Self-diagnosis Framework for Mobile Network Services

Maria Mykoniati and Costas Lambrinoudakis

***Abstract*—Self-organizing networks (SON) are nowadays considered a very important asset for mobile network providers since automatic solutions for fault diagnosis, network management, configuration and healing are of vital importance for managing complexity of such networks. The first step of a self-organizing network is the self-awareness which is achieved when the system is aware of its own functional state, like its load status, the existence of any failure or any security incident etc. Usually this is performed by a monitoring system and through metrics of system variables which indicate system's state. Any deviation from acceptable levels that may cause any degradation to Quality of Service (QoS), provided by the system, may be considered as fault and is usually reported through alarms. The current paper will present a self-diagnosis framework for recognition of mobile network services failures which consist the first step for developing a survivable, robust, and reliable system. The focus of the presented proposal will be on signaling of Control Plane and User Plane of mobile systems and as a case study 4G LTE mobile network will be used for presenting examples of application of the proposed framework. The current paper is extended version of our paper with title "Fault Prediction Model for Node Selection Function of Mobile Networks" which will be presented to 9th International Conference on Information Communication and Management, and it is part of our research which focusses on methods for building Survivable and Reliable mobile networks.**

***Index Terms*—Self-organizing networks, self-diagnosis, mobile networks, network monitoring.**

I. Introduction

The current paper focusses on providing a framework for building a self-diagnosis mobile network. A "service" of a mobile network is any signaling between network nodes, needed to create a virtual connection between two end points (e.g. UE and PDN-GW). The particularity of mobile networks is that network nodes follow a whole sequence of messages (flow) to complete such a service, rather than just sending packets like traditional IP-networks. This means that any failure in this sequence of messages, or any failure to a certain node, to the system or to any interconnected system, may affect end user services which should be protected from failure.

The first step for protection of system's services is the recognition of failure. The complexity of such a process is very high since mobile networks are usually very large system of systems the monitoring of which may produce a huge number of logs, metrics. or alarms. So, the automation of this process is of vital importance for such systems. The current paper is improvement of the proposed monitoring system presented to our paper "Fault Prediction Model for Node Selection Function" [1]. A short description of this proposal will be presented in Chapter II.

Contrary to traditional monitoring frameworks that are focusing on the whole system, **the current paper provides a framework that focusses on system's services**, as they are the most valuable and vulnerable asset of mobile network systems and has as ultimate purpose to provide an end-to-end reliable system for these services. The idea comes from survivability of systems, which focusses on the continuation of services and not on continuation of the system itself. The term survivability may be described [2] as the "*capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents*". For this to be succeeded the first ability the system should have is failure recognition. Though, it should be mentioned that failure is not always known when system's survivability is considered. Therefore, we focus on the impact of any failure to the services instead of focusing on traditional threat analysis. This is presented in more details in Chapter III.

So, another contribution of this paper is that it provides a service – centric approach that automatically performs a detailed root cause analysis of failure, contrary of traditional monitoring and self-diagnosis sub-systems that are focused on general system performance indicators like failure counters, or general alarms. **The analysis proposed to be provided is very detailed, reaching message information elements (IE) level,** comparing to traditional Key Performance Indicators (KPIs) that are just connecting the failure with a cause which most of the times is misguiding for the root cause of failure to be investigated. Besides that, the root cause analysis of service failure is performed as soon as the failure happens. We support that if an operator or a developer or a tester of responsible vendor has this information, **the time needed for the analysis of the problem will be minimized**. Additionally, the current paper will be based on existing monitoring methodology described by 3GPP standard for telecommunication management [3], [4], and fault management [5], to provide a self-diagnosis framework in node level (Chapter IV). Service monitoring tasks will be performed at each node and any failures accompanied with a possible root cause, based on root cause analysis, will be reported to the network management entity (NM) to be used for other processes of SON, like self-configuration or self-healing, or to be handled manually.

The current research is a result of a few years of working experience in testing and development of 4G networks and of bug fixing for 4G networks.

II. Previous Work

Our previous work [1] focused on providing a framework for protecting critical services from failure to a mobile telecommunication system. More precisely, it was

Manuscript received July 30, 2019; revised November 12, 2019. This work has been partly supported by the University of Piraeus Research Center. Self-Diagnosis Framework for Mobile Network Services.

Maria Mykoniati and Konstantinos Lambrinoudakis are with the University of Piraeus, 80, M. Karaoli & A. Dimitriou St, 18534 Piraeus, Greece (e-mail: mmykoniati@gmail.com, clam@unipi.gr).

doi: 10.18178/jacn.2019.7.2.268

investigating how a mobile system could recognize a threat and react by self-configuration actions which was to select the "always-best" neighboring node. After analyzing possible failure causes, the paper examined the possibility of failure of a node by using predictive analytics on the past incidents of these failures, to each "sender" node. The node that was marked as "sender" was the one that send a message to another node, as part of a bigger flow (example on Fig. 1). Based on this possibility of failure, the "sender" node would choose the "receiver" node that would minimize this possibility to perform a service.

The current paper extends this idea by enriching this automatic root cause analysis system. The issue that needs to be addressed, and which troubled us during the investigation phase of the previous paper, is the **misleading information that is provided by monitoring systems in form of KPIs related to fault causes that are very difficult to localize and understand**. For example, a GTPV2 protocol failure cause is "No resources available". Though, someone who manually investigates this error or an automate self-configuration or self-healing system, needs more information in order to process the failure, like to which node there are no resources. Another example is the cause "System failure". This is the most misleading one since no real cause of failure is provided. We have absolutely no information about what went wrong. Though, there are other causes that are very clear but again we cannot localize which node has failed. For example, the cause "Mandatory IE incorrect". We understand that there was something wrong with one or more Information elements of the received GTP message. Though is really the sender node that has introduced the fault? If this message was sent to another node would also be rejected? These are questions that may be used accompanied with gathered failure KPIs to automatically localize and analyze the failure so that to be fixed manually or automatically.

III. 3GPP Fault Management System

Before examination of root cause analysis, a brief description of the network management system as it is described by 3GPP [3] will be presented.

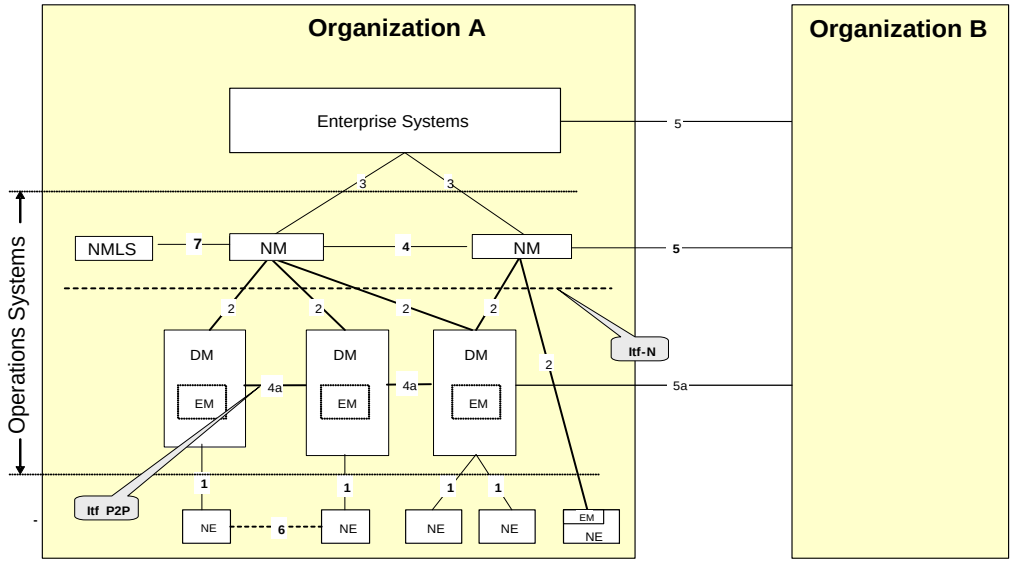

Fig. 1. 3GPP management reference model [3].

Firstly, the topology of such a system may be depicted at Fig. 1 above. What may be observed is that this constitutes a bottom-up reporting system, starting from Network Elements (NEs), which are the first elements of a PLMN that detect any event (ex. fault event) and report it in order to be handled. In NE level, a fault may be detected, localized, corrected or repaired. Additionally, its root cause may be determined, and reconfiguration actions or fault isolation actions may be performed. To conclude, NE has self-healing capabilities in order a service failure to be handled. Though, when more centralized actions are needed, the fault may also be reported through certain interfaces to upper levels of operations system like Element Managers (EMs) or Network Managers (NMs). NMs are the supervisors for the whole 3GPP system and are able to provide fault management capability containing filtering and synchronization of alarms, evaluation of history information related to alarms, further root cause analysis, keeping of operational and connection state and finally reporting the problem to Network Maintenance and Restoration Entity in order to be handled accordingly, or trigger self-configuration and self-healing management actions.

There are two different fault management architectures in upper levels of NMs or NEs that are presented as examples by 3GPP [3]. The first one is for service assurane (Fig. 2 below) and the second one for software fault management (Fig. 3 below). What we see is that in both cases the failure is reported in Network and Maintaince entity. In the first case, after a degradation of QoS of service no matter the root cause of failure, there is a reconfiguration of the network or the service. In the second case, the issue, with root caue of failure related to S/W error, is reported to the vendor company in order to be fixed by vendor's providing of a S/W correction.

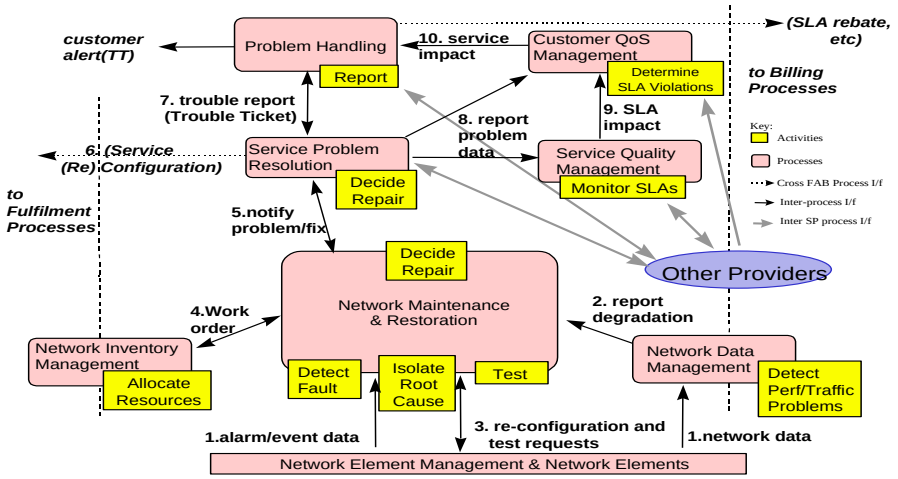

Fig. 2. 3GPP service assurance process flow [3].

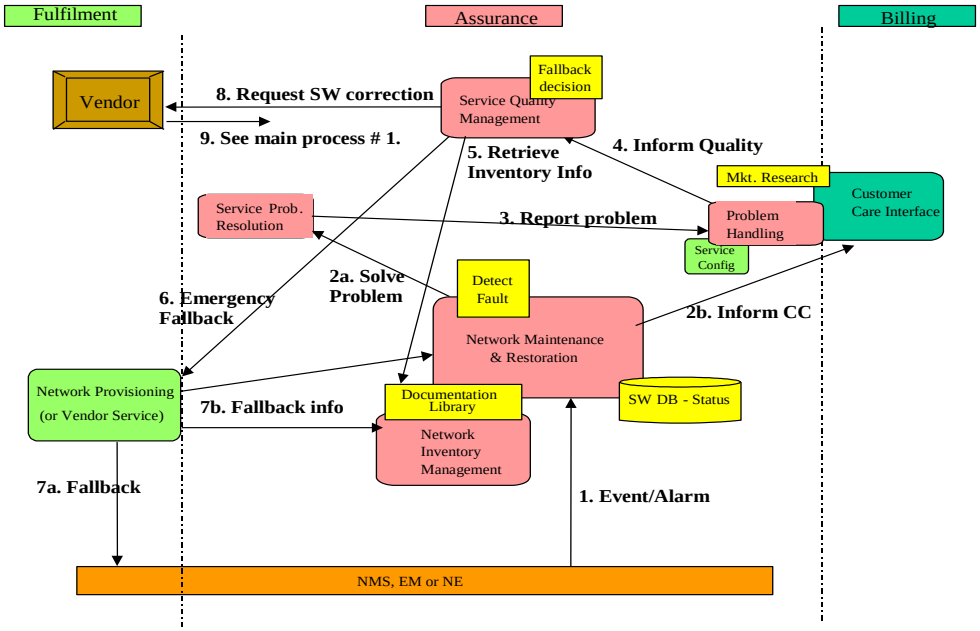

Fig. 3. Software fault management [3].

In the first case we assume that any failure which may be a result of H/W failure, syncrhonization issues, overload situations, connection failures etc should be reported and in the second case, the report should be concentrated to S/W error that have to be depicted and reported in order to be fixed. In any case, what the current paper tries to achieve is the provision of a more accurate **and detailed** root cause analysis for **service fault** in order **to minimize the time** needed for the fault correction, reconfiguration or isolation.

IV. Root Cause Analysis

As it has already been presented, services of mobile networks consist of many messages forming a message flow between network nodes. An example of such a flow is the PDN connection procedure and may be observed at the Fig. 4 bellow.

The framework that will be presented is focusing on **node-level diagnosis (NE level)** so that the root cause of a service failure to be depicted. This means that the scope is the service to be monitored at all levels and connections between nodes, message by message in order to reach the final goal of fault analysis as described by 3GPP "Fault Management" standard [5] which is: "*to minimize the effects of failures on the QoS as perceived by the network users it is necessary to detect failures in the network as soon as they occur and alert the operating personnel as fast as possible*". Here we could add: "and with as much detailed root cause analysis as possible."

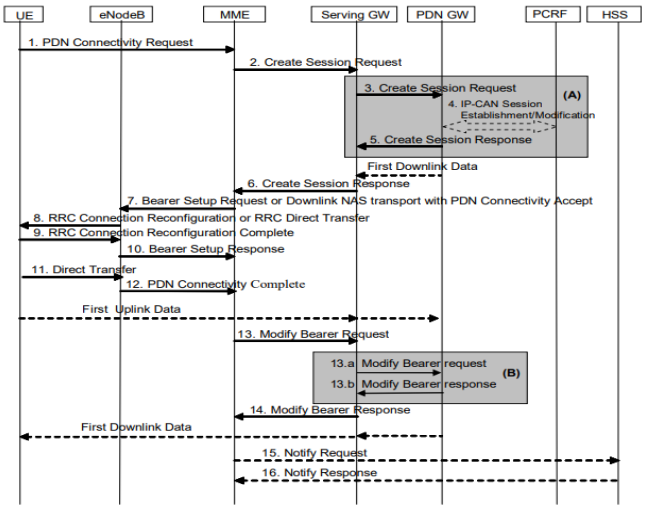

Fig. 4. UE initiated PDN connectivity [6].

So, in case of a service failure, the topology of the network nodes for which management system may extract some diagnostic evidences, may be described by Fig. 5 bellow. A "sender" node (for example MME) is sending a message (for example Create Session Request) and the "receiver" node (for example SGW) receives and processes it to continue with communication with another node (for example PGW), or to just answer back to the "sender" node. Any node is responsible for monitoring the message that it sends and report any failure to the whole message transaction, which is part of a larger service flow.

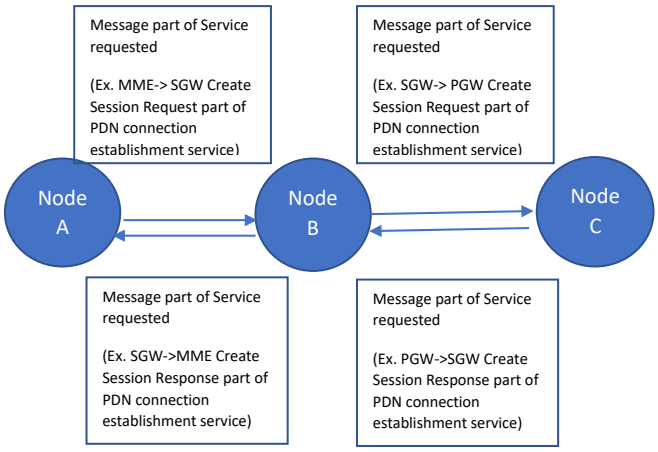

Fig. 5. Topology of NE fault management system.

The outcome of sending a message, may be summed up by three cases, two of which may be considered as service threats since they will cause degradation of QoS service under the acceptable limits, no matter what the root cause of failure is. These 3 cases are: the message is successfully answered; the message is never answered; or the message is rejected. Usually, most of monitoring systems reach this level of analysis using KPIs of attempt, success or reject of a message, in form of counters. Sometimes, rejection metrics are accompanied with a possible cause of rejection provided by the rejection message. The current paper tries to formulate a more sophisticated way of finding the root cause that may be used additionally to the current network management system established to operator's environment. The diagnosis framework proposed will use a top-down level approach for reaching the root cause, following or forming a tree structure (Fig. 9). Diagnosis of the root cause will be based on several questions that a developer or a tester would make in order to find the root cause of a failure and may be automated for taking self-configuration decisions by the SON. These questions or **levels of root cause analysis** are:

1) **First level** of root cause analysis has to do with **general metrics** regarding the failing node or the overall system. Are there any other failures of this kind happening at the same time? For example, when the system is loaded, all services failure indicators tend to increase since the system is not providing the QoS expected. Or is there any alarm raised when the failure occurred? These questions may give us information regarding the overall system status. For example, by this we may know that a node connected to the examined sender node is under restoration and an increase to a service failure is expected. In case this is true then there is no need to send any alarm of service failure. This is the reason why this level of analysis is first, in order to **avoid increased load of alarms**. Finally, is there any other service or services that their failing KPI may indicate the failure of the service examined? For this level analysis, regression analysis proposed to paper [1] may be used.
2) **Second Level**: Is this failure related to a **rejection message or the message sent was never answered and what is the frequency of this failure**? So, the case that a message is never answered or is rejected, may be valid occasionally, under certain circumstances, or may happen any time the service is requested. This information is very important for root cause analysis. As a result, the categorization of threats to a service that are proposed and examined through current paper may be listed below:
   a. Total Denial of Service causing service failure. Message is not answered. This may be permanent in cases of S/W failure or for some time in case of Overload or H/W failure until the failure is restored.
   b. Permanent Service Rejection. Message signaling between a pair of nodes always results to a failure by receiver node rejecting messages.
   c. Occasional Failure causing degradation of service QoS. This means that failure is not permanent but it leads to unacceptable levels of service QoS. In our previous work [1] we chose to define this by DPMO value which is near to 6sigma. This value is 3 failures / 1.000.000 attempts of service.
3) **Third Level:** Is this failure happening with all connected nodes of the same kind? Third Level of root cause analysis, is related to locating the node that is responsible for failure and it is based on statistics gathered for any nodes that the examined sender node is connected to. For example, an MME may select certain SGWs to serve different UEs, based on topological closeness of the UEs and SGW, or based on SGWs load state as 3GPP [29.303] DNS procedure preserves. If the failure is happening always with one SGW then the diagnostic system may propose that this SGW is the most likely node to be the root cause of the failure. If the failure is happening with all SGWs, then the sender node is the most likely to be the responsible node for the failure.
4) **Fourth Level**: What seems to be the probable cause of failure due to "3GPP Fault Management Standard"? For the fifth Level of root cause analysis what will be used is the classification of fault management standard by 3GPP. The "faults", as they are described by the "Fault Management" standard of 3GPP [3] are grouped into one of the following categories:

   d. "Hardware failures, i.e. the malfunction of some physical resource within a NE.
   e. Software problems, e.g. software bugs, database inconsistencies.
   f. Functional faults, i.e. a failure of some functional resource in a NE and no hardware component can be found responsible for the problem
   g. Loss of some or all of the NE's specified capability due to overload situations.
   h. Communication failures between two NEs, or between NE and OS, or between two OSs."

The service impacts related to these failures, and the fault analysis that could follow may be depicted by Fig. 8 below.

What should be mentioned here is that there are already many standard procedures that deal with such failures documented by 3GPP. For example, 3GPP [23.007] which is describing the restoration procedures in case of node failure or the 3GPP [29.807] that handles overload mechanisms. Though, there are cases that these resistance and recovery mechanisms are not performed since not all organizations have implemented them, or there is a failure in their implementation or other root cause has led to these faults. An example of the last case is overload of a node because of CPU or Memory load as result of hanging resources, and not because of service requests load. Finally, even if they exist they are not designed to predict the failure. They only indicate it after it has been realized. This root cause analysis may go on until the level of detail the fault management system is designed to handle.

The current paper proposes the Network Element (NE - the sender node), to report these levels of root cause analysis in an **XML-form** and send the report to the Network Management (NM) entity through ltf-N interface in order further actions to be applied. XML form is chosen since nowadays is used by most programs and programming languages. The architecture of the fault management system is described in [5] and may be observed in Fig. 6 below:

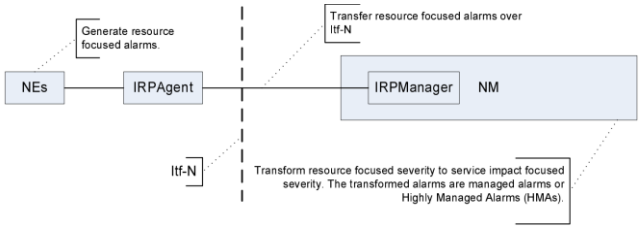

Fig. 6. Fault management system architecture.

The reference standard for XML schema alarms is [7]. For any of the causes described under existing tag “ProbableCause” a more detailed the current paper proposes a root cause analysis to be provided by an alarm that is already triggered or by a new alarm produced for the purposes of the current framework. This XML structure is presented in Fig. 7 below.

```
<element ref="xai:ServiceFailureRCA"/> <!--NEW element proposed by current document additionally to
                                           probableCause below-->
<element ref="xai:probableCause"/><!--Fourth Level of RCA already exists in [7]-->
                                        <!--Some Examples as defined at [7] are:
                                            Loss Of Synchronization, Out Of Memory,
                                            Software Error, Link Failure,
                                            Delayed Information, Denial Of Service,
                                            Message Not Expected etc-->
<complexType name="ServiceFailureRCA">
    <sequence>
        <element ref="xai:MessageFingerprint"/><!-- ServiceRequested -->
        <element ref="xai:ServiceFailureKPI"/><!--KPI of Service Failure -->
        <!--First Level of RCA:
        Start of list of KPIs
        that have been proved to be
        statistical important for
        calculating the probability of
        service failure.-->
        <element ref="xai:StatisticalRelatedKPI"/>
        ...
        <element ref="xai:StatisticalRelatedKPI"/>
        <!--END of list-->
        <element ref="xai:TypeOfFailure"/><!--Second Level of RCA -->
        <element ref="xai:FailedNode"/><!--Third Level of RCA -->
    </sequence>
</complexType>
<complexType name="ServiceFailureKPI">
    <sequence>
        <element name="ServiceFailureKPIName" type="string"/>
        <element name="TimesOfFailure" type="integer"/>
        <element name="ServiceNumberOfAttempts" type="integer"/>
        <element name="RejectionCause" type="string"/>
    </sequence>
</complexType>
<complexType name="StatisticalRelatedKPI">
    <sequence>
        <element name="RelatedKPI" type="string"/>
        <element name="ProbabilityOfServiceFailure" type="float"/>
    </sequence>
</complexType>
<simpleType name="TypeOfFailure">
    <restriction base="Integer"><!-- Second Level of RCA -->
        <enumeration value="0"/> <!-- 0: request never answered -->
        <enumeration value="1"/> <!-- 1: request always rejected -->
        <enumeration value="2"/> <!-- 2: request rejected occasionally but
                                        exceeded acceptable levels of failure-->
    </restriction>
</simpleType>
<complexType name="FailedNode"><!--Third Level of RCA -->
    <sequence>
        <element name="FailedNodeIP" type="string"/><!-- IP of sender/receiver node -->
    </sequence>
</complexType>
<complexType name="MessageFingerprint">
    <sequence>
        <element name="ReceiverNode" type="String"/><!-- Receiver's IP -->
        <element name="MessageId" type="integer"/><!-- message sent (ex. CSR id=0) -->
        <element name="Fingerprint" type="String"/><!-- values of message IEs that
                                                        represent type of message-->
    </sequence>
</complexType>
```

Fig. 7. Fault management system architecture.

V. Self-Diagnosis Framework

Any failure should be somehow related with the message sent in order the root cause analysis framework to provide more accurate results. Furthermore, it is very important to indicate which is the exact message sent since there are many messages of the same category. For example, there are many types of Create Session Requests like emergency request or IOT request depending on the values of the Indication flags of the CSR message. By using any message IEs, **a fingerprint of the message** containing those IEs that characterize the type of the message should be kept. The current paper has chosen to use a search tree to represent messages sent by each node to the receiver node. The search tree will be kept to the sender node to extract conclusions regarding the failure of the message sent. This search tree will have as parent node the sender node. The next level will be the nodes that this sender node is connected to and then the messages sent to these nodes Fig. 9. Each message could be extended to the IEs that it includes. Of course, we cannot use all IEs in our example because we would end up with a huge tree. But every combination that provides a failure should be kept in this search and reported in “messageFingerprint” field by the XML report file presented in Fig. 7.

In our example of Fig. 9, message Create Session Request (CSR) sent by a MME to a SGW, contains several information elements that indicate different types of service. Of course, IEs like IMSI or MSISDN are changing depending on the UE that requests the service. So, they are not good indicators for categorizing the different CSR messages.

Though, if IEs like indication flags, RAT type, APN etc are used, the categorization of the request is easiest. For example, the system may conclude that any time a CSR with indication flag “Control Plane Only PDN Connection Indication” for a certain APN is requested the SGW replies with a rejection message instead of just counting the number of Create Session Responses with general rejection cause.

Some examples of root cause analysis that may be extracted from the tree of Fig. 9 may be listed below:

For the example (1) what may be concluded is that any time the message Create Session Request with RAT Type = 0000 0000 and indication flag 0000 0101 0011 0001 for the APN = 0000 0001 is sent from MME to any SGW (SGW1, SGW2), the service fails. So most probably, the failure is on MME side. Number (1) is also marked on table of Fig. 8 with service possible root cause analysis.

Example (2) shows a case that no matter the value of Indication IE that follows, if the RAT Type IE has value 0001 0010, then the service is failing. Though, this is not true with

all SGWs that MME is connected to. So, it is safe to conclude that SGW_1 seems to be the node that causes the failure.

Example (3) is a representation of synchronization issue of node SGW_2 in certain CSR messages. We may also see that failure / attempt ratio is less than 1 which indicates that this is a random failure and not a failure that happens every time.

Example (4): Miss-configuration of SGW_3. The cause of failure "no resources available" indicated that.

Example (5): Permanent DOS failure indicating a H/W failure or overload or communication error of SGW_3 etc. If the error is not yet reported, the NE should do so.

	Hardware Failure	**Software Problems**			**Functional Faults**	**Overload/Increased Load**	**Communication failures**
		Sending Node	Receiving Node	Database Inconsistencies		Request Load	
SYMPTOM: Total Denial of Service	Node B H/W has failed, resources like CPU or memory are extremely loaded (other cause than increased requested load ex. hanging resources).	Malicious message is sent from node A and ignored to node B	Node B S/W bug result to wrong processing and ignorance of message sent by node A		Configuration issue of node A or B: ex. wrong IP addresses are configured.	Node B is overloaded by message requests.	Path between node A and node B has failed. (Path may be a physical path or a whole network)
Fault Analysis 5	Increased DOS failure indication - all messages sent to a certain node are not answered.	When sender node is sending this message to all nodes that is connected to and service is always failing.	When node is sending this message to one node or a family of nodes that is connected to, and service is always failing.		Increased DOS failure indication - all messages sent to a certain node are not answered.	Increased DOS failure indication - all messages sent to a certain node are not answered.	Increased DOS failure indication - all messages sent to a certain node are not answered.
SYMPTOM: Permanent Failure by Rejection message	Resources of receiver node or any other node connected to the receiver are unavailable	Malicious message is sent from node A to node B causing service failure 1	Node B S/W bug result to wrong processing of message received by node A which may result to malicious message sent to another node or sent back as an answer to node A causing service rejection Lack of robust mechanism to ignore malicious components of message in case it is possible.	Corrupted data may cause failure to service request. (3GPP 23.007 provides the recovery from this failure)	4 Configuration issue of node A or B: service is not supported by node B	Node B is overloaded by message requests and services are rejected by overload mechanism [3GPP 29.807].	
Fault Analysis	Usually certain rejection causes (ex. Resources Unavailable) will indicate this.	When sender node is sending this message to **all nodes** that is connected to and service is always failing. Certain cause may also indicate this ex. mandatory IE incorrect.	When node is sending this message to **one node or a family of nodes** that is connected to and service is always rejected. 2	Usually certain rejection causes will indicate this failure (ex. Unknown UE) no need for extra root cause analysis.	Usually certain rejection causes will indicate this failure (ex. service is not supported).	Certain rejection causes will indicate this failure (ex. Overload indication in answering messages).	
3 **Occasional Failure by Rejection Message**	CPU / Memory load causing message processing delays		Lack of robust mechanism to ignore re-sending messages Lack of robust mechanism to handle collision scenarios			Increased Load causing message processing delays	Synchronization inconsistencies in communication of the two nodes causing message delays, message re-sending or collision scenarios.
Fault Analysis	Occasional rejection of same messages.		Occasional rejection of same messages.			Overall messages rejection KPIs increase.	Occasional rejection of same messages.

Fig. 8. Root cause analysis for 5th level of analysis.

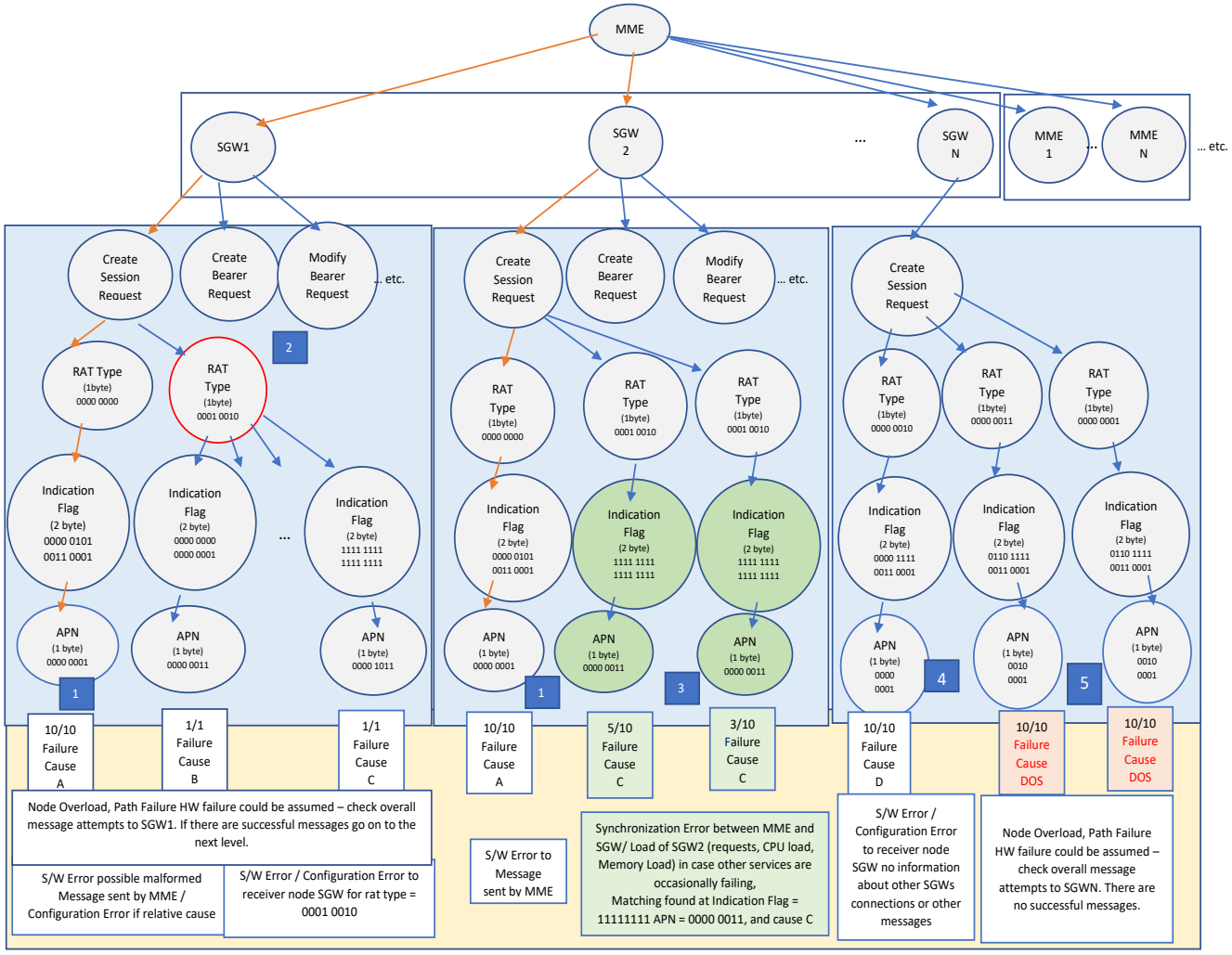

Fig. 9. Search tree of MME node.

VI. Conclusion

The current paper presents a self-diagnosis framework for analyzing the root cause of failure of mobile telecommunication networks based on fault management and high-level requirements of 3GPP standards. This work is extendable version of our first paper (1) describing a fault prediction model for node selection function. It describes a more detailed approach on root cause analysis of failure and how this could be reported to the whole telecommunication management system for mobile networks. The idea is based on previous working experience and current paper tries to automate this experience in order to extend the self-diagnosis framework proposed in the first paper. The next step, as future work, is firstly to describe how probability of failure could be used from DNS system in order to be part of selection function of mobile systems and secondly, to describe how Software Development Lifecycle of mobile systems could be changed in order the system to be built in such a way to provide self-diagnosis and self-configuration functionality.

References

[1] M. Maria and K. Lambrinoudakis, "Fault prediction model for node selection function of mobile networks," in *Proc. The 9th International Conf. on Information Communication and Management*, Prague, Czech Republic, 2009.

[2] R. J. Ellison, D. A. Fisher, R. C. Linger, H. F. Lipson, T. A. Longstaff, and N. R. Mead, "Survivable systems: An emerging discipline," in *Proc. the 11th Canadian Information Technology Security Symposium (CITSS), Ottawa, Ontario Canada, Communications Security Establishment*, pp. 2, November 1997.

[3] 3GPP 32.101, *Telecommunication Management; Principles and High-Level Requirements*, 15th ed. 2017.

[4] 3GPP 32.102, *Telecommunication Management; Architecture*, 15th ed. 2018.

[5] 3GPP 32.111-1, *Telecommunication Management; Fault Management; Part 1: 3G Fault Management Requirements*, 15th ed. 2018.

[6] 3GPP 23.401, *General Packet Radio Service (GPRS) Enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) Access*, Version 16.3, 2019.

[7] 3GPP 32.111-5, *Digital Cellular Telecommunications System (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Telecommunication Management; Fault Management; Part 5: Alarm Integration Reference Point (IRP): eXtensible Markup Language (XML) Definitions*.

Maria Mykoniati was born in Greece in 1983. She holds a B.Sc. in computer engineering from the Technological Educational Institute (TEI) of Athens in 2007, an M.Sc. in data communication systems from the Brunel University in 2009, and an M.Sc. in security of digital systems from the University of Piraeus in 2012. Currently, she is working at NOKIA as cyber security engineer. From 2014 until 2019, she was working at NOKIA as a software engineer and QA engineer. From 2010 to 2014, she was working at GNT and QX2.gr as a web developer. Additionally, from 2008 to 2014, she was working as a freelancer software engineer for Technical Educational Institute (TEI) of Athens. Her current research interest is in the areas of information and communication systems security and system's survivability, robustness and fault tolerance.

Costas Lambrinoudakis was born in Greece in 1963. He holds a B.Sc. in electrical and electronic engineering from the University of Salford in 1985, an M.Sc. in control systems from the University of London in 1986, and a Ph.D. in computer science from the University of London in 1991. Currently, he is a professor at the Department of Digital Systems, University of Piraeus, Greece. From 1998 to 2009, he has held teaching position with the University of the Aegean, Department of Information and Communication Systems Engineering, Greece. For the period 2012-2015, he was a member of the board of the Hellenic Authority for Communication Security and Privacy, while from 2016, he serves on the board of the Hellenic Data Protection Authority. Finally, from 2015 he is the head of the Department of Digital Systems and the director of the Systems Security Lab. His current research interests are in the areas of information and communication systems security and of privacy enhancing technologies. For many years he is working on issues related to the protection of personal data and the compliance of information systems to the National and European Legislation. He is an author of more than 120 scientific publications in refereed international journals, books and conferences, most of them on ICT security and privacy protection issues. He has served as program committee chair of 15 international scientific conferences and as a member on the program and organizing committees in more than 200 others. Also, he participates in the editorial board of two international scientific journals and he acts as a reviewer for more than 35 journals. He has been involved in many national and EU funded R&D projects in the area of Information and Communication Systems Security.