Modeling Fault Propagation in Telecommunications Networks for Diagnosis Purposes

A. Aghasaryan[*], **C. Dousson**[**], **E. Fabre**[***], **A. Osmani**[****], **Y. Pencolé**[***]

[*] Alcatel R&I – Route de Nozay – 91460 Marcoussis, *armen.aghasaryan@alcatel.fr*

[**] FTR&D – 2, av. P. Marzin – 22307 Lannion, *christophe.dousson@rd.francetelecom.com*

[***] IRISA – Campus de Beaulieu – 35042 Rennes, {*fabre,ypencole*}*@irisa.fr*

[****] LIPN – Av. J.-B. Clément 93430 Villetaneuse, *ao@lipn.univ-paris13.fr*

Abstract: This paper describes a formalism to model the behavior of telecommunications networks when a fault occurs and how the effects are propagated across equipment. The objective of such a formalism, which is derived from UML diagram sequences, is to ease the construction and the update of a model corresponding to the supervised network; this model can be used to simulate fault propagation in the network but also to process on-line diagnosis and determine primary causes of a set of observed alarms.

Keywords: network management, modeling, UML, distributed diagnosis.

1 Introduction

Telecommunications networks are growing in size and complexity, which means that a bigger and bigger volume of notifications needs to be handled by the management system. Most of this information is produced spontaneously by equipment (e.g. status change and malfunction detection) and this message flow must be pre-processed to make effective management possible. Filters based on a per-notification basis fail to perform an adequate information preprocessing required by human operators or by management application software, which are not able to process such amount of events. One must decrease this information stream by suppressing superfluous notifications and/or by aggregating relevant ones. In addition, the problem of expertise acquisition still remains: How to feed the filtering system? Which aggregation rules are relevant? Where is the necessary information available? And how to express it? It is a very hard and very long work to collect knowledge from experts and, the expertise thus gathered is insufficiently generic: evolution of networks is so fast that knowledge rapidly becomes obsolete. Moreover, experts don't exist for new systems as experts need a long time to acquire this knowledge. To deal with that, the MAGDA project proposes a modeling methodology which this paper is devoted to.

The last key point is how to enlarge the area of all these techniques in order to apply them to a real (and huge) telecommunications network. The current method for this relies on an administrative partition of the network management task (depending on client classification, equipment locations, and so on.). To conform with the scalability requirement, MAGDA must take into account the intrinsic distribution of telecommunications network and its hierarchy. We propose to allow to distribute correlation and diagnosis algorithms in order to make the supervision system scalable. So, the modeling methodology must allow hierarchical representation and algorithms will be able to be distributed among equipment and to cooperate.

This paper is organized as follows. Section 2 describes the applicative context of the project and the MAGDA experiment which was carried out during the project. Sections 3 and 4 define the formalism used by MAGDA in order to model the supervised network. Then, section 5 is devoted to an overview of the distributed algorithms for the MAGDA diagnoser. Finally, we conclude and prospect on future work to make MAGDA operational.

2 Overview of the MAGDA Experiment

In the scope of MAGDA project, we have studied the Synchronous Digital Hierarchy (SDH) telecommunications networks [1], where the communication between equipments is done *via* virtual synchronous connections. The management task of connections is ensured by a set of objects located at the Managed Information Base (MIB) of the corresponding physical equipment. In particular, we consider a ring topology shown on the figure 1 (left).

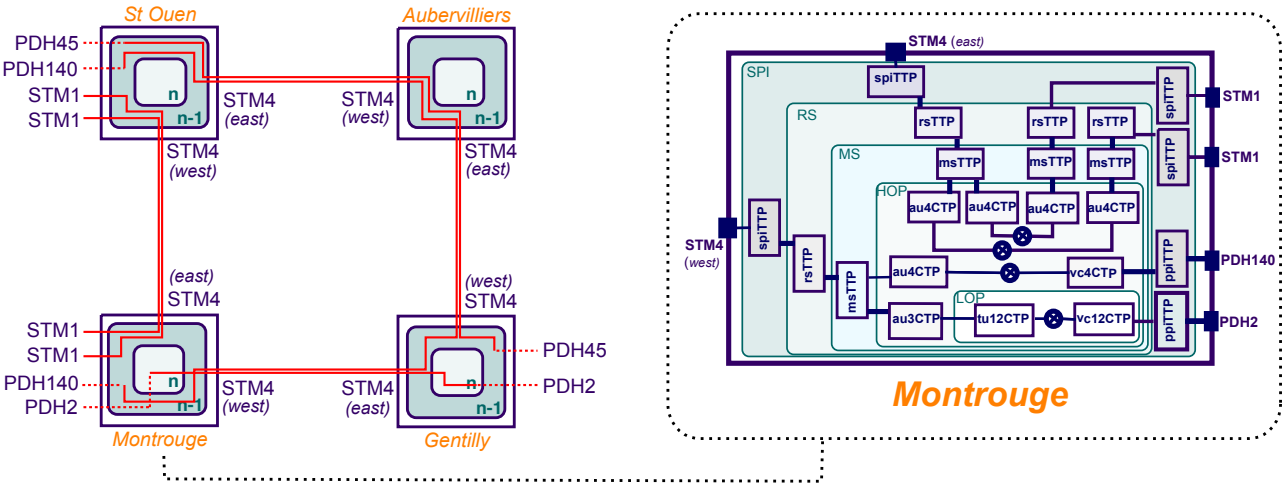

Figure 1: *Topology of the modeled network (left) and components of Montrouge ADM (right)*

We are interested in faults and their effects which are propagated across SDH layers on a given equipment (vertical propagation), and among equipments by following physical or logical connections (horizontal propagation). The figure 2 shows an example on the physical layer, a laser breakdown: (i) the emitter detects the problem and generates a Transmission Fail (TF) alarm, (ii) the right hand receiver detects the absence of signal and generates a Loss of Signal (LOS) alarm, (iii) the Automatic Laser Switch (ALS) stops the emitter which in its turn generates a TF, and finally (iv) the left hand receiver generates a LOS. All these steps are accompanied by vertical propagations, i.e. the higher level components will also detect a problem and will issue alarms. As a result, all the components become disabled and only an analysis of the causality between alarms could determine the primary cause: this is the purpose of the MAGDA diagnoser.

The result is a distributed generation of an amount of alarms which are difficult to explain. Consider as an example the breakdown of a laser (see figure 2): the emitter detects the problem, emits a TF alarm to the up level. Then the peer receiver detects that there is no signal (induced propagation of the phenomena) and sends a LOS. At the same time, the ALS stops the emitter which sends a TF. The peer detects also that the signal disappears ans so send a LOS. In this example, notice that all the components become disabled and only an analysis of the causality between alarms could determin the primary cause: this is the purpose of the MAGDA diagnoser.

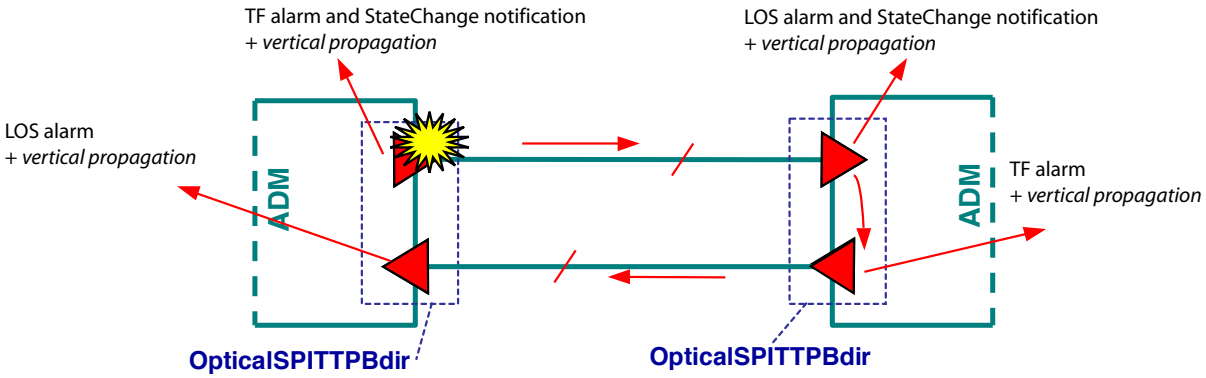

Figure 2: *Propagation of an initial fault occurring on the laser.*

3 Topology and Structural Modeling

Our model simulates exchanges of messages (alarms and notifications) related to the communication objects in the network only at the management level [2]. The objects in the MIB are structured according to the type of service which they ensure, there are Managed Objects (MOs) for breakdowns and protection and reconfiguration, and so on. In our model, we represent only (some part of) the objects allowing the management of breakdown situations. We define elementary components which reproduce the MO behavior and contain information on the fault interactions between components. These MOs do not contain information on virtual connections and network topology; in fact, his information is characterised by the position in the timeslot in temporal order. In real networks, management information is conveyed in the headings of the frames, which circulate in the network. In our model, we define elementary components which reproduce the MO behaviour and contain information about the type of propagation and about virtual connections. This section describe the structural model which is composed of a static structural part (physical network topology and network equipment) and a dynamic structural part (connection model) [3, 4]. The model consists of a set of interconnected components. Each component consists of a collection of elementary components (which will be detailed in section 4).

Physical network topology. The physical network topology is defined by a set of network equipments and their physical links, see the figure 1 (left). During the construction of the topological level, network equipments must be identified as instances of the model library (e.g. "$STM4$-$1651SM$" for a type of Add and Drop Multiplexer (ADM)) which implies that the properties (the number of affluents, the capacity, etc.) of each equipment are known. This allow to instantiate the respective elementary components at the next step of model construction.

Network equipment. Each function of a network equipment is represented in the local MIB by a MO. For example, the establishment/deletion of a virtual connection corresponds to the creation/removal of MOs in the MIB. Any change in the equipment generates notifications for the network manager.
The model of physical equipment takes into account only those MOs which play a role in the production of alarms or notifications or in the propagation of faults.
The components of a network equipment are shown on the figure 1 (right). Each box corresponds to an elementary function, with its own (fault) state attributes and transitions. These components are interconnected in a hierarchical structure based on two kinds of relations: (i) containment relation (across SDH hierarchical levels, e.g.. msTTP contains au3CTP and au4CTP, and (ii) cross-connect relation (on the same hierarchical level, e.g. between two au4CTPs).
Observe that a network element is composed of several instances of the same function (e.g. au4CTP), hence the modeling effort is limited to these generic components. The model itself is obtained by

creating instances of these components and by connecting them, which can be done automatically. The fault behavior of a network equipment is obtained by composition of behaviors of its components described in the section 4.

Connection model. In addition to the above steps of model construction, one shall describe the logical connectivity provided by the network, see for example the end-to-end connections in the figure 1 (left). In SDH, this information is characterized by the position in the time-slot of the SDH frame. So, the distant peers (e.g. two au4CTPs on “Montrouge” and “Gentilly” ADMs, respectively) can be identified simply by carrying the same position number in their identifier. The interaction of distinct elements like “Montrouge” and “Gentilly” ADMs is done simply by sharing variables corresponding to their physical level (SPI), and also by identifying in each connection (au4CTP) the name of its peer.

4 Behavior Modeling: Elementary Sequence Diagrams

In this section we describe the fault behavior of model components with the help of UML sequence diagrams. For this purpose, we introduce the notion of Elementary Sequence Diagram (ESD) that represents the elementary behaviors of components reproduced in different fault scenarios; these entities are the building blocks of the fault scenarios and they correspond to the notion of *tile* in the Viterbi-based diagnosis algorithm elaborated in the scope of the project. The advantage of such tile-based modeling with respect to more classical automata-based approaches is that here the exhaustive knowledge of the global objet behavior is not required.

4.1 Definitions

The following figure 3 illustrates the definition of an elementary sequence diagram. Some notations used below do not exist in the current version of UML, although they can be expressed in terms of UML, in expense of the compactness of the representation, see [5].

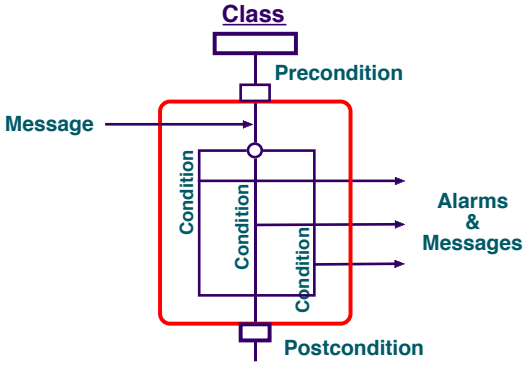

Figure 3: An Elementary Sequence Diagram.

Below are some definitions used in the figure 3.

- The top box (class) represents a MO and the vertical axis is its life-line (top-down); only the events on the same vertical line are ordered in time.
- The external rectangle represents the interface of an elementary sequence diagram.
- The point of life-line entry into this interface can be marked by a rectangle labeled with an applicability precondition. The precondition is a logical expression on the class attributes (state variables) which must be satisfied for the diagram to be applicable.

- Similarly, the point of life-line exit from the interface can be marked by a rectangle labeled with a postcondition of diagram application. The postcondition specifies some part of the object state after the application of the diagram.
- A circle on the life-line represents a logical OR (non-exclusive) expression on parallel (concurrent) life-lines which follow the circle. Each of them can have its applicability condition.
- The arrows leaving the interface represent the sent messages and alarms, the arrows entering the interface the messages received. At most one in-going arrow is allowed.

In our model, we distinguish explicitly between *alarms* and *fault propagation phenomena*. Once a problem is detected the respective alarms are generated and communicated in some way to the network manager, however they do not circulate between different network components. On the contrary, the propagation of faults from one component to another is hidden from the manager's view. Consider the example of figure 2, if the faulty laser stops signal emission, then the remote receiver detects the absence of signal and generates a LOS; the alarm itself is not sent between components. We model these propagations between different components with exchange of *messages*.

Notice that we do not need to consider *all the alarms* in our model since some of them could be filtered: in this case, we can delete them from the diagrams. Moreover, if we keep them in the model and filter them in the network, the proposed algorithms in section 5 are able to deal with missing alarms which are considered as lost. The only drawback is that the diagnostic result could be more ambiguous if the discrepent alarms are lost.

This remark motivates the introduction of two communication primitives for the output arrows.

- $alarm(<alarm\_name>)$ - send an *alarm* (or a notification)
- $send(<message\_name>)\ \ to <direction>$ - send a *message* to another network component. The message can represent implicit fault propagations like in the above example, as well as propagations carried out by the automated mechanisms in the management systems. The expression of message sending contains a constraint $to < direction >$ which is to specify the gate of communication in the component model. In fact, following the architecture of SDH hierarchy, a managed object of layer N can send messages to (i.e. impact in some way) 1/ the layer N-1, 2/ the layer N+1, or 3/ the *peer* object of the layer N linked by a cross-connection.

  A similar notation can be adopted for input arrows: $receive(<message\_name>)\ \ from < direction >$. However, for the sake of simplicity the word 'receive' can be omitted as no ambiguity exists at the reception (no alarms, only messages).

4.2 Local phenomena to be modeled

We classify all the possible faults into two categories: *primary causes* and *secondary causes*. The visible part of the system is described with *detectable phenomena*.

Primary causes. A primary (or initial) cause is a software or hardware fault which occurs spontaneously within some component, independently of faults elsewhere in the network. The localization/identification of the primary causes and their comprehensible presentation to the operator are the principal objectives of the diagnosis process. In addition, these faults require reparation actions.
Examples: a fault on the optical receiver/laser, a cut of fiber ...

Detectable phenomena. In general, a fault is not observed directly, but through some detectable phenomenon that it causes. For example, a problem on the optical receiver can be detected through

the power level observation. The detection of such a phenomenon is followed by generation of alarms, and possibly, by propagation of its effects on other components (or inside the same component). This propagation, of course, remains invisible to the operator.
The entire set of detectable phenomena within a network equipment is determined by the deployment and sensitivity of sensors. It is important to note that a one-to-one correspondence between the faults and the detectable phenomena does not necessarily exist.
Examples: an absence of signal on the receiver (alarm LOS, and propagation through higher levels) or on the laser (alarm TF, etc.) ...

Secondary causes. The major part of the ambiguity in the interpretation of alarms, and hence the need for elaborated diagnosis approaches, come from the fact that some *local* phenomena are different from primary causes but have as a consequence the same detectable phenomena; we call them *secondary causes*.
Example: a remote laser fault will cause, by propagation, the absence of signal on the receiver of a local component; locally, this will have the same effect as if the receiver itself was in fault.

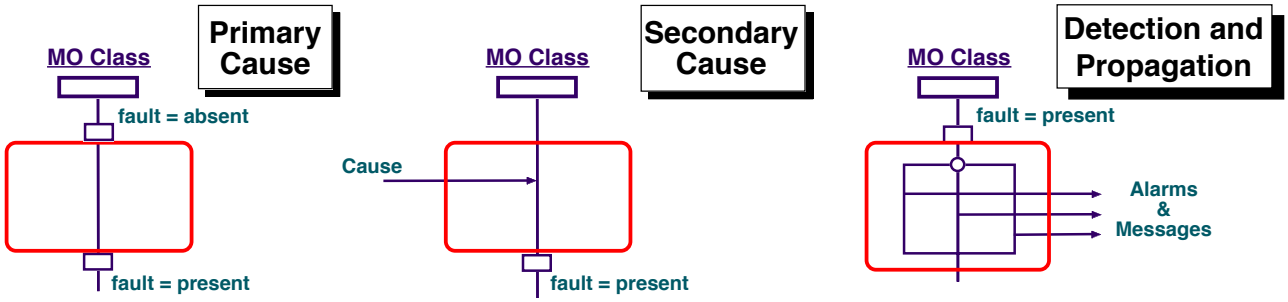

Figure 4: Representation of the three types of local phenomena by ESDs.

The figure 4 shows that each of three categories of local phenomena can be represented by its typical ESD. So, to each primary and secondary cause one can associate an ESD the application of which establishes a certain attribute value, $fault = present$ in the postcondition. As a primary cause can be triggered spontaneously, its diagram can be applied without expecting an external input, but whenever the fault is not yet active (see the precondition). By contrast, the diagram representing a secondary cause will expect reception of an external input, namely a message. [1].

Further, the value of the postcondition set by primary and/or secondary causes matches the precondition of the detection diagram that represents the corresponding detectable phenomenon. So, whenever this precondition is satisfied, alarms and propagation messages are generated.

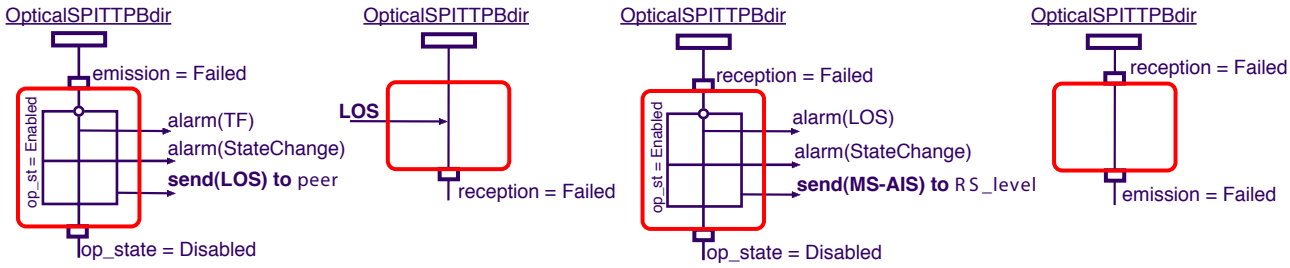

Figure 5: *Examples of ESDs representing the fault scenario of figure 2. The ESD that triggers the initial fault ($emission = Ok \rightarrow emission = Failed$) is omitted.*

By transitivity, the secondary causes can be explained by other secondary causes, and so on until

[1] Whenever the secondary cause is explained by a local fault (as in the example of ALS), the representation of the causality by a message transmission may be redundant as it can be done simply in terms of precondition.

the initial cause(s). These relations define partial orders with primary causes as maxima. Due to the description of fault behaviors in terms of primary and secondary causes, we avoid the exploration of these partial orders in the modeling phase; note that the model is non-deterministic and many partial orders are possible, each one representing a particular fault propagation scenario. The chosen approach allows to construct these partial orders on-line along with the observation of alarms, by connecting the ESDs like tiles of a puzzle.

In conclusion, we achieve an easy model maintainability due to the modular approach in structural as well as behavioral parts of the model. Furthermore, as explained in [5], the ESDs support genericity: (i) for MOs of the same class, (ii) for MO classes of a given SDH layer, and (iii) for messages of a given type. This is why hundreds of MOs in the ring example are described with about 30 ESDs.

5 Distributed Diagnosis Algorithms

Given a set of transitions describing the behavior of the system (the elementary sequence diagrams of section 4), and given some observations raised by transitions of that system (the "alarms"), the diagnoser must find out what are the possible runs of the system that could explain the observations. Some particular events in these runs, the primary causes, can then be proposed as possible explanations. Here, designing a global automaton for the entire network is unrealistic: the number of possible states of this system quickly explodes with the number of network elements. We advocate instead to explicitly use the distributed nature of the model we consider: a telecommunications network is composed of complex elements which interact on a limited set of events. It is thus appropriate to solve the diagnosis problem "by parts". Coming back to the example of figure 1, the network is composed of four elements. Each one has a private clock and evolves independently of the others, except for synchronisation on special events: the emission or reception of messages through the optical links which connect them. We thus propose a distributed diagnosis process which parallels the structure of the network: a supervisor in charge of each element, and cooperating with the supervisors of neighboring elements. This idea is further justified by the fact that alarms may also be collected in a distributed manner, i.e. alarms produced by each equipment may rather be collected (and handled) locally, instead of being centralized.

Once a model is available for each piece of equipment, the distributed supervision algorithm works in the following way. Each local supervisor computes runs of its equipment which explain alarms observed locally. This task only makes use of the local model, and thus handle the state of a reasonably small system. Hence some synchronization of local solutions must be performed to make them compatible. This is done by checking the compatibility of local runs projected on variables shared by two neighbor equipments. Thus, again, this operation is local and involves only small objects; the coordination between neighboring supervisors can be done either incrementally during the computation of local runs, or off-line (at the end). More details on algorithms can be found in [6, 7].

6 Conclusion

The modeling methodology presented in this paper has been used to define and apply a common reference model for all partners in the MAGDA project. We used it to translate the natural-language recommendations into a non-ambiguous formalism, to define the SDH ring and its equipment, to simulate faults through SDL and extract information for the diagnosis. The common model ensured global consistency for the knowledge of academic and industrial partners. It also proved sufficient to implement all necessary elements for automatic identification of primary causes.

Moreover, to show that this model is also sufficient to implement all the stuff to automatically detect primary causes in a network: the complete proposed diagnosis process was also implemented on the Alcatel Almap management platform and fault scenarios on the SDH ring (single or multiple faults) are correctly diagnosed .

Finally, even if the experiment was carried out for a small network, we can reasonably suppose that the approach can be extended successfully to larger, more complex networks, for the following reasons: (i) the number of classes of component will always be small, and (ii) the distributed nature of the algorithms makes them well adaptable when the size of the network grows. The main limitation of the approach so far is that the structural model is assumed to be unchanged, which is not the case in real telecommunications networks.

This work will be continued in the MAGDA2 project in order to drop this constraint and allow dynamic configuration of the model. We are currently studying how to extend this approach in order to deal with end-to-end connections in self rerouting networks. Namely, we consider Label Switched Paths in GMPLS controled networks.

References

[1] ITU-T Recommendation G.803 - "Architecture of transport networks based on the SDH."

[2] F. Krief , A. Osmani. Modélisation d'un anneau SDH pour les besoins de la gestion de fautes. Annals of Telecommunications, France Telecom (March 2002), *in french*.

[3] ITU-T Recommendation G.805 - "General Functional Architecture of transport networks."

[4] ITU-T Recommendation M.3100 - "TMN: Generic network information model."

[5] Website of the MAGDA project: `http://magda.elibel.tm.fr`.

[6] Y. Pencolé, M-O. Cordier, L. Rozé, "A decentralized model-based diagnostic tool for complex systems" 13^{th} *IEEE ICTAI'01, pp. 95–102, Dallas Texas, USA, 2001*

[7] E. Fabre, A. Benveniste, C. Jard, "Distributed Diagnosis for Large Discrete Event Dynamic Systems", *IFAC'02, Barcelona, Spain, July 2002*