Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: December 31, 2015

M. Toy
Comcast

June 29, 2015

Architectural Framework for Self-Managed Networks with Fault Management Hierarchy
draft-mtoy-anima-self-faultmang-framework-00.txt

Abstract

This document describes a self-managed network identifying network problems during failures and repairing them. Self-managed Network Element (sNE) architectures and Network Management System (sNMS) architectures for centrally and distributedly managed networks are described. A hierarchy among repairing entities is defined. An in-band message format for Metro Ethernet networks is proposed for the fault management communication.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 31, 2015.

Copyright Notice

Table of Contents

1. Introduction

The industry is focused on auto-configuration [GANA], [SUSEREQ], [SELFMAN] and monitoring of network resources and services, isolating problems when there are failures, and fixing them by sending technicians to the sites most of the time or downloading certain configuration files remotely for configuration related problems. The concept of network identifying problems by itself and fixing them and only sending technicians to the failure site only when there are single-point of hardware failures (i.e. there is no hardware redundancy) is not practiced [SMN,SMCEN]. Tools for self-managed networks are not developed either. On the other hand, auto-configuration of network elements (NEs) such as cable modem (CM) and cable modem termination system (CMTS) is being practiced by Multiple System Operators (MSOs) using Data Over Cable Service Interface Specification (DOCSIS) back-office systems. Similar procedures are also used by DPoE networks [DPoE] for auto-configuration of NEs and services. This draft does not discuss the auto-configuration, but focuses on fault management aspects of self and centrally or distributedly managed networks.

This draft describes a self-managed network where each self-managed NE (sNE) in a network monitors its hardware and software resources periodically, runs diagnostics tests during failures in a hierarchical fashion, identifies problems if they are local to the sNE and fixable by the sNE, and reports failures and fixes to a centralized network management system (sNMS) to be accessed by network operators, field technicians, customers, and other sNEs in the network. If the problem is not locally fixable by the sNE, the Self-Managing Regional NMS (sNMRn) or sNMS runs its own rule-based logic to determine if the problem is fixable remotely by the sNMRn or sNMS. If it is not, a message (i.e.notification) is sent to a network operator or field technician to fix the problem.

Failure type; if the problem is fixable locally by sNE, remotely by sNMRn or sNMS, or remotely by a technician; and estimated fix time are communicated with a newly defined message format. The hierarchy of fixing failures is network architecture dependent, as discussed in sections 8 and 9.

2. sNE Architecture

An sNE (Figure 1) consists of an intelligent NE (iNE) and intelligent agents.

The intelligent NE (iNE) is built to have redundant hardware and software components as depicted in Figure 1, where each hardware or

software component is intelligent enough to run its own diagnostics and identify faulty subcomponents. Self-managing agents (i.e. intelligent agents) may take over after internal diagnostics of each component is completed. Furthermore, iNE keeps a redundant copy of its current or default configuration.

The intelligent subcomponents can be smallest replaceable units such as chips, operating system, and protocol software that are capable of periodic self-checking, declaring a failure when it is unable to perform its functions, running diagnostics and identifying whether the faulty entity is within the subcomponent or not, escalating the diagnostics to the next level in the hierarchy when the diagnostics are inconclusive.

When there is a failure, if failed entity is unidentified as a result of the diagnostics tests run by the intelligent subcomponents, the iNE is able to run diagnostics for a pre-defined set of subcomponents that are collectively performing a specific function. A pre-defined set of subcomponents can be a collection of components that are contributing to the realization of a main function such as packet forwarding, deep packet inspection, event forwarding, etc.

If the diagnostics tests ran for a pre-defined set of subcomponents cannot identify the failed entity, the iNE is able to run diagnostics at NE level to determine the failure. After the failure is identified to the smallest replaceable hardware (e.g. chips, wires connecting chips, backplane, etc.) and/or software entity (e.g. kernel, log, protocol software, event forwarding discriminator, etc.), the responsible intelligent agents determine if the failure is fixable and initiates a message to related parties with estimated fix time to repair. If the iNE diagnostics are inconclusive, then that will be communicated as well.

Each self-managing agent (i.e. intelligent agent) monitors the entity that it belongs to, and may run additional diagnostic tests to identify problems during failures, initiates a failure message, fixes problems, and initiates a fix notification to the central or regional self-managing systems and other related entities. The message (i.e. notification) indicating that the fixing entity is sNE, is communicated to other sNEs, regional and central network management systems systems, field technicians and customers (if desired). If the problem is determined to be not fixable locally after two-three tries or without a try, depending on the problem, a message is sent to the regional or central network management systems by the sNE indicating that the fixing entity is unidentified.

The intelligent agents are one or more intelligent Hardware Maintenance Agent(s) (iHMA(s)), intelligent Operating System

Maintenance Agent (s) (iOMA (s)), intelligent Application Maintenance Agent (s) (iAMA (s)), and intelligent Capacity Management Agent (s) (iCMA (s)), depending on the implementation.

The iHMA is capable of periodically monitoring hardware entities such as CPU, memory, physical ports, communication channels, buffers, backplane, power supplies, etc., and initiating pre-defined maintenance actions during hardware failures. iOMA is capable of periodically monitoring operating system and initiating pre-defined maintenance actions during Operating System failures. The iAMA is capable of periodically monitoring application software and protocol software, and initiating pre-defined maintenance actions during application and protocol software failures. The iCMA is capable of periodically monitoring system capacity, load and performance, and collecting measurements. When capacity thresholds are exceeded, the iCMA initiates pre-defined maintenance actions.

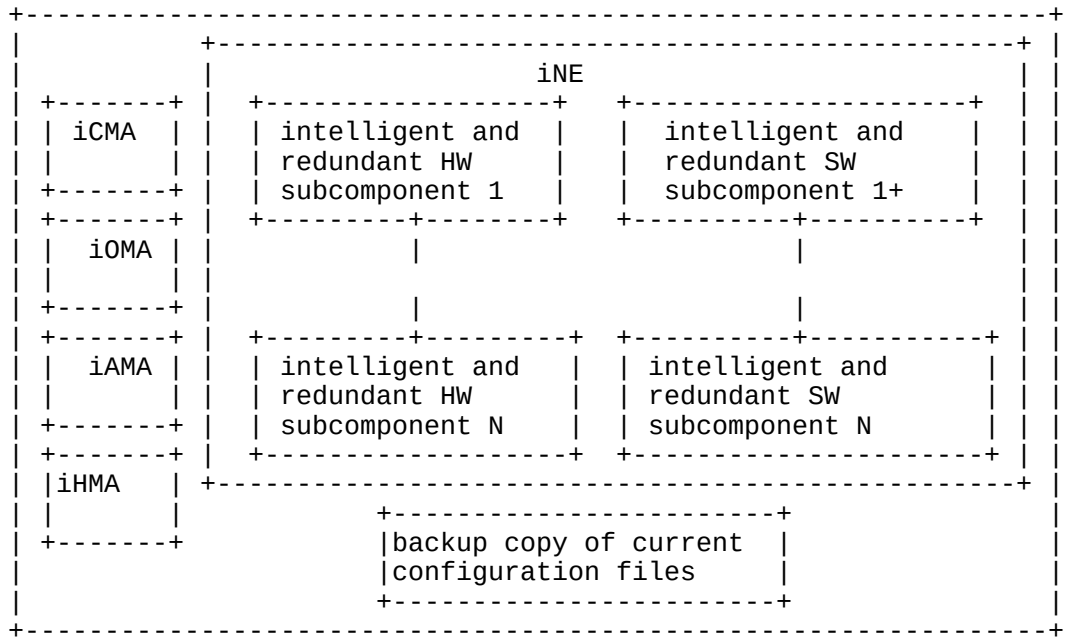


Figure 1: Self-Managed NE Architecture

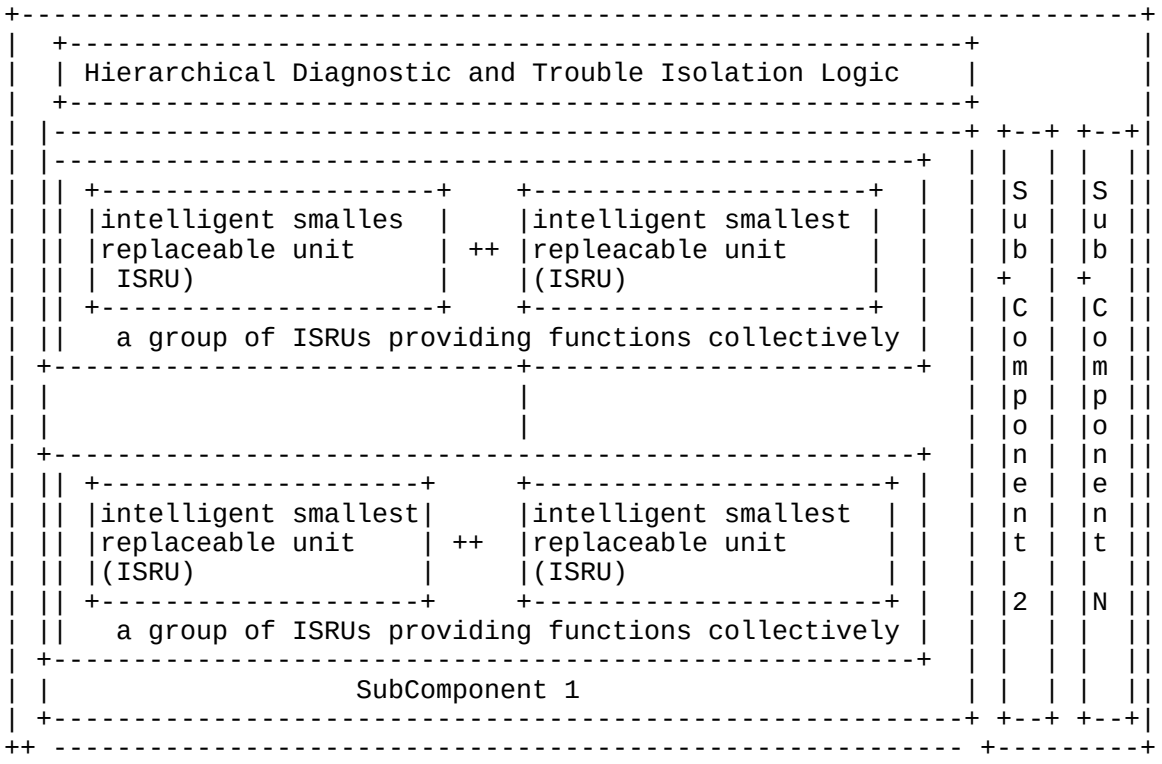


Figure 2: Intelligent NE Architecture

3. Self-Managing Network Management System (sNMS) Architecture

A Central or Regional sNMS consists of an intelligent NMS (iNMS) that mainly deals with remote fixes, a Task Manager (TM) to manage tasks to be executed, copies of software modules for each type of sNE, a Traffic Manager (TrfMgr) to deal with network level traffic management issues such as routing policies, load balancing, connection admission control, congestion control, Event Forwarding Discriminator (EFD) to forward failures and fixes to network operators and customers, data base(DB) to store data, and a user interface such as a Graphical User Interface (GUI) (Figure 3). The sNMS is redundant where the active sNMS is protected by a stand-by sNMS. The iNMSs in active and stand-by units perform periodic self-checking. When the active sNMS fails, the stand-by sNMS takes over the responsibilities.

The user interface provides human and machine interfaces. A Database (DB) stores user interface events and data collected from network. A

Task Manager prioritizes and schedules execution of the tasks including repair and configuration of activities that can be performed remotely using a Rule Based Logic module. A Data Handler collects end-to-end connection level measurements and sNE level capacity measurements, and stores them in the DB to support the TrfMgr.

The Task Manager (TM) of sNMS manages tasks to be executed by the sNMS. The Rule-Based Logic determines if the problem is remotely fixable by the iNMS.

The iNMS is expected to include a Fix Manager (FixMgr) for each sNE type to fix the sNE problems remotely; store software modules specific to the sNE; and capable of running network level traffic management algorithms such as routing policies, load balancing, connection admission control and congestion control. Furthermore, the iNMS holds a copy of each sNE agent and remotely loads into sNEs when needed.

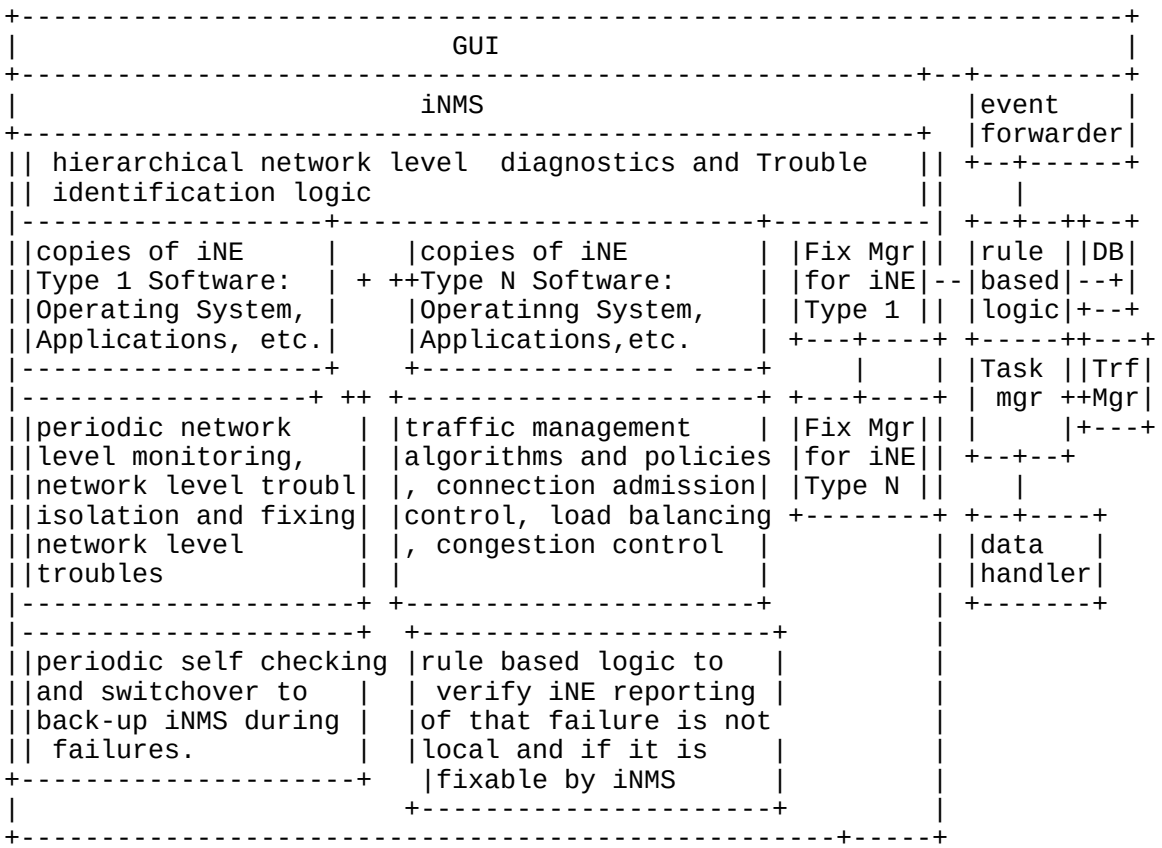


Figure 3: Self Managing NMS Architecture

An intelligent NMS (iNMS) (Figure 3) periodically monitors the network that sNMS is managing, identifies network level failures, estimates and communicates the fix time to related parties, and fixes them. When the sNE reports that the failure is not local (i.e. either tests are inconclusive or sNE is not capable of fixing it), the Rule Based Logic of the sNMS verifies if the sNE failure is not local.

There are no changes introduced to interfaces between the management systems and the network for self-management. The well-known protocols such as SNMP, IPDR (IP Detail Record) for usage information, Network Configuration (NETCONF) for manipulating configuration data and examining state information, and YANG modeling can be employed.

4. Intelligent Agent Architecture

The intelligent agent architecture is depicted in Figure 4. Its Rule Based Logic module determines problems and initiates fixes if the problems are local to sNE, initiates tests for the fixes, determines if the fix procedure or a step or some of the steps are to be repeated, and initiates a message to all related parties about the fix. If the problem is not local to the sNE, the agent informs all related parties including the sNMS for its conclusion which is that the fixing entity is unidentified. If the result of diagnostics cannot identify the failed component which is inconclusive, that will be conveyed as well.

A Scheduler module determines the priority and order of the tasks for each functional entity within the sNE that it belongs to. An Application Programming Interface (API) provides an interface to various types of Software and Hardware entities within the sNE. A Data Handler module collects necessary data for the sNE, performs the fix, and keeps the data associated with the task. The Authorization (AUTH) module authenticates local user access and remote user access from the sNMS interface to sNE agents. The Utilities module supports various file operations.

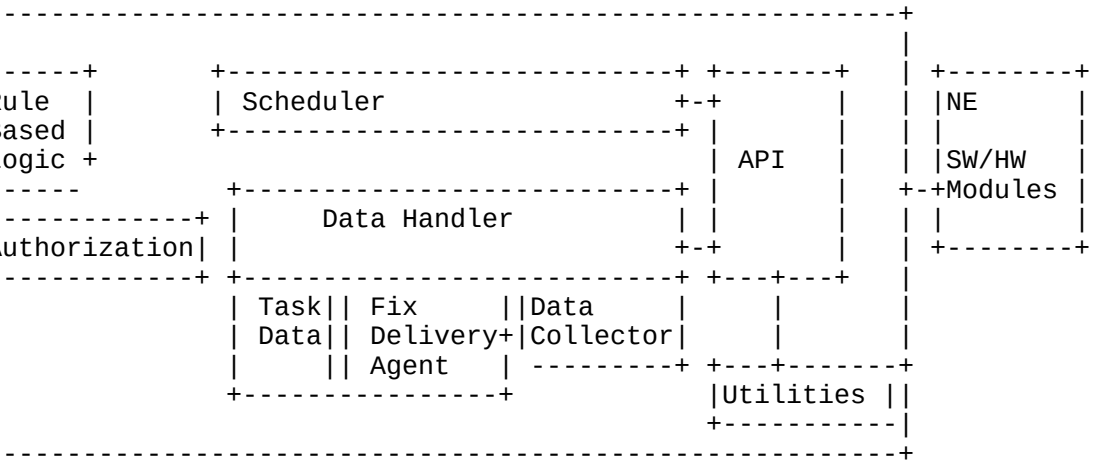


Figure 4: Intelligent Agent Architecture

5. Self and Centrally Managed Networks

A self and centrally managed network architecture consisting of self-managed NEs and self-managing NMS is depicted in Figure 5.

sNE related failures are handled locally by the sNE. If the problem is determined to be not fixable by the sNE after two or three tries or without a try, depending on the problem, a message is sent to the sNMS by the sNE indicating that the fixing entity is unidentified. If the problem is locally fixable, sNE send a message to SNMS, other sNEs, field technicians and users, indicating the fixing entity and how long the fix is going to take.

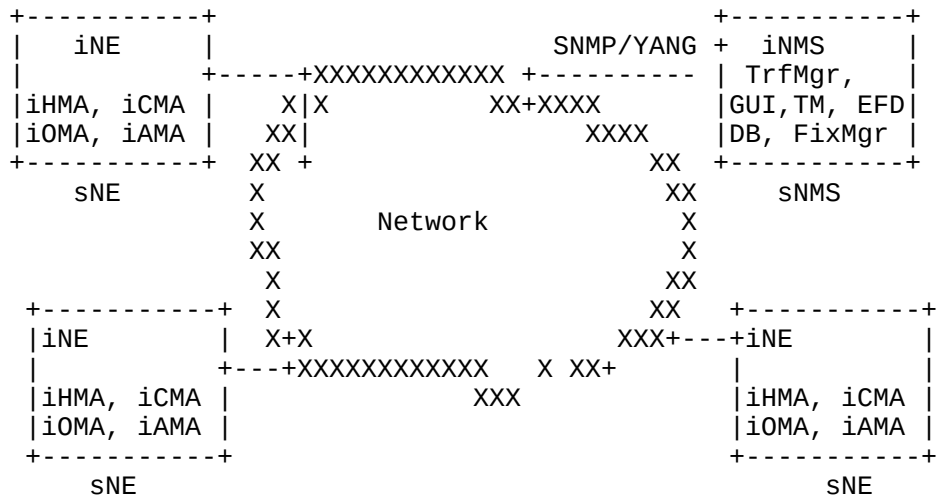


Figure 5: Self and Centrally Managed Network Architecture

6. Self and Distributedly Managed Networks

A self and distributedly managed network architecture is given in Figure 6. The network is divided into multiple regions where each region is managed by a self-managing NMS (sNMRn). One of the sNMRs in the network acts as the central sNMS. The regional self-managing sNMRs and central self-managing NMS are connected to each other via in-band and/or out-of-band communications facilities.

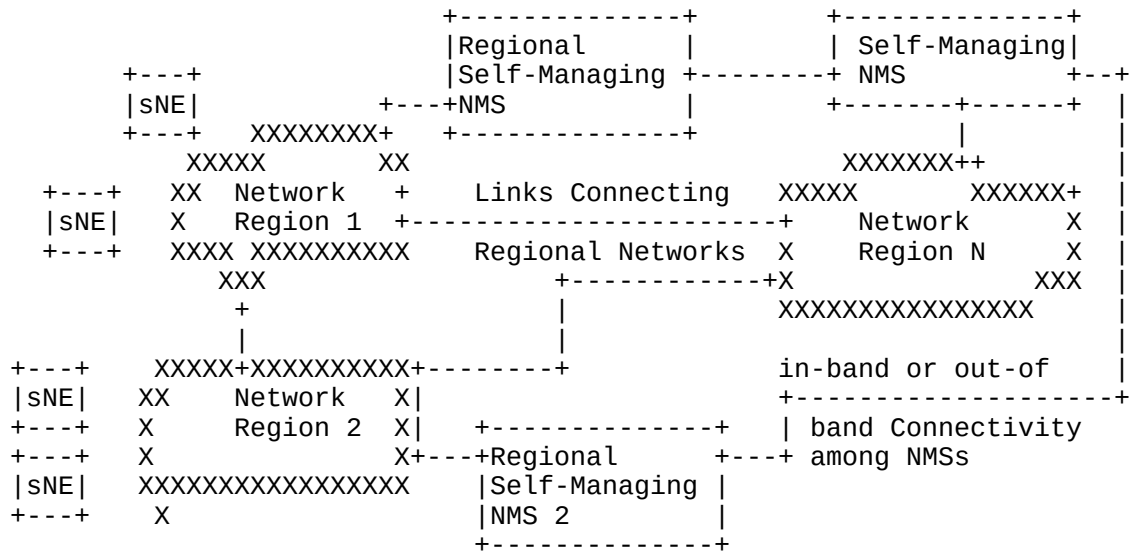


Figure 6: Self Distributedly Managed Network Architecture

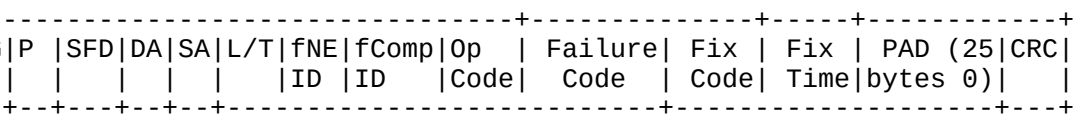
sNMRn provides all the centralized management functions for its own subnet and informs the central sNMS about its activities. End-to-end network level activities beyond region boundaries will be left to sNMS. These activities can be Connection Admission Control (CAC), load balancing, and congestion control at end-to-end network level.

7. In-band Communications of Failure types, Estimated Fix Time and Fix

In today's networks, failures related to equipment, ports and connections are mostly reported to an NMS via SNMP traps or in-band communications to NEs via AIS (Alarm Indication Signal), RDI (Remote Defect Indicator), Connectivity Check Message (CCM) related events such as Loss of Continuity (LoC) [Y.1731], etc. These alarms and traps identify the failed NE, port, or connection, but don't identify the component contributing to the failure. Furthermore, each has a different format.

For self-management, it is necessary to identify faulty components, estimate the time for fix, and communicate that to all parties involved (i.e. sNEs, sNMRn , sNMS, field technicians, and customers), so that working sNEs can store (if desired) data routed to the failed sNE(s) for the duration of fix or re-route traffic around the failed sNE(s) or port(s). For simplicity, all messages should have the same format.

Figure 7 depicts a possible Ethernet frame for Ethernet networks to carry all the information described above. Similar messages are to be created for other types of networks such as IP, MPLS and IMS.



IFG: Interframe Gap, 12 bytes
P/SFD (Preamble/Start of Frame Delimiter)-8 Bytes(P-7 bytes, SFD-1 byte)
L/T (Length/Type) : Length of frame or data type, 2 bytes (0x8808)
CRC: 4 bytes
DA: 01:80:C2:00:00:02 (6 bytes)-Slow protocol multicast address
fNE ID: 6 bytes, Failed sNE Identifier
fComp ID: 4 bytes, Failed Component Identifier
Op Code: 2 bytes-0x0202 for Disabled and 0x0303 for Enabled status
Failure Code : 4 bytes
Fix Code: 1 byte identifying fixing entity, NE (x00), sNMS (x01), sRMS (x02), sNMS-v (x03), RNMS-v (x04), sNMS-s (x05), sRNMS-s (x06), field technician (x07), unidentified entity or inconclusive diag(x08)
Fix Time: 4 bytes indicating fix time in seconds by NE, NMS, or field technician

Figure 7: Self-Managing message frame format for Self-managed Ethernet networks

For Ethernet networks, slow protocol multicast address can be used to inform sNEs, sNMS, and field technician devices connected to the network. fNE ID indicates MAC address of the failed sNE. fComp ID indicates the failed component identifier within the sNE. Op Code indicates whether the sNE or port is operationally disabled or enabled.

This operational status is disabled during failures and becomes enabled after the failure is fixed. Failure Code indicates failure type. If failure type is unidentified thru diagnostics, Failure Code will be unidentified or inconclusive or the failure is not-local to sNE. Fix Code identifies repairing entity whether it is sNE, sNMRn , sNMS, or a field technician.

It is possible to allocate six bytes to Fix Code field to indicate MAC address of the fixing entity. It is also possible to identify the failure type and not fix it. In this case, fixing entity is unidentified. It is also possible that both failure code and fix code are unidentified. Fix time indicates the estimated time in

seconds for repair which is set by the repairing entity. In order for sNE, sNMRn , or sNMS to provide the estimated fix time, the fix time for each type of failure needs to be stored in sNE and sNMRn or sNMS. If the failure is going to be fixed by a field technician, the technician may enter fix time manually into the related management system to communicate that to all related parties.

Given the sNMRn and sNMS interface uses network management protocols such as SNMP, the information in the message (Figure 5) needs to be conveyed to sNMS via an SNMP trap. Similarly the SNMP trap from sNMRn and sNMS needs to be converted into an in-band message to convey the information to self-managing NEs.

8. Failure Fixing Hierarchy in Centrally Managed Networks

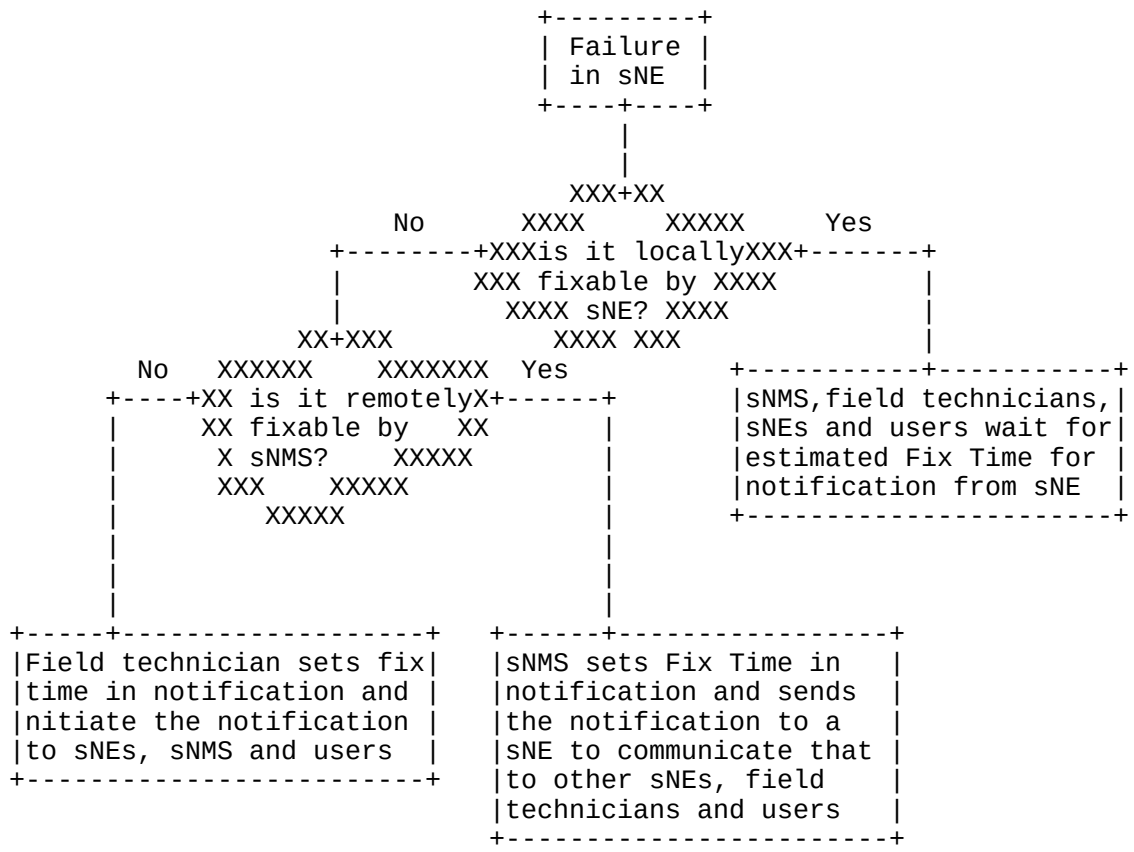


Figure 8: Fault Management Hierarchy for Self and Centrally Managed

Networks

In a centrally managed network, when there is a failure, sNE determines if the failure is local to the sNE or not. If the failure is local, then the sNE informs other sNEs, sNMS, field technicians and customers about failure type and fix time. If NE decides that the failure is not local to sNE, then sNE escalates the problem to the sNMS. The sNMS verifies that it is not local to the sNE and determines if it can fix the problem. If the sNMS can fix the problem, the sNMS communicates the failure type and fix time to sNEs, field technicians and customers. If the sNMS determines the failure is not fixable, the sNMS escalates the problem to field technicians. The field technician communicates fix time to sNEs, the sNMS and customers. After the fix is completed, the fixing entity initiates a self-managed notification with Enabled status (i.e. Opcode is set to Eanabled) to other sNEs, the sNMS, and customers. Both sNMS and field technicians use one of the sNEs to send notifications to the remaining interested parties.

The sNMS and field technician communicates failures and fixes via a message from the sNMS. If there is a node failure (i.e. sNE completely fails due to a power failure for example), neither the sNMS nor field technicians is able to communicate with the sNE. Therefore, the sNMS and field technicians would use another sNE to communicate the failure.

9. Failure Fixing Hierarchy in Distributedly Managed Networks

In distributed architecture, the network is divided into sub-networks (I.e. regional networks), where each sub-net has its own sNMRn .

sNMRn provides all the centralized management functions for its own subnet and informs sNMS about its activities.

End-to-end network level monitoring and problem fixing beyond regional boundaries are left to sNMS. These activities can be Connection Admission Control (CAC), load balancing, and congestion control at network level.

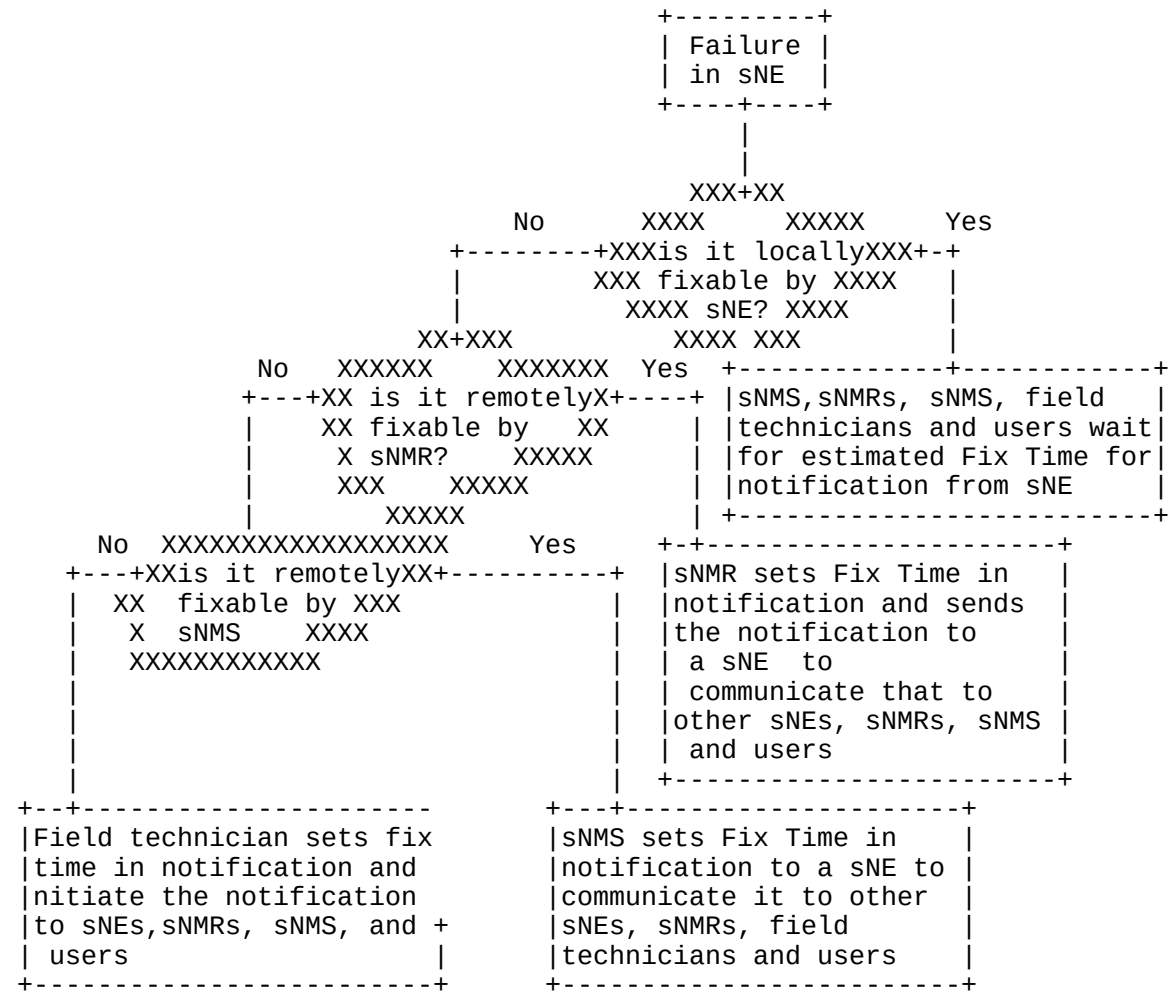


Figure 9: Fault Management Hierarchy for Self and Distributedly Managed Networks

10. Conclusion

Self-managed network concept for fault management, self-managed NE and self-managing NMS architectures, and a fault management communication mechanism for centrally and distributedly self-managed networks are introduced. A hierarchy for fault management for these networks are described.

11. Security Considerations

It is expected that all sNEs, sNMS, and sNMRn are authenticated during the network configuration manually or automatically. If there are security mechanisms established among sNEs, sNMS, sNMRn for exchanging messages, they would apply for exchanging the fault messages described here. There is no need for additional security procedures for the fault management messages described here.

12. IANA Considerations

This document does not request any action from IANA.

13. References

13.1. Informative References

[GANA] ETSI GS AFI 002 V1.1.1 : Autonomic network engineering for the self-managing Future Internet; Generic Autonomic Network Architecture, 2013-04

[SUSEREQ] ETSI GS AFI 001 V1.1.1 Group Specification Autonomic network engineering for the self-managing Future Internet (AFI); Scenarios, Use Cases and Requirements for Autonomic/Self-Managing Future Internet, 2011-06

[SELFMAN] Keller, Alexander; et al. (Eds.), Self-Managed Networks, Systems, and Services Second IEEE International Workshops, SelfMan 2006, Dublin, Ireland, June 16, 2006, Proceedings

[DPoE] E. Malette and M. Hajduczenia, Automating provisioning of Demarcation Devices in DOCSIS Provisioning of EPON (DPoE), IEEE Comm. Magazine, September, 2012

[SMN] M. Toy, Self-Managed Networks, Comcast internal document, November, 2012.

[SMCEN] M. Toy, Self-Managed Carrier Ethernet Networks, April 2014, MEF Meeting in Budapest, self-managed-networks-comcast-mtoy.pdf., https://wiki.metroethernetforum.com/display/OWG/New+Work

[Y.1731] ITU-T Y.1731, OAM functions and mechanisms for Ethernet based networks, 2008

Author’s Address

Mehmet Toy
Comcast
1800 Bishops Gate Blvd.
Mount Laurel, NJ 08054
USA

Email: mehmet_toy@cable.comcast.com