Automatic train controller safety simulation

R. A. V. Gimenes, J. R. de Almeida Jr. & T. R. Nogueira
Safety Analysis Group – Escola Politécnica da Universidade de São Paulo, Brazil

Abstract

Nowadays, we are observing an increased demand for better and safer mass transport systems. The supervision and control of these systems is made through an architecture known as Automatic Train Controller (ATC). The use of processors in ATC provides new challenges in a safety analysis. A typical railway system has a Speed-Distance Profile Generator that determines the maximum allowed speed in each track circuit. Dangerous situations are verified through the relative positioning and speeds between trains, switching machine positioning and other restrictions from operational commands generated by the ATC. Independent Safety Auditors should consider the failure modes of hardware and software in use. The safety analysis should consider alternative techniques to complete the complex task of evaluating how safe is the use of the processor and its associated software. Therefore, the use of simulation can improve and increase safety analysis, searching for fault states that could not be found in a static analysis. The main goal of this paper is to describe the development of a tool that simulates the behaviour of trains' movement in a subway system, with boolean expressions. The set of boolean expressions coordinates all the movements in a subway line and the simulation provides the possibility to find out lack of safety, considering different combinations in those boolean expressions. Another important goal is to simulate equipment faults in order to investigate problems not visible in a static analysis or even in a practical field test. Preliminary results have shown that the use of a simulator to execute boolean expressions offers a great variety of tests, allowing the detection of unsafe situations, complementing software tests validation in a final release. Through simulation, it is possible to observe the behaviour of simulated objects in specific internal points which improves the completeness in safety analysis.

Keywords: railway system, environment, safety, simulation, automation, modelling, boolean expression, microprocessor.

WIT Transactions on The Built Environment, Vol 88, © 2006 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR060141

1 Introduction

Nowadays, we are observing an increased demand for better, safer and more available mass transport systems, inclusive in the railway systems. The supervision and control of these systems is made through an architecture known as Automatic Train Controller (ATC).

The main ATC System goal is to automate the train circulation and to obey the operational commands with all safety requirements. The rails that compose the track are divided in track circuits. So, one of the main functions of the ATC is to send speed codes to the track circuits and, at the same time, to verify if each track circuit is occupied by a train.

Other components of a railway system are the gates and the switching machines. The Gates are important to control the entrance in an Interlocking Location, which contain switching machines that coordinate the train crossing through different track circuits.

The Track Circuit equipment informs the ATC if there is a train within it. The ATC also receives information from the switching machines position and others ATCs. Those information shows to ATC snapshots from the railway, providing capacity to determine changes about the maximum speed allowed for all Track Circuits, switching machines position and gates states.

The railway has a Speed-Distance Profile that determines the maximum allowed speed in each track circuit. The speed control must consider the routes desired for each train, the moment where each route was requested and it has to verify if there are not unsafe situations in such routes. Unsafe situations are verified through the positions and speeds of other trains, switching machines position and others restrictions from possible operational commands.

2 Uses of Microprocessors in railway systems

Railway signaling exists to guaranty basic needs of safety. Its development, otherwise originates new facilities, considering the increased, performed by higher speeds and new types of control [1].

Traditional railway systems are supervised and controlled by a set of electromechanical relays that are very expensive and hard to maintain. On the other hand, they have a high reliability level. By the time, processors are replacing the electromechanical relays, offering great flexibility to maintain and upgrade the system, with higher availability.

To guarantee the correct train flow control, it is necessary to define a set of state variables logically controlled by boolean expressions. Therefore, each physic element has associated a group of boolean expressions. The origin of those boolean expressions comes from relay systems. An interesting and cheaper solution consists in import those boolean expressions and translate them to a computer code. A typical boolean expression is: A = B * C + (.D + A);

The boolean expressions contains vital variables that use the “logical AND operator” and “logical OR operator”. The “logical NOT (symbolized by a dot)” inverts the value of a variable, but it cannot invert a whole expression.

In contrast with relays, which are designed to have fault tolerance, all boolean expressions in a processor are stored in memory positions, without a direct physic fault tolerance mode.

The use of processors generates new challenges in a safety analysis. Independent Safety Auditors should consider the failure mode in hardware and software in use. The safety analysis should consider alternative techniques to complete the complex task of evaluate how safe is the use of processor and its associated software [2]. Therefore, the use of simulation can improve and increase a safety analysis, searching for fault states that could not be found in a static analysis [3].

The main goal of this paper is to describe the development of a tool that simulates the behavior of trains' movement in a subway system, through the use boolean expressions. The set of boolean expressions coordinate all the movements and the simulation provides the possibility to find out lacks of safety, considering different situations and combinations in those boolean expressions.
Another important goal is to simulate equipment faults in controlled conditions, in order to investigate problems not visible in a static analysis or even in a practical field test.

3 ATC environment simulation

Simulation is concerned with building a computer model of equipment or assets under study and to evaluate different scenarios, that can even explores unrealistic situations, too expensive or too harder to implement in real word [4].

This way, the use of simulation has been using to evaluate a real Railway System. The main core of this study is an ATC Environment simulation. The whole project uses the oriented-object modeling concept.

This kind of modeling provides a natural evolution in the software modules complexity. The modeling uses objects as trains, track circuits, switching machines, the ATC and others that can be interesting to use at the simulation (Figure 1).

An important ATC module is composed by one Translator (including a lexer and a parser) capable to process the boolean expression language. This first Translator is specific to the boolean expression language, but it is possible to substitute it without changing the environment simulation.

The Environment Simulator module is the object responsible to manage all other objects. It defines the time base and the state machine changes. The use of a time base represents the ATC point of view, as a discrete system, transferring, this way, snapshots from the other objects. When the ATC receives the snapshots, it has information enough to feed the boolean expressions and then update its outputs.

The Environment Simulator module has the mission to be a layer, linking the ATC and all objects. It has to control the ATC inputs and outputs, assuring a realistic approach in simulation of a railway system.

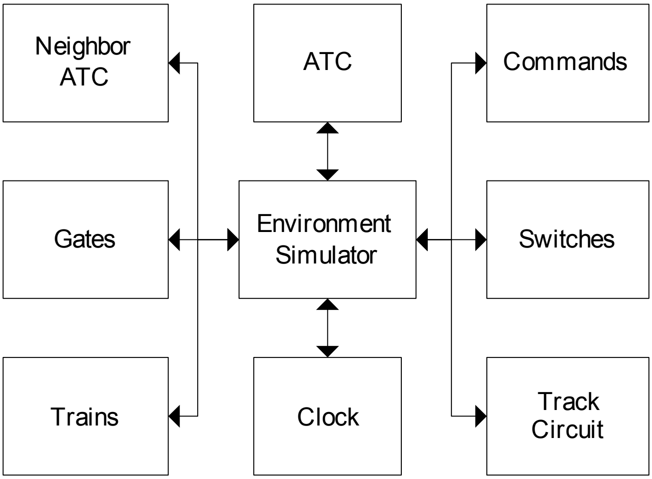


Figure 1: Example of an ATC environment modeling.

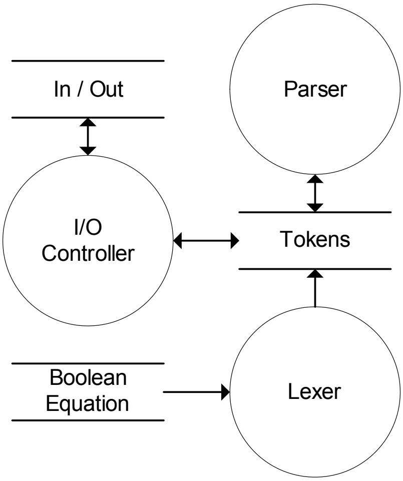


Figure 2: ATC module.

The ATC contains the Lexer, the Parser and the I/O Controller (Figure 2). The I/O controller depends on how much knowledge the designer has from a real ATC. If the ATC is a white box, the simulation will represent the I/O Controller very well. Therefore, as much detail from the real system is available as much accurate is the simulation.

After the definition of each object, using information from real systems, the Environment Simulator can overstress the ATC module. The overstress simulation offers a range of situations harder to be detected in a real railway system [5].

The first step is to execute all boolean expressions considering an initial state. The Environment Simulator controls the initial state and feed the objects with those states. Then, the simulation continues through all known relevant objects, using information about their situation. It is important to notice that the ATC can execute the boolean expressions with just the initial values of each boolean variable.

This way, even with a simple Environment Simulator, it is possible to force situations and then observe the results in a specific set of boolean expressions. That modeling offers a great range of very flexible simulation choices.

3.1 Implementation

The implementation has been developing in a Microsoft Visual Studio .NET 2003 framework. The language used is C# and all classes consider future expansion, which means, the modules can receive different configurations of railway systems domains or else different grammar rules to boolean equations. The Figure 3 shows the some classes like Parser and Lexer.

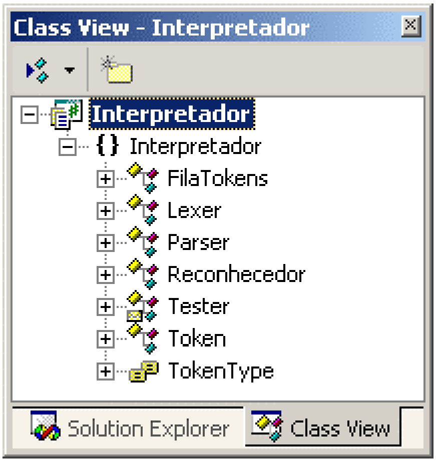
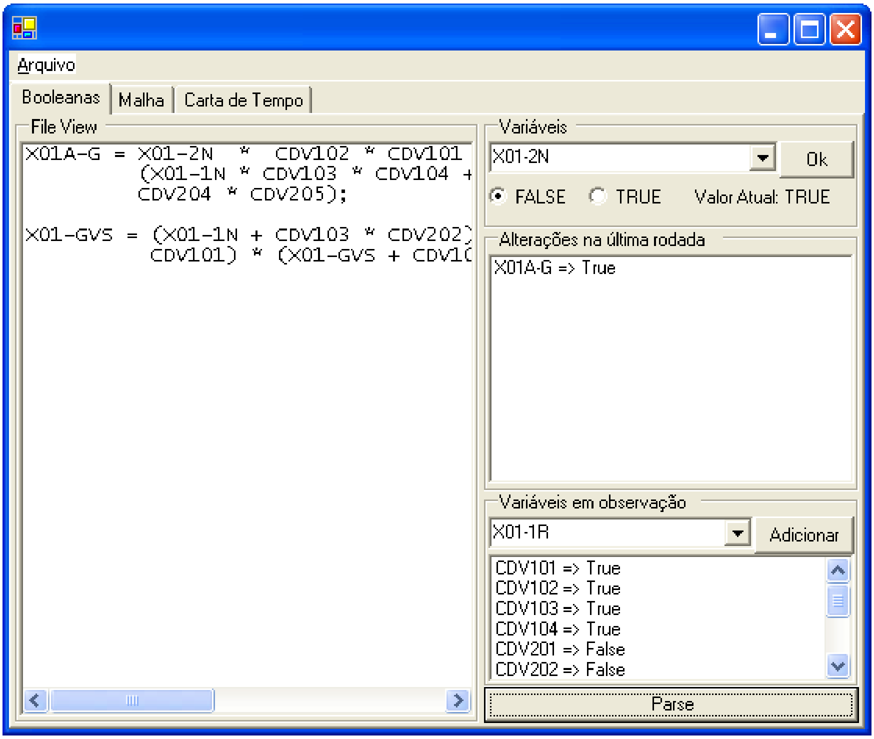


Figure 3: Example of classes used in this project.

Our implementation has an initial interface (Figure 4) which is possible to monitory all project variables. It is very important to know the value at boolean variables in each simulation step, because the Environment Simulator need those information to define the state changes of all element in simulation. The parser checks the grammar and then calculates the equations considering the entrances it receives from the Environment Simulator.

This interface offer a possibility to configure in a simple way, any variable that could concern in a simulation (one parser step) and discover which equations has updated their values. The environment uses our native language (Portuguese), but it is flexible to translate to any language.



```
X01A-G = X01-2N  *  CDV102 * CDV101
         (X01-1N * CDV103 * CDV104 +
         CDV204 * CDV205);

X01-GVS = (X01-1N + CDV103 * CDV202
          CDV101) * (X01-GVS + CDV1
```

Variáveis: X01-2N

● FALSE ○ TRUE Valor Atual: TRUE

Alterações na última rodada:

X01A-G => True

Variáveis em observação: X01-1R

CDV101 => True
CDV102 => True
CDV103 => True
CDV104 => True
CDV201 => False
CDV202 => False

Figure 4: The parser user interface.

4 Conclusion

Preliminary results have shown that the use of a simulator to execute boolean expressions offers a great variety of tests, allowing the detection of unsafe situations, complementing software tests validation in a final release.

Through simulation, it is possible to observe the behaviour of simulated objects in specific internal points. As example, such internal points can expose how much cycles the ATC executes the boolean expressions to change the value of a boolean variable.

The ATC Simulation improves a typical safety analysis, which has to deal with the completeness problem [6]. The simulation makes possible to analyse dynamic states and then offer to a safety analysis group a powerful tool to determine the reliability of a computer based railway system.

The use of computer-based architecture offers a huge range of project options, instead of relays architecture. The computer flexibility can imply in unnecessary redundancies, a volume of expressions bigger than it is really necessary and feedbacks between expressions that can result in unexpected states.

In contrast with relay systems, the execution of all boolean expressions is serial. Then, the order that the expressions are written can influences the amount of necessary cycles of update an output value. Such characteristic can hide hazard situations in a great amount of boolean expressions of a real ATC System.

Despite this environment simulator being initially focused in boolean expression, its architecture allows that the simulator stress others modules. It is possible to include more details in objects like trains, including its own controller's objects. Then, in a complete simulation, it is possible to detect how specific faults in one module can alter values in another.

Such kind of simulations has the main goal to improve aspects of safety and other RAM elements (Reliability, Availability and Maintainability).

References

[1] Cunliffe, J.P. *A survey of railway signalling and control* Proceedings of the IEEE, Publication Date: April 1968, Volume: 56 , Issue: 4On page(s): 653 – 674, ISSN: 0018-9219

[2] Matsumo, M. *The revolution of train control system in Japan*, Autonomous Decentralized Systems, 2005. ISADS 2005. Proceedings, On page(s): 599 – 606

[3] Li, J.J. *Prioritize Code for Testing to Improve Code Coverage of Complex Software* Software Reliability Engineering, 2005. ISSRE 2005. 16th IEEE International Symposium on 2005, On page(s): 75- 84, ISSN: 1071-9458, ISBN: 0-7695-2482-6

[4] Yazdi, H. Roberts, C. Fararooy, S. *Intelligent condition monitoring of railway signalling equipment using simulation* Birmingham Univ. Condition Monitoring for Rail Transport Systems, IEE Seminar on 1998, London

[5] M.J. Martin, J.D. *Synchronisation of cyclic coset codes for railway track circuit data* Collins, Sch. of Electron. & Electr. Eng., Bath Univ., UK; Railroad Conference, 1996., Proceedings of the 1996 ASME/IEEE Joint On page(s): 137 – 141, 1996, Oakbrook, IL

[6] Del Frate, F., Garg P., Mathur, A., Pasquini, A., *On the correlation between code coverage and software reliability* Proc. 4th Int'l Symposium on Software Reliability Engineering (ISSRE), pp. 124-132, 1995.