Complex Fault Management Solution for VoIP Services

PÁL VARGA, ISTVÁN MOLDOVÁN
Budapest University of Technology and Economics,
Department of Telecommunications and Media Informatics, {pvarga, moldovan}@tmit.bme.hu
GERGELY MOLNÁR
Ericsson Hungary, gergely.molnar@ericsson.com

Keywords: VoIP, quality of service, fault management

The more widespread the quality VoIP solutions are getting, the more users utilize these services – without even noticing it. High QoS requirements necessitate adequate Operations and Maintenance (OAM) support, as well as a fault management system specialized for VoIP services. Our current study describes a complex fault management system customized for VoIP service providers. This system covers the functions of detecting and processing of alarms, moreover, it features a unique root cause analysis subsystem, also. The solution integrates the classic passive, rule based event processing approach with a method operating by active, fault localizing checks. The appropriate elementary checks are scheduled by an approach novel in the Fault Management field: Petri nets. On one hand the eye-catching nature of this solution makes root cause analysis steps easy to understand, and on the other hand it is fast, due to the simultaneous submission of active checks.

1. Introduction

Voice over IP (VoIP) services are employed not only in cases when we run the VoIP client on our Internet-capable computer to make a phone call. There are numerous hidden or unnoticed cases of VoIP usage – even when we initiate a simple call from our desk phone. Achieving a certain quality with the calls carried by VoIP networks require careful network planning and – once the service is operational – effective execution of operations and maintenance tasks.

The operation of VoIP services with high availability over a carrier network requires a complex Fault Management System (FMS) to be running. This paper describes an FMS supporting VoIP *services-related* Operations and Maintenance (OAM), covering VoIP *network-related* OAM issues, too. Overcoming faults related to QoS of the VoIP service is the task of the service provider, whereas handling carrier errors is the task of the network operator.

During the continuous detection and collection of events arriving from the managed objects, the FMS is able to filter the alarming events, to find their original sources (root cause) and to suggest corrective actions. This way the job of the operating personnel gets significantly simplified, although it never gets superfluous, since the detection and elimination of complex faults still remains their major role.

QoS measures of a VoIP service can be degraded by the faults originated from the network element errors, the misbehavior of the IP network as a managed object and the malfunction of VoIP-related applications, too. Any status-change of the above entities can generate “normal” or “alarming” event reports, which arrive to the FMS. Evaluating the standalone event reports would not allow the FMS to grade the event in question. In most fortunate cases the report could be explicitly alarming, describing the exact root cause, but in the majority of the cases it only reports about propagation or a side effect of an error – not to mention the cases where event reports describe normal and harmless status changes. Since managed objects can report events about different representations of the very same fault, the data to be analyzed by event processing is redundant. Finding the root cause of the fault from this redundant data is quite a challenging job.

Fault Management itself is the process of eliminating the alarms through the steps of event detection, alarm processing and correlation, root cause analysis and fault correction. During event detection the reports of the various event sources get collected together, and transformed to a common format that is easy to process. This set of events gets processed by filtering algorithms (tailored to reduce the number of alarms), event correlation and trend analysis methods (introduced to provide new, more descriptive alarms). The outputs of event processing are alarm notifications *(alarms)*. Each of these alarms describes a specific error, being a subject of the root cause analysis (RCA). The automated Fault Management process results a suggestion for corrective actions. In the sequel, we specify the requirements set toward the mentioned Fault Management steps, together with their design considerations.

In the telecommunications field one of the classic, best documented, complex Fault Management framework is TMN (Telecommunications Network Management) [11]. This has been enriched with some novel ideas while being applied to Hungarian conditions. This TMN adaptation is described in [12]. The drawback of these systems is their hierarchical setup, which makes them inflexible and ineffective for networks suffering from frequent topology changes. This disadvantage puts them out of the picture when choosing the FMS solution for

rapidly changing networks. A brief description of traditional Fault Management frameworks as well as modern root cause analysis methods can be found in [6].

The prototype of our VoIP Fault Management framework has been run at the Department of Telecommunications and Media Informatics of BUTE. The system planning, development and installation has been carried out with the kind help of Ericsson Hungary and Kovax'95 Ltd. The call data record (CDR) evaluation theories had been verified by using anonymized CDRs from the VoIP network of NIIF, Hungary. The description of our FMS framework and the corresponding event processing methodology was presented at the workshop [10], too.

2. Fault Management

A complex FMS should cover the following functionalities:
- detects and stores the event and alarm notifications,
- supports filtering the alarm notifications,
- initiates diagnostic checks to find out root causes of errors, suggests corrective actions.

The FMS continuously watches the state changes of the network and the various managed objects, "hunting" for suspicious events and clearly alarming reports. The appearing errors are handled through the steps of
- event detection,
- alarm processing and correlation,
- root cause analysis and
- fault elimination (see *Figure 1*).

The outputs of this process from the FMS point of view are the result of diagnostic checks carried out by the system and the list of the suggested corrective actions. Based on this information, the operator can start eliminating the fault. Since the FMS may not be able to provide accurate details of the root cause, the operator can *analyze the diagnostic results* and continue refining the details of the original error without having to initiate (duplicate) the diagnostic checks already finished.

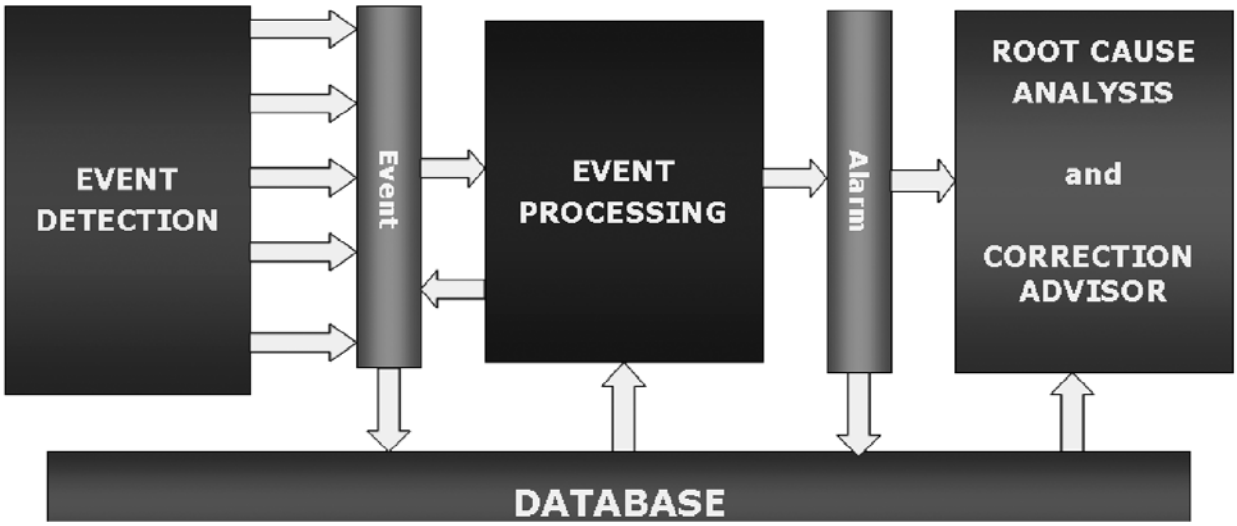
To ease the referencing of the process elements in this article, we should clarify the difference between *events* and *alarms*. We use the notion "event" as a short form of *event notification*, the output of event detection sub-process. This covers the status-reporting events sent by the network elements (including, but not limited to alarming events of direct error notifications) and the events reported by the active elements of the FMS (call generator, active monitor, etc.). On the other hand, we use the term "alarm" as a short form of *alarm notification*, which is the actual output of the event processing subsystem. These cover all the alarming entities that become the input of the RCA, being serious enough to be taken special care of. The operator should analyze each of these alarms separately.

3. Event Detection

The aim of event detection is to perceive the errors endangering VoIP service availability as early as possible. To support this, the main function of this subsystem is forwarding events (arriving from various sources) to event processing in a standardized format. Another task of this subsystem is feeding these events into the event database. Building a database with standard fields requires the database records to be in a similar format. This is another reason for the event detection subsystem to standardize the event notifications before storing and forwarding them. Events arriving to the processing subsystem in a general format ease their handling – both to the automatic FMS algorithms and the operator.

During the fault management of VoIP services the following types of events appear to be interesting:
- reports sent by error detecting entities installed in the network (Syslog, QoS monitor (automatic call generating and evaluating system) [2],[3]),

Figure 1. The elements of Fault Management System and the interaction among them



– reports on call data records generated for VoIP calls (RADIUS records [4],[5] – e.g. frequent appearance of release cause codes classified as erroneous or alarming can indicate faulty system elements),
– events generated by the active monitor (checking the availability of key network nodes and processes),
– errors logged by operators in the HelpDesk system (e.g. triggered by a subscriber complaint).

Current FMS is designed to evaluate the erroneous cases to be corrected by the VoIP service provider. The system is capable of indicating some types of carrier grade errors; suggesting detailed lists of corrective actions for these kinds of faults, however, is out of its scope. To satisfy such demands it is recommended to apply special fault management applications focusing on lower, network level issues.

4. Event processing

As mentioned before, messages of the event detection subsystem get stored in the event database (Figure 1). These event-description types of records form the input data set of the event processing subsystem. Manual processing of this database is not feasible, since it grows in a rate of tens of records per second – not to mention that it contains redundant information (record contents can be multiplicated both physically and logically).

Event processing should be considered as an automated data processing method, resulting in the presentation of alarms in front of the operator. The general process is depicted in *Figure 2*.

Events are stored in the database for further utilization. Their processing starts in the correlator module. The trend analysis module works on the event database, searching for alarming trends in the data set. These two modules actually generate new kinds of events, as opposed to the filter modules, which aim to reduce the number of events by removing redundancies and "highlighting the essence" of an event-pattern. These modules are detailed in the following sections.

4.1. Filters

Event filtering is the simplest among the event processing algorithms. We have chosen *rule based filtering* [7] out of the practical filtering approaches used in current FM systems [6]. During event processing, the filter module checks if the current event meets or fits into the description of any of the active rules. If there is no such filtering rule, the event should not be filtered, hence an alarm notification will be generated (propagation by default). If the event satisfies a rule, it either gets suppressed or promoted to be an alarm – depending on the nature of the rule. The filter criteria include parameters such as validity period, source type, event code, threshold value, etc. We have found that defining merely four types of filters (namely: suppress, counter, redundancy and dominance) are enough to address all raising issues of event suppressing [1]. In brief, our filter types provide the following functionalities:

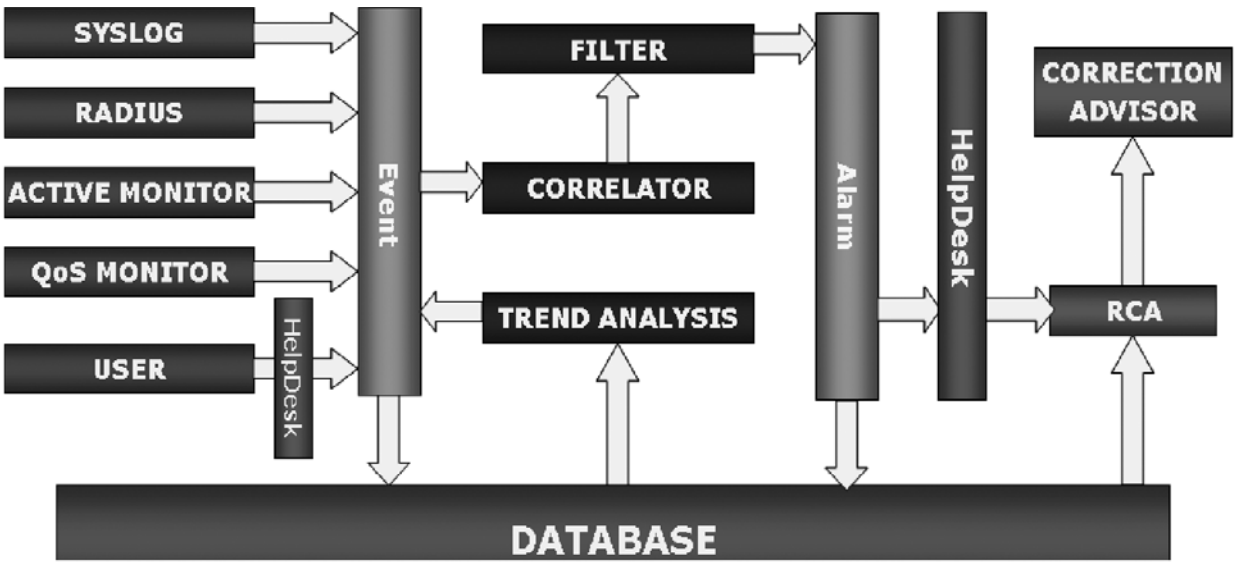
- *Suppress:* complete suppressing of event types
- *Counter:* supressing events of a kind if their amount is under a given threshold
- *Redundancy:* passing through only one event of a kind during a given period – and suppressing all the others arriving meanwhile
- *Dominance:* more serious events override others with lower priority or less descriptive power

The operator can easily change the active rule-set at any time: as rules are defined in a human readable format, it is as easy to create new alarms, as to modify or delete them.

4.2. Event correlation

Event processing is often referred to as "event correlation". In our terminology event correlation does not include filtering (suppressing), but it enriches the event

Figure 2. The flow of event processing



space with information found by correlating various event types. This way we have the opportunity to extract information from events arriving more or less at the same time and provide an essential alarm with high descriptive power, featuring parameters taken from several (different) events. Our event correlation module does not suppress events, but passes all incoming events to the filtering modules together with the specific, correlated ones. Supressing the events successfully exploited by event correlation is a typical task of dominance filters.

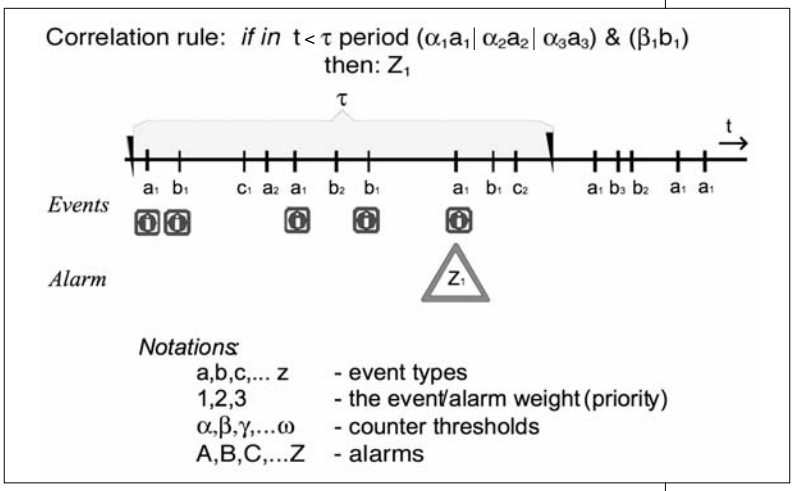


Figure 3. Rule-based event correlation

Among the practical correlation methods used in FM systems [6] we have chosen the rule-based approach. Our event correlation rules can be defined in the same manner as described for the filters. The effect of these rules is depicted in *Figure 3*. The example shown in this figure describes how a new, correlated alarm (Z_1) is generated when the correlation criteria (several number of a_1 events and a given amount of b_1 events arrive in the τ timeframe) gets satisfied [8].

The ultimate reason of event correlation is providing events with the highest descriptive power, gathering many elementary fault-descriptions into a more specific description of the root cause.

4.3. Trend analysis

Utilizing trend analysis methods allows us to predict misbehavior from event patterns of a longer time-scale. To solve trend analysis problems in general, the task is to find an appropriate trend function for the prediction. This method is hard to be applied for our case, since the "trend functions" could use only a few samples of suspicious events before the actual fault happens. This leads to another approach, namely: we should search for exact event patterns predicting future faults. The method should notify about possible future faults when the observed variable (e.g. a parameter inside specific types of event notifications) gets higher than a predefined threshold. The order of the predictive events is indifferent, however, they should arrive inside a given timeframe (otherwise there could be no correlation between them).

We have found that trend analysis methods can only be powerful in predicting accumulative type of faults (where the fault grows out of the continuous degradation of some system resource). Suddenly hitting, catastrophic faults are impossible to be predicted with trend analysis methods, since there are no patterns to predict from.

5. Root cause analysis and advise of corrective actions

We have implemented a novel Root Cause Analysis (RCA) approach during the definition of the FM framework focusing on VoIP services. Traditional FM systems operate with merely passive processing of the events. Some types of alarms generated by such methods were descriptive enough to provide the root cause of the fault, whereas others were too complex. Processing of these complex alarms were still left to the operator, who runs some active checks searching for specific proofs of the misbehavior, this way getting closer and closer to the root cause.

Whereas *passive* RCA systems try to find root causes by taking merely event notifications as input, the *active* approach is to carry out diagnostic checks continuously. Our system integrates the advantages of *passive*, model based event correlation systems that are able to follow topology changes in a flexible way with the automatic scheduling of *active* checks that belong to the every day routine of the operator personnel. The result is a complex fault management system, utilizing both passive and active (checking/intrusion) algorithms.

One of the principles of the passive, rule based event correlation systems is the following: using high complexity rule-sets allow the system to provide alarm descriptions featuring the place of the error and its probable cause. Extracting these descriptive parameters from the complex, correlated alarm report is the simplest RCA task: these alarm parameters are the actual output of the "root cause analysis".

This method, however, can only be used for root cause analysis using passively correlated events. The greatest drawback utilizing this passive rule-set is the inability of actively requesting status and configuration information from the nodes. Without the possibility of fetching key pieces of information from the nodes themselves, it is impossible to automatically provide root cause information for all types of alarm reports. The key questions when solving these problems are when to request the information and how to fetch the required data. Our algorithmic root cause analysis approach – described in the following section – aims to answer this, as well as other issues raised by these basic questions.

5.1. Algorithmic root cause analysis

During the RCA process we utilize all kinds of information (topological, node configuration, node status) to determine the most probable cause of the fault. The result of this process is a list of suggested corrective actions. Taking this list into consideration can greatly ease the job of the network operator in charge.

We have defined the following set of requirements against the fault management system specialized for VoIP services. The system should be capable of

– searching in the event database,
– initiating active checks and scheduling these initiations,
– running these checks and database searches at the same time (simultaneously),
– handling the topology changes in a dynamic way.

Beside these key principles we have born modular system design and short implementation period in mind.

RCA methods in the literature are grouped into three categories: *alarm correlation based systems, statistical methods* and *model based systems*. We have defined a framework capable of initiating active checks and evaluate their result. To meet all the above-listed requirements, we have integrated this framework into a model based RCA system.

One of the advantages of model based RCA systems is their ability of describing network topology in a flexible way. When a new node gets introduced to the system, the correlation and filtering rules can be automatically generated, rather than having to manually redefine them. Considering VoIP architecture, the topology description includes the IP addresses assigned to the physical nodes, and the identifiers of the first hops of the particular node.

The rule-set follows this flexibility as well. It handles the topology information dynamically; hence topology changes can be introduced without having to reconstruct the rules. Event hierarchy is also taken into consideration in the model.

5.2. Petri net of checking routines

In fortunate cases event filtering and correlation provide alarm reports describing the features and the place of the fault relatively well. The parameters of these alarms include the ID, type, and – beside others – the set of node-IDs related to these alarms. The aims of alarm report evaluation are identifying the root cause(s) and suggesting a list of corrective actions. *Figure 4* depicts this process.

Event notifications arrive to the RCA and correction advisor module from the event processing sub-module. There is an *RCA descriptor graph* for each alarm type. These descriptors are standalone Petri nets. We have chosen the principle of Petri nets because it controls the RCA processing by taking into account what pieces of information are available to run specific tests. RCA descriptor Petri nets depict the connection of basic checking routines and the parameters required to initiate these checks. Such a graph is shown by *Figure 5*, which will be explained later in this section in detail.

The *execution scheduler* determines the execution order of elementary checks. Once all the predefined checks have returned with some result, and there are no other executable tasks, the scheduler passes the RCA result to the advisor module. This assigns appropriate corrective actions to the result and presents this list to the operator.

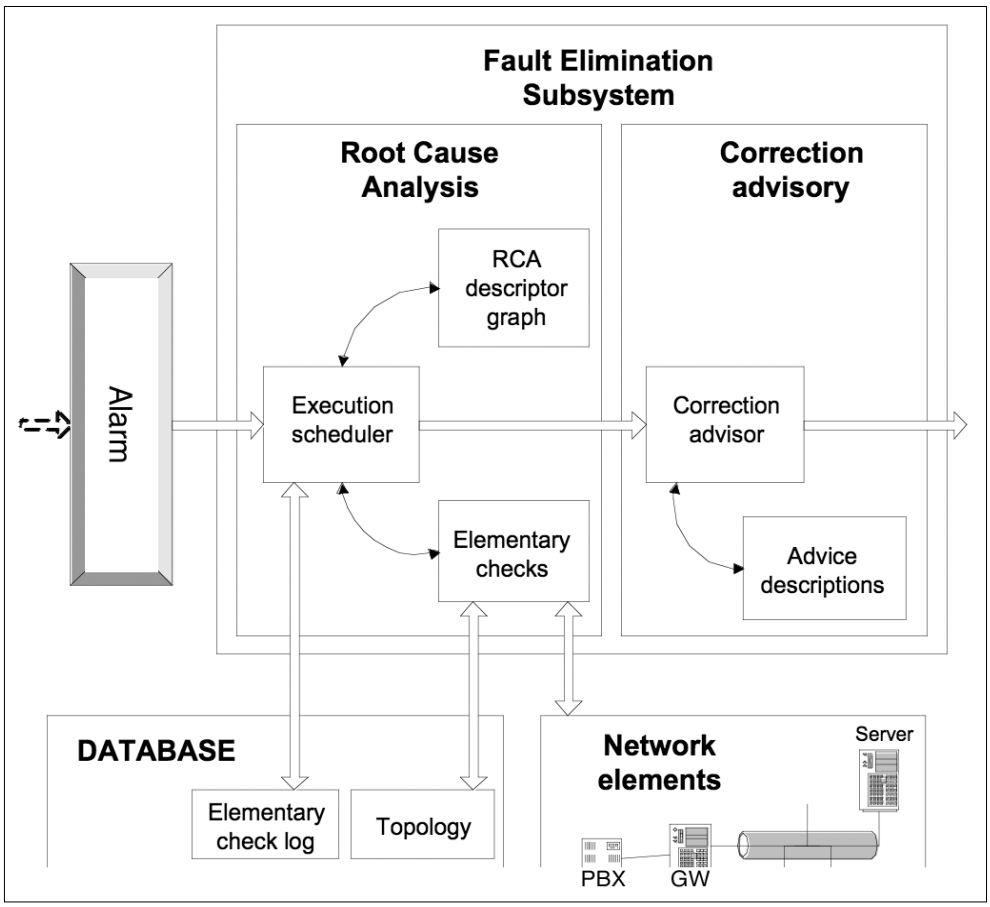


Figure 4. The architecture of the Fault Elimination Subsystem

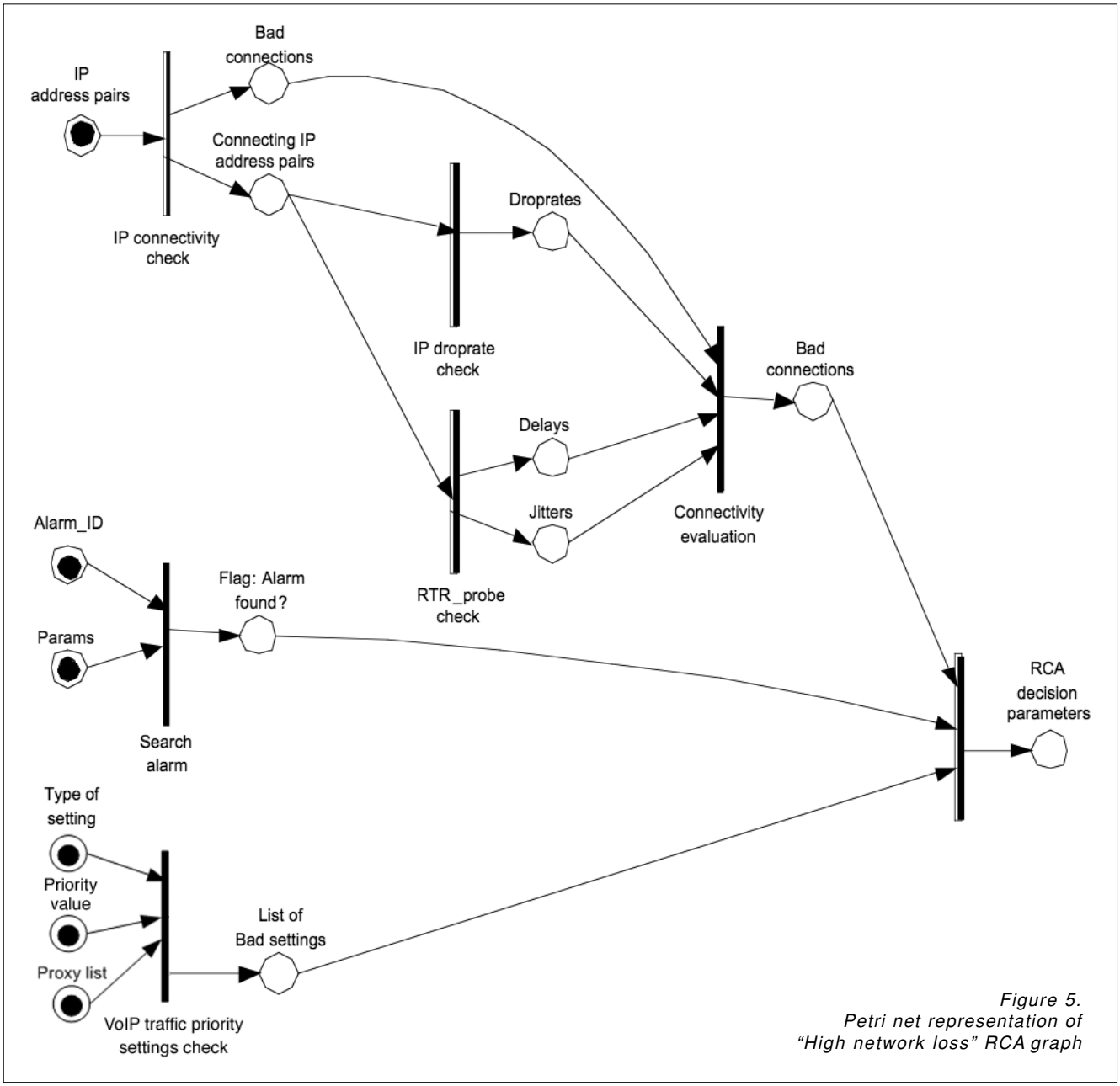


Figure 5.
Petri net representation of
"High network loss" RCA graph

The central element of the above process is the RCA descriptor graph. We have implemented this by utilizing Petri nets, a framework novel to fault management [9]. Using this framework allows the system to simultaneously execute elementary checks (the *transitions* of the Petri net), and schedule new checks in the order of the availability of their input data (the *nodes* of the Petri net). Once the data is available the Petri net node representing that data becomes "tokened". An elementary check gets triggered (the representing transition fires) when all its input data available (all its input nodes are tokened). As a result of the elementary checks, the corresponding output data become available – the nodes representing these data get tokened. The graph can include transitions representing data-request type of functions. These allow the system to fetch data from any kind of source (e.g. interface configuration file, topology database). Fetching such data is an important part of the process, since these complete the input data set of the elementary checks.

Fault management of VoIP services requires various elementary routines to be initiated at some point. These fall into the following categories:

– functions requesting interface status information,
– configuration data fetching,
– active checking routines,
– other
(e.g. active search of alarm patterns).

Let us take the alarm notification of "high packet loss" as a case study for demonstrating the RCA process. Figure 5 depicts the corresponding Petri net. The execution scheduler activates this RCA descriptor when the corresponding alarm arrives to the RCA module. At the first execution step all the *transitions* (checking and data fetching routines) having completely tokened input nodes *fire* (get triggered).

After completing an elementary routine, its output node gets tokened. The data associated with this node could be an input of other elementary routines. The execution scheduler "spends some time" evaluating each active RCA descriptor, in a round robin fashion. During its stay at a designated Petri net, it evaluates if there is a transition able to fire – if there are elementary functions having all their input parameters available.

For every firing transition the scheduler triggers the corresponding elementary function. Once the output of the last (typically the final decision-maker) transition gets tokened, the scheduler passes the results to the advisor module and ceases the RCA entity.

Due to the Petri net based RCA description the execution time of the RCA processes can be cut to half [8], moreover, the execution order does not need to be determined in advance. The task of assigning checking routines to alarm types cannot be eliminated, but still this should be determined only once for each alarm-type. The checks will be executed following the connections of the Petri net, in order of the data availability.

6. Summary

The amount of events, their diversity and arrival rate are all factors making the job of the VoIP network operator personnel extreamly hard.

Using appropriate filtering rule-set the alarm flooding phenomenon – when extream amount of alarms get generated and presented to the operator – can be avoided.

The arriving events can be clustered, counted, prioritized, temporarily suppressed or passed through. Applying proper event correlation and filtering we can generate alarm reports describing the fault better than merely analyzing standalone events. The job of the operator can be eased very much by presenting these verbose alarm notifications rather than the event floods themselves. Being in possession of longer term event and alarm information, we can predict some types of errors by utilizing trend-analysis mechanisms. In the FM case the trend-analysis method covers pattern matching algorithms applied to the event database for predicting saturation-type of faults.

Employing merely passive event processing methods does not always lead close enough to the root cause of the fault. The FM system should initiate the active root cause analysis. During the RCA process the FM can check the possible fault sources by initiating active verification routines. After evaluating the results the FM puts forward a proposal for the place and nature of the root cause, moreover, it suggests corrective actions.

To schedule and evaluate the active tests we have developed a method novel to the FM field: we used Petri nets to describe the connection between active RCA steps. This data-driven framework allows simultaneous submission and evaluation of active checks. Due to this method the RCA process is easy to understand and fast to execute.

The research and development activities described in this study was kindly supported by the Hungarian Ministry of Education, under the project identifier IKTA-00092-2002.

References

[1] Management System for Integrated Voice-Data Networks (Technical Appendices, Phase I. and II.) – BUTE-TMIT, Ericsson Hungary, Kovax'95, IKTA-00092-2002 report, 2003 (in Hungarian).
[2] ETSI TS 101 329-5 V1.1.2; Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; End-to-end Quality of Service in TIPHON systems; Part 5: Quality of Service (QoS) measurement methodologies – ETSI, 2000.
[3] ITU-T Recommendation P.862; Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs – ITU-T, 2001.
[4] RFC 2866 – RADIUS Accounting, 06/2000.
[5] RFC 2865 – Remote Authentication Dial in User Service (RADIUS), 06/2000.
[6] Dilmar Malheiros Meira, "A Model For Alarm Correlation in Telecommunications Networks", 1997.
[7] Roy Sternitt, "Discovering Rules for Fault Management", Eighth Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS '01), 2001.
[8] Tamás Szíjártó, "Fault Management Considerations for VoIP Networks" Scientific Student Conference at BUTE-FIEE, 2004 (in Hungarian).
[9] James Lyle Peterson, Petri Net Theory and the Modeling of Systems, Prentice-Hall, 1981.
[10] Pál Varga, István Moldován, Gergely Molnár, "Fault Management for VoIP Services", Networkshop 2005, Szeged (in Hungarian).
[11] Recommendation M.3000 Series – TMN: Telecommunications Management Network, ITU-T, 1992-1997.
[12] Roxán Reznák, OSS Concept of the Hungarian Telecom