21

Validation and Extension of Fault Management Applications through Environment Simulation

Roberto Manione, Fabio Montanari

CSELT, Centro Studi E Laboratori Telecomunicazioni S.p.A.
Via G. Reiss Romoli 274 - 10148 Torino (ITALY)
Tel. +39-11-2286817, Fax. +39-11-2286862
e-mail: manione@cselt.stet.it

Abstract

Fault management systems are complex applications. Early evaluation of prototypes as well as thorough testing and performance evaluation of the final versions before their deployment are a must.
The present paper presents a simulator of plesiochronous transmission networks, SPRINTER, which has been used to generate test patterns for alarm correlation systems, working on the same kind of networks.
Thanks to the choice of a versatile simulation environment, particularly suited for distributed systems, YES, the implementation of SPRINTER turned out to be elegant and easily extensible.
The approach has been applied to the validation of the alarm correlator SINERGIA; however, the alarm streams generated by SPRINTER could be used to test other correlators working on the same kind of networks.
Furthermore, the proposed simulation approach seems generalizable to other network management applications and areas.

Keywords

Models, Distributed Systems Simulation, Fault Management, Alarm Correlation, Fault Diagnosis, System Testing

A. S. Sethi et al. (eds.), *Integrated Network Management IV*
© Springer Science+Business Media Dordrecht 1995

1 INTRODUCTION

Fault diagnosis of Telecommunication networks is a fairly complex task, mainly due to the interactions among the different network components along the digital paths; as a consequence of such interactions, a number of equipments across the network emit alarms as a consequence of a single fault.

To cope with this alarms proliferation, correlation techniques are used: their purpose is the isolation and diagnosis of the faults starting from the equipment alarms.

A number of approaches have been proposed; among them are SINERGIA [1] [2], which performs rule based correlation and diagnosis using heuristics taken from the network experts, and IMPACT [3], which uses a model based reasoning approach. In both cases the diagnosis system needs to be verified and validated before its deployment with an extensive number of real cases (i.e. not just test data used in the debugging phase).

On the other hand such real alarm streams are not easy to obtain, particularly during the development phase of the Network Management system; the main disadvantages of the use of real streams are that they span over long time intervals (i.e. weeks), hence they take long times for their collection; furthermore generally they do not contain all the kinds of faults over all the kinds of network equipments in all the network topologies which the diagnosis system claims to deal with

A network simulator can instead be used to generate the test alarm streams; such a simulator is also useful when high volumes of alarms are needed to test the ability of the diagnosis system to stand with given alarm throughputs.

A totally different use of a network simulator is the generation of the diagnostic knowledge to be used by the correlator: new topologies, not known to the correlator, can be simulated and the relative alarm versus fault relations extractedfrom the simulation results.

In the following, the above mutually exclusive usages of a simulator will be called Validation and Extension, respectively; both have been experimented with the SINERGIA alarm correlator.

This paper presents a network simulator, SPRINTER (Simulator of Plesiochronous tRansmIssion NeTworks alaRm handling) built for the validation of the fault diagnosis system implemented at our labs, SINERGIA. The structure and the behaviour of the various network equipments, as far as the alarm handling and propagation is concerned, have been coded into a library of equipment models, usable in the composition of the networks.

A significant number of networks have been built out of the equipment models and extensively simulated. The simulator is able to inject given faults over given equipments and to obtain a timed list of the alarms generated all over the network as a consequence of the faults, either in single or multiple fault contexts; SPRINTER is also able to simulate the ceased alarms stream coming from mending actions over the faulty equipments.

The paper is organised as follows: in section 2 the fault diagnosis system under validation is sketched; section 3 presents the simulation environment, while Section 4 deals with the overall simulator architecture; section 5 reports the validation results on SINERGIA and the first approaches to its extension; finally section 6 draws the conclusions.

2 FAULT DIAGNOSIS IN PLESIOCHRONOUS NETWORKS

The goal of a generic fault diagnosis system is to locate faults in the *digital paths* along the transmission network, which are caused by failures of Lines, Line Terminals (T), Multiplexers/Demultiplexers (M) as well as of the trunk interfaces of Exchanges, Digital Cross Connects (DXCs) and other network devices.

CCITT Recommendations G.704 [4], provide a mechanism for the generation and propagation of fault indication signals (in the following referred to as alarms) in the digital

transmission paths, aimed a the easy identification of the faulty equipments; however, almost always, the occurrence of a trouble in one equipment originates alarms from a number of equipments somehow related to it. The main task of the fault diagnosis is to group together all the alarms which are originated from the same physical fault and to find out which equipment needs to be repaired; a more precise diagnosis which identifies the fault diagnosis within the faulty equipment is of course a plus of the diagnosis system. The diagnosis process is not straightforward and sometimes is still carried out by maintenance experts.

In the Italian network the transmission equipments are monitored by proper Mediation Devices, which make the state variables of each monitored equipment available to the diagnosis system. Such variables are in turn driven by the operating status of the equipment and of the digital paths connected to it (as specified in the CCITT recommendations G.704).

Figure 1 pictorially shows of what happens in a real plesiochronous network, e.g. made of Equipments (Multiplexers, M and Line Terminals, L in the picture) and Lines: faults occur from time to time over its components and alarms are generated by the Equipments; in general, different alarms are emitted by a number of equipments in front of any single faults occurring at one equipment of line; alarms are forwarded to a Network Management Center for their correlation, aimed at the isolation of the faulty equipment.

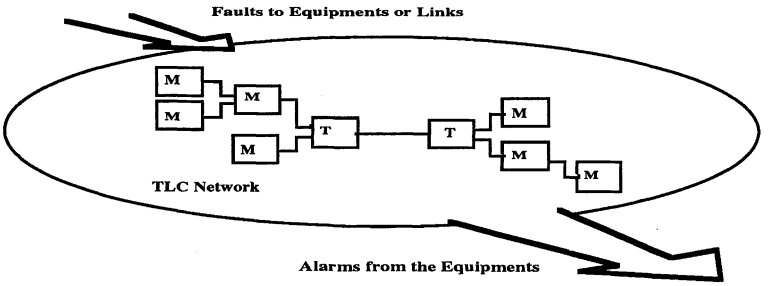


Figure 1 Alarm generation and propagation in Telecommunications Networks

2.1 The SINERGIA alarm correlator and fault diagnostician

The knowledge built in SINERGIA is basically organized as follows: a number of network topologies (e.g. templates) have been selected in a way such as any real network topology can be expressed by instances of such templates; for each template all the feasible alarm patterns have been listed; for each pattern the faulty equipment has been identified, together with the respective fault diagnosis; each template with the associated list of alarm patterns and fault diagnoses has been named Data Sheet; figure 2 shows an example of data sheet.(see [1] for a more formal definition of the templates)

Each fault propagation pattern listed in a Data Sheet holds two different kinds of knowledge: a *Topological Knowledge* of the involved devices, their interconnections and physical characteristics (such as the type of equipment, its bit-rate and its manufacture technique), and an *Expert Knowledge*, derived from the maintenance Experts' experience, regarding the expected alarms for each specific fault. Moreover each pattern embodies the fault diagnosis (i.e. the indication of the faulty device together with the description of the occurred trouble). We can therefore say that the alarm pattern is the fundamental piece of knowledge on which the SINERGIA diagnosis process has been built. Each pattern has been coded as a forward chaining rule; at present about 400 such rules have been encoded into the system.

MPX	TL	TL	MPX	Technique 2Mbit/s CC(N2)	CC(N2C)	FO	PR	Diagnosis
INT	INT	EXT	EXT	X	X			Power Supply
EXT	EXT	EXT	EXT	X				Line Fault or Regenerator
EXT	EXT	·	(-)	X				One way regenerator
(-)	INT	EXT	(-)		X			Tx interface or Power supply
EXT	INT	EXT	(-)	X				Tx interface or Power supply
EXT	·	·	(-)	X	X	X	X	Rx Line term. Interface or Rx MUX interf
·	NUR	·	·	X	X	X		Line regenerator BER > 10^-6
(-)	EXT	EXT	(-)		X	X		Rx interface of Line Terminals
(-)	EXT	·	(-)		X			Rx interface of Line Terminal
INT	URG	(-)	(-)		X	X		Interface
(-)	URG	URG	(-)			X		BER > 10^-3 or Regenerator
EXT	ALIM	ALLRX	(-)				X	Power
(-)	ALLTX	ALLRX	(-)				X	Tx Alignment
(-)	MIR	·	(-)				X	No signal received
EXT	BER	·	(-)				X	BER > 10^-3
(-)	ALLRX	·	(-)				X	No signal received
INT	MIR	·	(-)				X	No signal transmitted
ALIM	MIR	·	(-)				X	Power Supply

Figure 2 An example of Data Sheet

The overall correlation methodology of SINERGIA is built up of two main reasoning steps that implement a sort of *generate and test* paradigm as is depicted in Fig. 3.

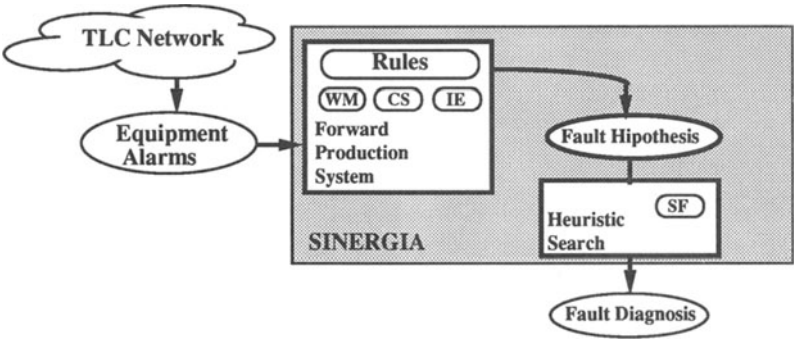


Figure 3 The SINERGIA Architecture

The first step is based on a set of rules (which encode the fault patterns of the Data Sheet) which instantiate fault hypotheses, whilst the second is a heuristic search to determine the best solution among the hypotheses (the fault diagnosis result). In the figure 3 the fundamental blocks of the first step are also depicted. In fact the execution of the rule component relies mainly upon the Working Memory (WM), useful to determine what rules are executable, the conflict set (CS), which contains the executable rules, and the Inference Engine (IE) which governs the whole process.

The rules block works mainly on the alarms collected from the telecommunications network to produce an intermediate result, the Fault Hypotheses set.

The Heuristics Search block selects among the Fault Hypotheses and delivers the Fault Diagnoses, which are the optimal subset of the Fault Hypotheses which best explain, according to a set of criteria, the alarms received from the network; a Scoring Fiunction (SF) is used to rank the Hypotheses Subsets.

Among the more remarkable features of SINERGIA is the ability of its algorithms to exploit the Topological and Heuristic Knowledge, worked out under the hypothesis of single fault, even in case of multiple faults, extending it automatically.

3 SPRINTER MOTIVATIONS AND IMPLEMENTATION CHOICES

A number of needs led us to undertake the implementation of the behaviour of alarm generation across Plesiochronous; they all have been experimented on SINERGIA but are applicable to other diagnosis systems as well:

- to validate the correctness of the knowledge and of the algorithms of Alarm Correlation and Fault Diagnosis systems; in this case the simulation output is the timed stream of alarms coming from the faulty network; such a stream can be sent to the system under test and its diagnosis can be matched against the original faults which was injected in the simulated network;
- to stress Alarm Correlation and Fault Diagnosis systems with heavy load conditions; in this case a low MTBF is specified for the equipment models, in order to obtain time-dense alarm streams from the simulation ;
- to extend the knowledge of rule based Alarm Correlation and Fault Diagnosis systems; in this case the simulation output is the list of all the alarm streams coming from a given (hopefully small) network topology when all the applicable faults are injected, one at a time; this is particularly useful when new alarm correlation rules are to be inserted into the diagnosis system: the simulator can supply the "expert knowledge", once taken from the experts.
- to train the network operators on the alarm correlation task.

With the above needs the main requirement was that the simulator was not "wired" into a monolithic program, but could instead be extended and modified; furthermore, every feasible network topology which could be implemented out of the known equipment types had to be easily described within the simulator.

The main requirements for the simulator were:

- to know about the behaviour of its equipment types; any number of equipment of each type could be used within a given simulated network;
- to allow for the modelling of any network topology which is feasible in the real world;
- to simulate the given network model for a given amount of simulated time: within this interval a number of faults (either given or randomly chosen among the legal ones, for the various equipment types) will be injected; furthermore, after some time (either given or random) the proper ceased alarms will be generated, simulating the mending of the fault;
- to allow for the modelling of new equipment types at any time.

In order to meet the above requirements it was chosen to keep the network models as close as possible to the reality: each equipment was modeled by a Finite State Machine (FSM) and each digital link among the equipment was modeled with a channel. In this way any equipment models has the same interfaces of the respective real equipment and can be interconnected following the same rules.

The chosen simulation environment is YES (Yet another Event driven Simulation environment) [5], developed in CSELT for the functional simulation and for the performance

evaluation of generic distributed systems; the simulation language of YES is PROMELA+, a CSELT extended version of the PROMELA language, originally defined at AT&T Bell Labs [6], in turn based on the Hoare's CSP.

The atomic entity of PROMELA+ is the Process, which allows for the modelling of FSMs; processes can communicate either asynchronously or synchronously by means of Channels.

The implementation of SPRINTER within YES turned out to be fairly elegant, since a TLC network model became a distributed system, each equipment became a process and the network topology was represented as a network of channels linking the processes; in this way the behaviour of the equipment could be precisely inserted into the models.

Figure 4 shows the architecture of SPRINTER, based on the library of Equipment models; the fault sequence can be either given or randomly generated among the legal faults for any equipment model.

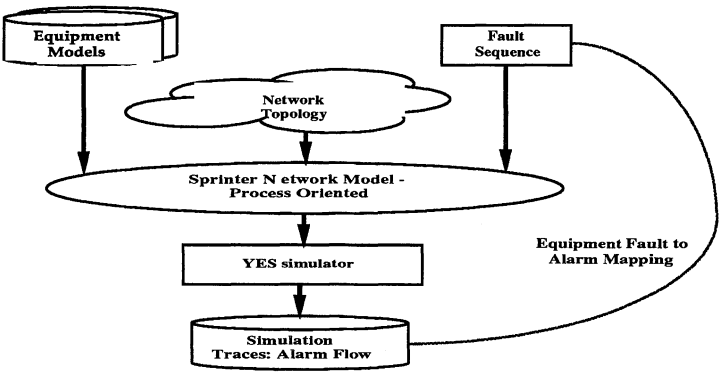


Figure 4 The SPRINTER Architecture

4 THE NETWORK SIMULATOR

The purpose of SPRINTER is to model the behaviour of real generic plesiochronous networks, with respect to the alarms; alarms are generated by individual equipments as a consequence of fault conditions due to internal damages as well as damages occurred over the physical connections among the equipments, as already shown in Figure 1.

4.1 Modelling aspects

SPRINTER models both the structural and the behavioural aspects of the transmission equipments; however, since only the fault management is of concern, only the alarms sent via the alignment and signalling frames are simulated, while the payload streams are not taken into account.

On the structural side, each equipment is further partitioned into its main subfunctions; a subfunction is defined as a module which can be characterised from the fault management point of view by a boolean working state variable (either Working or Out_of_order) and the working state of the module conditions the working state of the whole equipment.

The following functions hold among the internal state S of an equipment (e.g. the state of the FSM implemented by the equipment) and all the fault conditions which can occur:

$$\mathbf{W_e} = w_e(\mathbf{S_e}, \mathbf{M_e}, \mathbf{Li_e}) \quad (1a)$$
$$\mathbf{S_e}(n) = \mathbf{F_e}(\mathbf{S_e}(n\text{-}1), \mathbf{M_e}(n), \mathbf{Li_e}(n)) \quad (1b)$$
$$\mathbf{Lo_e} = l_e(\mathbf{S_e}, \mathbf{M_e}, \mathbf{Li_e}) \quad (1c)$$

where:
S_e is the equipment internal state, boolean vector
W_e is the equipment working state
M_e is the module working state, boolean vector
Li_e is the input link state, boolean vector
Lo_e is the output link state, boolean vector
F_e is future state function, boolean vector

Figure 5 shows, as an example, the structural partitioning of a Line Terminal equipment; the header of the model of such an equipment, with parametric bit-rate, is:

```
proctype TL_8_34_140(chan mux_in, mux_out, lin_in, lin_out, bit_rt, ...)
```

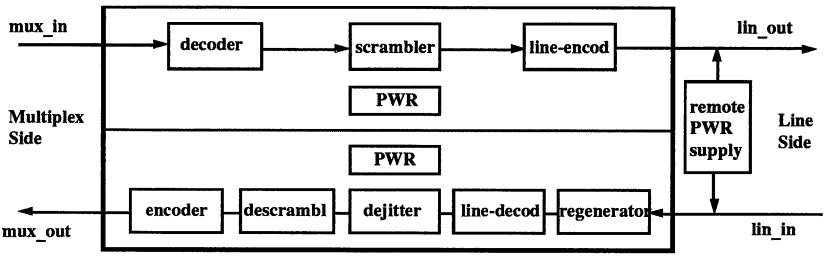


Figure 5 Structural partitioning of a Line Terminal Equipment

On the behavioural side, as stated in (1), each equipment has two main sources of stimuli and two main outputs: figure 6 shows a causal graph of the four entities.

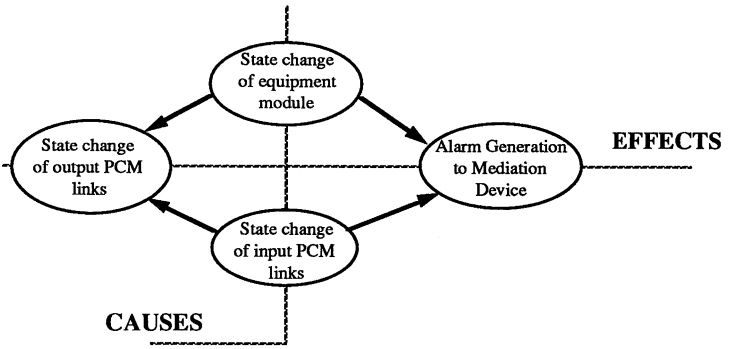


Figure 6 Behaviour model of equipments

The equipment models wait for changes in either the two sources of stimuli; as soon as a

new stimulus is received, it updates the internal working state and send the appropriate alarms and signals over the respective outputs, according to what stated in (1).

4.2 Implementation techniques

The characteristics of the YES simulation environment, and particularly of its language, allowed to easily model the transmission networks, keeping models closely adherent to reality.

Every equipment type has been implemented as process template. In the description of a generic network each process model can be instantiated several times to create the different equipments of the same type. Equipments are distinguished one from the other by means of an identification number unique within the network.

PCM links among equipments are modelled with channels; a particular message structure has been defined to represent the relevant information contained in the PCM alignment and signalling frames, according to the CCITT Recommendations (see [4]). Typically they are the AIS (Alarm Indication Signal) and ATL (alarm indication to the remote end) signals.

Due to efficiency reasons, while in the real networks messages are sent continuously within the PCM frames, until ceasing of the cause, in SPRINTER each message is sent only once; no messages are sent across channels until conditions are to be propagated; in this way only the differences among messages are effectively sent.

However, in spite of the reduced message number, message handling in SPRINTER is generally more complex than in reality, since the Plesiochronous transmission technique allows equipments to be transparent to frames of lower hierarchical levels.

For this reason a routing algorithm has been implemented into the equipment models which composes and decomposes the frames for the tributaries, simulating the operations of multi-demultiplexing: in order to allow this every message has been structured into fields containing information about the hierarchy of the message and the tributary number for every possible hierarchy level.

When a multiplexing process receives a transit message from a tributary, it tags it with the tributary number and forwards it to the next higher hierarchical level. Viceversa, during demultiplexing, messages arriving from the composite channel are routed down to the right tributary using the tag contained in the proper field of the message.

The equipment FSM models all the faults and signals processing functions built in the real equipments, as required by the CCITT Recommendations: in particular the generation of alarms to the Mediation Device and signals to the adjacent equipments.

Faults and fault ceasings can be injected to each subfunction of each equipment; when a fault is injected the target equipment updates its state vector accordingly and takes all the appropriate actions; then Grouping and Filtering are performed in order to handle multiple faults: a fault condition may be Masked by the contemporary presence of another fault. Eventually alarms are evaluated and emitted to the outside: a dedicated process (i.e. the "Mediation Device") gathers all the alarms and stores them; Figure 7 shows the above process.

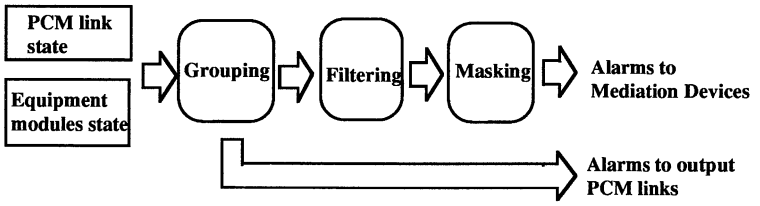


Figure 7 Intermediate steps for alarms generation

4.3 Fault simulation with SPRINTER

The main blocks of knowledge within the simulator is the library of equipment models; however other auxiliary modules are available, which simplify the task of modelling a network: the fault generator, the fault mender and the alarm collector.

The task of modelling a network consists in the instantiation of the equipments and the PCM links connecting them; explicit faults can be specified or in alternative equipments MTBF and repair frequencies can be specified as well.

With the above model the SPRINTER simulates the fault behaviour of the network over a given period of simulated time. A typical structure of a network model is shown in Figure 8; in this case the real network reported in Figure 1 is modeled.

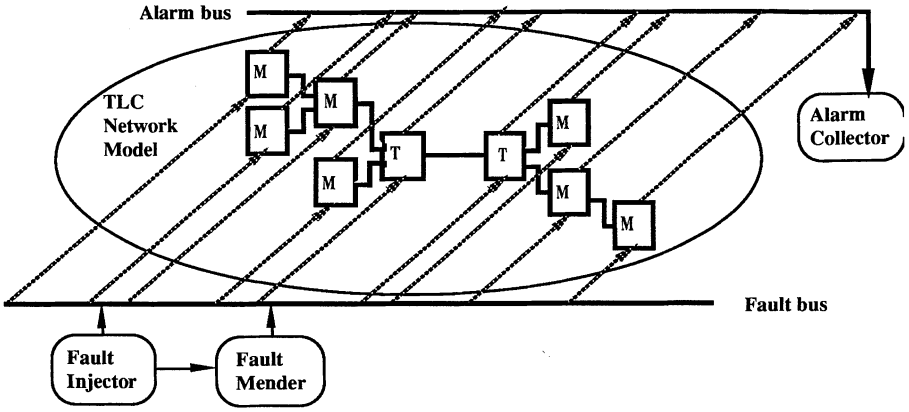


Figure 8 A TLC Network as modeled within SPRINTER

Each equipment shown on the figure 8 is the model of the respective real equipment; all the links wich connect the real equipments are represented in the model, in order to reproduce exaclty the real network topology. Two particular channels classes are highlighted in the picture: the *alarm bus*, by means of which equipment alarms are collected and the *fault bus*, by means of which faults and ceased faults are injected into selected equipments/links; while the former appear also in the reality, although in a less standard form, the latter is obviously used for simulation purposes only: a fault generator process creates faults and injects them through a bus common to all the equipments of the network.

The fault generator works with arbitrary networks; it sends faults not only to the network equipments, but also to a mender process which in turn produces, after a random delay the fault ceasings; working on the distribution of this delay the desired average number of faults present in the network at a time can be obtained; the fault collector simply lists alarms on a file

4.4 Validation of the SPRINTER equipment models

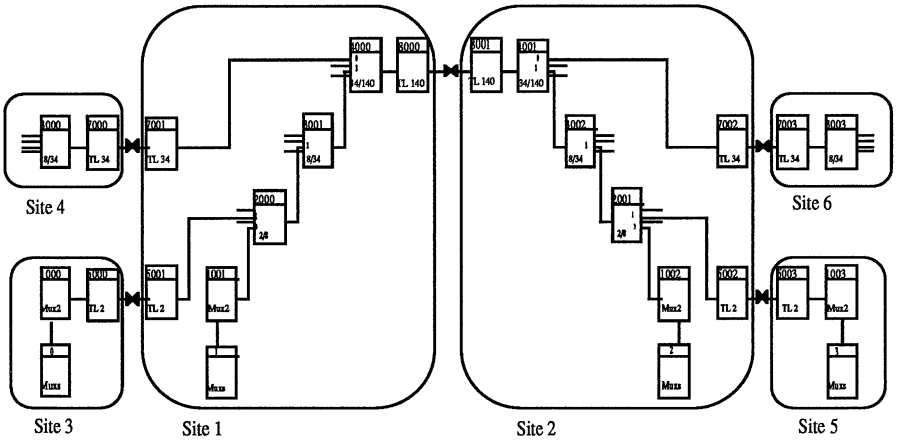
Several tests have been performed on SPRINTER results, at the purpose of validating the model library; the SPRINTER generated alarm traces have been matched against the alarm streams of real network portions; various network topologies have been simulated and large number of faults have been injected on their real equipments; the alarm streams generated by SPRINTER have been compared with the alarms generated by the real test networks. They have shown substantially equal results.

The faults injected in the real equipments were restricted to power supply faults and equipment link faults, because of the lack of controllability of the working state of the real equipments; i.e. the injection of an internal fault could only be done physically acting on the interior of the equipments.

5 VALIDATION AND EXTENSION OF A FAULT MANAGEMENT SYSTEM

5.1 Validation of SINERGIA

SPRINTER has been used to test SINERGIA: a number of networks have been simulated and the simulation results have been submitted to the alarm correlator and diagnostician; SINERGIA outputs have then been matched against the original faults injected into the equipments in the SPRINTER model.




```
Fault 34 on eqp  N. 5003   @ time 4
Equipment  N. 5003    EXT = 1 @ time 4
Equipment  N. 1003    IND = 1 @ time 4
Equipment  N. 3     IND = 1 @ time 4
Equipment  N. 1000    IND = 1 @ time 4
Equipment  N. 0     IND = 1 @ time 4
Fault 6 on eqp  N. 1002   @ time 46
Equipment  N. 1002    INT = 1 @ time 46
Equipment  N. 2     EXT = 1 @ time 46
Equipment  N. 2001    TRIB3 = 1 @ time 46
Equipment  N. 1001    IND = 1 @ time 46
Equipment  N. 1     IND = 1 @ time 46
Fault 45 on eqp  N. 7002   @ time 68
Equipment  N. 7002    EXT = 1 @ time 68
Equipment  N. 7003    EXT = 1 @ time 68
Equipment  N. 3003    IND = 1 @ time 68
Equipment  N. 3000    IND = 1 @ time 68
Repaird Flt 6 eqp N.  1002 @ time 120
Equipment  N. 1002    INT = 0 @ time 120
Equipment  N. 2     EXT = 0 @ time 120
Equipment  N. 2001    TRIB3 = 0 @ time 120
Equipment  N. 1001    IND = 0 @ time 120
Equipment  N. 1     IND = 0 @ time 120
Fault 5 on eqp  N. 1000   @ time 162
Equipment  N. 1000    EXT = 1 @ time 162
Equipment  N. 1000    IND = 0 @ time 162
```

```
LEGEND:
Fault 34 : BER > 10^-3
Fault  6 : Power Supply
Fault 45 : Line Interruption
Fault  5 : Receive Interface
```

Figure 9 A network example and its SPRINTER log

Figure 9 shows on its top a small network example made of 26 transmission equipments; below a part of the SPRINTER generated Alarm Trace is listed.

The test session run on SINERGIA has shown the substantial correctness of the knowledge about its correlation rules and of the algorithms which exploit it in the diagnosis process.

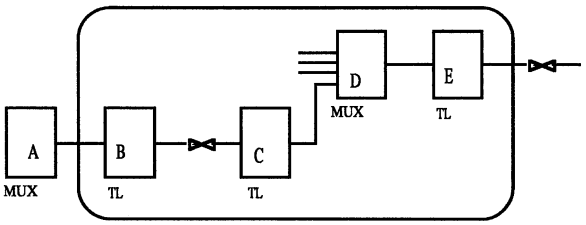
However, being the Data Sheets produced by human Experts, they could have been somehow wrong and/or incomplete: actually the test session reported one entire missing Data Sheet and 10 missing/wrong rules on known Data Sheet, out of over 400 rules.

5.2 Extension of SINERGIA

SPRINTER has also been used to extend the rule base of an alarm correlator and diagnostician: actually it has been used to generate in advance all the alarm configurations over a given network portion, associating each of them with the fault which caused it.

This process has been exploited in the generation of the missing SINERGIA Data Sheet and in the refinement of the existing ones; in an automated way.

The early results of such experiments are as follows: the TTMT data sheet, the one whose lack was pointed out by the SPRINTER generated test suite has been automatically generated, defining a small network representing the desired topology plus some more equipments at its borders. The TTMT subnetwork, shown of figure 10, has been simulated exhaustively and 36 faults have been injected, in about 15' CPU time on a SUN SPARCSTATION 20; out of the faults 15 different rules have been extracted.



A	B	C	D	E	Diagnosis
(-)	EXT	EXT	-	-	LINE between B and C
(-)	-	EXT	-	-	ERRH,NORXL over C
-	-	NURG	-	-	ERRL over C
(-)	-	INT	-	-	NORXM,DECM,DECL,DESC over C
(-)	INT	INT	-	-	SCR over C
(-)	EXT	INT	-	-	CODL,ALTX over C
(-)	-	INT	TRIB0	-	CODM,ALRX over C
(-)	EXT	INT+ORTAL	-	-	TAL over C
(-)	-	-	EXT	-	FAT,NORX over D
(-)	-	-	TRIB0	-	NORX0,OVTX0,OVRX0 over D
(-)	-	-	INT	-	ALD over D
(-)	-	-	INT+(-)	INT	ALM over D
(-)	-	INT	TRIB0	-	AL0 over D

Figure 10 The TTMT topology analyzed with SPRINTER

The above table reports the TTMT Data Sheet generated by SPRINTER in the case where the E bit rate is 34 Mbit/s.

As a concluding remark about the extension on the knowledge base of an alarm corelator it must be remarked that the new knowledge, derived by network simulation cannot be validated with simulated alarm streams, since possible errors in the equipment models could affect both the knowledge and the test cases, preventing their capture.

6 CONCLUSIONS

Fault management systems and particularly alarm correlators and fault diagnosticians are complex Network Management applications. The creation of a comprehensive functional test suite is not straightforward, since a very deep knowledge of the networks and their equipments is needed; furthermore, even in that case, manually generated test suites cannot guarantee requested coverages.

With the right choice of the simulation environment, fault simulation of networks has proven an effective approach to the test suites generation, for both the functional and performance point of view. Furthermore the same tool has also proven effective in the correction/extension of a fault management application.

The results obtained with SINERGIA confirm the effectiveness of the proposed approach; however SPRINTER could be used virtually without modifications to validate other correlation systems working on plesiochronous networks.

With the encouraging results on the fault management we think that applications belonging to other areas of network management could also take advance of simulation techniques for the modelling of the environment in which they will operate.

REFERENCES

[1] S. Brugnoni, G. s, R. Manione, E. Montariolo, E. Paschetta, L. Sisto, "*An Expert System for Real Time Diagnosis of the Italian Telecommunications Network*", Proc. of ISINM '93, San Francisco, CA, April 1993.

[2] R. Manione, E. Paschetta, "*An Inconsistencies Tolerant approach in the fault diagnosis of telecommunications networks*", Proc. of NOMS '94, Orlando, FA, February 1994.

[3] G. Jakobson, M.D. Weissman, "*Alarm Correlation*", IEEE Network, Nov.93 pg 52-59.

[4] CCITT Recommendations, "*Digital Networks Transmission Systems and Multiplexing Equipment*", G.701-G.941, Yellow Book, Vol. III - Fasc. III.3, Geneva 1981.

[5] E. Chiocchetti, R. Manione, P. Renditore, "*Specification based Performance Evaluation of Distributed Systems for Telecommunications*", (Short Paper), 7th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation, Vienna, 1994.

[6] G.J. Holzmann: "*Design and Validation of Computer Protocols*", Prentice-Hall Int., 1991.

AUTHORS BIOGRAPHY

Roberto Manione graduated in EE in 1983 from Politecnico di Torino. Since then he is working in CSELT; his research interests were formerly in the field of Silicon Compilation, when he was involved in National and European research projects and authored several international publications; since some years he is working in the Network Management field and is project leader in the development of various tools aimed at the functional and performance validation and testing of distributed Network Management systems.

Fabio Montanari graduated in EE in 1994 and has worked in the SPRINTER project for the development of his thesis and after his graduation.