An automatic fault diagnosis and correction system for telecommunications management[1]

G. Penido, J.M. Nogueira
Federal University of Minas Gerais
Computer Science Department
Tel (+55 31) 499-5863
Fax (+55 31) 499-5858
Belo Horizonte, MG
Brazil
e-mail: (georgia,jmarcos)@sis.dcc.ufmg.br

C. Machado
Telemig
Telecommunications of MG
Tel (+55 31) 229-3838
Fax (+55 31) 229-3623
Belo Horizonte, MG
Brazil
chris@telemig.com.br

Abstract

In heterogeneous telecommunications networks with a variety of legacy systems of many kinds, the activity of network supervision is a difficult task and involves a large number of operators with a diversity of skills. In such environments, the time to diagnose problems and correct faults usually is high and subjected to many errors. This paper discusses these problems and some techniques and methods for automated fault management. Next, is presents the design and implementation of a system that automatically corrects faults collected by a fault management system. Its utilization greatly reduces mean time to repair faults, therefore increasing the quality of service perceived by clients. The system also reduces the cost of fault management, as it reduces the required work for supervision. The main characteristics of the system enforces its generality: it can correct faults of any network element type and it is a case-based reasoning system, as so enabling the inclusion of new cases to be corrected without interfering in the cases already being treated.

Key words: integrated network management, fault management, fault handling, fault correction, quality of service management.

1 Introduction

Traditionally, network management activities, such as fault management, have been performed with direct human involvement. The management of faults in small homogeneous networks is straightforward. However, as a network becomes increasingly large and heterogeneous, network management activities is becoming more demanding and data intensive.

[1]This work was partly supported by Telemig in a joint project with the Federal University of Minas Gerais and partly by CNPq (a Brazilian Research Agency). Telemig is a telecommunications company of the Minas Gerais state and it is part of the Telemar system.

In fact, fault management activities, when performed with direct human involvement, present some limitations and characteristics: complexity and difficulty in the operation of the management system, as the number of alarms reported increases, response times based on human ability to diagnose and respond with corrective actions, possibility of errors/defects inclusion during the correction process, execution of repetitive work by operators, creating a typical situation for automation and dependency solely on problem solving abilities of an expert troubleshooter.

For these reasons, the automatic management of networks has become necessary and constituted a technological differential of organizations, as it reduces the mean time to repair a fault, improving the quality of service offered to the clients and reducing expenses with labour force.

1.1 Motivation

The motivation of the work presented here comes from a real network management center in charge of supervising a medium size heterogeneous telecommunications plant using a distributed management system.

It is known that typical networks were not designed and built to be globally managed, specially the legacy systems. With the purpose of managing such networks, the SIS platform (*System for Integration of the Supervision*) was designed [6]. It is a distributed management platform which integrates individual devices, transmission networks and supervisory systems to achieve global fault recovery.

In this particular instantiation, the system supervises a huge variety of network elements (NEs), such as workstations, routers, EWSD Siemens switches, Northern Telecom Cellular switches, Elcom Batik switches, AXE Ericsson and many others. The NEs are spread over a geographical area as large as France.

The supervision of a great number and variety of NEs requires a lot of operators working 24 hours a day. Their main activities are the analysis of alarms collected by the management system, the correction of faults and the evaluation of the results of the corrective actions performed.

For example, the supervision of 16 EWSD Siemens Switches out of the 26 existing ones requires four people per each eight-hour period, 24 hours a day. These people just analyze the EWSD switch alarms. There are also nine specialized technicians that are able to correct the faults and configure the switches. This kind of labour force is very costly demanding for the organizations. The operators, although being highly qualified, remain performing repetitive actions almost all the time.

In spite of the number of people involved, the mean time to repair the faults remains high. For example, the EWSD "Trunk group alarm" fault, which can represent critical or major severities and which accounts for 7% of the total EWSD alarms reported, has an average life time of two hours (its incidence rate is about 150 alarms per day).

This high mean time can be easily reduced from hours to minutes by an automated real-time corrective system. This paper presents the design of a case-based reasoning generic system to automatically diagnose and correct faults. It relies on operators' previous experience to adapt solutions to problems according to each situation.

The rest of this paper is organized as follows. Section 2 gives a brief overview of techniques and methods for automated fault management. Section 3 describes the SIS platform where this work was developed. Section 4 presents the automatic fault diagnosis and correction system proposed in this paper. Section 5 discuss the results obtained so far and, finally, Section 6 presents some conclusions and future work.

2 Techniques and methods for automated fault management

Among the more difficult task of fault management are alarm filtering and correlation, fault identification, and correction. Many of these functions involve analysis, correlation, pattern recognition, categorization, problem solving, planning, and interpreting data from a knowledge base that contains descriptions of network elements and topology. Artificial intelligence (AI) technologies are ideal for these types of functionalities [2].

In this case, AI methodologies that involve techniques used for problem solving, new strategies and plans generation, and even more knowledge generation can be used. AI techniques comprise the declarative knowledge and simple inference rules to use the knowledge in different problems. In general, expert systems provide a way of attacking problems that are considered unsolved by machines. They capture human strategies for problem resolution and, selectively, use them in specific circumstances.

A hybrid AI system can be ideal due to the diverse nature of the fault management task. Rather than performing the whole task with one technique that is not ideal for all aspects, a set of techniques are used as appropriate. Thus, the strengths of each technique are emphasized while the weaknesses are overcome by the other. A drawback of this approach is that knowledge acquisition must be performed many times and in very different ways [2].

2.1 Knowledge representation

The knowledge representation area studies methods of systematically encoding what experts know about one subject, or how humans solve problems.

The problem of knowledge representation is that of finding a knowledge representation paradigm that fits the kind of task that is to be automated. Unfortunately, there are no general methodologies for matching real-world tasks and the various knowledge representation schemes that are available to us. The main criterias for measuring the power of encoded knowledge are the logical suitability (ability to express the knowledge to be represented), expression power (existence of a language with complete and well-defined semantic and syntax) and the notational convenience (ease of information reading and writing).

Various representation schemes have already been proposed. Below there is a description of some of them.

Rule-based reasoning is a representation of knowledge as a conditional, if-then clause.

The building of a rule-based expert system for problem resolution can involve various iterations of a cycle (interview with experts, implementation, tests) until achieving a correct system [3].

If the knowledge does not change very often, little maintenance is necessary. However, if the fault correction system is used to solve faults in unpredictable or rapidly changing domains, two problems inevitably occur [3]. The first one is the **system brittleness**, which means that novel cases will make the system fail. Thus, the system can become obsolete very fast. This problem is related with the lack of ability to adapt existing knowledge to a novel situation or to learn from experience. One solution for this problem could be to limit the coverage of the rule-based system. The second problem is the **knowledge acquisition bottleneck**. It happens when experts try to hide special rules and control procedures that will cover unforeseen situations. This problem generates an unwieldy system, unpredictable and unmaintainable.

Model-based reasoning consists of representing a system by a structural model and a functional model [5]. For the telecommunication networks management, the structural representation involves the description of network elements (NEs) and of the network topology. As the real plants tend to be complex, so are the models used in this technique.

Case-based reasoning is an alternative approach to problem-solving that offers potential solutions to the problems of brittleness and knowledge acquisition bottleneck [3]. Its main idea is to recover, adapt and execute past episodes of problems solution in the evaluation of present problems. Past episodes are represented in the form of cases in a case library. The experience acquired with the solution proposed is stored in the case library for future references.

The objectives of a case-based reasoning system are learn from experience, offer solutions for new problems based in past information and avoid intensive maintenance.

One advantage of this kind of reasoning over the rule-based reasoning is the fact that a case-based reasoning system specifies a complete action plan, requiring just one cycle for the selection of each problem. The rule-based reasoning system requires various cycle tests. So, one can conclude that the case-based reasoning system has a better performance.

2.2 An approach for the development of automated systems

Besides all rule-based problems in the fault management automation, it has turned out that they also suffer from lack of communication with their environment (e.g., exchange of information). The model-based reasoning does not become attractive for fault management as it requires the structural modeling of complex and heterogeneous telecommunication network elements.

One important requirement in the development of automated systems to support fault management is a complete understanding of how operators manage faults in complex systems.

Klein's theory of recognition-primed decision making (Klein, 1993 apud [7]) offers a useful construct for describing the actions of problems solvers in complex settings. Klein states that decision making in such settings is based on the recollection of previous experiences, which are then modified to meet the needs of the current situation. Experienced decision makers do not rely on formal models of decision making, but rather on their previous experience. They use their expertise to adapt solutions to problems to fit the current situation.

Rasmussen also suggests that the heuristics and short cuts used by experienced operators may be "recognition-primed" (Rasmussen, 1993 pg. 141 apud [7]). That research makes a fundamental assumption that recognition-based decision making is both a behavioral characteristic of experienced operators and diagnosticians in familiar circumstances and an effective decision making strategy to underpin an architecture for encoding fault management expertise. Such an assumption is reasonable, for such behavior has been observed in experienced operators (Rasmussen, 1986; Sheridan, 1992 apud [7]). The goal of supporting fault management using recognition-primed decision making is to enable an operator, in unfamiliar circumstances, to approximate the recognition-primed decision making behavior of an experienced operator faced with similar but familiar problems.

Artificial intelligence research offers one strategy for accomplishing this. The case-based reasoning process is similar to that used by human expert problem solvers (Kolodner, 1993 apud [7]).

2.3 Some examples of case-based reasoning systems

Some case-based reasoning systems have already been proposed and implemented. In the field of fault management, FIXIT and Critter are examples.

FIXIT (*Fault Information Extraction and Investigation Tool*) is a system for encoding fault management experience and making it available to operators confronting similar anomalous situations [7]. It does not correct faults.

Critter, a case-based reasoning trouble ticket system, was presented by Lundy Lewis in his book entitled "Managing Computer Networks" [4]. Until the book was published, the system was not totally implemented. As we will see in Section 4, the system we propose is much like Critter's, as it has case-based reasoning and it is integrated to the fault management system. However, Critter's approach to the fault resolution problem is to add a case-based reasoning problem-solving component to a standard trouble ticket system. Our approach is to add a case-based component to an alarm collection agent, what makes a decision making in a much earlier stage of fault management. Our approach tries to reduce the number of alarms the operators can see by correcting faults before sending them to the operators.

Other automatic correction systems can be found in the literature. Most of them are proposals. The interested reader can see *Prism*, a case-based telex classifier, *SMART*, a Support Management Automated Reasoning Technology for Compaq Customer Service and *Canastra*, a crash analysis troubleshooting assistant[4].

3 The SIS platform

As already discussed in Section 1, the network management center, which motivated this work, has a preferential management system. Here we describe the system in more details, present the current scope of the application and analyze its fault management functionality and process.

The SIS is a distributed management platform for computer and telecommunication network management. It enables the design of applications and functions in the management areas of fault, configuration, performance, accounting, security and service. The system has been developed by the Federal University of Minas Gerais (UFMG) jointly and funded by the Minas Gerais Telecommunications Company (Telemig) since 1991.

The platform's objectives are to supply and support the management needs of a large-distributed-heterogeneous telecommunications plant. They include the operation, administration and supervision of transmission, switching, data communications (mobile, digital, etc) and computing equipments. It has also provided service management functions.

The SIS's modular and expansible architecture provides the integration of several supervisory systems from a variety of technologies, protocols and manufactures in a referential platform, human interface and data base management system. In addition to its intrinsic interoperability goal, the system also offers operability and portability.

3.1 The SIS's architectures

Functional architecture

The distributed platform presents a hierarchical structure of logical central units (UCs) levels as it is shown in Figure 1(a). This hierarchical tree reflects the operational and administrative structures of most telecommunication companies.

There are three levels of *Operations Systems* (OSs). At the topmost level is the main central unit (UCP). It has the view of the whole plant. The regional and secondary central units (UCR and UCS) have autonomous operations of their corresponding plant segments. Located on the lowest system level are the UCSs. They are the contact points between the equipment's supervisory systems (SSS) and the SIS platform. They are also responsible for the interface between the SIS and the NEs. Generally, UCRs and UCSs correspond to LAN segments connected together on a wide area management network.

Software architecture

The SIS software was designed with the purpose of programs being configured and installed as a logical unit, playing any of the tree level roles. There is a great flexibility to allocate processes at the LAN hosts. The communication among system modules (Figure 1(b)) is reached through the *client-server* distributed programming paradigm, which is implemented by the RPC facilities.

There are basically four software families: information access (Managers, Monitors, UCAs and Agents), DBMS interface (DB-Interface), human-machine interface (Sisterm) and system self-management (Ucproc).

Interfacing with network elements

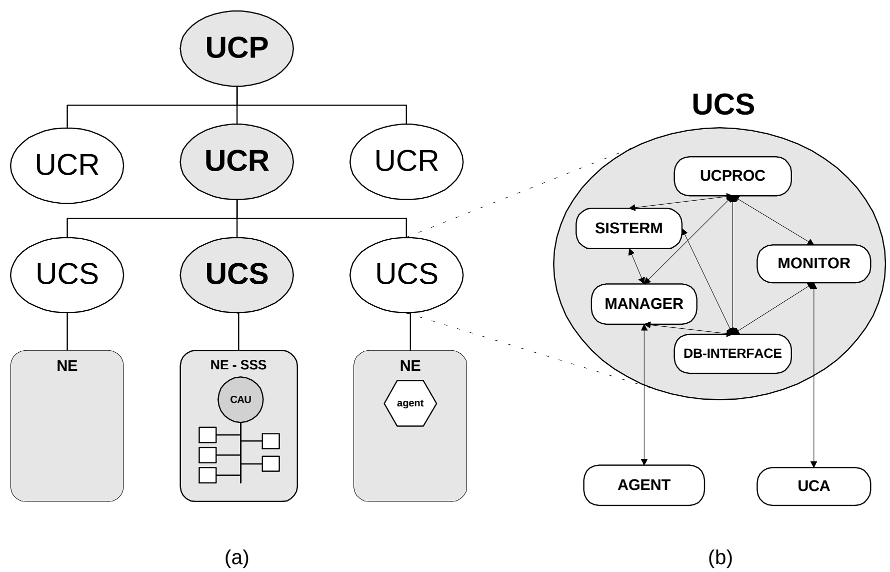

Figure 1: The SIS's functional (a) and software (b) architecture

The SIS distributed platform interfacing with NE can be achieved in three different ways: QSSS, a proprietary interface aimed at communicating with legacy supervisory systems consisting of a data collector (UCA), a bus network and remote units, QSIS, a proprietary CMIS-*like* interface comprised of software agents that communicate with managed resources in private or ICMP/SNMP protocol and Q3, a standard interface comprised of software agents to communicate with managed resources with Q3 interface.

3.2 The SIS into operation

Nowadays, at Telemig's plant, the system is completely installed. It is spread out over a large geographical area of approximately 586,000 km^2, having 2.0 million telephone terminals and 500,000 cellular terminals.

The large area is divided into six regions, each one having one supervisory center or network management center. Each center has one UCR and various UCSs. The UCP resides in the main center, where there is a 24-hour supervision of the plant.

The SIS can gather, store, transfer and process a great amount of data collected from thousands of points. There are approximately 45 telesupervisory systems that are able to collect approximately 17,000 registered telessignals (QSSS) and 335 digital switches, 8 cellular switches and 1,000 other objects of 12 different kinds (QSIS). The daily average number of alarms events is 60,000.

The system can also perform some service management functions, such as cellular terminal activation, subscriber line testing, clients support, account and performance data collection, etc, by executing actions (remote operations) in the NEs. The system interface for the request of services uses the Web technology. Every day there are approximately 15,000 turn on/off of terminal (35,000 in peak days), 3,000 Web queries and 1,500 cellular operations (3,000 in peak days).

3.3 A brief analysis of SIS's fault management process

In order to better understand the fault management features of the system aiming at proposing features for fault diagnosis and correction, here is presented a brief analysis of SIS's fault management process.

For now, as a guide to the analysis, lets consider that a complete fault management process is comprised of the following sequential phases: (1) alarms collection, (2) customer satisfaction maintenance, (3) alarm filtering and correlation, (4) fault diagnosis, (5) development and implementation of a corrective action plan, (6) verification of fault elimination and (7) statistical analysis of fault management process. We will identify these phases in the SIS platform.

Alarms collection: the SIS platform is able to collect alarms continuously from different kinds of NEs, offering different interfaces (QSSS, QSIS and Q3).

Customer satisfaction maintenance: the network management center may take immediate actions for customer satisfaction maintenance.

Alarm filtering and correlation: considering the alarm filtering is one kind of alarm correlation [5], the SIS platform has two kinds of alarm correlation: **filtering correlation**, which is made in the alarm collection phase, and can be applied by severity and **selective suspection**, which consists in the temporary suspection of alarms according to a dynamic context (hierarchy, speciality, alarm severity, etc) that is continuously checked.

Fault diagnosis: it is done manually by operators at the network management center. They try to discover the causes of the faults. This difficulty is a weakness of the system, as it may, among other things, increase the mean time to repair the faults. This is a common characteristic in several management systems.

Development and implementation of a corrective action plan: when the operator recognizes an alarm, he decides if an action is necessary. If it is, he can correct it manually by configuring the NE or create a trouble ticket that automatically opens an in-site service order for a technician. The operator activities in the fault management constitute a repetitive process (a candidate to be automated).

Verification of fault elimination: it is done by visual inspection. After taking the corrective actions, they analyze the alarms list to see if a cleared alarm appeared for the fault corrected.

Statistical analysis of fault management process: the SIS platform stores the alarm events and the corrective actions performed in two different logs. Therefore it does not correlate the commands with the performance of fault management.

We can conclude that, while having some interesting features, some new development has to be done in order to have a completely automated process. This is the case of the diagnosis and correction system presented in the next section.

4 The automatic fault diagnosis and correction system - ACS

A fault diagnosis and correction system could be designed in many ways. It could be integrated to the fault management system or it could be isolated from it in two different manners: a specific system for each kind of NE or a distributed/centralized system for multiple kinds of NEs.

Having a fault management system with defined information models, a management database, interfaces to the NEs and an open architecture, makes a paved way to build an integrated correction system with a number of advantages. For example, the access to the NEs can be performed by this system, using the capillarity of the fault management system.

The SIS platform is a system already in operation, therefore having the desired capillarity. It also offers some software modules that can be used by an automatic fault correction system, such as, for example, an alarm event report module. So it was decided to take advantage of these features in the building of an integrated fault diagnosis and correction system.

4.1 General behavior and system architecture

The ACS is a real-time system. When an alarm is collected by an agent that communicates with the NE (QSIS interface), it is sent to the ACS, which tries to retrieve a "case" from the cases library that fits to the problem described by the alarm. If one case is found, the tests and corrective actions specified in it are performed. These actions are sent to the NE through the agent.

After the execution of the corrective actions, the ACS waits for a period during which the NE should send a cleared alarm.

If a cleared alarm is not received, the ACS ends the correction process of this alarm and sends it to the supervision database. A message of failure in the corrective process is added to the additional information of the alarm. At this moment the operators of the system visualize the alarm.

Otherwise, if a cleared alarm is received, the ACS waits for a period during which the alarm should not happen again. If it happens, the ACS sends the alarm to the supervision database including in its additional information a reincidence message. If it doesn't happen a successful correction were performed and the operators receive this information through a warning alarm sent to the supervision database by the ACS.

Corrections are done simultaneously with fault management through the use of threads. This feature increases the efficiency of the system while keeping low the number of processes and the complexity.

The ACS architecture is depicted in Figure 2.

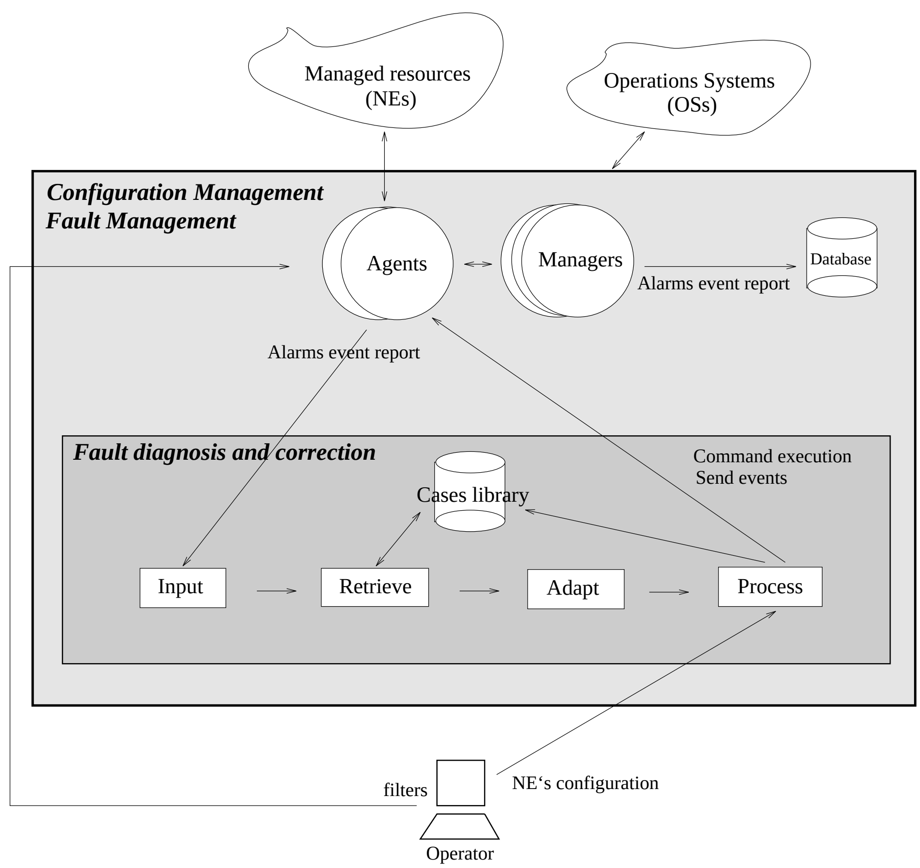

Figure 2: System architecture

4.2 The alarm description language

The exchange of information between NEs and management systems is a major concern in heterogeneous networks. Basically, there are two kinds of information modeling for NE management: "message-based paradigm" and "object-based paradigm". In our case, because of the proprietary interfaces offered by the NEs, we chose the "message-based paradigm". In this kind of communication, the exchanged messages are typically formed by character strings. The structure specification of the message enables an easy understanding by readers, without requiring detailed informations about the protocol. It also eases the software development. However, this method does not support the use of powerful software engeneering techniques, like abstraction and reutilization [1].

In order to have a generic system, the description of the alarms that come from the NEs must be a string of characters following a defined grammar. The defined grammar is described below.

Alarm description or test reply	:= *string*
string	:= *Token,token*;
Token	:= *Token,token* or *token*
token	:= *discriminator=value*
discriminator	:= TNE or DSC or OB1 or VO1 or OB2 or VO2 or OB3 or VO3 or IA1 or IA2 or IA3
value	:= *sentence*
sentence	:= `[^\t\r\n,;]+`

No discriminators can be duplicated and TNE and DSC are mandatory. It will appear in the additional text of the alarm and will be the source information for the diagnosis of the problem. The meaning of the discriminators is described in Table 1.

Discriminator	**Meaning**
TNE	Kind of NE
DSC	Alarm category
OB1	Target object
OB2	Secondary target object
OB3	Tertiary target object
VO1	Qualification of OB1
VO2	Qualification of OB2
VO3	Qualification of OB3
IA1	Additional information 1
IA2	Additional information 2
IA3	Additional information 3

Table 1: Description of discriminators

4.3 The cases library

A case can be seen as a way of solving a specific problem. Therefore, the structure of the database is designed in order to decrease the time it takes to search the cases library by grouping the cases with respect to the kind of the NE and to the category of the problem. Each case has also a list of discriminators to match.

As so, the diagnosis algorithm consists of the best matching of the description of the alarm to a case from the cases library. The type of the NE (TNE discriminator) and the category of the alarm (DSC discriminator) must match perfectly to a case. However the other discriminators are expected to match in a best fit manner.

Besides the matching fields, each case specifies a time interval to be waited before initiating the correction of a fault, the corrective actions that must be performed during the correction period, a normalization timeout that is the maximum time to wait for a cleared alarm, and a reincidence timeout that is the time to wait for the appearance of the same alarm again (what is considered a failure in the correction process).

Initially the cases library will store defined cases. In the future, semi-automatic corrections will finish with the inclusion of a new case in the cases library.

We choose a relational database technology to implement the cases library. The entity-relationship diagram is presented in Fig. 3.

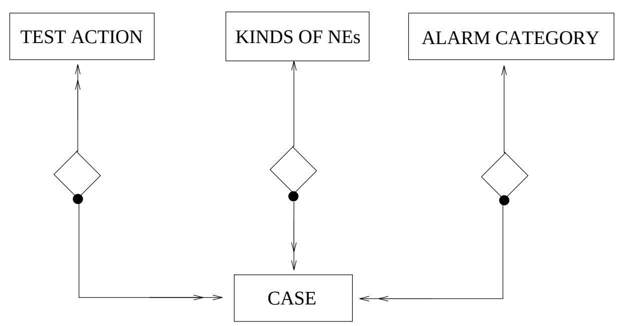

Figure 3: Entity-relationship diagram of the cases library

The test relationship has a "command" field, which contains a test action that need to be performed on the NE before the execution of the correction actions (generally to collect additional information to compose the next commands), and a field called "mask" that enables the parsing of the results of the test. In this way, when a new case requiring a test execution is to be stored in the cases library, nothing has to be done in the source code of the system (the new command will be automatically understood by the system because the mask clearly indicates what are the desired values in the test reply and where they are placed).

In the present implementation, it was used a commercial DBMS to store the cases library (the same used by the SIS platform, from Sybase).

5 Results

As generally stated in the beginning of this work, the expected results were a reduction in the mean time to repair the faults, a reduction in the labour force needed for the faults management and a decrease in the number of errors inserted during the manual correction process.

The reduction of the mean time to repair the faults were immediately perceived. While with manual correction this value was about two hours, with the system in operation the mean time to repair the initially corrected faults decreased to approximately one minute (Graph 1 of Fig 4). This result applies to 95% of the faults of EWSD Siemens switches that can be automatically corrected. The remaining 5% were not configured yet to be automatically corrected. This kind of NE makes possible the automatic correction of approximately 8% of the volume of alarms in the plant. The remaining 92% of the alarms require an in-site maintenance. It is proposed the correction of faults of various kinds of network elements. Thus we foresee a significant decrease in the mean time to repair the faults in the whole plant.

An overview of the operational state of ACS can be seen in Figure 4. The graphs in the top part show the mean time to correct faults and the mean number of correcting commands per fault correction. The particularities of each category of alarm are evident, as the category that has a higher mean number of commands has the lower mean time to correct faults. This happens because the "Failure with configuration" category requires the execution of a slow command to correct the faults. The graph in the botton part analyzes the results of the correction commands per category of alarm. As it can be seen, the percentage of faults corrected is not total because out of the alarms that permit automatic correction, some alarms still may require in-site maintenance.

As the system has been in operation only for a part of the plant, the medium and long term results did not appear yet. By this we mean improvements in the quality of service, reorganization of the labour force, and reduction in operational costs. This is expected to happen in the next months.

6 Conclusions

We designed, implemented and put into operation an automatic system for faults correction. The system is integrated with the SIS platform, a distributed management system that, in the present study, manages a medium size and heterogeneous telecommunications plant.

Despite initially correcting solely a few faults, the great majority of faults will be further automatically corrected.

The system proved to be greatly helpful for the organization, as it immediately decreased the mean time to repair the faults. In the future we expect to have other benefits from this system.

Currently the system is embedded in the SIS platform. However, little work needs to be done to get a stand-alone system. This work consists of defining the communication interface between the system and the network elements and separating the cases library from the SIS database.

A possible future work is the inclusion of a semi-automatic fault correction mechanism, which could help operators correcting a specific fault and would insert the new cases corrected semi-automatically in the cases library.

For instance, we expect to use the system in the configuration and service management areas.

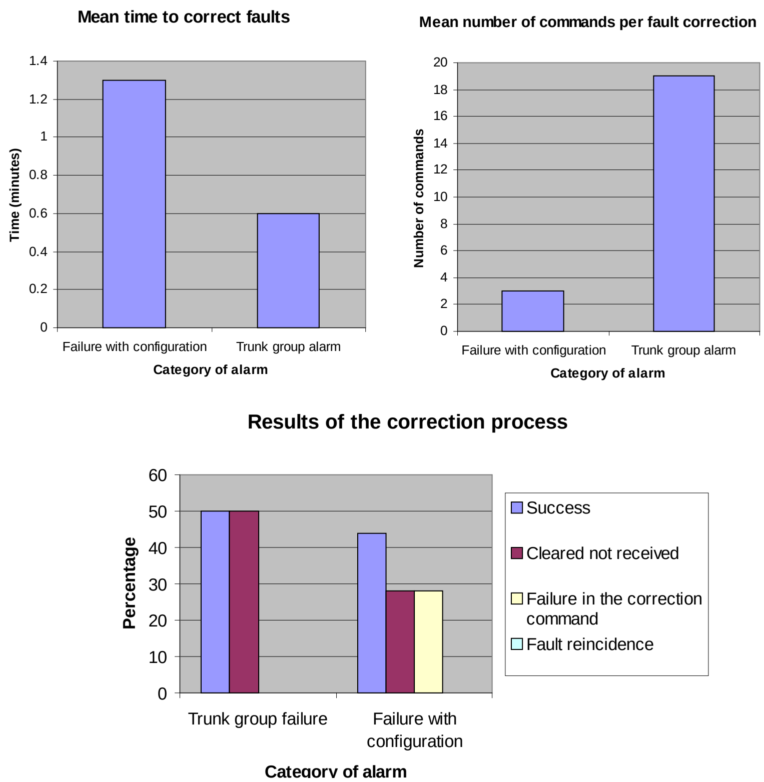

Figure 4: Overview of the operational state of ACS

References

[1] Salah AIDAROUS and Thomas PLEVYAK. *Telecommunications Network Management - Technologies and Implementations*. IEEE Series on Network Management. IEEE Press, 1997.

[2] Denise W. GURER, Irfan KHAN, and Richard OGIER. An Artificial Intelligence Approach to Network Fault Management. Technical report, SRI International, Unknown.

[3] Lundy LEWIS. A case-based reasoning approach to the resolution of faults in communications networks. *Proc. of the Third IFIP/IEEE Integrated Network Management Symposium*, pages 671–681, 1993.

[4] Lundy LEWIS. *Managing Computer Networks - A Case-Based Reasoning Approach*. Artech House Publishers, 1995.

[5] Dilmar Malheiros MEIRA. *A Model for Alarms Correlation in Telecommunications Networks*. PhD thesis, UFMG-ICEx-DCC, Belo Horizonte-MG, 1997.

[6] José Marcos Silva NOGUEIRA and Dilmar Malheiros MEIRA. The SIS Project: A Distributed Platform for the Integration of Telecommunication Management Systems. In *1996 IEEE Network Operations and Management Symposium.* IEEE/IFIP, IEEE Communications Society, April 1996.

[7] Andrew J. WEINER, David A. THURMAN, and Chistine M. MITCHELL. Applying case-based reasoning to aid fault management in supervisory control. *IEEE International Conference on Systems, Man and Cybernetics*, pages 4213–4218, October 1995.