Using Formal Methods in a Retrospective Safety Case

Lars-Henrik Eriksson
lhe@it.uu.se

Department of Information Technology*
Uppsala University
Box 337
SE-751 05 UPPSALA, Sweden

Abstract. Today the development of safety-critical systems is to a large extent guided by standards that make demands on both development process and system quality. Before the advent of these standards, development was typically done on a "best practise" basis which could differ much between application areas. Some safety-critical systems (e.g. railway interlockings) have a long technical and economical lifetime so that today we have many legacy safety-critical systems in operation which were developed according to practises that would be regarded as unacceptable today. Usually, such systems are allowed to continue operating by virtue of past performance. If there is doubt about the integrity of a legacy system, an alternative to replacement could be making a "retrospective" safety case demonstrating that the legacy system is indeed safe to use. Using as example a case taken from railway signalling, we will show how formal verification can be used in a retrospective safety case. In this application of formal methods several particular problems arise, such as uncertainty about the original requirements and the required safety level of the various system functions. We will discuss such problems and the approach taken to deal with them in the example case.

1 Introduction

On January 4th 2000 two trains collided head-on near the Norwegian village of Åsta, causing the death of 19 people and injuries to 30 more. A government commission [1] was formed to investigate the accident. It was determined that one of the trains had passed a signal which should have been at "danger". The commission could find nothing specifically wrong with the signalling system that could have caused the signal to incorrectly show a "clear" aspect, nor could it determine with certainty that it did show a "danger" aspect. The conclusion was that the cause of the accident was either a technical malfunction or a driver error, but that it was not possible to state which.

* The work presented herein was done while the author was employed by Industrilogik L4i AB, Box 3470, SE-103 69 STOCKHOLM, Sweden. I wish to thank my former colleagues for their involvement in this work.

A question raised by the commission was the reliability of existing signalling systems in view of today's requirements. Today there are a number of standards (e.g. CENELEC EN50126 [10], EN50128 [11], ENV50129 [12]) for the development of railway signalling systems. Previously, development of signalling systems was typically done on a "best practise" basis with relatively few development rules, to a large extent guided by the knowledge of experienced engineers. This is a situation not limited to railway signalling systems but is generally relevant to "legacy" safety-critical systems with long technical and economical lifetime. The continued operation of such systems today is typically motivated by operational experience without incidents, but is that always sufficient?

In the opinion of the Åsta commission a safety review of the existing railway signalling system was necessary. Primarily this concerned the particular signalling system (*interlocking*) in operation on the railway line where the collision occurred (NSB-87) but also – although less urgent – of a different signalling system (NSI-63) which was the most wide-spread system in use in Norway. The result was a decision by the National Norwegian Rail Administration (Jernbaneverket – JBV) to make a "retrospective" safety case for the NSB-87 based on the CENELEC standards for railway signalling systems.

A central part of a safety case according to the standard ENV50129 is the "Technical Safety Report" which presents the arguments that the technical design of the signalling system is sound. The Swedish consultancy Industrilogik L4i AB was employed to develop the Technical Safety Report for the NSB-87 safety case and on their suggestion formal verification was used as the basic means of analysing the interlocking design.

This paper describes the role of formal methods in carrying out the retrospective safety case. In section 2 the general structure and contents of the retrospective safety case is described while the the particular formal methods process used is described in section 3. Section 4 discusses the problems encountered and experiences made while actually carrying out the work. The conclusion section 5 ends the paper.

2 The Safety Case

2.1 General

According to ENV50129, a safety case is "the documented demonstration that the system complies with the specified safety requirements". Already this definition is troublesome for a "retrospective" safety case, in that in all likelihood there are no proper "specified" safety requirements in the sense of the Standard suitable for the existing system. In the case of NSB-87, the safety requirements had to be compiled from a variety of sources.

Of course, this should not be taken to mean that the NSB-87 system was developed without any safety requirements, only that – in retrospect – the safety requirements used were not as complete or well-defined as the Standard requires of new systems. Indeed, it is stated in the Standard itself, that it is intended to be used for the development of new systems, rather than assessment of old ones.

A safety case according to ENV50129 is divided into the following main parts:

System Definition A precise definition of, or reference to, the system to which the Safety Case refers, including version numbers and modification status of all requirements, design and application documentation.

Quality Management Report The evidence of Quality Management.

Safety Management Report The evidence of Safety Management.

Technical Safety Report The evidence of Functional and Technical Safety.

Related Safety Cases References to Safety Cases of any sub-systems or equipment on which the main Safety Case depends, if any.

Conclusion A summary of the evidence presented in the other parts of the Safety Case, and an argument that the system is adequately safe, subject to compliance with the specified application conditions.

The principal parts of the safety case will be discussed in turn.

2.2 System Definition

The NSB-87 is an interlocking type in use at 18 different installations. The installations are generally very similar, differing only in detail. One particular installation, that of Rena railway station, was selected to be the representative system to which the safety case refers. The differences between the installations were deemed to be so small that no separate safety cases were needed for the other installations beside Rena.

The structure of an NSB-87 installation is shown in figure 1.

The interlocking logic is encoded in a PLC (Programmable Logic Controller) program. When the NSB-87 was developed, the use of industry standard PLC's in railway interlocking was unusual and there was doubt both about the safety integrity of the PLC and how to demonstrate the safety of the program and of the PLC itself. Also, a single channel PLC was used with no duplication or redundancy either in software or hardware.

To avoid the safety issues with using a single channel standard PLC, a minimal interlocking based on traditional relay technology was added as a "filter" between the PLC and the trackside objects. Electrical circuits from the PLC to the trackside objects passes through relay contacts in the relay interlocking so that, in effect, 2 out of 2 voting between the PLC and the relay interlocking is done when executing possibly dangerous manoeuvres (clearing signals, reversing points...).

Should the PLC behave in an unsafe manner, the relay interlocking will thus "contain" the fault and protect trains operating in the area controlled by the interlocking. In effect, the burden of maintaining safety integrity is placed on the relay logic. Following this principle, in the formal analysis of the interlocking system the PLC was treated as a "hostile" part of the system which could exhibit arbitrary (i.e. unsafe) behaviour.

Since the manoeuvring system of the NSB-87 is also in the PLC, the relay interlocking receives input from the traffic control centre through the PLC. This function of the PLC is also not considered safety-critical as the purpose of a railway interlocking is reject commands that would be dangerous to carry out.

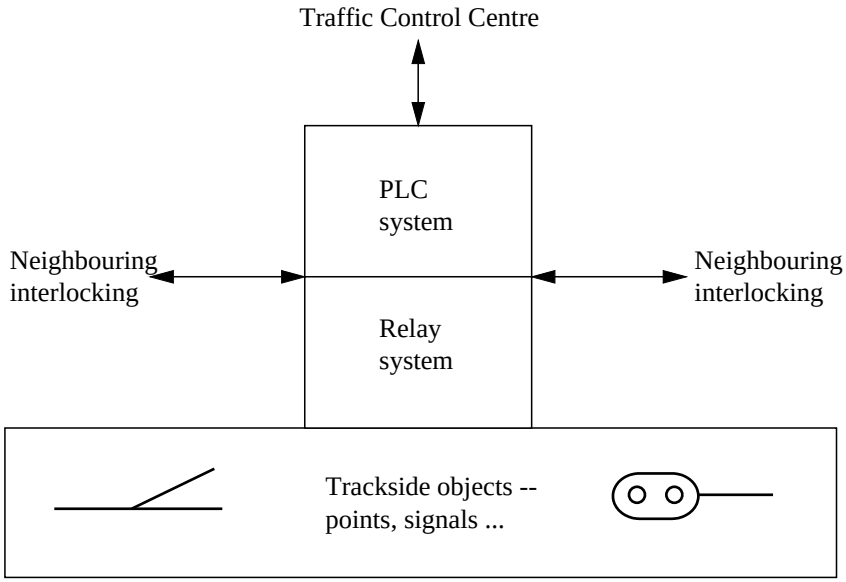

Fig. 1. Structure of a NSB-87 interlocking

2.3 Quality Management Report

At the time the NSB-87 was developed, quality management was done in an informal manner. This section of the safety case briefly describes the practises used. When possible reference is made to more recent documents based on these earlier practises.

2.4 Safety Management Report

At was the case with quality management, safety management was done in an informal manner. Some general principles about "fail-safe" operation were applied. Again, when possible reference was made to more recent documents based on earlier practises.

2.5 Technical Safety Report

This section forms the bulk of the safety case. Most of the contents of the Technical Safety report refers to the formal verification. As there was no "original" safety requirements specification in existence, one had to be compiled anew from various sources. Under those circumstances many minor deviations from the requirements specification could be expected. All deviations found (see section 4.3) were subject to a traditional risk analysis to determine whether or not they posed an acceptable or unacceptable risk. Thus safety properties at the system level were established using a combination of formal verification and risk analysis.

3 The Formal Verification Process

3.1 Basic Principles

As usual, we use the term "formal methods" to refer to the use of mathematically precise notation to describe requirements ("formal specification") and to use mathematically rigourous methods to determine whether the requirements are satisfied of a particular system ("formal verification").

The formal verification process used in the retrospective safety case is based on automated theorem proving in propositional (mathematical) logic. The requirements specification is represented by a number of logic formulae. Likewise, the system to be verified is modelled as a set of logic formulae. The (requirements) specification formulae determines the *permissible* behaviour of the system, while the (system) model formulae determines the *possible* behaviours of the system. The model of a correct system is said to be a *refinement* of the specification.

Clearly, if every possible behaviour is also a permissible behaviour, the system fulfils its requirements specification[1]. It is a result in mathematical logic that this relation is equivalent to the fact that a proof can be made of every specification formula using the model formulae as assumptions.

With modern theorem-proving algorithms such as Chaff [9] and fast computers, very large proofs can be attempted automatically in seconds. Should the proof fail, the algorithms will give a detailed counterexample (countermodel), describing a concrete situation where the requirements fail to hold. In this particular case, the descriptions of such situations were used as input to a traditional risk analysis.

The formal verification process is outlined in figure 2 and its various parts are described in the following sections. The process is described in more detail in [4][5][7].

Model Checking [3] is a different technique which is more commonly used for automatic formal verification and has been very successful in a number of industrial application areas. In the case of railway interlocking systems, though, there is some evidence that model checking works less well compared to theorem proving. [8] describes an application of model checking to formal verification of interlocking systems which was not feasible unless a number of optimisations were made to the model checker input. In our experience, no such optimisations are necessary when doing automatic verification by theorem proving.

We have not made any proper investigation as to why this would be the case, but we believe that the reason why theorem proving works well is that interlockings are typically designed so that most requirements hold even in unreachable states. In the few cases where this is not true, it is usually straightforward to characterise the relevant unreachable states by hand. The important function of a model checker of enumerating or characterising reachable states is thus not needed.

[1] As far as the behaviour can be correctly modelled in logic.

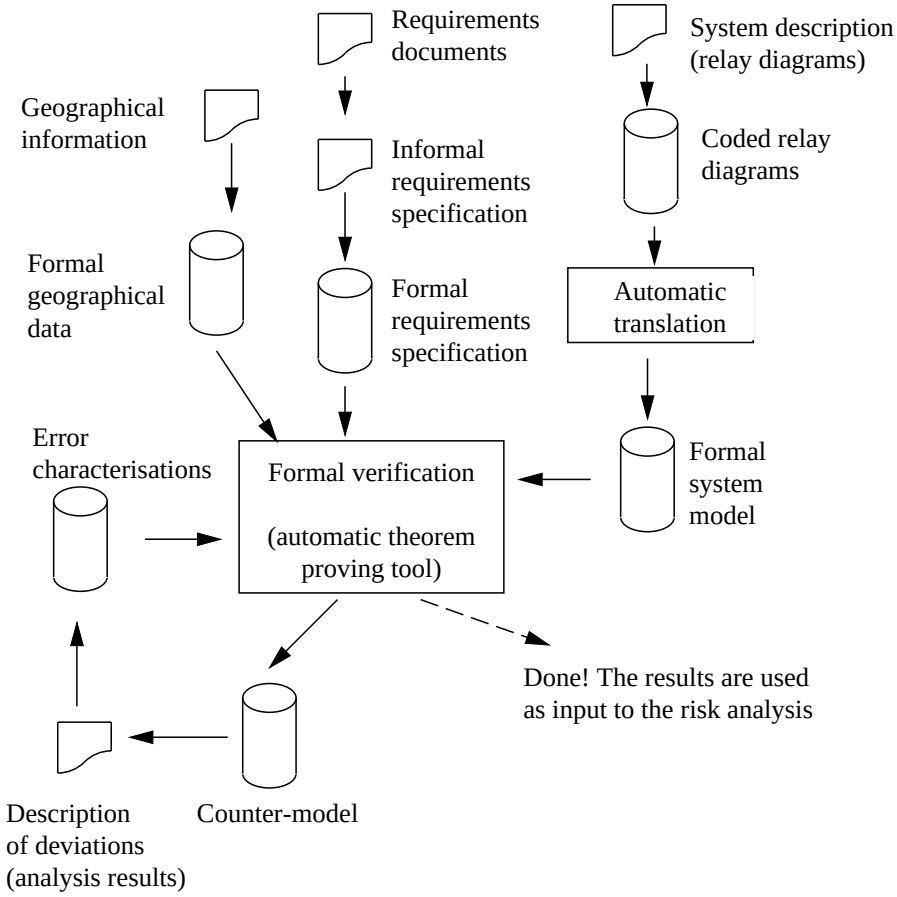

Fig. 2. The formal verification process

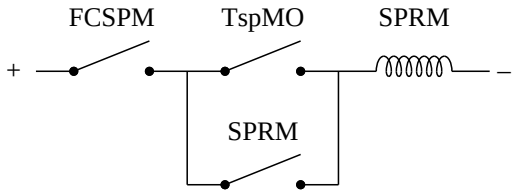

Fig. 3. Sample relay circuit

3.2 Specifications

Safety-critical systems in general (and railway interlocking systems in particular) are governed by a general set of requirements, while the systems have to be configured for any particular installation. It is desirable to have a general formal specification that can be used with any installation to avoid the work (and quality assurance issues!) of writing a separate specification for every installation.

This is done by writing the specification in a logic language (predicate logic) where it is possible to refer to information about a specific installation. In the railway context this information is called the "geographical data" and describes the geographical layout of the particular part of a railway where the interlocking operates (e.g. the location of tracks, signals, points etc.). By combining the general specification with installation-specific data an installation-specific specification is obtained which can be used as input to the formal verification process[2].

Example 1. A typical informal requirement:

> "A point is locked when any of the following conditions are met [...] the point is part of a locked train route [...]"

The corresponding logical formula is

```
ALL pt (SOME tr (routelocked(tr) & part_of(pt,tr)) ->
                 pointlocked(pt))
```

Here `pt` and `tr` are variables ranging over points and train routes. The subformula `part_of(pt,tr)` is a reference to the geographical data. Part of the geographical data will state precisely which points are parts of which train routes.

3.3 System Model

To make formal verification possible, a representation of the system in logic must be created. In the case of NSB-87, the system to be verified was the relay circuitry of the interlocking. This circuitry was documented by circuit diagrams which were translated into logic to obtain the system model. As the safety case must also demonstrate the safety of the system in the presence of faults, the translation method employed took into account the possibility of failures (single mode in this case) in the circuitry according to a specified set of possible failure modes.

Example 2. Figure 3 shows a sample circuit of the relay interlocking. `FCSPM` is an output contact of the PLC, while `TspMO` and `SPRM` are relays. The translation into logic is:

[2] The combining process includes a step where the predicate logic of the general specification is transformed into propositional logic so that it can be used with automatic propositional theorem provers.

```
SPRM <-> FCSPM & (TspMO # PRE SPRM) & ~SPRM_broken # SPRM_stuck
```

`#` stands for logical disjunction (or). `PRE SPRM` is an expression referring to a variable representing the state of the `SPRM` relay at the previous moment in time (see section 3.6). `SPRM_broken` and `SPRM_stuck` are variables modelling failures of the circuit. `SPRM_broken` is true if the relay does not pick up because the electric circuit has been broken or because of a relay fault. `SPRM_stuck` is true if the relay `SPRM` does not drop out because it is stuck or its contacts have welded[3].

The translation into logic was carried out automatically. A textual encoding of the relay circuitry was made by hand and stored in a file. A translation program read the file and automatically produced a set of logic formulae representing the behaviour of the relay system. This set of formulae was then used as input to the theorem prover.

Computer software such as PLC programs can also be translated into logic, but for this work the PLC software was not (initially, see section 4.3) considered safety-critical and therefore was not formally verified.

3.4 Counter-model Analysis

More often than not, a proof fails because the system does not quite follow the specification. The theorem prover algorithm will then automatically generate a "counter-model", a description of a situation where the system exhibits a behaviour which is not allowed by the specification. The counter-model itself is of limited use partly because of its excruciating detail but also because it does not in itself explain the reason for the incorrect behaviour. Most of the information described in the counter-model is coincidental to the particular requirement being violated. E.g. the counter-model will include state information of parts of the interlocking not related to the failing function.

Example 3. Suppose that the system did not satisfy the sample requirements formula of example 1. As part of the counter-model the theorem prover would give information about the particular point and train route involved. Also, the states of all relays – including failures – and other parts of the circuitry in the error situation would be given.

An important step of the process is the analysis of a counter-model to provide a characterisation of the situations where the system exhibits an erroneous behaviour. The analysis is presently carried out by hand and requires a substantial understanding of how the verified system works. The result of this analysis is used as output from the verification process to describe a particular problem with the verified system.

[3] Since interlocking relays are safety relays with forced contacts all relevant single failure modes can be represented using combinations of values of the `_stuck` and `_broken` variables for the various relays.

Example 4. In the case of the sample requirements formula of example 1, the conclusion could be e.g. that for point 4 and train route AL, the requirement would not be satisfied if a failure of relay `FrLok` occurred[4].

The characterisation is also used to find other situations where the same (or other) requirement is violated. The theorem proving attempt is repeated with the added assumption that the just characterised error situation can not occur. This will cause the theorem prover to disregard this particular error and it will either report that the proof succeeded (in which case all errors have been found), or it will generate a counter-model for a different error situation which can in its turn be analysed. By successively analysing counter-models and retrying the proof in this manner, all ways in which the system violates some requirement will eventually be found.

3.5 Risk Analysis

The risk analysis phase is not properly a part of the formal verification process, but it was an important step in producing the NSB-87 safety case so its relation to the formal verification will be briefly discussed.

In this formal verification process no attempt is made to calculate the frequencies or consequences of failures. Indeed, the logic is discrete and deterministic so the representation of failure modes simply states that the failure of a component of the verified system *may* happen. Also, there is of course always the possibility that the system does not function according to the specification even in the absence of component failures.

To determine whether deviations from the requirements have dangerous consequences and, if so, whether the probability of a deviation actually occurring is too high to be acceptable, a traditional risk analysis can follow the formal verification. Using known or estimated frequencies of component failures and behaviour patterns of the system environment (the railway in this case) the risk analysis can estimate the likelihood and consequences of incorrect system behaviour for each deviation found during the counter-model analysis.

In formal methods, a successful formal verification is normally used by itself as evidence of the safety of the system. When used in conjunction with risk analysis the formal verification need not be "successful" in the usual sense of the word. In contrast, it can serve the purpose of finding possibly dangerous situations which are subject to risk analysis. By using formal methods a much more complete set of findings can be obtained compared to what can be expected using traditional (manual) methods.

3.6 Handling Time

The behaviour at any given point in time of virtually every system found in practise is dependent of what has happened previously. In other words, the formal

[4] This is a hypothetical error and does not represent any actual error found in the analysis of the NSB-87 interlocking.

notation must be able to refer to different moments in time. There are many approaches to handling time. In the present work, the synchronous hypothesis [2] is adopted. This basically means that time can be seen as a sequence of separate infinitesimally small instances and that all actions of the system (e.g. change of output) take place from one time instance to the next. That is, the actions are taken to be instantaneous. This simplification (or abstraction) of real time is meaningful if we assume that the system will always react "quickly enough" to changes in its environment.

The requirements specification as well as the system model are transition relations, describing what states are permissible/possible in the present moment of time given the system state at the previous moment of time.

A drawback with this model of time is that it does not allow for specific time intervals, e.g. that two events are separated by a 5 seconds (say) interval. It turns out that in the present application, there is seldom a need for such specific time intervals. When they do occur they are simulated by assuming the existence of a clock which will signal when the given time interval has elapsed.

3.7 Tools

The process is supported by a software tool, GTO, developed by Industrilogik L4i AB. This tool can "plug in" any automatic propositional theorem prover, and various theorem provers such as SATO [14], zChaff [9] and Prover Technology CL-Tool (based on "Stålmarck's method" [13]) have been used with GTO.

4 Experiences from developing the safety case

4.1 The Requirements Specification

A prerequisite to any verification and validation effort is a requirements specification. In the case of the NSB-87 system no proper requirements specification was made when the system was developed[5]. Instead, the system was designed to generally have the same behaviour as previous systems and to follow the operational rules of the railway. Many of the actual requirements for an interlocking system were simply embodied in the design of previous signalling systems.

In order to carry out the safety case, a proper requirements specification had to be developed. To this end the relevant regulatory documents issued by JBV were studied. These documents were written after the development of the NSB-87 and it was known that they did not always correspond to earlier practise. Also, much information was missing from the regulations. Partly this was because "obvious" requirements were omitted, but also since the style of the documents in many cases was to state requirements by giving examples of correct behaviour, leaving the reader to extrapolate the requirements in situations not covered by the examples.

[5] This is a general phenomenon with older railway interlocking systems and not particular to the NSB-87.

To address the problems of the missing requirements as well as to save time, an existing formal requirements specification for Swedish railway signalling [4][6] was used as a basis for the requirements specification for NSB-87. Swedish and Norwegian practises are similar so the structure of the specification could be kept while details were changed in accordance with the JBV documents.

In order to validate the requirements specification, an informal version of the formal specification was made. This informal specification was essentially a direct translation of the logic formulae into plain text, preserving the structure and concepts of the formal specification. The informal requirements specification was turned over to JBV experts for inspection and approval and necessary modifications were made.

During the requirement engineering process, care was taken to maintain traceability from individual requirements formulae back to the documents or other sources motivating each formula.

As the formal verification work began, it turned out that in spite of the inspections and approval, some requirements were incorrect or not applicable, so the specification was further revised after consultation with JBV signalling experts. In our experience, it is typical that domain experts do not always understand the full ramifications of a particular requirement as they are used to follow rules that do not always work and to make ad-hoc deviations from them when circumstances make it necessary.

Creating the geographical data from a description of the layout of Rena station was straightforward.

4.2 The System Model

The relay circuitry was documented using a set of relay circuit diagrams. The manual coding of the circuit diagrams was very time-consuming and tedious. As the diagrams had been written using a CAD system, they were stored on CAD files and in principle the manual coding could have been replaced by an automatic interpretation of the representations of symbols and lines in the CAD files. However, the cost of developing such an interpretation program and uncertainty about how well it would work in practise was too great for a single project.

Occasionally, mistakes were found in that the circuit diagrams did not correctly represent the actual relay circuitry in the interlocking system. The mistakes were in most cases not detected until the verification process gave unreasonable results.

In most cases, an experienced signalling engineer would have spotted the mistakes, but as more time passes since the diagrams were made up the greater is the risk that engineers would fail to spot the mistakes, causing errors to be introduced into the system during modification or repair.

4.3 Verification

The formal verification revealed a large number of deviations from the requirements specification. This was caused partly by the fact that the requirements

were made up some 15 years after the system had been initially developed. What today would be considered incorrect behaviour would not necessarily have been seen as such at that time. In some cases, the deviations motivated a revision of the requirements specification. In other cases deviations could be motivated by special circumstances, while the requirements specification applied uniform requirements in all situations.

However, there were a number of cases where the system did not always behave as expected even by the standards applied at the time of development – particularly in the presence of faults.

These remaining cases were subject to risk analysis and in the end it was determined that the risks were acceptable. It was also found that – contrary to the intention when the system was first developed – some safety critical functionality was located only in the PLC system and not also provided by the relay "filter". Once this was realised, the risk analysis was used to determine if the situation was acceptable.

Nothing was found indicating incorrect behaviour that could have caused the accident at Åsta.

The manual analysis of the counter-model generated by the automatic theorem prover was very tedious and time-consuming. It also required a substantial understanding of how the circuits of the verified system actually worked.

A suitably different interlocking design could have simplified the verification and counter-model analysis. E.g. the design could have been structured in a similar way as the specification and with more redundancies in the circuitry – particularly in "don't care" situations. When making new designs it is important to consider how to "design for (formal) verifiability".

5 Conclusions

In summary, the formal verification of the NSB-87 interlocking was quite successful. In spite of some difficulties, the project produced the desired output and was completed within schedule.

Determining the safety requirements was not straightforward – particularly not considering the amount of detail and precision required of a formal specification. The original specifications from the 1980's were not documented in a complete or uniform way and it was also not clear to what extent they are relevant today. On the other hand, if the requirements of today would have been used without changes, it would be certain that in many cases they would not be fulfilled by the system.

For these reasons it was necessary to accept that a large number of deviations from the requirements were found during formal verification. To determine whether or not the deviations were acceptable, they were subject to a traditional risk analysis. In a sense, the main function of the formal verification was to provide input to the risk analysis rather than itself establish the safety of the system. This was the major difference between the use of formal methods in the

retrospective safety case compared to using formal methods when developing a new system.

The use of formal verification gave a considerably better verification coverage of the system than would have been expected by a traditional analysis based on manual inspection of circuit diagrams. This was particularly true when the effect of faults was considered as it is difficult for a person to fully consider and grasp the effects of all possible faults on a system.

The manual steps in the formal verification process – the coding of circuit diagrams and the analysis of counter-models were very time-consuming. The coding could in principle have been automated, and in any case modern systems are typically based on computer (e.g. PLC) technology where the program source code is machine-readable from the start and no manual coding is necessary.

The counter-model analysis is a different matter, as it requires insight into the workings of the system to determine what part of the system design contributes to a deviation from a particular requirement. Nevertheless, it is expected that much of this analysis could also be automated – however it is basically a research issue to determine how this could be done.

The involved staff of the Norwegian National Rail Administration were generally quite impressed with the results that could be achieved using formal methods. The same methodology was again used successfully for another retrospective safety review of the different interlocking system NSI-63 (also a recommendation of the Åsta commission). Additionally, the NSB-87 safety case indirectly led to new industrial adoption of formal methods as a component in the development of safety cases for new signalling systems.

References

1. Åsta-ulykken 4. januar 2000 – Hovedrapport, The Norwegian Ministry of Justice and the Police (2000)
2. Benveniste, A. and Berry, G.: The Synchronous Approach to Reactive and Real-Time Systems, Proceedings of the IEEE, Vol. 79 No. 9, pp. 1270–1282 (1991).
3. Clarke, M.E., Grumberg, O. and Peled, D.A.: Model Checking, MIT Press (1999).
4. Eriksson, L-H.: Formalising Railway Interlocking Requirements, Technical report 1997:3, Swedish National Rail Administration (1997)
5. Eriksson, L-H.: Formal Verification of Railway Interlockings, Technical report 1997:4, Swedish National Rail Administration (1997)
6. Eriksson, L-H.: Specifying Railway Interlocking Requirements for Practical Use, In Schoitsch, E. (ed.): Proceedings of the 15th International Conference on Computer Safety, Reliability and Security (SAFECOMP'96), Springer-Verlag (1996)
7. Eriksson, L-H. and Johansson, K.: Using formal methods for quality assurance of interlocking systems, In Mellit, B. et.al. (eds.): Computers in Railways IV, Computational Mechanics publications (1998).
8. Huber, M. and King, S.: Towards an Integrated Model Checker for Railway Signalling Data, In Eriksson, L-H. and Lindsay, P.A. (eds.): FME'2002: Formal Methods – Getting IT Right, pp. 204–223, Lecture Notes in Computer Science 2391, Springer-Verlag (2002).

9. Moskewicz, M., Madigan, C., Zhao, Y., Zhang, L. and Malik, S.: Chaff: Engineering an Efficient SAT Solver, Proceedings of the 38th ACM/IEEE Design Automation Conference (DAC'01), pp. 530–535, ACM/IEEE (2001)
10. Railway Applications: The Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS), European standard EN50126, CENELEC, Brussels (1999)
11. Railway Applications – Communication, signalling and processing systems – Software for railway control and protection systems, European standard EN50128, CENELEC, Brussels (2001)
12. Railway Applications – Safety related electronic systems for signalling, European standard ENV50129, CENELEC, Brussels (1998)
13. Sheeran, M. and Stålmarck, G.: A Tutorial on Stålmarck's Proof Procedure for Propositional Logic, In Gopalakrishnan, G. and Windley, P. (eds.): Proc. 2nd Intl. Conf. on Formal Methods in Computer-Aided Design, FMCAD'98, pp. 82–99, Lecture Notes in Computer Science 1522, Springer-Verlag (1998)
14. Zhang, H.: SATO: An Efficient Propositional Prover, In McCune (ed.): Proc. 14th International Conference on Automated Deduction (CADE-14), Lecture Notes in Computer Science, Springer-Verlag (1997)