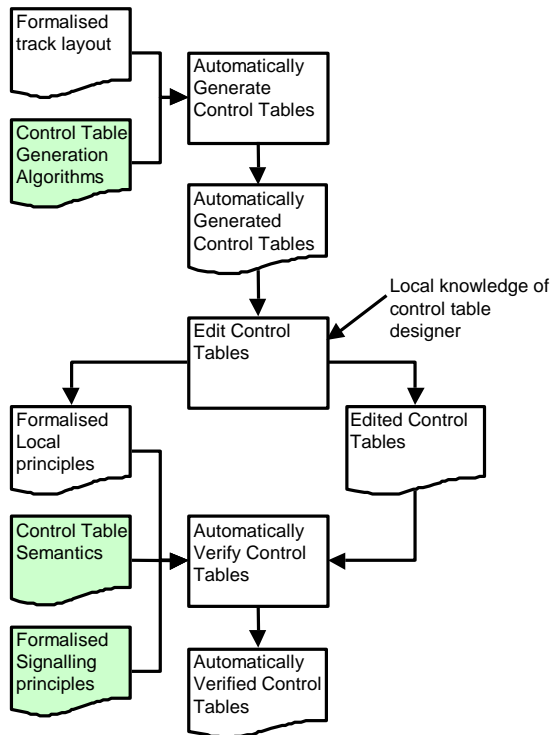


Figure 3 – The Signalling Design Toolset Concept

The editing process allows the addition of ‘difficult to design’ parts of the functionality. It also allows local exceptions to the generic Signalling Principles to be incorporated into the Control Tables. Where the local rules violate or add to the generic Signalling Principles, the local ‘Principles’ must also be formalised by the designer for later use by the verification tool. This means that the formalism used must be understandable to the signalling engineers who need to use it in practice – an important constraint on the choice of formalism.

The edited Control Tables are then automatically verified against the Signalling Principles. This is the most complex part of the process and is shown in more detail in Figure 4. The formalised Signalling Principles and Local Principles are merged together. Merging is not a trivial operation, because it must resolve contradictions between the local and generic Principles. The merged Signalling Principles are then applied to the particular track layout. Generic Signalling Principles can be thought of as predicates, quantified over sets of signalling objects (e.g. Signals, Points, Track Segments). As described in (Eriksson, 1996), the act of applying the principles to the layout instantiates the predicates. The output is a set of propositional formulae – the *Applied Signalling Principles*.

Before the verification can take place, the (edited) Control Tables must be converted to formulae, using a predefined set of *Control Table Semantics*. Finally, the Control Table formulae are verified against the Applied Signalling Principles. A successful verification proves that if all the Control Table formulae are true then all the formulae in the

Applied Signalling Principles are true. In other words, the Control Tables satisfy the Signalling Principles. Our intention is that the verification will be complete, i.e. the tool will verify 100% of the Control Table content. Note that the Signalling Principles are primarily functional safety requirements – they specify what should not be permitted to happen, rather than what should be permitted. For example, the Signalling Principles state that “Points shall be unconditionally locked by occupancy of the track containing the points”, but they do not state the conditions for unlocking the points. Thus, a successful verification demonstrates the *safety* of the Control Tables, but does not demonstrate the *liveness* of the Control Tables. In our example, even if the Control Tables were successfully verified, there is no guarantee they would ever permit the points to be moved. Further discussion of the verification process is provided later in this paper.

The introduction of the SDT will modify the existing QR signalling design process. In particular, it will reduce the emphasis on Principles testing and will encourage more trust in the Control Tables. The future process will also include an additional step of testing the implementation directly against the Control Tables.

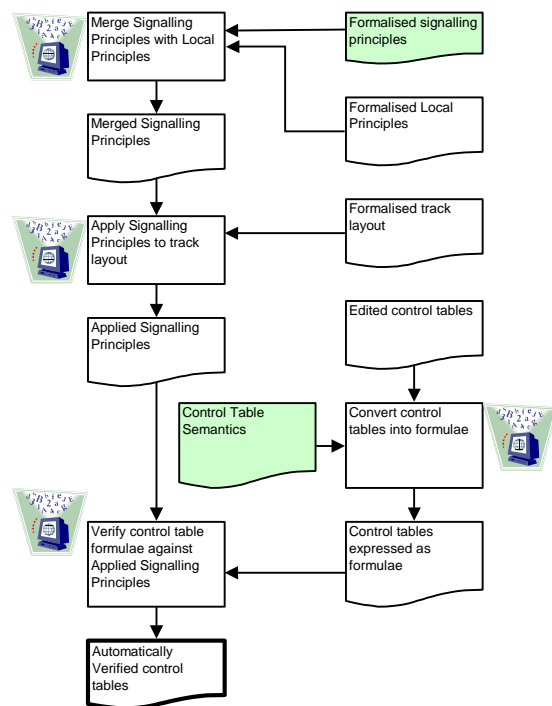


Figure 4 - Automatic Verification of Control Tables

STANDARDS RELATING TO DESIGN TOOLS FOR SAFETY-RELATED APPLICATIONS

Signal Interlockings (in software and/or hardware) play a vital role in the safety of railway operations. They are designed to ensure that operation of trains is safe. Errors in interlockings can lead to unsafe operation of trains and thus to possibly fatal

accidents.

The SDT plays an important role in the interlocking design and development process by generating and verifying interlocking designs as represented in the control tables.

Traditionally, standards for development of safety-related systems have not focussed on tools used to support the development process such as the SDT, instead focussing on actual operations software and hardware. Use of development tools is often regarded as enhancing the integrity of the development process, but the integrity of the tools themselves is not specifically addressed.

Clearly, if development tools are relied upon in the development process, then a degree of integrity is required in those tools, depending upon how they are used and the extent to which they are relied upon.

There is little explicit guidance available on assessing the required integrity of development tools. The British Computer Society has issued a document (BCS, 1997) giving some guidance on the adoption of such tools. This guidance points out a number of concerns in the use of development tools that are potentially applicable to the SDT:

- there is potential for new risks to be introduced by additional computerisation when insufficient care has been taken to develop the tools for its intended use in a safety context;
- the availability of sophisticated tools can create a false impression that fewer human resources or personnel with lower levels of competence can be employed;
- someone may use a tool who is not competent to understand its usage or limitations, and who may therefore place undue and uncritical trust in the tool's output.

The BCS guidance draws a distinction between tools that directly generate or affect object code, such as operating systems and compilers and other tools. It suggests that tools can be classified into levels of influence on the safety of the target applications:

- **Direct influence:** where the tool directly embeds code or directly affects the software structure, configuration or data and could directly insert errors
- **Indirect influence:** for example, by affecting requirements or run-time efficiencies
- **V&V influence:** for example, a testing tool that only has the potential to fail to detect an error introduced elsewhere in the lifecycle.
- **No influence:** for example, a tool that affects project management only.

In terms of this classification the Control Table generation and editing tools would be of 'indirect influence' and the Control Table Verification tool would be 'V&V influence'. (A possible future extension would be to use the control tables to generate interlockings directly. In that case the

control table generator would be 'direct influence'.)

The guidance suggests that 'the combination of this classification for a tool and Safety Integrity Level (SIL) for the application could give a measure of the importance of the tool to the project'. However no more specific guidance is given.

The railway standard prEN50128 (CENELEC prEN 50128, Clause 18) includes guidance for systems configured by application data, specifically the development of application data that directly affects the operational software through being accessed by the operational software or being used to configure the operational software. Such data has a more direct influence than the Control Tables. Clause 18.4.2.3 requires that an integrity level be allocated to any hardware or software tools used in the data preparation lifecycle. 'This integrity level shall be derived from the required integrity level for the data, and the degree to which the output of each tool is checked by other manual or automated procedures'.

The standard DO178B (DO178B, clause 12.2) gives some guidance on qualification of development tools. It focuses on tools that eliminate, reduce or automate activities of the software life cycle. The objective of the tool qualification process is to ensure that the tool provides confidence at least equivalent to that of the processes eliminated, reduced or automated. Software tools are classified as one of two types:

- **Software development tools** whose output is part of operational software
- **Software verification tools** which cannot introduce errors but may fail to detect them

In terms of these classifications the Control Table generation and editing tools are software development tools and the Control Table verification tools is a software verification tool.

DO 178B requires that for software development tools the software level assigned to the tool should be the same as that of the software it produces, unless a reduction in software level can be justified on the basis of the importance of the function performed or the likelihood that other verification activities would have detected the same errors. In any case, software development tools should be verified to comply with their operational requirements by means of review, demonstration, requirements-based coverage analysis, structural coverage analysis, robustness testing and analysis of potential errors produced by the tool.

For software verification tools the standard requires demonstration that the tool complies with its operational requirements under normal operating conditions is required.

In summary, standards such as IEC 61508 (IEC 61508), CENELEC prEN50128 and DO178 all provide relevant guidance on techniques to be employed once an integrity level is assigned to the software. The usual methods for assigning integrity in IEC 61508 and CENELEC are not regarded as applicable without modification to development tools.

Guidance that is available on assigning integrity levels to development tools requires an assessment of the importance and role of the development tools in the development process and the criticality of the systems they are used to produce.

HAZARDS AND REQUIRED INTEGRITY OF THE SIGNALLING DESIGN TOOLSET

An early analysis of the risks of the SDT provides guidance in its design, development and use. Introduction of the SDT changes control table verification from a wholly manual process to a partially automated one. There are risks associated with both approaches, and it remains to be justified that the risks of the modified process are no higher than the risks of the original process. An initial safety case has been constructed to assess the hazards and risks of the SDT, which will be updated as the project evolves.

The SDT does not directly control the setting of signals or any other aspect of railway operation. Nor does it directly generate interlocking code. Rather the tools are used to generate and verify aspects of interlocking requirements and design.

There is good reason to think that use of the toolset should enhance the integrity of the interlocking design and development process. Standards for the development of safety related systems, for example (IEC 61508) and (CENELEC prEN0128) recommend the use of a number of techniques which will be embodied in the SDT, for

formal proof that the control table design satisfies safety requirements as specified in the Signalling Principles. In itself the formalisation of Signalling Principles and Control Table semantics should contribute to the precision, consistency and completeness of these requirements and design elements.

The proposed new signalling design process is a definite modification of the existing one, in which the SDT plays an integral role. Human checking and testing is to some extent reduced, in that the principles test is reduced from its current scope and there is increased reliance on the correctness of control tables in the testing process. New hazards may be introduced (e.g. misunderstanding of the formalisation) and hazards may arise during the changeover to the new process. Thus it is important to analyse the proposed new process and the role of the SDT in it.

Errors generated by the SDT or which fail to be detected by the SDT could, if not detected elsewhere in the process, lead to accidents involving multiple loss of life.

The hazards of the SDT have been identified through application of the Functional Failure Analysis (FFA) technique to the proposed design process incorporating the SDT. In this technique, a functional description of the system under consideration is analysed, by considering the possible failure modes of each function. Possible failure modes are identified using a simple set of guidewords. The use of FFA to identify hazards is further described in (Lindsay 2000). A sample of the

Process	Deviation	Cause	Local	Verified Control Table	Global	Mitigations
Automatically generate control tables	Omission	Software wrong. Hardware failure.	Control table missing some safety controls.	Could have incomplete control table, if same error present in verification part.	Accident	Verification process. Manual inspection during editing process. Possible verification against signalling principles. Consider some completion checks against AS Plan.
	Commission	Software wrong. Hardware failure.	Control table has extra safety controls	Too restrictive control tables	Inefficient operation	Minimal solution check Control table editing process Principles test
	Wrong	Software wrong. Hardware failure.	As for omission / commission, or garbage generated, or nothing generated	As for omission / commission, or control table garbage	As for omission / commission	Verification process

Figure 5- Sample Functional Failure Analysis

example formal methods in requirement specification (see e.g. IEC 61508-7 Annex A Clause 9), formal methods in software design and development (IEC 61508-7 Annex A Clause 11) and formal proof for Verification and Testing (IEC 61508-7 Annex A Clause 12). The signalling principles can be regarded as safety-related requirements that are formalised for use in the SDT. The control tables are a representation of interlocking requirements and design, the semantics of which are formalised for the verification process. The control table verification is a

FFA for the SDT is illustrated in **Figure 5**. It analyses the “Automatically generate control tables” function of **Figure 3** and reveals that omission of a control could lead to an accident, whereas an extra control could lead to inefficient operation.

In addition to identifying hazards, the FFA process resulted in a number of modifications to the proposed design process to improve its safety, and identified a number of possible additional checking tools that could be used to enhance the integrity of the design process.

We intend to set Safety Integrity Level targets for functions of the SDT on the basis of a claim that the modified signalling design process, with the SDT in place, is as safe as the existing signalling design process. This involves identifying the changes between old and new processes and quantifying the effectiveness of those parts of the current process that will be reduced in the future process. The quantification will be based on data gathered from design and testing records. This process will also consider the risk that Designers will put more trust in the automatically verified control tables and thus manual checks on the Control Tables will be less effective than at present.

An initial safety case has been produced based on the description of the SDT concept. This is intended to increase confidence in the concept and to highlight any important safety issues early. Often, safety cases are not produced until much later in the development lifecycle, when it is costly or impossible to address the safety issues they raise. Early attention to the hazards which may arise enables appropriate safety requirements to be captured early in the tool design process.

AUTOMATIC VERIFICATION OF DESIGN SPECIFICATIONS

As described earlier, the automatic verification process ultimately involves a proof that one set of propositional formulae satisfy another set of propositional formulae. The number of variables in a typical interlocking is large. For example, the number of track segments, each of which can be in one of two states, may typically be 40. Thus, the number of possible combinations of occupation of track segments (without consideration of all the other variables in a railway) is 2^{40} – a state space which appears to be infeasible to deal with. The problems associated with such large state spaces are known as *state explosion* problems.

Fortunately, trains move in a reasonably predictable manner, occupying track segments sequentially. Based on assumptions of how trains move through a layout, there are a limited number of states which can follow on from the current state of the railway (defined by the current state of all the track segments). This reduces the state space significantly and should make the automatic verification a feasible problem to solve.

There are existing tools available that are capable of automatically performing proofs in propositional logic. One example is the NP Prover tool as described in (Sheeran, 2000) and (Petersen, 1996). This tool uses a patented proof method which has been shown to be efficient in certain large industrial, including railway, applications. Another alternative technology is model checking, as described in (Cimatti, 1998) and (Simpson, 1997), which checks models defined

in process algebras such as CSP or FDR. These technologies provide techniques specifically designed to address the state explosion problem.

The research project is currently investigating the available technologies for automatic theorem proving.

SUPPORT AND MAINTENANCE ISSUES

A key factor in determining the success of the SDT will be whether or not it can be integrated effectively into the signalling design process and whether the toolset can be supported and maintained effectively.

The SDT will, to a large extent, rely on the correctness of Signalling Principles, Control Table Generation Algorithms and Control Table Semantics. These are all artifacts that will need to be maintained by the signalling engineers. The Signalling Principles are particularly large and complex and will need to be maintained in both formal and informal (English language) forms. The formal and informal version will need to be traceable, in that if one changes it will be vital that the other changes to suit. Thus the choice of formalism and the presentation of the formal specification alongside the informal specification is of utmost importance.

When someone formally specifies requirements, it greatly improves their understanding of the problem domain. This has proven to be a key strength of formal specification. In our case, we expect the formal specification of the signalling principles to greatly improve the understanding of the signalling principles themselves. However, historically formal specification has not proven to be so effective in communicating such understanding to other people. Non-experts consider the formal notations to be hard to understand, and even experts make mistakes in interpreting the notations. There are therefore risks in the use of formal specification which will need to be addressed in the project.

CONCLUSIONS

We have presented a tool framework which we believe can support practical automatic generation and verification of railway signalling designs. It incorporates:

- generation and verification of intermediate designs in the form of control tables
- partial automatic generation with provision for human editing
- the use of local safety principles which may extend or modify general safety principles
- derivation of relevant safety principles applicable to the particular design before formal verification

We believe these ideas are applicable beyond the

signalling design domain.

Work already undertaken on the SDT demonstrates the utility of early attention to the role of design tools in the overall design process and hazard analysis of the tool functionality in that context. This work enables early identification of safety requirements and early assessment of the required integrity of the tools.

We have identified issues relating to ongoing maintenance of formalised safety principles as significant and requiring further attention on this project.

We have good reason to believe that automatic verification of control tables within our framework will be of feasible complexity using known automatic verification methods. Investigations have already begun to determine the most appropriate technologies for the application.

REFERENCES

- BCS, *Guidance for the Adoption of Tools for use in Safety Related Software Development*, British Computer Society / Institute of Electrical Engineers, 1997
- CENELEC PrEN 50128, *Railway Applications: Software for railway control and protection systems*
- Cimatti, A; Giunchiglia, F; Mongardi, G; Romona, D; Torielli, F; Traverso, P; *Formal Verification of a Railway Interlocking System using Model Checking*, Formal Aspects of Computing (1998) 10: 361 – 380
- Cullyer, John; Wong, Wai, *Application of formal methods to railway signalling - a case study*. Computing and Control Engineering, February 1993
- Eriksson, LH, *Specifying Railway Interlocking Requirements for Practical Use*, SAFECOMP 1996
- Hachiga, A, *Algorithmic approach to the verification of a railway interlocking table*. Computers in Railways V, Vol 1, pp 91-100; International conference on computer aided design, manufacture and operation in railways and mass transit systems, 1996
- IEC 61508, *Functional safety of electrical / electronic / programmable electronic safety-related systems*, 1999
- Lindsay, PA; Tombs, DJ; McDermid JA; *Deriving Quantified Safety Requirements in Complex Systems*, SAFECOMP 2000
- Petersen, JL; *Formal Requirement Verification of a Swedish Railway Interlocking System*, Technical Report of the Technical University of Denmark, 25 September 1996
- RCTA/DO-178B, *Software Considerations in Airborne Systems and Equipment Certification*, RCTA, December 1992
- Sheeran, Mary; Stalmarck, Gunnar; *A Tutorial on Stalmarck's Proof Procedure for Propositional Logic*, Formal Methods in System Design, 16, 23-58, 2000
- Simpson, A; Woodcock, J; Davies, J; *The mechanical verification of solid state interlocking geographic data*, Formal Methods Pacific '97 Proceedings of FMP '97. Springer-Verlag, Singapore; 1997; vii+320 pp. 223 - 242