Deliverable D3.7

Adaptation of learning and operation methods to specific needs of future networks and services

Grant Agreement	257513	
Date of Annex I	25-07-2011	
Dissemination Level	Public	
Nature	Report	
Work package	WP3 – Network Empowerment	
Due delivery date	01 September 2012	
Actual delivery date	17 September 2012	
Lead beneficiary	NEC	Johannes Lessmann Johannes.Lessmann@neclab.eu

Authors	UniS: Stylianos Georgoulas, Majid Ghader ALBLI: Rouzbeh Razavi INRIA : Rémi Badonnel, Martin Barrere, Olivier Festor FT: Zwi Altman, Richard Combes TID: Beatriz Fuentes, Carolina García-Vázquez UPRC: Kostas Tsagkaris, Panagiotis Demestichas, Vera Stavroulaki, Panagiotis Vlacheas, Yiouli Kritikou, Nikos Koutsouris, Aimilia Bantouna, Dimitris Karvounas, Evagelia Tzifa, Assimina Sarli, Marios Logothetis, Andreas Georgakopoulos, Louiza Papadopoulou, Vassilis Foteinos, Dimitris Kelaidonis, George Poulios NEC: Johannes Lessmann, Zarrar Yousaf TIS: Antonio Manzalini, Giorgio Calochira UT: Ramin Sadre, Anna Sperotto ALBLF: Leila Bennacer, Laurent Ciavaglia NKUA: George Katsikas, Panagiotis Spapis and Nancy Alonistioti

Executive summary

This deliverable builds upon the work in task T3.3 previously presented in the deliverable D3.2, and focuses on the various techniques within the realm of observation and action. In addition to presenting the progress made in the techniques reported in the first year, the deliverable places the techniques together in the context of integration within the Unified Management Framework (UMF), the framework and integration structure developed in WP2 for unifying management functions that targets the embedding of autonomic paradigms in networks. As the work that has been done in T3.3 relates to quite diverse application contexts and is as such unsuitable to be combined into a single overall storyline, this deliverable is structured in terms of use cases as defined in WP4.

Thus, the chapters 2 to 7 each pertain to a specific WP4 use case. Each chapter lays out the details of the work done on the different techniques (also called Network Empowerment Mechanisms or NEMs, for short) related to the given use case. The context, content and merits of each of the NEMs are given. Emphasis was put on substantial evaluation results and the progress beyond previously reported work. At the end of each chapter, a use case wide discussion is presented that relates the NEMs of a use case in terms of potential interworking, and explains the usefulness of UMF for such a collaborative environment.

The presented techniques cover a significant spectrum of observation related domains (discovery, aggregation, diagnosis, mining, and (self-) modelling) and methodologies (fuzzy logic, self-organizing maps, reinforcement learning, case-based reasoning, Bayesian networks and statistical analysis). In Use Case 1 (“Self-diagnosis and self-healing for IMS VoIP and VPN services”), methods for root cause analysis (with much improved accuracy), anomaly detection (with high effectiveness while avoiding manual parameter tuning), congestion prediction (highly expressive and proactive instead of reactive), flow self-diagnosis of QoS (with high success rate) and context acquisition (with significantly compressed yet representative data representation) are described. Use Case 2 (“Networks' Stability and Performance”) comprises methods for performance improvement of TCP Vegas (by classifying the cause of round trip time changes), orchestration of control loops (by automatically analyzing the configuration parameter space of the individual sub-loops) and vulnerability management (allowing autonomic agents to assess their security exposure in an automated way). Use Case 3 (“Dynamic Virtualization and Migration of Contents and Servers”) covers items such as context discovery improvements (by means of rule-based reasoning on raw data) and more efficient video delivery (reducing buffer starvation while minimizing capacity needs). In Use Case 4 (“SON and SON collaboration according to operator policies”), capacity enhancements of in-band relay links (via learning-based self-adaptation) and coverage optimization in dense base station deployments (via an intelligent reward distribution scheme) are covered. Use Case 6 (“Operator-governed, end-to-end, autonomic, joint network and service management”) addresses two load estimation mechanisms for RAN cells (one based on Self-Organizing Maps with very good accuracy, one based on statistical analysis) and an admission control scheme based on Explicit Congestion Notification (where the latter’s parameters are automatically adjusted without knowledge about the underlying traffic aggregate). Finally, Use Case 7 (“Network and Service Governance”) presents a new work on FTTH related fault analysis with very high accuracy.

Their effectiveness in operation was demonstrated mostly through simulation or mathematical analysis. In addition, we also consider the effectiveness of the techniques when used together in the same network, which reinforces the need for coordination and exchange of information between these techniques, and identify the location and nature of the integration that is required through the UMF in order for the techniques to draw a maximum benefit out of the collaboration.

One of the objectives of this deliverable was to identify what method is best-suited for a given observation and action related problem. This particular objective has been addressed in milestone MS31, which provides a systematic methodology to assess this question by means of a questionnaire, and the results will be conclusively presented in the handbook deliverable D3.9.

Table of Content

Foreword

The main objective of the project UniverSelf is to devise a Unified Management Framework (UMF) that allows for trustworthy interoperability of individual autonomous functionalities. We term the functionality *Network Empowerment Mechanism (NEM)* that is defined as: a functional grouping of objective(s), context and method(s) where "method" is a general procedure for solving a problem. A NEM is (a priori) implemented as a piece of software that can be deployed in a network to enhance or simplify its control and management (e.g. take over some operations). An intrinsic capability of a NEM is to be deployable and interoperable in a UMF context (in a UMF-compliant network).

In UniverSelf these individual functionalities are approached from three different perspectives in three different work packages (WPs). From the WP3 perspective, the algorithmic view (method) prevails, applying appropriate methods to solve the use case problems: e.g. use of Bayesian inference (*the method*) for fault diagnosis (*the objective*) in FTTH environments (*the context*). From the WP2 perspective, the integration view (framework) prevails as an intrinsic capability of a NEM is to be deployable in a UMF context (in a UMF-compliant network) considering the interdependencies of the functionalities, which entails for trustworthy interoperability and interworking. From the WP4 perspective, the system view prevails as a NEM is (a priori) implemented as a piece of software that can be tested, validated and deployed. In this context, the ultimate goal of the project is to "put everything together" in a comprehensive and elegant way, i.e. shaping the project solution portfolio.

The Network Empowerment work package (WP3) aims to provide the most efficient methods to deliver a toolbox of solutions covering selected operator scenarios. It covers all the work needed to study, design and to evaluate various algorithms with self-x and cognitive capabilities (hereafter termed methods) together with the requirements for their embodiment into network functions to assure trustworthy federation of heterogeneous networks. The work in this work package is based on use cases' problems and should provide the best-suited methods to solve these problems. WP2 as the integration part of UniverSelf will eventually embody the algorithms designed in WP3 into the network through the design of enabling mechanisms and facilities.

The main focus of this deliverable (D3.7) is the work on actions that are taken on certain observations done in task T3.3, while deliverable D3.5 focus on work on optimization of system parameters in the task T3.2, and deliverable D3.8 will focus on work within task T3.4 on the role of cooperation strategies between control loops and network entities. In addition to observation, diagnosis and learning techniques, this deliverable also has a focus on integration through the UMF capabilities, and therefore involves close collaboration with WP2. In particular, the outcome of milestone MS26, where information of the NEMs described in this deliverable was used to work on the UMF integration of the different solutions.

This deliverable covers techniques for discovery, aggregation, diagnosis, mining, and (self-) modelling of data, previously reported in deliverable D3.2, and on how these techniques can interact beneficially and purposefully through the UMF.

1 Introduction

The main aims of this deliverable are to:

- present the progress to work performed on the optimization methods in T3.3,
- and to put into view how the various techniques would fit together under the UMF.

The following chapters focus on the various methods/techniques for observation and action. Within the context of UniverSelf, these techniques would be implemented in the form of Network Empowerment Mechanisms (NEMs). NEMs, as defined in deliverable D2.2, are pieces of software deployed in the network that address a networking problem. In the context of this deliverable, the description of the functionality of the relevant NEMs will focus on the various algorithms that they implement. The NEMs implement a range of different techniques, and covers different areas of the network. In order to give structure to the set of NEMs and also to be able to illustrate and discuss the usefulness of UMF in supporting interactions between different NEMs, this deliverable is structured in terms of the Use Cases specified in WP4.

The Use Cases present the natural context where the various optimization techniques are performed. Therefore, each of the chapters 2-7 start with a brief introduction to the given Use Case. After that, the different NEMs belonging to that Use Case are described. The focus of the NEM descriptions in this deliverable is mainly on the progress that has been made compared to deliverable D3.2 and on the evaluation results that substantiate the conceptual explanations which were given previously. At the end of each Use Case related chapter, there is a detailed discussion on how the involved NEMs can benefit from and interact via the UMF.

Figure 1 gives an overview of the various NEMs that are part of T3.3 and thus this deliverable. The conceptual relationships between the NEMs can be of different nature. While some NEMs (such as “SOMs in Support of TCP Vegas” and “PCN based Admission Control”) work towards similar goals (in this example “congestion control”), other NEMs employ different methods to achieve the same or a similar goal (e.g. “Load Level Estimation” and “Load Prediction in a RAN”).

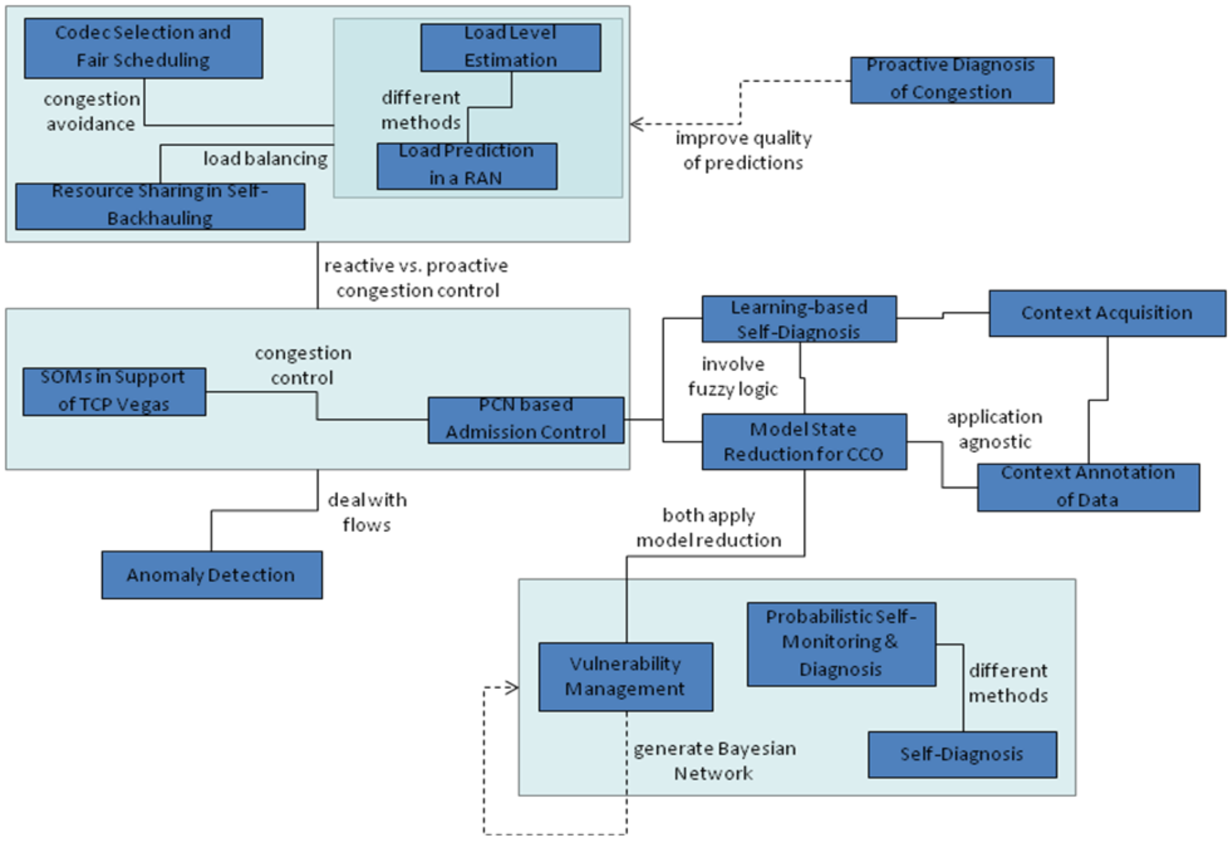

Codec Selection and Fair Scheduling
congestion avoidance
Load Level Estimation
different methods
Load Prediction in a RAN
load balancing
Resource Sharing in Self-Backhauling
Proactive Diagnosis of Congestion
improve quality of predictions
reactive vs. proactive congestion control
SOMs in Support of TCP Vegas
congestion control
PCN based Admission Control
Learning-based Self-Diagnosis
Context Acquisition
involve fuzzy logic
Model State Reduction for CCO
application agnostic
Context Annotation of Data
deal with flows
Anomaly Detection
both apply model reduction
Probabilistic Self-Monitoring & Diagnosis
Vulnerability Management
different methods
generate Bayesian Network
Self-Diagnosis

Figure 1: Overview of NEMs in T3.3

Other NEMs again use the same method class but apply it to different problems ("Learning-based Self-Diagnosis", "Model State Reduction for CCO" and "PCN-based Admission Control" all use fuzzy logic). Finally, NEMs might have a temporal relationship, where one NEM can use the output generated by another NEM, for instance (this is the case for the dashed arrows in Figure 1).

Work has been performed by partners to identify the precise benefit and role of UMF for each of the relationships depicted in Figure 1. It would, however, be beyond the scope of this deliverable – whose focus is on the NEMs themselves rather than their interaction or UMF – to present the outcome of all that work here.

2 Use Case 1 Related NEMs

2.1 Introduction

This UC considers self-diagnosis and self-healing features with two mains applications for a same network topology: Self-diagnosis and healing of IP networks and IMS services, and Self-diagnosis and healing of VPN networks.

The objectives are to enable proactive and improved reactive diagnosis and healing capturing the following properties:

- Proactive: to prevent incident impact (not done today).
- Improved reactive: to reduce the delay between incident or anomaly, detection and reparation and then improving the customer experience.
- Reusable and flexible: to prevent the redesign of the process from scratch for new services or change in the network configuration.
- End to end: to face multiple network domains and technologies which are currently managed by dedicated teams with dedicated & ossified tools and to enable an end user diagnosis and healing.

The proliferation of networks and services (heterogeneity of networks and increasing number of services, vulnerabilities), highlights the crucial role of assurance processes. Fault and Performance Management as defined before are critical functions towards ensuring the quality of the provisioned services or network capabilities, especially in an end to end way.

The current status is a reactive management: Following customer complaint or/and alarms, operator is then handling the problem with lot of ossified tools, isolated and dedicated teams. Operational teams are overwhelmed by the amount of data to analyze and to correlate. It implies lot of customer care, IS and human effort.

The goal is to improve reactive management by reducing the delay between incident/anomaly occurrence and detection reparation delay, and to enable a proactive management to prevent incident impact. It is also to enable micro-granularity management when necessary to focus on end-user issue diagnosis, facing high amount of data.

2.2 NEM 2: Self-Diagnosis based on Bayesian Networks and Case Base Reasoning

2.2.1 Context of the work

Fault diagnosis is a critical task for operators in the context of the e-TOM (enhanced Telecom Operations Map) assurance process, whose purpose is to reduce network maintenance costs and improve availability, reliability, and performance of network services. Although necessary, this operation is complex and requires significant involvement of human expertise.

This operation is considered as one of the main functions regarding fault management. For that reason, intensive research has been conducted over the past years to improve the current diagnosis process, investigating the application of numerous techniques from the field of artificial intelligence and graph theory. However, there is still a need for developing new solutions able to fulfil network operators' requirements and face real deployment constraints.

2.2.2 Content of the work

We propose a new hybrid approach combining Case-Based Reasoning (CBR) and Bayesian Networks (BN). According to literature, Bayesian Networks are currently the most powerful and popular diagnosis method. However, the complexity of inference in BN increases exponentially with the number of nodes. Hence, this technique is not suitable for large scale systems including a large number of components such as current and

future networks with hundreds or thousands of elements. To overcome this limitation, we propose a combined case-based and Bayesian reasoning approach to improve the BN inference while keeping the advantages of BN technique, the resulting solution improves the degree of automation of the diagnosis process and requires less intervention of human expertise.
In order to optimize the diagnosis process, we consider only a subset of the BN structure, and, inside this subset, only the nodes where variations of the monitored parameters have been observed. This subset is identified thanks to Case-Based Reasoning. CBR is also used for the learning phase of our approach. CBR learning allows improving the process efficiency over time by accelerating the identification and resolution of previously encountered pathological cases.
The part of each technique is summarized in the following:

1. Fault model management challenges

Use Bayesian Network technique for building, and updating the diagnosis model
Use the chi-square test for causality graph
Leverages multiple observable parameters

2. Operational challenges

Use Case Based Reasoning technique to optimize the diagnosis process

3. Learning

Use Case Based Reasoning for improving the process efficiency over time by accelerating the Identification and resolution of previously encountered pathological cases.

The details of the approach are available on [1] and [2].

To create and simulate a network topology, a set of nodes is generated using the BNJ (Bayesian Network Tools in Java), an open-source suite software for BN. Each node is characterized by a marginal probability table, calculated from statistics collected from the observed network.
We evaluate and compare the performances of the combined CBR-BN approach to those of the classic BN technique. The evaluation results are organized according to a set of metrics, namely: Accuracy, Reliability and Speed.

Accuracy: This figure shows that, whatever the size of the original network, our NEM can precisely identify the root cause with greater accuracy than the reference approach of fault diagnosis.

Number of nodes in BN	BN approach	CBR-BN approach
40 nodes	3 to 4	1
60 nodes	3 to 4	1
80 nodes	3 to 4	1
100 nodes	4 to 5	1
200 nodes	4 to 6	1
500 nodes	6 to 8	1
1000 nodes	15 to 21	1
2000 nodes	25 to 40	1

Table 1: Test of accuracy

Reliability: According to the dataset, the confidence interval, in 95% of cases, is approximately (96.5, 98.7).

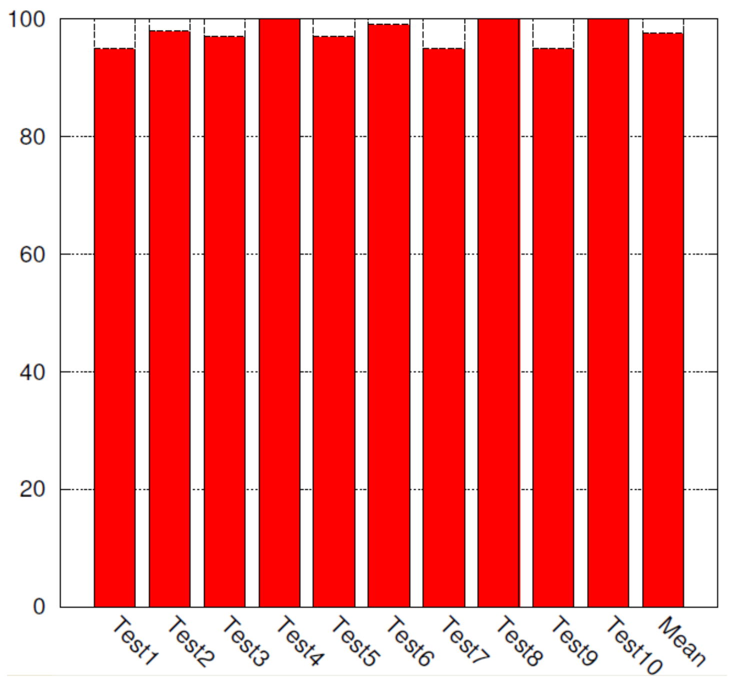


Figure 2: Test of reliability

Speed: The increase in the original network size has less impact on the detection time for the CBR-BN approach than for the BN approach (diverging curves).

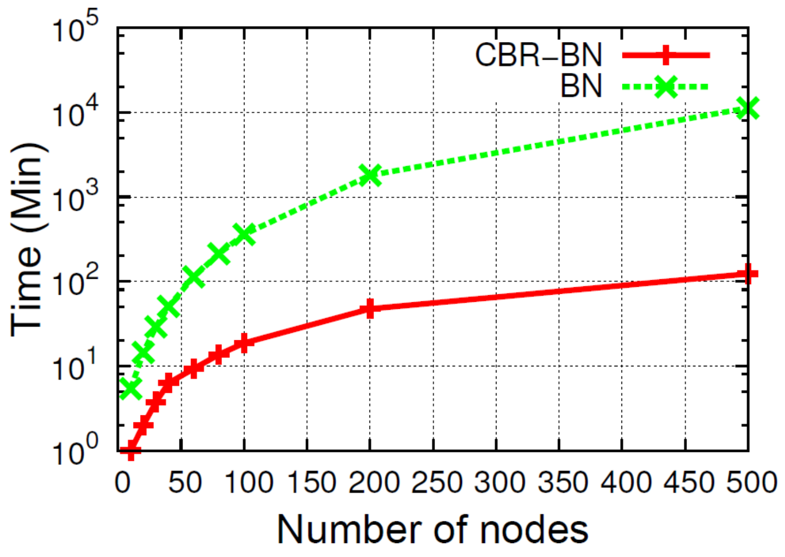


Figure 3: Test of detection time

2.2.3 Merit of the work

The added-value of our NEM is essentially the reduction of the complexity which appears in a significant reduction in the number of nodes involved in the diagnosis process.
In order to illustrate this phenomenon, the following table details the number of nodes involved in the diagnosis process for the BN and CBR-BN cases. For networks greater than 40 nodes, the CBR-BN solution enables on average a reduction of the resulting network to a mere 1/10th of the original network size.

Number of node in BN	Size of the case for 9 tests									Median
10	2	2	3	2	3	3	3	3	2	3
20	2	3	4	4	4	3	5	3	4	4
30	4	4	3	2	3	4	4	4	5	4
40	4	5	6	3	4	5	3	5	5	5
50	5	5	4	6	5	5	5	4	6	5
80	7	8	5	7	5	6	5	7	7	7
100	10	9	10	11	10	11	11	10	10	10
200	23	21	21	18	19	16	21	20	22	21
500	46	54	49	53	59	42	54	49	62	53
2000	126	269	420	74	139	305	126	275	431	269

Table 2: Reduction of network size using CBR-BN

2.3 NEM 13.2: Anomaly Detection

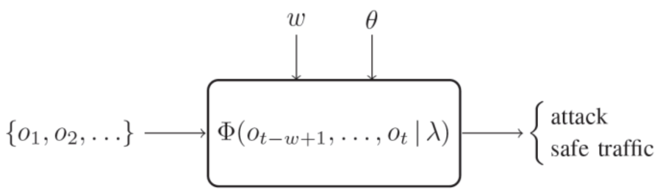
2.3.1 Context of the work

Anomaly-based intrusion detection systems (IDSs) aim to classify an activity as benign (normal) or malicious (anomalous) by comparing it with a model of normality. Since the Eighties, many flavours of anomaly-based systems have been developed, depending both on the type of information processed (network traffic or system logs, for example) as well as the type of modelling technique used (among others: statistical or machine learning techniques). In all cases, however, an anomaly detection system can be more or less sensitive to a change in normality. The sensitivity of a system to anomalies controls which instances of traffic are flagged as anomalous and, in general, it determines the performance of the system. The sensitivity level of an anomaly detection system usually depends on a set of system parameters. Such parameters play a central role, since they determine the overall behaviour of the detection system and how it reacts to an intrusion. There exists a natural trade-off between detecting all anomalies (at the expense of raising alarms too often), and missing anomalies (but not issuing any false alarms).

Despite the deep impact that the system parameters have on the performance of the detection system, the literature has put little emphasis on the topic of automatic parameter tuning for intrusion detection systems. The task of tuning the system is carried out by the system manager or by expert IT personnel and it requires detailed knowledge of both the detection system as well as the network to be protected. While tuning a system, the system manager implicitly aims to achieve the best compromise, i.e., keeping the false positives as low as possible while trying to detect as many real attacks as possible. We propose to replace this manual tuning by an autonomic approach that optimizes the parameters of the detection system toward a high-policy goal.

2.3.2 Content of the work

The proposed approach allows the system performance to be optimized with respect to the aforementioned trade-off between detected anomalies and false alerts. The goal is to allow the system manager to control the system performance by specifying high-level policies. The following figure presents a high-level overview of the proposed detection system. We assume the system to be based on a probabilistic model of normal behaviour, indicated by λ. The system observes a stream of network measurements $o_1, o_2, \ldots$ and analyzes a window of length w of past measurements, to which we refer as observation sequence.



The system evaluates the likelihood Φ that the observed sequence matches the model of normal behaviour. A threshold *θ* is then applied to this likelihood to detect unlikely events. Obviously, the selection of the parameters *θ* and *w* does play a crucial role in the effectiveness of the detection system.

We allow the system operator to choose the goal of the optimization in form of a high level policy by defining the relative importance of correctly detected anomalies over false alerts. Mathematically, this can be described as the optimization problem

$$\max_{w,\theta} \alpha \cdot TN + \beta \cdot TP$$

where TN and TP give the true negative rate and true positive rate, respectively, of the detection system and *α* and *β* their respective weights ("importance"). The goal is to find those values *θ* and *w* that maximize the above expression.

The optimization problem can be solved by considering a probabilistic interpretation of TN and TP. Let X^w and Y^w be random variables equal to the likelihood Φ for an arbitrary anomaly-free, respectively anomalous, observation sequence. Then, TN is equal to the probability $P(X^w \leq \theta)$ and $TP=1- P(Y^w \leq \theta)$. The distributions of X^w and Y^w can be determined empirically and depend on *w* and *λ*.

We have applied the above approach to an intrusion detection system for the detection of Secure Shell (SSH) brute-force attacks in high-speed networks. The system monitors flow-based measurements of network traffic and uses a Hidden-Markov Model (HMM) to compute the likelihood of an observed sequence of measurements. The following steps have been performed:

1. Train the model to normal traffic.
2. Determine the distributions of X^w and Y^w for a small set of window sizes.
3. Fit Gaussian mixtures to the distributions. The fitted Gaussian parameters are extrapolated to other window sizes by polynomial fits.
4. Solve the optimization problem numerically by replacing TN and TP by the Gaussian mixtures fitted to the distributions.

We tested the procedure using six data sets of SSH time series of flows per seconds, each containing both legitimate and malicious traffic. Two data sets are synthetically generated. The other four data sets consist instead of time series created directly from real traffic collected at the University of Twente network. We used two sets for the training of the HMM, while the other data sets have been used for testing.

Table 3 summarizes the results of the optimization procedure for one of the data sets.

			Confusion matrix		Detection and normalization lags			
β/α	w_{opt}	θ_{opt}	TP	TN	$\overline{D}_{\mathbf{A}}$	$std(D_{\mathbf{A}})$	$\overline{E}_{\mathbf{A}}$	$std(E_{\mathbf{A}})$
0.1	1000	3923.12	0.22	0.99	640.80	523.94	0	-
0.2	1000	3854.63	0.23	0.98	623.20	531.67	0	-
0.3	1000	3803.11	0.23	0.97	607.00	530.02	0	-
0.4	1000	3752.98	0.24	0.95	597.60	526.69	0	-
0.5	1000	3691.12	0.24	0.93	583.80	526.55	0	-
0.6	1000	3565.83	0.29	0.88	565.80	516.73	0	-
0.7	1000	3351.39	0.45	0.76	467.50	464.88	178.00	398.02
0.8	1000	3247.08	0.63	0.71	442.17	454.67	208.20	408.37
0.9	1000	3179.27	0.70	0.67	416.50	436.71	29.75	59.50
1	1000	3128.34	0.74	0.62	395.67	425.35	33.25	66.50
1.5	1000	2975.06	0.90	0.46	310.17	383.29	146.50	179.09
2	1000	2888.05	0.93	0.36	250.83	349.33	140.67	243.64
4	310	795.16	0.98	0.12	28.00	68.10	∞	∞

Table 3: Results of Optimization Procedure

As expected, we observe that the choice of the ratio β/α controls the achieved TP and TN. A large value of β/α implies a preference for TP and, hence, the optimization procedure chooses a low threshold Θ. In contrast, small values of β/α enforce a large threshold, resulting in a high TN. In addition, we have measured other characteristics of the detection system. For example, the detection lag D_A gives the time (in seconds) the system needs to detect an attack after it has started. On the other hand, E_A gives the time the system needs to return to the normal situation after the end of the attack. As can be seen, a large ratio β/α and, consequently, a high TP result in a short detection lag because the detection system behaves more aggressively.
A detailed and extensive discussion of the optimization procedure and its performance has been published in [3].

2.3.3 Merit of the work

In deliverable D3.2, we have described how to approach the parameter-tuning problem of anomaly-based intrusion detection systems by formalizing it in terms of an optimization procedure. In the present deliverable D3.7, we provide a detailed description of the necessary steps in order to apply the procedure to a concrete intrusion detection system. The optimization procedure allows to explicitly relating the system parameters and the performance measures in the form of an optimization problem. A key characteristic of the proposed solution is that it regards optimality according to *high-level* policies provided by the management environment, in this way replacing the manual tuning of the system parameters by the system manager or by expert IT personnel.

We have validated the procedure by applying it to an IDS for the flow-based detection of SSH brute force attacks in high-speed network traffic. Our results show that, by varying the relative importance, we are able to fine-tune the system to favour either the detection of all the anomalies or the detection of attacks only when they are certain. Our findings also show that the relative importance has impact on the detection rate and on how timely the system is able to detect an attack or recover from it.

2.4 NEM 14: Proactive Diagnosis of Congestion

2.4.1 Context of the work

This NEM is proposed in the context of improving network management of the system. In particular, addressing the problem of congestion prediction it targets at offering the system the capability to proactively handle such situations. Towards this direction, the NEM combines network data that can be monitored and reports how close a core link is to a potential congestion. This information can be delivered through the Unified Management Framework (UMF), following the processes described in the knowledge (KNOW) UMF core block, to the respective decision making mechanisms of self-healing so as to treat potential congestions of the links proactively and avoid them.

2.4.2 Content of the work

The function of the NEM and the provided predictions are based on the past experience of the network. In particular the NEM builds knowledge with respect to past monitored network data and what they meant in terms of congestion for the monitored link. Towards this direction, the unsupervised machine learning scheme of Self-Organizing Map (SOM) has been employed. SOM is a special type of artificial neural network, consisting of one input and one output layer, and trained using Learning Vector Quantization (LVQ). The output layer is a 2D rectangular lattice of units representing points in the input space (usually multidimensional, >2D) and arranged by their resemblance. The input space consists in this application of SOM consists of monitored data related to either network traffic, in terms of incoming bytes and/or packets and their respective trends (i.e. first- or second- order differences), or to network and link capabilities such as buffer size of the node that outputs traffic to the link, average queue load and link capacity have been exploited.

Each set of values of the monitored parameters was mapped on a 2D map followed by a metric that designated how close to congestion the link was. The metric that defined how close to congestion the link was, was the percentage of packet drops over the total packets transmitted. Its potential values are three, namely: a) not congested (0% drops), towards congestion (<10% drops), congestion (>10% drops).

After building the necessary knowledge and designing the mechanisms, the evaluation of the methodology took place. This involved the comparison of the predicted values of congestion with the real observed ones.

The best performance of the mechanisms was observed when the map of Figure 4 had been created from the parameters of a) Incoming Bytes, b) the trend of the incoming Bytes, c) the link capacity, d) the queue size and e) the buffer size. Using this particular set of parameters, the NEM performed with a percent of correct predictions equal to 86.6%.

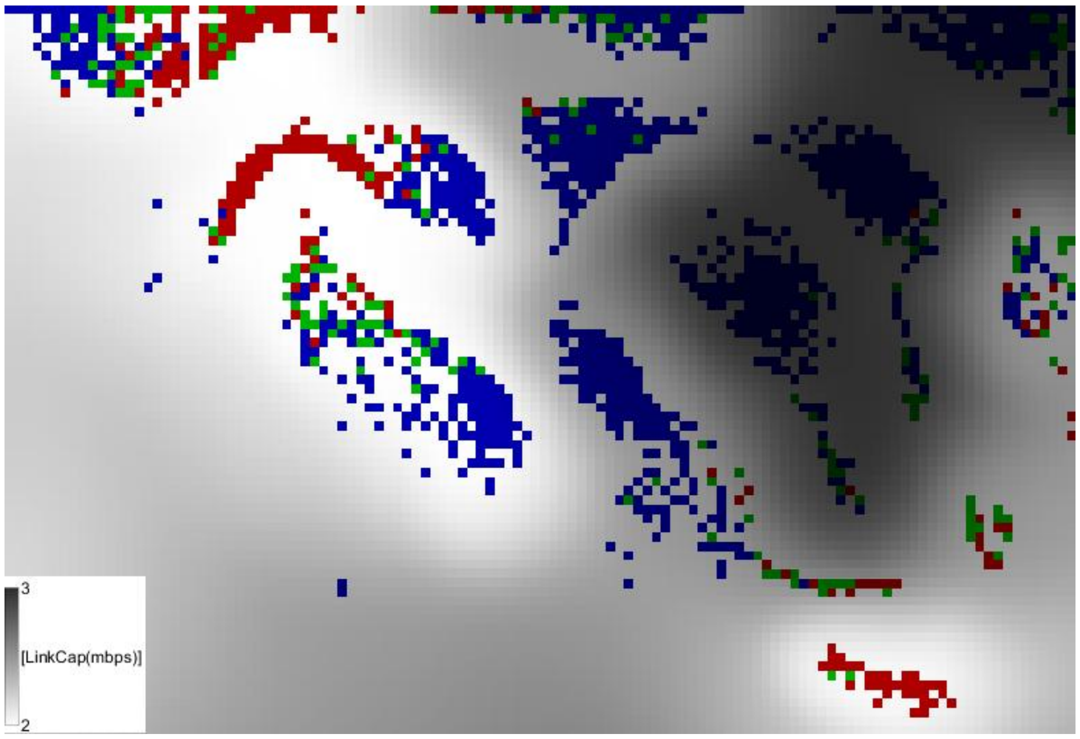


Figure 4: SOM depicting the relation between the capacity (light through dark) and the congestion levels (0 in blue labels when the link can serve all the traffic, 1 in green labels when some packets drop but yet is not treated as a congested link and 2 in red labels when the link is expected to become congested) of the under question link.

Useful information can be obtained from the clusters created on the resulting grid by examining the monitored data values of any individual unit or by drawing component matrices for any of the monitored parameters (as for instance in Figure 4). More details on the exact methodology used and the respective results can be found in [1][4][5].

2.4.3 Merit of the work

The main merit of the mechanism is that it targets at discovering potential congestion proactively, i.e. before they reach the user or at least at their very beginning. This enables the network operator or the system per se to also handle it proactively minimizing the consequences that reach the user, e.g. deterioration of the network performance/ application and thus deterioration of the QoS and/ or the QoE. Through UMF, alarms can be created and delivered to self-healing mechanisms soon enough in order to avoid congestions and their consequences.

The delta from Deliverable D3.2 description of this mechanism is the addition for individual component visualization functionality (as depicted in Figure 4). Furthermore, this section contains results from the application of the proposed mechanism while Deliverable D3.2 does not.

2.5 NEM 20: Self diagnosis based on network and service data

2.5.1 Context of the work

Taking into account the description provided in [1], we have proposed the introduction of a self-diagnosis mechanism which will allow the management system, using inputs from network elements (network and service data) to proceed in identification of QoS degradation events in IP networks, entitled “Learning-based self diagnosis based on network and service data". More specifically, we propose the introduction of a scheme that will exploit service information and by combining it with network measurements will identify whether this service flow experiences a problematic situation or not. The self diagnosis mechanism will monitor the served flows and by exploiting service information derived from IP Multimedia Subsystem (IMS) servers will identify the flows where a specific remedial action should take place in order to tackle the low QoS event.

For the case study under consideration and for the VoIP service, delay, jitter and packet loss are identified as playing a significant role in QoS degradation, thus the event identifier evaluates the aforementioned inputs for every active session; in other cases (e.g., heavy load) other monitoring inputs could be used. The membership functions of the identified parameters describe the input and output parameters. More specifically, the membership functions indicate the values of each parameter, the range of each value and the magnitude of their participation. The "Delay", the "Jitter" and the "Packet Loss" per flow comprise the input vector, whereas the output is the "QoS degradation". Such NEM provides a generic tool for self-diagnosis in core network elements; the situation perception and the reasoning scheme is the same for all network elements, and is induced to them, upon their initial configuration. This is a major drawback due to the fact that, the environment conditions might change so the environment perception should change as well. Thus, the incorporation of a scheme for the evolution of the environment perception is part the self diagnosis NEM. The evolution/adaptation is based on the use of clustering schemes and statistical analysis of the gathered/measured data.

2.5.2 Content of the work

In [1] we have provided a description of the initial scheme, with the incorporation of a well known clustering method, the k-Means. An enhancement to the initial algorithm is related to the statistical analysis of the measurements, instead of clustering, in order to proceed in more sophisticated adaptations of the self diagnosis mechanism.

The statistical analysis of a dataset provides information regarding the special characteristics of the dataset; the extraction of the probability distributions provides crucial information as regards the available data. Many probability distributions are well described in the literature (e.g., Gaussian, Pareto, Log-normal, etc); the Gaussian is a very well-known one and is used widely in the literature for the description of various datasets. The reasons for such acceptance are related to its nice behaviour and formal mathematical description. Furthermore, in case we observe large datasets, based on the central limit theorem [7] the sampling distribution of the mean becomes approximately normal regardless of the distribution of the original variable. As regards the sampling distribution of the mean, this is centred at the mean value μ of the original variable and the standard deviation of the sampling distribution approximates $\sigma / \sqrt{N}$.

The general formula for the probability density function of the Gaussian distribution is:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x-\mu}{2\sigma^2}^2}$$

where μ is the mean value of the measured objects while σ is the standard deviation.

2.5.3 Merit of the work

Initially, we configure the network elements with a generic membership function configuration (Table 4) and we compare with the developed ground truth. For the extraction of the ground truth, we use fuzzy reasoners that are built specifically for the environment of the data that we generated; thus we consider that when the data set is evaluated by them, the decisions will be correct. Using a generic configuration the self-diagnosis scheme achieves a success rate of 64%. Given the fact that such self-diagnosis scheme is built to operate adequately in all environments we consider this success rate as acceptable. Then, we apply the clustering adaptive algorithm in this self-diagnosis scheme and have a success rate of 70.01% (amelioration of 8.6%) (Membership functions in Table 5).

	Low	Medium	High
Delay (in ms)	0 – 80	40 – 150	>120
Jitter (in ms)	0 – 40	20 – 80	>60

Packet Loss (%)	0 – 0.0035	0.0025 – 0.008	>0.006

Table 4: Initial configuration of the input membership functions of the self-diagnosis fuzzy reasoners

	Low	Medium	High
Delay (in ms)	0 – 20	5 – 80	30 – 200
Jitter (in ms)	0 – 40	35 – 100	55 – 200
Packet Loss (%)	0 – 0.005	0.004 – 0.0057	0.0055 –0.01

Table 5: Input membership functions of the self-diagnosis fuzzy reasoners after the clustering adaptation procedure

By incorporating the statistical adaptive mechanism and once we have followed the methodology presented in [1] we modify the input membership functions (Figure 5). As it is obvious, the input states are being captured by new membership functions, which are being described by Gaussian distributions, with higher overlap areas. The success rate of the adapted scheme reaches 84% compared to the ground truth (amelioration 23.8%).

		Low	Medium	High
Delay (in ms)	μ	9.8	30.32	64.69
	σ	10.48	22.2	0.2281
Jitter (in ms)	μ	18	48	106
	σ	12	34	26.09
Packet Loss (%)	μ	0.0022	0.0047	0.0073
	σ	0.0017	0.0028	0.0018

Table 6: Mean values (μ) and standard variations (σ) of the input membership functions

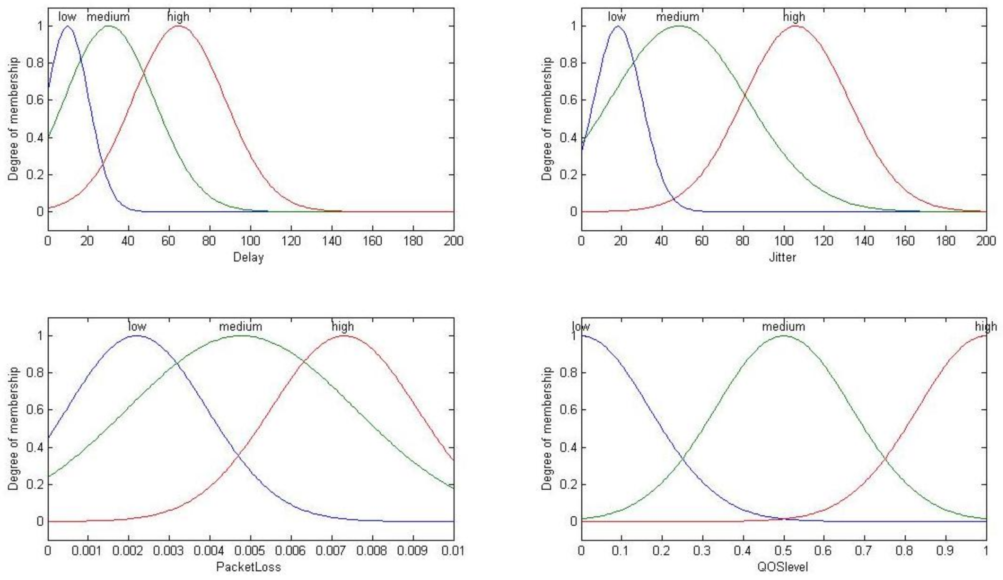


Figure 5 : Input (Delay, Jitter, Packet Loss) and output (QoS level) membership functions after the adaptation procedure.

2.6 NEM 21: Optimization of context acquisition and dissemination

2.6.1 Context of the work

In this section, the performance analysis of the Context Acquisition and Dissemination mechanism will be presented in detail, in the context of the respective NEM (Context Acquisition and Dissemination). This mechanism is proved and developed under the scope of UC1 and UC3. Specifically, the proposed context acquisition and dissemination framework is based on Data Mining methods for processing, classifying and labelling a set of network parameters (e.g. Delay, Jitter, Packet Loss) as monitored by a network element. This procedure produces a QoS/Load indicator that captures the network element's state and can be also used as an initial step to perform several Machine Learning algorithms.

For addressing the needs of UC1, Context acquisition and Dissemination mechanism is considered as a key part of the UC1 concept, namely Self-diagnosis and healing of IP networks and IMS services. Towards this direction, the goal to achieve is the proactive diagnosis of QoS degradation events at the core network elements by exploiting network (monitoring each network element) and service (provided by IMS servers) measurements. In order to meet this target, context NEM needs interaction with NEM 20 (learning-based Self-Diagnosis) by exploiting UMF functionality for inter-NEM communication through the COORD block. Specifically, NEM 21 will provide a first decision (Low, Medium, and High QoS) of the QoS events, these events will be further processed by NEM 21 for extracting context which will finally calibrate the decision making process of NEM 21. This procedure is also depicted in Figure 7.

2.6.2 Content of the work

Based on the formal proof of concept of this work, already presented in the Deliverable 3.2 [1], the distributed Data Mining framework takes as input a large set of network measurements and produces a classification scheme according to the geometric characteristics of the input data. Furthermore, a QoS label is added to each measurement in order to address the network element's state. The produced results can be exploited by the network in order to reduce the amount of the exchanged information among the devices, thus preserving resources.

For the experimental validation of this mechanism, a set of measurements is aggregated in order to feed the algorithm with raw network data. Such data are related to specific VoIP metrics, namely Delay, Jitter and Packet Loss, randomly generated by MATLAB, taking into strong consideration the real ranges of these key performance indicators. For this reason, based on [8], a realistic approach of these ranges is presented below:

VoIP Input Measurements Range		
Metric	Range	Excellent QoS Threshold
Delay (msec)	0 – 2000 (ms)	≤ 100 ms
Jitter (msec)	0 – 500 (ms)	< 75 ms
Packet Loss (%)	0% – 100% (lost/total packets)	≤ 1%

Table 7: VoIP KPIs Ranges.

Several datasets were generated and tested using the framework, each one based on a different method of data representation. As described in [1], for the data pre-processing procedure, two main techniques were developed, namely the normalization and the kernel transformations. To this direction, three different datasets are produced. The first one contains the real, raw values of the measurements whereas the second presents the normalized values of each metric so as to equalize the order of magnitude among the three input parameters and facilitate the arithmetic operations. The latter is a Euclidean kernel transformation of the first file. Because of the kernel properties, thoroughly presented at [1], the dimensionality of the latter dataset gets double due to the dot product properties. Specifically, the form of each tuple *i* for each dataset can be depicted as follows:

Input Data Representation	
Initial Dataset	$(Delay_i, Jitter_i, Packet\_Loss_i)$

Normalized Dataset	$(\frac{Delay_i}{Max\_Delay}, \frac{Jitter_i}{Max\_Jitter}, \frac{Packet\_Loss_i}{Max\_Packet\_Loss})$
Kernel Dataset	$(Delay_i^2, Jitter_i^2, Packet\_Loss_i^2, \sqrt{2 * Delay_i * Jitter_i}, \sqrt{2 * Delay_i * Packet\_Loss_i}, \sqrt{2 * Jitter_i * Packet\_Loss_i})$

Table 8: Input Data Representation.

Proceeding to a further analysis of the datasets, it should be mentioned that the total length of the first two datasets is 5.06MB, consisting of 50.093 tuples, with 101.01 bytes per tuple while the latter dataset contains the same number of tuples (50.093) with 202.02 bytes per tuple (double size) because the kernel transformation has increased the dimensionality of the attributes.

2.6.2.1 Normalized Dataset

Starting with the simulation analysis of the normalized dataset, the normalization process takes place according to the formal proof of the algorithmic framework. The input measurements are bounded to the interval [0, 1] in order to reduce the magnitude and facilitate the arithmetic operations. The respective results are presented at Table 9 and Table 10.

Dataset	QoS Label	Cluster Centroids			Cluster Radius
		Delay	Jitter	Packet Loss	Euclidean Value
Normalized Dataset	Low	0.182001	0.306748	0.692039	0.456726
	Low-Medium	0.260908	0.410934	0.725792	0.986675
	Medium-High	0.100436	0.136387	0.254454	0.509971
	High	0.149716	0.251389	0.297616	0.607888

Table 9: Normalized Dataset's Cluster Characteristics.

Dataset	QoS Label	Cluster Bounds				Tuples/Cluster
		Direction	Delay	Jitter	Packet Loss	
Normalized Dataset	Low	Left	0.1930	0.2584	0.4850	15963
		Right	0.1821	0.3069	0.6921	
	Low-Medium	Left	0.2607	0.4107	0.7257	9838
		Right	0.8358	0.9121	0.8260	
	Medium-High	Left	0.0047	0.0037	0.0030	10635
		Right	0.1005	0.1365	0.2545	
	High	Left	0.1497	0.2513	0.2976	13657
		Right	0.4946	0.6721	0.0263	

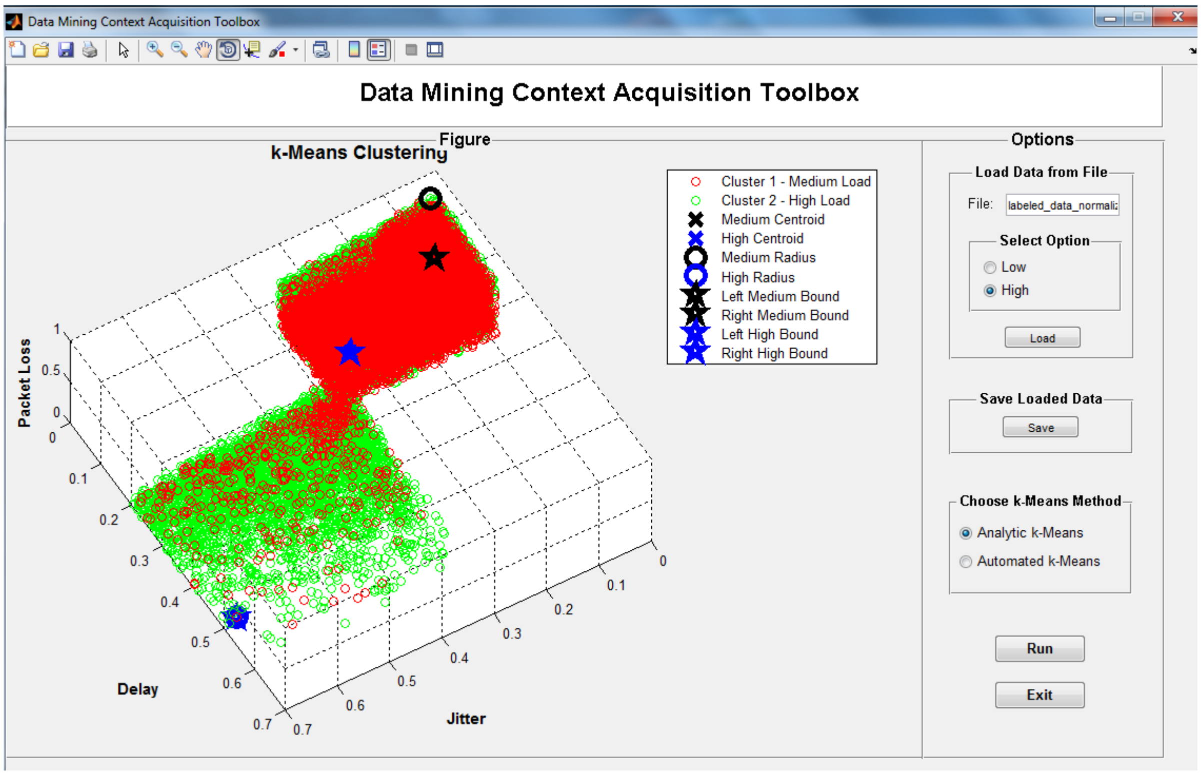
Table 10: Normalized Dataset's Cluster Bounds.

The extracted information is significantly reduced compared to the initial data volume since the involved network elements will only exchange 12 tuples instead of 50093. Specifically, 4 tuples contain the cluster centroids and radius as depicted in Table 9 while eight tuples are used for the cluster bounds representation (2 bounds per cluster) as depicted in Table 10. Considering that tuple size is 101.01 bytes, only 1212.12 bytes will be exchanged instead of 5.06MB, therefore the derived context reaches (0.0024%) of the initial volume.

Following this approach, the extracted context which is related to the geometric characteristics of each cluster, is very accurate since the successfully classified instances ratio is 100%. Instead of communicating the whole dataset, Normalization techniques' contribution produces a very descriptive set of data that each NE can use/disseminate to describe its QoS state. Specifically, as depicted in Figure 6, the generation of the clusters (i.e. spheres) using the produced results for centroids, radius, bounds, etc. Is adequate for the efficient representation of both the Low QoS space (black points) and the High QoS one (blue points).

2.6.2.2 Initial Dataset

Regarding the first dataset, the data acquired by the network are provided without pre-processing to the k-means algorithm. The produced clusters as well as the centroids, radius and bounds are acquired in order to achieve context extraction, however, the correctly classified instances of this dataset reach 56% (the respective ratio of the normalized dataset is 100%) due to the disparity among the input data measurements' scale. Finally, further experimental results are available for providing the complete proof of concept of this work (i.e. initial dataset and kernel-transformed dataset); they will be presented in detail at the next deliverable.



File: labeled_data_normali

Select Option: ( ) Low [✓ High]

Choose k-Means Method: [✓ Analytic k-Means] ( ) Automated k-Means

Figure 6: Normalized Dataset for the QoS related Context Acquisition

2.6.3 Merit of the work

The results highlight the need of introducing novel data mining concepts in order to infuse cognitive capabilities to network elements. The dissemination of the aggregated context by each device in the network forms a resource-demanding process in terms of CPU, memory and high bandwidth for the communication among the network elements. For this reason, by applying the proposed Distributed Data Mining framework, the bulk network data can be replaced by a minimal thus meaningful context. The latter is able to adequately describe the whole dataset and preserve the resources required for exchanging all the aggregated measurements. Finally, this innovative method can also leverage several Machine Learning capabilities for solving complex problems. A respective presentation of the mechanism merits is also available at Milestone 31 [13].

2.7 Discussion

The UMF is organized into different segments in charge of ensuring the smooth deployment and interaction of NEMs. We illustrate below how it can be beneficial to the detection/diagnosis/healing methods.

2.7.1 Knowledge

The knowledge plane appears as a natural element both to exchange information between NEMs, and to define their inputs and collect their outputs. It also appears as the adequate interface between the managed networks/services and the methods deployed to supervise them.

For the diagnosis activity, one needs first to define the *scope* of a given NEM, i.e. to declare what element types are managed, but also which instances of these elements in the actual network. This scope definition concerns the identification of network elements, network segments, which services, which functions, which flows or which sessions, etc. Its unambiguous definition is necessary for smooth inter-NEM coordination. Some reflectivity of the network is therefore necessary to establish this common knowledge. It can further be exploited by model-based methods that need to be able to navigate among and inside the managed entities (see the self-modelling approach for example, or the need to rely on a flow-based representation of the network for QoS monitoring).

Further, NEMs need to access inputs from the network (or from the governance). Therefore they must have access to state information about their managed resources. This is particularly crucial for the diagnosis NEMs since they are triggered by alarms localized on specific managed objects, or by queries about the state of some functionality. Similarly they need to access to configuration information about the managed objects, to be able to collect observations (i.e. collect the value of some state variables, on equipment or services), or perform tests.

The outputs of fault management NEMs populate the knowledge plane in the same manner, by raising elaborate alarms on specific managed objects, or by assembling (correlating) sets of events behind the same root-cause. This is true for both early detection (proactive diagnosis) or for root-cause analysis (reactive diagnosis). Further, impact analysis can take as well the form of warning events attached to some objects and related to a known malfunction, and the repair suggestions can certainly assume a similar shape.

Finally, for NEM cooperation, the knowledge plane appears again as a natural candidate for information exchange. This is true in particular when several network segments must be jointly managed by the cooperation of several NEM instances, one per segment. The same holds for the cooperation of a network management NEM with another NEM in charge of the end-to-end services deployed over these physical resources, which is one of the objectives of UC1.

All this suggests that one should definitely consider structuring a knowledge plane in order to facilitate the deployment and interoperation of NEMs, in particular through a uniform representation of the managed objects and of their relations.

2.7.2 Governance

Governance concerns all aspects related to the definition, parameterization and operation of diagnosis NEMs.

The definition captures the scope of the NEM (which objects it is managing, which vulnerabilities it is managing), its deployment pattern (how many instances, which cooperation mode between these instances), the definition of its inputs/outputs and possibly the interaction modes with other NEMs located upstream (e.g. early detection) or downstream (e.g. impact analysis), the definition of its detection policy, and its declaration for visibility by the other NEMs.

Parameterization captures the setting of all internal values or resources that are necessary to the operation of the NEM, for example the balance between levels of early detection and false alarms, the access to generic models of the managed elements for diagnosis algorithms based on self-modelling, the access to the OVAL repository for the detection of vulnerable configurations, etc. It also captures the definition of the reporting frequency, the contents of these reports, etc.

The term operations gathers the different functioning modes of a NEM, which can be in installation, deployed but idle, in test/trial/benchmarking, in stand-alone, or stopped. But the diagnosis NEMs have spurious modes characterizing for example the learning phase, the building knowledge phase, whether the NEM works in query mode (triggered by the operator) or in supervision mode (triggered by network alarms or other NEMs), etc.

2.7.3 Coordination

The NEMs presented in this chapter do not display direct cooperation patterns, in the sense that they do not organize in a nice stream as early detection, root-cause analysis, impact analysis, healing, etc. This is mainly due to the different network segments or layers addressed by each of them. Nevertheless, such a form of coordination is possible and can (should?) rely on information exchanges through the knowledge plane, as discussed above.

The coordination can also be necessary within a single NEM. This is the case for example when several instances of this NEM have different scopes, i.e. manage different areas of a network, but nevertheless need to coordinate their work. Diagnosis is a typical instance of such a situation: network resources are usually managed independently of the services running on top of them. The network layers themselves are managed in a distributed manner, some teams being in charge for example of the access segments, another of the core network, others of the metropolitan networks. It is desirable that a NEM in charge of diagnosing end-to-end failures in the global IMS network is decomposed into NEM components reproducing such a classical division. But the coordination of these NEM components is necessary: for example, a user complaint about service quality can be due to errors in the provisioning of resources (e.g. incompatible codecs) or to network saturation. Such coordination can be hidden to the UMF and hard-coded into the NEM architecture, or conversely exposed to the UMF which will then be in charge of ensuring the correct coordination of the NEM components through its knowledge plane.

In any case, it is desirable to carefully design the knowledge plane in order to facilitate both the definition of NEM scopes and objectives (at least to check NEM compatibilities or conflicts), and to facilitate the exchange of information between NEMs (interface design).

2.7.4 Specific example

In this section, the usefulness of UMF is demonstrated using the specific interaction of the NEM “Adaptive Self Diagnosis” (cf. Section 2.5) with the NEM “Context Acquisition and Dissemination” (cf. Section 2.6).

Without the UMF the Adaptive Self Diagnosis and the Context Acquisition and Dissemination could not synchronize their operations. The Adaptive Self Diagnosis would have to gather information separately, in order to proceed in the adaptation process, which would require much time, given the fact that such an adaptation requires a bulk of data (training data) in order to begin its operation. Alternatively, the Self Diagnosis mechanism should have the capability to receive the disseminated information from the neighbouring network elements that participate in the scheme. Thus, the Context Acquisition and Dissemination acts as an enabler for the operation of the Adaptive Self Diagnosis.

Let us assume that:

- Both procedures are on the active phase.
- Both procedures operate independently from each other, i.e. different period of execution.
- Both procedures monitor the same parameters.

With the UMF we can harmonize the operation of the two separate procedures:

- The Self Diagnosis requests for inputs from the KNOW as regards context information (the parameters and metrics are defined at the installation phase).
- In case that KNOW does not contain the required information, it communicates with Context Acquisition NEM via COORD.
- Context Acquisition operates and extracts context information and once triggered by the GOV it updates the KNOW.
- COORD notifies Self-Diagnosis NEM that KNOW contains the requested context. A new configuration is provided to the Self Diagnosis.
- Self-Diagnosis is executed again, given the new configuration, in order to adapt the previously taken decisions according to the extracted context of the previous phase. This is a feedback loop that periodically (or on demand) calibrates the way that a NEM perceives its environment.

The self-Diagnosis NEM starts its operation by requiring context from KNOW. In case that KNOW doesn’t contain the context, it triggers COORD for the communication with the Context Acquisition NEM. COORD triggers this NEM by interacting with GOV and once the requested context is produced, KNOW is updated and a notification is sent to COORD by GOV. Finally COORD provides the results from KNOW.

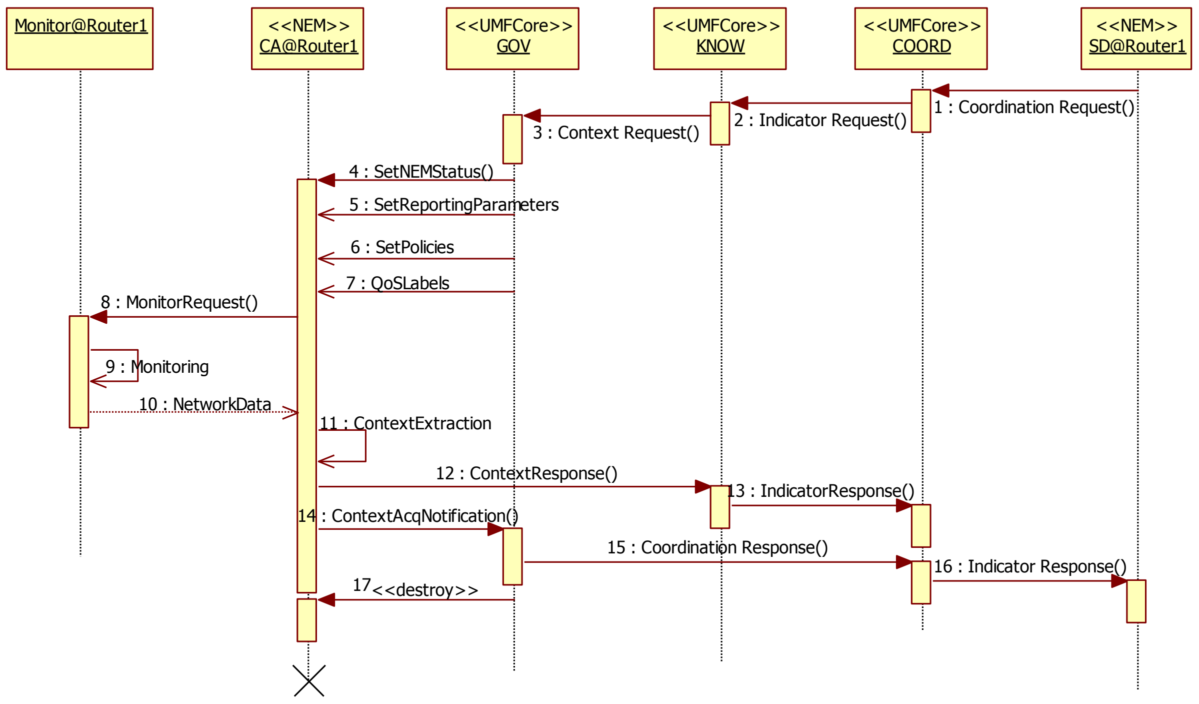


Figure 7: Self-Diagnosis and Context Acquisition NEM communication via UMF.

3 Use Case 2 Related NEMs

3.1 Introduction

Dimensions, dynamicity and complexity of today's networks are growing continuously. Understanding and controlling/managing the network behaviour to meet technical and business objectives is becoming more and more complicated, costly and challenging. This is likely to exacerbate in the future, when it is expected that the networks will become capable of interconnecting large numbers of interconnected real and virtual resources.

Then one of the major challenges of future networks will be managing resources in a cost effective way in order to meet technical and business objectives whilst ensuring stability, in a context which is becoming more and more complex and dynamic. Actually instability in communication networks may have primary effects both jeopardizing the network performance and compromising an optimized use of resources. As an example, instability of end-to-end communication paths may be dependent both on the control plane components (e.g. specific to flow control and dynamic routing) and the management plane operations. Also the arguments for introducing advanced flow admission control[1] are essentially derived from the observation that the network otherwise behaves in an inefficient and potentially unstable manner. Even with resources over provisioning, a network without an efficient flow admission control has instability regions that can even lead to congestion collapse in certain configurations. Another example is the instability which is characteristic of any dynamically adaptive routing system. Routing instability, which can be (informally) defined as the quick change of network reachability and topology information, has a number of possible origins, including problems with connections, router failures, high levels of congestion, software configuration errors, transient physical and data link problems, and software bugs. Eventually another potential risk of instability is emerging from the dynamic provisioning of real and virtual resources in software-defined network. From a security perspective, the stability of a network also depends on its capability to prevent vulnerable configurations. Operations and changes that are performed during self-management activities may generate vulnerabilities that expose the network to multiple security threats.

The main goal of UC2 is to develop and demonstrate methodologies to detect and control the occurrence of instabilities in diverse network contexts. Given the complexity of the matter, the work described here doesn't intend providing a solution for all potential network instabilities; at this stage two approaches are explored (with simulations and prototype demonstration), specifically by solving "constrained optimization problems" (i.e. maximizing of Network Utility Functions) and by preventing vulnerable configurations.

3.2 NEM 15: Self-Organizing Maps in Support of TCP Vegas

3.2.1 Context of the work

This NEM is proposed in the context of UC2 and thus the need for stability in the future dynamic networks. Towards this direction, it focuses on the instabilities that may occur when trying to use the proactive congestion control mechanism of TCP Vegas in a dynamic, in terms of routing, network. In such a case, TCP Vegas is misled by the changing Round-Trip Time (RTT) during a reroute, the increase of which designate congestion for TCP Vegas and thus decreases the congestion window and the utilization of the link. In other words, these misinterpretations lead to instability of the utilization of some links. In order to enhance TCP Vegas functionality and make it more stable, this NEM offers to TCP Vegas the knowledge if this RTT change was related to congestion or to another reason (e.g. reroute due to a fallen link).

3.2.2 Content of the work

TCP Vegas NEM deals with the instabilities and the underutilization of resources by the TCP Vegas algorithm when operating on dynamic network conditions. Under certain circumstances, the congestion avoidance mechanism employed by Vegas misinterprets a change in the network as congestion and forces TCP flows to reduce the congestion window while the appropriate action would be the opposite. As TCP Vegas is based on

[1] Admission control consists in refusing a new flow if the addition of its traffic would lead to an unacceptable quality of service level for that or any previously accepted flow.

RTT delay measurements to proactively avoid congestion, the network change which results in misinterpretation is any re-configuration that would increase the measured RTTs (reroute to longer path, congestion in the backward path-delayed acknowledgements etc) whereas the network change will actually result in an improvement. The purpose of this NEM is to enhance TCP Vegas with learning capabilities using Self-Organizing Maps (SOMs), so as to support it in deciding whether an increase in RTT is caused by congestion or not. This is further translated to a proposal by the learning mechanism to reset the baseRTT (i.e. the minimum observed RTT between two nodes) parameter of Vegas (case of no congestion) or to maintain it (case of congestion).

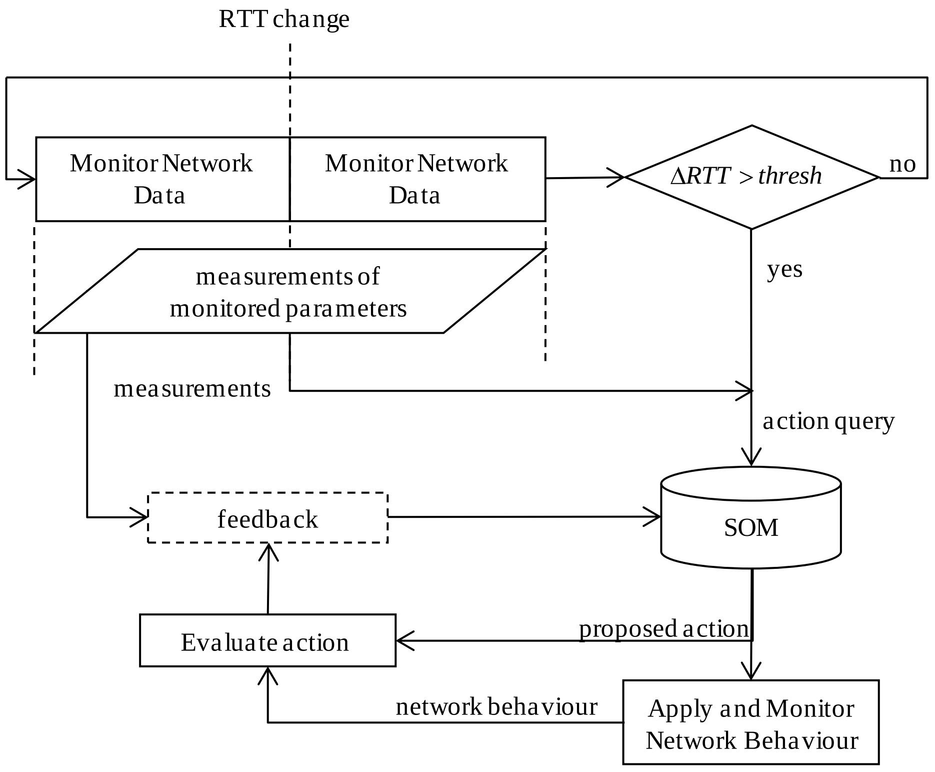


Figure 8: The followed approach for learning whether a baseRTT reset is needed or not

This is achieved following the process below: If a significant change in RTT is monitored, the SOM is queried for the proposed action with inputs such as the average of RTT and congestion window measurements before and after the RTT change (see Figure 8). The proposed action is applied and the flow is monitored for retransmissions. If there are no retransmissions within a certain time window, the correct action is considered to be reset, while if there are retransmissions the correct action is considered to be maintained. The evaluation of the SOM's proposal is fed back to the knowledge base in order to improve further predictions.

As can be seen in Figure 9 and Figure 10, the methodology indeed allows to use higher congestion window that the TCP Vegas on its own and thus enhances the utilization of the network.

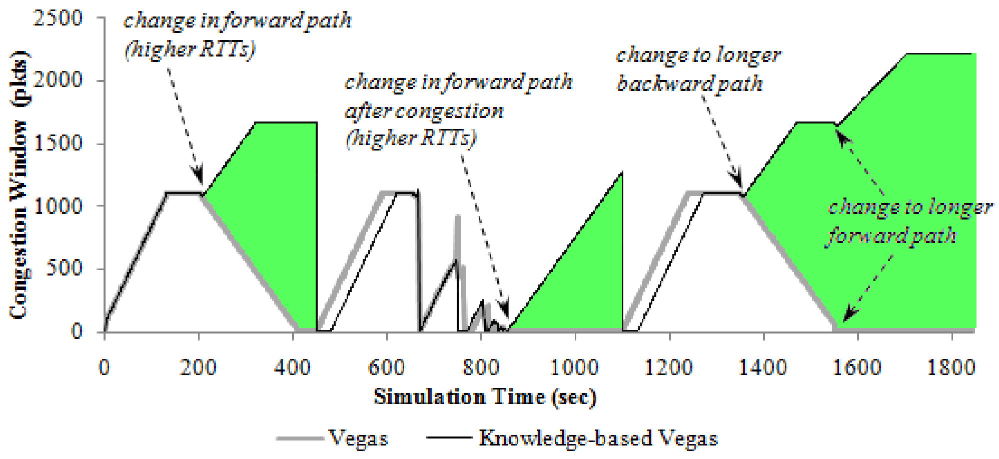


Figure 9: Fluctuation of the congestion window when using TCP Vegas with and without the support of SOM

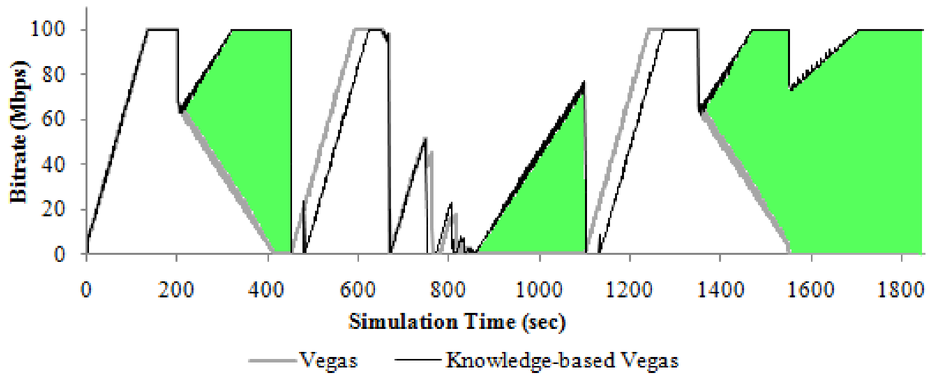


Figure 10: Fluctuation of the bit-rate of the TCP flow when using TCP Vegas with and without the support of SOM

More information and results from the evaluation of the NEM can also be seen in [6].

3.2.3 Merit of the work

The main merit of the work comes from the fact that TCP Vegas is the only congestion avoidance mechanism that acts proactively instead of reactively. In particular, other mechanisms such as TCP (New) Reno and cubic reacts to retransmissions while TCP Vegas acts before any retransmission is needed, it acts based on the RTT that it counts. However, these misinterpretations are a considerable drawback for the future dynamic networks and their stability that needs to be ensured. The proposed NEM abstracts this drawback and makes the proactive congestion avoidance mechanism of TCP Vegas suitable for being used even in the dynamic environments of future network with no risk of instabilities.

3.3 NEM 47: Network Stability and Control

Self-organization in future networks will be achieved through exploitations of “constrained optimizations” (i.e. through protocols, control loops, methods, algorithms…spread across network layers). It should be noted even today Internet protocols can be reverse engineered as “constrained optimization” solvers.

Given the growing complexity of networks, in the future said “constrained optimizations” should be made more and more automatically (self-*). In order to maximize the benefits from “Constraints Self-Optimizations”, (exploited through control-loops…) it is necessary to orchestrate their actuations in order to avoid potential unwanted coupling or even competition causing instabilities, that can jeopardize network performance.

Actually, maximizing profits, minimizing costs, minimizing the loss are typical economics problems, which can be mathematically modelled as CO problems. They concerns the minimization (or maximization) of an objective function subject to constraints on the possible values of the independent variables. Interestingly the objective function can be a cost function (minimization), utility function (maximization), or, in certain fields, energy function, or energy functional. A feasible solution that minimizes (or maximizes, if that is the goal) the objective function is called an optimal solution.

NEM47 is dealing with network stability by progressing above approach. Scope is defining a practical approach for setting-up, configuring and tearing down sets of “control loops” in a network in order to achieve optimization and stability at the same time.

3.3.1 Context of the work

Let’s consider an infrastructure encompassing IT resources (offering virtual processing and storage) and network resources (offering virtual routers) and where (Virtual Machine) VM and (Virtual Router) VR can freely move from one physical node to another (the physical node merely serve as the carrier substrate on which the actual virtual node operate). Assume a dynamic provisioning of virtual resources, both VMs and VRs. This will allow load and traffic variations to be exploited in order to improve performance (e.g. limiting hotspots in the IT resources) and to reduce power consumption in the routers network. In other words, the size of the physical network can expand and contract according to load and traffic demand, by idling or powering down nodes not needed.

In case of hotspots in the IT resources, operators can change the allocation or migrate VMs to improve performance. At the same time, as the network traffic volume decreases, operators can migrate VRs to a smaller set of physical routers and shut down or hibernate unneeded physical routers to save power. When the traffic starts to increase, physical routers can be brought up again and virtual routers can be migrated back accordingly. In summary, in this network scenario there is the interaction of two main control loops: the former is in charge of allocating VMs across multiple networks for performance optimization; the latter is in charge of migrating VRs a smaller set of physical routers for saving power (by shutting down or hibernating unneeded physical routers). Although both control loops would be stable if operating alone, the combination of the two control loops may risk a positive feedback loop. Even if control loops are made explicit and operating regions are well-defined, interactions between them can result in behaviours complex and difficult to understand and control.

NEM NSC (Network Stability Control) is solving this problem. GOV core mechanisms are allowing a Network Operator to program the utility functions, the way to combine them in a global utility function, and also other parameters and elements related to the NEM operations. Overall the NEM NSC optimizes the global utility function in order to find and maintain stable working area in the network phase space.

3.3.2 Content of the work

In UC2, for defining the network stability, the concept of network state is introduced, which is a vector of data (or relevant network parameters, e.g. QoS, etc.) characterizing the state of the network upon a certain set of configurations. Imagine a phase space (with dimensions of said vector) which represents the network behaviour in terms of state trajectories changing over time: this phase space has areas where we want the network state to be, and other areas where we don’t want the network to be. Ensuring stability means avoiding abrupt phase changes in the phase space, specifically moving network states into “not desired areas” of said phase space.

NSC NEM scope is configuring multiple interacting control-loops (or methods) in order to maintain certain stable network states. The design is based on defining and associating utility functions (or functionals) to control loops, (or methods) used for network empowerment, elaborating and then maximizing a global utility function (associated to a network state). As known utility is a value that represents the desirability of a particular state or outcome, and a utility function maps states or outcomes to utility values. Two main reasons are motivating the use of Utility functions:

- Utility functions allow for a separation between the analysis of the data, and the planning and execution mechanisms, with the latter two handled by an appropriate optimization algorithm;
- Utility functions can serve as a very high level specification of the behaviour of the system. This allows business objectives to be directly translated into service level objectives when used with an appropriate optimization and modelling algorithm;

Moreover, research into utility functions progresses continuously. Interestingly it should be noted the new notion of conditional utility and its use to define utility difference networks. The goal of conditional utility is to satisfy additive analogues of the chain rule and Bayes rule, while the utility difference network is similar to Bayesian networks (this is for further study).

In our following examples, elements for estimating utility values of (sub-)networks can be related to a few principles, that can be monitored (and controlled by nodes configurations) Examples are: Availability, Delay, Latency, Network utilization, Network throughput, Network bandwidth capacity, Network costs, Energy Consumption.

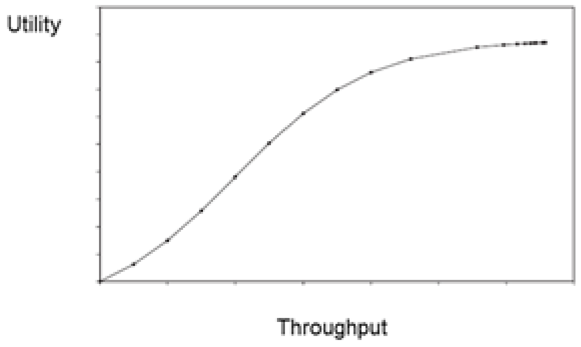


Figure 11: Example: Utility function of Network Throughput

Specifically it is conjectured that the optimization of a certain global utility function ensures network stability (at least in a certain portion of the network phase state): this means maintaining the network state in those areas of the phase space which have been designed and planned and avoiding phase transitions. Eventually, this global utility function can be seen as a multi-attribute utility function: for example a general expression of this aggregation can be a multiplicative form and such forms allow for modelling the interactions between control loops).

The correlation of this phase space with the global utility functionals of the control-loop of the network is still under study and should be part of the overall model.

NEM-47 harmonizes the configurations of a set of control loops or algorithms (intended as network CO solvers) in order to avoid conflicts and instabilities jeopardizing the network performance. Figure 12 is showing the model and the adopted methodology.

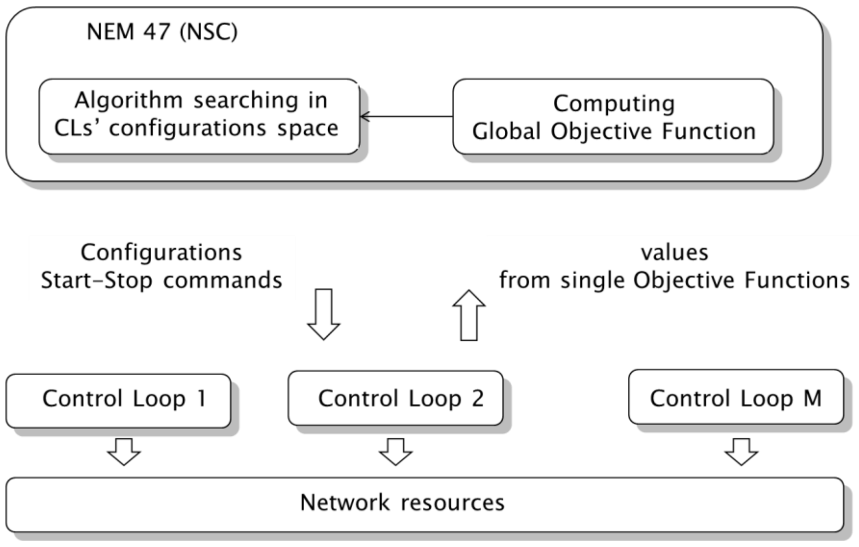

NEM 47 (NSC)
Algorithm searching in CLs' configurations space
Computing Global Objective Function
Configurations Start-Stop commands
values from single Objective Functions
Control Loop 1
Control Loop 2
Control Loop M
Network resources

Figure 12: Basic Principles used for simulating the NEM 47 (NSC)

Each control loop tries maximising its associated objective function. At the same time, an algorithm searches, in the space of control loops configurations (using, for example, a beam-search algorithm), those configurations allowing maximising the global objective function (which is a combination of the single objective functions). This is done at regular intervals, upon reaching a trigger or in reaction to changes in the global objective function.

Other methods are under consideration for automating the creation of utility functions: for example evolving them through genetic programming. To construct the utility function, the genome consists of a predicate grammar. On the other hand, such utility function can only provide Boolean values, where one of the main attractions of utility is to provide comparable values for optimisation.

3.3.3 Merit of the work

Preliminary studies results reported in Deliverable D3.2 have analysed stability in a network with several control loops. Analysis started by considering aspects related the communication and interactions between several controllers (which is one of the “Unsolved Problems in Mathematical Systems and Control Theory (Problem 4.4)” [41]. Specifically, analysis has assumed that each controller has to control its sub-network and it has just a partial observation of the overall network over which the end-to-end services are provisioned. The exchange of information between controllers implies only partial observations, state estimates, or input values; moreover there are constraints on the communication between controllers. The problem investigated in this context has been the coordinated control, i.e. the reaching of a stable consensus control of the controllers of the sub-networks composing the overall network. Not only the control objective should be reached, but it should be stable. Contributions in D3.2 have concerned the analysis the Kuramoto oscillator model: the assumption has been that synchronization problem of a network of oscillators is similar to consensus reaching (e.g. for synchronising certain actions or simply for start-stop) in a network of distributed controllers.

Following step (in preparation of the contributions to this Deliverable) has been extended this basic idea by using an approach which is feedback based (replacing Kuramoto oscillators with control loops). A feedback control-theoretic formulation of a self-* problem has involved identifying the primary components: the target system, the controller, the controlled variable (measured output), the manipulated variable (control input), and possibly transducers and a filter. For example in the case of (competing – cooperating) controllers in charge of resource allocations, we have use measured values of performance parameters (as feedback) to tune the control-loop (resource allocations). With this approach the controller gains are variables that must be configured according to the design requirements. These gains to steer the system towards a balance between the QoS (e.g. response time and the CPU share allocated to VMs, or energy consumption and resources allocated to VRs). Next step has been extending this approach to a more general methodology, based on Utility functions.

The merit of the contributions reported in this deliverable is having started elaborating a more general methodology (which is implemented and tested through simulation of NEM 47, NSC) based on defining and associating utility functions to above said controllers/control-loops and elaborating a global utility function (as an aggregated function of single utility functions) for network optimization and stability control. Key observation is that network might be viewed as “self organizing,” but that is achieved by “constrained optimization” (through control loops, methods, algorithms, etc).

We’ve started by [42] and [43], where Kelly et al. presented an innovative idea of formulating a network constrained optimization problem in terms of maximizing a utility function where the variables are the source rates constrained by link capacities and the objective function captures design goals. Merit of the work has been extending the idea of using the language of Network Utility Maximization (NUM) to distributed network resource optimisation and allocation. Also cross-layer interactions may be characterized by viewing the process of “layering as decomposition of a given NUM problem into many sub-problems. These sub-problems are “combined together” by certain aggregated utility functions. Actually, utility functions can be constructed based on user behaviour model, operator cost model, or traffic elasticity model. Eventually, in the contribution it is conjectured that the optimization of proper aggregated or global utility function ensures network optimization and stability. We have developed some optimized algorithms for assessing this approach from a quantitative viewpoint (simulations are still on-going and numerical results will be reported in the next deliverable).

3.4 NEM 13.1: Vulnerability Management

3.4.1 Context of the work

Autonomic computing constitutes a major paradigm for dealing with the complexity of large-scale network management. However, human administration errors and changes operated by self-governed entities may generate vulnerable states exposing the network to a wide range of security threats. Accordingly, vulnerability management plays a crucial role for maintaining safe configurations on devices in these environments. In addition to local vulnerabilities, distributed vulnerabilities have also to be assessed over a consolidated view of the network in order to detect vulnerable states that may simultaneously involve two or more devices.

3.4.2 Content of the work

The objective of this work is clearly to prevent vulnerable configurations that expose autonomic environments to security and safety threats. In order to increase the vulnerability awareness of autonomic environments, we have proposed to support the integration of vulnerability descriptions into their management plane [14]. We have therefore defined a policy-based strategy where knowledge taken from security sources such as OVAL [15] vulnerability description repositories is dynamically translated into policy rules. We have mathematically formalized the mapping of OVAL vulnerability descriptions into policies, defined a translation algorithm and implemented it into Ovalyzer, an OVAL to Cfengine [16] translator. The generated policies enable Cfengine agents disseminated in the network to assess network devices and to generate alerts when configuration vulnerabilities are observed (see Figure 13).

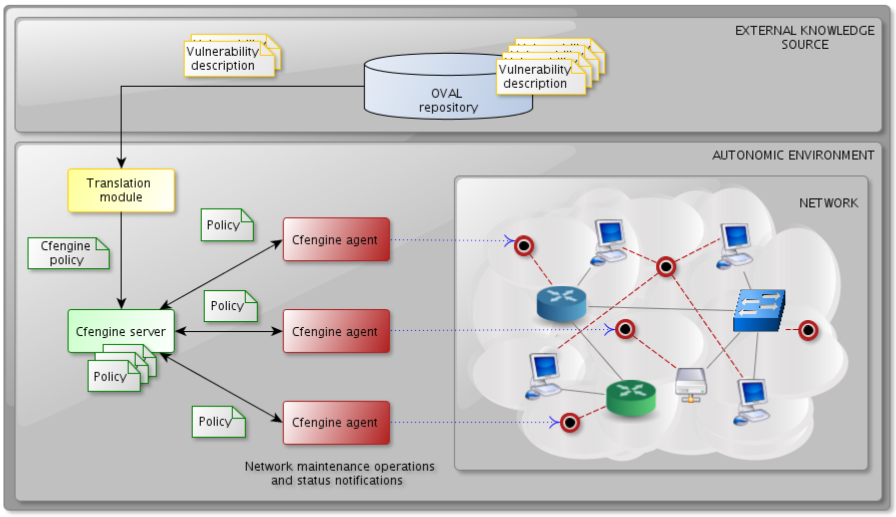


Figure 13: Vulnerability Assessment Strategy

Ovalyzer currently supports the translation of OVAL definitions for the IOS platform where only three plugins are required as illustrated in **Figure 14**. These plugins are capable of translating the appropriate OVAL tests used within all the vulnerability descriptions for IOS. The translation statistics depicted in Figure 15 show the feasibility of our solution where a linear behaviour in terms of the number and size of the generated files can be clearly observed. In order to extend our strategy to distributed vulnerabilities, we have introduced the DOVAL approach [17]. Traditional network-level vulnerability management mechanisms perform a global analysis by investigating each network element individually. Even though such approaches can detect sets of vulnerabilities that may allow an attacker to perform a multi-step attack, they do not provide the capability of detecting vulnerabilities that simultaneously involve two or more devices under specific conditions. The underlying problem relies in that each network device can individually present a secure state, but when combined across the network, a global vulnerable state may be produced. In order to cope with this problem, formal mechanisms for describing distributed vulnerabilities are required. Moreover, using standard means for achieving such objective can promote the exchange of security knowledge among practitioners and organizations.

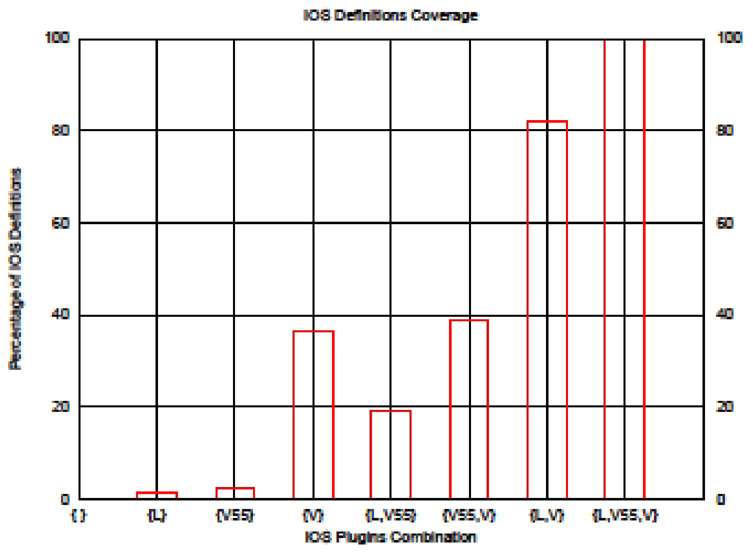


Figure 14: Vulnerability description coverage

Through the DOVAL approach, we have mathematically defined the concept of a distributed vulnerability and have inferred an OVAL-based language capable of expressing these formal constructions. The overall approach of DOVAL is illustrated in Figure 16. As in OVAL, DOVAL descriptions can constitute useful security repositories that in turn can be exploited by self-managed environments in order to ensure safe configurations. Based on this language, we have investigated optimized algorithms and collaborative strategies for assessing such descriptions. We have performed several experiments considering different probabilities for network devices to be involved in distributed vulnerabilities. The results show a linear growth rate in terms of time with a sequential assessment and a constant time when parallel assessment is performed.

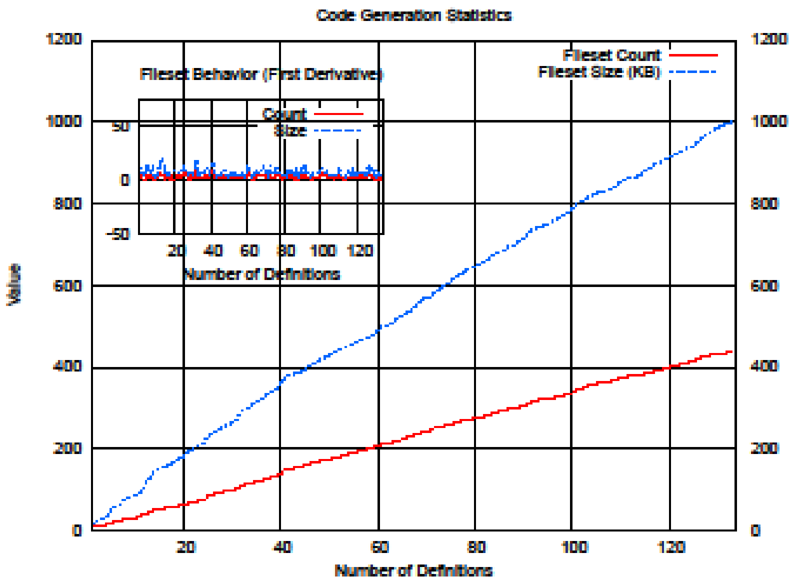


Figure 15: Policy generation statistics

3.4.3 Merit of the work

This contribution enables autonomic agents to assess their security exposure. Such property increases the overall security of autonomic environments and constitutes a foundational concept for building self-governed systems. With respect to deliverable D3.2, we have extended our approach to distributed configurations and performed further experiments. In this work, we have proposed mathematical models for both host-based and distributed vulnerability assessment methods. These models provide a robust background for building well-founded solutions from both a theoretical and a technical viewpoint. We have also developed optimized algorithms for evaluating security threats in autonomic networks as well as several experimental scenarios that show the feasibility of our solution. Usually, the concept of a distributed vulnerability is linked to the assessment activity over a network where its member devices are analysed sequentially or in parallel, but always individually. We believe that even though the assessment mechanism may be distributed, it is still a device-based approach what is actually being used on each device. In other words, the method used for auditing security issues in a network is distributed though it is actually a composition of individual activities performed on each device. We have extended this approach by considering a broader vision of the network and its relationships, being able to identify security problems that simultaneously involve more than one device and that are uncovered by traditional assessment mechanisms. Our work provides a novel approach for vulnerability management where new vulnerable scenarios (distributed) as well as normal ones (device-based) can be covered under a unified perspective. Such perspective provides the ability to describe a wider set of security advisories in a well-organized and standardized manner, thus highly increasing the vulnerability awareness of autonomic networks and systems.

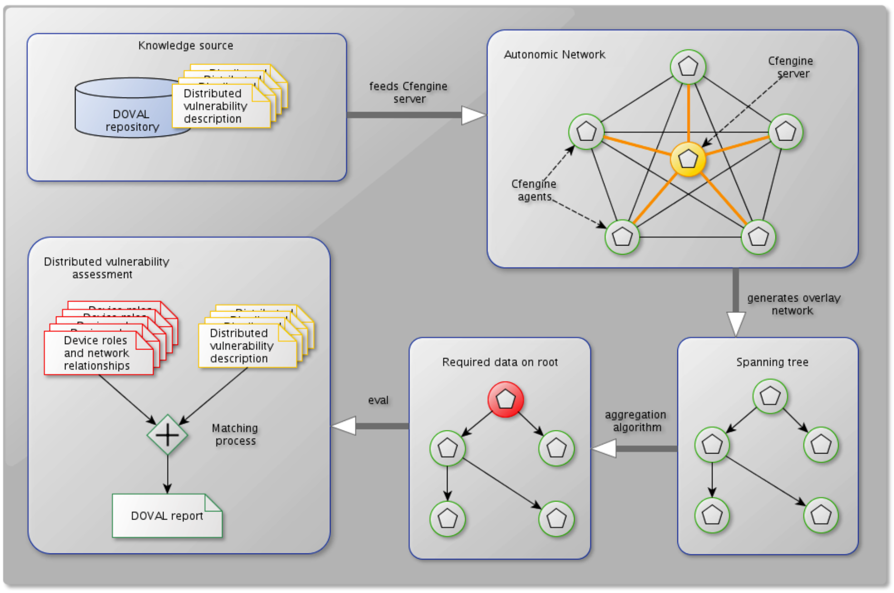


Figure 16: DOVAL Approach

3.5 Discussion

NEM NSC aims at providing a solution to one of the major challenges of future networks, which will be managing resources in a cost effective way in order to meet technical and business objectives whilst ensuring stability, in a context which is becoming more and more complex and dynamic.

3.5.1 Simulations of NSC: A flexible network moving VR and VM

A first example will be used to describe the benefit of having in the UMF a NEM NSC. Let's consider the infrastructure described in the previous section where VM and VR can freely move from one physical node to another (the physical node merely serve as the carrier substrate on which the actual virtual node operate). A Provider (or a Business Unit of a Provider) A develops and deploys a cloud-based service/application, using virtual nodes (providing computing, storage and router resources) provided from physical infrastructure Provider (or a Business Unit) B. For simplicity, assume A leases two virtual nodes (node 1 and node 2) from B, and dynamically load-balances (to avoid hotspots) incoming requests across the virtual nodes. Assume A's load balancer control loop operates in a control loop with a 1-minute period: after each minute it evaluates each node's current load based on that node's response time statistics during the past minute, and shifts load during the next minute to the less-loaded node. Suppose B running a second control loop, which attempts to optimize the power consumption of physical nodes. This control loop also has a 1-minute period: after each minute, control loop measures each node's utilization during the past minute and then can migrate virtual routers to a smaller set of physical routers and shutdown or hibernate unneeded physical routers to save power.

Suppose during one minute a hot spot in node 1 happens. A's control loop notices this and shifts some VM away from node 1 to node 2 in the next minute, while B's power optimizer notices traffic surge in node 1 and moves more VR to that node and reduce the number of VR in node 2 (saving energy in node 2). While either of these actions alone would lead toward convergence, the two in combination cause overcompensation: during the next minute, node 1 becomes more underutilized than it was over-utilized in the previous minute. Although both A's and B's control loop would be stable if operating alone, the combination of the two control loops may risk a positive feedback loop.

Use case analysis is based on forward and reverse engineering. In reverse engineering, global utility function (i.e. the utility function of the control loops) is implicitly determined by the two given control loops already, and is to be discovered rather than designed. In forward engineering, it have to be properly selected (e.g. as a combination – sum or product - of two sigmoid functions, one function of response time, the other one function of the routers energy consumption). In summary, in this use case we can see the interaction of two main coordinated control loops: the former is in charge of allocating VM across multiple networks for performance optimization; the latter is in charge of migrating VR a smaller set of physical routers for saving power (by shutting down or hibernating unneeded physical routers).

It should be noted that although both control loops would be stable if operating alone, without a proper coordination, the combination of the two control loops may risk a positive feedback loop.

The utility function of the control-loop in charge of allocating or migrating VM across multiple networks is based on optimizing certain performance parameter (or set of parameters); for example, we may consider a function that indicates a decreasing utility as the response time increases (but any other functions could be considered depending on the required metrics).

The algorithm for moving VM proceeds by considering the highest loaded VM on the highest loaded server and determines if it can be moved it on the least loaded physical server. The move is feasible only if that server has sufficient idle CPU to meet the desired resource allocation of the candidate VM. If sufficient CPU is not available, then the algorithm examines the next least loaded server and so on, until a match is found for the candidate VM. If no physical server can house the highest loaded VM, then the algorithm moves on to the next highest loaded VM and attempts to move it in a similar way. The process repeats until the utilizations of all resources on the physical server fall below their thresholds.

The utility function of the control-loop in charge of migrating VR a smaller set of physical routers can be based on saving electrical power.

Concerning the algorithm for moving VR objective function could be minimizing the overall CPU load physical routers – depending on the traffic – in order to move in idle or sleeping states the largest number of nodes. The algorithm may consider the heaviest physical routers and their nearest and lightest neighbour to exchange one or more virtual routers.

Regarding the energy savings, we've use the data reported in [18]: the base router system (chassis, processor and switching fabric) consumes about 430W. It has 8 slots for line-cards and the average consumption of a line-card is about 70W. The overall power consumption is 990W when the router is operating at full load. Then we have assumed that the router consumes about 60% of full load power consumption when it is idle (and from 10% to 20% when the node is into sleeping state i.e. router switched off): for example a line-card consumes 70W when working, 40W in an idle state. For sake of simplicity, we've assumed two states: full load and idle (i.e. hosting no VR).

If the two utility functions should not be maximised independently otherwise instabilities may occur. A global utility function should be addressed whose utility function is a combination of the above two functions.

In the simulation activities, the global control loops algorithm searches the space of control loops configurations using (for example) a beam-search algorithm. This could be done at regular intervals, upon reaching a trigger or in reaction to changes in the global utility function.

3.5.2 Workflow of the pathological scenario

Without UMF and NEM NSC instabilities may occur (as explained before)

- both control loops optimize independently from each other just according to the respective utility functions

With the UMF and NEM NSC the risk of instabilities is controlled:

- Information from control loop 1 to NEM NSC: metrics, time, utility values …
- Information from control loop 2 to NEM NSC: metrics, time, utility values …
- NEM NSC (in coordination with a UMF core method?) detects potential conflict between control loops or occurrence of instability. The following actions is performed to avoid the conflict:
    - The global utility function is computed
    - Configuration parameters are used to configure and active/deactivate the two control-loop

3.5.3 Enhancing TCP Vegas

Here we show the usefulness of UMF using the specific example of NSC and TCP Vegas adaptation.

TCP Vegas has the main objective of maximizing the aggregate utility of all sources subject to the capacity constraints of the network's resources, with a log utility function. Moreover, the congestion avoidance mechanism of Vegas can be interpreted as an approximate gradient projection algorithm to solve the dual problem. This model suggests that Vegas stabilizes around a weighted proportionally fair allocation of network capacity when there is sufficient buffering in the network, that is, when the network has enough buffers to accommodate the extra packets the algorithm strives to keep in the network. If there are no sufficient buffers no equilibrium is reached.

In this direction, TCP Vegas uses RTT. However, RTT may change due to another reason as well and not only due to a congested link, e.g. due to a rerouting. These cases are misinterpreted by TCP Vegas as congested links leading him to erroneous reactions (i.e., decrease of the rate with which it sends the packets). TCP Vegas NEM is used to build knowledge on such occasions and to guide TCP Vegas to avoid such misinterpretations.

NSC on the other hand aims at providing a solution to one of the major challenges of future networks, which will be managing resources in a cost effective way in order to meet technical and business objectives whilst ensuring stability, in a context which is becoming more and more complex and dynamic. Towards this direction, utility functions related to different aspects of stability in the network are used.

TCP Vegas NEM can provide such information to NSC. In particular, the metric that could be fed from TCP Vegas NEM as an input to the NSC NEM is the number of variations of the Congestion Window.

UMF, and more specifically KNOW UMF block, is used to enhance the information exchange between these two NEMs. For example, KNOW block holds this information for NSC, informs NSC if a significant change has occurred and can provide aggregation of the information if needed.

4 Use Case 3 Related NEMs

4.1 Introduction

Due to the fast pace of technological evolution in the field of wireless access technologies and user applications, the existing telecom networks are facing increasing pressure to meet the QoS/QoE demands of mobile users regardless of the type of UE and the underlying access network.

The use of time sensitive and bandwidth intensive applications, e.g. mobile video traffic (streaming and broadcast), is becoming an increasingly pervasive application and the QoS/QoE demands for the provision of such application ubiquitously by mobile users is putting a lot of pressure on mobile network operators and their respective infrastructures (especially the core and backhaul). To effectively meet the customer demands requires increasing network complexity resulting in increased CAPEX/OPEX.

At present, most of the network/content/application service hosting and management is being concentrated at the core. As a result all the user traffic has to go through the core over the backbone/backhaul and hence a lot of resources (bandwidth and processing wise) are consumed. This centralized organization of networks imposes serious operational and performance demerits, especially in case of mobile users using bandwidth intensive real time applications (such as streaming video content etc.). Due to mobility, such a centralized system poses several issues due to the constant re-routing of user data to its new point of attachment and this would imply dynamic management of bi-directional tunnels maintained between the UE and the Core (more specifically the PDN-GW). Such a scenario thus imposes extra burden on the backbone and the access links and can significantly impact QoS delivery to mobile users. Also such a centralized architecture makes it difficult for new service introduction and incurs high management costs. Most importantly, the concentration of service/content management introduces a single point of failure, and can potentially increase end-to-end delay for mobile users using real-time applications.

In view of this, UC3 focuses on the dynamic virtualization and migration of data/content and network entities (gateways and servers) nearer to users. The motivation is to provide/resolve the most frequently used resources/content/functions nearer to the mobile user in order to release the resource (e.g., bandwidth, processing, tunneling, routing) consumption from the core and distribute them autonomously, intelligently and dynamically (on a use-case basis) towards the edge of the network (i.e., access network). This approach is expected to enhance the overall QoS/QoE for the users with efficient resource utilization.

It also focuses on context-aware methods for delivering progressive video streaming services to multiple mobile users with user expected QoE over resource constrained wireless links.

4.2 NEM 7: Context Discovery from raw data

4.2.1 Context of the work

Context data extracted from users and environment, as well as the status of the network resources (nodes and links) are in the form of binary data. The raw data gathered from different sources are at the first instance meaningless to be relied on by an autonomic management system and should be processed and reasoned. The data processing from raw contextual data can be performed by using different techniques, including learning techniques, data mining and rule-based reasoning. Here we focus on rule-based reasoning.

The simplest form of a rule is and 'If… then… (else…)' construct. In a simple way, the reasoning is made by evaluating correctness of a predicate, and reasoning the consequent or alternative. An example of applying a rule for a context aware autonomic management scenario is:

If user-exits-from-the-office
then enable-offline-bulk-download

Reasoning from the contextual data is not always as simple as the above. There are situations, such as existing inadequate, incomplete or irrelevant information that doesn't fulfil a particular predicate hence not satisfying the rule. An example of such imperfect context and the applied reasoning is *defeasible reasoning* which is applied for contextual reasoning in [18]. In this work, local defeasible theories, defeasible mapping rules, and a preference ordering on the system contexts are extensions to multi-context systems.

Rule satisfaction rate can be improved by relying on external related information. Supportive information can be supplied from an external knowledge source, such as Internet. Semantic web can be employed here for gathering supportive information from the internet. In our proposal, information from Wikipedia, in combination with the contextual data can be used. Finally the reasoning from the combined information can be made by rules.

The reasoning process starts with annotation of the raw data with the information collected from semantic web [19]. More specifically, the data gathered from the resources, will be semantically enriched in a proper format (e.g. based on Resource Description Framework [RDF] or Web Ontology Language [OWL]), constructing ontology of the context data and annotating it with Linked data (such as DBpedia [20]) and reasoning from the semantic annotated data by applying rules expressed in proper language (e.g. SWRL). The external knowledge source can be queried by means of a proper querying language (e.g. SPARQL in the case of DBpedia). The entire process is depicted in Figure 17.

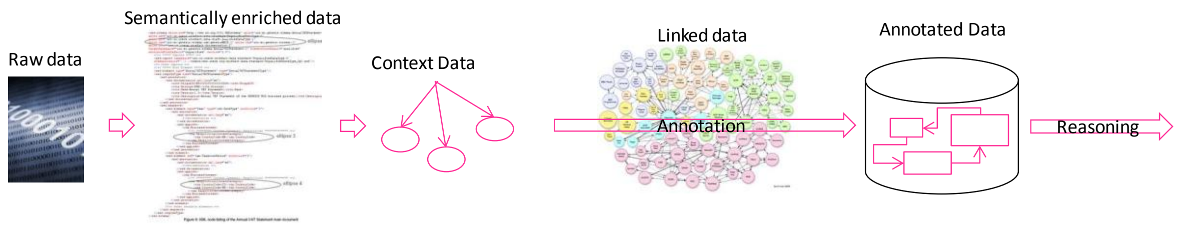


Figure 17: Reasoning process from contextual data

An example of usefulness of such reasoning follows here. Assume that the user movement pattern indicates that the user visits a particular shopping mall every Saturday. Annotating the data from the user's context (location taken from the GPS) and the information from the shopping mall (size, number of shops, number of annual visitors, etc.) can lead to reason user demands (access to particular web-based pricing web sites) to be locally cached in advance to a location closer to the shopping mall.

4.2.2 Content of the work

The first step of work focuses on the last stage of the processing: rule-based reasoning. For this step, the reasoning is made based on assumptions on annotation of contextual data with linked data. For this part, we assume that contextual information is annotated with linked data sampled from an exponential distribution with a predefined mean and standard deviation. The last stage of the reasoning is made in NS-2 [21]. A network is modelled containing several sources of contextual information (e.g. temperature and light sensors, movement and acoustic noise detectors, user's location and phone call patterns). In this model, a set of assumptions for the values of these sets of information are made and regarded as the reference point. The rules are static based on which the reasoning are made. The metric here is the success rate of discovered context compared to the referenced context. In the unprocessed case, context discovery relies on the static lookup tables based on the information regardless of the annotated data from linked data and also rules applied to the annotated data. The processed data is the result of applying the rules on annotated data and the amount of annotation made on the enriched context data.

The preliminary results show the improvement in success rate considering the real expected number of correct results over all the requests as a metric. In the unprocessed case, the average success rate over 50 runs each making up to 30 queries is investigated. In this case, the success rate is around 35%. In the processed case, the success rate improves up to a level of around 65%. There are still unsuccessful discoveries, which are because of unsuccessful replies from the network, unavailability of information sources, inadequate annotation of the linked data, etc. Figure 18 shows the results taken from the simulations.

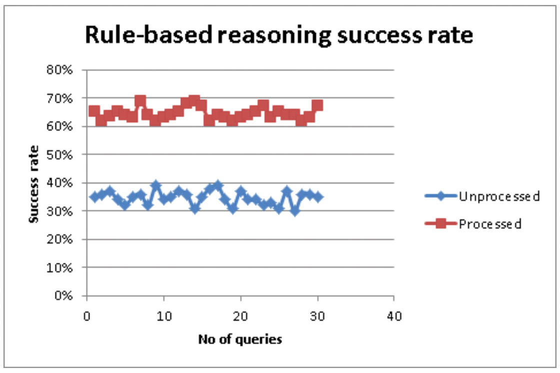


Figure 18: Success rate of rule-based reasoning from contextual data

In the continuation of work, traces of linked data will be used as source of data and integrated with the process. Moreover, other aspects of performance of this mechanism will be investigated, including evaluations based on other metrics, such as accuracy and cost.

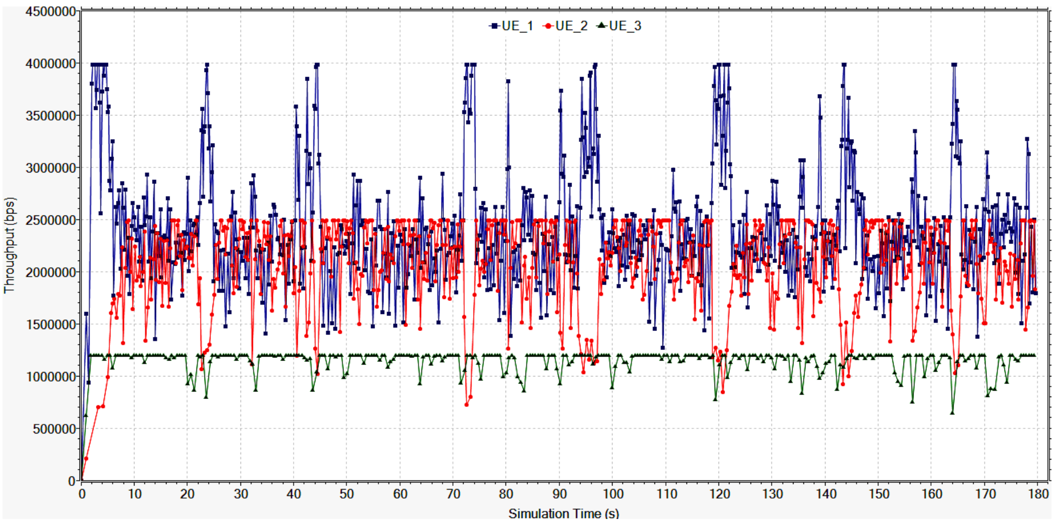
4.2.3 Merit of the work

Context discovery in UniverSelf is an important component contributing to the development of knowledge. This process is under development as a knowledge-building NEM; compared to D3.2, the reasoning code has now been implemented in simulation environment. The aspect of user and environment context is the focus-point of this work; however the algorithms are also applicable to network context. Based on the reasoned context, other performance improvements such as flow optimisation and data migration can be executed.

4.3 NEM 45: Codec Selection and Fair Scheduling

4.3.1 Context of the work

With the proliferation of smart phones and the exponential increase in demand for video, it has become a challenge to deliver popular video services (like YouTube and similar services) to multiple mobile users in a resource constrained wireless access networks. The main operational objective is to meet the user expected QoE while delivering such services.



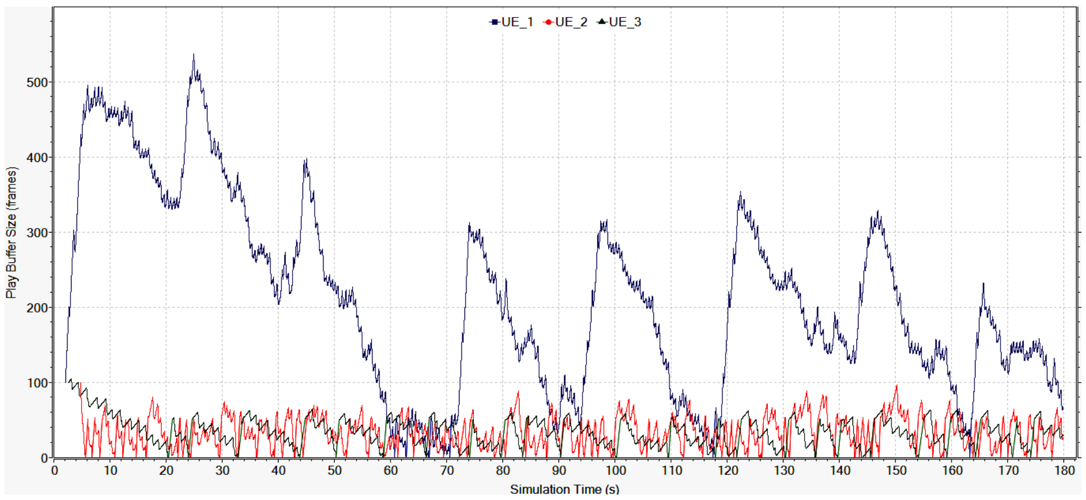


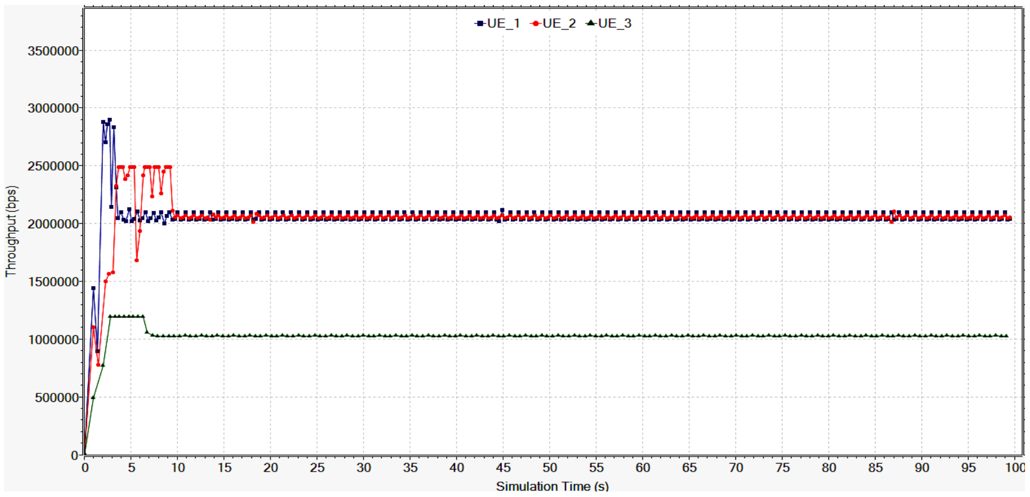
Figure 19: Throughput performance and buffer utilization at the clients without scheduling (a) Throughput experienced by three users. (b) Play buffer size variation at the clients' side

Such video services are delivered typically using HTTP-over-TCP, whereas the TCP ensures the reliable delivery of video data, which is being progressively delivered to the users. However, it has been observed in our simulations that ensuring the reliable delivery of video content will not ensure meeting the user expected QoE. This is due to the RTT-unfairness issue that is inherent in TCP, where the users with low RTT will be favoured over users with long RTTs. Also users with high access bandwidth will experience better throughput than users with low access throughput, in case there is congestion in the backhaul link. This is shown in Figure 19(a), showing the throughput experienced by 3 users, where UE_1 and UE_2 have access bandwidths of 4Mbps and 2.5Mbps respectively, whereas it is 1.2Mbps for UE_3. The backhaul link has a bandwidth of 5.8Mbps and is considered a bottleneck link. Figure 19(b) shows the buffer variation at the clients depicting frequent buffer starvations and hence interrupted viewing.

It is therefore required to develop a method that will balance the flows for different users in a way that the QoE of each user is preserved.

4.3.2 Content of the work

The main objective of the work is the design and development of a codec selection algorithm and an application level scheduler for ensuring to meet the user expected QoE during the progressive delivery of over-the-top (OTT) video content. In this respect the base framework of an application level scheduler has been developed, and Figure 20(a) shows the stable throughput experienced by all the three users while each user experiences uninterrupted viewing of the video content (see Figure 20(b)).



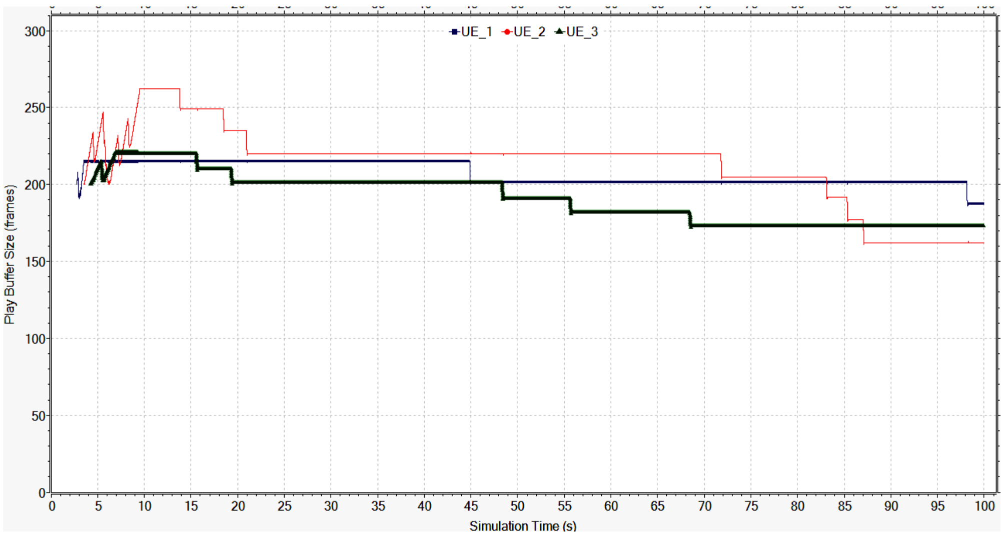


Figure 20: Throughput performance and buffer utilization at the clients with application level scheduling. (a) Throughput experienced by three users. (b) Play buffer size variation at the clients' side

This work is also taking into account context information (user, network, session, client buffer size etc) that will be correlated to derive knowledge enabling the dynamic scheduling between simultaneous flows in order to adapt to the dynamic and varying nature of the mobile traffic. Based on prevailing network conditions, a suitable codec profile will be selected for the user and then the scheduler will ensure optimal delivery of content data.
Finding the optimal policies for codec selection and fair scheduling of the simultaneous video streams can be seen as a stochastic optimal control problem. Though the continuous state space brings some complications, reinforcement learning (RL) can be applied to find optimal/good policy. For example, Tham and Hui [22] introduced RL based solution for dynamic bandwidth provisioning in DiffServ networks. This approach seems to be applicable for the rate control of video streams without big changes. If the feedback control is needed for codec selection, then there are several options like fuzzy logic combined with RL or case-based reasoning.

4.3.3 Merit of the work

The application flow control method (i.e., the Block Sending Algorithm) adopted by Youtube for the progressive delivery of video content is not optimum as it can cause congestion and packet losses (due to transmission of large bursts of packets) resulting in low throughputs and poor QoE. This is especially true for residential DSL clients where the block sending approach is responsible for over 40% of packet loss events resulting in data retransmission rate of 1.5% of all bytes sent once block sending began [23].

Our work has the potential to impact and optimize the efficient delivery of video services and maximize the utilization of available network/radio resources, especially in wireless cellular networks.

4.4 Discussion

NEM 45 has been proposed by NEC in order to select suitable codec profiles for mobile users that want to access YouTube-like services, and then ensure that the video content is being progressively delivered to multiple users, sharing the same channel or associated with the same Base Station (BS)/eNodeB(eNB) the user, in a fair manner.

The Codec Selection algorithm will ensure that the codec selected for/by the user is appropriate to the user device capabilities and underlying network conditions. Once selected, the fairness aspect will ensure that each user, regardless of its relative position/distance from the BS/eNB is able to receive video content at rates that will ensure the user to view it with the expected QoE. In other words, fairness will ensure that users with favourable channel conditions do not hog the bandwidth from users with lesser channel conditions. This is typically the case with TCP RTT un-fairness that will set large congestion windows (cwnd) for users with low RTT delay than for users with high RTT delay. The fairness algorithm ensures that each user is able to receive data at rates that will ensure that the playout buffer does not get starved, which otherwise would lead to image stalling or jerkiness degrading the QoE.

However, the CSFS NEM is dependent heavily on the timely availability of good quality context information. The context information can comprise User context, device capabilities information, session information, network conditions etc, or a combination of these, for the CSFS NEM to impart optimum performance.

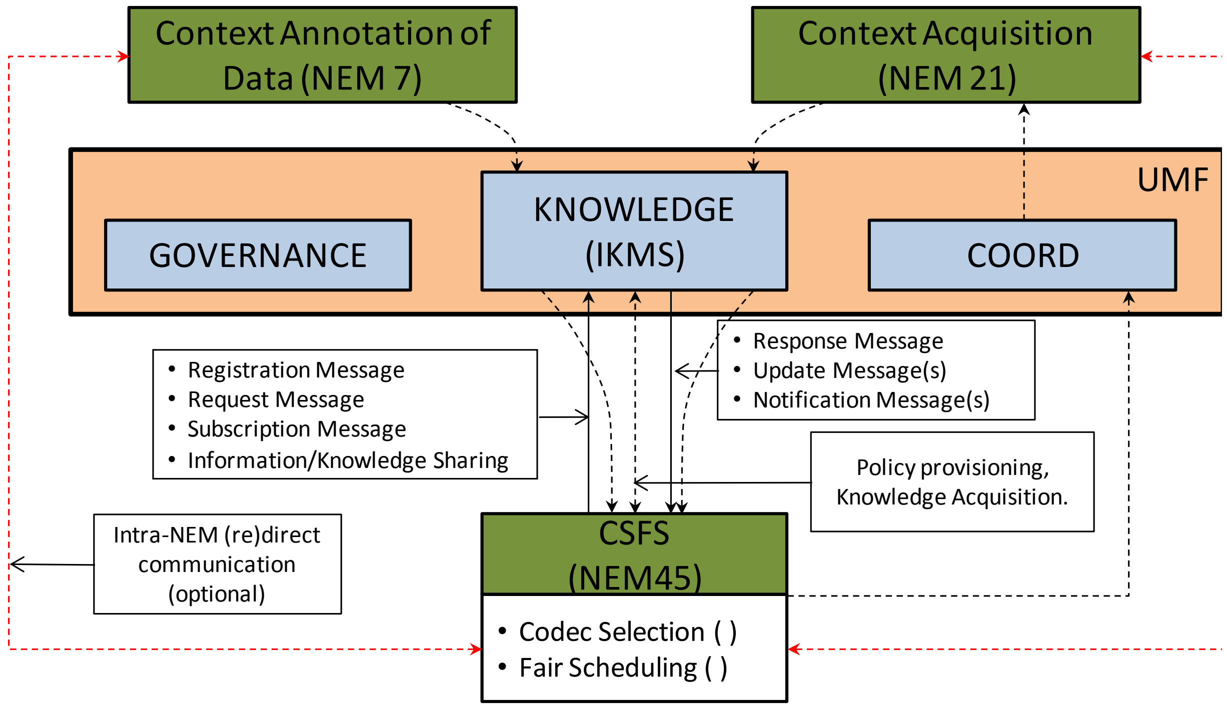


Figure 21: Conceptual overview of the interaction of NEM 45 with NEM 21 and NEM 7 via the UMF

Since CSFS NEM can be characterized as an “Actor NEM”, it relies on other Knowledge building/information sharing NEM. For this purpose, the CSFS NEM needs to intrinsically link with the services/features offered by NEM 21 (Context Acquisition) and NEM 7 (Context Annotation). It should be noted that NEM 21 monitors specific network KPIs (i.e. Delay, Jitter, PL) that are processed and classified in order to produce a QoS label that reflects the state of the network element.

For this purpose, the UMF will provide all the necessary functions that will allow the NEM 45 to interact with NEM 21 and NEM 7. NEM 45 should trigger NEM 21 via COORD and GOV, with the following sequence of events:

1) NEM 45 notifies COORD that needs interaction with NEM 21

2) COORD triggers GOV for this interaction.

3) GOV triggers NEM 21 to start, providing also the respective inputs.

4) NEM 21 will then start collecting the *raw context data* from relevant context sources

5) NEM 21 indexes and optionally stores the results (in the form of QoS labels and/or the raw KPIs) to KNOW and notifies GOV that it has finished its operation.

6) GOV responds to COORD and NEM 45 takes the results from NEM 21 or optionally KNOW.

NEM 45 can also potentially (and optionally) utilize NEM 7 by providing it with the policy information on the basis of which NEM 7 will process the results produced by NEM 21in order to derive the knowledge as per the *knowledge boundaries* specified by NEM 45, again via the UMF's KNOWLEDGE block. To achieve this, NEM 7 annotates the KPIs with the user and environmental data taken from different sensor networks. After this enrichment, the NEM 7 will optionally use Linked Data for further tagging and rating the contextual data to prepare the data for reasoning. In the reasoning stage, based on the policies specified by the NEM45, NEM7 will rely on certain rules and create necessary ontology from the raw context data and sends it back to NEM45 via the UMF's KNOWLEDGE block to take necessary actions. In summary:

1) NEM 7 receives the KPIs from NEM 21 or optionally KNOW storing the information produced earlier by NEM 21

2) NEM 7 relies on location (GPS, RFID tags attached to the users, Bluetooth devices connected to locationing sensors, etc.), environmental sensors (light, acoustic noise, temperature), body sensors (e.g. heartbeat sensor) and enriches KPIs by annotating these data to the KPI

3) NEM 7 performs the first stage of reasoning, and tries to reason the context. The reasoning can be in the form of e.g. user is on their desk, in a meeting, at gym, shopping mall, or relaxing at home, etc. This helps in reasoning the user demands based on the policies provided by NEM 45.

4) NEM 7 optionally relies on Linked Data to determine the context of the user, e.g. what is the best approach for providing the service to the user based on the existing knowledge. As an example, in case the information at stage 3 above is not accurate enough (e.g. the user's location is determined with an accuracy of +/- 500 m, or sensors are 15 km away from the user), then linked data will be used for further second stage reasoning.

5) The final contextual data will be provided by relative interfaces and indexed and optionally stored by KNOW.

5 Use Case 4 Related NEMs

5.1 Introduction

UC4 on “SON and SON collaboration according to operator policies” in LTE and LTE-Advanced networks, deals with the design of NEMs with a strong focus on self-optimization functionalities such as Coverage Capacity Optimization (CCO), Inter Cell Interference Coordination (ICIC), or load balancing. Two NEMs implementing Reinforcement Learning (RL) techniques are studied in the UC for the design of efficient self-optimization. The first investigates Fuzzy Q-Learning (FQL) and is implemented for self-optimizing a femtocells network. The FQL learns to optimally adjust femtocells transmit power to maximize the SINR of the indoor users while minimizing the interference imposed to the other passing by users in vicinity of the small cell base station. The second technique adapts the Policy Gradient Reinforcement Learning (PGRL) to the problem of resource sharing between the backhaul link and direct station to mobile links. The corresponding NEMs implementing the learning based self-optimization features can be embedded in the control plane, namely within the stations (femtos and relays). In the case of PGRL, an “always on” learning operation mode can be used. The operation of the NEMs can be governed via GOV core mechanism that can provide the specific choice of reward which guides the self-optimization process.

5.2 NEM 12: Coordinating CCO and traffic balancing

5.2.1 Context of the work

Relay stations are part of Heterogeneous Networks (HetNets) that have been introduced in 3GPP Release 10 as a mean to enhance coverage and capacity of the LTE-Advanced network. We consider here in-band relays in which both direct links (stations: macro/relays to mobiles) and backhaul links (macro-to relays) use the air interface and share the same frequency bandwidth. Resources are time multiplexed, and the portion of time allocated to each type of link is fixed. This contribution aims at improving the network capacity by introducing a self-optimizing (SON) NEM that dynamically balances the resources shared between the backhaul and direct links.

5.2.2 Content of the work

This work adapts a learning technique, namely the Policy Gradient Reinforcement Learning (PGRL) for the SON algorithm used by the self-optimizing NEM. The system state *S* is defined by the number *S_i* of users in the *i*-th direct link or the backhaul link. The system can take one of two actions $a \in \{0,1\}$: when a=0 the station to relay backhaul link is activated and shared in a Processor Sharing (PS) manner, and when a=1 the direct station to user links are activated and shared in a PS manner. The PGRL uses a stochastic weighted policy denoted by $P_{S,\theta}$. θ is a vector of weighting coefficients defining the policy. The probability of choosing a=0 or 1 is given by

$$P_{S,\theta}(S,0) = 1 - f(\langle S,\theta\rangle)$$
$$P_{S,\theta}(S,1) = f(\langle S,\theta\rangle)$$

where $f = \dfrac{1}{1-e^{-x}}$. θ is found by minimizing an average cost $J_{S_0}(P_{S,\theta})$ calculated along a sample path using the PGRL formulation. The method is described in details in [24] and [25]. A reduction about 30 percent in file transfer time (when chosen as the reward) is achieved as can be seen in Figure 1. The convergence of 3 of the controlled parameters is shown in Figure 2. To assess the quality of the solution, a global optimum has been calculated using a global search method (the particle swarm optimization). The cumulative distribution function of the performance gap between the learning and the global optimization method has been calculated [25]. In the worst case, the gap is of 25%, and the median performance gap is 11%. Hence despite its local nature and relatively low computational complexity, the learning procedure performs quite well when compared to a global search.

5.2.3 Merit of the work

The PGRL algorithm has a linear complexity with respect to the number of parameters, and is therefore scalable. Performance results presented in D3.2 and have shown that the reward (chosen as the file transfer time or blocking rate) decreases almost monotonically, as expected from the PGRL when a finite number of iterations are used for the gradient calculation. Hence performance deterioration in the learning phase is avoided and the method can be used in an "always on" mode. The convergence of the method is shown theoretically [24][25] thus providing the necessary assurance for real implementation. Finally, the solution is a control plane solution (on-line) that can be governed via GOV UMF core mechanism: the operator can select the type of reward to achieve operational goals.

New contributions with respect to D3.2: Numerical experiments illustrating the convergence of the algorithm in practical network setting have been performed (see Figure 22). The PGRL is a local type of search method. A comparison of the optimum with respect to a global search method based on Particle Swarm Optimisation (PSO) has been carried out. Figure 23 depicts the cumulative distribution function (c.d.f) of the performance gap between the learning process and the global optimum for 100 runs with arbitrary initial conditions. In the worst case, the gap is of 25%, and the median performance gap is of 11%. Hence despite its local nature and relatively low computational complexity, the PGRL method performs well when compared to a global search.

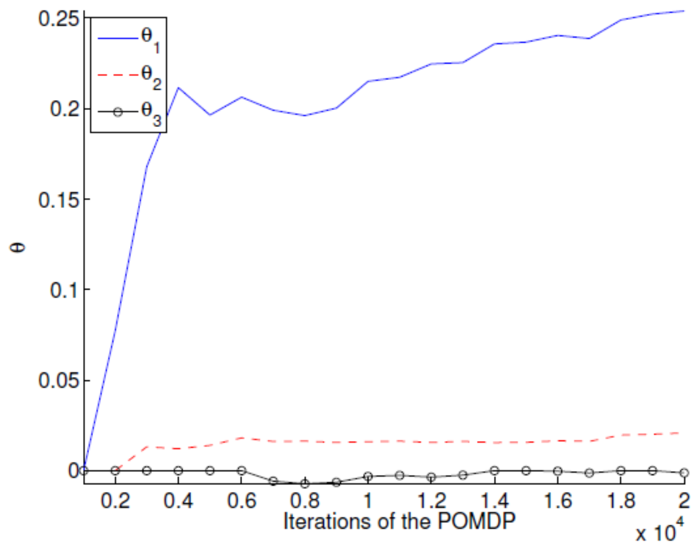


Figure 22: Controller parameters ($\theta_1, \theta_2, \theta_3$) during the learning process

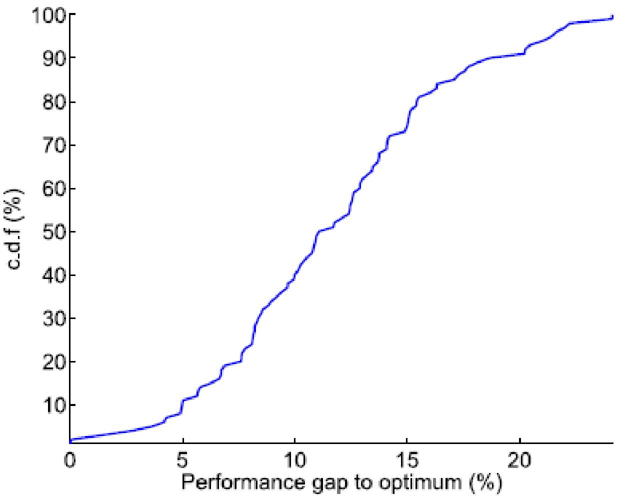


Figure 23: Comparison between local and global optimum

5.3 NEM 46: Model State Reduction for coverage optimisation

5.3.1 Context of the work

With the ever-increasing data traffic demand in today's mobile network in that 80 percent of the traffic being originated from indoors [26], data offloading through deployment of small cells has become a focus of the industry in the recent years. Since deployment of such cells are rather organic and ad-hoc, intelligent self-optimisation and self-configuration mechanisms are necessary to assure improved performance whilst keeping the OPEX at minimum. Considering dense deployment of such cells, coverage optimisation is particularly an important aspect that can directly impact the serving users' quality of experience. Since coverage optimisation is rather a complex problem (especially for dense deployments), as a part of its contribution, this work propose an improved learning algorithm that is based on combination of fuzzy logic and reinforcement learning. Subsequently, an Improved Reward Distribution (IRD) scheme is introduced in that the action space will be abstracted and the overall learning efficiency is enhanced.

5.3.2 Content of the work

The main advantage of combining fuzzy logic with the reinforcement learning is that the two frameworks can complement each other. However, as a result of combining the two frameworks a new issue arises. Due to the nature of the fuzzy systems, more than one states/actions are triggered and hence there is a need for a mechanism to distribute the reward between the contributing state-action pairs [27][28]. Our earlier work [27] proposed to modulate the learning rate of the reinforcement learning by considering both the input states and the triggered actions. The efficiency of the reward distribution and hence the overall learning process can be further improved by reducing and abstracting the action space. Figure 24(left) shows a simple example of the triggered output actions. The final executed action is the crisp output after defuzzification stage (that is zero/no change herein).

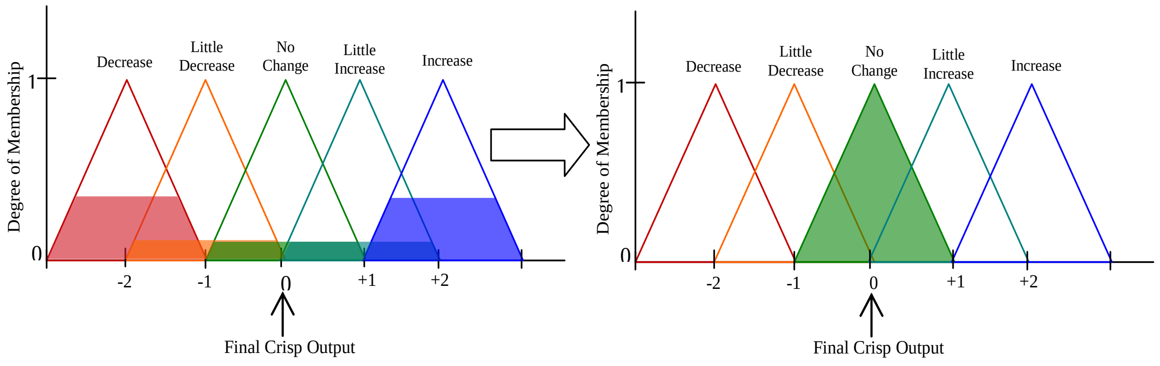


Figure 24 : Left) Example of triggered actions and Right) Equivalent abstracted triggered fuzzy action

Traditional fuzzy-reinforcement learning schemes credit all contributing actions and in fact the two extreme actions in the above example (Decrease and Increase) are credited most due to their high firing strength. Since the actual output is uncorrelated to those actions, this will be misleading and results in poor learning and delayed convergence. Figure 24 (right) shows the equivalent abstracted action where the reinforcement learning reward ought to be applied. However, since defuzzification is a non-linear process, finding equivalent action(s) is not very straightforward. Figure 25 shows the functional diagram of the fuzzy reinforcement learning framework with the proposed IRD scheme in that the equivalent action(s) are determined by using an additional Virtual Fuzzy Layer (VFL). VFL is a fuzzy layer that is used to determine the output-input map of the defuzzification stage. This can be realized through a simple Single Input, Single Output (SISO) fuzzy system that uses the same fuzzy membership functions of the action space. The defuzzification method will be also the same. The mapping can be extracted offline and then be implemented within the learning agent(s).

The coverage Optimisation problem considers dense deployment of small cells in that every base-station tries to configure its pilot power such that the overall users' Signal to Interference plus Noise Ratio (SINR) and subsequently their allocated throughput is maximized. Optimisation of throughput additionally involved load balancing between the base-stations in an implicit way.

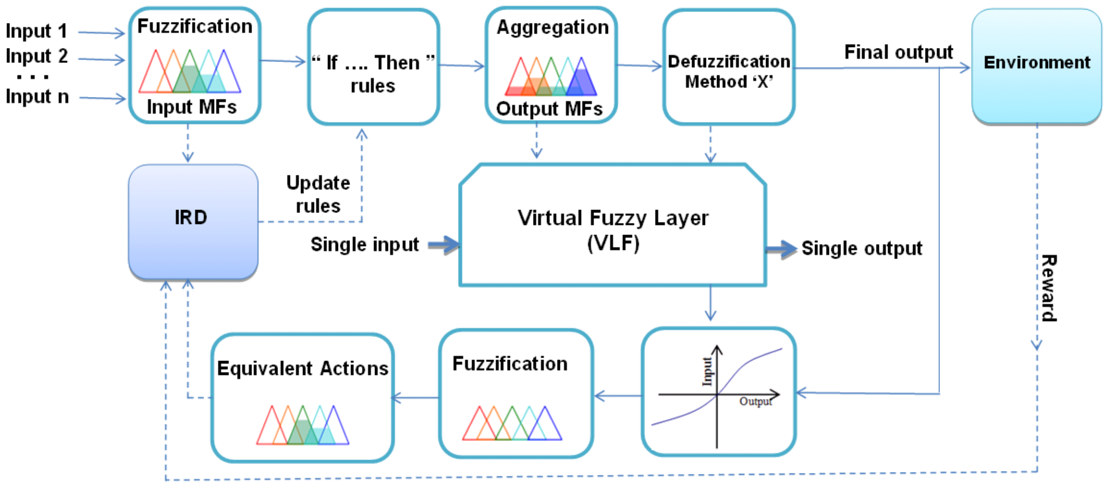


Figure 25: Functional diagram of fuzzy reinforcement learning framework with the IRD scheme

The fitness function, *F*, consists of the average users' throughput, T_m and the throughput allocated to the worst 5-percentile of users, T_w, i.e. $F = T_m + \beta\, T_w$ where *β* is a constant coefficient (set to 2 in this study).Higher *β* values account more for the worst users. Considering a dense deployment, base-stations exchange the average SINR of their users with each other through their X2 backhaul interface. This is the only message passing requirement of the algorithm. The rest of the procedures are fully distributed. The inputs of the algorithm are the difference between the average of the base-station users' throughput and the overall average throughput considering the neighbouring cells' reports, and the current pilot power of the base station. The latter is especially important since allocating more power to the pilot results in increased coverage which in turn commits the base station to serve more users. Not only this, but in addition, such increase results in less power budget left for data channels and subsequently impact the users' SINR and throughput. Both inputs are normalised and are fuzzified using three fuzzy membership functions. This results in a total of 9 input states in that individual inputs are combined using the "AND" operation. The action space consists of three membership functions and modifies the current pilot power of the base-station. The reward function is the change of the fitness function as a consequence of executed actions.
The simulation scenario refers to a large hot-spot (200mx200m) in that 10 small cell base-stations with the maximum power of 1Watt are randomly deployed. Using the proposed fuzzy reinforcement learning with the IRD scheme, each base-station tries to adjust its pilot power to maximise the fitness function. Simulation parameters including the path loss and shadow fading models are taken from [29]. Figure 26 (a) illustrates the fitness function against the number of iterations per each base station. The results are averaged over 100 independent runs and show how the fitness is improved as the algorithm progresses. Compared to the starting stage in that all base stations have similar pilot power ratio (10% of total power), the algorithm always improve the fitness function even during the learning phase where some new state-action pairs are explored. Additionally, the algorithm's performance and the convergence time are considerably improved compared to the case when IRD is not applied.

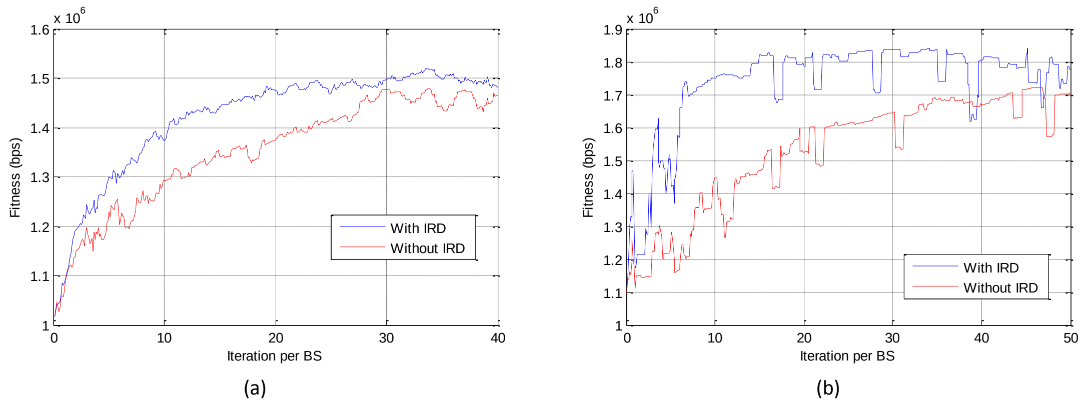


Figure 26: a) Fitness vs. no of iterations averaged over 100 runs b) variations of fitness for single simulation run

To get a better understanding of the fitness variations and the impact of explorations, Figure 26 (b) shows the variation of the fitness function for one simulation run. In this figure, the frequent fitness drops are due to the regular explorations. Such explorations enable the algorithm to adapt against the changes of the environment and provide functionalities such as self-healing. Figure 27 compares the SINR map of the simulation scenario from the initial stage (left) and after 10 iterations per base-stations (right) for one sample run.

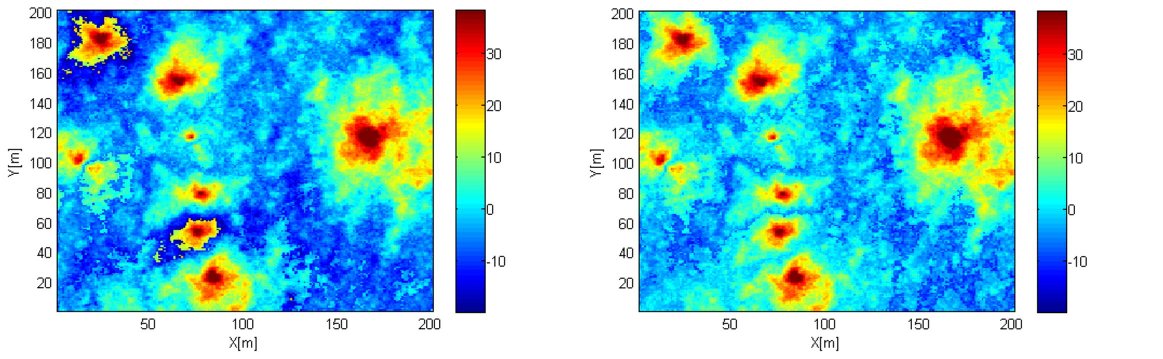


Figure 27: Example of initial SINR map expressed in dB (left) and SINR map in dB after 10 algorithm iterations (right)

5.3.3 Merit of the work

This study proposes a novel improved reward distribution algorithm for fuzzy reinforcement learning systems that results in enhanced learning and faster convergence of the algorithms. The implementation is also rather simple making it possible to compute the de-fuzzification mapping offline and to apply it to the learning agents as a simple look up table. This is especially useful for a complex and distributed optimisation task such as the coverage optimisation problem investigated here.

With respect to D3.2, the IRD mechanism is introduced and the performance is evaluated by defining a fitness function that considers both average and the worst case users' throughput. The results are compared with the case when IRD is not in place. The algorithm still benefits from last year proposal in that the learning rate is modulated both by input states and triggered actions.

We chose the fuzzy reinforcement learning framework because it is a very suitable method for Optimisation tasks that are highly complex and requires individual agents to learn through interactions with the environment. Unlike supervised learning schemes, reinforcement learning is appropriate for the cases when the correct answer is unknown and learning ought to be accomplished through indirect feedback. The main advantage of combining fuzzy logic with the reinforcement learning is that the two frameworks can complement each other. Reinforcement learning suffers from the curse of dimensionality problem in that the number of input states and actions can be increased dramatically leading to poor learning and delayed convergence. Fuzzy logic systems, on the other hand, can provide a high level of input states and action space abstraction but the fuzzy inference systems (including the If-Then rules) are rather static and required to be defined with the help of the experts' knowledge. Reinforcement learning can address this issue through indirect learning of the fuzzy rules.

5.4 Discussion

5.4.1 General

SON functionalities can be implemented in the management (e.g. Operation and Maintenance Centre) or the control plane (e.g. within the base stations: macro, micro, relay, femto, pico, or small cell). The advantage of implementing a self-optimizing feature in the control plane is the higher reactivity of the SON functionality and the better performance gain it can achieve. In the first self-optimizing features in LTE networks, operators are expected to have limited control on the SON functionalities (selecting certain parameters to be tuned or set their range). Due to the limited control on "control plane SON", operators may have a preference to "management plane SON" solutions.

UMF provides a much higher flexibility in embedding, governing and coordinating the operation of NEMs implementing the SON functionalities. For example, via GOV core mechanism, the operator can define parameters and their corresponding range, but also other elements related to the NEM operation such as the utility guiding the SON operation. In the case of load balancing between backhaul and direct (station to mobile) links, different utilities guiding the learning process could be introduced via GOV, namely file transfer time and blocking rate. In the case of CCO self-optimizing NEM, the learning objectives could be incorporated via GOV, to achieve specific optimization goals, such as maximizing users SINR, and minimizing interference for neighbouring femtocells. The ability to govern SON functionalities provides much more flexibility in managing the network according to the operator objectives. It will also favour control plane SON solutions which can achieve better performance.

It is noted that UMF provides as well a framework for operating several NEMs, thus ensuring the coordinated and stable operation of the NEMs. The self-backhauling NEM (or load balancing NEM) has been jointly triggered and coordinated with a CCO NEM using the COORD core mechanisms (within UC4 demonstration).

5.4.2 Specific Example

Here we substantiate our general explanations using interactions of NEMs 11/26 and NEM 12.

Workflow of a pathological scenario

Abbreviation: Resource Sharing Self-Backhauling (RSSB)

UMF brings about two advantages in the management of self-optimizing functionalities, and their corresponding NEMs:

(i) A framework for governing the NEMs. Via GOV core mechanism, the operator could introduce a new objective for the NEM which guides the self-optimization process according to the operator objective.

    <u>The corresponding pathological scenario</u>: the operator wants to change the utility (reinforcement) used by the RSSB NEM, and is obliged to purchase and install the new SON software feature.

(ii) A framework for coordinating NEMs. A conflict is identified following the installation / triggering of a new NEM. COORD core mechanism, via the joint optimization (JO), “takes over” and perform the joint (self-) optimization via a suitable joint (e.g. aggregated) utility function.

    <u>The corresponding pathological scenario</u>: due to an identified conflict following the activation of several NEMs, the operator is obliged to turn off certain NEMs. The benefit from the purchased SON functionalities is lost.

We consider next just the governance point (point (1))

Workflow:

(i) Cell load prediction NEM (11/26) predicts a cell overload. The NEM (11/26) sends a triggering message “overload prediction detected” to NEM RSSB (12).

(ii) NEM RSSB 12 sends an “Overload message acknowledged” to NEM (11/26)

(iii) NEM RSSB 12 sends a “Activation request” message to GOV

(iv) GOV sends NEM RSSB 12 “Policy_Provision” information for running the NEM, including specific information on utility and parameters to be used for the self-optimization process

(v) NEM RSSB 12 sends a “Policy installation” acknowledgement message to GOV

(vi) NEM RSSB 12 is activated, and sends an “RSSB active” to GOV. The load balancing between the backhaul and direct (station to mobile) links is triggered.

6 Use Case 6 Related NEMs

6.1 Introduction

A Network Operator (NO) wants to deploy new services and/or accommodate new traffic on top of its multi-vendor and multi-technology infrastructure involving both Radio Access Networks (RANs) and backhaul/core segments. The automation and achievement of coordinated, end-to-end performance in such a process is both desirable and challenging, especially considering the problems dealing with the different manifestations of heterogeneity in the considered underlying networks i.e. heterogeneity in the used technologies and domains, in the equipment coming from different vendors and of course in the management tools/systems.

Although there have been efforts towards integrating and automating such service deployment processes, there is a need for solutions enhancing the level of integration and automation towards the goal of exploiting an end-to-end approach.

This use case reports on the above problems and calls for solutions that will provide a unified, goal-based, autonomic management system for the service deployment and/or new traffic accommodation on top of heterogeneous networks encompassing both OFDM-based RANs and MPLS-based backhaul/core segments. As an example let us assume a Mobile Network Operator (MNO) that receives an urgent request concerning a real time, video-based application (e.g. video-streaming of a programmed event), an associated set of user classes for the application, and a set of QoS levels for each user class of that application. In addition, the request can designate a specific location e.g. a conference centre of Piraeus region and a specific time period e.g. evening from 16:00 to 18:00, where the application will be delivered to an also (roughly) given number of conference attendants [30].

For providing a unified autonomic management system for the service deployment in such a case different types of information need to be available either by directly monitoring data or by building knowledge upon monitored information. A number of mechanisms that are used towards this direction and an example of how UMF contributes in UC6 are being discussed in this section.

6.2 NEM 11: Load Level Estimation

6.2.1 Context of the work

In the context of the future dynamic service management, the system needs to be aware of the load that it will face when/ if a new service is applied/ added over the network and thus the existing network traffic. An example case where load level estimation mechanisms are invoked is in the candidate solution computation phase of UC6. The purpose of this NEM is to provide estimations to other mechanisms regarding traffic load levels for multiple RAN elements, in specific time periods. Such a goal is achieved through online, unsupervised machine learning schemes. Monitored data are fed into the NEM (by some monitoring NEM) in order to build a knowledge base (which consists of multiple Self-Organizing Maps) containing past experience on traffic load through time for the specific network element.

6.2.2 Content of the work

This NEM targets at estimating the load level of each RAN given the time (frame) that the Network Operator (NO) is interested in. Towards this direction, the functioning of the NEM is split in two phases. The first phase is responsible for building the respective knowledge and the second is related to the estimation of the load level per se.

Phase 1. During this phase the NEM builds knowledge on the traffic load that is/ can be observed at each base station with respect to time. Towards this direction, the unsupervised neural network known as Self-Organizing Maps (SOMs) is exploited. In particular, SOM uses historical data related to the load of the network with respect to time (i.e. parameters like *hour of day*, *day of week*, *week of year* etc) for each base station. Knowledge extracted from these historical data is depicted in a 2D rectangular grid of coloured units (the actual map), each representing a point in the input space, which, in turn, in this application, consists of a variety of time parameters for different granularities (hour, day, week). Similar units (i.e. units representing points with short Euclidean distances in the multidimensional input space) are placed closely on the map in order to form clusters. This process is known as Learning Vector Quantization (training of the map), and takes place once for each RAN (Figure 28). The load levels have been selected to be three, namely low (green), medium (yellow/ orange) and high (red).

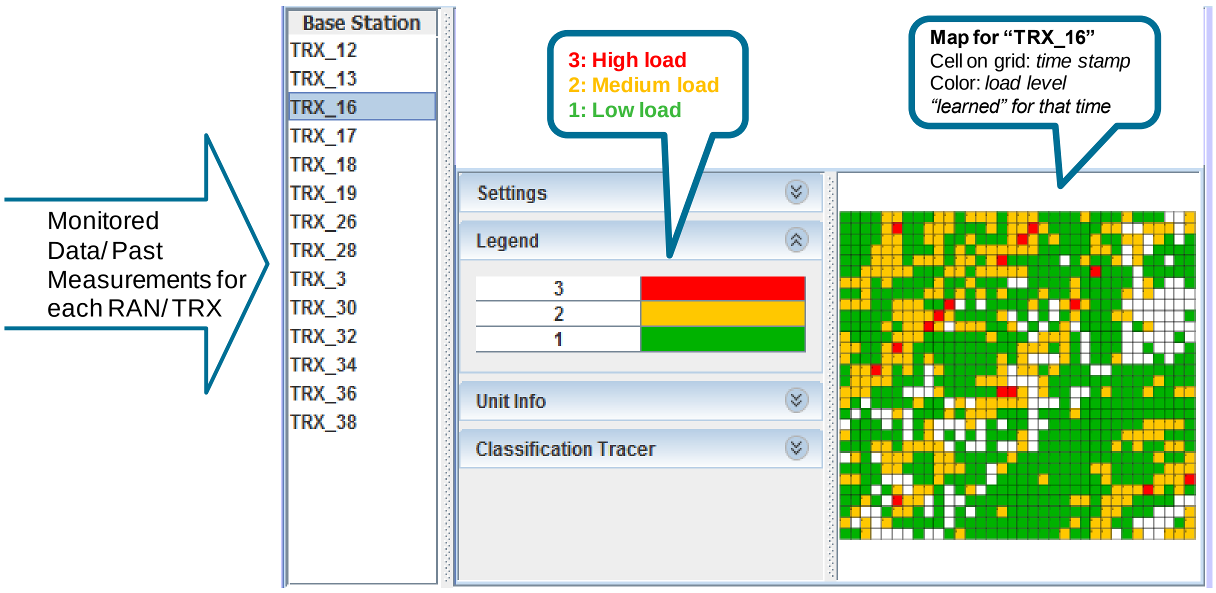


Figure 28: Self-Organizing Map of the 16th base station. Conclusions can be drawn by further examining the formed clusters, while there is no predefined interpretation of the x- or y-axis.

Phase 2. During this phase the NEM is capable to estimate the load level that will be encountered in a future moment/ time frame. For achieving this, the NEM exploits the knowledge that has been derived during the 1st phase. In particular, the NEM maps the data from the request on the SOM that corresponds to the specific RAN and suggests the load level that corresponds to this time frame. The estimation is performed as follows: the load level which responds to the request is the same with the load levels that corresponded to the rest data that were mapped on this cell (during the training/ 1st phase).

Results of the performance of the NEM in terms of correctness of load level estimations are also depicted in Figure 29. In particular, this figure compares the two diagrams that result from the estimation of the NEM during a day (red line) and the respective real measured load levels (blue line) that were finally encountered during the same day. The necessary data for these diagrams (and the training phase of the map, i.e. phase 1 of the NEM) were collected via a custom simulator.

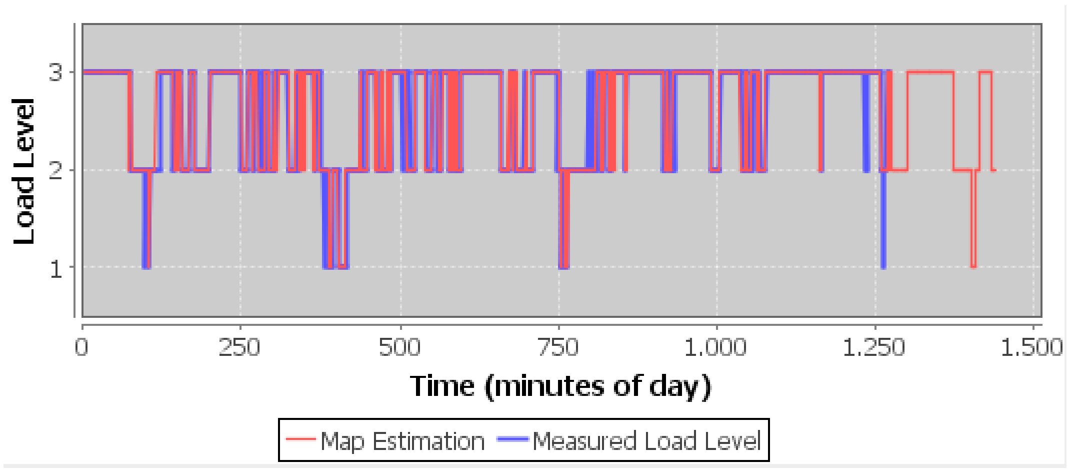
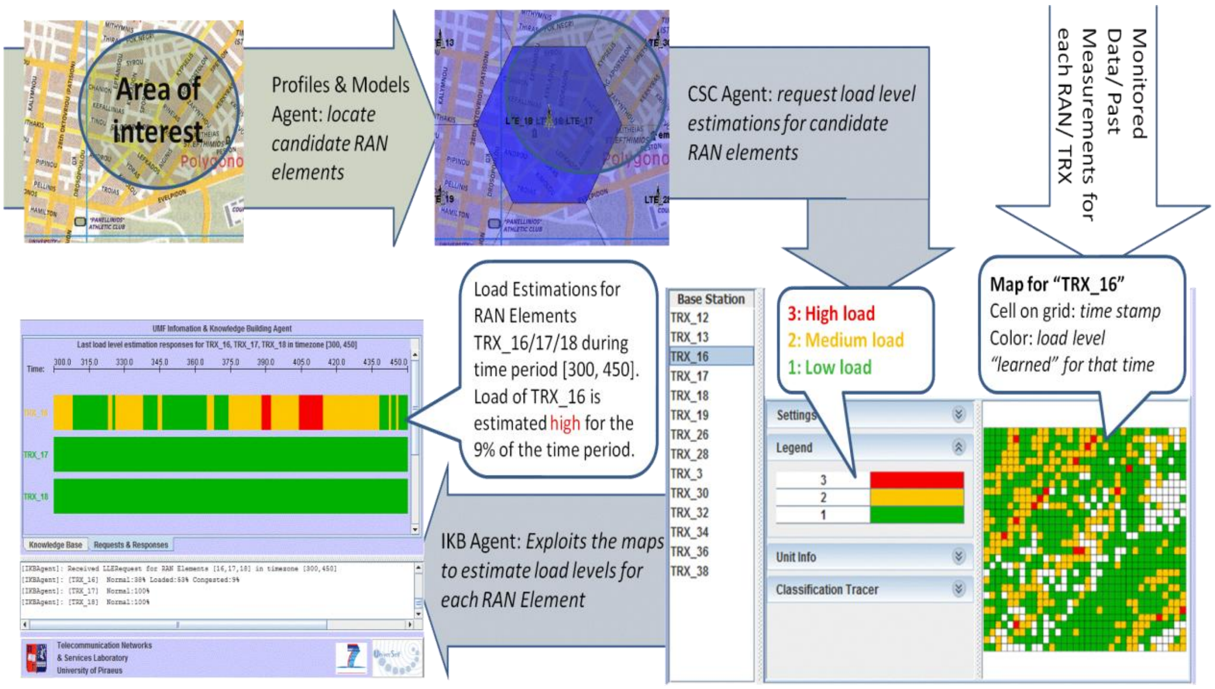


Figure 29: Comparative diagram of the real/ measured load level and the estimations provided by the NEM/SOM.

6.2.3 Merit of the work

This NEM, estimating the load levels that are encountered in the network for each RAN, contributes to efficiently moving to dynamic network planning systems. In particular, making the system aware of such information it enhances the network planning in terms of both speed and better decision making.


Area of interest
Polygono
Profiles & Models Agent: locate candidate RAN elements
CSC Agent: request load level estimations for candidate RAN elements
Monitored Data/ Past Measurements for each RAN/ TRX
Load Estimations for RAN Elements TRX_16/17/18 during time period [300, 450]. Load of TRX_16 is estimated high for the 9% of the time period.
3: High load
2: Medium load
1: Low load
Map for "TRX_16"
Cell on grid: time stamp
Color: load level "learned" for that time
IKB Agent: Exploits the maps to estimate load levels for each RAN Element
UMF Infomation & Knowledge Building Agent
Last load level estimation responses for TRX_16, TRX_17, TRX_18 in timezone [300, 450]
Time:
300.0
315.0
330.0
345.0
360.0
375.0
390.0
405.0
420.0
435.0
450.0
TRX_16
TRX_17
TRX_18
Knowledge Base
Requests & Responses
Telecommunication Networks & Services Laboratory University of Piraeus
Base Station
TRX_12
TRX_13
TRX_16
TRX_17
TRX_18
TRX_19
TRX_26
TRX_28
TRX_3
TRX_30
TRX_32
TRX_34
TRX_36
TRX_38
Settings
Legend
3
2
1
Unit Info
Classification Tracer

```
[IKBAgent]: Received LLERequest for RAN Elements [16,17,18] in timezone [300,450]
[IKBAgent]: [TRX_16]  Normal:38% Loaded:53% Congested:9%
[IKBAgent]: [TRX_17]  Normal:100%
[IKBAgent]: [TRX_18]  Normal:100%
```

Figure 30. Example use of Load Level Estimation NEM.

This capability of the mechanism, while not previously reported in D3.2, has been demonstrated in many events where UniverSelf (UC6) demonstration has participated. An example of how the NEM was used in these demonstrations can be found in Figure 30. During the demonstration a decision making mechanism, namely the Candidate Solution Computation (CSC), was asking for information about the load level of the RANs that could be used for adding a new service at a specific period of time. The LLE NEM was responding to CSC through the Information and Knowledge Building (IKB) which RAN(s) should be avoided for this time period according to the estimation of their load levels. Eventually, CSC was making its decision given this information.

6.3 NEM 26: Load prediction in a RAN based on previous data

6.3.1 Context of the work

In the framework of SON autonomic service management functions, the provisioning of a new requested service is considered. During the self-planning phase regarding the provisioning of the requested service, it is important to be able to estimate the load to which a certain portion of a Radio Access Network (RAN) will be subjected when the new service will be implemented. This knowledge is essential in the evaluation of the right RAN choice and QoS for the requested service. In this context, a reliable characterization of the traffic load could be useful to forecast the legacy traffic on a given RAN on top of which the load of the new service would have to be added.

6.3.2 Content of the work

Through a statistical analysis of the traffic process (e.g. generated traffic or throughput) based on the real time observation of the traffic in the area (cell or portion of the cell) an expected legacy load (with an associated probability) could be derived so that a suitable RAN can be selected and governance actions applied. In Figure 1 the first draft of the functional modules and their relations are depicted:

- Analysis Update

The statistical analysis is updated on the base of new data (measurements) collected since the last update. If the statistical estimation is still reliable (e.g. the goodness-of-fit -GoF- chi-square estimate[2] is under a given value) the existing model (probability distribution function, observation window) is kept and only the updated parameters are stored in the "Knowledge Base" (a1). If the existing model does not provide a sufficient reliability, a model update is needed and such a request is sent to the "Model Update" module (b1). The model changes proposed by the "Model Update" (b2) are stored and used for subsequent trend analysis (b3). If no reliable model is obtained, only simple data (e.g. mean and variance) are stored in the " Knowledge Base " to support a basic prediction (b3) and any other previous model is discarded.

- Model Update

  Based on the old and the new data, a statistical characterization (e.g. with the goodness-of-fit chi-square test) is performed by changing the data filtering (e.g. reduction/extension of the observation window, aggregation/separation in time/space of the collected data) and/or looking for a more appropriate probability distribution that fits the observed data. The new model is provided to the "Analysis Update" module (b2). If it is not possible to derive a model with a minimum reliability, this is notified to the "Analysis Update" module (b2)

- Legacy Load Prediction

  Upon request from an external NEM, the "Legacy Load Prediction" module uses the last updated model from the "Knowledge Base". If a comprehensive model is available (e.g. a PDF that fit the data), a thorough analysis will be performed (e.g. a cumulative) to derive the expected load. Simpler results (e.g. in terms of mean value and confidence interval) will be given when no model is available.

6.3.3 Merit of the work

This contribution is a piece of work that tries to statistically characterize the traffic process (e.g. load expressed in terms of throughput) based on the real time observation of the traffic in a localized area (cell or portion of the cell). In the on-going work it has been identified that once a good theoretical distribution is found for a set of data with low values of the goodness-of-fit chi-square test and whenever the characterization remains valid for the following days, the obtained distribution can be used as a good statistical estimation for the prediction of some days ahead. In Figure 32, there are some possible examples of evolution over time of the goodness-of-fit (determined by chi-square tests) of a good theoretical distribution determined at time T0 . *T(i,99)* and *T(i,95)* represent the maximum time period where the theoretical distribution can be used as reference with a goodness-of-fit probability of 99% or 95% respectively, for the data set *case i*. Based on the periodic observation and calculation of the goodness-of-fit, it should be possible to differentiate the sets in terms of persistence in time of the associated theoretical distribution and to use it for prediction purposes for a suitable period of time. The future work will then concentrate on further analysis of this evolution in order to obtain a reliable estimate of the prediction.

[2] The lower is the value the better is the GoF.

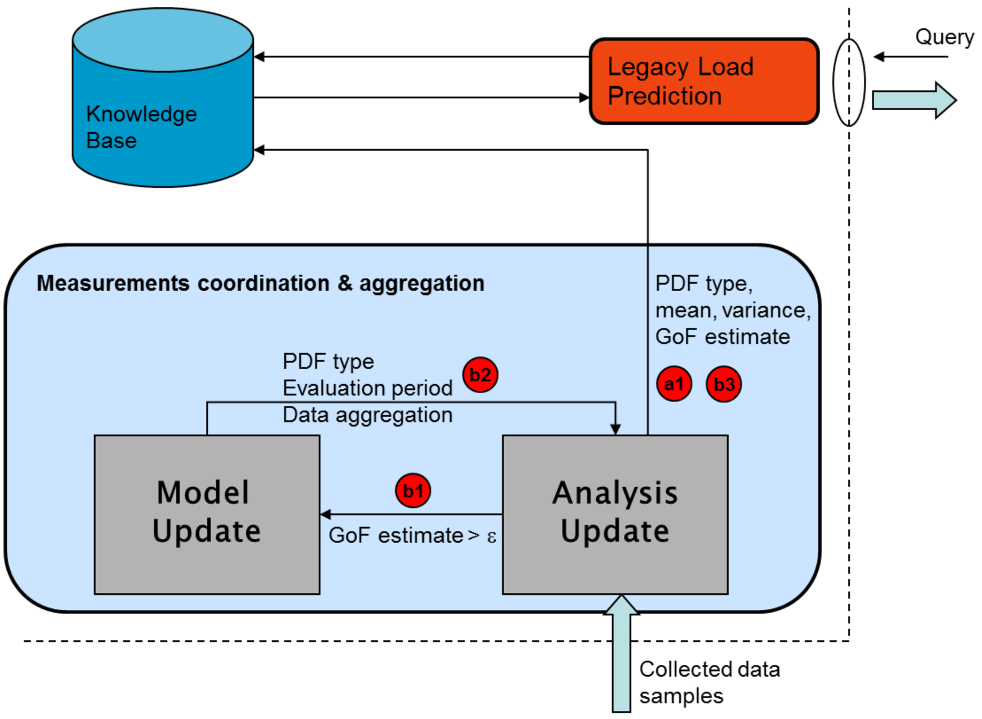


Figure 31: Draft definition of a method for defining and updating a traffic prediction model based on collected measurement data

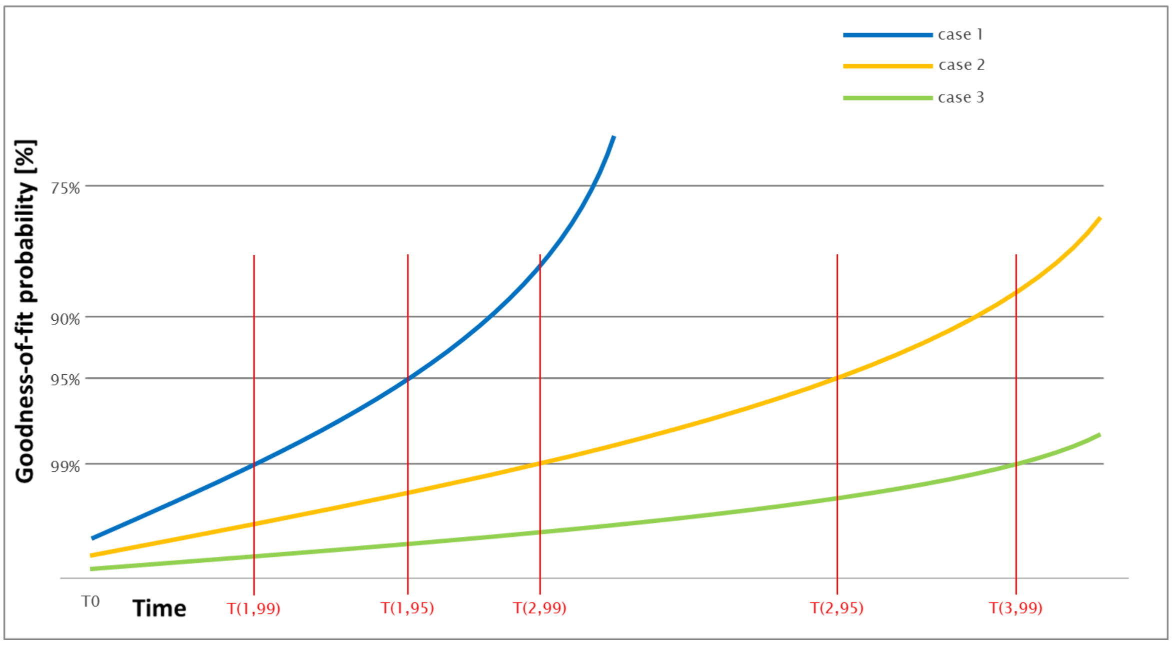


Figure 32: Evolution examples of goodness-of-fit when a good theoretical distribution is determined at T0

6.4 NEM 39: PCN-based admission control

6.4.1 Context of the work

The envisioned dynamicity of future Internet networks, where applications with different service requirements may appear makes Quality of Service (QoS) provisioning and service continuity a challenging issue that traditional traffic engineering approaches, usually based on offline optimizations through bandwidth provisioning, may not be able to address efficiently. Towards this end, dynamic service management functions such as admission control can play a significant role with respect to supporting QoS for application flows during the actual service delivery time, helping to overcome the inability of slow-changing network configurations to react adequately fast to shorter-term load fluctuations.

Even though admission control is a well-studied subject [31][32], most of the existing schemes suffer from the fact that they are based on some very rigid assumptions about the per-flow and aggregate underlying traffic models, requiring therefore manual reconfiguration of their parameters in a “trial and error” fashion as soon as the original assumptions stop being valid, in order to keep performing well [33]. That is they employ some tuning parameters that need to be initially manually set and also readjusted as soon as the traffic and network characteristics change. In this work we propose a novel, autonomic admission control scheme based on the increasingly popular Pre-Congestion Notification (PCN) framework put forward by IETF [34]. The proposed solution adapts autonomically to the characteristics of the traffic flows and underlying network traffic and can perform well under a variety of traffic and load conditions without making any assumptions about traffic models, flow dynamics and characteristics and with no need for manual and frequent reconfigurations.

6.4.2 Content of the work

The main concept of PCN is that packet markings when the traffic at each link exceeds a certain threshold can provide an early indication of congestion, giving the opportunity to prevent the actual congestion as it builds up by rejecting new flow requests as long as a certain amount of marked packets is observed. Through the use of fuzzy Q-learning (FQL) in our approach we are able starting from some default marking threshold value, to converge to a set of rules that drive the marking behaviour and threshold value readjustment, autonomically and on a *per scenario* basis as the traffic and network characteristics require.

Our scheme has been evaluated under a variety of traffic conditions incurred through differing flow arrival rates and flow durations of three types of MPEG-4 video flows with different bandwidth requirements. Figure 33 shows the evolution of the packet loss rate (PLR), the marking threshold and the utilization at the bottleneck link (100Mbps) for our scheme (FQLPCN) under one of the tested scenarios.

FQL PCN is able, after the initial period of PLR due to the high initial marking threshold, to reduce and vary the marking threshold so that PLR is equal to zero for most of the time and the utilization stays at high levels. During the initial period of PLR spikes (first 660sec on average among all simulation runs), PLR stays well below 0.5% whereas for the rest of the simulation runs the PLR spikes account in total for 12sec (on average among all runs) with a maximum PLR value of less than 10^{-4} (it is worth noting that for MPEG-4 video this latter value of PLR can be considered tolerable).

To illustrate the effect of learning in the rules used by the controller, in Figure 34 we show the incurred PLR and the variation of the marking threshold rate for a fuzzy logic (FL PCN) controller that uses the same initial rules fed to the FQL controller, without though any further updates to the rules through learning.

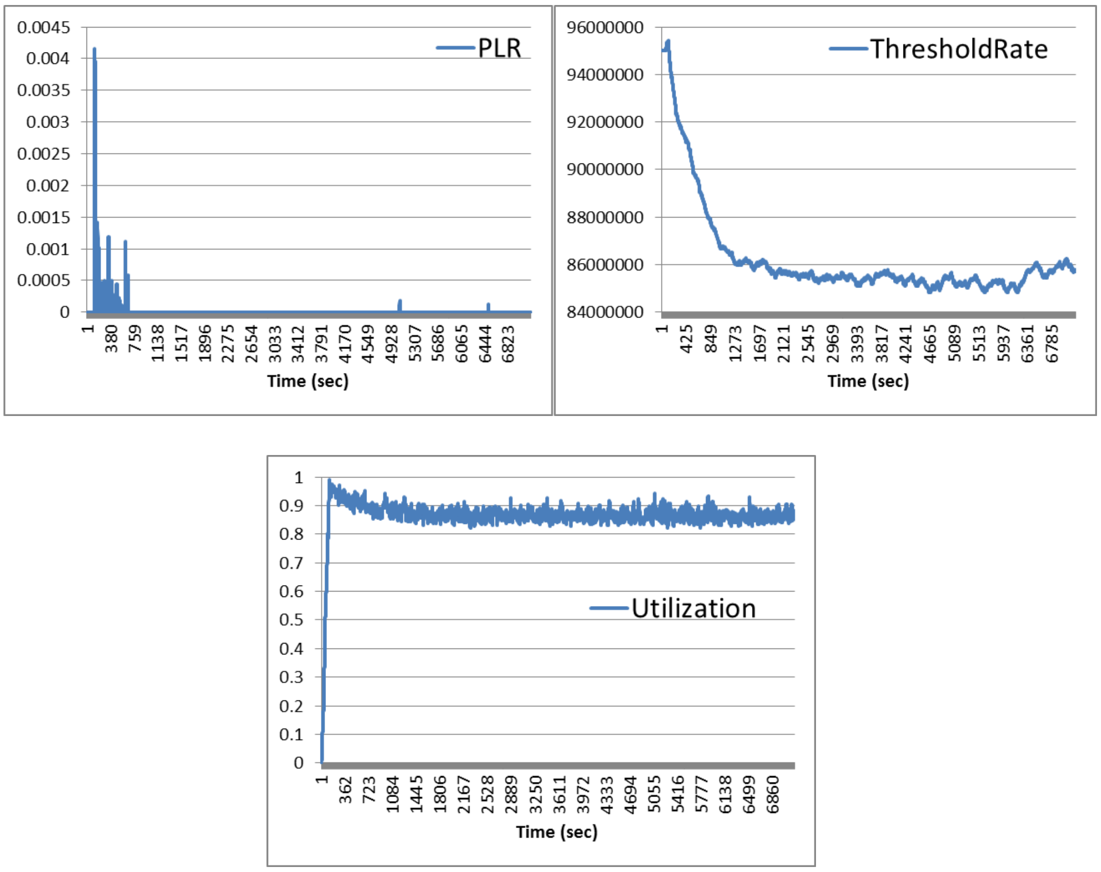


Figure 33 : Incurred PLR, marking threshold rate and utilization for the FQL PCN scheme.

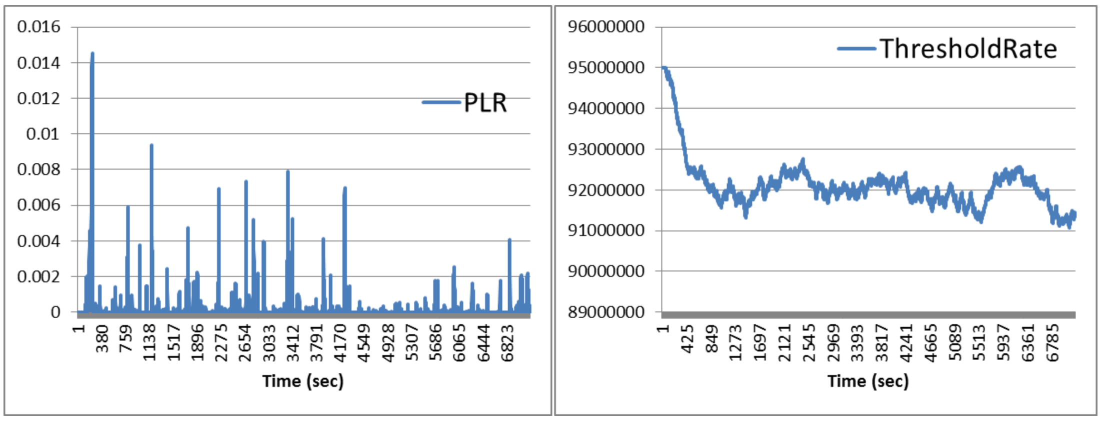


Figure 34: Incurred PLR and marking threshold rate for the FL PCN scheme

As Figure 34 shows for the FL PCN scheme there exist PLR spikes for the whole duration of the simulations due to the fact that the initially set rules are not optimized and are not adapted towards an optimized behaviour. In contrast the FQL PCN scheme, starting by exactly the same -apparently not optimized rule-set- is able through the learning process to redefine the rules accordingly.

6.4.3 Merit of the work

This contribution shows that it is possible through the use of Fuzzy Q-learning to create a PCN-based admission control scheme that is able to autonomically adapt its marking behaviour so that it meets the traffic and network requirements. The scheme requires no *a priori* knowledge about traffic flow or network characteristics, traffic models and flow dynamics and is able to adapt accordingly and on a per scenario basis without requiring any manual reconfiguration of its involved parameters on a per scenario basis.

6.5 Discussion

In some cases a problem could be solved using different methodologies. Generally speaking, using different methodologies could imply having different:

- Inputs
- Time for converging or coming to a solution
- Outputs

In this case study, the load prediction of the legacy traffic can be performed by looking at different network sources (inputs) and by using different time frames and methods/algorithms in the analysis, therefore resulting in different outputs. Specifically, both NEM 11 - "Load Level Estimation" and NEM 26 – "Load prediction in a RAN based on previous data" will forecast the legacy traffic of a RAN portion based on past observation. However, NEM 11 will indicate a load level in scale of three (low, medium, high) instead of the likely value (associated with a probability) that NEM 26 will indicate.

6.5.1 Without UMF

In general, without UMF, this functionality could be likely embedded in an overall network monitoring and management system without the possibility to choose between different methods and algorithms.

In the hypothesis of a possible choice, the network operator would have to choose one of the two methods up-front, and the switch between them would be fairly complicated as the input and output would have to be adapted to the control flow of the process.

6.5.2 With UMF

On the contrary, Universal Management Framework (UMF) enables the management system to choose between different approaches of the same problem, using the most appropriate NEM given the goals and the situation of the network at that time, thus adding flexibility and quality to the process. Towards this direction, a NEM that requests information addresses its request to the Information Collection and Dissemination (ICD) function of KNOW UMF core block. The latter retrieves the required information according to the operator policies, the needs of the other NEMs and the reliability of the NEMs by the Information Storage and Indexing (ISI) function of the KNOW in the correct time frame and/or granularity and responds. In case the information is not stored in the KNOW UMF core block at the desired granularity, then Information Aggregation (IA) operation is involved to produce it. On the other hand, if the required information is not stored at all (at any granularity), ICD asks the most appropriate NEM to produce it, according to the request, the operator policies, the needs of the other NEMs and the reliability of the offered service (Load Prediction in this case).

7 Use Case 7 Related NEMs

7.1 Introduction

Telco operators have the need to adapt their operations in order to reduce the time to market and the network maintenance costs, while at the same time increasing the customer satisfaction. There is an agreement in the research community that autonomic networks with self-configuration, self-diagnosis and self-healing capabilities will help in the automation of the provisioning and runtime phases, maintaining the quality of the services committed to the customer with minimal human intervention. UC7 focuses on mechanisms that address the gap between high-level specification of client performance objectives and existing resource management infrastructures. Such mechanisms should provide operators with means for decision oriented operational tasks based on the use of policies rather than low level command execution, thus should decrease the human intervention required for deploying new services, configuring and operating the network.

The use case will demonstrate the feasibility of a policy-based network management approach, both on fixed and mobile segments. The mobile network is based on a wifi connection on DSL network, while for the fixed environment Fiber-To-The-Home (FTTH) is the selected technology. In both cases, UC7 aims to develop self-* functionalities (e.g. self-monitoring, self-diagnosis and self-healing) which will enable the early detection and resolution of network and service problems.

In order to fulfil its mission, a set of NEMs have been designed, implemented and tested in UC7. On one hand, load balancing and self-healing cell outage NEMs act on the wireless segment, and are described in depth in Deliverable D3.5 [35]. On the other hand, monitoring and diagnosis NEMs have been developed for the FTTH segment. Monitoring NEMs interact with the network elements to gather data about their status and behaviour. As such, they do not embody any specific algorithm, and therefore will not be further described here. Diagnosis NEM implements a Bayesian diagnosis approach able to infer the most probable root cause of failure, and will be the subject of the rest of the chapter. All the mentioned NEMs can be governed via GOV core mechanism, and exchange information through the KNOW block.

7.2 NEM 6: Probabilistic self-monitoring and diagnosis in FTTH environments

7.2.1 Context of the work

Fault diagnosis is one of the areas in the network management field more sensitive to uncertainty [36]. Based on remote access to testing and information capabilities, Operation Support Systems try to come up with a conclusion but, in case the information gathered is incomplete or inaccurate, the process may reach wrong conclusions. Usually, the diagnosis of affected services is quite complicated, involving dozens of elements of heterogeneous technologies, and eventually requiring human intervention if the automated systems cannot find a conclusive diagnosis. As human expertise is a scarce and expensive resource, the research community tries to build systems that emulate network engineers working under uncertainty. Diagnosis automation supported by artificial intelligent techniques can help to reduce operation expenses, reducing human intervention. In the case of fault management, automation means to be able to detect, diagnose and repair possible problems in the system.

Additionally, FTTH rollout is today one of the main drivers of telecom business transformation, encompassing high investments in equipment and systems. These devices provide useful information, of heterogeneous nature and format, through different interfaces and using a variety of protocols, information that needs to be evaluated in order to find the root cause of a failure. Our approach aims to research a service diagnosis solution for FTTH network segments, which will enable the early detection and resolution of network, QoS, and QoE problems with limited or no customer impact.

7.2.2 Content of the work

One significant contribution of our solution to network management scenarios is its capacity to deal with uncertainty, since it is usual to face problems getting information and executing tests from different management systems. The probabilistic approach provides an answer even when not all relevant information is known, although the higher number of evidences are known, the more accurate the diagnosis will be.

Bayesian Networks (BN), a term coined by Judea Pearl [37], are based on probability theory. The problem domain is represented as a directed acyclic graph where the nodes represent variables, and the arcs, conditional dependencies between them. In our approach for FTTH diagnosis, the possible causes of service failures (hypothesis) detected in the network and the set of observable network variables (symptoms) are modelled as a BN. In this BN possible causes are defined as hypothesis nodes while symptoms as evidence nodes. Thanks to Bayesian inference, the probability of each of the possible causes of a service failure can be estimated. Thus, the result of the diagnosis procedure is a set of the most probable causes of the failure together with their probability values.

This algorithm is an iterative process where new evidences are added to the BN as they become available and Bayesian inference is repeatedly performed until enough confidence for a given hypothesis is reached or no extra information can be obtained.

The diagnosis process starts by creating an instance of the BN for the FTTH environment. Then, when the interruption in the service is detected, the diagnosis NEM gathers all known related evidences from the monitoring NEMs (detected failure plus other evidences used in previous diagnosis procedures that are still valid according to a time threshold). The selection of tests takes into account the cost (namely, use of resources and time needed for gathering the measurements) of each. Finally, it starts an iterative algorithm that encompasses the following actions:

1. Inserting into the BN all known evidences.
2. Performing Bayesian inference.
3. Evaluating the diagnosis result: if a hypothesis for the cause of the failure reaches a certain confidence threshold, the diagnosis stops. Otherwise, go to 4.
4. Selecting the best next test to perform, taking into account its cost and if there is enough data to request it. If no suggestion for a test is found (due to the fact that all possible tests have already been executed), the diagnosis stops. Otherwise, request the selected action to the appropriate agent. When it gets the results, go to 1.

Note that this algorithm continues until either the expected confidence is reached or there are no additional tests available.
Figure 35 presents a partial view of the Bayesian Network.

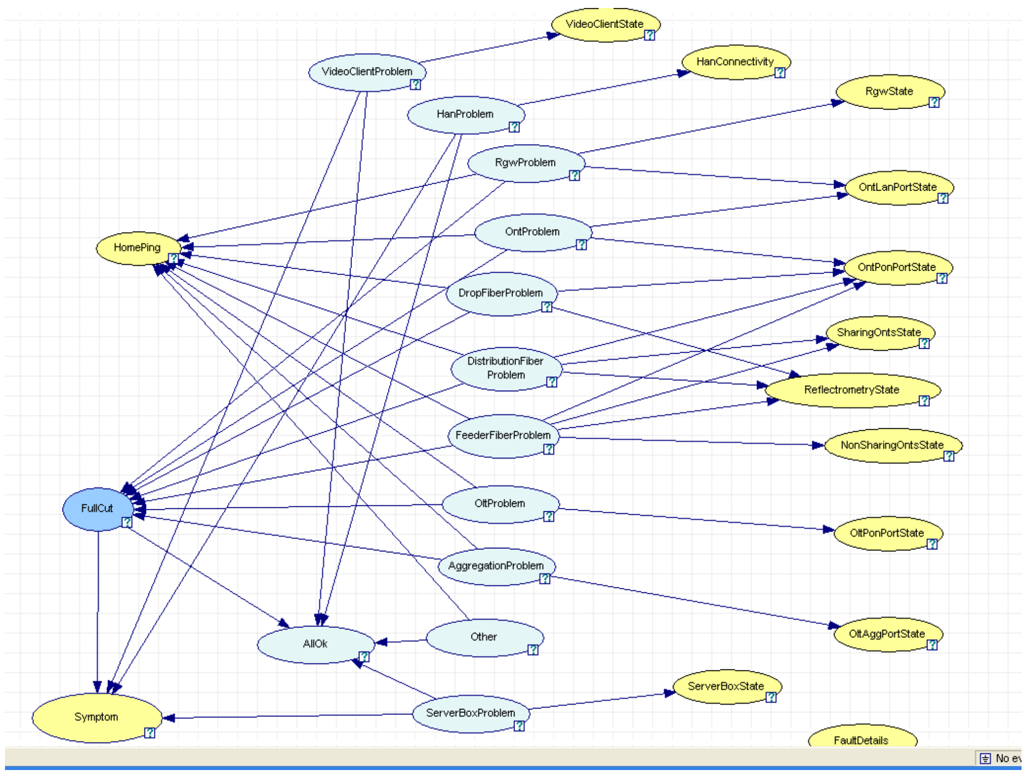


Figure 35: Partial view of the Bayesian Network for FTTH diagnosis

The algorithm described above has been implemented using JADE [38] multi-agent platform, where each agent embeds the behaviour of a NEM (monitoring or diagnosis). JADE is a complete and sound multi-agent Java platform based on FIPA standards [39] led by Telecom Italia. JADE allows the distributed deployment of Java coded agents with minimal requirements of CPU and memory on servers and devices. Following JADE principles [40], the agents of the presented solution are implemented using behaviours and all the communications between JADE agents are based on FIPA-ACL messages.

7.2.3 Merit of the work

The added-value of our NEM is essentially the accuracy obtained to diagnose failures in FTTH networks. We have tested the diagnosis NEM using the FTTH testbed located in a Central Office in Madrid city centre. Monitored data collected by the monitoring NEMs comprises the status of the different ports of the ONTs and OLT, network alarms, end to end connectivity, status of the ONTs sharing the splitters and reflectometric events. Tests were performed causing failures in the different network elements: router connectivity problems, ONT connectivity problems, fiber cuts in the distribution segment, fiber cut in drop segment, fiber cut in feeder segment, or failures in the video server, and then evaluated if the NEM reached valid conclusions. After running more than 1000 tests, the most probable cause of failure resulted to be the exact cause of failure in a 98.9% of the test cases, always with an associated probability greater than 85%. The recall resulted to be of 100% (of all the cases found as faulty, all of them indeed corresponded to a failure).

7.3 Discussion

UMF provides a flexible way to deploy and govern the monitoring and diagnosis NEMs. Via GOV core mechanism, the operator can easily order a deployment of new NEMs, or to configure already deployed NEMs, adjusting the parameters that define their behaviour. These features provide more flexibility to manage the network according to the goals set by the operator. This is an important difference with respect to the current functioning of the OSS, where usually the deployment or configuration of functionalities needs the involvement of the software experts, therefore impacting the cost and time needed to implement the changes.

Furthermore, UMF enables knowledge sharing between NEMs. UMF provides a seamlessly way to independently deploy monitoring and diagnosis NEMs, and allow them to use the infrastructure provided by the KNOW core block to exchange information. This decouples the development and implementation of different types of NEMs, increasing the flexibility of the whole system.

8 Conclusion

In this deliverable we have worked towards both the improvement of the observation and action methods, and also how these methods could benefit from integration through the UMF when deployed together.

In the chapters focusing on the techniques, the different techniques that were presented showed the merits of the techniques developed when applied to their respective application areas. The additions and refinements made over previous work were detailed, and the performance of the techniques was analyzed quantitatively, and their gains over equivalent benchmarks are shown. The presented techniques cover a significant spectrum of observation related domains (discovery, aggregation, diagnosis, mining, and (self-) modelling) and methodologies (fuzzy logic, self-organizing maps, reinforcement learning, case-based reasoning, Bayesian networks and statistical analysis).

In addition to presenting the individual NEMs, we put forward how the different techniques, when put together in an autonomous network, can benefit from but also require the UMF. Since the NEMs involved in T3.3 span various application environments, NEM interaction was shown and exemplified in the context of the Use Cases defined in WP4.

As the work progresses towards integration of techniques, we will now focus on the specifics of how the coordination (which is described at a high level here) can be formulated and implemented in detail, tying in the techniques developed in WP3, with the UMF developed in WP2 to deliver steps closer to a comprehensive, end to end autonomous network.

9 References

[1] UNIVERSELF - D3.2 - Identification of suitable classes of methods for learning and operations

[2] L.Bennacer, L.Ciavaglia, A.Chibani, Y.Amirat and M.Abdelhamid, "Optimization of fault diagnosis based on the combination of Bayesian Networks and Case Based Reasoning", accepted in proc IEEE/IFIP Network Operations and Management Symposium (NOMS 2012), 16 April, Maui, Hawaii,USA

[3] A. Sperotto; Mandjes M.; Sadre R.; P. T. de Boer & A. Pras. Autonomic Parameter Tuning of Anomaly-Based IDSs: an SSH Case Study. IEEE Transactions on Network and Service Management, 2012.

[4] A. Bantouna, K. Tsagkaris, V. Stavroulaki, G. Poulios, P. Demestichas, "Learning Techniques for Context Diagnosis and Prediction in Cognitive Communications", to be published in Cognitive Communications: Distributed Artificial Intelligence (DAI), Regulatory Policy & Economics, Implementation. H. Zhang and D. Grace, J. Wiley and Sons, August 2012

[5] M. Ghader, A. Bantouna, L. Bennacer, G. Calochira, B. Fuentes, G. Katsikas, Z. Yousaf, "On Accomplishing Context Awareness for autonomic network management", accepted at Future Network and Mobile Summit 2012, 4 - 6 July 2012, Berlin, Germany

[6] A. Bantouna, K. Tsagkaris, G. Poulios, A. Manzalini, P. Demestichas, "Knowledge in Support of Congestion Control Mechanisms", poster, Future Network Mobile Summit (FuNeMS) 2012, 4 - 6 July 2012, Berlin, Germany

[7] Oliver Johnson, "Information Theory And The Central Limit Theorem" , Imperial College, 2004, ISBN 1-86094-473-6

[8] Data Mining: A Preprocessing Engine - Luai Al Shalabi, Zyad Shaaban and Basel Kasasbeh Applied Science University, Amman, Jordan - Journal of Computer Science 2 (9): 735-739, 2006, ISSN 1549-3636.

[9] The Kernel Trick for Distances - Microsoft Research, TR MSR(2000-51), Redmond, WA.

[10] J. B. MacQueen (1967): "Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California Press, 1:281-297

[11] K-Means algorithm – Introduction, http://www.cs.cmu.edu/~dpelleg/kmeans.html

[12] Minacom, Service Level and Test Automation, "VoIP Service Quality Metrics and Thresholds", Reference Chart.

[13] UniverSelf Milestone 31, "Description of UC method and results of its algorithmic testing", May 2012: http://wiki.univerself-project.eu/topics/142-3048

[14] M. Barrère, R. Badonnel, and O. Festor. Supporting Vulnerability Awareness in Autonomic Networks and Systems with OVAL. Proceedings of the 7th IEEE International Conference on Network and Service Management (CNSM'11), October 2011.

[15] OVAL, Open Vulnerability and Assessment Language, Mitre Corp, http://oval.mitre.org/.

[16] Cfengine, Cfengine AS, http://www.cfengine.org/.

[17] M. Barrère, R. Badonnel, and O. Festor. Towards the Assessment of Distributed Vulnerabilities in Autonomic Networks and Systems. Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS'12), April 2012.

[18] A. Bikakis, G. Antoniou, "Defeasible Contexual Reasoning with Arguments in Ambient Intelligence", IEEE Transactions on Knowledge and Data Engineering, pp. 1492-1506, Vol. 22, No. 11, November 2010

[19] W. Wang, P. Barnaghi, "Semantic Annotation and Reasoning for Sensor Data", The 4th European Conference on Smart Sensing and Context (EuroSSC2009), Lecture Notes in Computer Science, Springer, September 2009

[20] DBpedia, http://dbpedia.org/About

[21] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/

[22] C.-K. Tham and T. C.-K. Hui, "Reinforcement Learning-based Dynamic Bandwidth Provisioning for Quality of Service in Differentiated Services Networks". Computer Communications, vol. 28, no. 15 (2005): 1741-1751.

[23] Shane Alcock and Richard Nelson. “Application flow control in YouTube video streams”. SIGCOMM Computer Communication Review, 41:24–30, April 2011

[24] R. Combes Z. Altman, E. Altman, “Self-organizing relays in LTE networks: queuing analysis and algorithms”, 7th International Conference on Network and Service Management, CNSM 2011, Paris France, Oct. 2011.

[25] R. Combes, Z. Altman and E. Altman, “Self-Organizing Relays: Dimensioning, Self-Optimization and Learning”, to be published in IEEE Transactions on Network Management, TNSM.

[26] M. Tolstrup, “Indoor Radio Planning: A Practical Guide for GSM, DCS, UMTS and HSPA”, Wiley Publications, 2008.

[27] Rouzbeh Razavi, Siegfried Klein, Holger Claussen “Self-Optimisation of capacity and coverage in LTE networks using a fuzzy reinforcement learning approach” IEEE Personal Indoor and Mobile Radio Communications (PIMRC) symposium 2010.

[28] Bonarini, A., “Delayed reinforcement, Fuzzy Q-learning and Fuzzy Logic Controllers”, Genetic Algorithms and Soft Computing, Physica Verlag (Springer Verlag), Heidelberg, Germany, pp: 447–466, 1996.

[29] 3GPP TR 36.814 “Evolved Universal Terrestrial Radio Access; Further advancements for E-UTRA physical layer aspects” v9.0 2010

[30] K.Tsagkaris, P.Demestichas , G.Nguengang, I.G.Ben Yahia, P.Peloso, Unveiling technical challenges for the governance of end-to-end service delivery and autonomic infrastructures to appear in Proc. IEEE Global Communications Conference (GLOBECOM 2012), Anaheim, California, USA, 3-7 Dec, 2012

[31] S. Wright “Admission Control In Multi-Service IP Networks: A Tutorial,” IEEE Communications Surveys & Tutorials, 2nd Quarter 2007.

[32] S. Lima, P. Carvalho and V. Freitas “Admission Control in Multiservice IP Networks: Architectural Issues and Trends,” IEEE Communications Magazine, April 2007.

[33] L. Breslau, S. Jamin and S. Shenker “Comments on the Performance of Measurement-Based Admission Control Algorithms”, IEEE INFOCOM 2000.

[34] P. Eardley (Editor) “Pre-Congestion Notification Architecture”, June 2009, Internet RFC 5559.

[35] Deliverable D3.5 “Adaptation and fine tuning of parameter optimization methods”

[36] A. Pras, J. Schönwälder, M. Burgess, O. Festor, G. M. Pérez, R. Stadler, B. Stiller: Key Research Challenges in Network Management. IEEE Communications Magazine, vol. 45, issue10, pp. 104-110 (2007)

[37] Pearl, J.: “Bayesian networks: A model of self-activated memory for evidential reasoning”, UCLA Report CSD-850017 (1985)

[38] JADE (Java Agent DEvelopment Framework). http://jade.tilab.com

[39] The Foundation of Intelligent Agents. [online] http://www.fipa.org/

[40] Bellifemine, F. L., Caire, G. and Greenwood, D.: “Developing Multi-Agent Systems with JADE”, John Wiley & Sons, Chichester, UK (2007)

[41] V. D. Blondel, A. Megretski (Eds), “Unsolved Problems in Mathematical Systems and Control Theory”, Princeton University Press, 2004

[42] FP. Kelly, “Charging and rate control for elastic traffic”. European Transactions on Telecommunications 1997; 8:33–37

[43] FP. Kelly, A. Maulloo, D. Tan, “Rate control in communication networks: shadow prices, proportional fairness and stability”. Journal of the Operational Research Society 1998; 49:237–252

10 Abbreviations

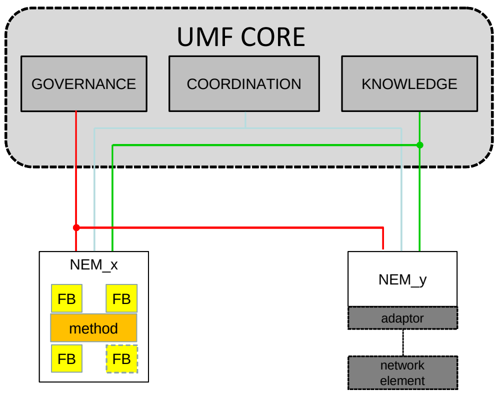
3GPP	3rd Generation Partnership Project
3GPP LTE	3GPP Long Term Evolution
3GPP SAE	3GPP Service Architecture Evolution
AFI	Autonomic network engineering for the self-managing Future Internet
AP	Access Point
API	Application Programming Interface
BoF	Birds-of-a-Feather
BSS	Business Support System
CAPEX	Capital Expenditures
DiffServ	Differentiated services
DoW	Description of Work
E2E	End-to-End
EMS	Element Management System
eNodeB	Evolved NodeB
ETSI	European Telecommunications Standards Institute
FG-FN	Focus Group – Future Networks
FMC	Fix Mobile Convergence
FTTH	Fibre To The Home
GUI	Graphical User Interface
GW	Gateway
H2N	Human-to-Network
ICT	Information and Communication Technologies
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IRTF	Internet Research Task Force
IMS	IP Multimedia Subsystem
IP	Internet Protocol
IRTF	Internet Research Task Force
IS	Information System
IT	Information Technology
ITU	International Telecommunication Union
ITU-T	International Telecommunication Union – Telecommunications standardization sector
KPI	Key Performance Indicator
LCCN	Learning-Capable Communication Networks
LE	Large Enterprises
LSP	Label Switched Path
LTE	Long Term Evolution
LTE-A	LTE Advanced
MPLS	Multi Protocol Label Switching
NaaS	Network as a Service
NMRG	Network Management Research Group
NMS	Network Management System
OAM	Operations Administration and Maintenance
OFDM	Orthogonal Frequency-division Multiplexing
OFDMA	Orthogonal Frequency-Division Multiple Access
OPEX	Operational Expenditures

OSS	Operations Support System
PDN-GW	Packet Data Network Gateway
QoE	Quality of Experience
QoS	Quality of Service
ROI	Return of Investment
RAN	Radio Access Network
RRM	Radio Resource Management
SGW	Serving Gateway
SME	Small and Medium Enterprises
SLA	Service Level Agreement
SON	Self Organized Networks
TCO	Total Cost of Ownership
TMF	TeleManagement Forum
UC	Use case
UMF	Unified Management Framework
VoIP	VoIP - Voice over IP
VPN	Virtual Private Network

11 Definitions

Network Empowerment Mechanism (NEM) – *a functional grouping of objective(s), context and method(s) where "method" is a general procedure for solving a problem. A NEM is (a priori) implemented as a piece of software that can be deployed in a network to enhance or simplify its control and management (e.g. take over some operations). An intrinsic capability of a NEM is to be deployable and interoperable in a UMF context (in a UMF-compliant network).*

Unified Management Framework (UMF) – *A framework that will help produce the unification, governance, and "plug and play" of autonomic networking solutions within existing and future management ecosystems. The objective of the UMF is to facilitate the seamless and trustworthy deployment of NEMs. The UMF has three core blocks that are used by the NEMs to achieve this, as shown in the figure below.*



Governance block (GOV) – *A core UMF block that aims to give a human operator a mechanism for controlling the network from a high level business point of view, that is, without the need of having deep technical knowledge of the network.*

Knowledge block (KNOW) – *An infrastructure that uses and/or manipulates information and knowledge, including information/knowledge flow optimization within the network.*

Coordination block (COORD) – *A core UMF block that aims to ensure the proper sequence in triggering of NEMs and the conditions under which they will be invoked (i.e. produce their output), taking into account operator service and scenario requirements and at the same time the needs for conflict avoidance, stability control and joint optimization through the corresponding functions.*

Use case – *A descriptor of a set of precise problems to be solved. It describes steps and actions between stakeholders and/or actors and a system, which leads the user towards an added value or a useful goal. A use case describes what the system shall do for the actor and/or stakeholder to achieve a particular goal. Use-cases are a system modelling technique that helps developers determine which features to implement and how to gracefully resolve errors.*

12 Annex A: Messages exchanged between NEM and UMF

This annex describes the description of the blocks and messages involved in the cooperation between NEMs through the UMF. Each of the subsequent sub-sections relates to interactions in the different Use Case chapter discussions. The specifications of the blocks and messages given here are preliminary and not final, but are given here to give context behind the ongoing work on integration, which is being done in collaboration with WP2.

A.1 Involved in Use Case 1

Block

Name	**Coordination**
Description	
List of functions	Orchestration
List of external interfaces	COORD-GOV, COORD-KNOW, COORD-NEM

Interfaces

Name	**COORD<->NEM**
Description	Interface between Coordination and a NEM
List of messages	Unique message names

Messages

Name	**Unique name: NEM-to-COORD**
Type	
List of source	*NEM*
List of destination	*COORD*
List of workflows	
List of objects	Parameters, Metrics, Timing
Constraints (?)	*Scope (e.g. what base stations are affected?)*

Name	**Unique name: COORD-to-NEM**
Type	
List of source	*COORD*
List of destination	*NEM*
List of workflows	
List of objects	Token, Constraint
Constraints (?)	

Objects

Name	**Unique name: Parameters**
List of fields	Delay, Jitter, Packet Loss, QoS/Load Labels

Name	**Unique name: Metric**
List of fields	Adapted QoS/Load Labels, Cluster Thresholds, Decision making success rate.

Name	**Unique name: Timing**
List of fields	Convergence time, Interval between two consecutive triggers.

Name	**Unique name: Token**
List of fields	Grant, Revoke

Name	**Unique name: Constraint**
List of fields	Acceptable output levels per service, expected interval between two triggers of the NEM, targeted elements.

A.2 Involved in Use Case 2

Block

Name	**Knowledge**
Description	
List of functions	Information Collection and Dissemination (ICD), Information Storage and Indexing (ISI), Information Processing and Knowledge Production (IPKP)
List of external interfaces	KNOW-NEM

Functions

Name	**Information Collection and Dissemination (ICD)**
Requirement	
Description	Receives and analyses the request, disseminates the requested information. In case the information is not available, it asks TCP Vegas to produce it
Constraints	
List of operations	Information Dissemination, Information Collection (if the requested information is not available)
List of implementations	

Name	**Information Storage and Indexing (ISI)**
Requirement	
Description	Advices ICD where the required information is stored
Constraints	
List of operations	
List of implementations	

Name	**Information Processing and Knowledge Production (IPKP)**
Requirement	

Description	Produces the information in the requested granularity if needed
Constraints	
List of operations	Information Aggregation (IA)
List of implementations	

Operations

Name	**Information Dissemination**
Description	Receives and analyses the request, disseminates the requested information.
Constraints	
List of input data	Request of the number of variations of the Congestion Window
List of output data	Response (number of variations of the Congestion Window)
List of non-functional requirements	

Name	**Information Collection (if the requested information is not available)**
Description	In case the information is not available, it asks to be produced from TCP Vegas NEM
Constraints	
List of input data	Request of the number of variations of the Congestion Window
List of output data	trigger to TCP Vegas NEM
List of non-functional requirements	

Name	**Information Aggregation (IA)**
Description	In case the information is not at the required granularity, it combines the available information and returns it to the requested granularity
Constraints	
List of input data	number of variations of the Congestion Window
List of output data	Aggregated information to the required granularity
List of non-functional requirements	

Interfaces

Name	**KNOW<->NEM**
Description	Interface between Knowledge and a NEM
List of messages	NoCWND_Request, NoCWND_Response

Messages

Name	**NoCWND_Request**
Type	
List of source	*NSC NEM*

List of destination	*KNOW*
List of workflows	
List of objects	Request for the number of variations of the Congestion Window
Constraints (?)	

Name	**NoCWND_Response**
Type	
List of source	*KNOW*
List of destination	*NSC NEM*
List of workflows	
List of objects	Request for the number of variations of the Congestion Window
Constraints (?)	

Name	**NoCWND_Request**
Type	
List of source	*KNOW*
List of destination	*TCP Vegas NEM*
List of workflows	
List of objects	number of variations of the Congestion Window
Constraints (?)	

Name	**NoCWND_Response**
Type	
List of source	*KNOW*
List of destination	*NSC NEM*
List of workflows	
List of objects	number of variations of the Congestion Window
Constraints (?)	

Objects

Name	**Number of Variations of Congestion Window**
List of fields	integer

A.3 Involved in Use Case 3

Block

Name	**Knowledge**
Description	
List of functions	Providing relevant reasoned context knowledge
List of external interfaces	KNOW-NEM

Functions

Name	**NEM7: Context data Annotation, Context data reasoning with Linked Data, Knowledge delivery**
Requirement	
Description	Enriches Contextual Knowledge based on rule-based reasoning
Constraints	
List of operations	Context Acquisition, Context Filtering, Annotation from environmental variables, Reasoning from linked data, Dissemination.
List of implementations	Not yet available

Operations

Name	**Context Acquisition**
Description	NEM 7: To acquire the context data that the Context Acquisition NEM (NEM 21) has provided from relevant sources NEM45: To acquire knowledge derived and provided by NEM21/NEM7.
Constraints	Availability of data from different sources in proper time and location The frequency of acquiring dynamic context data.
List of input data	KPI_id, KPI_Source
List of output data	Enriched information
List of non-functional requirements	

Name	**Context Filtering**
Description	To filter the context data being periodically provided by various context sources. Acquire specific context data with reference to the CSFS NEM specific method.
Constraints	
List of input data	Context_data_type
List of output data	
List of non-functional requirements	

Name	**NEM7: Annotation from environmental variables**
Description	To annotate information from different sources to the raw data (i.e., KPIs provided by NEM 21)
Constraints	
List of input data	User_contextual_data, Environmental_data
List of output data	Reasoned_Info_stage_1
List of non-functional requirements	

Name	**NEM7: Reasoning from linked data**
Description	Gathers related external knowledge from Linked data source
Constraints	
List of input data	Linked_data
List of output data	Reasoned_Info_stage_2
List of non-functional requirements	

Name	**Knowledge dissemination**
Description	Disseminates the derived knowledge required by NEM45, e.g., User context.
Constraints	
List of input data	
List of output data	Depends on Context Discovery NEM specific method requirement.
List of non-functional requirements	

Interfaces

Name	**KNOW<->NEM**
Description	Interface between Knowledge and a NEM
List of messages	Unique message names

Messages

Name	**Register_NEM: NEM-to-KNOW**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	
List of objects	Parameters, Metrics, Timing
Constraints (?)	*Scope (e.g. user, environment, network)*

Name	**Info_Push: NEM-to-KNOW**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	
List of objects	Parameters, Metrics, Timing
Constraints (?)	*Scope (e.g. user, environment, network)*

Name	**Info_Pull: NEM-to-KNOW**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	

List of objects	Parameters, Metrics, Timing
Constraints (?)	*Scope (e.g. what base stations are affected?)*

Name	**Subscribe_Context_data: NEM-to-KNOW**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	
List of objects	
Constraints (?)	

Name	**Request_Info: NEM-to-KNOW**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	
List of objects	
Constraints (?)	

Name	**Collect_Info: KNOW-to-NEM**
Type	
List of source	*KNOW*
List of destination	*NEM*
List of workflows	
List of objects	
Constraints (?)	

Objects

Name	**Unique name: Parameter**
List of fields	

Name	**Unique name: Metric**
List of fields	

Name	**Unique name: Timing**
List of fields	

A.4 Involved in Use Case 4

Block

Name	**GOV**
Description	

List of functions	Policy provisioning
List of external interfaces	COORD-NEM

Interfaces

Name	**GOV <->NEM**
Description	Interface between GOV and a NEM
List of messages	Activation request (NEM->GOV) Policy Provision (GOV->NEM) Policy installed (NEM ->GOV)

Name	**NEM <->NEM**
Description	Interface NEM 11/26 to NEM 12
List of messages	overload prediction detected @ station #

Functions

Name	**Policy Provisioning**
Requirement	
Description	Receives all parameters that determine the policies guiding operation of NEMs, and transmits to the NEMs
Constraints	
List of operations	Provide and install policies to NEMs
List of implementations	

Operations

Name	**Provide policy to install policy (for NEM 12)**
Description	Provide necessary information for executing the self-optimizing NEM as defined by the policy
Constraints	
List of input data	Parameters, range and step size; List of KPIs used for the utility functions; information on the utility function
List of output data	Controlled parameters
List of non-functional requirements	

Name	**Overload detection**
Description	Overload detection via measurements and predictions
Constraints	
List of input data	Measured or predicted load
List of output data	Activation trigger
List of non-functional requirements	

A.5 Involved in Use Case 6

Block

Name	**Knowledge**
Description	
List of functions	Information Collection and Dissemination (ICD), Information Storage and Indexing (ISI), Information Processing and Knowledge Production (IPKP)
List of external interfaces	KNOW-NEM

Functions

Name	**Information Collection and Dissemination (ICD)**
Requirement	
Description	Receives and analyses the request, disseminates the requested information. In case the information is not available, it asks to be produced from the most appropriate load prediction NEM
Constraints	
List of operations	Information Dissemination, Information Collection (if the requested information is not available)
List of implementations	

Name	**Information Storage and Indexing (ISI)**
Requirement	
Description	Advices ICD where the required information is stored
Constraints	
List of operations	
List of implementations	

Name	**Information Processing and Knowledge Production (IPKP)**
Requirement	
Description	Produces the information in the requested granularity if needed
Constraints	
List of operations	Information Aggregation (IA)
List of implementations	

Operations

Name	**Information Dissemination**
Description	Receives and analyses the request, disseminates the requested information.
Constraints	
List of input data	request
List of output data	response

List of non-functional requirements	

Name	**Information Collection (if the requested information is not available)**
Description	In case the information is not available, it asks to be produced from the most appropriate load prediction NEM
Constraints	
List of input data	Metrics used (and where), type of output, the timings of the algorithm (convergence time and frequency of updates)
List of output data	The most appropriate NEM to be used
List of non-functional requirements	

Name	**Information Aggregation (IA)**
Description	In case the information is not at the required granularity, it combines the available information and returns it to the requested granularity
Constraints	
List of input data	Available information related to the request
List of output data	Aggregated information to the required granularity
List of non-functional requirements	

Interfaces

Name	**KNOW<->NEM**
Description	Interface between Knowledge and a NEM
List of messages	Unique message names

Messages

Name	**Unique name: NEM-to-KNOW**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	
List of objects	Metrics, Timing, Output
Constraints (?)	*Scope (e.g. what base stations are affected?)*

Name	**Unique name: Load_Provision**
Type	
List of source	*KNOW*
List of destination	*NEM*
List of workflows	
List of objects	Load Prediction
Constraints (?)	

Name	**Load_Request**
Type	
List of source	*KNOW*
List of destination	*NEM*
List of workflows	
List of objects	Load Prediction
Constraints (?)	

Objects

Name	**Unique name: Output**
List of fields	Output type (ranges, values)

Name	**Unique name: Metric**
List of fields	Cell load

Name	**Unique name: Timing**
List of fields	Convergence time, Interval between two consecutive triggers

Name	**Unique name: Load Prediction**	
List of fields	Cell Load	

A.6 Involved in Use Case 7

The blocks, functions, operations and messages needed for the functionalities described above are the following:

Block

Name	**UMF Core Services Knowledge Block (KNOW)**
Description	KNOW is an infrastructure that manipulates information & knowledge, including information/ knowledge flow optimisation within the network.
List of functions	- Information collection and dissemination
List of external interfaces	- KNOW with NEMs Interface – N-KNOW (NEMs <–>KNOW)

Functions

Name	**Information collection and dissemination**
Requirement	Mandatory
Description	This function is responsible for activities related to information collection, retrieval, dissemination and querying.
Constraints	
List of operations	Information collection Information retrieval

	Information dissemination Information querying
List of implementations	Pub-sub information retrieval/dissemination (DistributedDecisionEngine, Siena and VLSP infrastructures) Push/pull information retrieval/dissemination (VLSP infrastructure) Information monitoring (NKUA, Lattice-IMO-VLSP and DDE) Information discovery protocols Information diffusion algorithms

Interfaces

Name	**KNOW with NEMs Interface – N-KNOW (NEMs<–> KNOW)**
Description	
List of messages	getParameter

Messages

Name	**getParameter**
Type	
List of source	*NEM*
List of destination	*KNOW*
List of workflows	
List of objects	Name/ID of the parameter to retrieve
Constraints (?)	

Objects

Name	Name/ID of the parameter to retrieve
List of fields	Identifier

Block

Name	**Governance**
Description	Governance aims to give a human operator a mechanism for controlling the network from a high level business point of view, that is, without the need of having a deep technical knowledge of the network.
List of functions	Distribution NEM Management
List of external interfaces	

Functions

Name	**NEM Management**
Requirement	Mandatory
Description	The NEM management function allows the control of the deployed NEMs and the management of their lifecycle (which includes the activation and deactivation of the autonomic functionality). Furthermore, the NEM management function includes the NEM registry, which stores the available NEMs, their activity phase, their definition (manifest), and current configuration parameters (mandate). The mandate includes runtime information like the network resources being managed by each of the NEMs. NEM Management function may request the report of specific network information (e.g. specific performance indicators, measurements and information related to NEM's operation and environment conditions, capabilities, behaviour). The reported information is processed and is forwarded to Policy Derivation and Management and to H2N, especially information that is related to detection of critical states of self-managed operations/devices. Distribution operation is included hereso to encapsulate the communication mechanism between Governance and NEMs. It allows the other functions of the Governance block to be independent of the communication aspects for the interconnection with NEMs.
Constraints	
List of operations	NEM insertion/modification/removal/retrieval NEM status modification NEM mandate modification Reporting Policy generation
List of implementations	

Name	**Policy distribution**
Requirement	Mandatory
Description	Distribution encapsulates the communication mechanism between Governance and NEMs. It allows the other functions of the Governance block to be independent of the communication aspects for the interconnection with NEMs.
Constraints	
List of operations	Policy distribution Reporting policy distribution Send NEM Mandate Change NEM Status
List of implementations	

Operations

Name	Policy distribution
Description	Sends to a given NEM or set of NEMs the policies.
Constraints	
List of input data	Policy, NEM id
List of output data	Policy (at NEM level)
List of non-functionalrequirements	

Name	Send NEM Mandate
Description	Sends a new Mandate to a given NEM. This operation is used for instance to change the activity phase of a given NEM.
Constraints	
List of input data	Mandate, NEM id
List of output data	Mandate
List of non-functionalrequirements	

Interfaces

Name	**GOV -NEM interface**
Description	Interfaces offered between NEMs and Governance. Governance uses it to retrieve information about the NEMs, to configure it through the mandate or policies, and to configure how the NEM is going to report information. NEMs use it to send information to Governance (notifications, reports, call for governance or register themselves)
List of messages	Set NEM Status Get NEM Status Set ReportingSpecification AddReportingSpecification GetReportingSpecifcation RemoveReportingSpecification Set Policies AddPolicies GetPolicies RemovePolicies Get Notifications Get NEM Manifest Se NEM Mandate Call For Governance Send Notification Register NEM

Messages

Name	**Set NEM Status**
Type	

List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	NEM Status
Constraints (?)	

Name	**Get NEM Status**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	NEM Status
Constraints (?)	

Name	**Set Policies**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	Policy
Constraints (?)	

Name	**AddPolicies**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	Policy
Constraints (?)	

Name	**GetPolicies**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	Policy
Constraints (?)	

Name	**RemovePolicies**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	

List of objects	Policy
Constraints (?)	

Name	**Get NEM Manifest**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	NEMManifest
Constraints (?)	

Name	**Set NEM Mandate**
Type	
List of source	*Governance*
List of destination	*NEM*
List of workflows	
List of objects	NEMMandate
Constraints (?)	

Objects

Name	**Status**
List of fields	StatusId

Name	**ReportingSpecification**
List of fields	ReportingPolicies

Name	**Policies**
List of fields	List<PolicySet>

Name	**Notifications**
List of fields	List<Notification>

Name	**NEMManifest**
List of fields	Name ProviderID Version Release Date Feature description URL Hosting Manageable Equipments Atomic NEM Atomic loop

	AcquiredInputs ExternalInputs AvailableOutputs PossibleActions ConfigurationOptions

Name	**NEMMandate**
List of fields	GOV_address COORD_address KNOW_address Instance ID ManagedEquipments ConfiguredOptions

Name	**Problem**
List of fields	Problem ID Originating system Priority Description First Alert Category Comments Time raised Reason Underlying problems Parent problems Affected location Root cause services Root cause resources