Dependability evaluation of networked control systems under transmission faults

Rony Ghostine, Jean-Marc Thiriet, Jean-François Aubry

DEPENDABILITY EVALUATION OF NETWORKED CONTROL SYSTEMS UNDER TRANSMISSION FAULTS

Rony Ghostine*, Jean-Marc Thiriet, Jean-François Aubry***

*Institut National Polytechnique de Lorraine (Polytechnical National Institute of Lorraine)
Centre de Recherche en Automatique de Nancy / Nancy Research Centre for Automatic Control CNRS UMR 7039
2 avenue de la Forêt-de-Haye, 54516 Vandoeuvre, France,
rony.ghostine@ensem.inpl-nancy.fr, Jean-Francois.Aubry@ensem.inpl-nancy.fr

** Laboratoire d'Automatique de Grenoble (UMR 5528 CNRS-INPG-UJF)
BP 46
38402 Saint Martin d'Hères Cedex, France
jean-marc.thiriet@ujf-grenoble.fr

Abstract: The validation of network systems is mandatory to guarantee the dependability levels, that international standard impose in many safety-critical applications. This article presents a framework for the dependability evaluation of a networked control system (NCS) taking into account the network behaviour when transition faults occurs. CAN-based systems are studied and a proposed method is described with an example. *Copyright © 2006 IFAC*

Keywords: Networked control system, communication network, Petri nets, dependability evaluation.

1. INTRODUCTION

Networked control system (NCS) is a type of distributed control systems where sensors, actuators, and other devices are interconnected by communication networks. The study of NCSs is an interdisciplinary research area, combining both network and control theory. A study for the computer network and control system domains is thus a must. At first, many researches have investigated the performance of the network with faults, mainly to estimate the worst response time and the WCDFP (worst-case failure probability) (Navet, et al., 2000; Portugal and Carvalho, 2003). In these approaches, the behaviour of the network is studied without any consideration of the application it supports, and they assume that the system calms down after one or n successive lost messages. These assumptions don't hold true in systems dealing with soft real time. In fact, when control systems lose a message, it doesn't consequently damage the whole system. In control systems many studies were developed for the dependability estimation. Most of these approaches ignore the network failure, or assume that the messages are always successfully transmitted (Moncelet, et al., 1997). In fact industrial environment is specified by the existence of electromagnetic interferences (EMI). These interferences generate faults in electronic circuits that affect the normal operation. In communication systems, these faults usually affect the medium leading to a transmission error. In (Barger, et al., 2003), the authors take into consideration the message loss by assigning a fixed probability with a special focus: analysing the scenarios leading to the NCS failure.

In this communication we try to join the information emerging from these two domains for a dependability evaluation of a networked control system, taking into account the behaviour of the Local area network (LAN) under transmission error. While this study can be extended to other LANs, we focused on Control Area network (CAN). Our approach consists in modelling the LAN behaviour under error transmission and then to integrate this model to the control system global model.

2. DEPENDABILTY ISSUES

In this section we present the different works related to the dependability evaluation of control system. The validation of the dependability of such systems is mandatory: designers are asked by international standards to provide figures showing the dependability of their systems. For this purpose, several techniques are available. They can be grouped in the following categories, as suggested in (Geffroy, et al., 2002): quantitative analysis techniques (like fault injection, reliability block diagrams, and nondeterministic state graph models), inductive qualitative techniques (like Failure Mode and Effect Analysis), and deductive qualitative techniques (like Fault Tree).

In (Moncelet,. et al., 1997), the dependability of a mechatronic system was studied. Quantitative dependability evaluation is obtained thanks to a Monte Carlo simulation. In this work the authors assume that the entire messages are successfully transmitted between the components. That is why they totally ignore the network. In (Barger, et al., 2003) the message lost is taken into account by assigning a fixed probability with a special focus: analysing the scenario leading to the NCS failure. In these two works**,** the Coloured Petri Nets (CPN) (Jensen, 1997) are chosen as the modelling approach. The following section discusses the Controller Area network.

3. CONTROL AREA NETWORK

CAN is a broadcast bus, with a priority-based access to the medium and non destructive collision resolution. Data to be transferred is encapsulated within communication objects called frame. Each frame contains an identifier (Id), unique to the whole system, which serves two purposes: assigning a priority for the transmission and allowing message filtering upon reception. The following section presents the CAN behaviour in Fault Scenarios, and some related works.

3.1 Behaviour in Fault Scenario

In (Unruh, et al, 1989) the authors estimate the expected number of undetected transmission errors during lifetime of a vehicle is lower than 10^{-12}. This performance is the result of a very efficient error detection mechanism being used in CAN. This mechanism can be divided into: a message level like CRC code, and a bit level. At the bit level**,** the transmitter monitors the bus signals and detects errors. Each transmitting station observes the signal on the bus and thus, detects the difference between the bit sent and the bit received. If one error is discovered by at least one station using the above mechanisms, the current transmission is aborted by sending an *error flag*. This prevents other messages to accept this message. After sending the *error flag***,** the sender automatically re-attempts transmission, and the message re-enters a scheduling list, and the one with the highest priority is selected and transmitted on the bus. The number of retransmission is an important parameter. In the specification of CAN, this number is not defined. To calculate the worst response time**,** the authors in (Cheong, 2003) include the n_m: number of retransmission as a variable that characterizes the frame periodic *m*. As cited below, the EMI can affect the medium and produce a transmission error. Detection mechanism is added to cope with these errors.

Error recovery mechanisms take some time in detecting and retransmitting the affected message. This lost time is defined as an inaccessibility period where the network isn't ready to provide its service. This behaviour and its consequences are studied in (Rufino and Verissimo, 1995).

A scheduling analysis of CAN is done by (Tindell, et al., 1995). Tindell extended his study to integrate the presence of faults, by adding an additional term error recovery function. The main disadvantage of the analysis is the use of a deterministic model to represent the fault occurrence, which is not realistic.

Another extension to Tindell's work is presented by (Punnekkat, et al., 2000) providing a more general fault model which can deal with faults caused by several sources. This model like the previous one assumes a deterministic model for the fault occurrences.

A stochastic fault model which is closer to EMI behaviour and more realistic, was proposed by (Navet, et al., 2000). Generalized Poisson Process is used to model the frequency of interference, as well as their duration (single errors and error bursts). In this work the authors introduced for the first time the WCDFP (worst case deadline failure probability) which provides a valuable knowledge on the system's reliability. This information is very important when dealing with a hard real time system where losing a message can lead to a global failure of the whole system.

4. STOCHASTIC ACTIVITY NETWORK

Stochastic activity networks (SANs) (Movaghar and Meyer, 1984) are a stochastic generalization of Petri nets. These models permit the representation of concurrency timeliness, fault-tolerance and degradable performance in a single model. SANs are more flexible than most other stochastic extensions of Petri nets, including stochastic Petri nets (SPNs) and generalized stochastic Petri nets (GSPNs) (Abdollahi and Movaghar, 2005).
Structurally, they consist of activities, places, input gates, and output gates. Activities which are similar to transition in normal Petri nets, are of two types: timed and instantaneous. Timed activities represent activities of the modelled system whose duration impact the performance of the modelled system. Instantaneous activities, on the other hand, represent system activities which occur immediately. Input gates and output gates control the enabling of activities and define the marking changes that will occur when an activity completes.

SAN models have been used to evaluate a wide range of systems and are supported by several powerful modeling tools such as *UltraSAN* (Sanders, et al,. 1995) and *Möbius* (Deavours, et al,. 2002).

SAN is defined with the express purpose of facilitating unified Performance/dependability evaluation as well as more traditional performance and dependability evaluation (Sanders and Meyer, 2001). Dependability evaluation is performed by defining a set of measures in the model. In the context of *SPN*, these measures are derived from the concept of reward (Malhorta and Trivedi, 1995).

Our models were developed using Mobius tools, a tool that supports the use of SAN. Mobius tools allow the combination of single models into a composed model. Models consist of two parts: a description of the net structure and of the desired performance variables and solutions methods to be used in the evaluation process. Solutions include analytical techniques as well as terminating and steady-state simulations.

5. CASE STUDY AND MODELING

The method proposed is described with an example of a small NCS designed to control the liquid level in a tank between two levels. In order to do this, the NCS is composed of (fig. 1):

- two sensors,
- a controller,
- an actuator,
- a communication network

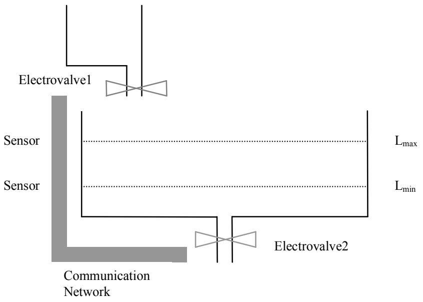


Fig. 1. The studied NCS example

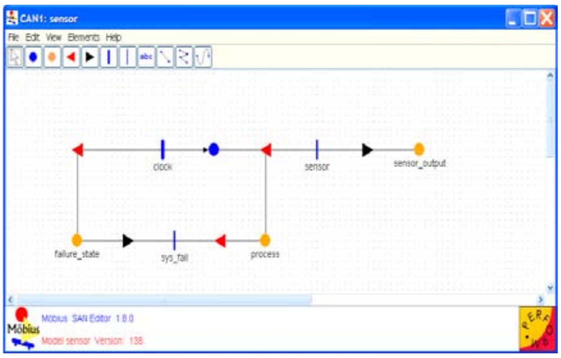
This example is similar to the one studied in (Barger, et al., 2000). The network chosen for our example is the Controller Area Network discussed in section 3. The global mission of the system is to maintain the liquid between two level values L_{min} and $L_{max.}$
In the initial state, the liquid level is set to L_{init}. And the electrovalve$_1$ is open while the electrovalve$_2$ is closed. The functional of the actuator is defined by the following rule denoted r_1:

- Each time the level exceeds the limit (L_{max} or L_{min}), open the closed electrovalve and close the opened one.

In the following is proposed a detailed description of the different sub-models of our system.

5.1 Sensor

A periodic sensor is considered. Every T_s (sensor's period) a new measure is obtained and prepared to be sent by putting a token in the *sensor_output* place. The instantaneous transition *sys_fail* tests if the liquid exceeds the limits. In this case this transition fires to indicate a failure situation and a token is added to the *failure_state.*



Möbius SAN Editor 1.8.0
Model sensor Version: 138

Fig. 2. Sensor

The extended place *process*, represented by an orange circle, contains information relative to the system state (liquid level).

5.2 Controller

The command delivered by the controller is defined by rule r_1 defined below.

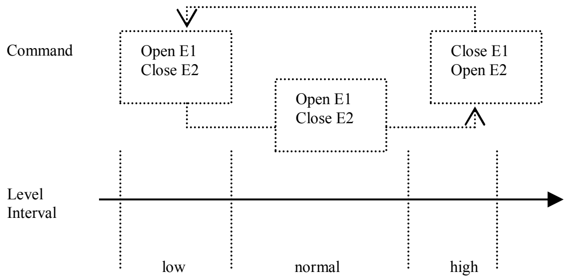


Fig. 3. Level intervals and the related command

The controller receives messages sent by the sensor, and executes the code of the transition *controller* which will operate directly on the actuator according to the r_1 rule, and sends the command on the medium to be sent to the actuator. The part in top is added to take into account the fact that the controller did not receive any message from the sensor (token in *Place1* is not consumed). At the end of a certain time T_c (the period of the controller), if the controller did not receive any message, the timed transition *T1* is fired, and its code is executed (this code takes the last value taken by the sensor) and a message is ready to be sent on the medium.

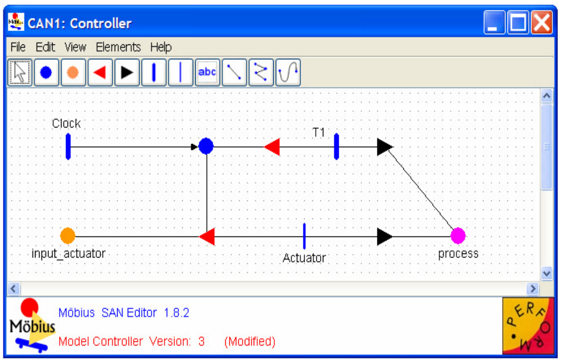


Fig. 4. Controller

5.4 Actuator

Input_actuator receives the messages sent on the medium, only the messages coming from the controller are accepted.

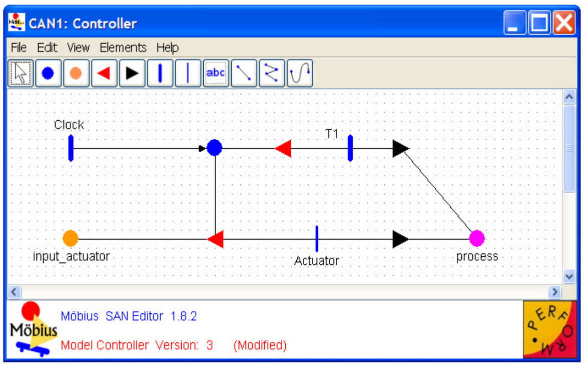


Fig. 5. Actuator

If after a T_a (actuator's period) unit of time the actuator did not receive any message, the timed transition *T1* fires and a security code is executed. For this example we close the electrovalve$_1$ and keep the electrovalve$_2$ untouched. We can imagine other strategies. After the fire of one of these transitions values in the process are updated to cope with the new state of the system.

5.5 Network

This submodel represents the behaviour of the CAN network. Messages to be sent are stored in the place *S*. The transition *TBT* is fired to indicate that a message is ready to be transmitted. If the medium is idle (token in *M*) and the message is the one who has the highest priority (code in the input gate of the *TTM* transition) *TTM* is fired and the message propagates on the medium.

If no interference disturbs the medium during the transmission, the message is correctly transmitted. Transition *ST* indicates time necessary to the transmission and depends on the size of the message. If an interference occurred (presence of a token in *IN*), an error is produced, and a token is removed to the *error* place which will activate the model *error_model* described later. A token is always present in the place *B*. If the message gains the access to the medium, another attempt to send is launched if none of these two conditions are satisfied:

1) The maximum number of retransmission for this message is reached.
2) A new message of the same type is ready to be sent (a token in place *S*).

If one of these conditions is true, the message is lost, and a token is added to the place *msg_loss*.

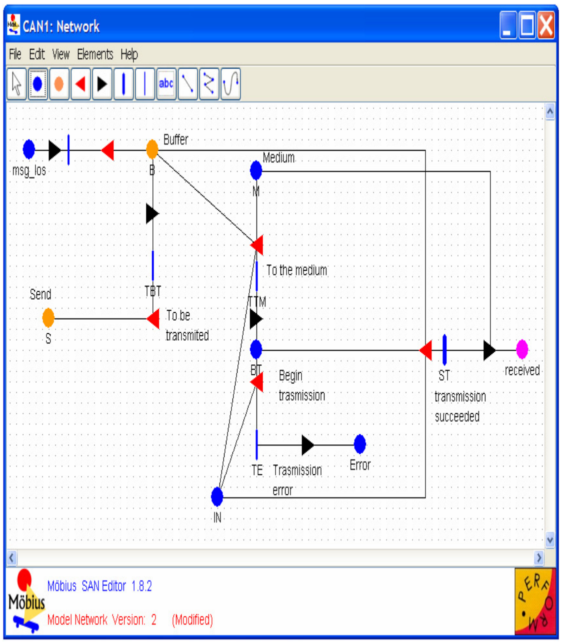


Fig. 6. Network

This is a simplified model of CAN operation, two assumptions were taken. The first is that transmission errors due to interferences on the medium are always detected and directly proclaimed by the error recovery mechanisms. This assumption is justified by the study of (Unruh, et al, 1989) which shows that the probability of undetected transmission errors during the lifetime of a vehicle is lower than 10^{-12}. The second assumption is that the arrival of a new message always deletes the old one.

This model is the link between all the other submodels, all other components, sensor, controller, and actuator are connected to the network model via common places who share the same variables. For instance, to join the controller to the network, two places are shared: *controller_output* with *MP* and *controller_entry* with *received*. This is done in the composed model thanks to the *join* option that allows combining two or more submodels using equivalence sharing.

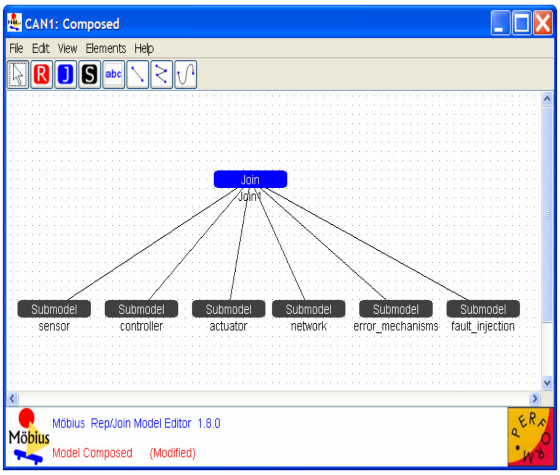


Fig. 7. Composed model

Error_mechanims represent the fact that the error mechanisms are launched and the medium is busy (no token in *TM*). We assume that the error frame takes always the maximum length (20 bits).

The *fault_injection* model injects faults representing an EMI via the medium (a token in *IN). *Faults occur in burst of random length and have a random duration, and their effects are equally observed by all the network nodes. This model compels with the analysis performed by (Kim, et al., 2000).

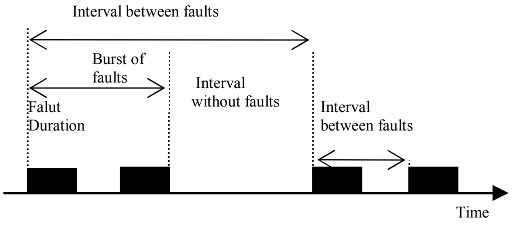


Fig. 8. Fault occurrence

6. RESULTS

As cited below the aim of our works is the dependability evaluation in the presence of transmission faults. The effect of the transmission error on the global mission is studied. We define the failure of our system when the level reached the limits (Pmax and Pmin). Faults are characterised by a fault duration *FD* and a faults rate *FR*. The Mean Time To Failure (MTTF) is studied according to these two parameters. Figures.9 shows the values of MTTF under different values of *FR* and *FD.*

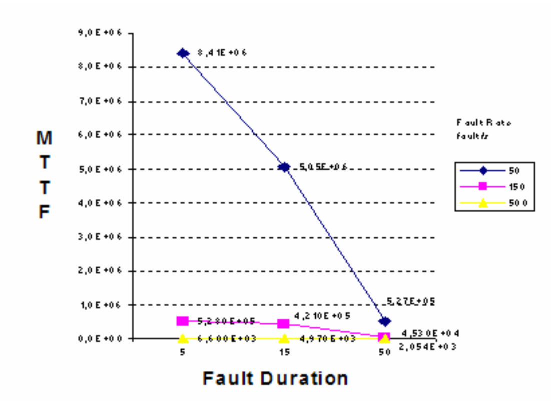


Fig. 9. MTTF

Three levels of faults rates are evaluated: 50, 150 and 500 faults/s. Fault duration takes three values 5, 15 and 50 bits. The interval with and without (Fig. 8) are defined as 1 min and 5 min respectively. The data length of all the messages circulating on the medium is set to 68 bits, for a data-rate of 250 Kbit/s. All the results are obtained by simulation under Mobius tools *(*Deavours, et al,. 2002) with a confidence interval of 0.1 and a confidence level of 0.95.

7. CONCLUSION

In this paper, we have presented an environment to assess the impact of the transmission errors on the global mission of a NCS. The approach consists of modelling the functional behaviour of classical control system components, and both functional and dysfunctional behaviour of the network. Thanks to our environment we were able to predict how the transmission errors on the network may affect the global mission. Special focus was given to the network; all the other components are considered as free fail. In future works, we will consider a global approach that takes into account both failures in the network and other components.

REFERENCES

Abdollahi Azgomi and A. Movaghar (2005), Hierarchical Stochastic Activity Networks: Formal Definitions and Behaviour, *International Journal of Simulation, Systems,* *Science and Technology*, **Vol. 6**, No. 1-2, pp. 56-66.

Barger, P., J.M. Thiriet, M. Robert (2003). Safety analysis and reliability estimation of a networked control system In: *SAFEPROCESS 2003,* Washington, D.C, USA.

Cheong SO, J.K (2003). Delay Modelling And Controller design for networked Control systems. Master of applied thesis. Department of Electrical and Computer Engineering University of Toronto.

Deavours, D.D., G. Clark, T. Courtney, D. Dalys, S. Derisavi, J.M. Doyle, W.H. Sanders, and P.G. Webster (2002). The Moebius framework and its implementation. *IEEE Trans. On Soft. Eng*, **Vol. 28,** No 10**,** pp 956-969.

Jensen, K. (1997). Coloured Petri Nets, Basic Concept, Analysis Methods and Practical use. *Monographs in theoretical Computer Science,* Springer-Verlag, 2nd correcting printing 1997.

Malhotra, M. and K. Trivedi (1995). Dependability Modelling Using Petri-Nets, *IEEE Transaction on reliability*, **Vol. 44,** No. 3, pp. 428-440.

Moncelet, G., S. Christensen, H. Demmou, M. Pauldetto and J. Porras (1998). Dependability evaluation of a simple mechatronic system using coloured Petri nets In: *Workshop on Practical Use of Coloured Petri Nets and Design CPN* (Jensen, K.), pp. 189-198. Aarhus University, Aarhus, Denmark.

Navet, N., Y. Song and F. Simonot (2000). Worst-Case Deadline Probability in Real-Time Applications Distributed over Controller Area Network. In: *Journal of systems Architecture*, **Vol. 46**, No. 1, pp. 607-617.

Portugal, P.J. and Carvalho, A. (2004). A Stochastic Petri Net *Framework for Dependability Evaluation of Fieldbus Networks - A Controller Area Network (CAN) Example*. *International IEEE Conference in Mechatronics and Robotics – MECROB*.

Punnekkat, S., H Hannsoon and C. Norstom (2000), Response Time Analysis under errors for CAN, Proceeding of IEEE Real-Time Technology and applications Symposium.

Rufino, J. And P. Verissimo (1995), A study on the inaccessibility characteristics of the controller area network, *Proceeding of the 2nd International CAN Conference* .

Sanders, W.H., W.D. Obal, M.A. Qureshi and F.K. Widjanarko (1995). The UltraSAN modelling environment. *Performance Evaluation*, **Vol. 24,** No. 1-2, pp 89-115.

Unruh, J. H.J. Mathony and K.H. Kaiser (1989). Error detection analysis of automotive communication networks, *Technical report, Robert Bosh GmbH.*

William H. S., John F. Meyer. (2002). Stochastic activity networks: formal definitions and concepts, *Lectures on formal methods and performance analysis: first EEF/Euro summer school on trends in computer science,* pp 315-343. Springer-Verlag New York, Inc., New York, NY.