SERVICE ASSURANCE METHODS AND METRICS FOR PACKET SWITCHED NETWORKS

Summary of the Ph.D. Dissertation

by
Pál Varga

Supervisors:

Péter Tatai
and
Géza Gordos, DSc.

BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
DEPARTMENT OF TELECOMMUNICATIONS AND MEDIA INFORMATICS
2010.

Chapter 1

Introduction

Users take full advantage of network services nowadays, without even noticing the fact of accessing a computer network. Utilizing applications available at a remote host became as simple as running a program in the local desktop computer.

Operating and maintaining the servicing networks have remained the hidden functions of service providers and network operators. Network and service management has always been a complex task, even inside the operators' own network. Furnishing satisfactory services in a multi-provider environment is even more challenging. Under real-life conditions the user data can traverse the territory of numerous network operators while reaching the servicing node.

Assuring end-to-end service in a multi-provider environment requires appropriate methods and metrics for realizing service degradations and localizing their causes. These theses aim to point out the issues of assuring service quality for networked applications by proposing a service assurance architecture, measurement methods, and advanced metrics. The proposed framework fits into the prevalent code of practice, in which the network operators and service providers manage their own territory.

The advances of this dissertation lay in the integration of active and passive fault management methods; the enhancement of the traditional Quality of Service (QoS) measures; and the idea of correlating the service quality perceived by the user (Quality of Experience, QoE) with the analysis results of core network measurements. Furthermore, the new measurement and analysis methods and metrics are integrated into the current autonomous network management concept of Knowledge, Monitor and Action planes.

My dissertation focuses on network and service quality measurement methods and metrics, and the utilization of the analysis results in network and service management.

Chapter 2 presents the objectives of my research, followed by the methodology in Chapter 3, where some definitions are also provided. Chapter 4 presents the new results grouped into four theses.

Thesis 1 describes a complete service assurance framework that consists of five subsystems, namely (i) event notification collection, (ii) preprocessing, (iii) root cause analysis, (iv) data mining, and (v) advice and modification. The purpose of the framework is twofold. First, it can be used for fault recognition, localization, and correction. Second, when utilized as a service assurance tool, it can recognize the violation of service level agreements and pro-

pose corrective actions for their elimination. The framework includes a feedback mechanism for learning, based on the initial problem and the advice given at the end of the analysis process.

Thesis 2 proposes a unique, data driven root cause analysis method that can be used either as part of the service assurance framework, or as an independent technique. It is based on Petri-nets and follows the problem solving approach of human experts. The practical applications of Theses 1 and 2 are demonstrated through case studies taken from the fields of VoIP service management, and fault elimination in a multi-provider Ethernet environment.

Thesis 3 deals with the problem of network bottleneck detection based on passive measurements. The subject of this thesis fits in as an event notification input of the presented framework, since the bottleneck detection algorithm, when applied, outputs warnings about localized bottlenecks. The presented method utilizes both spatial and temporal aspects of traffic analysis. The new metric that has been found effective in bottleneck detection, is based on packet interarrival times (PIT) distribution – the actual metric is the fourth central moment (called kurtosis) of PIT. The effectiveness of the method and the metric has been validated through simulation and real-life measurements.

Thesis 4 puts the previous theses into the prospective of the Monitor, Knowledge, and Action planes – the threesome represents a current management concept of autonomous networking. Beside the traditional spatial and temporal analysis of the traffic, the ideas of traffic mix and traffic matrix are also introduced in this thesis, together with novel methods of their extraction.

Finally, some applications of my theses are described in Chapter 5. It is followed by cited references and the list of my own publications.

Chapter 2

Research Objectives

The objective of the dissertation is to provide a complex framework for network and service management that can be easily applied to practical environments, and as part of this framework, to provide new, passive methods for network bottleneck detection, to construct an efficient and fast root cause analysis method, to validate the performance of these methods and metrics in various networking fields, and to extend the methods of the framework with traffic analysis techniques also used in autonomous network management.

More exactly, the main tasks addressed in the dissertation are the following:

- to create a service assurance framework that takes all available input of the network from network and management prospectives and provides corrective action for network failures and service level violation issues;
- to model the problem solving behavior of networking experts and develop a fast and effective root cause analysis method based on the model;
- to create and validate a network bottleneck-detection method based on a limited number of passive measurements on aggregated links and spatial analysis of the traffic;
- to create and validate an effective bottleneck-detection metric based on temporal aspects of traffic analysis;
- to extend the service assurance framework in order to be used in autonomous network management, and map its tasks with the Monitor, Knowledge and Action Planes;
- to define traffic analysis mechanisms at the Monitor Plane, including spatial analysis, temporal analysis, calculating the traffic mix and compiling the traffic matrix.

For this purpose, I have carried out the following studies:

- analysis of the operations and maintenance (OAM) requirements and limitations of complex networking services such as high quality VoIP and multi-operator, end-to-end Ethernet;
- definition, creation and validation of the service assurance framework and its subsystems to solve the issues raised by service OAM analysis;

- measurements and analysis of the traffic at the academic backbone of BME and at several parts of the core network of two major Hungarian ISPs in order to identify the practical limitations introduced by data collection at high speed interfaces, encapsulated protocol messages and on-the-fly processing power of current CPUs;

- creation of a unique, Petri net-based Root Cause Analysis method that models the problem solving approach of human experts;

- creation and validation of the service assurance framework and the Petri-net based root cause analysis method for real life cases, including VoIP service assurance and multi-provider Ethernet service OAM

- temporal analysis of packet interarrival times in order to find patterns of bottleneck-behavior;

- evaluation of MGR-PS arrival model, and high order statistics of packet interarrival times (PIT) to correlate with bottleneck behavior and with Quality of Experience;

- definition of a bottleneck detection algorithm and successful application of the newly defined PIT kurtosis metric for bottleneck detection;

- QoE-related measurement and evaluation of bottleneck detection metrics, in order to determine whether they correlate with the experienced service quality;

- research and definition of traffic analysis algorithms at the Monitor Plane, including temporal and spatial aspects, application identification and traffic matrix decomposition.

My research is limited to the area of infocommunications networks. The results have not been verified at – and hance may not be applicable to – packet switched networks of other domains.

Chapter 3

Methodology

Network QoS, and after all, the users' QoE can be improved by effective control of the network processes and service status. In order to support this effort, a complex and integrated service assurance framework was needed to be developed. The historical background and the state-of-the-art description of the field is discussed in the full script of the dissertation.

Definition: Service Assurance – In this thesis a generalized meaning of Service Assurance (SA) is used, namely: it covers all the functions that help assuring fluent services over the managed network.

Effective performance management can not be operated without a standalone performance monitoring system that is physically independent from the managed nodes. The main function of these equipment is to capture traffic at the monitored interfaces, process the data, and provide statistics on the traffic properties. Such performance monitoring tools can report statistical data as well as provide events of crossing some thresholds via SNMP as well. The SA framework utilizes SNMP, Syslog and other standard status-reporting and -requesting methods.

Processing of raw, timestamped data gathered at live infocommunications network links has been carried out at frame-, packet-, and flow-levels (i.e. analyzing at Ethernet, at IP and at TCP layers, respectively) by me and my colleagues for various verification purposes. Furthermore, application layer analysis were necessary for QoE and traffic mix investigations. The objects and subjects of the actual analysis, as well as the environment for data collection are described in the dissertation together with the analysis results.

In Thesis 1. I created an SA framework that covers the issues faced by current network and service Operations and Maintenance Centers (OMCs). There is a unique Root Cause Analysis method proposed in Thesis 2., which can be used either as part of the SA framework or independently of it. These results were gained through *empirical* studies, supported by simulations and then real-life measurements.

One of the major root causes of service degradation are network bottlenecks. These can be caused by unplanned, high traffic composition at certain weak points of the topology. Such bottlenecks are hard to detect and even harder to find the root cause of their existence.

In Thesis 3. I researched and defined various bottleneck detection techniques based on passive monitoring, and compared their efficiency. My work was based on observations of bottleneck behavior through traffic analysis on the packet level. The created metrics – MGR–PS-based delay factor, the skewness and the kurtosis of the Packet Interarrival Time (PIT)

was deducted analytically, and also proven through simulations and real-life measurements.

The research on the Knowledge Plane involves the integration of experiences gained through network monitoring and traffic analysis together with the issues raised and answered by the service assurance framework. The results are put into the viewpoint of autonomous networking, paying special attention to tasks of the Monitor plane. This is summarized in Thesis 4.

Chapter 4

New Results

4.1 The Service Assurance Framework

Thesis 1 - Created and validated a unique, integrated service assurance framework for packet switched networks, which considers all types of incoming events for various networking service types; clearly defines and separates filtering, event correlation and root cause analysis functions; introduces a minimal set of event filtering methods that provides effective alarm generation [J2] [J4] [C4]

This section briefly describes a general framework for Service Assurance (SA) functions methods and activities. The framework is unique, since it covers all possible event notification sources, and suggests a processing methodology to eliminate faults and violations in the system, hence assuring its services.

The proposed service assurance framework should be considered as an umbrella, utilizing – beside others – Connectivity Fault Management (CFM), Performance Monitoring (PM) [ITU92] and SLS Verification [AETP04] tools, methods and metrics. The event notifications generated by these subsystems are analyzed by Fault Management (FM) functions, which are also part of the SA framework. This thesis describes the model from the event processing point of view (from the FM prospective).

SA Framework Overview

Thesis 1.1 - Defined a general and complete service assurance framework that covers event notification, preprocessing, root cause analysis, data mining and decision making functionalities, which is general in the sense that it can be applied to any networking service, and complete so it detects and pro-actively supports the elimination of any service assurance-related issue

---

The philosophy of the SA framework can be briefly summarized by following the standard flows of information at Figure 4.1. The details of each module are described in the dissertation.

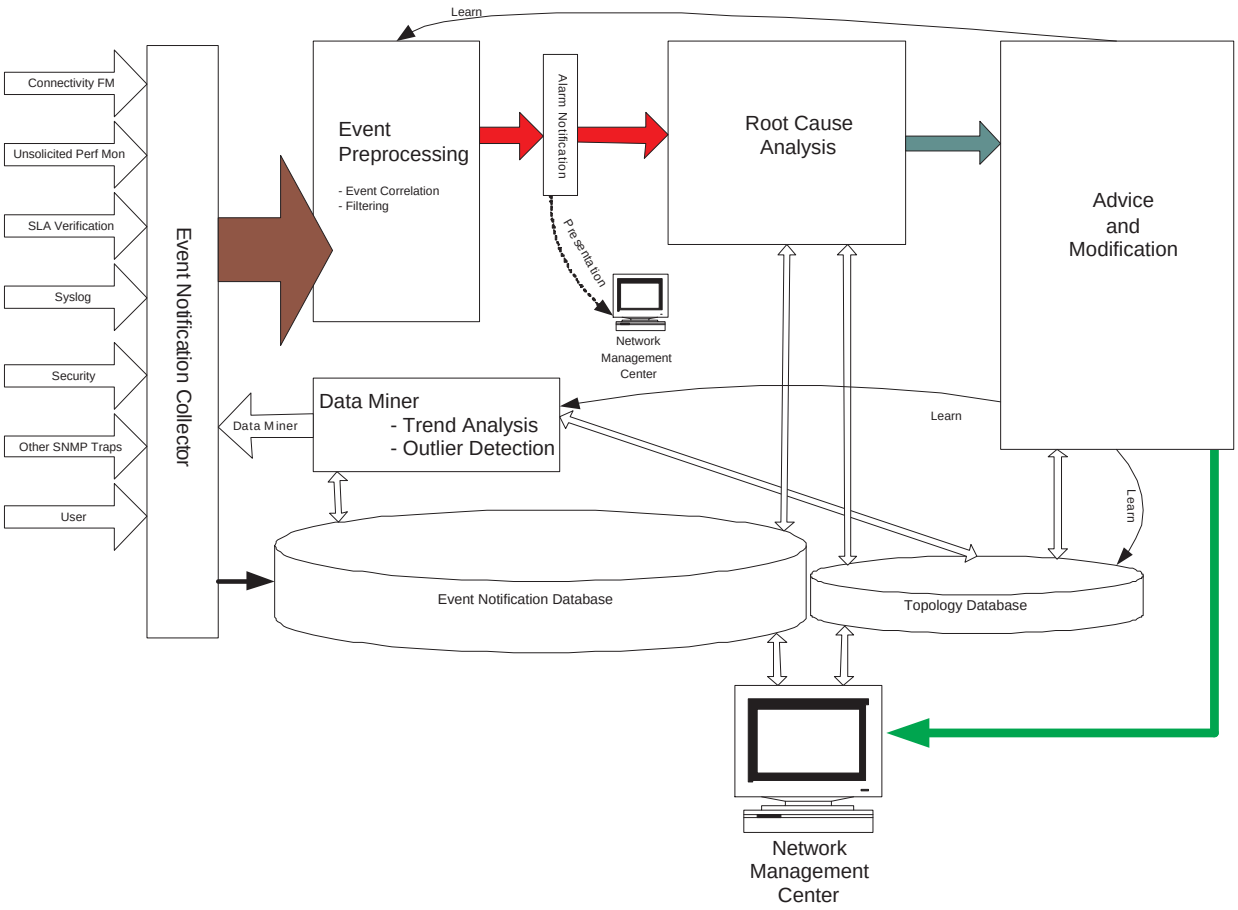


Figure 4.1: Connection of elements of the event processing and alarm handling model

As Figure 4.1 suggests, events arrive to the event processing and alarm handling model from various sources. The Event Notification Collector maps these to a standard format, then sends them to Event Preprocessing and stores them into the database. The Data Miner works on the stored events (seeing more historical events as well), and generates a new event when necessary. Once the events get *correlated* and *filtered*, only those alarms get presented to the NMC (Network Management Center), which should be acted upon. The Root Cause Analysis (RCA) for these alarms starts immediately. When the root cause is found, a fault description gets forwarded to the Advisor Module. This decides whether the corrective actions should be taken automatically, or only the advice (and the log about the tests carried out during RCA) should be forwarded to the human operator. The operator then decides what to do and he/she is also able to check if the automatic corrections taken were appropriate. The model includes learning mechanisms as well, which are indicated in Figure 4.1, as well.

The following sections briefly summarize how the events get generated and fed into this FM-flow, then detail the elements of the SA model.

Event Notification Collector

The event sources are all addressing their notifications to this entity. It collects the events sent in different formats and reshape them to a standard notification format. Events will be stored in this standard format into the Event Notification Database, and sent to the Event Processing module, as well. The proposed event-generator entities are listed at the left side of Figure 4.1, they are further described in the dissertation.

Trend Analysis and Outlier Detection

This module takes its input from the Event Notification Database. It continuously runs trend analysis algorithms and detects outliers that do not fit into periodic patterns on the data set and generates alarming events targeted to the Event Notification Collector.

Event Preprocessing

Thesis 1.2 - Created an event processing algorithm, which utilizes event correlation, event filtering, and root cause analysis functions and *efficiently* splits them – in order to populate events by correlation, allow filtering from a set of events with maximized population, and use root cause analysis merely for alarms rather than for events

---

Event correlation (EC) is a term generally used in the literature for algorithms that enable gaining extra knowledge of correlating events arriving from various sources rather than independently investigating them. EC requires further clarification in the current fault management context: in here, EC algorithms take Events as input, pass them through, AND generate extra events based on the matching EC rule. This approach allows event filtering to work from an enlarged knowledge base. Furthermore, well defined and focused correlation rules provide a good foundation for later RCA activities, since their result can explicitly indicate the place of the fault.

Event filtering by default suppresses low severity events, passes high severity alarms. A refinement of this approach is described in the following sub-thesis. Nevertheless, even this basic definition enables the "important", original events and the EC-based events to become Alarms.

Root cause analysis (RCA) is generally mixed with event correlation, which is wrong. EC does not show the Root Cause of a fault, but a highly sophisticated symptom of it. Another way to describe their difference is that EC takes events for input, and provide sophisticated events or alarms for output – whereas RCA takes alarms from input and provides the root cause of the fault as the output.

In this context, the most effective way to connect EC, filtering and RCA modules is exactly in this order.

In the implementation, the Event Preprocessing module consists of a Correlator and a Filter sub-module. The Correlator looks for correlations between events. This sub-module passes all event notifications (either "correlated" or simple) to the Filter sub-module. The Filter suppresses unnecessary event notifications. The events that pass Event Preprocessing are the actual Alarms, and are taken care of the RCA.

Correlation

To reduce complexity, the Correlator in this SA framework is a rule based event correlation module. If a correlation rule is matched, a new event notification is being generated.

Filtering

Thesis 1.3 - Researched event filtering sets and algorithms and defined a small set of rule-based filters – consisting merely of *counters*, *redundancy filters*, *dominance filters* and *suppress function* – that, when applied together with event correlation, significantly decreases the number of generated false alarms while allowing the valid alarms to get through towards the RCA

The utilization following four types of rule-based filtering methods – when combined with the output events of the event correlator – significantly decreases the alarm state-space, hence makes the later stages of alarm processing (RCA) more effective. Giving further, researched types of filter methods [Mei97] would not lead to more effective alarm generation. This is due to the fact that by default all low severity events get **suppress**ed, except the ones with high occurrence (see **counters**). On the other hand, all high priority events get passed, except with high occurrence (see **redundancy)** or in the historical presence of an even higher severity event (see **dominance)**. The counter function together with EC maximizes the number of populated events, whereas the three other suppress-type filters enable their reduction to the pure alarms to be handled by RCA.

- *Counters* pass an alarm through if the corresponding event arrives in a predefined number of times during the given time period. It is useful for low priority events that would be suppressed when arriving "alone", but should be acted upon when arriving in higher amounts (i.e., an unauthorized access attempt may indicate mistyped password, but hundred consecutive ones indicate a security problem);

- *Redundancy* filters pass the first occurrence of the alarm, and then suppress the other occurrences for a given duration. This type of filter is used for notifications of major and critical alarming events. They must be acted upon, but further notifications of the same kind are redundant, hence must be suppressed;

- *Dominance* filter needs two alarm IDs to be defined, A and B. It will pass A (higher severity), and then, during a predefined period suppress any occurrences of B (lower severity). Network Connectivity alarms (A) usually dominate service availability notifications (B);

- *Suppress* filter simply suppresses event notifications. Low severity event notifications (i.e., a Syslog message about a harmless configuration change), base events of correlated alarm notifications, alarming events with ceasing arrived - these are the basic event types that should be included in suppress filter rules.

As part of the SA framework, filtering out transient network failures comes from the very own nature of the filter module. It eliminates these transient conditions by means of a

verification period. If the reported condition goes away during the verification period, the module concludes that it is transient and ignores it.

Alarm Notification Presentation

The output of Event Preprocessing are the alarm notifications. Each alarm notification will trigger a new fault localization (root cause analysis) entity.

Root Cause Analysis

Based on the descriptive parameters of the alarm notification, this module tries to find the root cause of the problem. The RCA-related chapter of the dissertation briefly describes these methods and detail my second thesis group, an advanced, parallelized algorithm that models human expert behavior.

Actuators and Databases

Finally, **Advice and Modification** module gives final instructions for fault recovery, the **Event Notification Database** stores all event notifications sent towards the management system, and the **Topology Database** stores node addresses, types (functionalities), and the list of connecting nodes (first hops, subnet addresses).

4.2 Data-driven Root Cause Analysis

Thesis 2 - Developed a new data-driven, parallelized, Petri net-based Root Cause Analysis method and applied it to various cases of service assurance [J2] [J4]

This thesis group proposes a unique, data driven root cause analysis method that can be used either as part of the service assurance framework, or as an independent technique. It is based on Petri-nets and follows the problem solving approach of human experts.

The practical applications of Theses 1 and 2 are demonstrated through case studies taken from the fields of VoIP service management, and fault elimination in a multi-provider Ethernet environment.

Thesis 2.1 - Modeled the behavior of the human networking expert when searching for the root cause of a fault, and found that Petri-nets can be used for the representation of the parallelized action plan used by the human expert [J2] [J4]

System specialists make either conceptual or ad-hoc plans to find the root cause of an alarm. They decide - what measurements should be made, - what should be searched for inside event logs or other databases, and - what to do next depending on the result of these routines.

Consciously or not, experts fetch further input data (i.e., the IP address of a node, an interface number, etc.) to start some measurement or analysis processes. Simultaneous processes finish asynchronously, which forces the expert to make decisions on what to do

next: what kind of further checks can be initiated using the gathered input data? *Data driven* architectures are actually designed to address such issues, hence they could be used to mimic the simultaneous data processing capabilities of a human specialist.

An RCA algorithm aiming to simulate human expert behavior should satisfy the following requirements: - capability of extracting key parameters from the alarm description, - using this data, it should initiate active diagnostic checks, search routines, and correlation processes, - the initiation of independent checks, routines or processes should be made simultaneously, - once a result of a check is available, it starts new routines using the - new pieces of information until the root cause is found and no further checks are envisioned.

Flexibility and scalability issues lead to the "best practices" to human specialists: if there is only a low number of alarm types exist, it is sensible to create separate "action plans" for each alarm type. This action plan is going to be the base for the data-driven RCA descriptions. The most well-known data driven architectures are the Petri nets.

The Data Driven Root Cause Analysis Framework

Thesis 2.2 - Created and validated a unique data-driven, parallelized, Root Cause Analysis method that models the human expert behavior, and utilized Petri-nets for *actuating* the active RCA steps, as opposed to the well known practice of using them for *describing* event propagation in EC [J2] [J4]

There are known event correlation methods based on Petri nets [AFB+97]. However, they use this concept for modeling event propagation and use regular monitoring data for correlating events, rather than initiating active investigation checks.

Inside the SA framework, alarm notifications arrive to the RCA module from the event preprocessing module. Figure 4.2 depicts the internal structure of the RCA framework.

The root cause analysis sub-module takes each alarm separately and unleashes the hidden fault by going through an "action plan" provided by the RCA descriptor. The diagnostic elementary checks are parametrized by data taken from the alarm description, as well as connection information gathered from the topology database. Once the fault is localized and the root cause of the alarm is found, its description is passed to the advice and modification module.

Petri Net Based RCA Execution

The proposed RCA execution method schedules active checks, database searches and other processes depending on the availability of the input data of these routines.

Petri-nets in this case should be considered as "action plans" to find the root cause of an alarm. Each alarm notification has its own Petri net associated with it.

A Petri net consists of *places*, *transitions* and *directed arcs*, which connect the places with transitions. A transition can have input and output places, the naming convention depends of the direction of the arc between the given place and the transition. A transition gets activated if there are tokens in its every input place. In this case the transition *fires*: it (i) consumes the tokens from its input places, (ii) performs some processing task, and

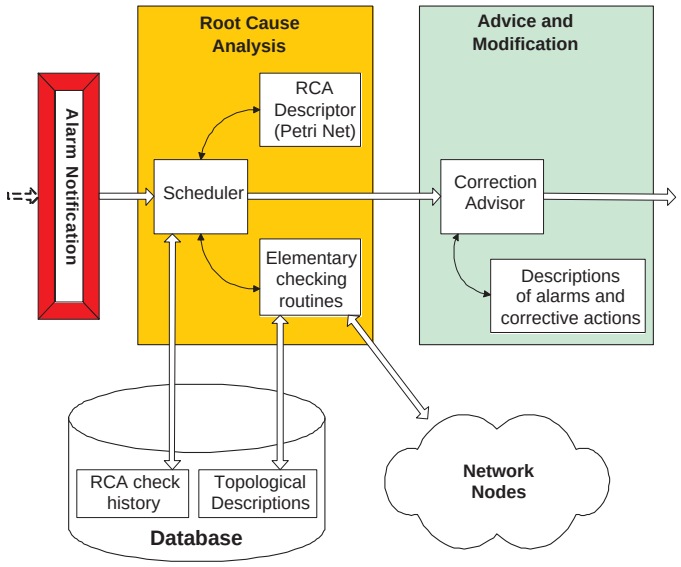


Figure 4.2: The architecture of the RCA model using active checks scheduled by Petri nets

(iii) places (in our RCA cases) one token into each of its output. These functions should be considered as taken in one, non-preemptible step.

Due to the non-deterministic nature of the Petri net, it is well-suited for modeling human expert behavior during fault localization and root cause analysis.

The active elementary diagnostic checks represent the *transitions* of the Petri net. Each elementary check has input parameters and output results. These represent the input and output *places* of the given transition.

A token marking a *place* means that the data associated with that place is available. This way all those transitions (elementary checks) can fire, for which the input places are all "tokened". In other words a diagnostic check will be initiated when all required input parameters are available. Since getting the result of a measurement or database search takes time, there can be numerous processes belonging to one alarm-analysis running simultaneously.

Based on the alarm type, the scheduler chooses an appropriate RCA descriptor (Petri net) for scheduling the analysis. This is depicted by Figure 4.2.

Thesis 2.3 - Created completely new, Petri net-based Root Cause Analysis descriptors for Ethernet OAM services and Voice over IP OAM services, which validate the feasibility and the practical usability of the earlier described Petri-net-based RCA-method [J2] [J4] [C4]

The following case studies are presented as part of the validation of the RCA-descriptor creation process. They also serve the purpose for better understanding of the RCA execution for the reader. There is one case presented in relation to end-to-end Ethernet service management, and another for VoIP service management.

Case 1: Ethernet Services, Connectivity Fault

Let us consider a basic connectivity error for the first case. It may be reported by the CFM as a connectivity fault. The Petri net representation of the RCA associated to this alarm is depicted by Figure 4.3.

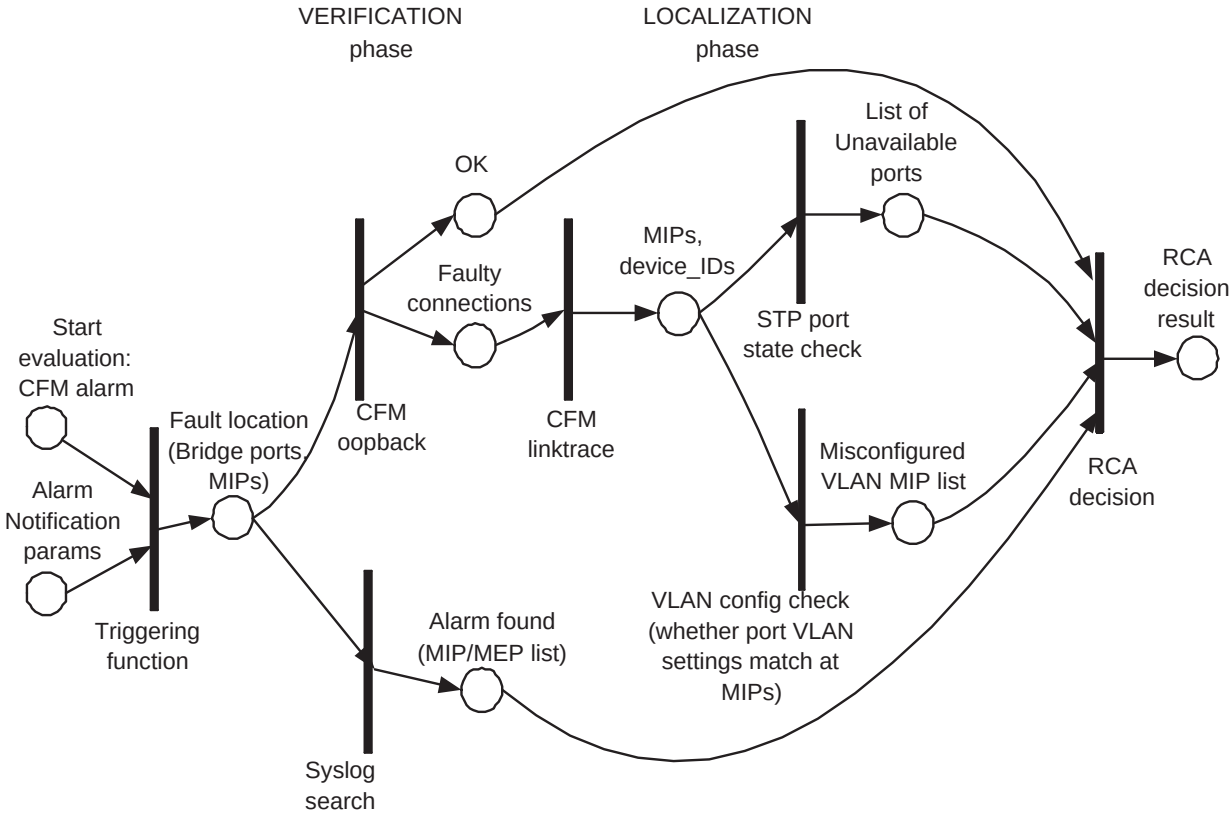


Figure 4.3: Petri net to drive the "CFM connectivity fault" alarm [J4]

The start of execution is triggered by a CFM alarm. The Petri net immediately branches to initiate active fault localization and passive database search.

After the parallel evaluation of the elementary checks, the decision of the root cause can be put further. If appropriate Syslog notification events were found, the exact fault is described in the result (i.e. interface down). If the fault verification returned the location of the fault, the VLAN and STP check results may clarify the root cause. The final output itself is a rule based decision, providing the fault description depending on the various test results.

Case 2: VoIP Services, High Packet Loss

This RCA case is taken from a complete fault management system created for VoIP service providers. One of the most complex RCA descriptions is related to the alarm notification of "high packet loss". Figure 4.4 depicts the corresponding Petri net.

The simultaneous execution of elementary diagnostic checks clearly reduces the time of the RCA execution – which affects the duration of the fault correction. A study on the

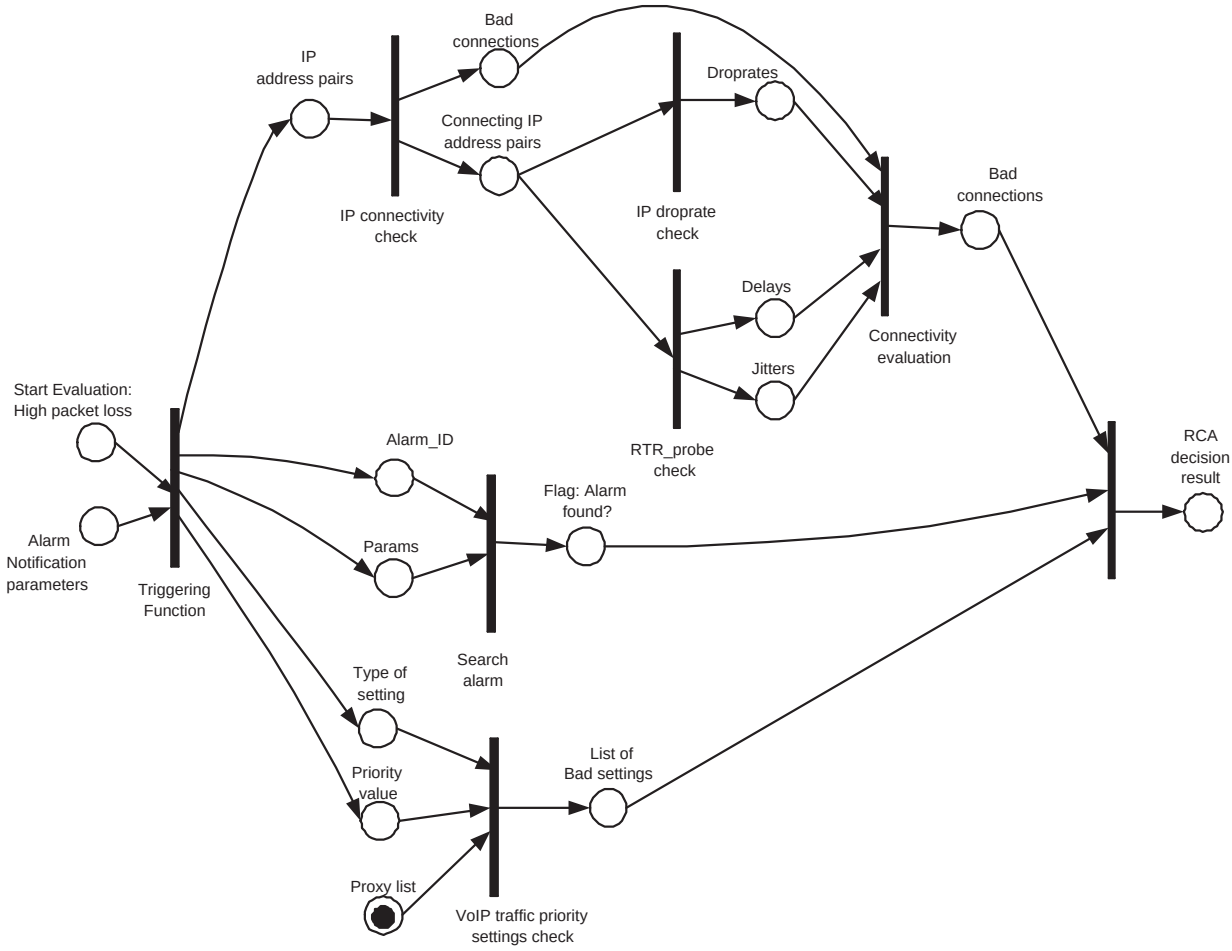


Figure 4.4: Petri net to drive the Root Cause Analysis of "High VoIP packet loss" [J2]

VoIP-tailored version of this RCA model has found that the average execution time of a complete, multi-step diagnostic process was cut to half due to the concurrent nature of the Petri net execution.

4.3 Events Generated by Detected Network Bottlenecks

Thesis 3 - Defined and validated a new bottleneck-detection method and a unique metric, which are much more effective than others used by passive measurements on aggregated links

---

This theses group describes a new method of determining network bottlenecks through passive measurements on aggregated network links, and a very effective, unique metric for bottleneck detection. Furthermore, the validation of the method and the metric are also described.

Definition: Bottleneck link – A link is considered as "bottleneck", where packets experience continuous, severe queuing and even being dropped due to the finite queue-

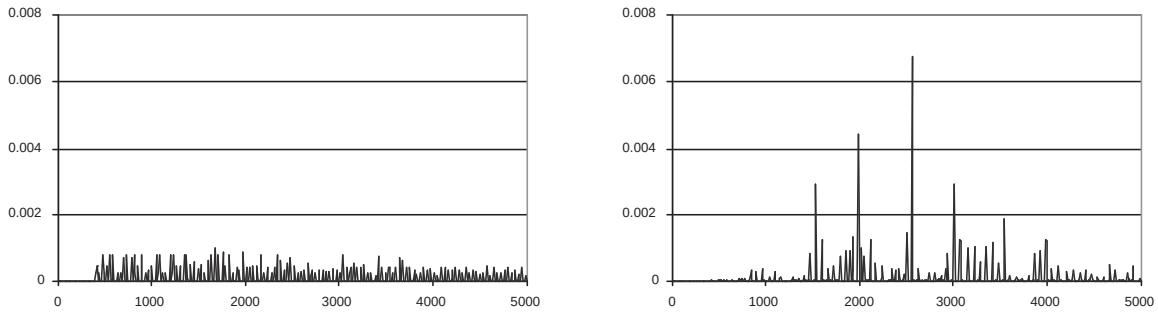
lengths. Let us take an other viewpoint: consider a user utilizing similar networked services on server a and on server b (the servers offer similar processing performance). The user can reach server a on route A, whereas he/she gets serviced by server b on route B. In case this user is satisfied with the network performance towards server a, but he/she can notice performance problems towards server b, then route B contains bottleneck link(s) – at least more of them, than scenario A does.

Thesis 3.1 - Defined a packet-level metric called "PIT-kurtosis" for bottleneck detection by passive measurements, and showed that other metrics earlier introduced to detect network bottlenecks are less effective for such purpose that PIT-kurtosis [J3] [C3]

Definition: Packet Interarrival Time (PIT) – The time difference measured at an observation point of a network link, between the arrival of the first bit of two, consecutive packets.

PIT should not be mixed with inter-packet gap (IPG), which is the time difference calculated between the last bit of a packet and the first bit of the following packet – i.e. this time duration is a ”silent gap”, where no packet-data is traversed on the link.

Packet interarrival time (PIT) distribution reveals hidden properties of bottlenecks. The more packets experience queuing at a network node, the more of them leave the node back-to-back. This behavior makes the PIT probability distribution function (PDF) less "flat", since spikes starting to appear at certain PIT values. As queuing turns into severe congestion, the spike around the lowest possible PIT value gets dominant. At an underutilized, aggregated link, where packets arrive from various sources, no high modus appear on the PIT PDF, which appears to be flat at the measurement point, as shown in Figure 4.5.a. On the other hand, when certain sources start to get congested, spikes appear at the certain interarrival times, similarly as shown by Figure 4.5.b. This peakedness is indicated by the fourth central moment - kurtosis - also: it gets more positive.



a. PIT PDF - no queuing

b. PIT PDF - slight queuing

Figure 4.5: Packet interarrival times PDF on an aggregated link with no queuing (left) and on an aggregated link, with eventual queuing (right) [J3]

Kurtosis, the fourth central moment of the PDF, characterizes the relative peakedness or flatness of a distribution compared to the normal distribution. Equation 4.1 shows the definition of "kurtosis excess" [KK51], which is widely used in the practice of mathematical

statistics. The outcome of this type of kurtosis is normalized for easier comparison with the normal distribution. "Kurtosis proper" is by definition the fourth central moment, and it misses the normalizing element of -3.

$$\gamma_2 = \frac{E[(\xi - E(\xi))^4]}{\sigma^4} - 3. \tag{4.1}$$

Theoretically what one expect during the observation of a link getting congested is the following. As the observed link starts showing bottleneck behavior, most of the eventual spikes of the PDF gets less noticeable and the spike around the lowest possible interarrival times starts dominating the PDF. Under severe congestion this spike fully dominates the PDF, as packets arrive back-to-back during the whole measurement period. When calculated for a severely congested link, PIT kurtosis exhibit significantly more positive values than for a non-congested link.

Thesis 3.2 - Validated the effectiveness of the PIT-kurtosis metric through simulations and measurements in real, operational links at ISPs [J3] [C3]

OPNET was chosen as a simulation tool, as previous work on bottleneck detection has proven its applicability. Traffic was simulated to traverse the network for a 15 minute measurement period. The traffic multiplexed at each ISP (aggregation node of an Internet Service Provider) is a mixture of service types such as e-mail, ftp, http and database-access.

As a rule of the thumb, it was found that **positive kurtosis suggests bottleneck behavior**, while negative kurtosis is a property of underutilized links. Kurtosis values being very close to zero is hard to evaluate, but certainly hide suspiciously serious queuing in the path.

In order to further validate metrics for passive bottleneck detection, a series of measurements have been taken place at sites of a major Hungarian network operator. During the measurements a passive network monitoring tool (Network Associates' Sniffer and Snifferbook Ultra) have been used, capturing continuous data traversed on Gigabit Ethernet interfaces. The measurements covered normal and busy hour conditions, connections to the operator's Internet Data Center and to routers/switches at the edge of the core network. The measurements captured traffic during severe bottleneck conditions also.

Table 4.1: PIT kurtosis values of measurements at the operator's site

Link name	normal conditions	OpR_1 overloaded
OpR_0 - $OpSW$	-0.46420	-0.45571
OpR_1 - $OpSW$	-0.06267	1.15119
OpR_2 - $OpSW$	-0.11180	-0.13289

Table 4.1 summarizes the results of PIT kurtosis calculations after a pair of measurements. These kurtosis values clearly meet the expectations set up earlier. Traffic measured under normal network conditions (underutilized links) provide negative values. During the congestion on this route (in this scenario not the monitored link, but an ATM link being two hops behind the monitoring unit has been congested), kurtosis reached the positive domain,

suggesting severe bottleneck condition. In this period noticeable packet loss and over 90% link utilization was observed as well (using monitoring tools covering the targeted, lower capacity links).

Thesis 3.3 - Created an analysis method to detect network bottlenecks at segments by passively measuring a further placed, aggregated link [C2]

The general idea of bottleneck detection by passive monitoring is limited by the practical approach of operators: covering the whole network with monitoring probes is simply not feasible. In practice there is usually very limited number (one or two) of high-speed aggregated links per network segment can be put under passive monitoring. Nevertheless, bottlenecks can be identified inside that segment, if the topology and the routing method is known. This leads to both spatial and temporal analysis of the data.

First, the traffic under analysis must be split into flows, and the general route have to be identified for each flow based on the topology and the routing rules (spatial analysis). Second, the interarrival time distribution has to be analyzed for the flows (temporal analysis).

When calculated on the aggregated link, the PIT PDF is always considerably flat according to the aggregation level. The mentioned dominant spikes become discernible only if the PITs are generated on *packets sorted by their source direction*. Practically that means a filtering for distinct directions (IP address spaces, MPLS tags, VPN IDs). The process can be repeated using finer filtering until the problematic link is found. All in our investigations the network topology (including router addresses as well as the utilized IP address spaces) was known.

Thesis 3.4 - Created and validated a new bottleneck-detection metric called delay factor, based on the M/G/R–PS arrival model [C2] [C8]

Delay factor of flows can be used as a key performance indicator supporting network-wide management tasks. This metric is derived by applying the M/G/R–PS model formula, which requires only flow interarrival time, flow size and link capacity information to be present. The original calculation method has some drawbacks because the data provided by the transport level flow definition does not fit easily to the boundary conditions of the M/G/R–PS model.

Delay factor (DF) is a metric originally defined for elastic flows. It indicates the relation between the average measured sojourn time and the ideal transport time of a flow with a given size in a network segment with a specified minimum capacity.

Delay factor can be derived from interarrival time of flows as well, with the help of M/G/R–PS model. According to the model, the transmissions will be served by a number of servers (R), which could be derived in the following way:

$$R = \lfloor C/r_{peak} \rfloor, \tag{4.2}$$

where C is the capacity of the specific aggregated links, and r_{peak} is the maximum transfer rate of the flows, determined by the access rates of the users.

According to the model the expected sojourn time of elastic flows in the M/G/R–PS system:

$$E\{T(x)\} = \frac{x}{r_{peak}} \left(1 + \frac{E_2(R, R\rho)}{R(1-\rho)}\right) = \frac{x}{r_{peak}} \cdot f_R, \tag{4.3}$$

where x is the size of the transferred flow, ρ denotes the utilization of the link, R is the number of servers and f_R is the delay factor. E_2 stands for Erlang's second formula. Utilization could be specially derived from the flow arrival rate (λ_e), the average flow size (x_{mean}), and the capacity (C) of that link [RPBP00]:

$$\rho = \frac{\lambda_e \cdot x_{mean}}{C}. \tag{4.4}$$

The M/G/R–PS model can be reduced to M/G/1–PS, if the r_{peak} peak rate is larger than or equals to the aggregated link capacity [RPBP00]. In our case this assumption is satisfied. According to this, the next expression – derived from Equation 4.3 – should be calculated for one server:

$$f_R = 1 + \frac{E_2(R, R\rho)}{R(1-\rho)}. \tag{4.5}$$

Erlang's second formula could be simplified further with the assumption of $R = 1$, since $E_2(1, \rho) = \rho$. This way Equation 4.5 can be calculated as follows:

$$f_1 = DF = 1 + \frac{\rho}{1-\rho} = \frac{1}{1-\rho}. \tag{4.6}$$

To validate the usability of the inter-arrival time-based delay factor in bottleneck-link detection, we carried out some measurements at a local operator as well, by capturing continuous traffic at a router exchanging data at the rate of 640 Mbit/sec (64% utilization). Having only limited information about the monitored network, we processed the routing table of the measured router and used a simplified, one-hop measurement topology. Altogether we collected and analyzed data from 11 links connected to internal domains of the operator.

Table 4.2: Results on Operator's Network based on delay factor

	linkA	linkB	linkC	lindD	linkE	linkF	linkG	linkH	linkI	linkJ	linkK
f_1	1.439	1.045	1.129	1.002	1.004	1.000	1.002	1.003	1.001	1.000	1.000
f_S	777.1	1084	813.6	759.9	697.3	1894	474.7	780.6	98.26	536.1	867.7

As we had limited memory allowance for the devices used at the measurement, the interval for capturing continuous traffic was limited as well. The measurement period was defined as the whole time of the measurement. In this way there was only one simple value for each link and for each method. The results are presented in Table 4.2, where f_1 stands for inter-arrival time based-, while f_S is sojourn time based delay factor.

According to results on f_1 we can say there were some bottleneck behavior towards *linkA* and *linkC*, where the delay factor values were 1.439 and 1.129 – these were the links routed towards MLLs (Managed Leased Lines). All the other links transmit data without any congestion, their delay factor values were very close to 1.

Concerning f_S metric delay factor is significantly higher than 1. Sometimes even the relatively loaded links have lower delay factor than unloaded ones. The explanation is that the flow sojourn times can range on a broad scale and faster flows can not balance the effect of some extremely slow flows, which indicates that this measure is not suitable for bottleneck detection because of its high sensitivity to delay variance.

After refining the calculation methodology of the delay factor metric by reshaping the data, we found the usage of the advanced algorithm described in [C8] gives much more precise results for bottleneck detection than the traditional method (described above). Nevertheless, this advanced algorithm is computationally intensive, and so far was not used during on-the-fly investigations.

Thesis 3.5 - Compared PIT-kurtosis to other possible bottleneck-detection metrics by passive measurements and found it the most effective [J6] [C5]

After the investigation of several metrics to be used in bottleneck detection by passive monitoring, we found that these work with different accuracy under different conditions. The combination of these metrics – *loss rate, speed averages, variance of the flow-level throughput, delay factors* – can make the final decision more accurate. Nevertheless, the simple usage of *PIT kurtosis* is less complex computationally, the metric is easy to evaluate, and provides more accurate result faster.

Detecting higher *loss rates* on specific links suggests tighter bottleneck. Because of its additive nature, loss rate cannot be used alone as a bottleneck indicator, rather as part of a complex, loss-based metric. Such technique would need a longer history of losses for the aggregated traffic, the number of flows and the throughput for all observed links during the measurement period.

The throughput-like metric, *speed averages* of flows – the actual mean of flow speeds measured on the link – can also suggest bottleneck under certain circumstances of balanced spatial characteristics of the traffic.

In case of normal operation (if there is no severe congestion along the path), the *variance of the flow-level throughput* for elastic traffic could be high on a link. Under bottleneck conditions we assume that elastic flows crossing the bottleneck share the bandwidth equally, thus their throughput would be similar, causing the throughput variance to be low. In theory, this is a promising, but still computationally expensive method, although it was found to be inaccurate during several real-life measurements.

Delay factor of a transport-level flow can be interpreted as the quotient of its measured sojourn time (average transmission time between source and destination) and its expected sojourn time. Unfortunately, although this metric was promising under simulated conditions, and some real-life measurements, it was found to be inaccurate during another set of the real-life investigations [J6].

Skewness, the third central moment of the PDF, characterizes the degree of symmetry - or rather, the asymmetry - of a distribution around its mean. Positive skewness indicates a distribution with a probability-peak on the lower values and an extending tail towards the higher values. When a link is congested, and packets follow each other back-to-back, the PIT PDF gets more skewed towards the lower PIT values – in such cases skewness take more and more positive values. For the simulation scenarios, skewness of PITs noticeably acquired positive values in bottleneck scenarios, although it had do so for links with eventual queuing, as well. For real-life measurements, skewness analysis resulted positive values for all cases. The bottleneck link provided higher PIT skewness value comparing to the others, but the difference was not significant, leaving skewness to be unable to detect bottleneck.

Thesis 3.6 - Described the relation of network bottlenecks and the Quality of Experience (QoE) and showed that high PIT-kurtosis values reveal poor QoE [C7] [J6] [C5] [C6]

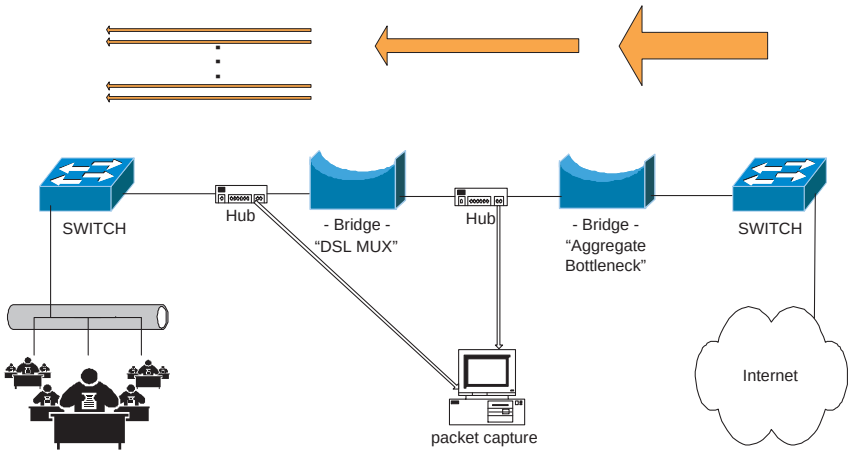
User perception of a networking service is usually very different from the operators' understanding of service usability. Quality of Experience (QoE) metrics are supposed to describe the service from the end-users' point of view - although QoE is hard to measure for mass services. Collection and analysis of QoS and SLS (Service Level Specification) properties of networking services are daily tasks of the operators. These metrics, however often provide misleading description of user satisfaction. The ultimate aim in this field is to find methods and metrics determining QoE by passive measurements on an aggregated network link. The experimental results of my research show that there is a correlation between the severity of a network bottleneck and the experienced service quality. During the real-life measurements, a core network segment was deliberately narrowed by us while it was loaded with various kinds of service requests. The degradation of perceived quality of various services (including audio streaming, P2P traffic, Skype traffic and web browsing) was notable – it is summarized in Table 4.3.

Available Bandwidth	**Video stream**	**Audio stream**	**Skype**	**P2P [kBps]**	**Web browsing**
7.0 Mbps	perfect	perfect	perfect	33-95	good
5.0 Mbps	occasional squares in picture	perfect	perfect	33-90	good
4.5 Mbps	playing pauses at key-frame changes	perfect	perfect	33-88	slow
4.0 Mbps	sec-long pauses at key-frame changes	perfect	perfect	12-14	slow
3.5 Mbps	bad; squares and pauses are regular	short pauses	scratching	3-10	unbearably slow
2.5 Mbps	very bad; not enjoyable	longer pauses	scratching	3	extremely bad

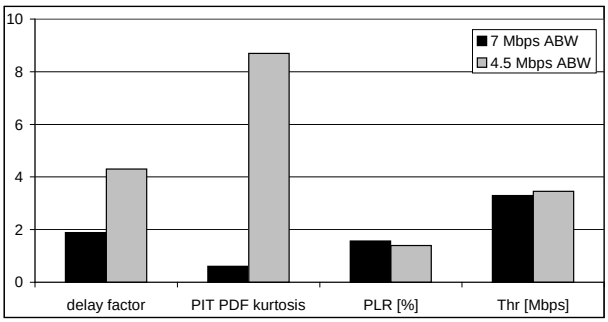
Table 4.3: Perceived quality of network services at different bottleneck scenarios

The packet level traffic was captured by passive monitoring equipment during these measurements. Some of the metrics based on packet interarrival times, packet size information and packet loss information has been calculated show strong correlation with service degradation.

Figure 4.6.a depicts the measurement scenario, including the artificially generated bottleneck and the measurement point. Figure 4.6.b introduces the results by presenting the most interesting of them, thus from the seven scenarios only two could be seen: the 7 and the 4.5 Mbps cases. When the available bandwidth (ABW) was 7 Mbps the network was not overloaded: all the applications worked properly, while at 4.5 Mbps the network load caused service degradations in case of some applications. The analysis have found that the calculation of an average packet loss ratio cannot reflect the severity of the congestions. Likewise the throughput metric: the measured amount of traffic was approximately the same in the two cases. In spite of these the average delay factor and much rather the kurtosis of PIT PDF shows a significant difference between them.



a. Measurement and related download volumes



b. Average values of metrics

Figure 4.6: Measurement setup and results for two available bandwidth scenarios [C7]

4.4 Connecting the Knowledge Plane with the SA Framework

Thesis 4 - Defined specialized monitoring services and researched new, effective traffic analysis methods – including Traffic Mix and Traffic Matrix – to serve the Knowledge Plane of autonomous networks [J5] [J7]

The presented Fault Management- and Performance Management-oriented approach of the introduced Service Assurance framework hide the fact that the SA framework can actually be used as a practical implementation of certain autonomous networking concepts. One of the philosophies that answer the issues raised by the requirements of self-management, self-configuration and self-optimization is the Knowledge Plane.

According to the main argument of [CPRW03], the users and the operators suffer from the lack of a serious, purposeful optimization effort in the traditional Internet. The transparent core has no knowledge about the data transported, and even if the intelligent edge nodes realize that there is a problem, the core might not be aware of what should be done.

The solution for gaining knowledge about network status and traffic characteristics is to gather and process such data, which then provide a basis to trigger corrective actions. The authors of [CPRW03] suggest to handle this knowledge in the Knowledge Plane (KP), an

abstract entity that completes a triad together with Data Plane and Control Plane.

The IST-MUSE project resulted in many ideas and implementations in relation to KPlane. Beside separating the Monitoring Plane from the KPlane in [VdMS+06], they further introduced the Action Plane (APlane). They also defined a Knowledge Base that is commonly reachable by KPlane, MPlane and APlane. Figure 4.7 depicts their connection and relation to the network.

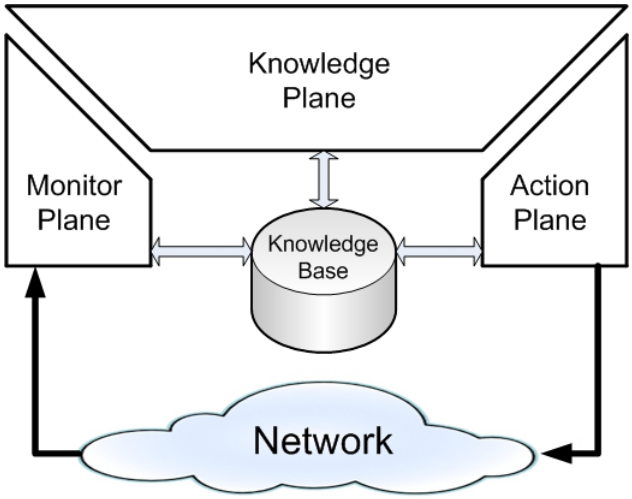


Figure 4.7: The relation of the Monitor, Knowledge and Action Planes to the Network [J7]

It is clear that the concept of the split Knowledge Plane is widely used in various levels of network and service management. Nevertheless, general traffic analysis is not yet utilized in order to support decision making in the KPlane. In the following theses (i) the relation between the elements of the SA framework and the split KPlane is described, (ii) a scalable traffic analysis concept is introduced for the MPlane, and (iii) the development details of extracting valuable information for the the traffic mix and the traffic matrix are also revealed.

Thesis 4.1 - Described the connection between the new service assurance framework and the current autonomous networking concepts of MPlane, KPlane, and APlane

---

The input of the SA framework is traffic and status information, whereas its output is advice of modification or the actual execution of modification steps. The same applies for the Knowledge Plane concept. The functions of the MPlane corresponds to the Event Notification Collector subsystem, its interfaces, and to the Event Preprocessing subsystem. The Root Cause Analysis and Data Miner subsystems are practical implementations of the actual KPlane functionalities. The APlane tasks correspond to the Advice and Modification subsystem and its interfaces. The Knowledge Base of the SA framework is the Event Notification Database and the Topology Database.

The core function of the MPlane is to provide complete and detailed view of the network and its services. Probes at every element (access nodes, routers, switches, content servers, links, etc.) monitor the element status as well as traffic parameters.

Although probe modules built into network nodes seem convenient for providing analysis data, they are by design incapable of sharing information with high granularity. Due to this

reason, passive probing is more desirable for traffic analysis of an extensive, heavily loaded network.

Thesis 4.2 - Defined a highly scalable, distributed data capture and analysis architecture for the MPlane, to be used even in high-speed, core network connections

---

Depending on the traffic volume, and the depth of the analysis, detailed traffic analysis can be carried out by one or many processors. In order to keep up with the ever increasing traffic and the demand for complex analysis, the processing system must be distributed to support the requirement of high scalability. This distribution has two levels: at the first level the **Capture probes** receive the packets, time-stamp, filter them, and based on the distribution law in the given environment, pass their essential information (i.e., the truncated headers) to the distributed processing units called **Monitors**.

The time-stamped, filtered, truncated packets must be processed by the Monitors in order to reveal network and service statuses. After the analysis, traffic information must be inserted into the Knowledge Base by the Monitors. Both packet- and flow-level analysis reveal important characteristics on losses, delays, and delay variations in the traffic, routing specialties, network structure changes and violations of the SLS (Service-Level Specification).

The Advantage of Monitoring at Highly Aggregated Connections

The inevitable function of the network monitoring probes is catching, filtering, and preprocessing the traffic. These tasks should be completed for the whole network. Since installing and maintaining such a monitoring network could be an enormous effort for the operator, introducing the MPlane at the highest aggregation parts (i.e. monitoring the fastest links) can be a good decision. Monitoring these relatively few points allows gathering all packets that traverse the network, although some locally looping traffic could be left out of the analysis.

Essential Functionalities of Capture Probes

- *Creating timestamps for the packets.* Time-stamping done by hardware (firmware) facilitates much more precision than by software, since it avoids possible latencies due to the operating system;

- *Filtering on hardware level.* High-speed traffic (i.e. currently 10 Gbps or above) presently allows no option for on-the-fly filtering in software. Clearly defined, low level filters are very useful: they can dramatically decrease the data to be analyzed;

- *Truncating incoming packets.* For the majority of the network analysis functions, statistics-counting, or fingerprint analysis, it is not necessary to use the whole IP packet, but the first portion of it;

- *Encapsulation and presentation of preprocessed data.* The resulting, digested packet information must be structured and packed when passed over to the Monitors.

The tasks of the Monitors in this architecture are the following:

- collection and decoding all the incoming information continuously (in 7/24 manner);
- checking filtering rules predefined by the network operator, execute conditional controlled orders/commands (i.e., conditional packet saving, alarming);
- structured data storage (i.e., raw data, statistics, assays, alerts);
- generation of packet- and flow-level counters on volume, loss, delay, delay variation;
- generation of specialized traffic reports, such as traffic mix and traffic matrix;
- database handling, remote access/query (i.e., Remote Capture, Session/Flow Trace).

Thesis 4.3 - Researched, chosen and applied efficient algorithms to gather and present traffic mix and traffic matrix analysis data for the KPlane

---

Traffic Matrix

Traffic Matrix is a network planning and development tool. During Traffic Matrix analysis, basic QoS statistics are periodically created on flow-level, and matched to originating and destination routes, network segments, or endpoint pairs (such as IP address(-range) pairs, MPLS tunnel endpoints, etc.). The first step of the analysis is determining the flows by an n-tuple (i.e., "5-tuple": from-IP, to-IP, from-port, to-port, protocol), and building/refreshing the flow-database. Once the targeted data structure is clarified, the algorithms of Traffic Matrix calculation are of low complexity. Such algorithms are described in [T3]. The result of the measurement can be used to display periodical statistics that support network planning or service marketing activities.

The actual Traffic Matrix can easily contain endpoint-pairs in the magnitude of 10e5. It is challenging to display such huge amount of data in a way that humans understand. While the raw results should be made available for reference in the Knowledge Base, some kind of data grouping should also be applied for visual presentation. One example of a good solution is to group the matrix elements into network segments, based on their destination addresses. The aim of the grouping algorithm is never to display high, invisible amount of segments (e.g. more than 15). When the operator wishes to peek inside a segment's statistics, he/she get it displayed as a deeper layer of the matrix. This way the calculated QoS parameters show up in an aggregated manner in the segment-to-segment relation. If the system allows manual definition of segment-creation rules, operators can gather valuable information by grouping their endpoints into various segments.

Traffic Mix

Traffic mix analysis is the classification of traffic flows into application types, and then evaluating these for the service parameters important for the given application type. Flows are classified by means of statistical indicators and, if necessary, behavior heuristics. Currently operators are mostly interested in the determination of the application mix for video stream, video conference, audio stream, VoIP, and Peer-to-peer (P2P). An application belonging to

a traffic-class can be identified by using static identifiers (e.g. port-based), dynamic identifiers (e.g. changing ports, fingerprints) or by applying packet-level, temporal and spatial statistics-based evaluation methods. In my traffic analysis practice, the powerful identification methods for VoIP, video and P2P applications are described in [BMM+07], [T2], and [KBFc04] respectively.

Once a traffic flow is identified (i.e., based on 5-tuple), various metrics are calculated in order to help identifying the traffic-class. These metrics are the following:

- *throughput* - transferred data bytes per second;
- *packet loss* - the rate of received packets and total transmitted packets in a given time interval, or during the connection;
- *packet delay* - depending on the network topology and link load it takes a certain amount of time to receive a packet after it was sent; there is also a gap (delay) between packets on the wire;
- *packet delay variation* - network load is not always static: as conditions and usage changes over time, packet delay changes as well - this is called packet delay variation;
- *round-trip-time* - interactive applications require fast replies, which can be characterized with this parameter;
- *out of order/duplicated packets.*

Chapter 5

Results Applied in Practice

The Service Assurance Framework, which is described in Thesis 1, was applied and implemented as a prototype for VoIP by Ericsson, Kovax and BME-TMIT as part of the IKTA-00092-2002 program. The framework was generalized afterwards, and applied for multi-operator Ethernet services, as well. This latter work was part of the IST-MUSE project, an integrated research initiative, involving "almost all major players in Europe in the area of broadband access".

The Petri net based RCA method, which is described in Thesis 2, is a major component of the Service Assurance Framework. It was used similarly in the IKTA and IST-MUSE programs, both for finding the root cause of VoIP service degradations and of multi-operator Ethernet connectivity faults and service level agreement violations. As part of their application, the routines for Petri-net execution were implemented, various Petri-net descriptors had to be developed and configured (to fit the actual RCA routine), and, naturally, all the customized elementary check routines related to VoIP and end-to-end Ethernet faults had to be implemented.

The passive monitoring-based method for bottleneck detection, which is described in Thesis 3, was first applied in BME-TMIT and in the NTT-DoCoMo research lab, in parallel. While simulations show the effectiveness of the MGR-PS-based delay factor metric, real time tests at Magyar Telekom's sites were not always convincing. Later, when techniques based on PIT skewness and most importantly, PIT kurtosis were developed and implemented, they were applied to reveal real-life network bottleneck detection issues. PIT kurtosis was found very effective during these set of measurements and analyses. Furthermore, the bottleneck detection method and the calculation algorithm of PIT kurtosis got integrated into the SGA1GEM, which is a network QoS testing environment with Gigabit Ethernet capabilities.This product was developed during the Jedlik research and development program, NKFP2-00015/2005.

An implemented example of the distributed Traffic Analysis concept at the MPlane includes the SCALOPES C-board (as Capture probe) and distributed Monitors. In this concept, the distinct analysis tasks – such as flow separation, application identification, QoS-related parameter calculation per flow/application/route – are managed by separated modules, so the parallel tasks can be run on distinct processors in the same time. Moreover, the inactive modules can be turned off to save power. The C-board has been developed as part of the ARTEMIS SCALOPES research project [T3], and it is a standalone, FPGA-

based hardware, equipped with 2x 10 Gbps Ethernet interfaces and 16x 1 Gbps Ethernet interfaces. When used as part of the Monitor plane, it is also preprocessing the packets, but rather than passing their data to one CPU, it distributes them among many Monitor units through its 1 Gbps Ethernet interface. The standalone Monitors then carry out traffic analysis, and present the results to the knowledge base. These traffic analysis include traffic mix and traffic matrix calculations, which are also described in Thesis 4. This concept is also applied in the EUREKA project CELTIC TIGER2, where the correlated management of IP, GMPLS and Ethernet procedures are supported by decisions at the Knowledge Plane, which uses the traffic analysis results provided by the Monitor Plane.

Acknowledgements

I would like to express my gratitude for all people who have supported my research. My special thanks go:

- to Péter Tatai, who provided me with endless technical ideas together with motivation during my studies and work as an engineer,
- to Géza Gordos, who introduced me to the field of communications engineering, for his continuous support during my studies,
- to Gyula Marosi, whose engineering qualities and personal encouragement helped me push for further limits,
- to István Moldován and Gergely Kún, László Osváth and József Bíró for the long discussions and over many interesting research topics, including fields of telecommunication engineering, service and network management, and queuing theory,
- to all the colleagues at Ericsson, Tecnomen, AITIA and BME, for the challenges they let me face with as an engineer, and sometimes even as a person,
- to J.P. Martin-Flatin, Aiko Pras, Gábor Vattay, János Végh, and the to the many anonymous reviewers for their helpful suggestions and critical comments on my research,
- to last but not least, my family, for their love, and for all the wonderful possibilities I have.

Thank you.

References

[AETP04] A. Asgari, R. Egan, P. Trimintzios, G. Pavlou, Scalable monitoring support for resource management and service assurance. In *IEEE Network*, November/December2004.

[AFB+97] A. Aghasaryan, E. Fabre, A. Benveniste, R. Boubour, and C. Jard, "A Petri net approach to fault detection and diagnosis in distributed systems," in *Proceedings of the 36th IEEE Conference on Decision and Control, IEEE CDC'97*, San Diego, CA, USA, 1997.

[BMM+07] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli. Revealing skype traffic: When randomness plays with you. In *SIGCOMM*, Japan, 2007.

[CPRW03] D.D. Clark, C. Partridgeo, J.C. Ramming, and J.T. Wroclawski. A knowledge plane for the internet. In *Conference on Applications, technologies, architectures, and protocols for computer communications*, Karlsruhe, Germany, 25–12 August 2003.

[IEE07] IEEE standard for local and metropolitan area networks - virtual bridged local area networks - amendment 5: Connectivity fault management, IEEE802.1ag, 2007.

[ITU92] ITU-T Recommendation X.700 - Management Framework for Open systems Interconnection (OSI) for CCITT Applications, 1992.

[KBFc04] T. Karagiannis, A. Broido, M. Faloutsos, and Kc claffy. Transport layer identification of p2p traffic. In *SIGCOMM*, Sicily, Italy, 2004.

[KK51] J. F. Kenney and E. S. Keeping. *Mathematics of Statistics*, volume 2. Princeton, NJ: Van Nostrand, 2nd edition, 1951.

[Kle75] L. Kleinrock. *Queuing systems, volume 1: Theory.* John Wiley and Sons, Inc., 1975.

[Mei97] D.M. Meira A Model For Alarm Correlation in Telecommunications Networks Dilmar Malheiros Meira, 1997.

[RPBP00] A. Riedl, M. Perske, T. Bauschert, and A. Probst, *Dimensioning of IP access networks with elastic traffic*, First Polish-German Teletraffic Symposium (PGTS 2000), Dresden, Germany, 2000.

[VdMS+06] B. De Vleeschauwer, W. Van de Meerssche, P. Simoens, F. De Turck, B. Dhoedt, P. Demeester, E. Gilon, K. Struyve, and T. Van Caenegem. On the enhancement of qoe for iptv services through knowledge plane deployment. In *Broadband Europe*, Geneva, Switzerland, 11–14 December 2006.

[ZO98] A. P. Zwart, O.J. Boxma, *Probability, Network and Algorithms (PNA), chapter Sojourn Time Asymptotics in the M/G/1 processor sharing queue*, PNA-R9802 ISSN 1386-3711, 1998.

Publications

[J] **Journal Papers and Book Chapters**

[J1] P. Varga, I. Moldován, and G. Molnár. Komplex hibamenedzsment megoldás VoIP-szolgáltatások felügyeletéhez. *Híradástechnika Folyóirat*, 60,10:50–55, October 2005. selected papers.

[J2] P. Varga, I. Moldován, and G. Molnár. Complex fault management solution for voip services. *Infocommunications Magazine*, 60,12:15–21, December 2005. selected papers.

[J3] P. Varga. Analyzing packet interarrival times distribution to detect network bottlenecks. In *EUNICE 2005: "Networked Applications" 11th Open European Summer School.*, pp.17–29. Springer, 2006.

[J4] P. Varga and I. Moldován. Integration of service-level monitoring with fault management for end-to-end multi-provider ethernet services. *IEEE Transactions on Network and Service Management*, 4:28–38, 2007.

[J5] P. Tatai, P. Varga, and Gy. Marosi. Távközlő hálózati folyamatok monitorozása. *Híradástechnika Folyóirat*, 62,8:49–55, August 2007.

[J6] P. Varga, G. Kún, and G. Sey. Towards estimating quality of experience with passive bottleneck detection metrics. In *Advances in Information Systems Development: New Methods and Practice for the Networked Society*, pp.115–125. Springer, 2007.

[J7] P. Varga and L. Gulyás. Traffic analysis methods to support decisions at the knowledge plane. *Infocommunications Magazine*, 65,10, October 2010.

[J8] P. Varga, I. Moldován, D. Horváth, and S. Plósz. A low power, programmable network platform and development environment. In *Advances of Network- Embedded Management and Applications*, Chapter 2., pp.19–36. Springer, 2010.

[J9] P. Varga, A. Kőrösi, A. Gulyás, and J. Bíró. Stochastic Bounds on the Expected Loss Ratio in Deterministic Queuing System Models. *Publicationes Mathematicae Debrecen.* to appear.

[C] **Conference and Workshop Papers**

[C1] P. Varga, G. Kún, P. Fodor, J. Bíró, D. Satoh, and K. Ishibashi. An advanced technique on bottleneck detection. In *IFIP WG6.3 Workshop, EUNICE 2003*, Balatonfüred, Hungary, 2003.

[C2] P. Varga and P. Tatai. Advanced Methods in GPRS Network Analysis. In *IFIP WG6.3 Workshop, EUNICE 2004*, Tampere, Finland, 2004.

[C3] P. Varga and G. Kún. Utilizing higher order statistics of packet interarrival times for bottleneck detection. In *IFIP/IEEE Workshop on End-to-End Monitoring Techniques and Services (E2EMON)*, Nice, France, 2005.

[C4] P. Varga, I. Moldován, and G. Molnár. Voip szolgáltatások hibamenedzsmentje. In *14. NIIF Networkshop*, Szeged, Hungary, 2005.

[C5] P. Varga, G. Kún, and G. Sey. A qos mérőszámok és a felhasználói elégedettség közötti kapcsolatok. In *15. HTE Távközlési és Informatikai Hálózatok Szeminárium*, Eger, Hungary, 2006.

[C6] P. Varga, G. Kún, and I. Moldován. Correlation of advanced bottleneck metrics and quality of experience. In *9th IEEE/IFIP Asia-Pacific Network Operations and Management Symposium, APNOMS*, Busan, Korea, 2006.

[C7] P. Varga, G. Kún, G. Sey, I. Moldován, and P. Gelencsér. Correlating user perception and measurable network properties: Experimenting with qoe. *Autonomic Principles of IP Operations and Management*. Lecture Notes in Computer Science, LNCS 4268, pp.218–221, Springer, 2006.

[C8] G. Kún, and P. Varga. Utilizing MGR-PS Model Properties for Bottleneck Characterization. In *World Telecommunications Congress, WTC 2006*, Budapest, Hungary, 2006.

[C9] P. Varga, J. Bíró, M. Martinecz, and Z. Heszberger. QoS Provision for Bufferless Statistical Multiplexing. In *International Conference on Telecommunication Systems, Modeling and Analysis: ICTSM 2009.*, Dallas, TX, USA, 2009.

[C10] S. Plósz, I. Moldován, P. Varga, and L. Kántor. Dependability of a network monitoring hardware. In *3rd IARIA International Conference on Dependability, DEPEND 2010*, Venice, Italy, 2010.

[T] **Technical Papers**

[T1] P. Varga, I. Moldován, T. Dinh Dang, Cs. Simon, G. Kún, and P. Tatai. Developing a passive measurement-based methodology for detecting network bottlenecks in ip networks. Technical report, Study for Hungarian Telecom, in Hungarian, 2004.

[T2] P. Varga, L. Kovács, I. Moldován, A. Illés, G. Kún, G. Sey, and G. Turzó. Analysis of media communication over the internet. Technical report, BME Report for Hungarian Telecom, 2007.

[T3] P. Varga, I. Moldován, G. Kródi, G. Sey, L. Kovács, D. Horváth, and S. Plósz. Report on new architectural platform and specification of example sw code for analysis. Technical report, ARTEMIS SCALOPES Deliverable DA1.3., AITIA, BME, 2010.

[R] **Academic works which cited my publications**

[R1] C. Marquezan, A. Panisson, L. Granville, G. Nunzi, and M. Brunner. Maintenance of monitoring systems throughout self-healing mechanisms. *Managing Large-Scale Service Deployment*, 20,2:57–70, Lecture Notes in Computer Science, LNCS 5273, pp.176–188, Springer, 2008., cited [J4]

[R2] C. Jagadish and T. A. Gonsalves. Distributed control of event floods in a large telecom network. *International Journal of Network Management*, 20,2:57–70, John Wiley and Sons, 2009., cited [J4]

[R3] M. Miyazawa and T. Otani. Real-time root cause analysis in OSS for a multi-layer and multi-domain network using a hierarchical circuit model and scanning algorithm Source. In *11th IFIP/IEEE international conference on Symposium on Integrated Network Management*, IEEE Press, 2009., cited [J4]

[R4] N. Pogkas, N. Ploskas, G. Panagopoulos, and N. Fleischer. VICTORY open-source middleware framework and prototypes of P2P network entities. *Deliverable 3.3, IST-VICTORY*, 2009., cited [C7]

[R5] E. Ibarrola, F. liberal, I. Taboada, and R. Ortega. Web QoE Evaluation in Multi-agent Networks: Validation of ITU-T G.1030. In *5th IARIA/IEEE International Conference on Autonomic and Autonomous Systems*, Valencia, Spain, 2009., cited [C7]

[R6] O. Dini, P. Lorenz, A. Abouaissa, and H. Guyennet. Dynamic Feedback for Service Reputation Updates. In *6th IARIA International Conference on Autonomic and Autonomous Systems*, Cancun, Mexico , 2010., cited [C7]