# An experience of dependability assessment of a typical industrial safety critical Programmable Logic Controller

Michele Minichino, Ester Ciancamerla
ENEA CRE Casaccia, Roma, Italy
{minichino, ciancamerlae}@casaccia.enea.it

Silvano Chiaradonna
CNUCE/CNR, Pisa, Italy
silvano.chiaradonna@cnuce.cnr.it

Andrea Bondavalli
DSI, University of Florence, Italy
a.bondavalli@cnuce.cnr.it

## Abstract

In this paper, we describe an experience of dependability assessment of a typical industrial Programmable Logic Controller (PLC). The PLC is based on a two out of three voting policy and it is intended to be used for safety functions. Safety assessment of computer based systems performing safety functions is regulated by standards and guidelines. In all of them there is a common agreement that no single method can be considered sufficient to achieve and assess safety. The paper addresses the PLC assessment by probabilistic methods to determine its dependability attributes related to Safety Integrity Levels as defined by IEC61508 standard. The assessment has been carried out by independent teams, starting from the same basic assumptions and data. Diverse combinatorial and state space probabilistic modelling techniques, implemented by public tools, have been used. Even if the isolation of teams was not formally granted, the experience has shown different topics worthwhile to be described. First of all, the usage of different modelling techniques has led to diverse models. Moreover models focus on different system details also due the diverse teams skill. Also slight differences in understanding PLC assumptions have been occurred. In spite of all, the numerical results of the diverse models are comparable. The experience has also allowed a comparison of the different modelling techniques as implemented by the considered public tools.

## 1. Introduction

Safety assessment of computer based systems, which perform safety functions is regulated by standards and guidelines. The majority of them are specific to an industrial sector or to a type of application within a sector. Currently, an emergent standard which does not address any specific industrial sector is IEC 61508. IEC 61508 provides a safety framework within which any safety critical system, based not only on programmable electronic systems can be addressed. The strategy proposed by IEC 61508 takes into account the fact that failures spread across the whole safety lifecycle and gives emphasis other than on technical requirements, on management of activities bearing on functional safety and on competence of people involved in safety activities, for the whole safety lifecycle. In IEC 61508, as well as, in all the other standards and guidelines there is a common agreement that no single method can be considered sufficient to achieve and assess safety. A very important concept in IEC 61508 is that of Safety Integrity Level (SIL). SILs are used as the basis for specifying the safety integrity requirements for the safety functions to be implemented by the safety critical system. Target failure measures are associated with SIL. That means that a safety critical system with a specific SIL will have the same safety integrity characteristics irrespective of the application in which it is used. As far as it concerns the determination of the appropriate Safety Integrity Level, IEC 61508 is based on the concept of the risk and provides a number of different methods, both qualitative and quantitative, for determining it. The present paper deals with probabilistic methods to assess dependability attributes of a typical industrial Programmable

Logic Controller, based on two out of three voting policy, referring to the Safety Integrity Levels as defined by IEC61508 standard.

The probabilistic dependability assessment of the PLC has been performed starting from its specification and data [1], [2] and selecting the appropriate probabilistic modelling techniques. The paper reports in detail the assessment by three different modelling techniques and related tools, one representative of combinatorial modelling technique (Fault Tree Analysis) and the other two representative of state space modelling techniques (Generalised Stochastic Petri Nets and Stochastic Activity Networks) [4], [5]. Particularly, the paper proceeds as follows. Section 2 describes the PLC architecture and its fault-tolerance capabilities. Section 3 contains the set of common assumptions underlying all the modelling efforts performed by the different teams and identifies the dependability measures objective of the assessment. Section 4 introduces the probabilistic modelling techniques and some main selection criteria. Sections 5 describes the PLC models respectively obtained using Fault-tree Analysis, Generalised Stochastic Petri Nets and Stochastic Activities Networks. Section 6 contains the results of the assessment. Section 7 reports some main remarks on this experience and finally, section 8 contains our conclusions.

## 2. The Programmable Logic Controller

At the present time, safety critical PLCs are usual*y* characterised by multi-channel structures either in individual areas or in the entire system. The required safety is achieved by automatic tests and comparisons or voting. Depending on the detected fault and on the structure of the system, a module, a channel or the entire system is safely shut down. In some cases degraded operation is permissible for a limited period of time.

The particular solution of the Programmable Logic Controller (PLC) under consideration consists of three different channels, which are used to process the input signals and of a two-out-of-three hardware voter which collects channel results to produce the output [1], [2]. The PLC has been specified at different levels of detail. The block diagram of the safety critical Programmable Logic Controller (PLC), at the first level, is shown in figure 1.

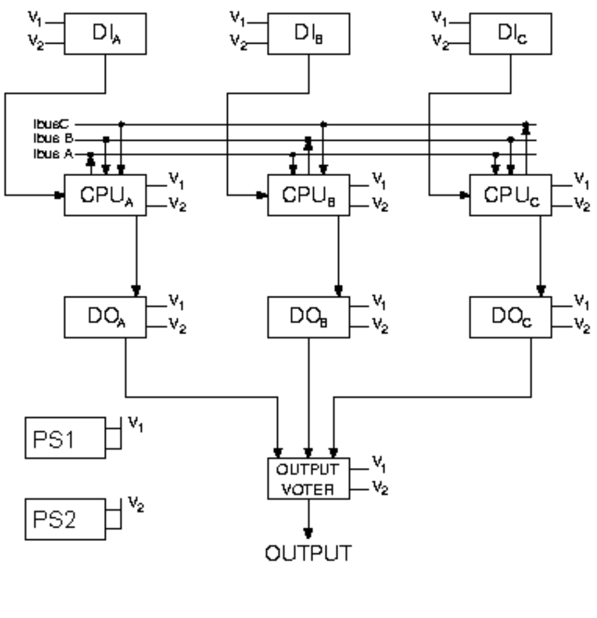


**Fig. 1.** The PLC block diagram

For each channel (identified as channel *A*, *B* and *C* respectively) a digital input unit (*DI*), a processing unit (*CPU* ) and a digital output unit (*DO*) are employed. The digital signal elaborated by a channel is transmitted among units through a special bus called *IO* bus. Moreover, input redundancy is adopted at the *CPU* level; indeed, each *CPU* receives from its

*DI* the signal to be elaborated, but it also receives a copy of the signal from the other *CPUs*. For doing this, three buses are used, called *Inter-channel bus A*, *Inter-channel bus B* and *Inter-channel bus C* respectively. Each *CPU* performs then a software majority voting 2 out of 3 for determining the input signal. Finally, redundant power supply is provided: two independent power supply units (*PS 1* and *PS 2*) are connected to all the components, in such a way that only the breakdown of both *PS* units prevents the system to work because of power supply problems.

At more detailed levels, each PLC elementary block is decomposed in components and sub-components. At the deeper level, the failure data of each sub-component are available as failure rate input data [2]. Some sub-components are equipped with diagnostic capabilities [2].

## 3. PLC Basic assumptions, input data and required measures

The same basic assumptions, common to the diverse teams, have been undertaken:

- software is perfect;
- hardware components failure rate is constant;
- all failures are considered permanent;
- failures of different components are statistically independent;
- operations start with all components properly working;
- no repair is allowed.

Starting from the common basic assumptions, each team added its own assumptions, according to his individual understanding of specifications, the abstraction level of the models and his skill. Moreover, starting from the given failure rate data of each sub-component at the most detailed level, each team derived the specific failure rates to apply according to his own assumptions and to the abstraction level of his model.

The PLC dependability measures to be computed by the diverse teams are:

- MTTF (Mean Time To Failure);
- the interval of time in which the probability of failure of the PLC is lower than the upper bound of the values of SIL2 of IEC 61508;

and, depending on the technique adopted, also

- the critical set of components;
- sensitivity analysis to input failure parameters.

IEC 61508 identifies two categories of systems: low demand mode of operation and continuous/high demand mode of operation and 4 Safety Integrity Levels. The standard defines the quantitative requirements on the failure rate/probability for each level. The target failure measures for the 4 Safety Integrity Levels are specified in Table 1.

| SAFETY INTEGRITY LEVEL | DEMAND MODE (Probability of failure on demand) | CONTINUOUS MODE (Probability of a dangerous failure per hour) |
|---|---|---|
| 4 | >=$10^{-5}$ to <$10^{-4}$ | >=$10^{-9}$ to <$10^{-8}$ |
| 3 | >=$10^{-4}$ to <$10^{-3}$ | >=$10^{-8}$ to <$10^{-7}$ |
| 2 | >=$10^{-3}$ to <$10^{-2}$ | >=$10^{-7}$ to <$10^{-6}$ |
| 1 | >=$10^{-2}$ to <$10^{-1}$ | >=$10^{-6}$ to <$10^{-5}$ |

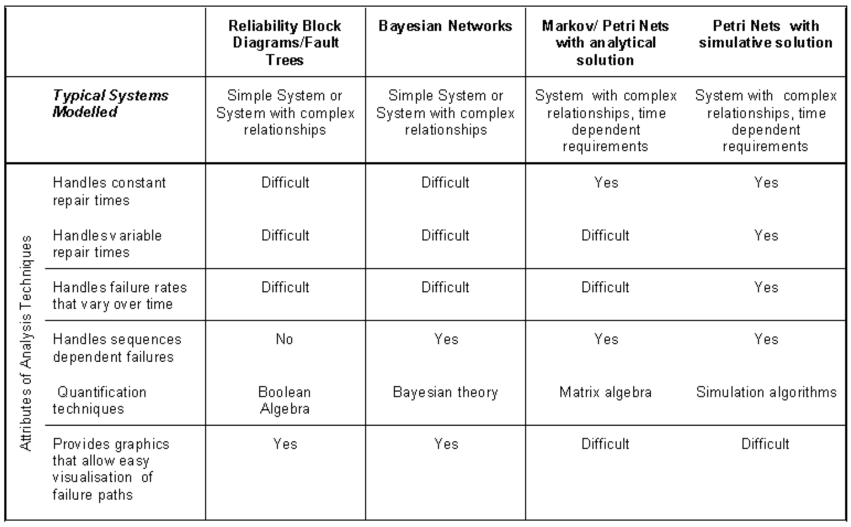
**Table 1.** Target measures for the SIL levels of IEC 61508 standard

Our PLC works with continuous/high demand mode of operation and its safety function is the correct delivery of the elaborated digital inputs. The PLC fault tolerant architecture and the failure modes of its constituent components, are meant to be compatible with Safety Integrity Level 2, also assuming the worst case of system diagnostic coverage equal to zero.

## 4. Selecting PLC modelling techniques

Dependability modelling of systems, in which statistical independence of the faults activation affecting the components can be assumed, is relatively simple. In such a framework,

combinatorial models, such as reliability block diagrams and fault tree analysis, can be used [11]. Combinatorial models have scarce modelling power but are able to deal with large dimension problems, avoiding state explosion and are often cheaper in term of computational costs. Where combinatorial models can be used, obviously state-space based models can be used too [3]. On the contrary, if system components are statistically dependent, models based on the space of states are needed to be used. Particularly if system activities, such as failure and repair activities, are characterized by time constant parameters, probabilistic modeling techniques based on Markov Chains can be used. When system activities are more complex, in terms of statistical dependence of its components and/or with time variable activities, probabilistic modeling techniques based on extended Petri Nets, can be used.
Table 2 shows a comparison among probabilistic modeling techniques.



| | | **Reliability Block Diagrams/Fault Trees** | **Bayesian Networks** | **Markov/ Petri Nets with analytical solution** | **Petri Nets with simulative solution** |
|---|---|---|---|---|---|
| | ***Typical Systems Modelled*** | Simple System or System with complex relationships | Simple System or System with complex relationships | System with complex relationships, time dependent requirements | System with complex relationships, time dependent requirements |
| Attributes of Analysis Techniques | Handles constant repair times | Difficult | Difficult | Yes | Yes |
| | Handlesvariable repair times | Difficult | Difficult | Difficult | Yes |
| | Handles failure rates that vary over time | Difficult | Difficult | Difficult | Yes |
| | Handles sequences dependent failures | No | Yes | Yes | Yes |
| | Quantification techniques | Boolean Algebra | Bayesian theory | Matrix algebra | Simulation algorithms |
| | Provides graphics that allow easy visualisation of failure paths | Yes | Yes | Difficult | Difficult |

Table 2. Comparison among probabilistic modelling techniques

Among such techniques, table 2 includes Bayesian Networks (BN). BN constitute an interesting intermediate proposal between combinatorial techniques and state-space based techniques [10], allowing to consider local statistical dependencies among components. Markov provides great modelling flexibility, analytical solution, various dependability measures (i.e. reliability, safety, and availability), and various indexes of measure (transient probabilities, steady-state probabilities, and mean sojourn time). A useful extension of Markov modelling is to associate a reward or a cost variable at the states of the model thus allowing performability metrics to be evaluated. Anyway modelling by Markov presents limits: -difficulties in defining the values of transition rates for each state of the system and more in general to assign proper values to the parameters; -scarce flexibility: introducing even little changes in system architecture requires completely different models; -restriction to the use of exponential distribution of all system activities.

Modelling by Petri Nets allows immediate representation of logical interaction among subsystems and of system activities (synchronisation, sequentially, concurrency,…) due to the higher expressiveness. When it is possible represent the probability distribution of the system activities with exponential distribution, analytical solution of Petri Nets (SPN,GSPN) is granted through the solution of the underlying stochastic process which is a Continuous Time Markov Chain. When activities are not exponentially distributed, general simulative solution can be obtained, although, in some specific cases, more complex underlying stochastic processes can be identified as the marking process of the Petri Net and solved analytically (Markov Regenerative Processes, Non Homogeneous Markov Processes, etc.).

Due to the PLC basic assumptions, any of the probabilistic modelling techniques, collected in table 2, could be selected. Within the experience we made, some of them have been selected (either because available or better known to the teams ) and then applied to the PLC dependability assessment by using public tools. In particular we used:

- Fault tree analysis, by Sharpe [7] and Item software [6]
- Generalised Stochastic Petri Nets, by Surf-2 [8]
- Stochastic Activity Networks, by UltraSAN [9]

## 5. PLC models and analysis

### 5.1 PLC Fault Tree Analysis

The goal of FTA is to represent the combination of the elementary causes, called the primary events that lead to the occurrence of an undesired catastrophic event, the Top Event. Events are combined by logical gates, namely AND gates and OR gates or derived logical gates, namely «k-of-n» gates. Events are binary, that is they may assume two values corresponding to the faulty and non-faulty (working) behaviour of the represented elementary block.

The FT for this PLC system is reported in figure 2

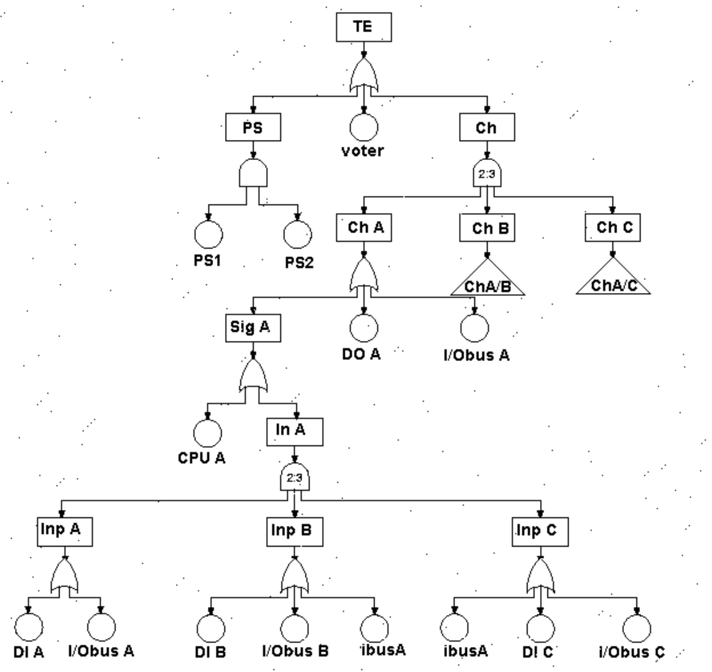


**Fig. 2.** The Fault Tree of the PLC

The Top Event represents the PLC failure and the primary events represent the failure of the elementary blocks of the PLC as shown in figure 1. The failure rate of each elementary block has been derived taking into account the failure rate of its sub-components and assuming that the failure of the block is induced by the failure of any of its sub-components. Two-out-of-three gates are used to model the fault of the input part of each channel (since each CPU adopts a 2-out-of-3 majority voting) and the fault corresponding to the fault of at least 2 channels (since the voter also uses a 2 out of 3 majority voting). Given the FT of the PLC, typical analyses have been performed.

*Qualitative Analysis*. to determine the Minimal Cut Sets (MCS), i.e. the minimal (with respect to set inclusion) set of primary events causing the occurrence of the TE. From the logical point of view they are the prime implicants of the TE.

*Quantitative Analysis*. i) to determine the probability of occurrence of any event in the FT and in particular of the TE given probabilistic information about truth of primary events; ii) to determine the importance of MCS, also called their unreliability.

## 5.2 PLC Generalised Stochastic Petri Nets model

GSPN allow a representation of the PLC failure behaviour by a reduced number of basic elements (places, transitions and tokens) that can be analysed by a Markovian Chain defined on the overall state space of the system. The GSPN model of the PLC has been developed in two steps. Referring to the PLC block diagram, figure 2, three identical modules, related to fault behaviour of each channel, can be identified. According to that, at a first level, a GSPN model representative of the fault behaviour of a single channel has been implemented as shown in figure 3.

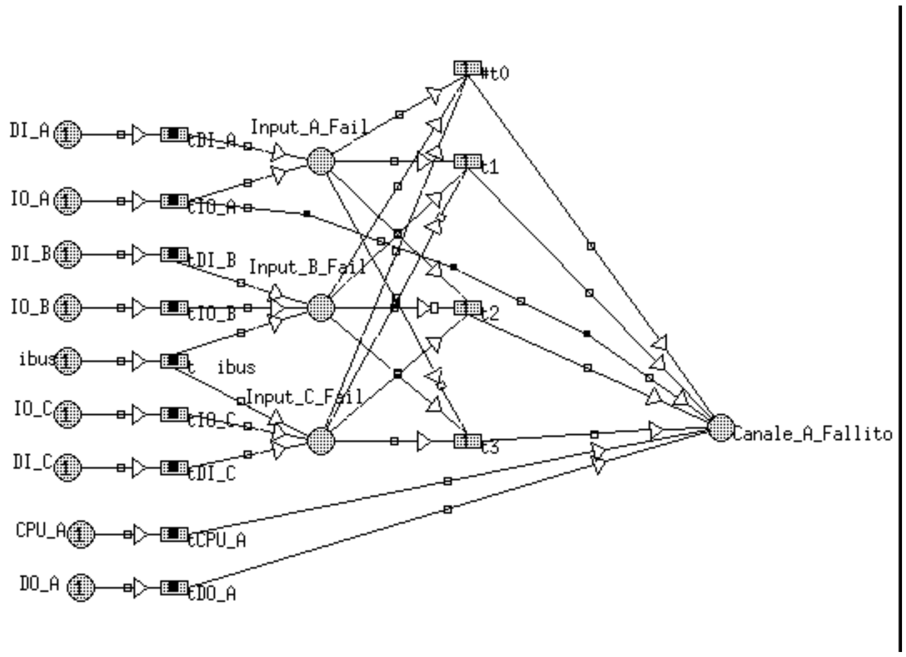


**Fig. 3.** GSPN model of the fault behaviour of a PLC single channel

At time *t=0* , all components of the channel are correctly working (one token is present in each place: *DI_A, IO_A, DI_B, IO_B, i_bus, IO_C, DI_C, CPU_A, DO_A* ). The failure of *DI_A* elementary block (firing of transition *tDI_A* ) or the failure of *I/O_A* bus (firing of the transition *tI/O_A* ) implies the *Input_A* failure (one token in the place *Input_A_fail* ). The failure of *DI_B* elementary block (firing of transition *tDI_B* ) or the failure of I/O_B bus (firing of the transition *tI/O_B* ) or the failure of the *Inter-channel bus* (firing of the transition *t_ibus*) implies the *Input_B* failure (one token in the place *Input_B_fail*). The failure of *Input_C* (one token in the place *Input_C_fail*) is modelled as the failure of the *Input_B*. The possible failure of the input part of each channel (represented by the presence of one token in each place *Input_A_fail*, *Input_B_fail*, *Input_C_fail*) is submitted to the software voter. The two out of three software voting is modelled by four transitions (*t0, t1*, *t2*, *t3*). In the case of at least two out of three failures in the input part of the channels ( the presence of one token in at least two of the places *Input_A_fail*, *Input_B_fail*, *Input_C_fail*) gives the failure of the single channel (one token in the place *Channel_A_fail*). The failure of the single channel is also due to the failure of its *CPU* (firing of the transition *tCPU_A*) or the Failure of its channel bus (firing of the transition *tIO_A*) or the failure of its digital output (firing of *DO_A*).

At the second level, a GSPN model representative of the fault behaviour of the PLC has been implemented as shown in figure 4.

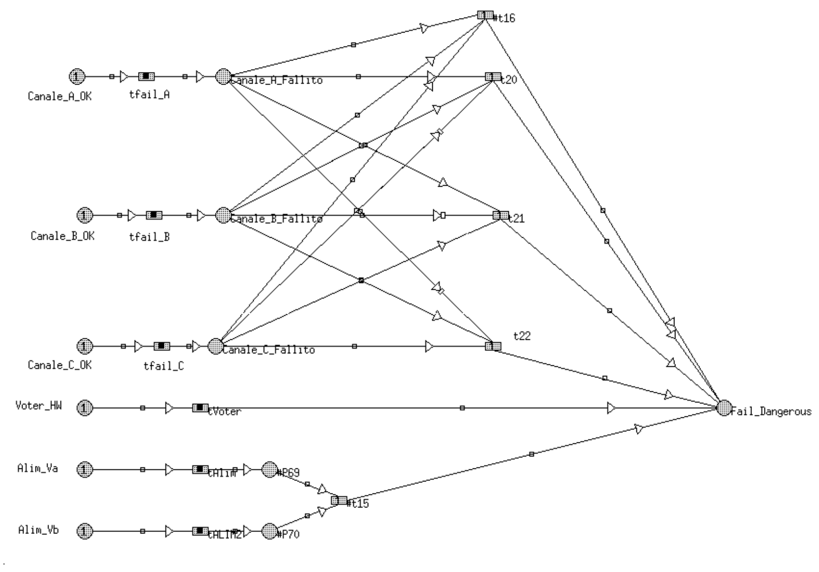


**Fig. 4.** GSPN model of the fault behaviour of the whole PLC

At time *t=0*, each channel is working (one token is present in each place: *Canale_A_Ok*, *Canale_B_Ok*, *Canale_C_Ok*), the redundant power supply is working (one token in each place *Alim_Va*, *Alim Vb*) and the voter hardware is working (one token in each place *Voter_Hw*). The Failure of the PLC (one token in the place *Fail_Dangerous*) is due to the failure of at least two out of three channels, or to the failure of both power supplies (one token in each place *#P69* and *#P70*) or to the voter failure. The failure of a single channel is represented by the firing of any of the transitions *tfail_A*, *tfail_B*, *tfail_C*; the failure rate of these transitions has been computed by the previuos model. The two out of three majority logic has been implemented as in the previous model. For modelling the PLC using GSPN, SURF-2 [6] has been used.

## 5.3 The Stochastic Activity Networks model

SAN are a powerful modeling formalism which extends GSPN in many ways. The most relevant high-level modeling constructs offered are the input and output *GATES*, which allow to specify controls on the net execution , and constructs for hierarchical modeling such as *REP* and *JOIN*. These last allow to build composed models based on simpler sub-models, which can be developed independently and then replied and joined with others sub-models. SAN are supported by UltraSAN [9].

The SAN model of the PLC keeps into account:

- PLC sub-components self checking capabilities, so that besides the normal faults accumulation, faults may result also in an isolation of the affected component, which induces a degradation of the system less severe than that induced by a non detected fault.
- Hardware voter redundancy, the voter has some redundancy at the level of its sub-component by which the PLC still delivers correct service despite the failure of one of the voter sub-component provided all the three channels are working properly.
- Faults of the busses in the channel (either *I/OBUS* and *Inter-channel bus*) prevent the CPU of the channel to communicate with the other components of the system. Thus, the busses and the CPU, have been collapsed together in building the sub-model of the channel.

The PLC model structure takes advantage of the modularity and hierarchical composition allowed by the SAN. Thus first sub-models have been defined for each component of the PLC, identifying also the common places through which these sub-models can interact. Then, these sub-models have been «composed» to define the overall model of the system using the operations provided by the tool.

Particularly, five sub-models have been identified: *DI*, *CPUBUS*, *DO*, *VOTER* and *ALIM*, connected and replied according to the "Composed Model" shown in Figure 5.

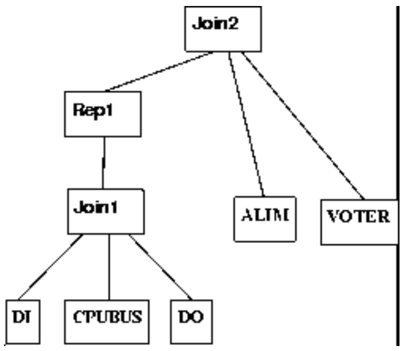


**Fig. 5.** Composed Model of the PLC

The logical structure of this composition is simple. The three sub-models *DI*, *CPUBUS* and *DO* are joined by the node *Join1*, defining the composed model for a single channel of the PLC. Node *Rep1* models the three channels of the PLC, replicating the model of the single channel (sub-tree *Join1*). Finally, the definition of the overall model of the PLC is obtained by composing the three channels (sub-tree *Rep1*) with the models *ALIM* and *VOTER* in node *Join2*, which is the root of the entire Model.

The five sub-models are modelled according to a common logic structure. Each sub-model is composed of two distinct parts. The first (left part in the Figures) models the faulty behaviour of the component, the second (right part in the Figures) models the effect of the fault, which can either remain confined to the component, affect the channel or give rise to the failure of the overall PLC. Figures 6 and 7 show the sub-models of *CPUBUS* and *VOTER* respectively.

The sub-model *CPUBUS* is the more complex of the five sub-models, because of the failure-detection mechanism and of the modelling in the same sub-model of the busses of channel. In addition, when a fault occurs in the *CPUBUS*, it is necessary to test all the others components to verify if the fault give rise to a failure of the PLC system.

At time *t=0*, the three places *cpu_ok*, *fd_ok* and *bus_ok*, contain one token each, meaning the correct behaviour of the components CPU, failure-detection circuitry and *I/OBUS* and *Inter-channel bus*, respectively; and no tokens are in the other places. The three timed activities *faulty_cpu*, *faulty_fd* and *faulty_bus* model the fault processes of the three components, and are enabled and disabled respectively by the input gate *i_cpu*, *i_fd* and *i_bus*. When the activity *faulty_cpu* completes (i.e. a fault occurs in the component CPU), the input gate *o_cpu* is performed. This gate first checks whether the failure-detection circuitry is still correct (one token is in the place *fd_ok*). If this is the case it adds one token in the place *isolated_chan*, representing the detection of the faulty state of the CPU, and the subsequent isolation of the entire affected channel. Otherwise, if the failure-detection circuitry is failed, the output gate *o_cpu* deposits one token in the place *cpu_bus_failed*. The completion of activity *faulty_fd* models the occurrence of a fault in the failure-detection circuitry. In this case, there is a probability (*prob_falsealarm*) that a false alarm is generated, i.e. that the non faulty channel is isolated. The completion of activity *faulty_bus* always implies one token is added to the place *cpu_bus_failed*. When any of the three activities complete, one token is added to the place *go_test*. This place and the instantaneous activity *i_test* serve only to perform the output gate *test_di*, *test_cpu_bus*, *test_do*, *test_voter* and *test_isolated*. These gate perform tests and updates of the state of the entire PLC.

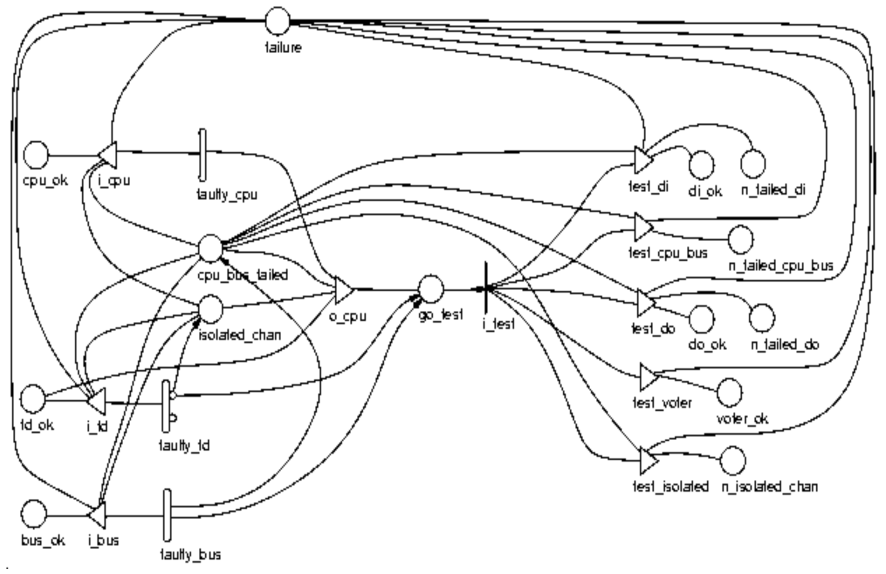


**Fig. 6.** Submodel CPUBUS

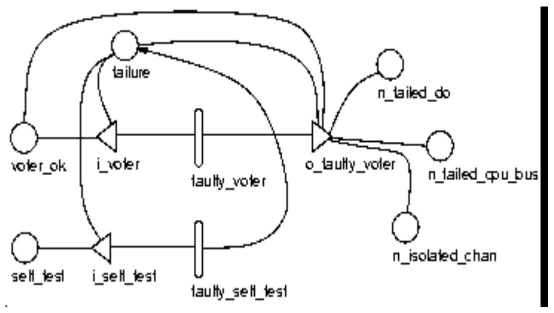


**Fig. 7.** Sub-model VOTER

The *VOTER* is composed of four switches (and the correspondent drivers) which implement a 2-of-3 voting logic. In the sub-model *VOTER*, shown in Figure 7, the four switches are represented at the time *t=0* by four tokens in the place *voter_ok*. The activity *faulty_voter* models the fault process of the switches, it has an exponential rate dependent on the number of tokens in *voter_ok*. The input gate *i_voter* disables the activity after the failure of the PLC. The places *n_failed_do*, *n_failed_cpu_bus* and *n_isolated_chan* are in common with the other sub-models.

When the activity *faulty_voter* completes for the first time (first switch or driver failure), the output gate *o_faulty_voter* tests whether there are other faulty components, which lead to the failure of the overall PLC, and in such a case deposits a token in the place *failure*. When the activity *faulty_voter* completes for the second time, the output gate *o_faulty_voter*, deposits one token in *failure* without performing the tests. At the time *t=0*, the place *self_test_ok* contains one token and models the correct behaviour of the two functions of test «Stuck ON Conditions Simulation Test" and "Cables Open Test". The activity *faulty_self_test* models fault process of the components used by these functions. A pessimistic, conservative assumption has been considered here: one fault in one of these components gives rise to a failure of the overall PLC system (one token is added to the place *failure*).

## 6. Results of the PLC analyses

This section reports the results of the PLC analyses, obtained by using FTA, GSPN and SAN. The main common assumption was that each PLC elementary block fails on the failure of any of its sub-components. The failure rates of the PLC elementary block used for the analysis are shown in table 3.

| PLC elementary block | Failure rates [failures/h] |
|---|---|
| DI | $\lambda_{DI} = 2.8\ 10^{-7}$ |
| CPU | $\lambda_{CPU} = 4.82\ 10^{-7}$ |
| DO | $\lambda_{DO} = 2.45\ 10^{-7}$ |
| I/O Bus | $\lambda_{I/O} = 2.0\ 10^{-9}$ |
| In_ch bus | $\lambda_{TB} = 2.0\ 10^{-9}$ |
| Voter HW | $\lambda_{votet} = 6.6\ 10^{-8}$ |
| Power supply | $\lambda_{al}= 3..37\ 10^{-7}$ |

**Table 3.** Failure Rates for PLC components

## 6.1 Results of the PLC Fault Tree Analysis

Sharpe [7] and Item Software [6] have been used for the Fault Tree Analysis of the PLC. Sharpe has been used to compute the mean time to reach the TE, its variance and the probability to reach TE along the mission time.

*Mean Time to Failure = 8.3221 e+05* (about *96 years*) with *variance = 2.8801e+11*.

The probability of system failure P(TE) have been computed from time *t=0* to time *t = 4*10E5 hours*.

At time *t = 4*10E5 hours* , *P(TE) = 0.22053*.

The interval of time in which the probability of failure of the PLC is lower than the upper bound of SIL2 of IEC 61508, is *T<60000 hours* (about *5 years*) and have been derived from Sharpe measures.

The results obtained with Sharpe have then been verified using Item software. Since fault_tolerance of the PLC is a major issue, determining the criticality of system components or sub_systems is very important. The criticality of PLC components have been obtained analysing the Minimal Cut Sets of FTA. SHARPE does not allow to dermine Minimal Cut Sets, thus Item software has also been used to determine them and their importance, also called their unreliability.

MCS determination is usually based on minimisation techniques on the set of boolean functions represented by the gates of the FT. The cardinality of a cut_set is called its order. The FT of the PLC, figure 2, has 59 MCS, one of order 1 (corresponding to the fault of the voter) and the remaining 58 of order 2.

For instance, a MCS of order 2 that can immediately be derived from the FT of figure 2 is *{PS 1, PS 2}* corresponding to the fault of both power supplies. Given the unreliability of the PLC Minimal Cut Sets is possible to derive the criticality of components of the PLC.

MCS have been ranked in order of unreliability; table 5 shows the first 11 MCS and their corresponding unreliability. We can easily verify that the most critical components of the PLC, are the *CPUs*, since the Cut Sets involving the failure of at least two *CPUs* are those showing larger unreliability. The failure of *Voter*, the only MCS of first order, is less important of the Minimal Cut Sets involving the failure of at least two *CPUs*. Notice also that Cut Sets involving digital output units have a higher unreliability than those containing digital inputs units; this is

| MCS | Unreliability |
|---|---|
| $\{CPU_A, CPU_B\}$ | 0.03075 |
| $\{CPU_B, CPU_C\}$ | 0.03075 |
| $\{CPU_A, CPU_C\}$ | 0.03075 |
| {Voter} | 0.02605 |
| $\{CPU_A, DO_C\}$ | 0.01637 |
| $\{CPU_A, DO_B\}$ | 0.01637 |
| $\{CPU_B, DO_A\}$ | 0.01637 |
| $\{CPU_B, DO_C\}$ | 0.01637 |
| $\{CPU_C, DO_A\}$ | 0.01637 |
| $\{CPU_C, DO_B\}$ | 0.01637 |
| $\{PS_1, PS_2\}$ | 0.01590 |

reasonable considering that the difference in failure rates between the two units is very small and the input part of a channel has more redundancy than the output part of a channel.

**Table 4.** Top 11 MCS for the PLC System

## 6.2 Results of the PLC Generalised Stochastic Petri Nets analysis

The results obtained from of the GSPN analysis are the following. From the single channel model, the failure rate of the single channel is *$\lambda$=2.9131 e-6*. From the whole PLC model, the PLC reliability in function of the time, computed by Surf-2, is shown in figure 8.

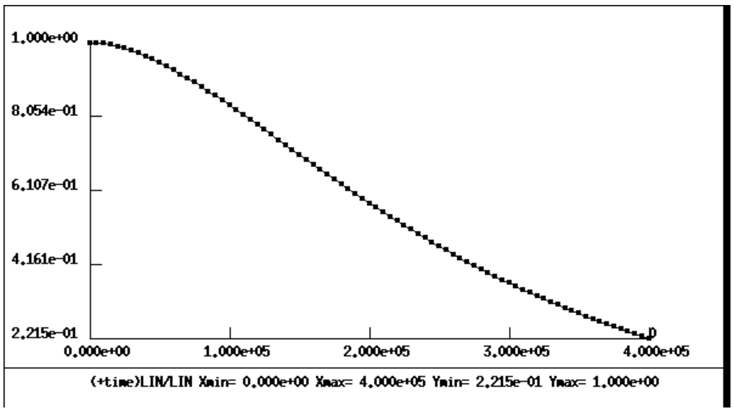


**Fig. 8.** PLC reliability function versus time

*MTTF* is been derived from PLC reliability Function. The *Mean Time to Failure = 6.8729 e+05.* The differences in MTTF results, obtained by GSPN analysis and FTA, are possibly due to different simplifications adopted in PLC modelling , i.e different interpretation of the redundant inter-channel bus.

## 6.3 Results of the PLC Stochastic Activity Networks Analysis

The model of the PLC defined by the Composed Model in Figure 5 has been solved using UltraSAN. The tool generates automatically the stochastic process representing the system behaviour and since some parts of the model are repeated (as the PLC channel), UltraSAN adopts optimisations, obtaining a Markov reduced chain which, in our case, contains 1976 states.

Figure 9 shows the curves of the Unreliability as function of the time. Below the bold line the component satisfies the SIL-2 requirement. The interval of time in which the probability of failure of the PLC is lower than the upper bound of the values of SIL-2 is 3 years and half whereas the *Mean Time To Failure* (*MTTF*) of the PLC results to be *91.1 years* (*798,000 hours*).

As an example of sensitivity analysis to input parameters, Figure 10 shows how the Unreliability at *t=4 years*, varies as a function of the fault rate of the switch components of the *VOTER*. In the entire range considered the unreliability of the PLC exceeds the upper bound for the SIL-2.

It must be observed that the results obtained by the different teams using the diverse modelling techniques and tools, are all of the same order of magnitude.

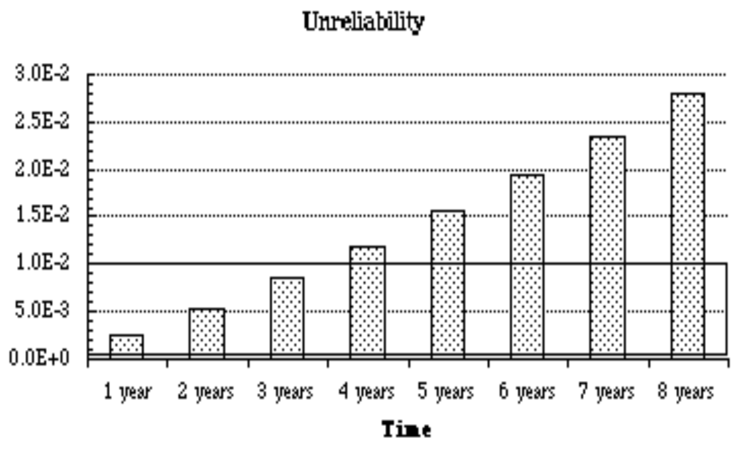


**Fig. 9.** Unreliability versus Time (years)

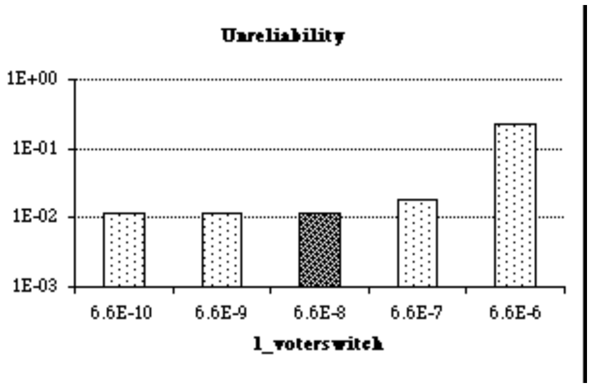


**Fig. 10.** Unreliability at *t=4* years, as a function of the fault rate *l_voterswitch* of the VOTER

## 7. Main remarks on the experience

Along the experience of the PLC dependability assessment, carried out by the diverse teams, differences worthwhile to be described came out in terms of modelling techniques, PLC models and modelling tools. The most important are reported hereafter.

### 7.1 The modelling techniques

FTA has shown to be completely adequate to perform the PLC dependability assessment, including the identification of PLC critical components, due to the very broad modelling assumptions. A weakness of FTA is the fact that events must be considered as statistically independent. State space based approaches overcome this limitation, but require the solution to be computed over a set of equations whose number increases exponentially with the number of components. GSPN allows to reduce the number of model primitives. The PLC system tolerate faults by channel replication and to take into account the replica of identical parts of the PLC, the GSPN model has been developed in two steps, introducing simplifications in modelling not always allowed. SAN allow to deal with model composability. By keeping into account symmetries and using modularity SAN reduce the state explosion problem. Modelling by GSPN or SAN allows to account for several possible deviations from basic assumptions as summarised in table 2. However, when FTA can be used, increases exponentially with the numbers of components. Among the state space based approaches, modelling can be more immediate. For instance, modelling the 2 out of 3 PLC voting policy by FTA is directly obtained by using the 2 out of 3 gate. On the contrary, GSPN and SAN techniques require a specific implementation of such a PLC voting policy, which considers explicitly all the possible combination of the three events. FTA also allows the immediate identification of PLC critical components by the importance of PLC Minimal Cut Sets. That is not immediate in the state space modelling techniques. Criticality of PLC components could be derived starting from structural analysis of Petri Net which allows to check qualitative properties of the model.

## 7.2 The models

The FTA and GSPN models of the PLC assume that each elementary block fails on failure of any of its sub-components. This assumption is conservative and simplifies the models. Analysis at a deeper level of detail is needed to evaluate the impact of such assumption on PLC dependability assessment, as Failure Mode and Effect Analysis (FMEA). FMEA starting from the failure modes of each PLC sub component, evaluates their effects on the failure of the PLC elementary block. Unfortunately the failure modes of each sub component are not available and moreover their effects were not easy to extract by PLC Specifications [1],[2].

The PLC model by SAN partially overcomes the lack of a FMEA, by making the following assumptions. PLC sub-components have self checking capabilities, so that besides the normal faults accumulation, faults may result also in an isolation of the affected component, which induces a degradation of the system less severe than that induced by a non detected fault. Hardware voter redundancy is considered, the voter has some redundancy at the level of its sub-component by which the PLC still delivers correct service despite the failure of one of the voter sub-component provided all the three channels are working properly. Faults of the busses in the channel (either *I/OBUS* and *Inter-channel bus*) prevent the CPU of the channel to communicate with the other components of the system. Thus, the busses and the CPU, have been collapsed together in building the sub-model of the channel.

## 7.3 The tools

The following tools have been applied to the PLC dependability assessment: Sharpe and Item software for FTA; Surf-2 for GSPN; and UltraSAN for SAN. The basic methodology of Sharpe is different from the classical fault tree analysis [7]. Sharpe allows to convert the Fault Tree model into other formalisms, such as Markov Chains and GSPN. The used version of Sharpe has a textual interface so the Fault Tree structure of the PLC, the input data and the required measures, have been input as a sequence of statements. Sharpe does not allow the Minimal Cut Set computation while Item software does. The Fault Tree structure of the PLC has been given in input to Item software by a graphical editor. Numerical values have been entered in generic data models which were then assigned to the primary events of the tree. SURF-2 is strongly oriented to dependability measures. All measures can be derived from the same single model. SURF-2 accepts Markov, Markov with Reward Variables and GSPN models and provides just analytical solution. UltraSAN is based on SAN, a Petri Net extension, and provides both analytical and simulative solutions. It is more oriented to performance and performability measures. Dependability measures have to be extracted from reward variables. The tool allows modular modelling by a composite editor. UltraSAN allows graphical representations of results, by a Report Generator that provides standard interface to standard graphical tools (i.e. Splot, Gnuplot).

## 8. Conclusions

The basic concept of design diversity, that has origins in the software engineering literature (e,g,Littlewood and Miller 1989), is here informally extended to the modelling and analysis of a PLC, with the aims to reduce the impact of errors and ambiguities in specification, to improve comprehensibility of modelling techniques and tools and to increase confidence and belief on models and analysis results. Here, a PLC dependability assessment has been carried out by independent teams, even if a complete isolation was not formally granted. Starting from the same basic assumptions and data, each team has used diverse combinatorial and state space probabilistic modelling techniques, implemented by public tools. Slight differences in the assumptions made have occurred. The usage of different modelling techniques has led to diverse models. Models focused on different system details also due the diverse skill of each team. Diversity in teams allowed ambiguity and errors in specifications to emerge and to be better overcome. Diversity in modelling techniques, has improved assurance that computational models were valid in relation to the modelled system and that results are computed correctly. In fact modelling techniques are themselves complex and there is a problem of understanding whether the derived models are faithful representation of the system and whether the underlined analysis is correct. Moreover there is no sufficient assurance that the tools which implement modelling techniques have been properly validated according to the dependabiliy requirements of the system under consideration. As final conclusion we remark that PLC assessment results obtained by the different teams are all of the same order of magnitude. This is quite important since extreme precision in probabilistic modelling is not required and not believed.

## 9. Acknowledgements

We would like to thank dr. Sandro Bologna who has believed and allowed the research activities, from which this paper has been extracted. A special thank to prof. Andrea Bobbio, who was extremely supportive of the basic work underlining the preparation of the present paper. Moreover we would thank dr. Ivan Mura, eng. Vincenzo Conti and the last year undergraduated student Luciano Tosi, who operatively contributed to the experience. The PLC specification and the input failure data have been extracted from documents of GdL Ispesl, as referred.

## 10. References

[1] GdL ISPESL - Guideline for the valuation and Verification of the Dependability (reliability) of Critical Computer Systems, Part 1: Hardware - June 1995
[2] GdL ISPESL - PLC Safety Critical: Hardware Components Failure Rates, Starting assumptions and required measures - January 1999
[3] M. Malhotra, K. S. Trivedi - Power Gerarchy among Dependability Model types - IEEE Trans. Reliability, Vol. 43 - 1994
[4] Ciancamerla, V. Conti, M. Minichino, L. Tosi - PLC Safety Critical Evaluation - GdL ISPESL Report - August 1999
[5] Bondavalli, S. Chiaradonna, I. Mura - Modelling the Safety Critical PLC - GdL ISPESL Report - July 1999
[6] Item Software - Fault Tree+ : Fault and Event Tree Analysis Program -Isograph Limited – 1998
[7] R. Sahner, K. S. Trivedi, A. Puliafito - Performance and reliability analysis of computer systems, An example based approach using the Sharpe software package - Kluwer Academic Publishers - 1998
[8] Surf-2 User Guide - CNRS LAAS - Toulouse
[9] Ultrasan User's Manual Version 3.0 - University of Illinois at Urbana - Champaign, 1994
[10] Bobbio, E. Ciancamerla, M. Minichino, L. Portinale - Comparing Fault Trees and Bayesian Networks for Dependability Analysis - Safecomp'99 – 18th International Conference on Computer Safety, Reliability and Security – Toulose, France – September 27-29, 1999
[11] E. Ciancamerla, V. Conti, M. Minichino - Dependability Evaluation of Safety Critical systems by Probabilistic Structured Modelling - Automation '99 - Annual National Workshop of ANIPLA - Roma - 24/25 Novembre 1999.