COVER PAGE

FORMAL METHODS IN THE EVALUATION OF A SAFE DRIVER-MACHINE INTERFACE

Istvan Majzik[1],
Andrea Bondavalli[2],
Štépan Klapka[3],
Tatiana Kozlova Madsen[4],
Danilo Iovino[5]

[1]*Budapest University of Technology and Economics, Budapest, Hungary*
E-mail: majzik@mit.bme.hu

[2]*Università degli Studi di Firenze, Firenze, Italy*
E-mail: bondavalli@unifi.it

[3]*AŽD Praha s.r.o. , Research and Development, Prague, Czech Republic*
E-mail: Klapka.Stepan@azd.cz

[4]*Aalborg University, Aalborg, Denmark*
E-mail: tatiana@kom.aau.dk

[5]*Ansaldo Segnalamento Ferroviario, Torino, Italy*
E-mail: Iovino.Danilo@asf.ansaldo.it

Short abstract: Driver Machine Interface (DMI) is a slave unit of the train onboard computer (EVC) in the ERTMS-ATC system. The SAFEDMI project aimed at the development of a DMI which fulfills the requirements of Safety Integrity Level 2 according to the CENELEC development standards. Formal methods were successfully applied in the quantitative evaluation of the DMI. This paper presents an overview of these methods applied for the evaluation of the architecture, the wireless communication protocol, and the detection codes and residual errors.

Desired type of session: Poster session

Intended scope: Methods and Tools for Modelling, Validation / Verification and Tests (Evaluation of RAMS-Parameter, System related and functional Validation, Limits of formal Methods)

Contact author:
Istvan Majzik
Budapest University of Technology and Economics, Dept. of Measurement and Information Systems
Magyar Tudosok krt. 2., H-1117 Budapest, Hungary
Phone: +36-1-463-3598
Fax: +36-1-463-2667
E-mail: majzik@mit.bme.hu

FORMAL METHODS IN THE EVALUATION OF A SAFE DRIVER-MACHINE INTERFACE

I. Majzik[1], A. Bondavalli[2], S. Klapka[3], T. Kozlova Madsen[4], D. Iovino[5]
[1]*Budapest University of Technology and Economics, Budapest, Hungary*
E-mail: majzik@mit.bme.hu
[2]*Università degli Studi di Firenze, Firenze, Italy*
E-mail: bondavalli@unifi.it
[3]*AŽD Praha s.r.o. , Research and Development, Prague, Czech Republic*
E-mail: Klapka.Stepan@azd.cz
[4]*Aalborg University, Aalborg, Denmark*
E-mail: tatiana@kom.aau.dk
[5]*Ansaldo Segnalamento Ferroviario, Torino, Italy*
E-mail: Iovino.Danilo@asf.ansaldo.it

Abstract: Driver Machine Interface (DMI) is a slave unit of the train onboard computer (EVC) in the ERTMS-ATC system. The SAFEDMI project aimed at the development of a DMI which fulfills the requirements of Safety Integrity Level 2 according to the CENELEC development standards. Formal methods were successfully applied in the quantitative evaluation of the DMI. This paper presents an overview of these methods applied for the evaluation of the architecture, the wireless communication protocol, and the detection codes and residual errors.
Keywords: Safety-critical systems, stochastic modeling, quantitative evaluation

1. INTRODUCTION

The European Rail Traffic Management System (ERTMS) programme provides unique functional and non-functional requirements for the onboard Automatic Train Control (ATC) systems. Besides the European Vital Computer (EVC) that implements the main control functions of the train, the ERTMS architecture includes a Driver Machine Interface (DMI) as well. The DMI is responsible for the display of the information originating from the EVC (that supports the train driving) and the acquisition of driver's commands. Accordingly, its main hardware components are a display and a keyboard. Its functions and ergonomic requirements are defined in CENELEC requirement documents (CENELEC, 2005).

The increasing level of train traffic is demanding an increasing level of safety in the ATC systems. Although the DMI is required to operate in a quite critical context, formerly it was considered as a non-safety-critical slave unit of the EVC. In the last few years, however, many railway operators started requiring from their providers DMIs which reach at least Safety Integrity Level 2 (SIL 2) according to CENELEC development standards (EN 50129, 2000, EN 50128, 2001).

The objective of the *Safe Driver Machine Interface for ERTMS Automatic Train Control* (SAFEDMI, 2006) project was to design and develop a DMI system that distinguishes itself from other train borne DMIs by the following properties:

- It is able to *satisfy SIL2* with all the related implications. The DMI is capable of providing in a safe way visualisation, driver input data acquisition (processing the driver's commands) and data transfer between itself and the EVC.
- It is able to *provide safe wireless communication* for configuration, software uploading and diagnostic purposes. A traditional approach usually required mechanical operations to extract the DMI and access the required information, while the emerging wireless communication technologies allow very quick DMI maintenance (configuration and upgrade).

The development of the DMI was determined by the related CENELEC standards (cited above) and included the following main steps: (1) investigation of the railway application

scenarios to specify the detailed requirements and risks to be considered, (2) design of the architecture and the related fault tolerance mechanisms, (3) development of safe and non-safe protocols for wireless communication, (4) quantitative evaluation of the safety and availability properties, (5) development of a testing framework aiming at the removal of design faults, and finally (6) implementing a prototype that demonstrates the operation of the DMI and the feasibility of the technical solutions.

This paper focuses on those design and evaluation steps that were supported by the use of formal techniques. Chapter 2 presents an overview of the DMI architecture and the applied modelling methods. Chapter 3, 4 and 5 focus on the quantitative evaluation tasks that formed the core part of the DMI verification. Chapter 6 concludes the paper.

2. OVERVIEW OF THE DMI ARCHITECTURE

From the point of view of the external interfaces, the DMI (as a slave unit) is connected the EVC via commercial field bus links (such as Profibus, Fig. 1) using safe protocols. If a hazardous error is detected then a specific exclusion logic cuts off the field bus connections towards the on-board system and switches off the display to put the DMI in a safe state. In this way the DMI is isolated and cannot propagate errors. If the EVC detects that the DMI is not available it triggers and emergency mechanism (braking the train) and returns to normal operation only if the DMI will become available again. A spare DMI component deployed onboard can be turned on to replace a failed DMI in order to increase system level availability.

The internal architecture of the DMI was designed to address several demands. To reduce the costs, the number of hardware components used in the DMI was to be kept at minimum and generic (off-the-shelf) components were used. Reliability, availability and safety properties were associated to the architectural design and the related mechanisms. The applied techniques for error detection and error recovery were as much as possible independent from the specific components.

The design adopted the *single electronic structure based on reactive fail-safety* principle that assures safe operation by proper detection and negation of hazardous faults that occur in the DMI.

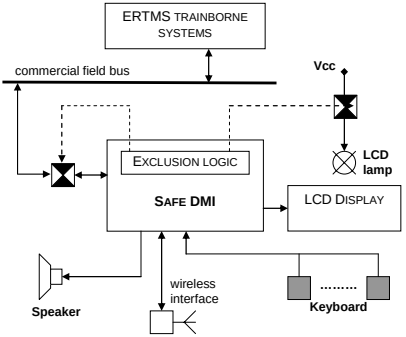

Fig. 1. The external interfaces of the DMI

According to these demands and principles, the internal DMI hardware architecture consists of single CPU that is connected through its bus to the following peripherals: RAM, Flash ROM, LCD display (with its controller and LCD lamp), keyboard unit, EVC interface, wireless communication interface, audio device, log device, thermometer, and cabin identifier.

The wireless communication interface is implemented using a separate hardware unit called Bridge Device (BD) that acts as a bridge between the open wireless network (used to communicate with a remote Maintenance Centre) and a closed wired network (towards the DMI). Note that communication in open networks shall satisfy specific safety and security requirements (EN 50159-2, 2001), which required the development of a safe protocol stack.

The internal software architecture of the DMI consists of modules that implement the *ERTMS related functionality* (data acquisition, processing and visualization functions considering the ergonomic requirements as well), the *safe protocol stack* required to perform the wireless configuration and upgrade, and a set of *support functions* that implement mode changes, error detection and recovery mechanisms. As these latter functions and mechanism are crucial from the point of view of the evaluation, their role is presented in more detail.

The main component of the *safe protocol stack* is a safety layer that manages the interactions between off-the-shelf protocols and services as well as adds additional functions enabling a required integrity level of transferred data. This layer is split into two sub-layers: Low

Safety Layer (LowSL) located at the BD and the Maintenance Centre (MC) and High Safety Layer (HighSL) located at the DMI and the MC. This splitting is motivated by a classification of communication safety requirements into two levels: (1) message transmission over open wireless communication system; (2). communication at the application level. The LowSL is responsible for providing a secure connection between BD and MC. The purpose of HighSL is to guarantee a safe, secure and dependable end-to-end connection.

The *operational modes* of the DMI include Startup, Normal, Configuration and Safe modes with the obvious roles (here Safe mode means the isolation of the DMI from its environment). Moreover, a Suspect mode is implemented that is entered whenever an error is detected. In this transitory mode the number of subsequent errors in a given time interval is counted. If a predefined limit is reached then the Safe mode is forced, otherwise the Startup mode is entered (attempting to restart) in which detailed self-tests are executed. If a start-up test detects an error then the Suspect mode is entered again to take into account a subsequent error detection. If no error is detected then the Normal mode is entered, resetting the error counter and the timer.

The efficient *fault detection* has a key role according to the reactive fail safety principle. Note that detected faults are handled in the Suspect mode by the restart mechanism and by the fail-stop behaviour if necessary. The applied fault detection techniques target random transient and permanent faults (systematic faults are to be handled in the development process). In the Startup mode *permanent faults* are detected by rigorous test procedures relying also on the assistance of the driver. In Normal mode *latent permanent faults* are detected by periodic testing. Active *permanent faults* and *transient faults* in the hardware components are detected on the basis of their effects on software execution. Considering the recommendations of the safety standards, the following on-line (run-time) software fault detection techniques were selected:

- *Data acceptance/credibility checks* (including detection codes) were applied on intermediate and input/output data in communication and configuration modules.
- *Control flow monitoring* was applied in control related functions in which no data acceptance criteria could be given, but the related hazards were sensitive to the missing or incomplete execution (e.g., mode changes, global monitoring, tests checks).
- *Multiple computation and comparison* were adopted in cases when the internal processing was sensitive to data related transient faults in the CPU but no proper data acceptance criteria could be established (e.g., in the case of visualisation).

The software architecture of the DMI was designed using UML as modelling language (applying Eclipse based UML tools). All components of the DMI and the behaviour of the critical mechanisms were modelled. The documentation process followed the Open Distributed Processing recommendations (ODP, 1999). These models were the basis not only for the further refinement and implementation phases, but also for the *quantitative evaluation* as the core part of the verification life cycle phase.

On the basis of the SAFEDMI and standard related requirements, the quantitative evaluation of the DMI consisted of the following tasks that were effectively supported by the application of formal methods:

- The *evaluation of the DMI architecture* addressed the hazardous failure rate (quantified according to SIL2), reliability and availability requirements. Here the main challenge was the proper evaluation of the efficiency of the applied error detection and error handling mechanisms.
- The *evaluation of the wireless communication protocol* addressed performance requirements (upload and download times), the optimization of the connection management, and the defences specified in (EN 50159-2, 2001).
- The *evaluation of the applied detection codes* addressed the requirements regarding the detection quality of codes in safety related communication (EVC-DMI and wireless) and integrity checks.

These tasks are presented in the following chapters in detail.

3. EVALUATION OF THE DMI ARCHITECTURE

According to the standards, the quantitative evaluation of random failure integrity shall be carried out by means of probabilistic calculations. To support this, a *dependability model of the DMI architecture and mechanisms* was constructed. It is a mathematically precise

model representing the fault activation in system components, the error propagation among them according to the given architecture, and the applied error detection, error handling and recovery mechanisms.

The formalism adopted for the construction of the dependability model was *Stochastic Activity Networks* (SAN, Sanders and Meyer, 1991). It can be considered as an extension of Stochastic Petri Nets with input gates (specifying firing conditions), output gates (specifying complex actions belonging to firing), transitions with general firing time distributions, and reward functions.

Two levels of modelling and analysis were distinguished. The lower level analysis investigated the safety and availability properties of the DMI itself. The upper level modelling investigated the effect of DMI failures on train missions considering spare DMIs as well.

3.1 Evaluation of DMI properties

The state-based dependability model of the DMI was constructed in a modular way, by assigning sub-models (SAN subnets) to the architectural components (resources, tasks) and composing them according to their relations. This composition was effectively supported by the *Join* and *Repeat* composition operators offered by the SAN formalism (Rojas, 1996). The former allowed to connect subnets via shared places, while the latter supported multiple "instantiation" of a subnet. The construction of the subnets followed the principles described in (Serafini at al, 2006) to model the on-line error detection techniques that determine the safety and reliability of the DMI. Accordingly, the following sub-models were defined:

Resource sub-models: Each hardware resource is assigned a subnet that represents its fault activation process, the potential effects of its faults on the software (tasks), and the startup and periodic testing activities. Fig. 2 presents the structure of a generic resource subnet template. Transition *faultOccurrence* models the occurrence of permanent and transient faults. These faults may be relevant to different tasks (*Task$_i$_Errors*) that use this resource. The propagation probability is characterized by the ratio of resource usage (e.g., memory areas of tasks). The occurrences of permanent faults are registered separately to model how periodic tests (with given rate and coverage parameters) can detect these permanent faults. A detected error triggers a change in the operational mode (*HWTrigger*), while undetected errors are collected separately. Periodic tests and startup tests can be activated only in the given operational mode (this is not presented in Fig. 2).

Note that residual errors (resulting from configuration and software upload) are activated in a similar way like faults in the ROM resource, however, these cannot be detected by startup tests and periodic tests.

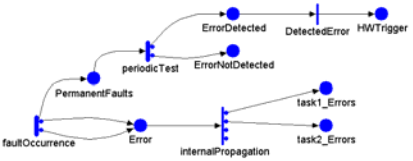

Fig. 2. The sub-model of a resource

Propagation sub-model from a passive resource to a task (Fig. 3): An error in a resource that is relevant to a task may be activated with a given rate (average rate of using the given resource by a given task). The activation has three cases: (1) the error is activated and its effect remains in the system (permanent faults are included here), (2) the error is activated but without remaining effect, (3) the error is overwritten. An activated error results in a task failure. In the case of an active resource, a fourth case may occur: the activation of an error results in another error as well. The on-line data acceptance and credibility checks (related to communication resources) are modelled as separate transitions in the propagation flow that can prevent activation with a given probability (i.e., the coverage of these checks).

Fig. 3. The sub-model of an error propagation from a passive resource

Task sub-models (Fig. 4): The task level sub-model of a task represents the on-line error detection techniques that are used in the case of that task. The activated errors (*Failure* from a propagation sub-model) can be detected with a delay and probability (coverage) characterising the given technique. Here duplicated execution, control flow checking and acceptance/credibility

checking are considered. Detected failures result in an operational mode change, while undetected ones contribute to hazards.

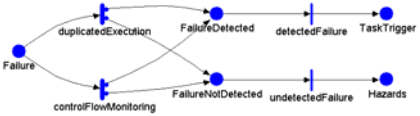

Fig. 4. The sub-model of a task

Operational mode sub-model: Operational mode changes are triggered by external activities (configuration requests characterised by a given rate, startup tests characterised by a duration) or internal events (errors detected by periodic tests or on-line error detection). The operational modes and the retry recovery implemented in the Suspect state are represented in this sub-model. The main model elements are the places belonging to the modes, the error counter and the transitions representing mode changes and timeout of the suspect timer.

These subnets are composed by shared places (representing task errors, task failures, mode change triggers). The overall analysis model of the DMI consists of more than two hundred places and transitions.

Although the definition of these submodels was a manual task, their assembly was supported by our automated tool (Majzik et. al, 2007) on the basis of the UML architecture model (class and object diagrams) of the DMI. The tool identified the types of the UML model elements (resources, tasks and the links between them), then instantiated and connected the corresponding sub-models retrieved from a library of analysis sub-models. The dependability related parameters could be assigned either in the UML model (these were copied to the analysis model by the tool) or could be directly accessed in the SAN model (that was useful in the case of sensitivity analysis).

The solution of the dependability model by simulation in the Möbius tool (Daly et. al., 2000) provided information about the mean time to detected failure as well as to undetected failure. This latter contributed to the computation of the hazardous failure rate (that must not exceed the value allowed by the SIL2 requirement). Sensitivity analysis was useful to check the effects of changing the rate of periodic tests, the delay and coverage of on-line error detection techniques (e.g., tuning the resolution of control flow monitoring). The results showed that the availability and safety requirements can be satisfied with acceptable cost and overhead.

3.2 Evaluation of the effects of DMI properties on system level QoS parameters

It was also important to investigate the effects of DMI failures on train missions and QoS parameters observable by the passengers. These effects depend on the interaction scenarios between the EVC and the DMI and the application of on-board spare DMIs.

Two types of communication protocols were analyzed that could be used for the EVC-DMI interactions, based, respectively, on cyclic and acyclic exchange of messages. In a cyclic protocol the EVC periodically checks the operational status of the DMI, while in the acyclic case the failure of the DMI is detected only if a specific aperiodic request is sent to the DMI and no proper answer is received.

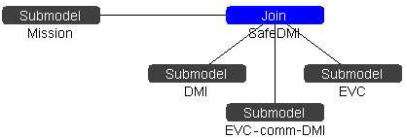

Fig. 5. The overall analysis model

Again, a modular modelling methodology was adopted and the following SAN sub-models were constructed and assembled (Fig. 5):

- The aggregated model of the DMI represented not only the failure of the DMI (here incorporating the results obtained from the evaluation of the DMI properties) but also the potential switchover to the on-board spare DMI and the repair process.
- The EVC sub-model represented the behaviour of the EVC in case of DMI failures (time-out mechanisms and emergency stop).
- The EVC-DMI communication sub-model represented the communication protocols.
- The mission model represented the typical scenario of missions (train rides).

Significant dependability-related indicators were evaluated like the probability to successfully complete a mission without unscheduled train stops, and the steady-state availability of the DMI-EVC system. The analysis of the obtained results pointed out the improvements in adopting an acyclic EVC-DMI

communication protocol. In case of a DMI failure this protocol increases the chance to boot a spare DMI and switch to it between two aperiodic EVC requests (while the short timeout of the cyclic protocol does not allow such seamless switchover). Sensitivity analysis quantified the effects of (1) applying different aperiodic timeouts, (2) having DMI with better MTTF parameters, and (3) having a spare DMI with a faster boot procedure (Lollini et. al, 2008).

4. EVALUATION OF THE WIRELESS COMMUNICATION PROTOCOLS

This subtask focused on performance evaluation, availability and dependability estimations and the impact of security. Performance evaluation includes the study of system behaviour in terms of commonly used parameters, such as throughput, delay, packet error and packet drops at different layers of the protocol stack. Dependability of a wireless connection is understood in the way that if the connection has been established, the file transfer (software uploading) should be completed within a predefined time. Availability of a wireless link means a successful security association setup that consists of a number of request-reply messages (security association hand-shake) that should be exchanged within 10 sec. Finally, quantification of the resource impact of adding the desired security functions was conducted.

The studies of the wireless communication protocols were done using experimental and simulation based approaches. Simulation approach has an advantage of a completely controllable environment; however its accuracy depends on how well the underlying models reflect the reality. The implementation was based on the de-facto standard Network Simulator 2 (NS2, 2005). This simulator is dominant in simulation based analysis of communication systems. The relevant channel extensions to enable wireless link properties at layers 1 and 2 were added to the selected simulation tool.

The approach to model the complex behaviour of the considered communication system was to split it into several sub-models that are tied together by the overall model design. Defining multiple sub-models enables their individual development primarily with the purpose of complexity reduction. The used feed-forward approach in modelling is illustrated in Fig. 6.

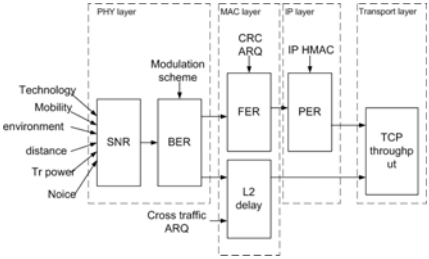

Fig. 6. The feed-forward modelling chain

The definition of sub-models can partially be based on layers in the protocol stack. This allows a starting point in already studied models such as channel models. The physical environment plays an important role in relation to the performance of the communication, since physical phenomena deteriorate channel conditions. The purpose of the first sub-model is to estimate Signal-to-Noise Ratio (SNR) due to phenomena existing in wireless channel, such as path loss, shadowing and multipath fading. Depending on the modulation scheme used, the second sub-model maps SNR to bit error rate (BER). BER influences both frame losses (FER) on layer 2 and delays due to retransmissions. Additionally, delay to access the channel depends on the amount of the cross-traffic. The modelling analysis considers 802.11b specifics for these sub-models. FER translates into packet error rate (PER) on IP layer. Delay and PER are used as input for the next model. The output of this sub-model is the TCP throughput.

The main focus of the overall model is to give dependability estimations and to give an answer in which environmental conditions the performance criteria are met. Under the assumption that a file transfer of 20 MB should be completed within 5 min, layer 4 throughput should be at least 66.6 kB/s.

The developed simulation model was constructed to reflect the SAFEDMI wireless communication solution.

To support these results real experiments were conducted. Additionally, the impact of the security solution based on IPsec was investigated. The test environment is depicted in Fig. 7. To consider a worst case analysis the bottleneck of the system is the Bridge Device. It has fewest resources while it has to execute cryptographic algorithms in the communication towards the Maintenance Centre. The MC has to perform the same functions but is however

assumed to be a powerful computer with capabilities exceeding the BD.

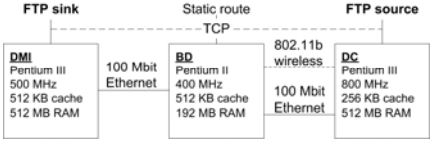

Fig. 7. IPsec testbench

The baseline performance of the system has been established without IPsec enabled. In the BD a static route has been defined that forwards packets from one interface to the other. A mean goodput of a file transfer using an FTP connection was calculated. The same measurement approach was repeated with the IPsec functions enabled for varying key sizes of the AES encryption algorithm. It was observed that for some cases the CPU of the BD was fully utilized, and thereby the BD is likely to be the bottleneck component. However, the actual BD throughput was sufficient to meet the performance requirements. Evaluation of the key management procedure showed that the delay due to establishment of the session keys is of order of seconds and, thus, it has a limited impact in relation to timely requirements (Grønbæk et. al, 2008).

5. EVALUATION OF THE DETECTION CODES AND RESIDUAL ERRORS

This subtask is focused on the analysis of detection properties (ability) of the detection and correction codes. The detection (correction) codes are used in the DMI mainly for integrity checks, which is essential for detection coverage in general. The malfunction of detection codes is described by the probability of an undetected error. To quantify this parameter for the linear detection codes the model of Binary Symmetrical Channel (BSC) was used.

The investigated codes can be split into several groups depending on the purpose of their use. The first group (one) includes the detection and correction codes which are used by the off-the-shelf protocols (lower layers of the communication protocol stack as Ethernet, TCP/IP). The second one comprises the detection codes of safety layers (safety codes). This group contains the safety codes used in EVC-DMI communication (CRC_SL2, CRC_SL4) and detection codes for file integrity checks. The last one is the detection code for ROM integrity checks. The above splitting is motivated only by the type of the stated requirements. In case of the first group, it is the reliability of communication and analysis of hazards from this level of communication. In case of the second and third ones, it is the required quantitative safety target.

The next splitting was performed when the type of detection codes was considered. The majority of the investigated codes is shortened cyclic codes. For this type of detection codes the inherent computational complexity of the BSC calculation is improved by several techniques. This analysis is based on the calculation of weight distribution with the help of a dual code (see Castagnoli et al, 1993) and search of the extreme behaviour of the undetected error probability $P_{ud}(p_e)$ (see Kárna 2008).

Our results point out that the complexity of the BSC calculation (analysis) is realizable up to 48 redundant bits and 5MB data length on a recent personal computer. None of the investigated CRC codes assures that the probability of an undetected error is less than the usually used bound 2^{-c} (where c is the number of redundant bits) for all possible lengths of code words. At the worst case (CRC_SL2) the result is 50 times greater than the usually used bound (see Kárna 2008).

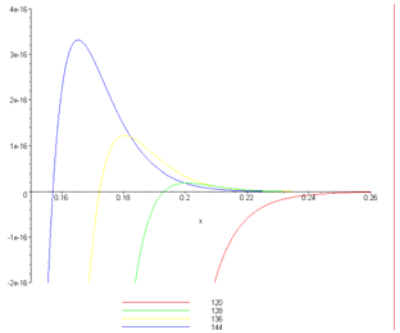

Fig. 8. Graph $P_{ud}(p_e)$-2^{-32} for code CRC_Cast – (code word lengths 120 – 144 bits).

One of the proposed CRC codes for file integrity checking was the 32 bits CRC_Cast analysed by Castagnoli et al. in 1993. This CRC is much more close to the estimation 2^{-c} than the other CRC, but the probability of an undetected error of this code is not upper bounded by this value. At the worst case the result is 1.0000014 times greater than the usually used bound.

To get the result presented on Fig. 8, it is necessary to apply high precision arithmetic for the extreme behaviour calculation. The Newton's method for finding the zero points of the derivation of the function $P_{ud}(p_e)$ was used with arbitrary precision calculation in the Maple tool. Our results are in slight contradiction with Castagnoli's ones, but from the practical point of view the differences are negligible.

6. CONCLUDING REMARKS

The quantitative evaluation of the DMI included the solution of analytic models, the application of simulation tools and experimental techniques. This paper focussed especially on the application of formal methods. It is important to note, however, that the formal evaluation techniques were effectively supported by (1) experimental measurements (e.g., measuring resource usage and timing of tasks, performance of wireless communication) that provided parameter values for the analysis and simulation models, and (2) fault injection experiments that increased our understanding of the efficiency of the applied error detection techniques. It is worth mentioning that the advantage of a model based design approach manifested itself in the evaluation phase (e.g., when dependability models were assembled) and also in the test generation phase (when the models were used as references).

ACKNOWLEDGEMENT

This work has been supported by the EC IST Project SAFEDMI (Contract n. 031413).

REFERENCES

CENELEC (2005). ERTMS – Driver Machine Interface Part 1-6, CLC/prTS 50459.

EN 50129 (2000). Railways applications – Communications, signalling and processing systems – Safety related electronic systems for signalling, CENELEC, April 2000.

EN 50128 (2001). Railways applications – Communications, signalling and processing systems – Software for railways control and protection system, CENELEC, March 2001.

EN 50159-2 (2001). Railway applications - Communication, signalling and processing systems - Part 2 – Safety related communication in open transmission systems, CENELEC, 2001.

ODP (1999). Information technology – Open Distributed Processing – Reference Model: Overview. ITU T Rec. X.901, ISO/IEC 10746.

SAFEDMI (2007). IST-FP6-STREP-031413 Safe Driver Machine Interface (DMI) for ERTMS Automatic Train Control. http://www.safedmi.org/

D. Daly, D. D. Deavours, J. M. Doyle, P. G. Webster and W. H. Sanders (2000). Möbius: an extensible tool for performance and dependability modeling. In Proc. *11th Int. Conference TOOLS 2000*, LNCS-1786, Springer Verlag, pp 332-336.

I. Rojas (1996). Compositional construction of SWN models. *The Computer Journal*, 38(7):612-621.

W. H. Sanders and J. F. Meyer (2001). Stochastic Activity Networks: Formal definitions and concepts. *Lectures on Formal Methods and Performance Analysis*, volume 2090 of LNCS, Springer Verlag, pp 315-343.

M. Serafini, P. Lollini, and A. Bondavalli (2006). Modeling On-line Tests in Safety-critical Systems. In Guedes, Soares and Zio (eds.), Safety and Reliability for Managing Risk, Vol. 1, Taylor & Francis Group, pp 231-238.

I. Majzik, P. Domokos and M. Magyar (2007). Tool-supported Dependability Evaluation of Redundant Architectures in Computer Based Control Systems. In Proc. FORMS/FORMAT 2007, pp 342-352, GZVB, Braunschweig.

P. Lollini, L. Montechi, M. Magyar, I. Majzik, A. Bondavalli (2008). Analysis of the Impact of Communication Protocols on Service Quality in ERTMS Automatic Train Control Systems. Regular paper submitted to FORMS/FORMAT 2008.

NS2 (2005) - The Network Simulator. http://www.isi.edu/nsnam/ns/

J. Grønbæk, T.K. Madsen, H.P. Schwefel (2008). Safe Wireless Communication Solution for Driver Machine Interface for Train Control Systems. In Proc. ICN 2008, Cancun, Mexico.

L. Kárná, Š. Klapka, M. Harlenderová (2008). Quantitative Assessment of Safety Code. Regular paper submitted to FORMS/FORMAT 2008.

G. Castagnoli, S. Brauer and M. Hermann (1993). Optimization of Cyclic Redundancy-Check Codes with 24 and 32 Parity Bits. *IEEE Transactions on Communications*, 41, 883-892.