M Ű E G Y E T E M 1 7 8 2

Scalable Quality of Service Solutions for IP Networks and Services

Csaba Simon
Electrical engineer

PhD Dissertation

Doctoral School of Informatics
Faculty of Electrical Engineering and Informatics
Budapest University of Technology and Economics

Supervised by
Dr. Tamás Henk

Department of Telecommunication and Media Informatics
Faculty of Electrical Engineering and Informatics
Budapest University of Technology and Economics

Budapest, Hungary
October, 2011

Table of Contents

List of Figures

Chapter 1 Introduction

The telecommunication industry has always been able to incorporate the new technologies. Starting from the early telegraph systems it has continuously evolved to offer richer and diversified services to customers. The recent decades challenged again the telecommunication industry, as the digital telephone systems are integrated with the technologies originally developed to support computer networks. This process led to the convergence of networks and Internet Protocol (IP) emerged as a common platform that forms the basis of future telecommunication systems.

The major appeal of the IP technology relies in its flexibility to implement new services to the customers, which is the key to remain competitive in a global market. The quality of these services experienced by the users is a key factor to accept and embrace new services. Although pricing and marketing are becoming more and more important in current telecom markets, there are still open technological questions, which, if properly addressed, give a competitive edge to the service providers.

The most widespread standards that answer these challenges are the Integrated Services [1] and Differentiated Services [2]. The latter scales better with the number of flows and one of its variants is the relative service differentiation [3], introduced as a per hope based service [4][5]. The advantage of this service is its predictability at node level (i.e., per hop), but the operators should offer their services over their domains, which calls for a network-wide service definition. A much detailed insight into these issues is provided in Part I of this Dissertation.

The extension to global level and the rich set of services come with a price though, as they require the introduction and deployment of new network and service enablers. The management of this evolving network and service architecture hits the limits of established (legacy) management frameworks on scalability. In legacy network management, the management-plane is a centralized process [6][7], being responsible for a vast area of functionalities, as described by the Fault, Configuration, Accounting, Performance, Security (FCAPS) structure of Telecommunication Management Network (TMN) [8]. In order to meet the above challenges of future networks [9] management systems have to become more dynamic, more self-managing, and they have to participate more actively in (inter)-networking [10], going beyond “classical” or legacy network management paradigms, as detailed in Part II of this Dissertation.

The rest of the dissertation discusses the above two challenges in two parts (Part I and Part II) with their complete introduction, related works, solutions and evaluations and summaries. At the end of the dissertation in Part III an overall conclusion is drawn and applicability of the new results is enumerated.

Part I Network-wide proportional services

Chapter 2 Introduction

The economies of scale offered by the use of single integrated architectures that span the globe and enable customized services would offer both social and economical benefits to all stakeholders. Several existing technologies can be used as building blocks of such global picture [1] [2], but these blocks cannot be glued together as one would do with the pieces of a puzzle. The major difficulty is not the missing interface, but the fact that these building blocks have their intrinsic limitations in terms of scalability or the combination of these will not yield a fully predictable system. As already mentioned in Chapter 1, the relative service differentiation [3], a DiffServ variant is a proposal to achieve this goal.

The relative service differentiation, similarly to DiffServ, handles packets on a per-hop basis. Nevertheless, the way to achieve end-to-end differentiation between the predefined classes with the combination of these per-hop services remains a challenge [11]. The difficulties stem from the fact that delay or reliability are additive performance metrics. The path reliability is additive after computing logarithms [12], the path delay is the sum of the delays over each link and router.

For these cases, a possible approach to provide end-to-end differentiation over a per hop QoS infrastructure would be to decompose the end-to-end QoS requirement into a series of local QoS objectives. There were proposals to tweak around and provision end-to-end QoS in IP networks. The authors of [13] propose a solution to assure end-to-end weighted proportional service for responsive TCP/IP flows, but their solution requires the modification of the end systems. The authors of [14] propose and end-to-end solution, but related to the now obsolete Asynchronous Transfer Mode technology, while [15] proposes a complex formal model and mechanisms to configure routers to achieve the desired end-to-end behavior. Nevertheless, the optimal partition of QoS constraints is a NP-hard problem [16], thus approximate algorithms should be used [17]. Also, tunable performance on a per-hop basis does not result in easily controllable end-to-end behavior [11]. (e.g., opposed to packet losses) and decomposition techniques cannot be applied to non-additive QoS parameters.

The objective of this part of the Dissertation is to develop a solution that assures the QoS of a network in a predictable and scalable manner in IP networks.

In order to be able to build a global network that meets user expectations the network must be able to offer predictable Quality of Service (QoS) to the applications, and maintain this offer over a global scale. We introduced a domain-wide proportional service that guarantees that the ratio between the qualities of service experienced by two users served in different classes will remain the same irrespective of the changing network conditions. This approach differs from the widely used per hop behavior (PHB) [18] based ones and we proposed several algorithms that impose the proportional service

in the network and validated them through simulations. Thus a user who selected a higher service class will know that she/he always gets a better service and she/he will also know how much better this service will be.

The next Chapter presents our literature survey on prior art and on related research, including solutions that will be used later. Then Chapter 4 presents the proposed network-wide service model and its analytical analysis. The next Chapter presents three algorithms for different traffic and architecture variants to enforce the proposed service. Finally we summarize this part of our work.

Chapter 3 Related works

The QoS of IP based networks has a vast literature. In the first part of this section we review those works that are related to our proposals. These proposals deal with variants of the proportional service differentiation. Some of these methods serve as the starting point for our work, as detailed in Chapter 4. In the second subchapter we point to those works that are not covering the proportional service differentiation area, but are used or referenced in our work.

3.1 Relative Service differentiation methods

The definition of relative service differentiation is that a class *i* is better (or at least not worse) than class (i-*1*), without respect to the classes' loads. In the case of relative differentiation the QoS parameters are relative to each other, depending on the overall network capacity.

It has been shown in the literature that in the case of the "classical" DiffServ disciplines, if the higher quality classes are overpopulated compared to a lower class, then the flows classified in the higher class might receive worse service than the lower class flows [3]. The advantage of the proportional services approach is that the QoS differentiation is a-priori known to the flows, as opposed to the original DiffServ PHB based solution.

Price differentiation

Price based service access over the Internet has been introduced in the early 1990s by the authors of [19]. Their model, called smart markets, proposed that each packet should carry a bid that represents how much the user is willing to pay for transmitting it over the network. Then the service of the packets over congested links is granted to the packets that offer higher prices and fit within the capacity bounds of the link. Obviously, the service level in terms of throughput or other QoS parameters is hard to be expressed. Also, this model lacks the feedback mechanisms that would a priori notify the users about the minimum required bid to obtain a desired service level.

This idea has been later extended by the authors of [20]. Their idea was to achieve relative service differentiation by using a simple strict priority scheduler (or buffer management scheme) that statically differentiate the service classes without considering their loads, and an appropriate pricing scheme like the Paris Metro Pricing scheme [20]. The pricing scheme changes the prices of the service classes according to their loads, supposing that the load of a class can be reduced if its price is increased. Several follow up works tried to derive conditions of congestion functions that can guarantee the viability of this concept [21][22][23]. According to [3], the drawback is that this scheme is effective over relative long timescales resulting in an inconsistent and unpredictable class differentiation.

Proportional service differentiation – original proposal

Another approach to provide relative service differentiation is to use dedicated forwarding mechanisms (schedulers or buffer management schemes) that dynamically

allocate resources to classes according to their loads. The [3] proposes such a forwarding mechanism, which achieve proportional service differentiation between classes over a relatively short time interval of (t, t+τ):

$$\frac{\mathrm{Qi}(t, t+\tau)}{\mathrm{Qj}(t, t+\tau)} = \frac{c_i}{c_j} \qquad (1)$$

where $Q_i(t, t+\tau)$ is the performance measure (delay or packet loss) for class *i,* while c_1 $c_1 < c_2 < \ldots < c_N$ are parameters. In [4] $Q_i(t, t+\tau)$ is the average packet loss, while in [5] it is the average queuing delay of class *i*.

As explained in [3], the relative differentiation model provides the network operator with "tuning knobs" to adjust the quality spacing between classes, independent of the class loads. When this spacing is feasible in short timescales, it can lead to predictable and controllable class differentiation.

In [3][4] and [5] the proposed forwarding mechanisms use different packet queues for each service classes, that are scheduled/managed in such a manner that the above formula will be approximately true. For proportional delay differentiation the proposed scheduler is called waiting time priority (WTP) scheduler, where the priority of a packet from class *i* increase proportionally with its waiting time:

$$p_i(t) = \frac{w_i(t)}{c_i} \qquad (2)$$

Using simulation the authors found that the scheduler approximates the proportional delay differentiation in heavy load conditions, even with a monitoring timescale of a few tens of packet transmission times.

For proportional loss rate differentiation a scheduler, an aggregate backlog controller and a dropper are used [4]. The backlog controller monitors the aggregate backlog of waiting packets in the forwarding engine, and decides whether a packet should be dropped. When it should, the aggregate backlog signals the dropper to drop a packet from the class *j* with the minimum normalized loss rate:

$$j = \arg\min_i \left(\frac{l_i(t)}{c_i} \right) \qquad (3)$$

The loss rate l_i is calculated using an infinite buffer in which case the proportional loss rate constrains is met more accurately and it is simpler to implement, or a buffer with size *M* in which case the dropper is more adaptive to non-stationary traffic loads.

Other works on the PHB-based proportional services

Starting from the above works there has been published a lot of improvements/extensions to the original PHB-based ideas, aiming to introduce new schedulers or achieve more robust operation, e.g., [5] [24] [25], but a more recent and complete discussion of the related literature is to be found in [26]. Some of the works propose the combination of loss/delay guarantees [24] [28]. Recently several authors made attempts to provide

support for a limited form of absolute QoS based on relative differentiation mechanisms [26] [29] [30].

Relative service differentiation for TCP flows

The provision of relative service differentiation to TCP flows has been primarily investigated in the context of wired networks. Authors of [31] deploy traffic metering besides queue management and scheduling mechanisms to achieve class-based service differentiation, whereas [32] proposes the combined use of marking algorithms and explicit congestion notification (ECN). For the same purpose the authors of [33] propose a weighted proportional window control mechanism, which is based on the advertised window of TCP connections.

Since TCP flows are reacting in a different way to the wireless medium, where packet losses are not necessary the indication of congestion, the authors of [34] investigate the aspects of relative proportional services for packet delays, but they focus on wireless environment only. Their primary goal is to alleviate the anomalies suffered by TCP flows in wireless environment; therefore they propose a cross layer scheduling to achieve the desired differentiation. Authors of [35] tightened their area of investigation even further, focusing on satellite networking. They also propose a cross layer solution, where both layer 2 and layer 4 controls are introduced.

End to end, per-flow solutions

The goal of the Dynamic Packet State method is to approximate the service provided by a stateful, IntServ like, network. The authors claim that they can achieve fair bandwidth allocation [36], approximating the service provided by a network in which every node provides fair queuing. In [37] the authors show that the network called SCORE (Scalable Core) can provide end-to-end per flow delay and bandwidth guarantees as defined in the IntServ architecture.

In Dynamic Packet State, each packet carries state information in his header. This information is initialized by the ingress router. Core routers process each incoming packet based on the state carried in the packet's header. The information in the packet's header will be updated in core routers and transmitted towards the next nodes by the packet itself. The core routers also update their internal state. Thus, in core routers packet classification is no longer necessary. Packet scheduling is based on the state information carried with the packet headers.

Dynamic Packet State is used to approximate the following schedulers:

- Fair queuing with Core-Stateless Fair Queuing (CSFQ)[36]
- Weighted Fair queuing with WCSFQ [38]
- Jitter Virtual Clock, which is a non-conserving version of the Virtual Clock algorithm [39], with Core –Jitter-Virtual Clock [37]

For admission control, a lightweight signaling protocol is considered. The core routers try to estimate a close upper bound on the aggregate reserved rate. In this way, the admission control's complexity can be reduced, because we don't need to maintain the dynamic state for flows in each node.

A much more recent attempt to a similar, per-flow proportional architecture focusing on packet loss probabilities is presented in [12].

Random early drop based solutions

An other proposed forwarding mechanism for relative service differentiation is called RIO (Random Early Detection with *In/Out* bit) [40] for 2 service classes (*In* profile class and *Out* profile class) and its variants for more than 2 service classes [41][42]. The idea of RIO is to discriminate the *In* and *Out* packets in time of congestion, by using a single packet queue and the Random Early Drop (RED) [43] buffer management scheme with different parameters for the 2 classes.

The RED parameters can be seen in Figure 1.

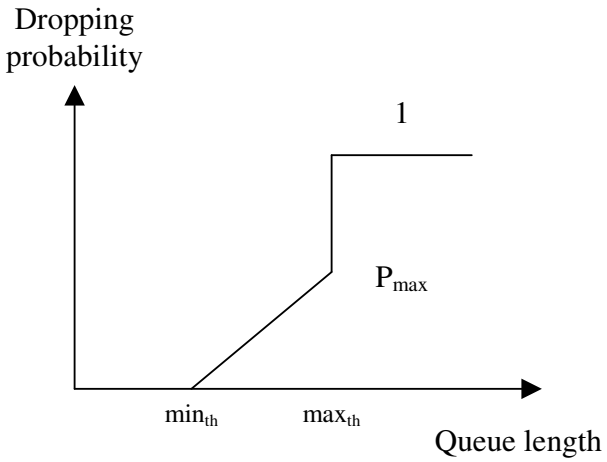

Figure 1. RED scheme

The discrimination between *In* and *Out* classes in [40] is achieved by:

Choosing the min_{th} lower for *Out* packets then for *In* packets (in this way *Out* packets are dropped much earlier than *In* packets)

Choosing the max_{th} lower for *Out* packets then for *In* packets (in this way *Out* packets are dropped much earlier with probability 1 than *In* packets)

Choosing the P_{max} higher for *Out* packets

Choosing the total average queue to drop *Out* packets, in contrast with choosing only the average *In* packets when dropping *In* packets.

The drawback of these buffer management schemes are the unpredictability of the service differentiation in a quantitative way. In other words, it is hard to provision how much better a higher class than a lower is.

3.2 Methods for flow measurement and control

All the above theoretical definitions of QoS in IP based networks rely on the notion of flow. Since IP is a packet based protocol, the practical implementation of the above

disciplines requires the identification of the flows in the networks. The flows in many cases are described using traffic matrices. Therefore we review the literature related to the measurement based characterization of networks and the methods to infer the traffic matrix based on measurement. Finally we present a work on practical implementation of traffic engineering and QoS handling in networks, which proposes an interesting solution to avoid congestion collapse in the networks.

Flow Measurement Methods

An important technical aspect of the control and management of info-communication networks is how to compute the load on the network links. Network operators ran across this issue since long ago, as they need precise picture of the traffic load of their networks, their goal being to tune the configuration of the network to adapt to changes in the traffic demands. Although there has been earlier work in this field, a pioneering work in this research area is the one that investigates the task of describing traffic load in large Internet backbones [44] (i.e., operated by tier-1 ISPs). This article also provides a good review of the related earlier works. The difficulty of this task is coming from the large size of data and the high speed of the network. The authors observe that for (the operators of) ISPs the most important issue is to have a domain-level view on their traffic, since their services are perceived by customers on that level. Therefore the authors propose to compute the offered loads of a flow based on measurements only at the ingress routers. The authors emphasize the importance of the daily, weekly or longer profiles, as well.

This article, as many later works in this field (e.g., the ones referenced later on in this sub-chapter) rely on the passive flow-level measurements that operators routinely collect in today's networks. This is an important aspect, because valuable data is readily available without any extra cost. Typically this type of data is collected using Cisco's NetFlow [45], which is a further advantage, as it is a well established technology and vendors do have experience in implementing it in an efficient manner, assuring that their routers perform according to the specifications even under heavy traffic load conditions.

An other author together with his several co-authors with large impact on this field has an activity spanning across the last decade [46][47][48][49], in which they investigated various aspects of this field. Some of his works were co-authored by the authors of the aforementioned [44] article. The importance of his works is given by his access to the US incumbent telecom operators' backbone networks, where together with his coauthors they tested both scalable and efficient mechanisms in one of the world's most complex networks. In [46] the authors combined both passive and active monitoring mechanisms and they have shown that, if cleverly deployed, active components, including a mesh of measurement machines located at major router centers can even cope with the performance requirements of such complex backbone networks. In [47] the authors proposed that routers sample packets in a consistent fashion, to compute a "trajectory" that follows a subset of the packets as they flow through the network. Recent work [49] focused on inferring accurate routing information using passive monitoring methods, which is important in identifying flow paths over the network.

While the previously mentioned works have been written in the context of the US backbone networks, the issues of NetFlow based traffic matrix estimation are researched over European backbones, as well [50][51][52]. Among these, the INTERMON project

[53] proposed a complex tool system to deal with various aspects of traffic measurement and QoS parameter assessment.

Although inferring traffic data based on raw traffic demand has its challenges, especially in reducing of the measurement and storage costs [54][55][56], there are promising proposals to alleviate this problem [57][58]. Works published later that obtained precise traffic matrices, by combining analytical modeling with partial flow measurement [51] or proposing distributed versions of centralized measurement-based solutions augmented with model that is inferred based on earlier measurements on the network [50].

This field is still active research-wise. One research direction is the use of Open Flow architecture, as it promises to provide a future-proof solution to several networking challenges [59]. Authors of [60] developed a tool, called OpenTM, which accurately estimates traffic matrices at the cost of low overhead using direct measurements in a setting where the switches keep track of flow statistics, reducing by an order of magnitude the flow table entries and the number of control messages.

A different research group followed the path of enhancing the accuracy of their traffic matrix estimations with various methods and tested the precision of their proposals in large backbone networks. In [62] they used neural networks, in [61] simulated annealing and a joint time-frequency analysis in transform domain in [63].

The solutions cited in this section up to now mainly focused on providing the traffic matrix at the time of the measurement. A special issue within traffic matrix inference is the prediction of the traffic load based on past observation. Or put it in other way, the goal is to provide the traffic matrix in the future epochs after the time of the measurement. This topic has been researched from the early ‘90s, but it still presents interests to the researchers, who apply several modeling instruments. The authors of [64][65] rely on regression methods, the authors of [66] use the apparatus of machine learning, the approach of [67] is a wavelet based one, multifractal models are used in[68] and the authors of [69] propose a neural tree based solution.

Network Border Patrol

In paper [70] the authors present a novel architecture, which prevents the network from congestion collapse due to undelivered packets. This problem arise when the bandwidth is consumed by packets that later are dropped before reaching the destination. This is mainly true when the flows use the unresponsive UDP protocol, instead of flow controlled TCP. Local scheduling algorithms, such as Weighted Fair Queuing (WFQ), cannot eliminate congestion collapse, thus using such algorithms we cannot achieve global max-min fairness without the help of additional mechanisms. Network Border Patrol (NBP) cannot provide global max-min fairness to competing network flows alone. By combining with other techniques, such as fair queuing in core routers, we can achieve global max-min fairness though.

The basic principle of the NBP is to compare, at the borders of the network, the rates at which each flow’s packets are entering and leaving the network. If packets enter the network faster than they are leaving it, then the network is very likely to buffer or to discard the flow’s packets. NBP allows packet forwarding into the network only if the

input and output traffic rates are the same for the flow. The basic idea of this paper is later used in the Chapter 4 of this Dissertation.

To achieve the functionality, only the ingress and egress routers at the network borders (considering a DiffServ domain) are expected to monitor and control the rates of individual flows. The edge routes exchange feedback information with each other to gather information about the flows. The egress routers, using rate monitoring algorithms, determine the speeds with each flow's packets are leaving the network. The information is fed back to the ingress routers, whereas rate control algorithms are used to police the rate at which each flow's packets enter the network.

The simulation results show that NBP used with WFQ and Core-Stateless Fair Queuing (CSFQ), approximate global max-min fairness for competing network flows.

Chapter 4 Network-wide proportional services

In the previous Chapter we have reviewed the literature of flow-aggregate based QoS disciplines, the DiffServ. We have found that there is a variant, the proportional differentiation services, whose advantage is that the user will know in advance, even over short time periods the relative advantage of a certain class compared to any other one. This can be described as the relative differentiation "sets the knobs" of performance between QoS classes in a predefined manner. Nevertheless, we also learned that there is a problem: these QoS disciplines were defined on a per hop basis, but the network operators should define their services over their whole network domain. Conversion between network-wide (or end-to-end) and per hop based is not a trivial task.

Due to these shortcomings of PHB based differentiated services, including the relative differentiation variants, we consider that there is a need for a new network architecture that assures *relative differentiation* to flows crossing an administrative domain, which will be referenced to as *network-wide proportional service***.** Therefore in this Chapter we propose a novel QoS discipline, the network-wide proportional service. Further below we will explain the properties of this discipline, we propose an architecture that can realize it and we analyze it numerically.

4.1 Service model

In our proposal we use the notation for the proportional service, as introduced by eq. (1), in which the time dependence is omitted for ease of notation. Although time dependence was used in former definitions, it was not utilized. The main reason was to stress that the Q is defined over shorter periods, e.g., τ is of order of seconds, and every computation has been performed and valid for arbitrary time instants *t*.

Definition 1.1 Let us consider the differentiation between *m* different classes, where q_i is a QoS performance measure for class *i*. The Proportional Service (PropServ) model imposes the following constrains on all pairs of classes:

$$\frac{q_i}{q_j} = \frac{c_i}{c_j}, \quad i, j = 1,2,...,m \tag{4}$$

where $c_1 < c_2 < \ldots < c_m$ are the generic quality differentiation parameters (QDP).

The proposed model is the extension of the DiffServ Proportional Service PHB, therefore the network handles the flows at aggregation level. In order to ease the discussion of the model we use the following terminology.

Definition 1.2 The *micro flow* is a packet-stream that crosses the network domain from a given ingress (entry point into the network) to an egress (exit point from the network), follows a given path and belongs to the same communication session. The aggregation of all the micro flows using the same path between a given ingress and egress is a *macro flow*.

Note that over a given path *(x)* there are lots of micro flows, each micro flow being part of a macro flow. Over the same path there are at most *m* macro flows, each belonging to a different QoS class. The set of all paths in the network is noted with *P*. The bandwidth of a macro flow is noted by $F_{i,in}$. If not noted otherwise, the term flow should be interpreted as a macro flow.

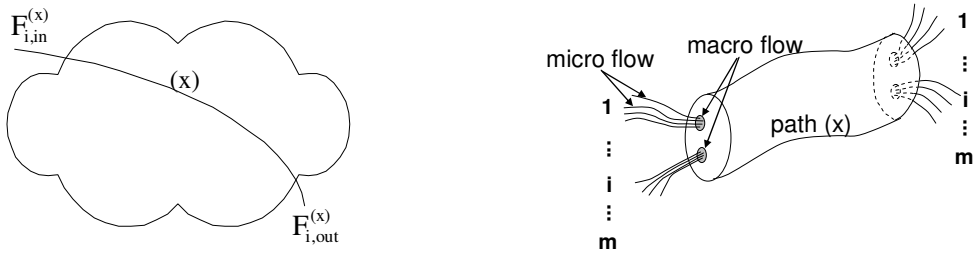

Figure 2. Input and output of a flow (on the left) and micro- and macro flows (on the right)

4.1.1 Quality Differentiation Parameters in the network-wide proportional service

The network-wide proportional services are defined for the goodput performance measure (Quality Differentiation Parameter in the relative service differentiation terminology). Since there are slightly different interpretations of the goodput in the literature, we give the definition of our interpretation below.

Definition 1.3 For a given QoS class *i* the goodput, as quality parameter in the end-to-end proportional services model is G_i

$$q_i = G_i^{(x)} = \frac{F_{i,out}^{(x)}}{F_{i,in}^{(x)}}, \quad i, j = 1,2,...,m \tag{5}$$

where $F_{i,in}$ is the offered load of a flow F_i, the input bandwidth that appears at the edge (ingress) of the domain. Similarly, $F_{i,out}$ is the bandwidth of the same flow F_i, as it leaves the network domain. The upper index *(x)* denotes the path of the flows, as depicted in Figure 2.

This definition can be applied to both responsive and non-responsive flows. The literature gives several options to classify the IP traffic according to its capacity to respond to congestion in network. Two of most used definitions call them responsive and non-responsive flows [71], or refer to them as streaming and elastic traffic [72]. We will use the following terminology in this document.

Definition 1.4 A *responsive flow* is a flow that deploys a congestion control mechanisms, thus modifies its sending rate as a function of observed packet losses (e.g., TCP sessions). A *non-responsive flow* is a flow that does not deploy congestion control (e.g., UDP traffic)**.**

For non-responsive, UDP flows the $F_{i,in}$ *is* the actual number of bytes arrived at the ingress in a unit time.

For responsive, TCP flows the interpretation of $F_{i,in}$ is slightly more complicated, as detailed later in the next Chapter.

We propose the *network-wide proportional service***,** which is a QoS differentiation model applied to *all the flows* of this network and which holds the properties below [C1][C3][J1]. A given class *i* receives the same service differentiation over the whole network domain, thus the per-class QDPs are defined at network level (c_i is the same for all flows from class *i*, irrespective of their path in the network). This service is based on the *goodputs* of the flows and it holds the following equation

$$\frac{q_i}{q_j} = \frac{G_i^{(k)}}{G_j^{(k)}} = \frac{\frac{F_{i,out}^{(k)}}{F_{i,in}^{(k)}}}{\frac{F_{j,out}^{(k)}}{F_{j,in}^{(k)}}} = \frac{c_i}{c_j}, \quad i, j = 1,2,...,m, \quad \forall (k) \in P \tag{6}$$

Note that in this model there is no Call Admission Control – every flow is accepted in the network and it receives the service according to the QoS class it belongs to.

The enforcement of the proposed service is performed once along the path of the flow, that is the QoS shaping and/or policing is performed at the ingress (the entry point of the network).

Although the service is defined for goodputs, in the proposed network architecture the *loss rates* of the flows are automatically also proportionally differentiated. Also note that the network-wide total (including propagation, queuing and processing) delay of the packets of a class is no worse then the delay of the packets of any lower quality class (i.e., if we note with d_i the total delay for class *i*, then $d_j \geq d_i$, *if* $j<i$ $i,j=1,\ldots m$).

We propose a networking architecture that is able to enforce the the *network-wide proportional service* in that network. In this network, in order to function as desired, we need to implement an algorithm that computes the parameters of the network-wide proportional service (as proposed above in eq. (3)). Right now just let us suppose that these algorithms exists, we will propose such algorithms in detail in Chapter 4.

The proposed network architecture holds the following further properties, as follows. It is able to differentiate (classify) the flows based on its paths within the networks and its QoS class. It is able to treat flow rate and number, flow path, link capacities, flow IDs crossing the links as input parameters to the algorithms computing the service parameters. This means that the architecture is able to provide the required input parameters for the algorithm. Further on, it is able to enforce the bandwidth rates at the ingresses, received as the outputs of the algorithms and it is able to separate the handling of the packets of flows with different QoS class. Finally, it is able to support both centralized (at domain level) and distributed algorithms [C1][C3][J1].

The generic network mode of the proposed architecture is presented in Fig. 3. The micro-flows with the same ingress and egress points are aggregated. For each ingress, *a*, and egress, *b*, there are *m* flows, denoted $F_i^{(x)}$. For each flow, *F*, we have the offered load (input bandwidth, F_{in}) at the ingress and the achieved throughput (output bandwidth, F_{out}) at the egress. In Fig. 3 there are two paths presented, *(x)* and *(y)*. Although the two paths

share a common link within the network, the class differentiation is applied "pathwise" – as seen in eq. (3), it defines the relations between flows from *different classes* but the *same path*.

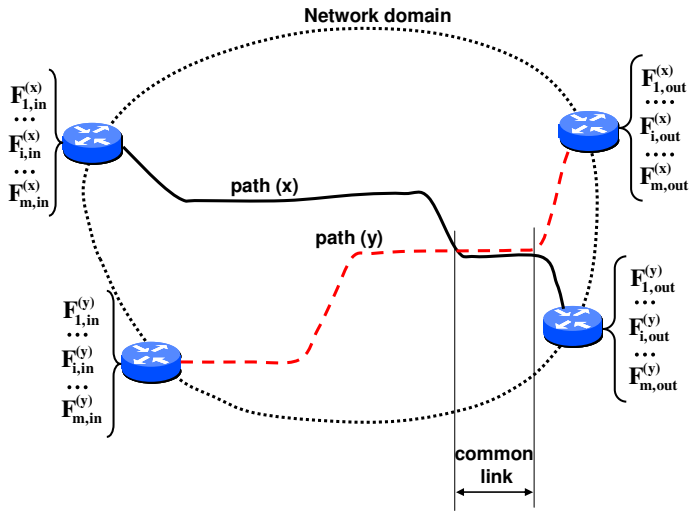

Figure 3. The flows of the network domain

Because in the proportional service model unique QDPs were proposed (QDPs of each class are path independent), the notation can be further simplified, introducing the following notation.

Definition 1-5. In the proposed network wide PropServ model the QDPs can be denoted by $\alpha_{i,j}$, as follows:

$$\frac{G_i^{(k)}}{G_j^{(k)}} = \frac{\frac{F_{i,out}^{(k)}}{F_{i,in}^{(k)}}}{\frac{F_{j,out}^{(k)}}{F_{j,in}^{(k)}}} = \frac{c_i}{c_j} = \alpha_{i,j} \quad i, j = 1,2,...,m, \quad \forall (k) \in P \tag{7}$$

where, c_i and c_j are network wide differentiation parameters. Note that $\alpha_{i,j} > 1$ if i>j.

The notation can be further simplified to $\alpha_i = \alpha_{m,i}$ $\quad i = 1,2,\ldots,m,$

Note that

$$\alpha_{i,j} = \frac{c_i}{c_j} = \frac{\frac{c_i}{c_m}}{\frac{c_j}{c_m}} = \frac{\frac{1}{\alpha_{m,i}}}{\frac{1}{\alpha_{m,j}}} = \frac{\alpha_j}{\alpha_i} \tag{8}$$

and we have: $\alpha_i > 1$ for i = 1,2,…,m-1 and $\alpha_m = 1$.

For the highest (m) and any other QoS class, this equation shows that the goodput of the best quality class (m) is α_i time bigger then the goodput of certain i class.

The novelty of this proposal is that, opposed to the PHBs, it assures this relative differentiation *over a domain, not over a single hop*. Therefore the service received by the user is clearly defined, it is not the result of the combination of per-router traffic manipulation experienced along the path. Note that calculating a priori the effects of independent per-hop processing is not a trivial task. This network architecture the QoS parameters are determined by the algorithms detailed in Chapter 4 before the traffic enters the network. Additional advantage of the proposal is that the behavior of the flow can be better communicated to the user, since it can be described by one parameter.

In the followings we present in detail the architectural details of our proposal for both the centralized and distributed approach. Based on the choice of the network administrators they can select among two implementation alternatives. Although the main principles are the same, there are differences in the building blocks of these two alternatives. In both cases the routers at the edge of the network domain have a distinguished role. The ingress and egress functionalities are different, but in practice they might (and most probably will) be collocated. In both approaches the core routers will have to maintain only information on the aggregate flows, therefore the proposal conforms to what has been called the *core-stateless architecture* [36].

4.1.2 Implementation alternatives – centralized approach

The architecture of a *centralized* network-wide proportional service network is depicted in Fig. 4 [C1][C3]. In this case a new networking element is needed, called Central Broker. In several DiffServ-related work the authors introduced a Bandwidth Broker which is responsible with network-wide traffic engineering tasks [73][74][75]. The noteworthy difference is that while bandwidth brokers are designed with call admission functionality in mind, our architecture does not rely on call admission. Still, bandwidth brokers keep a central view on the traffic load and topology of the network domain and control the marking, shaping and policing processes at the ingresses. The scalability and implementation issues have already been researched by the proposers of the bandwidth broker based solutions. Since the functionalities of these two architectural elements are similar, our central broker can be collocated with the bandwidth broker.

Fig. 4 presents in detail the internal structure of the ingress, core and egress routers, respectively. The ingress router (the leftmost figure) has the task to classify the packets to flows. For this it relies on a database, configured by network administrators (e.g., source-destination pairs, service classes, etc.). Based on the identification the packet is marked, and from now on all the routers of the domain will treat the packet according to this mark. The router deploys a CBQ queuing discipline, with separated FIFO queues for each aggregated flow.

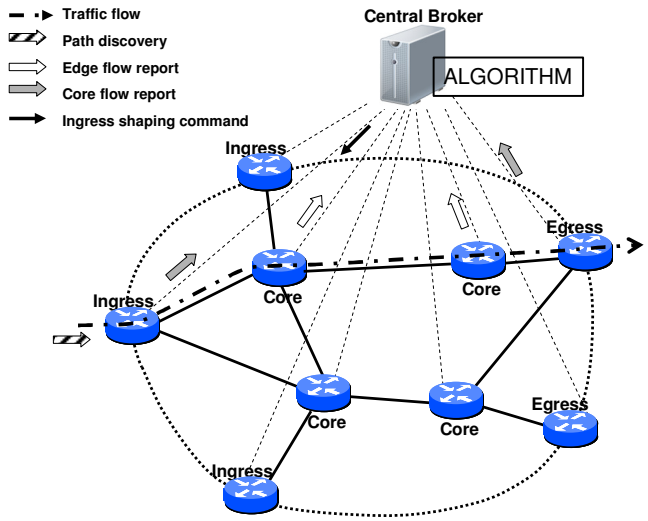

Figure 4. Architectural elements of the network-wide proportional service domain (centralized approach)

The ingress also sends regular reports to the Central Broker. The ingress receives configuration data, according to which it sets the queuing disciplines. Finally, a path discovery module is deployed which maintain path information.

The core router just reports the state of its queues to the central broker and configures its queues according to the received configuration commands. Additionally it takes part in path discovery process, initiated by the ingress.

The egress apart of path discovery and reporting it also has to “free” the packets from the markings (classification data) of the ingress.

The signaling flow is also depicted in Fig. 4. As described above it required reporting from the routers and distribution of the configuration data (commands) to the routers.

The Central Broker collects data from the routers, it keeps the flow data in a database and based on triggers (coming from the ingress) it runs an algorithm that assures that the network operates according to the proportional service model. Finally it sends the necessary configuration data to the routers.

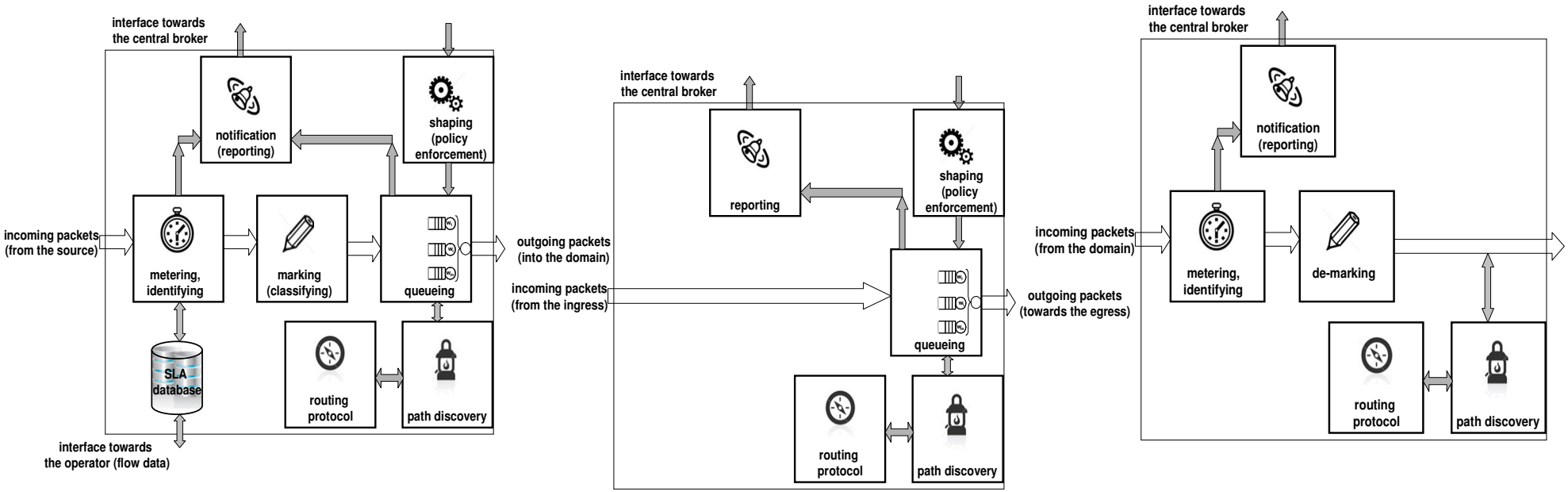

Figure 5. Functional blocks of the routers (centralized approach)

Information flow in the centralized approach

The *Central Broker* must acquire a number of information from the network nodes. The following pieces of information are required for the algorithm:

1. Topology information: the connectivity matrix, interface speed information (bandwidths)
2. Load information: offered load at ingress and egress
3. State information: queue lengths, flow route information

The acquisition of this information is not straightforward, due to the dynamic behavior of the required information. In order to seize the problem we make the following assumptions:

Topology information: The connectivity matrix is initialized when the Central Broker is configured, and continuously updated through LSAs. The Central Broker will execute the algorithm using the above-mentioned topology and routing information.

Load information: The traffic matrix is constructed from the offered load of the incoming flows, divided according to the egresses they leave the network. Thus ingress and egress routers will send traffic-information to the Central Broker, the traffic matrix being updated when a new flow is entering the network.

The ingress routers will report to the Central Broker the characteristics of the flow. There are two possibilities to assess the offered load of the flows. The easiest method would be to have a flow-descriptor at the ingresses.

Apart of the above ideas, much more detailed solutions can be applied to collect the necessary information. Our review in section 3.2 provides a detailed discussion of the available alternative solutions.

In a general case we do not have any prior knowledge about the traffic characteristics. Thus we must *predict* the traffic in some way. Our suggestion is to divide the time in timeslots (of order of minutes), and predict the traffic in the N+1th slot based on the traffic of previous N slots and a daily profile. Due to the strong seasonality in traffic data (e.g. lower utilization on Sundays, holidays, so on) a network profile enhances a lot the accuracy of the prediction.

Traffic prediction is a vast research area, thus it can not be addressed in the frame of this dissertation. However, we may use the results of previous research, as detailed in section 3.2**.**

State information: We consider that one flow is always conveyed along one route, which we also call the path of the respective flow. This is easy to implement considering static routing, but dynamic routing can be followed, as well (e.g. in OSPF the LSAs can reach the Central Broker, allowing it to build the topology and run the Dijkstra algorithm in parallel).

For the construction of the traffic matrix we can choose from the following methods:

1. *measurement-based route discovery* is an other option. We collect at each node the information regarding the forwarded flows. Each node must be modified in the domain to observe and report the flows traversing it, and even so we still have

problems: the source address usually does not coincide with the ingress router's address, nor the destination address with the egress' one.

2. supposing that in the network the OSPF routing protocol (or any other link-state based routing protocol) is used, we can collect the routing information from the OSPF routing tables. When other routing protocols are used, *extraction of routing-information* needs some special workaround. Since link state routing protocols are widespread and QoS routing techniques also favor these protocols, we may require that this service *should* use OSPF as a routing protocol.
3. in order to assure total independence from the underlying routing protocol, the Central Broker should use a *path discovery* prior the flow's admission into the network.

The second solution will be used in our simulations, as the easiest-to-test solution. However, the third method is explained below, since it is the universal solution. In second stages of simulations we intend to compare against the 2nd solution.

When a new path is introduced, the path discovery procedure will be initiated. The destination address from the connection request is used in a *traceroute* like message. This message will traverse the same route what will be used for the admitted flow into the network. As already mentioned above, these three different concepts are basic concepts used in our research. From our point of view these are used for the proof of our concepts and they guided the design of our simulation tool. Nevertheless, several implementation alternatives for such solution, proving its feasibility are available in the literature, as detailed in section 3.2.

The address of the routers traversed by the path discovery message will be recorded, thus at the egress router we will obtain the path of the respective flow. The egress routers will transmit the path discovery messages to the Central Broker. Thus, the Central Broker will be able to always keep an update.

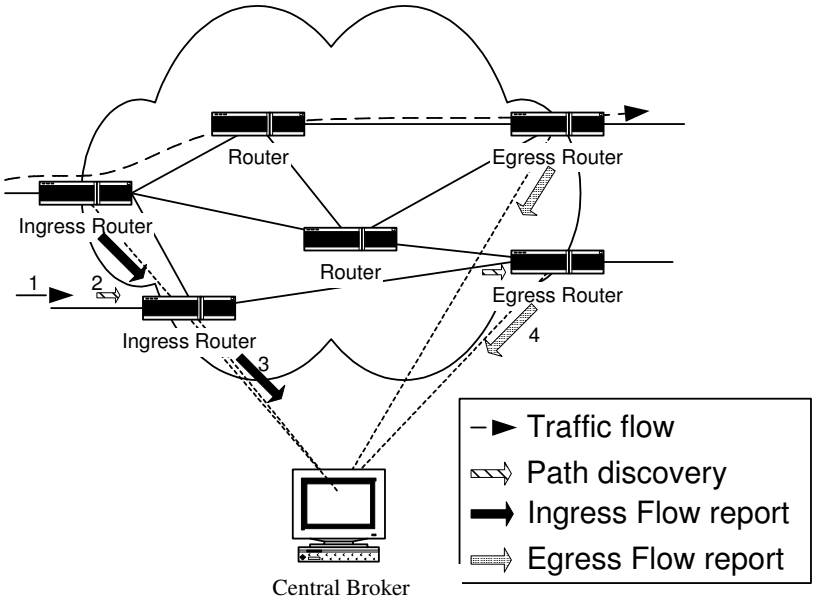

Figure 6. Updating the Central Broker

After the initial topology discovery the Central Broker creates the topology database with link capacity information. Then, edge routers send *Ingress* and *Egress Flow reports* to the ISP, which will build the traffic matrix. The intermediate routers will send flow information to the IPS when they receive a path discovery message.

The algorithm uses the acquired information to locate the network bottlenecks. The search for bottlenecks is an iterative process. When we have the bottleneck information for each incoming flow, we compute the weight information using the equations.

Finally the weight information update is sent to the routers.

4.1.3 Implementation alternatives – distributed approach

The architecture of a *distributed* network-wide proportional service network is depicted in Fig.7 [C1][C3]. The distributed version of the network architecture is simpler, since there is no need for the Central Broker. On the other hand the functionalities of the Central Broker are distributed among the routers in the network. The signaling data is transmitted along the path of a flow and there is no communication towards a centralized entity.

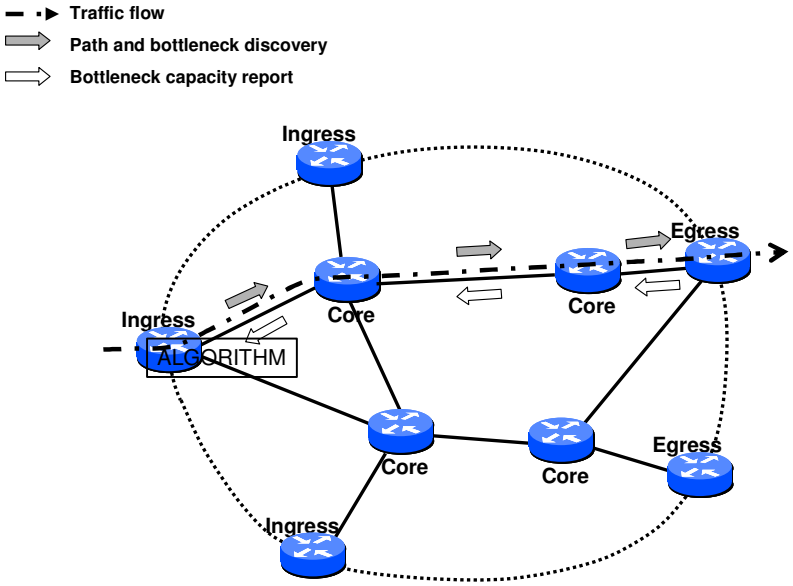

Figure 7. Architectural elements of the network-wide proportional service domain (distributed approach)

The routers inherit most of the functions from the centralized approach and their structure are depicted in Fig.8.

The ingress router will have to run the algorithm, and will have to distribute configuration data. On the other hand, it will have to take care only about those flows that enter the network there, which means that the computational burden will be significantly lower.

The difference at the core router is that it has to keep some information about the aggregated flows, therefore a new database should be set up.

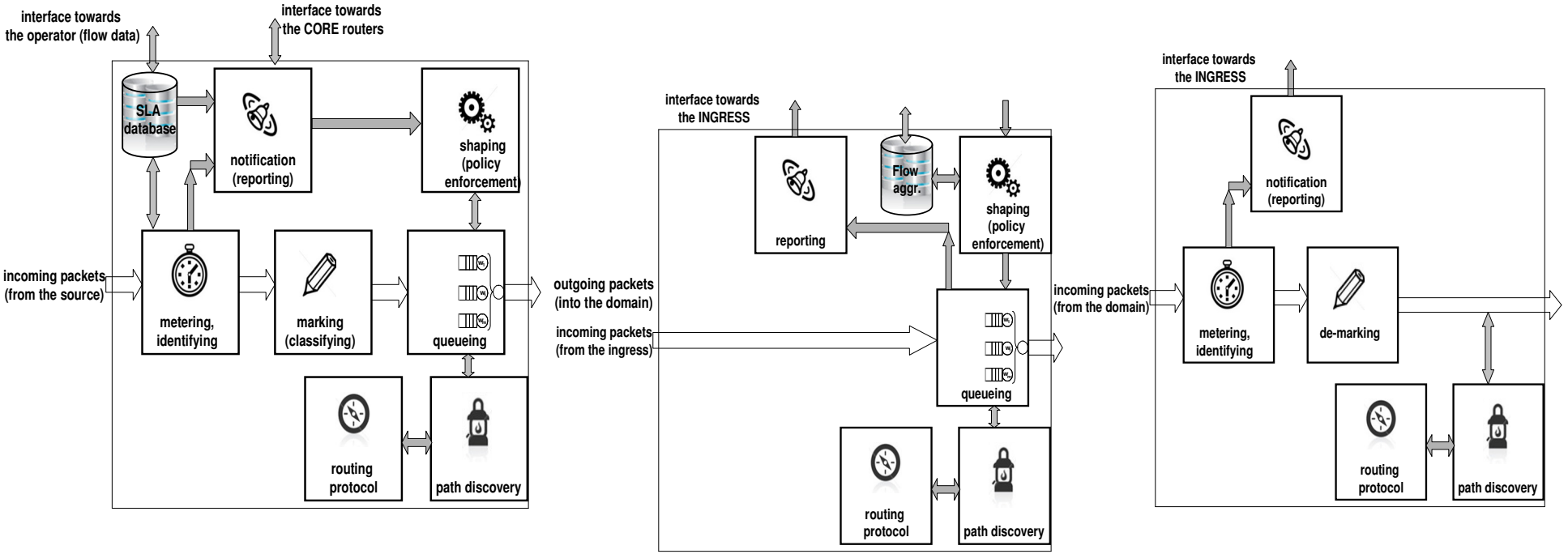

Figure 8. Functional blocks of the routers (distributed approach)

The egress is practically has no different role, the only change is that it has to report to the ingress, not the Central Broker.

In a classical IP network, when congested, the flows loose packets and suffer various delays at the intermediate (core) routers resulting in a reduced throughput at the egress. The achieved throughput is the result of a complex interaction among flows crossing the same links inside the networks, thus it is hard to predict. However, if this network deploys the network-wide proportional service model the bandwidth of a flow at the egress of the domain depends on the traffic matrix (the $F_{i,in}^{(x)}$ offered loads) and the QDPs, as conditioned by the eq. (7). The open question is whether the *throughput* (the capacity share of the flow in its bottleneck) can be analytically determined. If yes, it can be later used in algorithms to police the traffic at the ingress and reduce the load inside the network.

Information flow in the distributed algorithm approach

Most of the algorithms can be implemented also in a distributed manner. Some algorithms such as the algorithm with global optimization cannot be implemented as distributed since it would generate high signaling traffic. But the simple shaping algorithms can work in distributed manner as well.

The distributed algorithms require the following information for proper functioning:

Local *topology information*: the connectivity matrix of the node, interface speed information (bandwidths). The connectivity matrix is initialized when the node is configured. This matrix will be changed only if new interface is added or a link is removed.

Load information: offered load at ingress and egress. The traffic matrix is constructed from the offered load of the incoming flows. The egress capacity will be discovered using, signaling among nodes along the path.

State information: In this case we do not need route information. We consider that one flow is always conveyed along one route, which we also call the path of the respective flow. The path of the flow is discovered during the communication between nodes along the path. The only information required about the path is the next hop information. This information can easily obtained from the routing process or from the static routing table. The information to be stored in core routers is the flows (not *micro* flows!), for reasons detailed at the description of the distributed algorithm.

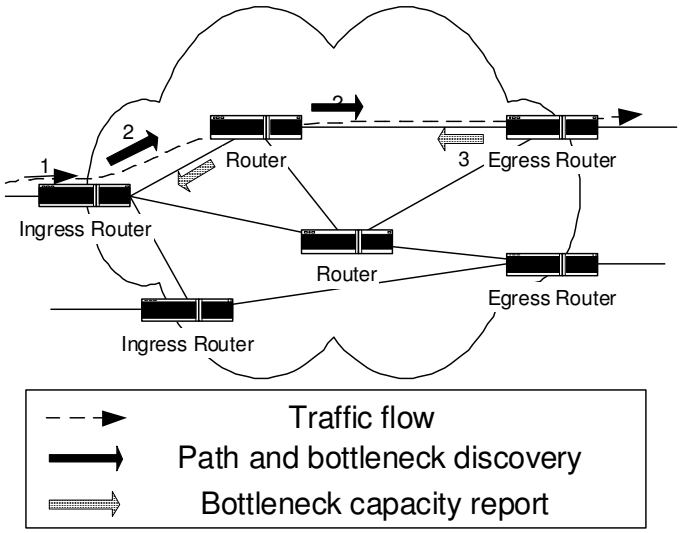

Figure 9. Updating the ingress routers

First of all, the algorithm must be triggered by change in network status. This trigger can be a change in offered load, internal structure change or simply a timer event. The most obvious trigger is the change in offered load for a flow.

4.2 Analytical Analysis of the end-to-end proportional services

The above framework proposes a model to provide proportional services in a network. We have set forth the QDPs the model should conform to, but as for now we can only evaluate ex post if these conditions were met by a network. In what follows we try to analytically analyze the behavior of the system. Given the input traffic matrix, we try to predict the throughputs that should be achieved in order to conform to the model [C1][C3].

4.2.1 The throughputs of the flows in the network-wide proportional service model

Goodputs in the network-wide proportional services model

Given a network that deploys a network-wide proportional service model, as depicted in Fig.3, it can be written the eq. *(7)* for all the paths that share the common bottleneck link l_b:

$$\left\{ F_{i,out}^{(z)} = \alpha_{i,j} F_{i,in}^{(z)} \frac{F_{j,out}^{(z)}}{F_{j,in}^{(z)}} \right\} \; i,j = 1,2,\ldots,m \text{ and } \forall (z) : l_b \in (z) \qquad (9)$$

The problem is that this set of equations is not fully independent, for every path there are only *m-1* independent equations. On the other hand, since the flows cross a bottleneck, it can be written the work conserving law for the bottleneck link:

$$\sum_{\forall (z): l_b \in (z)} \sum_{i=1}^{m} F_{i,out}^{(z)} = \mu_{l_b} \qquad (10)$$

Now, if we note with R the total number of paths that cross the bottleneck link l_b

$$R = |\,\forall(z) : l_b \in (z)\,|$$

there are $R(m-1) + 1$ equations, but Rm unknown variables, thus the resulting system of equations still can not be solved.

Eq. *(9)* defines the relation among different priority classes *within the same flow path*. It says nothing about what happens when two or more flows from the same priority class but from different paths share the same link(s).

It is a natural requirement that flows from the same class should share the common resources equally (in other words the same class flows are not differentiated). This desirable property can be considered a *fairness criterion* which specifies the relation among *concurrent flows*.

Definition 1-6. Fairness criterion. Let flows from the same class i but different paths (z) sharing the same bottleneck link have the same goodput

$$\frac{F_{i,out}^{(z)}}{F_{i,in}^{(z)}} = \frac{F_{i,out}^{(y)}}{F_{i,in}^{(y)}},\ \forall(z) : l_b \in (z), \forall(y) : l_b \in (y), i = 1,..., m \qquad (11)$$

(Note that this criterion is reasonable when the whole network domain is considered and the same c_i QDPs are used over the whole domain.)

For convenience let us order the R paths crossing the bottleneck link l_b. Now we have *(1), (2), …, (R)* paths. Furthermore, without losing the generality of our notation, we consider the *(z-1)* path for $z=1$ to be the same as *(R)*. Following this notation and using eq. *(11)*, it can be shown that over a bottleneck link l_b

$$F_{i,out}^{(z)} = F_{i,out}^{(z-1)} \frac{F_{i,in}^{(z)}}{F_{i,in}^{(z-1)}},\ \forall(z) : l_b \in (z) \qquad (12)$$

if, for convenience, for $z = 1$ we consider $z-1 = R$. Note that, similar to the logic followed at eq. *(9)*, this equation yields only $R-1$ independent equations.

Based on eq. *(12)* and selecting an arbitrary QoS class i, $R-1$ new equations can be added to the system of equations. Now there are $R(m-1) + 1 + R-1 = Rm$ independent equations and Rm unknown variables $\{ F_{i,out}^{(z)} \}_{i=1,...,m,\ \forall(z) : l_b \in (z)}$.

The above system of equation can be solved according to the logic as follows. First we insert the right hand side of eq. *(9)* into eq. *(10)*. Then we insert the right hand side of eq. *(12)* in the resulting equation. After we observe that $F_{i,out}^{(z)}$ can be pulled outside the summation sign and we re-arrange the equation, we get the following formula:

$$F_{i,out}^{(z)} = \frac{\mu_{l_b}}{\sum_{\forall(y):l_b \in (y)} \left(\frac{F_{i,in}^{(y)}}{F_{i,in}^{(z)}} \sum_{j=1}^{m} \alpha_{j,i} \frac{F_{j,in}^{(y)}}{F_{i,in}^{(y)}}\right)} \qquad (13)$$

Note that eq. (13) expresses the final throughput of a flow as a function of the QDPs, the elements of the traffic matrix (i.e., the offered loads) and the capacity of the bottleneck link.

Loss rates in the network-wide proportional service model

In the case of the differentiation of classed based on *loss rates (*loss differentiation) or *delay* (delay differentiation) the QoS service parameters are the inverse of of the loss rates or ingress-to-egress delays. The reason behind this is that the better the QoS of a class, the lower the loss rate (or delay) should be, whereas for the goodput the better service means higher goodput. In order to assure a unique framework we keep the definition of better classes from Def. 1-1., we use the following notations.

Definition 1-7. The QoS parameter for a loss-differentiation is

$q_i = \frac{1}{l_i}$, where l_i is the loss-rate of flow i experienced from ingress-to-egress.

Similarly

$q_i = \frac{1}{d_i}$, where d_i is the ingress-to-egress delay of flow i.

Similarly to Def. 1-5 we can simplify the notation for the *loss-differentiation* if we introduce the differentiation parameter, as follows.

$$\beta_i = \frac{q_m}{q_i} = \frac{l_i}{l_1} \tag{14}$$

Starting from eq. (13) and knowing that the loss rate over a path *(x)* for class *i* is

$$l_i^{(x)} = \frac{F_{i,in}^{(x)} - F_{i,out}^{(x)}}{F_{i,in}^{(x)}} \tag{15}$$

$$\frac{q_i}{q_m} = \frac{c_i}{c_m} = \frac{l_m}{l_i} = \frac{\frac{F_{m,in} - F_{m,out}}{F_{m,in}}}{\frac{F_{i,in} - F_{i,out}}{F_{i,in}}} = \frac{1 - \frac{F_{m,out}}{F_{m,in}}}{1 - \frac{F_{i,out}}{F_{i,in}}} =: \frac{1 - G_m}{1 - G_i} = \frac{1}{\beta_i} \tag{16}$$

Thus the $\beta_i^{(x)}$ differentiation parameter for the loss rates for a given class *i* over path *(x)* is:

$$\beta_i^{(z)} = \frac{1 - \frac{G_m^{(x)}}{\alpha_i}}{1 - G_m^{(x)}} \tag{17}$$

Remember from Def.1.4 that $G_m^{(x)}$ is the goodput of the best QoS class flow over path *(x)*.

By definition we have $G_m^{(x)} < 1$, $\alpha_i > 1$, thus $\frac{G_m^{(x)}}{\alpha_i} < G_m^{(x)}$, and it results

$$\beta_i^{(x)} = \frac{1 - \frac{G_m^{(x)}}{\alpha_i}}{1 - G_m^{(x)}} > 1 \qquad (18)$$

This also means that if $\alpha_i > \alpha_j$, then $\frac{G_m^{(x)}}{\alpha_i} < \frac{G_m^{(x)}}{\alpha_j}$, and it results that $\beta_i^{(x)} > \beta_j^{(x)} \quad \forall (x) \in P$.

This means that if we order the classes based on the goodput based QDPs, then the *loss rate* in a class will be less then to the loss rate in any worse (having lower QDP) class.

Packet delays in the network-wide proportional service model

If the distribution of the arrival rates of the flows in different classes are the same, then the delays experienced by the packets of the flows from a higher class are no worse (lower) than those experienced by the packets from a lower one. Opposite to the goodput (capacity) and loss differentiation, the delay differentiation also depends on the load of the queues at the ingress.

We base our argumentation on the fact that the buffer sizes in the core routers are small and therefore they are only used to buffer out the effect of short term bursts. Therefore the delays experienced within the network (the queueing and processing delays at the core routers and the propagation delays along the links of the core) are the same.

$$d_i^{(x)} = d_{i,ingress}^{(x)} + d_{i,core}^{(x)} \qquad (19)$$

$$d_{i,core}^{(x)} = \sum_{l_i \in (x)} (d_{i,link}^{(x)} + d_{i,queue}^{(x)} + d_{i,processing}^{(x)}) \qquad (20)$$

The difference in delay is due to the queueing delays experienced at the ingress. On the other hand in a CBQ discipline, the load of the queue of a better class is lower than in queue of a lower class.

Therefore for the delay based QoS service parameters we have

$$q_m \geq \ldots \geq q_2 \geq q_1 \qquad (21)$$

where the equation holds for the case where ingress queues are empty.

4.2.2 Link Utilization

Before we generalize, let us take an illustrative example, where m = 2. We use this low value for *m* to explain in an illustrative manner the details of the proposed discipline. If we have only two service classes in our system, we may call the higher class of service as the premium one, while the lower service class is the normal or regular one. For easier understanding, when we will use throughout the document the m = 2 case, we will note F_2 with *H* (*H*igher class of service) and F_1 with *L* (*L*ower class of service). Thus we have

only two flows, a high and a low priority one and both arrive to router *R*. The flows leave the router at the output link with capacity C. This configuration is presented in Figure 10.

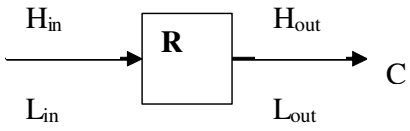

Figure 10. Flows H and L crossing router R

Bottleneck link: in the 'normal' case $H_{in} + L_{in} > C$, thus we have a bottleneck situation.

Excess bandwidth: However, if the offered load is not big enough the flows entering a router R have a total offered load less than the capacity of the output link. In this case we have $H_{in} + L_{in} < C$.

Loose bottlenecks: The third case is, when the bottleneck condition case holds

$H_{in} + L_{in} > C$,

but if we apply our formula (22),the computed H and L values will underutilize the output link:

$H_{out} + L_{out} < C$.

Consider flows H and L enter router R, and we apply the usual formulas. However, if we fully utilize bandwidth C, then we will need more high-class input and less low-class output to meet the eq. (6) condition. E.g., let us set the following values for the network in Fig. 5: H_{in}= 80, L_{in}=30 and C=100, $\alpha = 2$.

In this case, applying the eq. (6) formula we have

$$\frac{H_{out}^{A}}{H_{in}^{A}} = 2 \cdot \frac{L_{out}^{A}}{L_{in}^{A}} \tag{23}$$

and the relation $H_{out} + L_{out} = C = 100$,

we get: H_{out}= 84.2 and for L_{out}=15.8.

We can see that applying the above formulas we get a higher value for H_{out} than H_{in}. Because this is impossible, we will have at the output link $H_{out} = H_{in}$ and L_{out} as computed above. Thus the output link will be under-utilized, $H_{out} + L_{out} < C$.

We generalize this little example presented for the loose bottleneck situation by introducing the term shaped traffic demands as follows.

Definition 1-8 Shaped traffic demands. The shaped traffic demands are the flow outputs in the hypothetic case when the bottleneck link capacity allows the highest class flow to be carried and the rest of the classes are just shaped to assure that the service differentiation between classes. We note these values with $\overline{F}_{i,out}^{(z)}$:

$$\overline{F}_{i,out}^{(z)} = \frac{F_{i,in}^{(z)}}{\alpha_i}, \; i=1,\ldots, m \text{ and } z=1,\ldots,L \tag{24}$$

Note that for $i=m$ $\alpha_i=1$, thus $\overline{F}^{(z)}_{m,out} = F^{(z)}_{m,in}$, as stated in the definition.

These shaped traffic demands are a good starting point to shape the traffic at edge of the network, in order to achieve network-wide proportional service and it will be used in several algorithms, as detailed in the next Chapter.

Chapter 5 Algorithms and Evaluation

We have analyzed the network-wide proportional service, and we can give a closed form formula for the throughputs of the flows if we have the traffic matrix. First we show that applying these formulas to flows in an arbitrary manner does not result in the desired behavior. Therefore later in this Chapter we propose several algorithms that compute and assure that, based on these values the flows will conform to the service. The performance of each of the algorithms is analyzed by means of simulations.

5.1.1 Naïve approach

Consider a network ingress with a flow and two classes ($m = 2$). The proportional differentiation can be obtained by "shaping" in a first step intentionally *only* the *L* class packets according to the *shaped traffic demands* eq. *(24)*, and then in the second step directing both *H* and *L* class packets in a FIFO queue. We assume that in the FIFO queue both the *H* class and the "shaped" *L* class flow will be handled/dropped in the same way.

Considering that the incoming traffic is still larger than the available bandwidth, packets will be dropped with the same probability for all of the flows. The basic idea is to differentiate once, at ingress, then use FIFO queues in the core. This basic algorithm is a simple one. At the ingress point of the network the desired ratio will be set by dropping the low priority traffic to assure the $\alpha_{1,2}$ ratio between the H and L classes (setting the *shaped traffic demands*). This means that we drop packets even if the network can transport the full traffic.

Naïve Algorithm

At the core nodes we simply use FIFO queuing and therefore we suppose that the ratio will not be changed. According to our simulations for UDP flows in Appendix A, this method should work fine with non-reactive flows, such as UDP. Therefore we tested this naïve algorithm by means of simulation. For border routers, at ingress points of the network the flowchart diagram of the naïve algorithm is presented in Figure 11. For network core routers, we do not use any algorithm jut a simple FIFO queue.

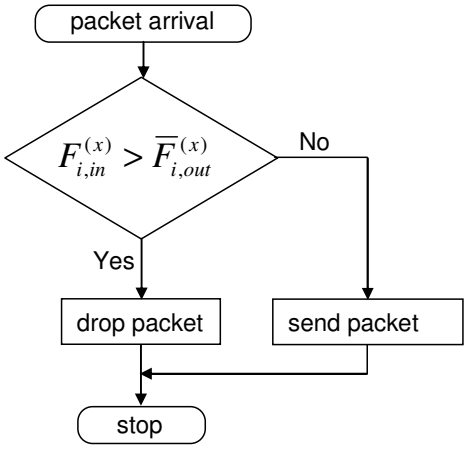

Figure 11. The naïve algorithm at the ingress router

The problem is that the algorithm works well only if in the network there is one single bottleneck for all the flows of the network. If we have distributed bottlenecks, which is quite probable in a real life scenario, the algorithm can not keep the desired $\alpha_{1,2}$ ratio.

We show a typical result achieved with the naïve algorithm in the following Figure 12. We wanted to keep $\alpha_{1,2} = 3$ in the four flows scenario with variable traffic (for network topology and simulation details see Appendix C). The graph shows the achieved $\alpha_{1,2}$ ratio for the four flows as function of simulation time. We can see that the naïve algorithm even if theoretically should work, gives an unsatisfactory result.

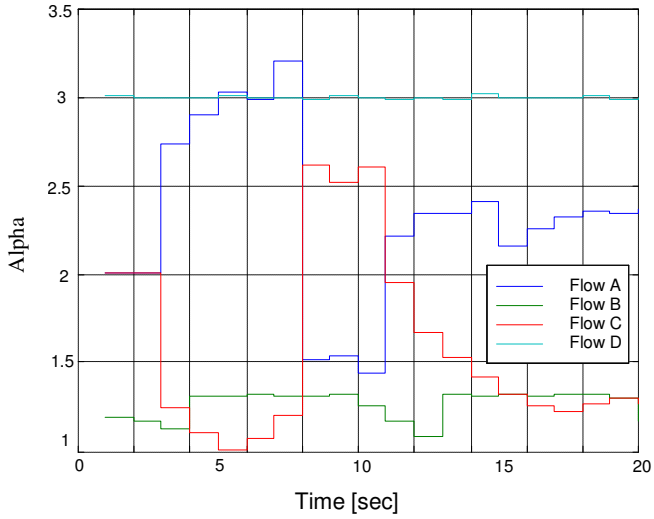

Figure 12. Achieved alpha parameters of the naive algorithm

As shown in this section, the natural approach which results in trivial methods to achieve network wide proportional services in a network do not yield satisfactory results. That is the reason why we propose two different algorithms that would achieve this goal, as presented later in this Chapter.

5.1.2 Domain model and notations

Before moving on, we introduce the notations used for describing the algorithms.

Let us consider a graph G
$G(V, E)$
$V = v_1, \ldots, v_{Vmax}$ – are the vertices (routers) in the domain and
$Vmax = |V|$ - the total number of vertices
$E = l_1,\ldots l_L$ - are the edges (links) in the domain
$L=|E|$ - the total number of edges

We define two subsets of the vertices, the ingresses (entry points of the domain) and the egresses (exit points of the domain).
$IN = \{IN_i\}$ - where IN_i is the i^{th} ingress, $i=1, \ldots, IN_{INmax}$ and $INmax=|IN|$ - the total number of ingresses

$EG = \{EG_i\}$ - where EG_i is the i^{th} egress, $i=1, \ldots, EG_{EGmax}$ and $EGmax=|EG|$ - the total number of egresses
$IN \subset V, EG \subset V$

Note that $IN \cap EG$ is not necessary an empty set.
In modern day (core) networks the connections among routers are bidirectional, and the typically deployed optical fibers use different (physically separated) threads for the two directions. Also, the internal architecture of the routers is such as the incoming and outgoing traffic is handled in different queues. Therefore the same router may act as ingress for one session/packet flow, and as egress for a different one.

A route, the series of routers a packet (of a flow) is traversing as crosses from an ingress to an egress, is a subset of V. Formally we define the route r_{ij} from ingress IN_i to egress EG_j and K hops long as:
$r_{ij} = \{w_i | w_i \in V, i=1,\ldots,K,K+1 \exists E_i$ connecting (w_i,w_{i+1}) and $w_1 = IN_i$, $w_{K+1}=EG_j$ and each w_i is visited only once$\}$
Note that there are multiple routes possible between the same ingress and egress, then we identify the routes in their upper index $r^1_{ij}, r^2_{ij}, \ldots$.

Since in our model the bottleneck links are playing an important role (as already can be seen in section 4.2.2, the capacity of bottleneck link influences the throughput of the flows), for us it is much more practical to use the notion of path instead of routes.
A path (p) is defined as an ordered list of links from and ingress IN_i to an egress EG_j of the domain and each link is visited once:
$(p) = \{l_i | l_i \in E, i=1,\ldots,K$ and l_1 is connected to IN_i , l_K is connected to EG_j, each l_i is visited only once$\}$
Note that the above definition allows that there are multiple paths connecting the same ingress, egress pair.
The active path of a domain is a set formed by that set of paths, which are in "use", i.e., there are packets in the domain that visit the links of (p) in the order of its links:
$P = \{(p) \mid \forall IN_i \in IN, \forall EG_j \in EG, \exists (p) : IN_i \rightarrow EG_j, (p) is\_in\_use\}$
$N = |P|$ - is the total number of active paths in the domain.
An aggregated flow (or flow) is defined by its class i, its path (x) and its offered load $F^{(x)}_{i,in}$.

5.1.3 Distributed Algorithm

We have proposed a distributed algorithm that shapes the traffic at the ingress in such a way that the flows of the network exhibit the properties of a network-wide proportional services model [C1][C3]. The flowchart diagram of the algorithm is depicted in Figure 13. The algorithm keeps track the bottlenecks for each ingress over the paths originating from that ingress. Then it computes the shares of each flow crossing the bottleneck, using eq.*(24).* We have shown with extensive simulations that this algorithm deployed in a distributed manner (fitting the distributed network architecture proposed in section 4.1.3) enforces the flows to fulfill eq. *(6).*

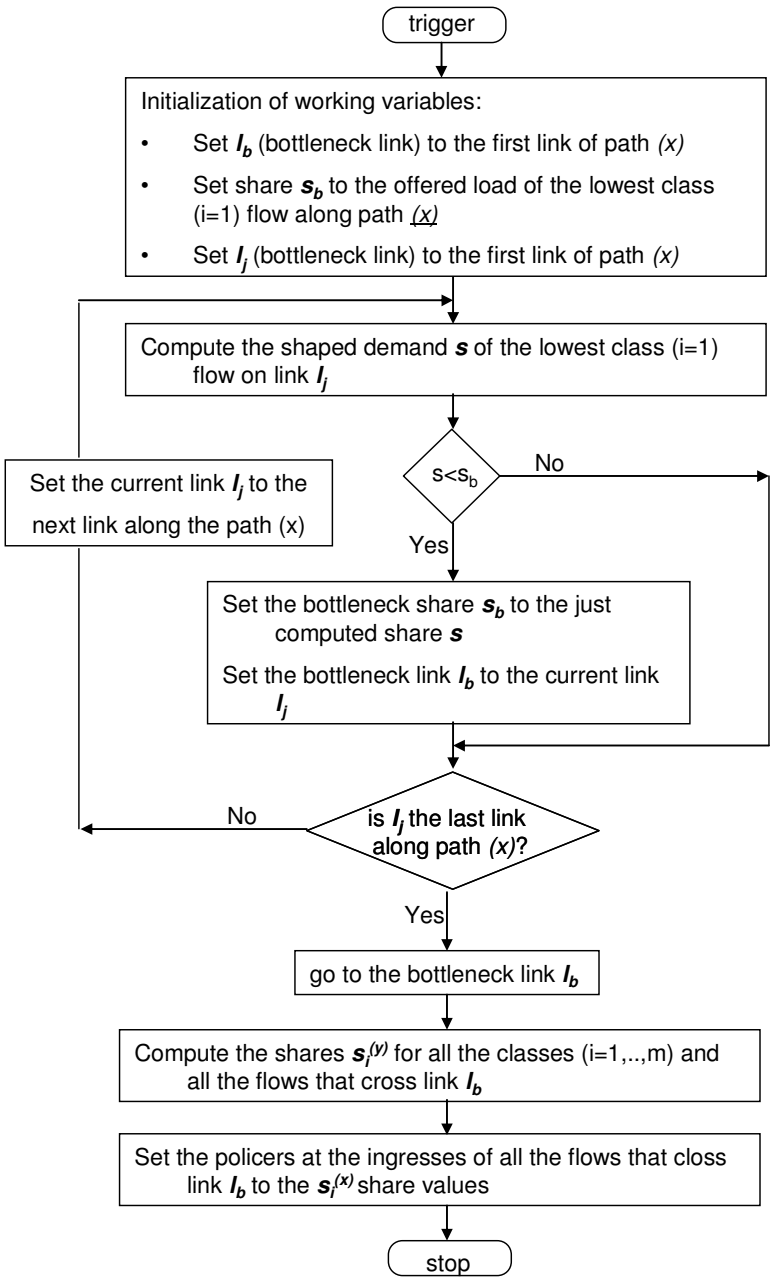

Figure 13. Flowchart diagram of the distributed algorithm

The trigger that starts the algorithm is a change in offered load (interrupt generated by measurement unit at ingress) and the bottleneck capacity is set to the offered load. Then, the algorithm computes the flow's share on the outgoing link. The share is determined using the analytical result from eq.*(6)*.

If the share on the link is less than the offered load, this share will be the flow's new bottleneck capacity. Then the algorithm moves to the next hop in the route, and computes the share again on the outgoing link. If the share is less than the previous one, the

algorithm changes the bottleneck capacity to the new value, and steps to the following node.

This procedure is repeated until the egress node is reached. When arrived to the egress node we have the bottleneck capacity of the whole path. Now from the bottleneck node (the one the bottleneck link is originating from) can compute the shares of all the flows crossing the bottleneck link and notify the ingress node to set its policers to enforce these computed throughputs. The intermediate nodes update the flow bottleneck information as well, reducing the share at bottlenecks accordingly.

In the following we present the algorithm in detail, using the notation introduced in section 4.3.2. The algorithm takes the path of the flow that triggered it, and investigates all the links of this path.

Let us note

$h_j^{(x)}$ = { the j^{th} link counted from the ingress in *(x)* }

and u_j is the capacity of this link

$$u_j = \mu_{h_j^{(x)}}$$

STEP 0

We use this step to initialize our variables.

j=1 (the first link along path *(x)*)

i=1 (the worst class)

Note that the worst class always suffers the worst conditions among all the flows over a given path, therefore it suffices to compare these flows.

$$\bar{s}_i = F_{i,in}^{(x)}$$

STEP 1

Compute the bandwidth share of flow class *i* over path *(x)*, as if this link were a bottleneck. Use eq. *(13)* from to do this.

$$s_i^{(x)} = f\left(\alpha_k \mid k = 1,...,m \;\; ; \;\; u_j \;\; ; \;\; F_{k,in}^{(y)} \mid k = 1,...,m, \forall (y) : h_j^{(x)} \in (y) \right)$$

$$s^{(x)}_i = F_{i,out}^{(x)} = \frac{u_j}{\sum_{\forall (y): h_j^{(x)} \in (y)} \left(\frac{F_{i,in}^{(y)}}{F_{i,in}^{(x)}} \sum_{g=1}^{m} \alpha_{g,i} \frac{F_{g,in}^{(y)}}{F_{i,in}^{(y)}} \right)}$$

Then take the index of that link, where the smallest share value was computed (i.e. the tightest bottleneck for the flow) and store it in *q*.

If $s_i^{(x)} < \bar{s}_i$, *then* $\bar{s}_i = s_i^{(x)}$, *q=j*

j=j+1

If *j* >|*(x)*| then go to **STEP 2,** where |*(x)*| denotes the length of the path.

Go to **STEP 1**

STEP 2

Now we have the bottleneck link over path (x), link $h_q^{(x)}$.

Compute the share values for *all* the flows that cross that bottleneck link:

$$F_{i,out}^{(z)} = \frac{u_q}{\sum_{\forall(y):h_q^{(x)}\in(y)}\left(\frac{F_{i,in}^{(y)}}{F_{i,in}^{(z)}}\sum_{g=1}^{m}\frac{\alpha_g}{\alpha_i}\frac{F_{g,in}^{(y)}}{F_{i,in}^{(y)}}\right)}, \text{ for } i=1,\ldots,m \text{ and } \forall(z): h_q^{(x)} \in (z)$$

Set the policers in the ingresses of these flows to the throughput values computed above.

Evaluation of the distributed algorithm

As we already mentioned in the beginning of section 5.1.1, we in our simulations we will use two classes within each flow ($m = 2$). This helps us to better illustrate and explain our results. Note that $\alpha_2 = 1$, thus there will be only one α that differs from 1, namely α_1. In our discussions on the simulation results we will refer to this α_1 as alpha or α. Throughout our simulations we used several values for α. We used $\alpha = 3$ for our initial investigations. This value implies heavy differentiation among flows. We also used $\alpha = 2$, which can be more intuitively communicated to the users (e.g., "twice as better than…"). We also run our simulations with a value close to one, $\alpha = 1.1$. We do not expect that operator would select such a low value, as this would results in hardly noticeable improvements for the subscribers of the better class. Nevertheless, we will illustrate our evaluation using this value because it tests the behavior of our approach under worse case conditions. Also, once a differentiation is available at this granularity, the operators will be able to create a "menu" for their customers with practically any differentiation parameter by the selection of the appropriate classes.

We have implemented three network topologies with increasing complexity to test our proposals, as presented in detail in Appendix C. Each scenario has more and more routers, links and flows. Also, these scenarios model typical backbone networks (e.g. simpler metropolitan, sparse and dense national/international backbones), as explained in Appendix C. Still, throughout the Dissertation we will present the charts of the representative results obtained only on the simplest one, because even in this case the results sometimes are hard to follow. The results obtained on all the networks are summarized and the numerical results presented in tables in a comparative manner.

The results are presented for time intervals of 1 second. The general proportional service states that the service should guarantee the differentiation on short timescales. The regular timescales in service level agreements are usually at order of magnitude of minutes, if not hours. Compare this with the scale of 1 second imposed by us during our investigations. Our reason for such short timescale was similar to the one at selecting so low value for the target α. If the service is working properly over such a short timescales, then it will surely work over larger timescales, as well.

Figure 14 shows the simulation results for the simple constant load scenario. We zoomed in around the target α value (on the y axis) for a better understanding. It can be seen that the results are as predicted by the algorithm. The regular deviations from the target α value are small and even they are smoothed out at timescales larger with one order of magnitude.

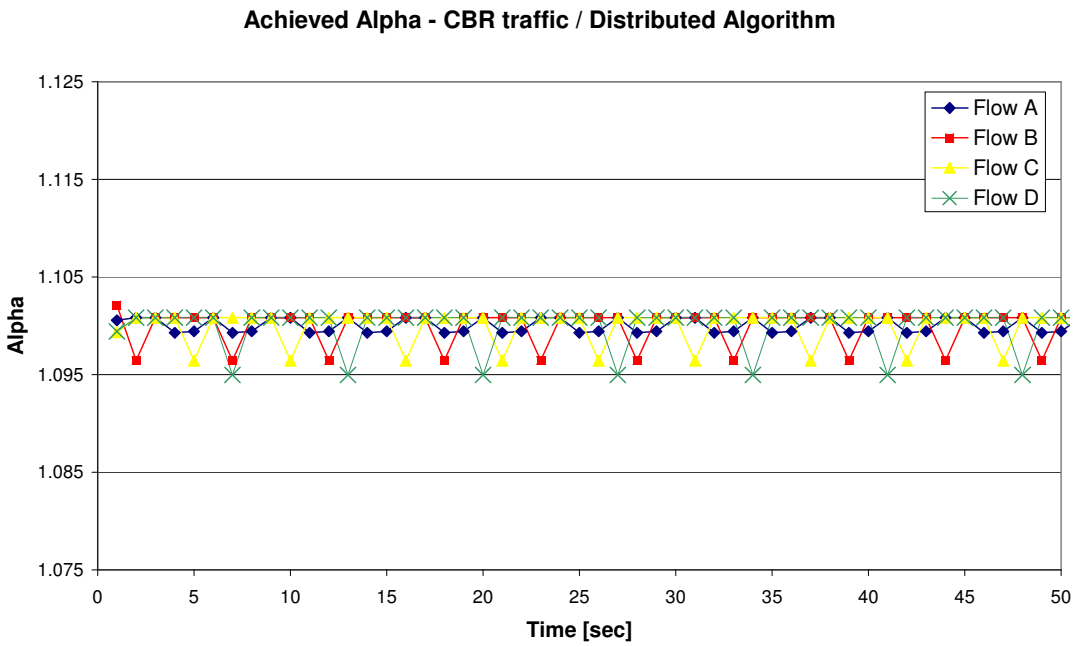

Figure 14. Simulation results for distributed algorithm, CBR flows

In Figure 15 we present the achieved α, simulated in a variable traffic condition.

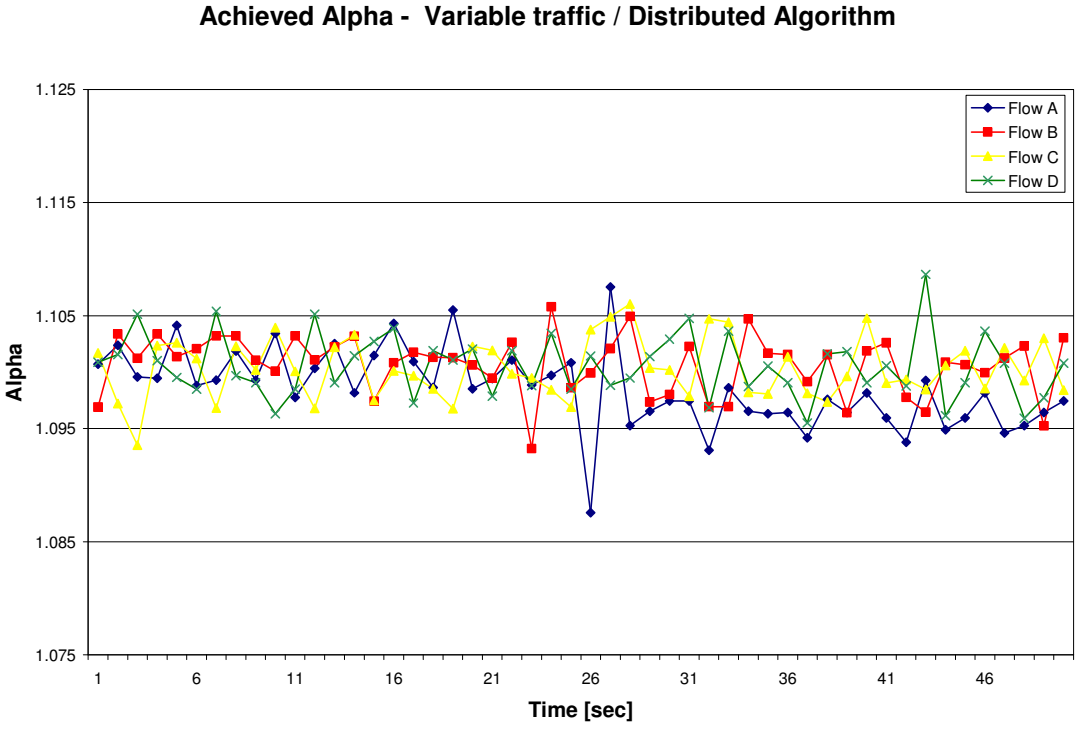

Figure 15. Simulation results for distributed algorithm, variable flows

It can be seen that the values are not as regular as in the previous case, but still are very close to the desired target. As illustrated in this section, we have found that the distributed algorithm can keep the α parameter close to the target value. We did simulations on all the three network scenarios and the detailed numerical results are presented in section 5.3.6.

5.1.4 Centralized Algorithm

As detailed in section 4.1 we have proposed to alternative solution for the network wide proportional service architecture. The distributed algorithm presented in the previous section is ideal for the distributed architecture, but in the case when a central broker can aggregate network wide information, we can follow a different approach, where all flows are handled together during one single run of the algorithm.

We have proposed a centralized iterative algorithm that computes the bandwidth of every flow at ingresses according to eq. *(13)* [C1][C3][J1]. We present the flowchart diagram of the proposed iterative algorithm in Figure 16.

The major difference of this iterative algorithm, compared to the distributed one is that it eliminates *all* extra traffic (i.e., at network level) at the ingress, while the one presented in the previous section eliminates only the extra traffic over the *path*. It achieves a global optimum, whereas the distributed algorithm may yield local optimum in transient periods (that is, until all the paths are updated).

The algorithm is an iterative one and is based on the enforcement of both the eq. *(13)* and the proportional differentiation eq. *(6)* over the flows sharing the bottleneck link. If the traffic is shaped at the ingresses of a network according to this algorithm then there will no losses *within* the network and the output bandwidth will be as given by eq. *(6)*.

The algorithm always searches for the most congested bottleneck in the network, and it does so based on an overload factor.

Definition 1-9. The *overload factor* γ^l of a link l is the fraction of the flow that should cross the bottleneck, if the condition from eq. (2) holds for all these flows.

The overload factor shows that if the link is how many times is more congested compared to the state of the link, when it is at the edge between being loose bottleneck and tight bottleneck. This latter link is characterized by the fact that it can accommodate all the shaped traffic demands, but nothing more. Therefore the overload factor is the ratio of the total capacity of the link and the sum of the shaped traffic demands of all the flows that cross it. Based on eq. *(24)* we can express γ^l as a function of offered load of the flows and the differentiation parameters

$$\gamma^l = \frac{\mu_l}{\sum_{(x)\_cross\_l} \sum_i \frac{F_{i,in}^{(x)}}{\alpha_i}} \tag{25}$$

The most congested bottleneck among a set of links is the one where the overload factor is the smallest one.

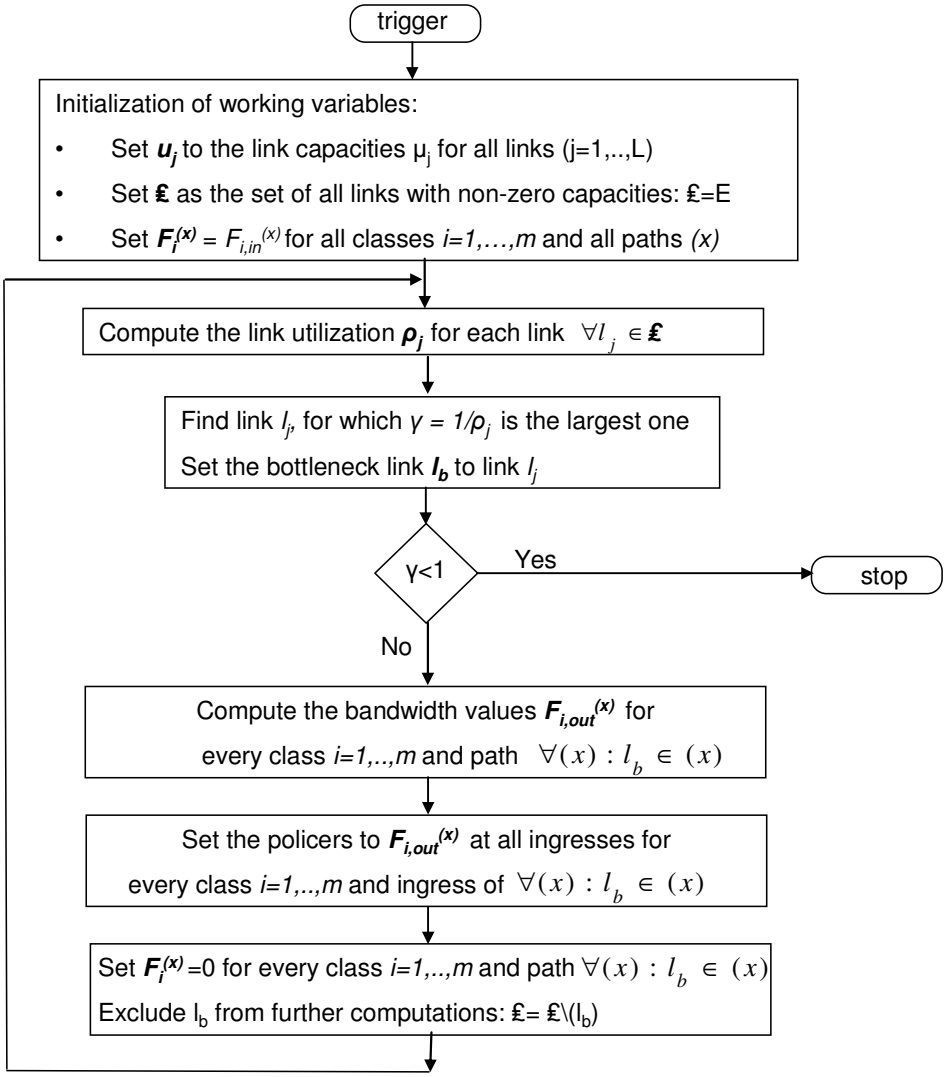

Figure 16. Flowchart diagram of the centralized algorithm

The detailed description of the algorithm, using the notations introduced in section 5.1.2 is as follows.

In step 0 we initialize the variables.

STEP 0

Initialize u_k with the capacity of the link l_k

$u_k = \mu_k$, $\quad k=1,...,L$

Initialize £ as the set of links with non zero capacities

$£ = \{l_j : \mu_j > 0, j=1,...,L\}$

Note that initially £ is the set of all the links: £=*E*. This set will be reduced in each round of the algorithm, as described later.

Initialize

$F_i^{(x)} = F_{i,in}^{(x)}, \quad i = 1,...,m \quad \forall(x) \in P$

This means that $F_i^{(x)}$ is a working variable in the algorithm, which is initialized with $F_{i,in}^{(x)}$, and it can be modified in each step of the algorithm, while the offered load $F_{i,in}^{(x)}$ is left unchanged.

STEP 1

The task of the first step of this iterative algorithm is to find the tightest bottleneck within £.

The link utilization factors ρ_j over the links l_j, $j=1,\ldots,L$ are calculated by summing up the shaped traffic demands for all flows that cross link l_j and divided by the capacity of the link u_j. Then the bottleneck can be found by searching for the link with the *highest* utilization factor.

$$\rho_j = \frac{\sum_{\forall(z):l_j \in (z)} \sum_{i=1}^{m} \overline{F}_{i,out}^{(z)}}{u_j} \qquad \forall j \in £$$

$$\rho_j = \frac{\sum_{\forall(z):l_j \in (z)} \sum_{i=1}^{m} \frac{F_i^{(z)}}{\alpha_i}}{u_j} \qquad \forall j \in £$$

Note that a link l_j is overloaded only if its $\rho_j > 1$.

The most heavily loaded link l_h is chosen next, comparing the utilization factors of all the links in £:

$$h = \arg\max_{\forall j:l_j \in £} \rho_j$$

STEP 2

Now the tightest bottleneck is already identified. In this step we calculate the achievable bandwidth of all the flows crossing that bottleneck.

The shaping parameter γ for the network-wide bottleneck link is given as

$$\gamma = \frac{1}{\rho_h}$$

If γ is greater than one ($\gamma > 1$) some of the higher class traffic demands can be satisfied and the links are underutilized. Nevertheless, the goal of this algorithm to differentiate the flows based on their classes. Therefore in this step, even if $\gamma > 1$ (that is, there is no bottleneck in £), we will police the lower class flows ($i=1,\ldots, m-1$). Thus the policed outputs of the flows are given below (note that $\alpha_m=1$):

$$F_{i,out}^{(x)} = \overline{F}_{i,out}^{(x)} = \frac{F_i^{(x)}}{\alpha_i} \qquad \forall(x): l_h \in (x), \quad i = 1,\ldots,m$$

The remaining task is to distribute remaining free link capacities to satisfy as much lower class demands as possible, which is done in **STEP 3** of the algorithm.

If $\gamma = 1$, then there is a link in £ that is at the point where still every shaped traffic demand can be satisfied. Technically this case can be considered as a specific sub-case of the situation where $\gamma > 1$.

If γ is less than one ($1 \geq \gamma$), link l_h is a tight bottleneck. All flows that cross this link will suffer, since their initial offered loads cannot be served by the network. Thus the achieved throughputs of these flows are given as follows:

$$F_{i,out}^{(z)} = \frac{u_h}{\sum_{\forall(y):l_h \in (y)} \left(\frac{F_i^{(y)}}{F_i^{(z)}} \sum_{g=1}^{m} \frac{\alpha_g}{\alpha_i} \frac{F_g^{(y)}}{F_i^{(y)}} \right)} \qquad \forall(z): l_h \in (z) \wedge F_m^{(z)} \neq 0, \quad i = 1,...,m$$

The excess traffic ($F_{i,in}^{(x)} - F_{i,out}^{(x)}$, $i=1,\ldots, m$) would be lost at the bottleneck link anyway, so it is desired to block this traffic at the ingress. By placing traffic policers at the edge routers (ingresses) the congestion collapse from undelivered packets can be avoided. A flow which has already been shaped to its bottleneck link, is called *fixed*. As the algorithm iterates, it will 'fix' the flows one after the other.

Note that the $F_m^{(z)} \neq 0$ condition means that those flows that already have been 'fixed' are not considered anymore (their share has been extracted from the link capacities along their path, anyway).

The capacities for all links l_j that are along flows *(y)*, which cross the bottleneck link l_h are decreased with the offered load of the 'fixed' flows:

$$\dot{u}_j = u_j - \sum_{\forall(z):l_h \in (z)} \sum_{i=1}^{m} F_{i,out}^{(z)}, \qquad \forall j: l_j \in (y) \mid l_h \in (y)$$

Also, the already 'fixed' flows will not be considered anymore.

$$\dot{F}_i^{(x)} = 0, \qquad \forall(x): l_h \in (x), \quad i = 1,...,m$$

Since the bottleneck link l_h is now fully utilized, it is removed from the network for further calculations, i.e.,

$\mathcal{L} = \mathcal{L} \setminus \{l_h\}$

Next, repeat **STEP 1** and **STEP 2** as many times as a tight bottleneck is found: go to STEP 1. Note that the upper bound on the nr. of steps is the total number of paths.

STEP 3

STEP 2 of the algorithm assures that all flows are scaled to get a fair share of the available capacities (according to the fairness condition), and the proportional differentiation between classes is also enforced. However, in a lightly loaded situation (loose bottleneck or no bottleneck) there are free capacities that could be used by the lower priority classes. Allocating more bandwidth for the low priority flows would not degrade the quality of high priority flows, only the strict proportional differentiation is neglected. It can happen that all classes get the same (lossless) service.

It is up to the network administrator how to use the excess capacity in this case. Using the excess capacity to increase the throughput of the lower-class flows causes deflection from this assumption, thus the achieved ratio among classes will differ from the target α_i values. One solution could be to downgrade this excess traffic to the lowest quality class. By this classification we keep the differentiation of the flows and at the same time we

increase the utilization of our network (since we do not eliminate traffic for which we have capacity to carry).

Evaluation of the Centralized Algorithm

Note that in the case of distributed algorithm it was enough to compare the *shares* s_i along the path *(x)*, because the investigated links were all along given path. This also means the same flow crosses all these links, and can be used as a basis for comparison. In the case of centralized algorithm we have to compare all the links, meaning that there is no single flow that be used in a similar role. We need a normalized parameter, which is independent of the load of a single flow – and the utilization factor γ is such a parameter. As can be seen in Figure 17, the algorithm performs well in CBR traffic conditions, as expected.

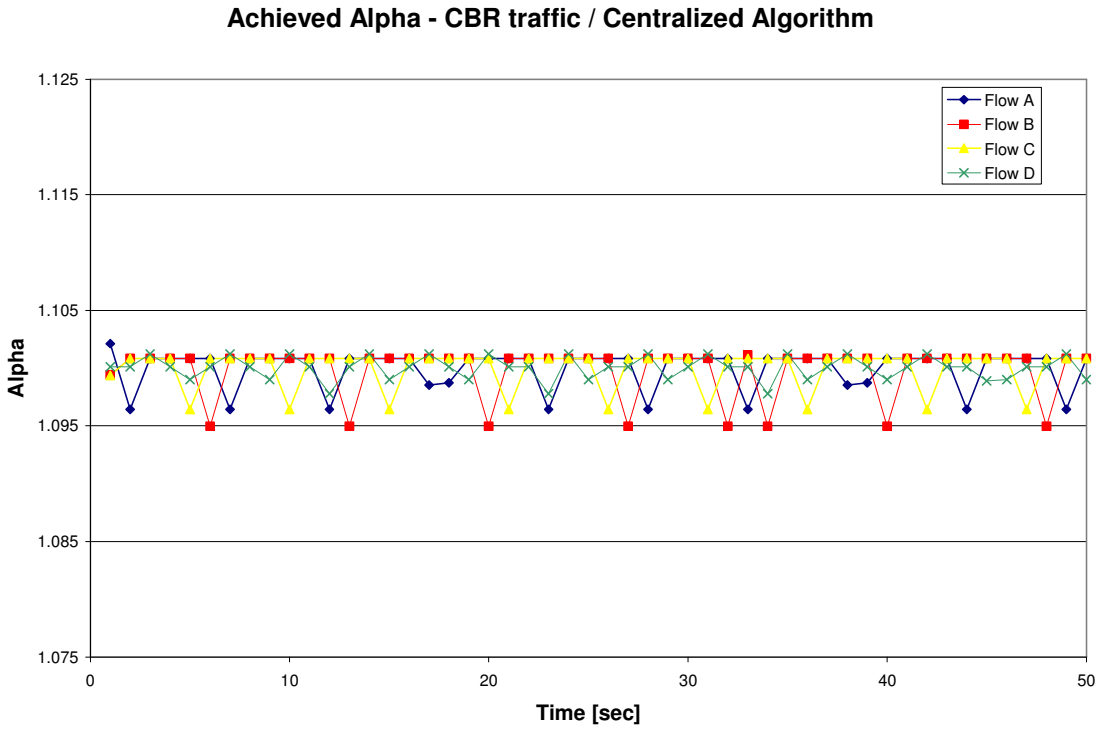

Figure 17. Simulation results for centralized algorithm, CBR flows

Then we turned to test the behavior of the model in dynamic traffic conditions, as well. We analyzed the case of *variable UDP flows.* The traffic model is described in Appendix A and tries to approximate an aggregated flow, which is dynamically joined and left by micro-flows. Using this scenario we tested whether the algorithm can respond to changes in offered load.

Figure 18 presents the achieved α ratio for variable load. It can be seen from the plot that the α ratio is kept very close to the target 1.1 value. If we compare the results to those obtained for the distributed algorithm (as presented in Figure 15), we can see that around

the 25th simulated second the distributed algorithm temporarily polices flow A with slightly less success, while this is not the case for the global algorithm. This is the result of transient periods, under which the distributed algorithm reacted locally to a change in the network. At that particular moment shares of flow A were not updated, but its behavior was still affected. Flow A had to "wait" until its shares were recomputed, as well, then it could assure the proportional differentiation among its classes. This is not an issue for the centralized algorithm, which sets the shares of all the flows during each of its run.

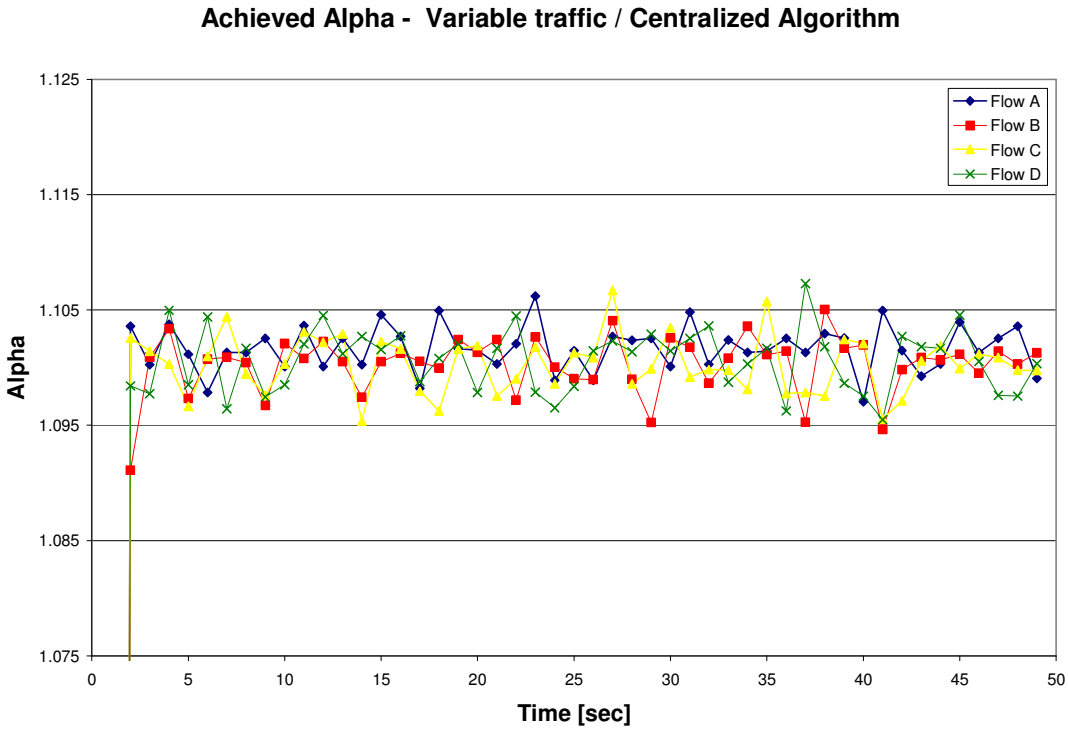

Figure 18. Simulation results for centralized algorithm, variable flows

Combining the algorithms with measurement-based predictions

Next we refined our architecture, and introduced a measurement-based prediction to approximate the offered load. This means that the input of the algorithm is not the exact traffic matrix, but an estimated one. We used a simple prediction scheme: the prediction takes the average load of the last five seconds as an approximation for the next five seconds. The results are shown in Figure 19.

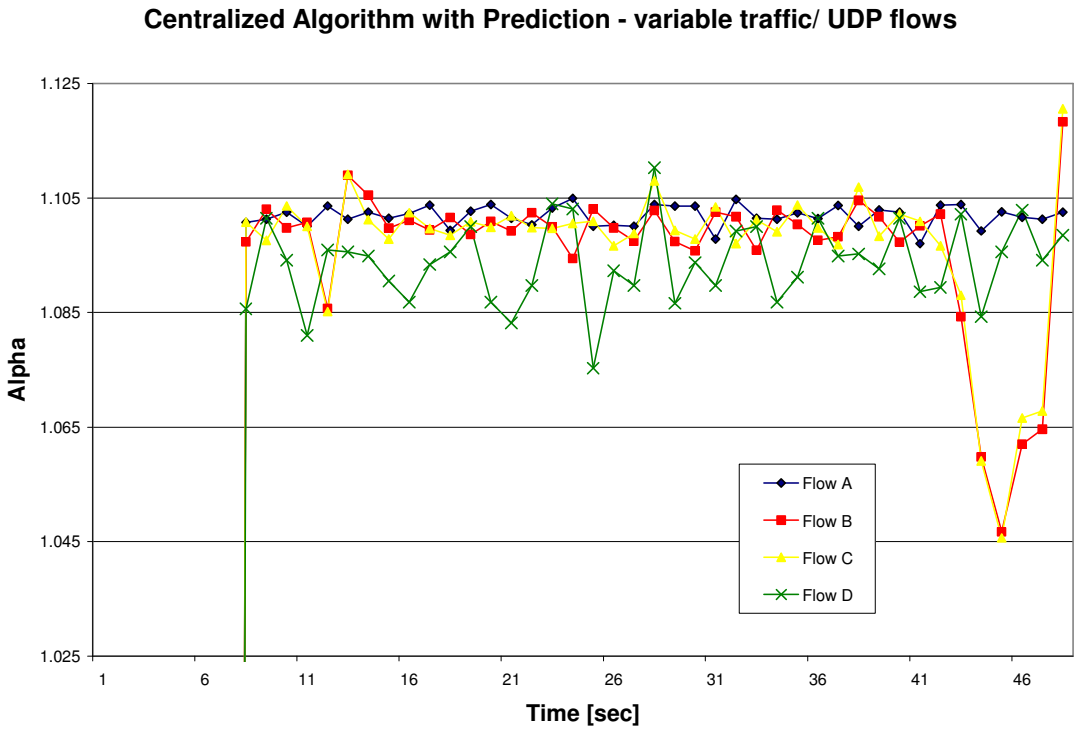

Figure 19. Centralized algorithm with prediction, variable flows

The first prediction interval starts from 2 seconds when all components of the flows are active. (Thus the relevant result starts after 7 seconds of simulation time.) As it can be seen on the charts, the output is mostly correct and it keeps the α parameter between 1.11 and 1.07. In the last period the deviance from the expected value is too high though and should be eliminated by refining the prediction schemes. One possible solution is to improve the accuracy of predictions, using more advance techniques, as presented in section 3.4. A different way is to combine the use of the distributed and centralized algorithms.

We simulated this latter solution as follows. We deploy the estimation based centralized algorithm, as presented above and we monitor the achieved α values. When we observe that the achieved α ratio is not maintained within reasonable bounds during an estimation epoch, we trigger the distributed algorithm. In this case we apply the local algorithm only at the ingress of the non-conforming flow. For this experiment we have set the threshold to 1.075: when we observe that $\alpha < 1.075$ we trigger the distributed algorithm (note that we replaced the trigger event from the flowchart in Figure 13, as it was originally set in section 5.1.3). Figure 20 depicts the result of such a hybrid scheme and it shows the performance improvement compared to the simple prediction scheme-based solution. Since the distributed algorithm has been triggered by the out-of order values of flows B and C and they share a common bottleneck, only their shares are recomputed.

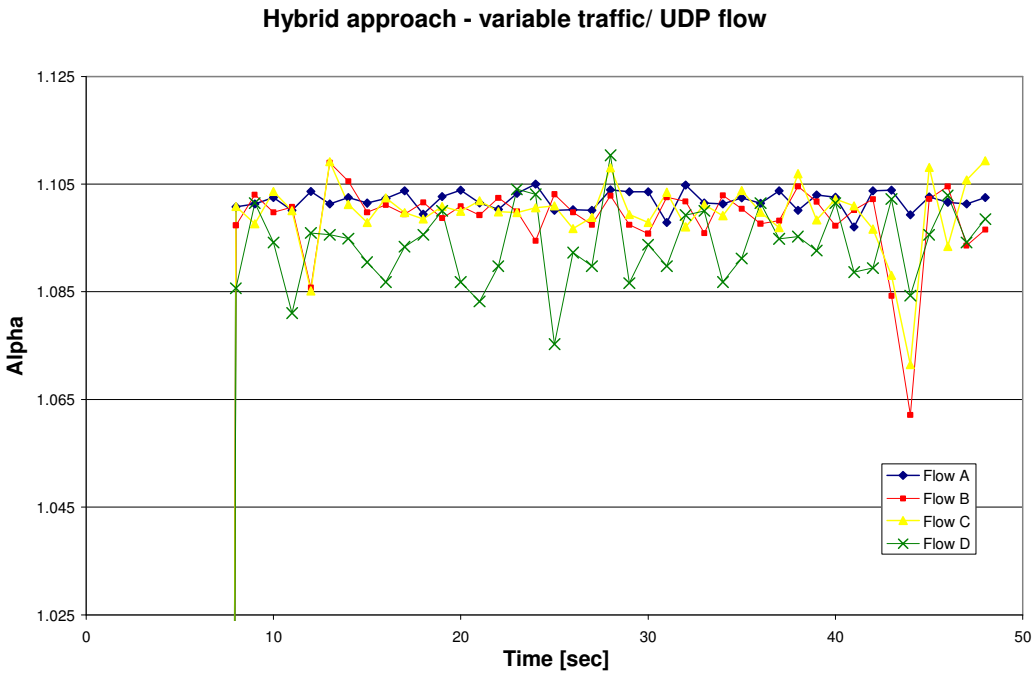

Figure 20. Hybrid approach, variable flows

5.1.5 Discussions on algorithms

In Figure 21 we summarized our simulations for both algorithms. We have simulated ten times each of the three network scenarios from Appendix C and averaged the results.

Algorithm (4 core routers)	Total throughput	Achieved Ratio	Standard deviation
Basic Shaping	96.54%	0.6317	-
Distributed	99.25%	1.1107	0.0148
Centralized	100%	1.1005	0.0026

Algorithm (9 core routers)	Total throughput	Achieved alpha	Standard deviation
Basic Shaping	94.88%	1.0410	-
Distributed	98.35%	1.1063	0.0127
Centralized	100%	1.1048	0.0024

Algorithm (15 core routers)	Total throughput	Achieved alpha	Standard deviation
Basic Shaping	95.53%	1.0308	-
Distributed	98.49%	1.1058	0.0131
Centralized	100%	1.1016	0.0020

Figure 21. Normalized throughputs and achieved alphas for different scenarios

As it can be seen, both algorithms successfully keep the target alphas. Nevertheless, the centralized algorithm outperforms the distributed one. We also show the results for the naïve algorithm, to illustrate that it gives unacceptable results in all the scenarios.

Based on these simulations we consider that both algorithms can be considered for deployment by the operators, and they have to select based on their own objectives, as detailed during the discussions in sections 5.1.3 and 5.1.4.

5.1.6 Algorithm for responsive flows

As mentioned earlier in Chapter 3, responsive flows can not be handled at the ingress as easily as non-responsive ones. In this section we propose a different way to interpret the network wide proportional service model for responsive flows and we evaluate the accuracy of this proposal with the most used responsive flow type, the TCP [J1][J2].

TCP and Proportional Services

The proportional differentiated service model initially was deployed to be implemented in the propagation (transmission) of UDP traffic. Characteristic to the UDP data transfer is that this protocol does not verify the transmission speed with acknowledgements (does not use feedback). Therefore from our point of view the policing done by the network does not affect the sending rate and we can consider the traffic measured at the ingress routers as the offered load of the flow. We have a different situation for responsive flows though.

TCP flows use a specific end-to-end (host-to host) mechanism to control the traffic, and to avoid congestion collapse; this mechanism is called feedback [77][78]. With the help of this specific mechanism, TCP flows guarantee that every sent packet will arrive to the destination. The cost of this guarantee, from the proportional services point of view is that the sender, the source does not have an offered load; the load has elastic adaptation to the state of the network.

This represents a problem for us, because the presented proportional service model cannot predict the offered load of the flows that enter the network: the offered load always adapts itself to the policed values at the ingress.

The offered load in case of TCP flows

Our proposal is to make this unpredictable offered load proportional with the numbers of the micro-flows within every class. In this approach the entering load of a certain class will be as follows:

$$F_{i,in}^{(x)} = n_i^{(x)} D \quad i = 1,2,...,m \quad \forall (x) \in P \tag{26}$$

where $n^{(x)}_i$ is the number of micro-flows in a certain flow-class over path *(x)*,
D is a network wide constant,
$F^{(x)}_{i,in}$ is the offered load for a certain, *i*, class over path *(x)*.

Now we can compute the achievable throughputs for the different flow-classes using any of the two already introduced algorithms. In this section we use the centralized algorithm.

Before this algorithm could use the eq. (26), we should understand what exactly represents this D value. Note that in the selection of initial capacities for the entering flows is important just the fact, that even if just one flow is entering the network, this flow should occupy the highest bandwidth possible. Otherwise it may happen that the algorithm assigns less bandwidth to the flows than the available one.

Theoretically, all the values should be recomputed, if a TCP flow enters or exits the network. Note that in normal networking condition, the number of flows present in the network is a large value. Thus the algorithm will not be resumed at every single TCP flow entering or exiting, because the modifications introduced by a determined number of flows can be tolerated for the good functionality of the model. In this condition, we will fix a limit for the number of micro-flows. In the moment when, after the last run of the algorithm, the number of micro-flows entering or exiting the network reaches the determined threshold, the algorithm is invoked again.

Active Queuing Mechanisms to police responsive flows

As shown in Appendix A, contrary to UDP flows, the TCP flows can not be policed with simple drop based FIFO disciplines. This has been already discussed in the literature as well [43]. Focusing on the improvement of the congestion avoidance of responsive (chiefly TCP) flows, active queuing mechanisms (AQM) have been proposed.

The most known among these and one of the first proposals is RED, which maintains an exponentially weighted moving average of the queue length which it uses to detect congestion [43]. RED detects increases in the average queue length and uses it to determine whether or not to drop or mark a packet. When the link is congested, RED randomly drops arriving packets, even if they would fit into the queue, to signalize the congestion to the end nodes. The probability of the packet dropping is in function of the average queue length. The fundamental weakness of RED and its variants, which impacts their ability to minimize packet loss, is that all of the algorithms rely on some form of the queue length in order to estimate congestion. While the presence of a persistent queue indicates congestion, its length gives very little information as to the severity of congestion, that is, the number of competing connections sharing the link.

Among several AQM mechanisms we decided that the proposed shaping mechanism to be actually the BLUE AQM [79][80]. This AQM maintains a single probability, which is used to mark (or drop) packets when they are enqueued. This algorithm is not based on averaged queue length; rather it uses packet loss and link utilization history to maintain the congestion signaling probability. If the queue is dropping packets due to queue overflows, the probability is increased. If the link is underutilized, the probability is decreased. To avoid oscillations, it freezes the probability after every change for a fixed time interval. BLUE converges to the ideal operation, even used with small buffers. Since this field of research is still active, newer AQMs are proposed. Nevertheless, the characteristics of BLUE, as presented above, were promising enough to support the network wide proportional service model and were used in our simulations. Our goal was to show that our proposal is feasible with at least one AQM, not to show that compare the performance of different AQMs .

Evaluation of the network wide proportional service with responsive flows

We modified the centralized algorithm, interpreting the offered load for TCP flows as described earlier. In this section we present the results of the evaluation of this solution. We also had to modify our traffic generator module, as described in Appendix C.

The results are shown in Figure 22. As one can see, the simulation results were in good agreement with the theoretical results. The variance of the monitored value appears because the monitoring interval separates some packets, starting to arrive very close to the sampling period's end, and a part of it will arrive in the next second. From practical point of view we considered that the result shows that under static traffic conditions the architecture achieves the targeted behavior, thus the simulation validates the proposal.

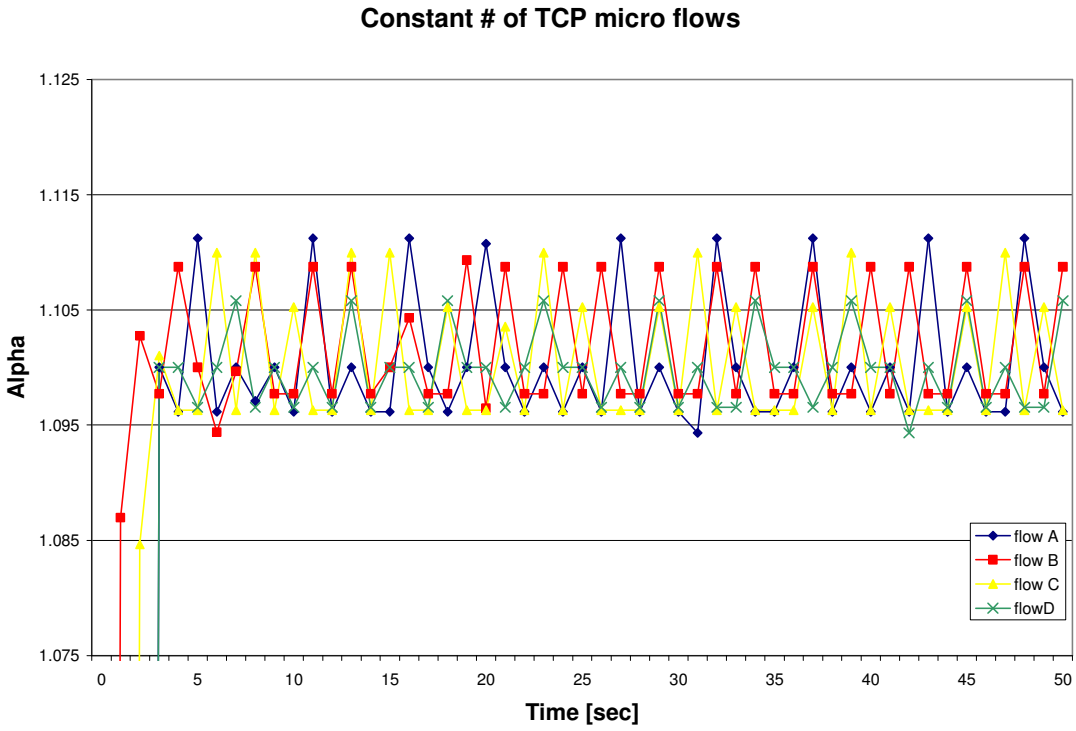

Figure 22. Achieved alpha ratios, static TCP traffic scenario

We also tested our proposal in varying traffic conditions. Figure 23 shows the results of a typical simulation experiment. It can be seen that the result is much more fluctuating, but the achieved alpha values are still close to the target value.

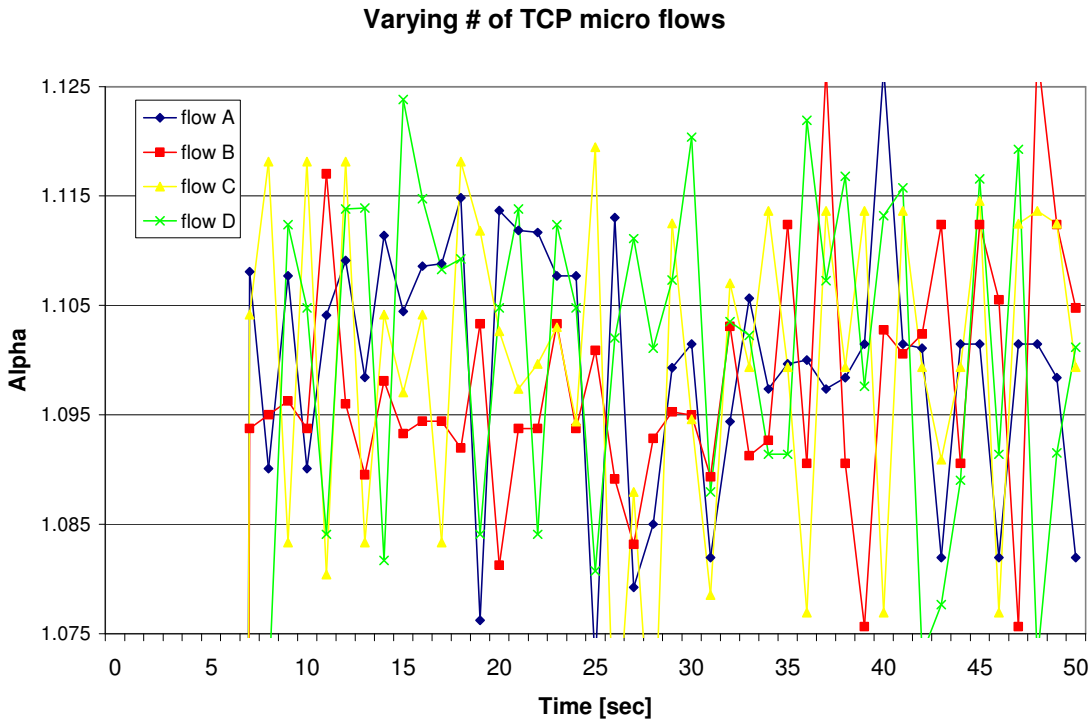

Figure 23. Achieved alpha ratios, varying number of TCP micro flows

We repeated the simulation experiments for both the 4 core routers and 9 core routers scenarios (see Appendix C). The results are summarized in Table 1 below. We already verified the centralized algorithm, and the BLUE AQM is tested already, as well. The results confirm that in both scenarios we achieved the desired differentiation, and the standard deviation is acceptable, as well.

Table 1. Evaluation of centralized algorithm with responsive flows

	4 core router scenario	9 core router scenario
Achieved alpha	1.1004	1.0999
Standard deviation	0.0131	0.0207

We have also tested the effect of the presence of UDP flows in the network on the TCP flows. We separated the TCP flows from the UDP ones, that is the earlier described TCP setups were kept for TCP. Additionally, we injected UDP traffic into the network. In this case the capacities of the links were decreased with the bandwidth “allocated” for this background flow. We did these simulations as a first test of a combined UDP-TCP proportional services network. The only difference was that the UDP throughputs were ’dummy’ values as for now.

Each path has been extended with a UDP flow. Therefore near every High/Low macro flow source-pair a new UDP source node has been introduced, resulting that along the paths of each flow we had three flows. In our first simulations the UDP background traffic was static and it did not affect the TCP flows. Then we changed the size of the

background traffic during the simulation, which in turn resulted in the change of the aggregated TCP traffic. Since from the perspective of the BLUE this event is similar to the change in throughput triggered by the algorithm, we expected that this should not be a problem either. Our simulation experiments confirmed this expectation.

Based on the results we can accommodate both responsive and non-responsive flows in parallel in the same network over the same path. The network operator is the one who has to decide on its strategy on how to divide the resources among the two flow types.

Chapter 6 Summary

In order to be able to operate a global communication infrastructure that meets user expectations the underlying communication network must be able to offer predictable Quality of Service (QoS) to the applications, and maintain this offer over a global scale. We introduced a domain-wide proportional service that guarantees the ratio between the qualities of service experienced by two users served in different classes will remain the same irrespective of the changing network conditions. This approach differs from the widely used per hop behavior (PHB) based ones. We have proposed two alternative network architectures that realize this discipline, following a centralized and a distributed approach, respectively. In both cases the core routers will have to maintain information only on the aggregate flows, therefore the proposal conforms to what has been called the core-stateless architecture.

We have derived analytically the capacity share of a flow in the network-wide proportional service model, which crosses a bottleneck within the network. Based on the analytical results we proposed two algorithms, a distributed and a centralized iterative one, which shape the traffic at the ingress in such a way that the flows of the network exhibit the properties of a network-wide proportional services model. We have proposed solutions to handle both responsive and non-responsive flows. We have used extensive simulations to show that the algorithms perform well in various traffic conditions and network environments.

The properties of the proposed mechanisms would allow the operators to deploy cheaper devices, save operational (management) and marketing costs (because the offered QoS levels can be easier communicated). The advantages of the network-wide proportional services can be particularly well exploited in those network domains that are congested and there are groups of users to be served at different priorities or QoS. In such cases the data of the users can be classified according to the assigned priorities and then the traffic from higher classes will enjoy better QoS. Note that in our proposal even the worst classes will be served, since there is no admission control. In the world where customers have been used to the free best effort services of the Internet this can be compelling, since nobody will be excluded (blocked) from the service. The operators will have the chance to treat preferentially their premium or fair users in a controllable way, by adjusting the ”knobs” according to their business strategies. E.g., ISPs can restrict (downgrade) those users, who abuse the flat rate internet subscriptions (excessively using multimedia streaming or file download applications) and affect the regular users. A different example is a mobile operator that rewards its subscribers who use premium services (e.g., pay-per view content, business services). The traffic volume generated by these premium services can significantly vary in space and time. In such case it is convenient to deploy our proposal, which automatically adapts the QoS to the traffic demands.

There are open issues related to the network-wide proportional services. One possible future research direction is to identify specific applications and service business strategies (e.g., premium multimedia services in the future mobile networks) and investigate the behavior of the system in those specific traffic conditions. A different task would be to identify specific network architectures (e.g., all-IP 3GPP core networks), fit the model

into those architectures and test the way it influences current services and user behavior the introduction of proportional differentiation over those network domains.

Part II Distributed Management Framework for Self-Organizing Networks

Chapter 7 Introduction

The paradigm shift in society and the opportunities enabled by new technological advances in devices, place completely new requirements on the evolution of today's Internet. Future Internet [9] will enable a multitude of new application sectors leading to the development of new markets. As a response to the above challenge a broad research effort has been mobilized on Future Internet all over the world: in the EU (http://ec.europa.eu/foi), in the USA (http://www.nets-find.net/ - FIND), in Asia (http://mmlab.snu.ac.kr/fif/ - Future Internet Forum and http://akari-project.nict.go.jp/ - AKARI) and in the IETF (http://www.postel.org/e2e.htm - End-to-End Research Group). These organizations compete to offer a (re-)design of new Internet architecture, an important aspect being the autonomous configuration of network infrastructures and services, generically referred to as Future Internet Architectures [9].

All the works on future networking scenarios emphasize two key issues [83]. Firstly, networks will become more technologically heterogeneous, accommodating old and new access systems as well as applications and services – indeed, migration and feature rollout will not be a 'one-off' but a constant activity. Secondly, networks will become organizationally heterogeneous: today's cellular systems will be complemented by a diverse mixture of other network types – personal, vehicular, sensor, hot-spot and more.

Some of these networks may even be useful in isolation, but the true potential of this infrastructure is only obtained when they become interconnected in a way which allows their resources to be shared, and new communications patterns to be established between their users and services. This translates into the need that such network of networks will have to form and re-form dynamically in response to changing conditions. At the same time these interactions must be achieved automatically because the dynamic nature of the set of network-network relationships rules out time-consuming or complex manual configuration. What was the vision of "All-IP" networks developed at the turn of the millennium during the last years is clearly heading towards this direction.

An increasing divergence in the network control layer: different control environments are established to facilitate services like virtual private networks (VPNs), security, mobility, quality of service (QoS), network address translation (NAT), multicast, and so on. For a multitude of services, data might still be handled by uniform Internet networking, but the control of such services is becoming increasingly fragmented. More and more, the network as a whole therefore diverts from the pure end-to-end view of the Internet. [84].

The appearance of novel wireless technologies, and the concept of network composition, where networks agree to (partially) share a common control space, represents a challenging scenario from a network management perspective. In order to meet these challenges management systems these networks have to become more dynamic, more

self-managing, and they have to participate more actively in (inter)-networking. Novel protocols and approaches have to be researched and developed to go beyond existing network management paradigms in order to respond to these challenges [85].

The objective of Part II of the Dissertation is to offer a network management solution for dynamic network composition in self-organizing networks.

The management of large, heterogeneous, global and dynamic networks cannot be kept centralized anymore, because they would not scale. According to today's practice user/operator intervention is required in order to combine networks or services, taking into account user/network context (e.g., her/his needs, preferences, location). We introduced a peer-to-peer hierarchical overlay that can hide local interactions, keeping the overall system scalable. We proposed new interactions called compositions between partitioned overlays that model the combination of two separated network domains. Thus network interaction is automated through online negotiations between devices and network, making it transparent to the services and/or to its users.

Chapter 8 Related works

8.1 Challenges on network management in the Future Internet

The challenges posted by the increased competition and cooperation in an environment with a multitude of access technologies, network operators and business actors ask for heterogeneous networks to interact and cooperate, on demand and transparently, without the need for manual configuration or offline negotiations between network operators. The development of these concepts into what it is expected to become the Future Internet is guided by market and business analyses carried out in European research projects [85][103][104][105] as well as by the conclusions and technology concepts discussed within the WWRF (Wireless World Research Forum) [135].

All the new concepts towards Future Internet burdens existing networking infrastructure with more and more tasks that sooner or later will not be manageable by human intervention alone. In the context of the future networking environments the network management and control should become more dynamic and flexible. The concept of self-managing networks is widely accepted as a solution that overcomes the rapidly growing complexity of the Internet and/or other networks and enables their further growth [10] [83]. Operators, as key responsible actors for shaping the future networking infrastructure, are willing to have new functionalities and mechanisms that allow the network to self manage with as little of human intervention as possible, providing higher availability of services, lower the provisioning time of new customers and reduce the time to market of new services [83].

Interactions among heterogeneous and dynamic networks are depicted by the compositions concept. Composition is the process through which two networks interact, negotiate and decide their cooperation intentions, combine their resources as decided during negotiation [85][84][87]. Specifically, while composing networks is an easy task when only packet forwarding is concerned, advanced end-to-end functions targeted to the likes of QoS and security as well as mobility are currently very difficult to establish across network boundaries. The organization of these functionalities is handled by control and management planes in current networks. In the next sections we review the background of this field, as the motivation and starting point of our work.

8.2 Network and System Management for interconnected networks and management domains

In current understanding the control plane functionality is related to establish, maintain, and tear down connections (basically refers to signaling in the network), and the management-plane for the "rest", or which can be thought of following the FCAPS (Fault, Configuration, Accounting, Performance, Security) structure of TMN (telecommunication network management) [6]. A potential division of control and management tasks within the Future Internet is as follows: the “operational” decisions to the control plane, while keeping the “strategic” decisions in the management-plane [C7].

According to this split the control-plane processes would handle the operational activities to be taken during compositions [88].

Since the role of the control-plane is to support the functionalities of the interworking networks in an operative way, the composition at control plane level basically translates to a bilateral network-to-network interaction among those networks, and the functional blocks within these networks. Control plane negotiation thus involves generic network attachment-related actions (connectivity, authorization, policy) and actions related to protocols implementing concrete functionalities (mobility management, QoS, etc). Some of these negotiations should be done earlier than others (as detailed in Chapter 6 of [B1]), while most of the functionalities controlling a service can be negotiated in parallel.

In contrast, the network management in the classic sense is a centralized process [90] [91][92][93], so if we use the classic network management paradigms, as well, then local interactions would propagate over the whole network, rendering frequent networks compositions within a world-wide network unfeasible. Therefore the proposed principle for control plane composition can not be applied for the management plane composition, because it would not scale. Just to exemplify this, in the classical networking environment a control level interaction between two local sub-networks can remain local (e.g., mobility if micro mobility is used), whereas the management process of those networks (e.g., registration of management information –MIBs- of the member nodes) would involve global actions (store/collect these MIBs in/by a central repository/manager agent).

This is the reason why in order to meet these (Future Networks) challenges network management systems have to become more dynamic, more self-managing, and they have to participate more actively in (inter)-networking. As a consequence we should treat management plane network composition and reconfiguration in a self-managed way, going beyond "classical" or legacy network management paradigms.

8.3 New network management frameworks and paradigms

According to the authors of [94], management solutions can be categorized as centralized, weakly distributed and strongly distributed approaches. This categorization is advantageous when we compare traditional and more comprehensive proposals. Current trend in network management evolves towards strongly distributed approaches and towards automation of management. One of the first proposals towards a distributed approach was the "management by delegation" paradigm [95]. Delegation consists of downloading software code to a remote agent allowing local processing of some or all of management data.

Novel network management frameworks

As for now we are aware of only a few project-related architectural solutions which propose a general and comprehensive approach to autonomic network management.

In the FOCALE system presented in [96] [97] human activity is reduced to the definition of business goals, therefore it is characterized by a high level of autonomy. Still, since the system is quite complex, it is difficult to understand, if there is a case of unforeseen

failure of the management system itself. The same goes for the ASA architecture [98], being a generic architecture, but focusing on managing path-based traffic engineering services, is encompasses different abstraction layers and heterogeneous resources, it is characterized by a high level of complexity.

Other network management architectures focusing on specific network environments (e.g. MANNA [99] in WSNs, Madeira [100] in P2P) are limited to their target environment and their lack of generality doesn't address the requirements of a heterogeneous environment.

Ambient Network (AN) [B1][84] will be later introduced in detail and our proposal made within the framework of this project is presented in Chapter 10. AutoI [101], a project partly following the ideas introduced by AN, advocates the introduction of multiple control loops in network management systems, creating new control planes to the likes of orchestration and contextualization planes. The issue with this complex model is the efficient interoperability among the various control planes.

The Autonomic Network Architecture (ANA) project described in [103][104] has proposed to develop a new autonomic network architecture beyond legacy Internet, looking to expose the properties and verify the feasibility of their proposal. The aims described by ANA are similar to the ones in the In-Network Management (INM) model of project 4WARD in that they both try to increase the level of automation in the network and they follow a clean slate approach [105]. By proposing the INM, the 4WARD project worked towards the high integration of management functions with the network components: management functions are seen as embedded capabilities, which differ radically from the traditional design and deployment of management functions as add-on features. ANA and INM follow a clean slate design for the Future Internet in an attempt to radically redesign today's networks based on novel principles. With this respect, 4WARD follows other similar clean-slate initiatives of different research communities, like [104][106]. Nevertheless, clean-slate designs are long term proposals which can not solve current and mid-term problems of the Internet.

New ideas in network management

In addition to the projects and architectures mentioned above, several enabling technologies, paradigms and models have been proposed for the implementation of network management frameworks, as reviewed below.

Expert systems also originating from the domain of artificial intelligence attempt to incorporate human knowledge into their programming, with the aim of further facilitating autonomy [107]. Similar to intelligent agents, expert systems tend to be incomprehensible to users if they capture complex situations.

The mobile agent technology has been proposed for the management of networks and distributed systems as an answer to the scalability problems of the centralized paradigm. Management tasks may be assigned to an agent, which delegates and executes management logic in a distributed and autonomous fashion. Even if these were promising proposals, there are severe security concerns with such approaches, because external agents are allowed to enter the system [108].

Active networks allow programs to be inserted into network components, running customized tasks on the data passing through. They can be thought useful in the realization of real-time adaptations. However, as with intelligent agents and expert systems, there are major security concerns [109]. In addition, they require more heavyweight support in both hardware and software [107] and there are also performance problems.

A different approach is the use of promise theory [110] a graph theoretical framework that simplifies the understanding of complex relationships in distributed network environments. The basic idea is that totally autonomous nodes interact with each other through promises that they make and cooperating autonomous agents are organized into new structures. The difficulty of the application of promise theory relies in the proper mapping of the expected behavior of the promising agent into the formal framework of the promise theory.

The pattern-based management paradigm aims at addressing the well-known drawbacks of centralized management [111] by investigating a distributed management approach based on graph traversal algorithms to control and coordinate the processing and aggregation of management information inside the network. A key feature of the approach is the separation of the distribution and computation/aggregation of information from the semantics of the management operation. This approach is best used for efficient monitoring of complex events in the network.

The biologically-based models are on the rise, because several biological systems to be found in the nature expose self-* properties that are expected from the autonomic management systems, as well [113]. The compatibility of such biologically inspired models with the existing network architectures and classical approaches is still unclear and it has to be investigated and proved.

8.4 Self-Organizing Networks

As discussed before in this Chapter, modern network management systems should be distributed and self-organizing. A natural solution to self-organization of networks is the peer-to-peer (p2p) paradigm and this section gives a background on this field.

The decentralized nature of pure p2p systems means that these properties are emergent properties, determined by entirely local decisions made by individual resources, based only on local information: we are dealing with a self-organized network of independent entities [114]. Moreover, p2p networks are self organizing also because they automatically adapt to the arrival, departure and failure of nodes.

A p2p network is formed by interconnecting end-systems (i.e. the peers), where a direct link between peers is a (potentially multi hop) layer 3 route. We can say that a p2p network forms an overlay on top of the underlying IP network. Currently there are two main P2P approaches to establish the overlay, namely using structured or unstructured topologies [115].

Unstructured p2p networks rely on statistical properties and have proven to suffer from scalability problems. In order to address these issues, several popular unstructured P2P applications (e.g., BitTorrent [115], FastTrack [116], Gnutella [117]) explored

heterogeneity of the underlying network to improve search performance. They introduce a hierarchy into the p2p network, organizing the overlay topology in two tiers, the lower representing normal peers and the higher representing SuperPeers (named SuperNodes, Hubs, UltraPeers, Reflectors, etc. depending on the system). These systems significantly outperform pure flooding systems by preventing low-performance hosts from transiting requests, but they still rely on flooding at the superpeer level [118].

The structured P2P networks, such as CAN [119], Pastry [120], Chord [121], Tapestry [122], Kademlia [123] assign keys to both peers and content (data). Currently the keys are hash values of the mapped data, therefore these structured p2p systems are commonly called Distributed Hash Tables (DHT).

Within the research of p2p networks a special attention has been paid to the representation of the peers within a peer group. A specific issue is how to handle the peers during the interactions of their peer groups. The use of the already mentioned superpeers to represent the peer groups is a straightforward and simple solution to increase the scalability of the process. This choice is natural and intuitive particularly when the peers are grouped according to some other criteria (e.g., locality, content type). Several papers deal with the details of superpeer election, focusing on the efficiency of this approach under dynamic conditions [118][124][125][126][127]. Note that the dynamism of the p2p systems continuous is affecting their performance mostly by the costs of update processes resulted from the joining and leaving of peers (churn) [128].

A different approach on handling peer group during interactions is the exploitation of the intrinsic properties of the DHTs. In structured p2p networks DHTs are deployed, anyway. Therefore the solution of tweaking the DHTs to be able to support such processes is at hand. The disadvantage of such solution might be the lack of a well known representative of the peer group. This issue has been researched in [J5], and further extended in [129], where the authors tried to reduce the cost of maintaining the DHTs in dynamic conditions, and in [130][131], where the authors tried to speed up the merging process of two separate peer groups (i.e., DHTs).

Chapter 9 Distributed Management Framework

There is a consensus on the principle that networks management systems should abandon the centralized, client-server paradigm and take new approaches to cope with the dynamic and heterogeneous new networks. Nevertheless, even the newer concept of self-management concept needs to be rethought. Self management, as local management operation that focuses on the network elements (see Figure 24), is not enough [85].

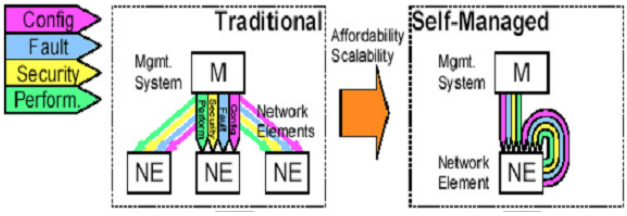

Figure 24. The earlier ("classical") management concepts

A new level of self organization should be introduced, which can be applied at network level. Throughout this Part II of the Dissertation we suppose that the networks hold self-organizing capabilities. This means that the network elements have enough computational capacity to run the self-organizing logic and have proper interfaces to manipulate the network elements. Also, unless noted otherwise, when we use the term "management", then we refer to network management.

In order to deal with network-to-network interactions we introduced the notion of composition. Composition is the process through which two management domains / authorities interact, negotiate and decide their cooperation intentions, combine their resources as decided during negotiation. Composition refers to domains, therefore this process is also referred to as *network composition*, where the network is the logical network controlled and managed within/by the management domain / authority. As such network composition covers all aspects of inter-network interactions and cooperation. We opted for this concept to fit the handling of management domain interactions and procedures into. In this respect the composition process of the management domains is governed by some composition manager functions and is realized by the control plane functions of the interacting networks. Both the management plane and the control plane functions of the networks might be affected (changed, enhanced or limited) throughout the composition process [B1][J3][24][25]. A further advantage of this approach is that brings the control and management planes under the same umbrella, facilitating the close cooperation between these two planes.

When two separate networks compose, one crucial issue is to combine their management systems into a consistent management system for the composed networks. Similarly, when a network splits into two (or more) networks, the management system has to dissolve into corresponding pieces in a consistent and predictable way. We have proposed a management framework and management processes that alleviates this problem.

9.1 Distributed management architecture

As detailed in the previous Chapter, there are several solutions to increase self-awareness and autonomicity in network management systems. There are also solutions that focus on specific networks (mostly wireless mesh and mobile adhoc networks) and scenarios (typically monitoring, fault detection and multimedia services). Those that propose a holistic approach instead go for a clean-slate approach, disregarding the huge potential still represented by the IP based networks.

9.1.1 Introducing the Distributed Management Framework for Self-organizing Networks

Following the above guidelines we propose a Distributed Management Framework (DMF) to handle the challenges of the future networks. The main feature of future networks is considered their self-organizing capability, which is a concept that wraps all other requirements towards them stem from their heterogeneous and dynamic nature. This means that networks should be able to react to the changes induced by the intrinsic properties of the member nodes and to the changes occurred in their external (inter-networking) environment. This is the reason why we focus on the adaptation of the DMF to the self-organizing networks (SON).

As the proposed DMF for SONs is placed in the context of network compositions, it should determine the ways the management domains (authorities) interact with each other to establish cooperation. Figure 25 below depicts the architecture of the proposed DMF for SONs [J3][J7].

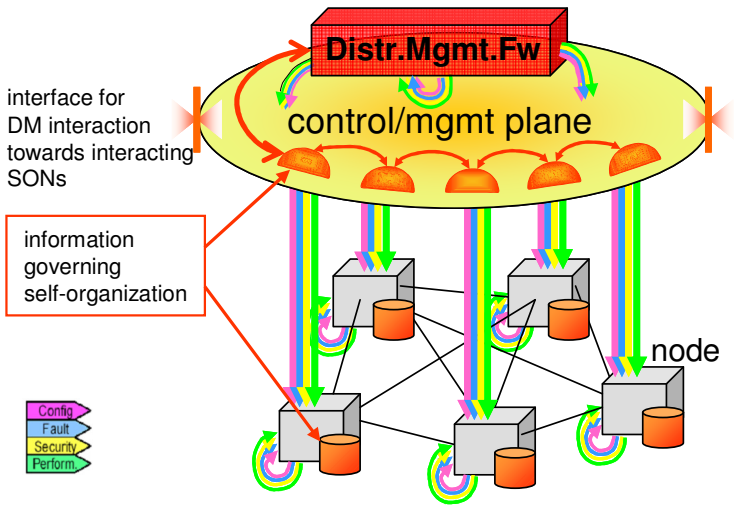

Figure 25. Distributed Management Framework – overview

The difference compared to the self-organization of networking elements depicted in Figure 25 above is that the DMF organizes itself and the control plane of the network, as well. Networking elements (i.e., nodes) keep their self-organizing properties, but they are combined at a newer level, the level of combined control- and management plane. We look at the control and management planes as complementary parts of a network. In future networks the border among these two planes is blurred.

In current understanding the control plane functionality is related to establish, maintain, and tear down connections (basically refers to signaling in the network), and the management-plane for the "rest", or which can be thought of following the FCAPS (Fault, Configuration, Accounting, Performance, Security) structure of TMN (telecommunication network management) [6]. In this respect, there is a potential overlap between control plane and management plane, in that connections may not only be established "on demand" via signaling procedures, but also "provisioned" via network management operations. We tried to avoid the uncertainties from this overlap through assigning the “operational” decisions to the control plane, while keeping the “strategic” decisions in the management-plane.

Note that we place the distributed information governing the self organization of the network into the control and management plane. This information is distributed in order to confer scalability and robustness to the architecture. The information is drafted from the per-node information (as depicted in Figure above by the “database” icons placed near each node), thus the necessary information to establish cooperation at the management plane by the DMF is self contained in the network, it does not require external information [J3][C5].

As explained later, we do not restrict the membership in a network to nodes (e.g., physical routers), but we allow networks of networks, as well. If a network has network members, as well, not only nodes, then this process are iterative, in the sense that information from nodes is combined in the network they are direct members. Then if this network is a member of a different network (let us call it higher level network), the combined information is further combined with the ones drafted from the other members of the higher level network, and so on. Figure 26 below depicts a case when a DMF manages a domain, which is formed on top of two other management domains. The parallel arrows illustrating management control point towards both the member domains, as well as towards its own domain. Nevertheless, the DMFs in the member domains do take over the management task within and below the member domains. The information governing self organization is drafted from members, similarly to the case illustrated in Figure 26 below, even if now the members are not nodes, but domains themselves.

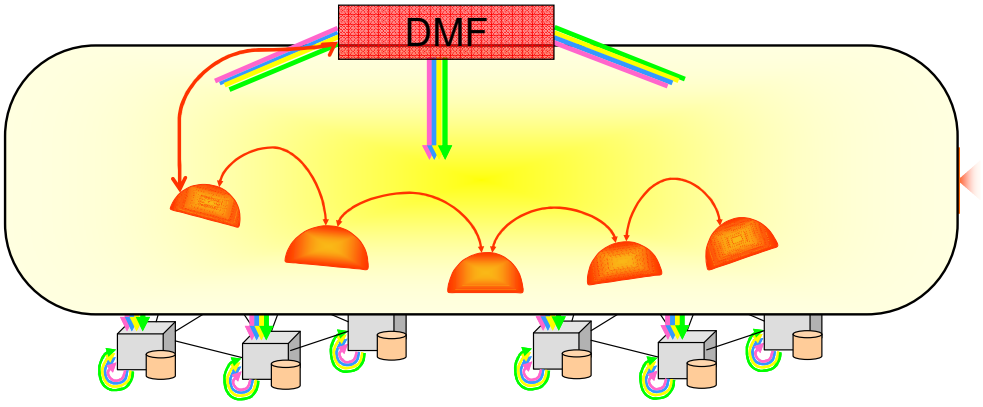

Figure 26. Distributed Management Framework of two composed SONs

In order to make a step from self-contained information towards self-organization, we need to establish the way of drafting the network-level information from the members of the networks. We call this a scoped process as it is focusing on the involved networks/nodes alone and only the management functions and functionalities that are just negotiating their cooperation. In order to interacting networks be able to decide (or *negotiate*) their cooperation, the necessary distributed information needs to be localized and disseminated, which in practice means lookup, retrieve and transfer of information to the negotiating party [B1][C5].

The management framework introduced above is solely responsible of the management processes that lead to the self-organizing behavior of the network. The interaction of cooperating networks is possible through an inter-networking interface. The DMF, besides the management processes, it also controls the behavior of the control plane functionalities of the management domains. Following the responsibility split between the control and management planes, the short term control plane processes are organized (created, enabled, terminated) by the DMF.

9.2 Management dynamics

The dynamism of the DMF is coming from the interaction and cooperation between the SONs. As introduced earlier in this Chapter, the interactions are done through compositions. Therefore in this section we look in detail to the compositions. We define the composition types and we show several examples for better understanding of the concept.

We defined and described *absorption* and *gatewaying* as two types of network composition types, which together with *bootstrapping* (initialization) and *de-composition* (termination) can fully describe a network composition model and is conform to the DMF for SONs architecture presented in the previous section [J4][J6][C7].

Absorption defines the *unconstrained information* sharing of the composing management domains, resulting in one single management plane as the union of the former ones (both resource and service wise). If absorption is not possible then *gatewaying* defines how to create a new management plane for the composed network and how to disseminate information from the embedded networks. Throughout gatewaying full control of resource and service sharing is possible.

Bootstrapping defines the basic self-configuration process that initiates a stand-alone management plane (e.g., powering up a stand-alone networking element). *Decomposition* defines the process of division of a management domain into two or more standalone management domains, which implies the identification of the self organizing properties and the proper separation of them during this process.

We have assumed throughout this Dissertation that during the network composition *control* over servicers and resource are restrictive, i.e., for each *gatewaying composition* the resulted network bears with less and less control over their embedded resources.

In DMF SONs network composition processes should work automatically without user interaction (as far as possible) and should be based on the predefined preferences of the users.

The SONs may deploy a policy framework to govern their self-organizing behavior. Although it is not integral part of this Dissertation, note that we have proposed a generic policy framework that is able to fulfill this task. In the simulations in next Chapter we will not use this policy framework. On the other hand we used it in our prototype testbed as explained in section 10.2. A detailed description of this generic policy system can be found in Appendix D. According to this policy framework users define their preferences by policy rules. These rules can be either strict rules that express explicit requirements of the user (e.g. a user belonging to a company wants to cooperate only with users of the same company) or more permissive rules that reflect only the wishes of the user (e.g. the user prefers networks providing location dependent services, so if there is more than one suitable network, the user will select the network providing the most number of location dependent services).

Users form networks with users that have similar (or at least mutually acceptable) policies. When the SON is formed, the overall policy reflects the common preferences of the community. This common policy should be continuously maintained at all levels according to the current policies of the current members. The common network policy defines the preferences of the community forming the network. This policy is used when different networks try to interconnect. Based on the policies of the two networks, the joining process may have very different outcomes.

The merging process is conducted by the super-peers of the networks. As described in [C6], every network has a superpeer acting as a representative of its network. It is responsible for negotiations with other networks and it has to decide whether the policy of the other network is acceptable by this network. Therefore the network composition process begins with the negotiation of the superpeers of the networks.

9.2.1 Absorption

The network composition process may finish in different ways. If the policies of the two networks are close enough to each other (there is no contradiction in the preferences and the differences between policy rules can be accepted mutually), the networks can join in a way that is called absorption. Absorption is the full merging of the two networks. They will form a single management domain with one common superpeer [J6][C7].

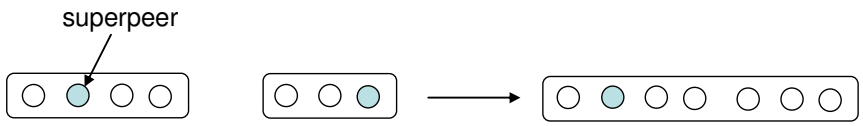

Figure 27. Absorption

The superpeer of the new network can be elected by super-peer election process (i.e. every node can be a candidate for the super-peer role) or one of the previous super-peers can remain the super-peer of the whole network. For convenience and for lower maintenance overhead the latter option is selected currently in our work.

The policy information of the new network will be merged from the policies of the former networks. These policy rules will express the preferences of the whole network, the common preferences of the members.

9.2.2 Gatewaying

In case the absorption of the networks is impossible according to the policy rules, there is a trade-off option between full absorption and full separation. This special partly joined way of operation is called: gatewaying [J6][C7]. The networks connected with gatewaying are not separated networks; they are composed networks but they do not share all information. A third network might interact with this common network or the original one, depending on the scope of interaction (e.g., what management services it want to reach).

Therefore, apart of the original networks a new network has to be represented. This new network has to be represented by one superpeer and one consistent policy database. Therefore, the superpeers of the joining networks will create an upper overlay level. This new level is a new SON. The members of this network will elect a superpeer that will represent the two networks, interconnected by gatewaying, to the outside world.

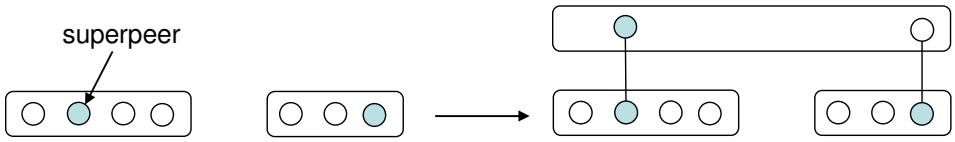

Figure 28. Gatewaying

In gatewaying the construction of the common policy database is more difficult, because there can be contradiction between the policy rules of the networks. Because the common policy database should express the common preferences of the community, the contradicting policy rules must be eliminated from the common policy. Therefore the joined network and the whole community is described by the common policy and represented by the highest level super-peer, but the involved networks are represented by their own super-peer and their own policy.

9.3 Design of the DMF

In order to realize the DMF architecture it must be mapped to a distributed protocol structure, as explained in this section.

9.3.1 Peer-to-peer Overlay

Peers of an SON share a common distributed control space and form a single management domain. An SON can be as small as a PAN or may even consist of a single network node but large networks of telecom operators may also be considered a SON. Control and management plane communication between SONs is done through a generic SON Interface [J5][J6].

The other most important feature of SONs is instant and automated network composition: SONs compose, decompose and recompose dynamically with each other in an ad hoc manner. In order to create a scalable system to manage SONs, the platform organizes SONs into overlays in the course of composition processes. Figure 29 shows the overlay structure of an example network.

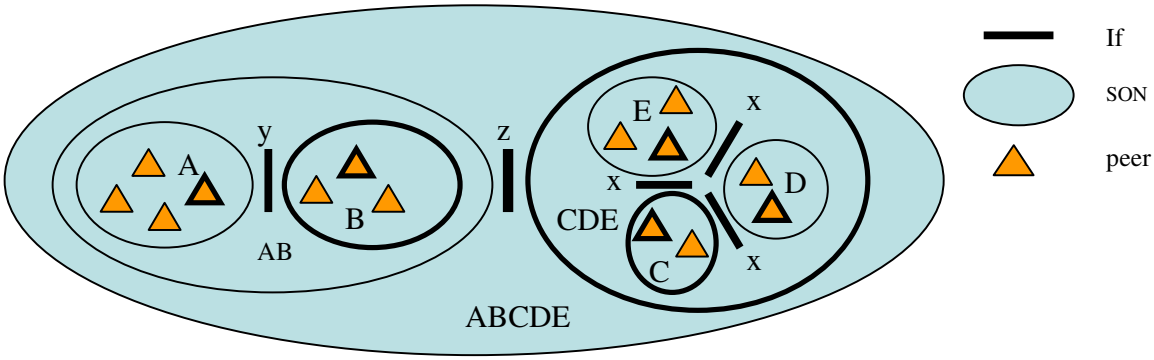

Figure 29. Peer groups and SONs

The basic components of this management overlay network are peers, superpeers and peer-groups. A peer-group is a set of peers forming a common management domain. Thus a peer-group is roughly equivalent to the SON. However the notion "peer-group" will be used mainly to describe overlay network structure, while the "SON" notion is rather used in network management contexts (e.g. to describe network composition).

Network management within a peer-group is distributed; there are no dedicated servers to manage relationships between peers of the group. Each peer-group elects a representative called superpeer to negotiate with other peer-groups. Logically, each peer group has one and only one superpeer. However for fault tolerance and fast recovery after superpeer crashes, superpeer management information is distributed within the group [126][125].

Superpeers may also form new peer-groups, thus create an overlay of overlay. The details of these structures and the way we will represent them are discussed later in this section.

Intra-SON management is distributed among peers. However, for inter-SON management purposes, each SON designates its super-peer. Thus inter-SON management is realized through superpeer-to-superpeer interaction. In Figure 29, super-peers are marked by thick borders. Several SONs can be grouped (or composed) into a new SON. This new SON can be considered an "overlay" above the first SONs. E.g., in Figure 29 SONs *E*, *C* and *D* are composed in the new SON *CDE*.

One peer already elected as super-peer in a stricter peer group may be superpeer in the broader level SON, as well. That is, the given peer is super-peer in multiple SONs. For example the super-peer of SON *C* is also super-peer of the upper level SON *CDE* and the top level SON *ABCDE*. The super-peer of SON *CDE* is elected from the superpeers of the composing SONs *E*, *C* and *D*.

In the overlay representation, each SON, that the peer is member of corresponds to an overlay. Thus peer *C* is member of overlays *C*, *CDE* and *ABCDE*.

As mentioned before, inter-SON management is performed by super-peers in a peer-to-peer manner. Super-peers communicate with each other through SON Interfaces. Figure 29 shows that each overlay has its own SON interface ("If"). In the previous example, the super-peer of the SON *C* will use SON Interface "x" to exchange management information with SON *D*. However, peer *C* as super-peer of the upper level SON *CDE*, it will use the SON Interface "z" for inter-domain management with SON *AB*.

Note that even if management tasks are performed by a dedicated peer (the super-peer) in each SON, the Management Information Base is stored in a distributed and redundant manner within a group (see next section). Thus whenever a superpeer crashes, another peer from the SON can immediately take over its role.

Peers with multiple network interfaces may also belong to several SONs not only at different hierarchy levels, but also on a per interface basis. Peers may be part of a separate SON at each of their network interfaces, but this is not mandatory, as peer groups assigned to different interfaces may also merge during composition process. An example scenario of separate overlays per interfaces could be demonstrated by a laptop being part of a WLAN network via its 802.11 adapter and also belonging to a PAN via its Bluetooth card. In this example, the basically different nature of the two access technologies makes it reasonable to handle the two networks as a separate management domain. Another useful area of application of per interface overlays is multi-homing.

Dynamics of the management - motivation

As just explained above, we model can be modeled and realized as a P2P overlays. In order to examine the implications of this choice, we discuss its several aspects below.

A main feature of P2P is the immediate interaction among *equal partners* that are called "peers." In P2P architectures, the networked nodes are highly *autonomous*: peers may leave or join a P2P network arbitrarily. This is similar to the SON environment where management stations of different domains may come and go in the user's domain perspective as a user traverses across domains. Another major characteristic of P2P service [115] is that the services provide a simple and efficient mechanism to pool and *share exchangeable resources* like disk space, audio/video files, or CPU cycles. These features allow any peer to be removed without resulting in any loss of service. Similarly in SON management, management information must be shared between homogeneous/heterogeneous management stations to enhance network composition and self-management. Another feature of P2P networks being similar to SON networks is that they are both de-centralized. Architectures of networks vary between two extremes: *pure P2P* and *pure client/server* distributed architectures; a pure P2P architecture is completely decentralized. Peers operate in a highly autonomous mode and are de jure equal entities. De facto, however, if left to themselves, most P2P architectures evolve into a *hybrid structure*, where some peers are "more equal than others." Hybrid structures may form themselves due to preferential attachment of peers to distinct peers. An involuntary instantiation of structure can be based on forms of application-level attractiveness of peers (e.g., content or metadata), or on network-level attractiveness of peers (e.g., bandwidth). Thus, P2P services are inherently distributed, and can be categorized according to the degree of decentralization incorporated, both with respect to the implementation of control and the location of shared resources.

P2P networks are operating in very large-scale environments under (in most cases) un-predictable conditions. Peers may come and go as often they like in a large system and so form highly variable topologies.

Similarly in an SON environment, end users move across different domains, resulting in dynamic network de/composition. As a consequence, in the users' point of view,

management stations of different domains are also autonomous: they come and go as they traverse, which is very much the same case in P2P. Thus management stations in SON have a highly variable topology. Both P2P networks and SON networks are expected to be large-scale. As a consequence, any management approach that suits P2P networks is highly adaptable to SON networks. The self-organizing aspect of P2P management provides valuable contributions towards a self-management system for SONs.

Another problem hindering the formation of compositions is the lack of applications that support the automated creation and administration of the composed network. The creation process includes the finding of adequate SONs, negotiation among them, the definition of business relations, the collection of configuration information and requirements, and the reservation of appropriate resources in the network infrastructure.

All these aspects motivated us to combine P2P overlays into a hierarchical structure, as detailed in the following.

Hierarchical structures

Hierarchical structures are commonly used methods to create scalable systems in a number of different networking areas like for example routing. The most important feature of our hierarchical management overlay network is that it is created dynamically and unlike in the above example for IP routing, it does not require network administrator interaction [C4][C6][J6].

Motivated by our observation above, we introduce a hierarchical structure, where peer groups at a given level form a new overlay, which is represents them in an upper level in this hierarchy. We already have introduced the elements of the overlay earlier in this section 9.1.3, these being the *peers, super-peers* and *peer-groups*.

We recall our representation of the SONs as p2p overlays, as depicted in Figure 29. Following that logic, but omitting the illustration of the interfaces and assigning a name to each node (that is peer in the bottommost layer) we give the example of a network in Figure 30. The larger circles in the figure show a peer group (thick lines mark super-peers). We illustrate over this little example the way we structure the SONs into a hierarchical network structure.

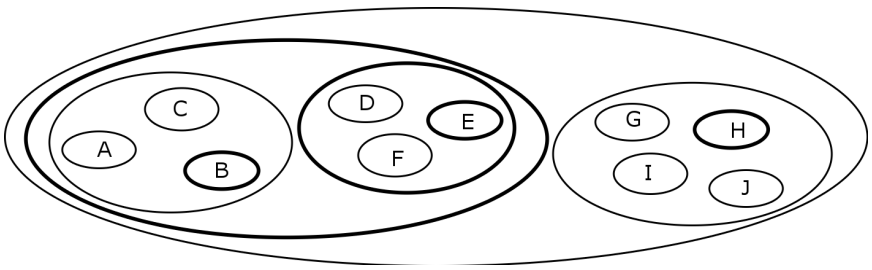

Figure 30. Hierarchical overlay network formed by peer groups

In this example network, both peers *A, B, C* and *D, E, F* form peer-groups. Also, superpeers of these two groups – *B* and *E* – form a higher level peer group. This representation is good in showing the overlays, but is not intuitive in expressing relations among the different overlay. For this purpose the tree view of the SONs is much more appropriate. Figure 31 presents this tree view of the same structure.

Using Figure 31 we can explain three important properties of the proposed hierarchical structure. First, the number of hierarchy levels in the network is not limited: one peer can be part of multiple peer-groups at different levels of the hierarchy (e.g. peer *E* is part of three different peer-groups at different overlay levels).

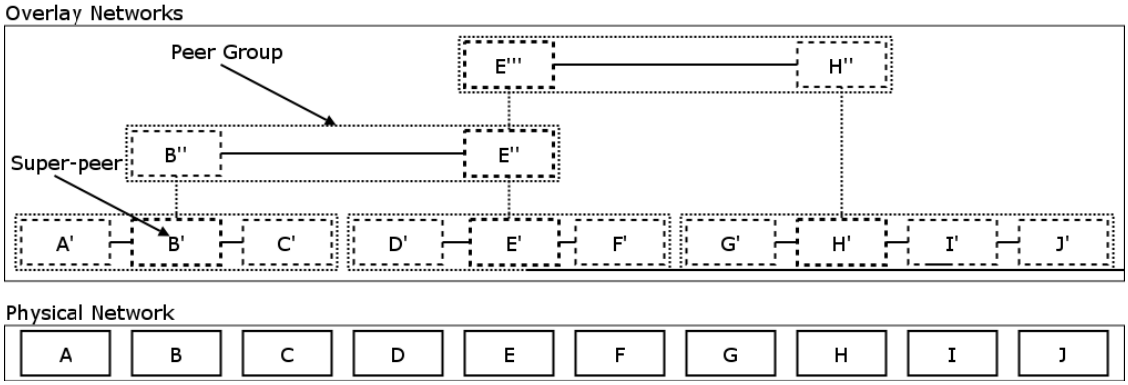

Figure 31. Hierarchical overlay network (tree representation)

Second, hierarchy levels are not absolute. This means that we can't assign an absolute hierarchy level index to a peer group (see again Figure 31, where the top level peer group comprises one peer at the 3rd and an other at the 2nd hierarchy level).

Third, a so called bottommost overlay is defined for all peers. This bottommost overlay is a dedicated one, directly related to the "real" physical network (practically this means data link layer (layer 2) level neighborhood). Two peers can be neighbors at the bottommost level peer group only if they are layer 2 neighbors. This restriction ensures that all members of a peer group can communicate with all other peers of the group (directly or through other peer group members). If the group is split into distinct parts that cannot communicate with each other then it can not be considered to be a common management domain. However layer 2 neighborhoods do not automatically imply neighborhood relationship within the bottommost overlay group. For example two peers belonging to two different competing operators will not establish neighborhood relationship in the bottommost overlay network even if they are layer 2 neighbors.

These restrictions do not apply for upper level peer-groups because the "Bottom-Up" network composition approach (see section 10.1.1) automatically ensures correct neighborhood relationships.

Dynamics in the overlay

Departure of peers or decomposition may result in hierarchy structure in which some peer-groups consist only of a single peer (see Figure 32).

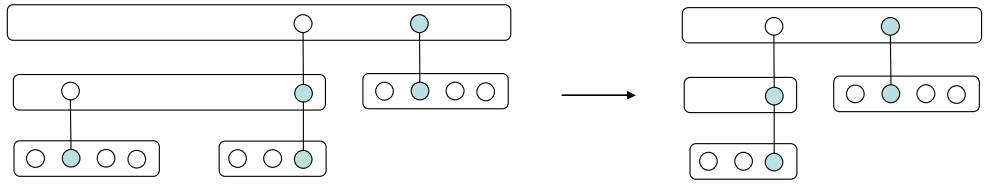

Figure 32. Abnormal overlay structures after departure of a peer-group

These single member groups present no problem in bottommost level peer-groups: some peers might want to manage their composition on their own without having to agree with other peers in the group. However at upper hierarchy levels, these peer groups are redundant. Taking again the example network presented in Figure 32, the single member peer group introduces an unnecessary intermediate level decreasing efficiency and breaking the logical structure of the hierarchy. But there is an even more serious problem: if the single peer in this group fails or leaves the network ungracefully, the structure will be broken.

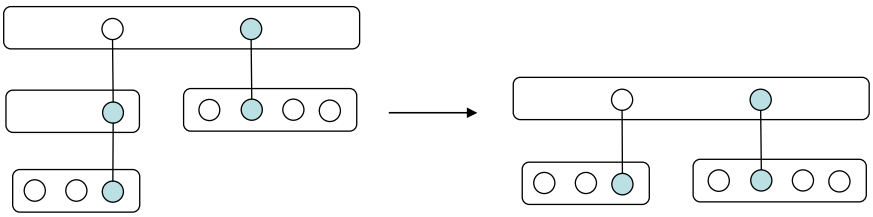

Figure 33. Repairing abnormal overlay structures

Therefore every time the number of peers in a group decreases to one due to a decomposition or departures, the overlay level corresponding to this peer-group should be deleted and the structure should be collapsed (see Figure 33). There is one exception: at bottommost overlay levels, single member peer-groups are allowed. Note: when deleting single member peer-groups, references to parent and child peer-groups should be updated both in groups below and above the deleted group.

9.3.2 Storing and maintaining overlay network structure

Maintaining the overlay structure in a highly dynamic distributed environment presents many challenges. Super-peers cannot be treated as fault tolerant highly available servers: they may fail or leave their group ungracefully at any time. Therefore network structure and topology information of a peer-group is stored in a redundant manner evenly distributed among group members [C8][C9].

To describe overlay topology, peers and groups need to be identified. Peers in the overlay network are identified by peer IDs. These are permanent cryptographic host identifiers derived from the public key of the host using one way hash functions. Unlike IP addresses, these IDs are permanent and only serve as host identifier, not locators. In this way, they are very similar to Host ID Tags (HIT) used in the Host Identity Protocol [132] and HITs could actually be used as peer IDs. It is important to note that even if a peer is part of multiple groups at different overlays and different hierarchy levels, it is identified by the same peer ID in all of these groups.

Group IDs present more difficulties: These should also be globally unique identifiers, reflect somehow members of the group and to reduce overlay maintenance overhead, they should not change too often. It is impossible to use permanent group identifiers as every time two previously separate groups merge by absorption, at least half of the peers should change their group ID. On the other hand, if all peers of the group were represented somehow in the group ID, then this ID would change every time a new peer joins or leaves the group causing very high overlay structure maintenance overhead. As a trade-off, we decided to incorporate only the peer ID of the super-peer in the group ID. This design decision has one more important benefit: inter-group communication is usually taking place between super-peers of two groups, thus only by knowing the group ID of the other group, it is possible to extract the ID of the super-peer of the group. To ensure uniqueness of group identifiers, beside super-peer ID, they also contain a randomly generated identifier (long enough to be globally unique with very high probability).

Generally speaking, the overlay topology database describes relationships among peer IDs and group IDs. The following paragraphs present in detail these relationships stored in every group by the distributed overlay topology database.

First of all, every group should maintain a list of its members. Besides peer IDs, this list should also store for every peer the group ID of all of their lower level groups (the group that the given peer is super-peer of at the next lower level). These lower level groups correspond to children in tree representation. Note that such child groups do not exist at bottommost overlays. Groups should also store the group ID of their parent (the group that the super-peer of the group is member of at the next upper level). Of course, top level overlays do not have parent group.

During normal operation these relationships provide enough information to navigate anywhere in the overlay structure. However, fast recovery from super-peer failures might require further data structures. Let's take the example overlay structure shown in Figure 31. If peer *B* fails, then the remaining peer group *A'C'* will elect a new super-peer and after the election, the new super-peer recovers the group member list from the distributed redundant topology DB. Finally it notifies its parent group *E'''* that it is the new super-peer member replacing the previous representative. However, if instead of *B* it is peer *E* which fails, then the situation is somehow more complicated as it is super-peer of the overlays *D'E'F'*, *B''E''* and *E'''H'''*. All three groups will elect new super-peers, and restore their member list but they will not be able to refresh parent-child relationships as group ID changes after super-peer election in all of the three groups. To solve this problem, groups also store the ID of all groups being member of the parent group (i.e., the IDs of their brothers). For example, if peer *E* fails in Figure 31 fails, then both group *B''E''* and *E'''H'''* will change their group IDs, however group *A'B'C'* will remain unchanged. In this way, children group *A'B'C'* is notified about its parent new group ID. After this procedure, all parent-child relationships will be restored.

9.3.3 SON Graph

The peer-to-peer network at its lowest layer consists of the physical network nodes (members of the bottommost overlays). For each of the management domains in the DMF an overlay is created, whose members are the elected representatives (e.g. superpeers) of the embedded p2p overlays corresponding to their management domains. We have further created a SON graph model together with six operations to model my DMF [C4][J6].

We define a physical topology graph *(C)* that models the link level connections of an IP network. This is an undirected graph, where the vertices *V(C)* correspond to the network elements and edges *E(C)* to the direct link-level connections.

We propose a hierarchy tree *(H), called SON graph* over this graph that covers the hierarchical domains in the above network. The leaves of this SON graph correspond to the vertices of *(C)*, *leaves(H) = V(C)*. The vertices selected from *(C)* to correspond to leaves of *(H)* form a connected sub-graph in *(C)*. This means that we define *(H)* over non-partitioned network *(C)* (or a partitioned network *(C)* is covered at least by two hierarchy trees *(H)* and *(H')*). All the vertices of *(H)* that are not leaves (do not correspond to physical networking elements) are virtual nodes: *V(H)\ leaves(H)*. Over a non-partitioned network *(C)* we may have a hierarchy tree *(H)* with multiple roots.

In *(H)* there are no absolute hierarchy levels (the distance of a leaves to the root is not bounded and it might be different within the same tree *(H)*). There is no constraint on the number of hierarchy levels the vertices should fit into.

Dynamics of the management

This SON graph gives a direct mapping between the distributed management framework and the peer-to-peer overlays as follows: each vertex in the SON graph corresponds to one management domain and one overlay network [C4][C5][J6]. This overlay network is run by the representatives of the overlays corresponding to the children vertex.

We have mapped the constructors for the network compositions as defined in section 9.2 into the SON graph model using the six operations: joining, leaving, merging, splitting, expanding and collapsing.

Join: The root node of the joining SON graph is attached to an existing node of the other SON graph (see Figure 34.i).

Leave: A complete subtree is detached from the original rooted tree. The result is two disjoint SON graphs over the same topology (see Figure 34.ii).

Merge: All the vertices belonging to the parents will be placed under a single parent, the other parent being deleted (note that only the leafs represent physical nodes, the rest of the nodes are virtual ones). There exists a single node in the SON graph into which all child nodes of the root node of the other SON graph will be attached as children nodes (see Figure 34.iii).

Split: Some children detach from a parent node and create a new parent node to group them all as children nodes (see Figure 34.iv).

Expand: If the node had a parent before, then a new node between the parent and the child node is created. Otherwise the node will create a new parent of its own (see Figure 34.v).

Collapse: A node that has a parent and has exactly one child is deleted and a direct edge is created between its former child and its former parent (see Figure 34.vi).

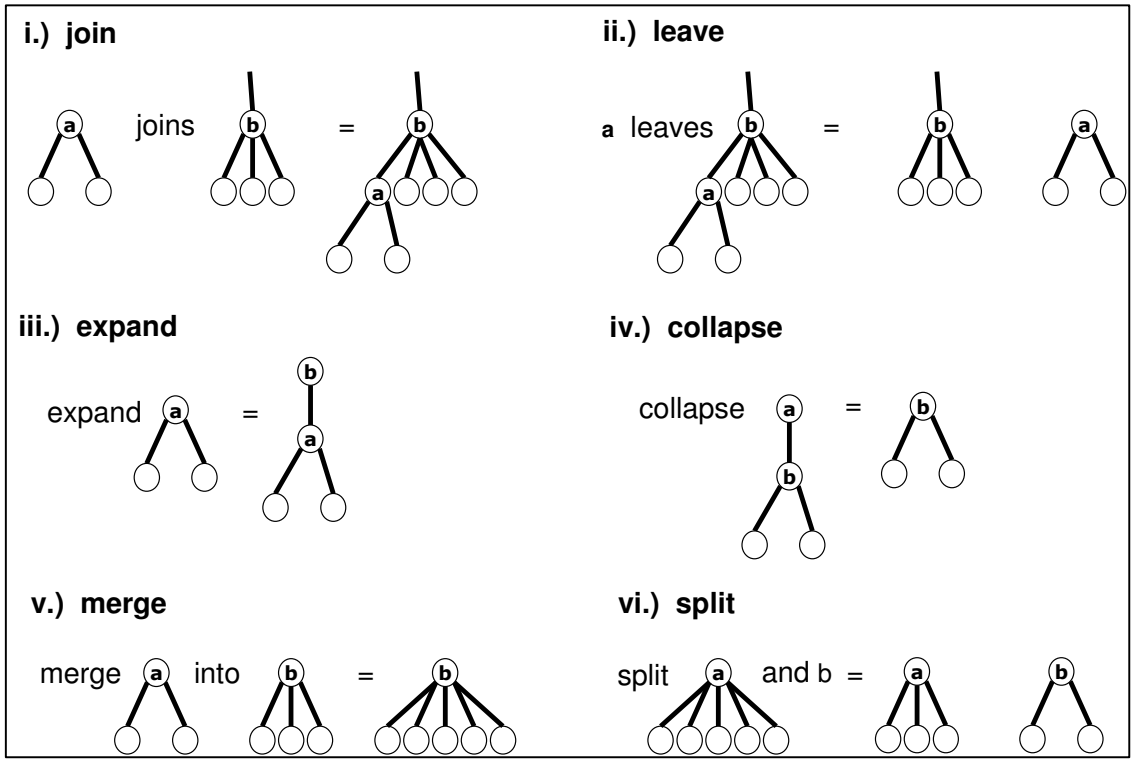

Figure 34. The six graph operations

Note that applying these operations to the SON graph it keeps its properties, and it remains a rooted tree. The SON graph will split in two parts if the underlying topology is.

9.3.4 Mapping peer-to-peer overlay and SON Graphs

We have introduced the SON graph, now we present the three way mapping between the layer 2 connectivity matrix of the physical nodes, the SON graph and the p2p overlays of the domains. Figure 35 below illustrates this mapping. There is a close relation between the bottom-most layer of the SON graph and the topology graph of the network. Note that the layer 2 connectivity of the nodes is not taken into account directly by the SON graph. Nevertheless, the connectivity among nodes is affecting indirectly the SON graph, as it is taken into account during compositions [C4][C5][J6].

A different mapping is between the overlays and the SON graph. Each vertice of the SON graph is expanded into an overlay. We illustrated this direct mapping by the assignment of the same numbers assigned to the vertex (in the graph) and to each overlay. Note that each overlay is represented by its super peer. Also the overlays can be mapped to the physical network. This makes possible that starting from the abstract identification of a SON (a vertice in the graph) we can reach to the overlay that realizes this SON. The

overlay also gives the practical details, like the member nodes of the overlay, the representative (super peer) of them and the relations among the peers (members).

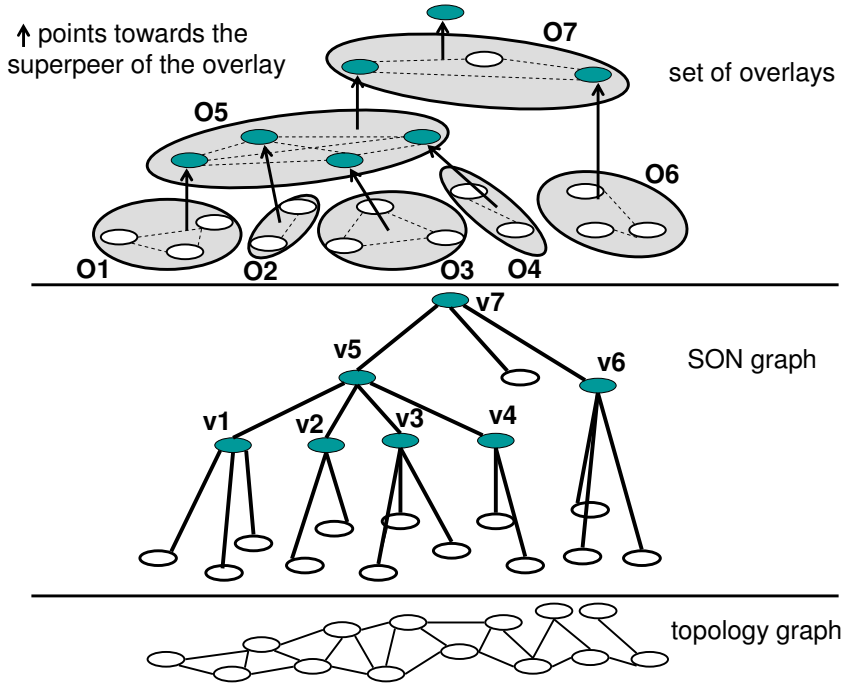

Figure 35. SON graph and the p2p overlays

Up to now we introduced the DMF for SONs and presented the composition types that describe the SON-interactions. We also gave a graph model that describes the relations among SONs. Nevertheless, the logic behind the composition process is not defined yet. In the next Chapter we propose a new algorithm and its evaluation that controls the composition process based on the utility of the attributes of each composing SON.

Chapter 10 Algorithm and Evaluation

10.1 Bottom-up composition algorithm

10.1.1 Objectives

Network self-organization is an important network management task in dynamic environments such as ad hoc networks. We defined network self-organization as the process of creating and maintaining a logical network structure on top of a dynamically changing physical network topology. This logical network structure can be used as a scalable infrastructure by various functional entities like address management, routing, service registry, media delivery, etc.

A formal method to describe static and dynamic behavior of hierarchical self-organization models has been described in the previous Chapter, but it doesn't specify the actual algorithm and the logics behind it. This section presents a self-organization algorithm that is proposed to help managing dynamic networks. The algorithm is based on grouping nodes with the largest common subset of properties, instead of grouping them with most compatible policies. The two self organizing principles based on common property subsets or compatible policies are quite similar, as both are used to describe restrictive composition processes.

Property set and policy based self-organization models

The property set based model tries to create a hierarchy by grouping together nodes with most possible common properties. Each physical node has a set of properties while property set of upper level tree nodes in the hierarchy is created as the intersection of property sets of their child nodes (thus consists of properties common to all children nodes).

Property sets

The abstract term property denotes an arbitrary key-value pair. A property can represent multicast membership, preference to use or provide a specific service, QoS requirements, the fact of being part of security or administrative domain.

Most of network related properties require aggregation of nodes having the same property into one group (ex. create a multicast group or form a security domain with nodes belonging to the same authority, etc…). The most important benefit of using property set based self-organization consists of providing a natural, easily understandable way to guide such interactions.

Property set vs. policy based model

The policy based self-organization model (e.g., the one presented in Appendix D) is a generalization of the property set based model. In policy-based algorithms, property sets are replaced by more generic policies allowing specifying rules and dependencies between these rules. The concept is very similar, but instead of maximizing the number of common properties inside one group, the aim is to group together nodes with most compatible policies.

Flexibility of the policy based model fits better the needs of SONs, however, for the sake of simplicity, functionality of the property set/policy based self-organization model will be presented using the property set based model.

10.1.2 A Bottom-up heuristic

The network composition process works in a "bottom-up" manner [C4][J4]. When a network meets another network and wants to join that network, it tries to join on the lowest hierarchy level. This means that the super-peer of the joining network will negotiate with the lowest level super-peer of the network to which it tries to join to. In this section we explain the bottom up approach with the help of policies. This means that we suppose that a generic policy framework exists, which can govern the self-organizing behavior of the management domains. Although it is not part of this work, note that we have drafted such a policy framework as presented in Appendix D.

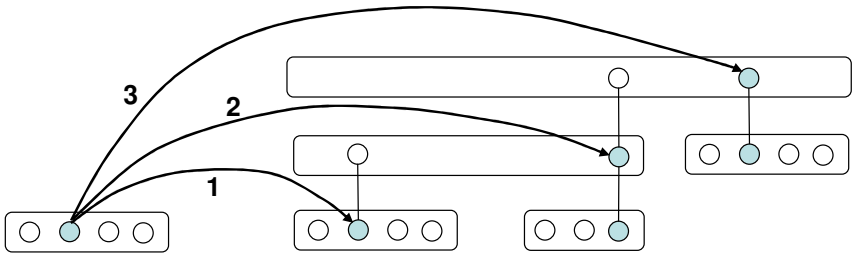

Figure 36. Bottom-up approach

If the negotiation is not successful it tries to join on the upper level and so on to the highest representative of the network. The process when, after a unsuccessful negotiation the composition negotiation is transferred to the upper layer of the hierarchy is referred to as "send up". Closed networks (e.g. enterprise networks) may define in their policy that any foreign network should join the highest level because of security reasons.

When a new network joins on a lower level it can modify the policy of the upper level overlay networks. In this case the upper level policies have to be refreshed according to the policy of the newly joining network.

Note that in a distributed network the nodes and networks may get into contact with each other in an uncontrolled manner, overloading the nodes with a huge computational burden. In order to propose a feasible way to compose networks in a global distributed environment, first we have drafted use cases and scenario that these composition mechanisms should be able to handle. Appendix E presents the uses cases.

Based on the above considerations we opted for the following guidelines. First, during compositions the smaller network is trying to compose with the different networks situated on the hierarchy levels of the larger one. The term "smaller" can be interpreted in many ways. First, according to the depth of the SON graph (i.e., how many hierarchy levels are between the root and the bottom most node that initiated the composition). A second interpretation could be based on the number of nodes within the network. Apart of these two, several other metrics could be introduced by the operators (e.g., economical considerations), which are not investigated in this work. From our point of view the important thing is to be able to decide which party will be the "smaller one". As later

explained, in our assessment of the proposed mechanism, the worse case scenario is the one where more hierarchy levels should be checked. That is the reason why we opt for this interpretation.

Based on these considerations we have proposed a bottom-up composition algorithm, which walks through the SON graph from the leaves (physical nodes) to the higher levels (wider and wider scale composed networks) and checks for possible *absorptions*. If no absorption can be realized then the point of *gatewaying* is determined (see the Bottom-up Algorithm description below).

Bottom-up Algorithm

Note that the bottom up algorithm can explicitly be realized by the super peers of the p2p management overlays. In a formal description, the bottom up composition principle states that the composition or de-composition is a succession of peer-to-peer interaction (represented by nodes of the SON graph) starting from the leaves (i.e., bottom-most node) of the interacting SON graphs. Every interaction ends up with a decision. The decisions are based on a utility function, which is a decreasing function over one SON from bottom to up. If an attribute match is found between the source and the target network level then absorption is executed. If no proper match is found then a new level (abstraction) is created to accommodate the best possible mix of the two networks [C4][J4][J6][C7].

Let us take the physical nodes *a* and *b,* and let us compare the SON graphs which they are members in based on their depth, as explained earlier. Without the loss of generality, let's assume that the SON hierarchy level of node *a* is lower than for node *b*. Let's denote with *a* the composing SON and *b* the target SON. Let's define a utility function (U), which counts the population of the attributes two SON nodes as follows:

$U(a,b) := \#[A(a) \cap A(b)]$, where $A(.)$ is the set of attributes at the overlay layer.

Now, the flowchart of the bottom-up algorithm that implements the bottom-up composition principle is given in Figure 37.

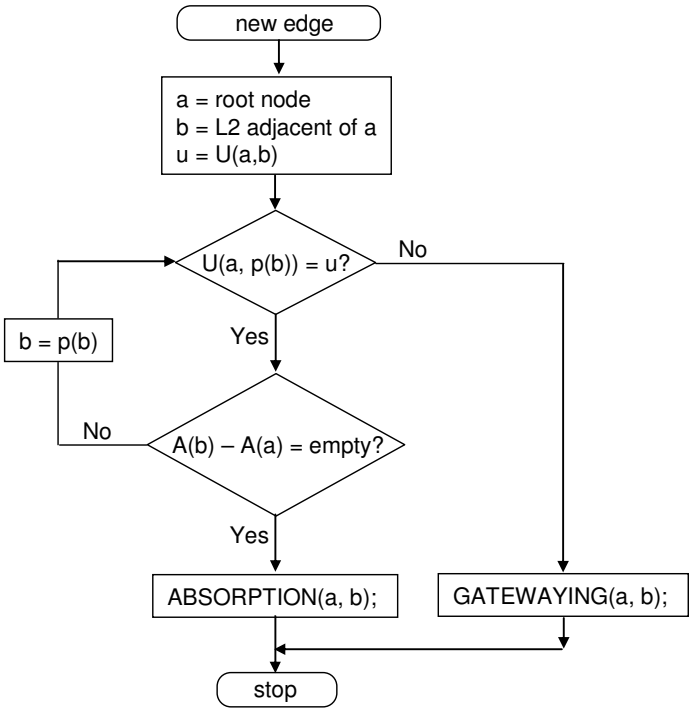

Figure 37. Bottom-up composition algorithm

The bottom-up composition principle states that the composition or de-composition is a succession of peer-to-peer interaction (represented by nodes of the SON graph) starting from the leaves (i.e., bottom-most node) of the interacting SON graphs. Every interaction ends up with a decision. The decisions are based on a utility function, which is a decreasing function over one SON from bottom to up. If an attribute match is found between the source and the target network level then absorption is executed. If no proper match is found then a new level (abstraction) is created to accommodate the best possible mix of the two networks [C4][J4][J6][C7].

The pseudocode of the bottom-up algorithm is given below.

Let *p(x)* denote the parent node of *x*, if *x* is the root node than *p(x) = 0*.

- *a :=* root node of the source SON;
- *b :=* layer 2 adjacent leaf node of the target SON;
- *u := U(a,b)* // *a*=root; *b* = leaf
- while (*U(a, p(b)) == u)* do //
    - // check for matching attributes
    - if *(A(b) – A(a) == empty)* then *ABSORPTION(a, b)*; exit;
    - *b := p(b)*;
- end while
- *GATEWAYING(a, b)*

ABSORPTION(a,b)

- merge(*a*,*b*);

end

GATEWAYING(a,b)

- b = expand(b)
- join(a,b)

end

10.2 Prototype based Validation

10.2.1 Introducing Ambient Networks as a SON

The properties required by our DMF proposal for SONs suppose the availability of layer 3 addressing and routing mechanisms. It accepts and is able to handle and cope with the mobility and heterogeneity of the nodes. Without losing the generality of our proposal, we consider a generic networking concept that adds ambient intelligence to the nodes in a networking environment with a multitude of access technologies, network operators and business actors. We refer to ambient intelligence as a mechanism that makes technology invisible and useable, whenever necessary. Such a generic concept is expected to embrace the heterogeneity arising from the different network control technologies so that it appears homogeneous to the potential users of the network services. The concept allows the agreement for cooperation between networks on demand, transparently and without the need of pre-configuration or offline negotiation between network operators by introducing this transparent intelligence into the networks.

According to this concept, end-users are increasingly not just owners of a terminal or a PC, they own and effectively operate a network of devices in their homes, offices and around the body. Consequently, they are included in this network of cooperation and are treated as operators of special, low-complexity networks. This approach generalizes to different kinds of networks that are currently appearing, such as inter-vehicle networks, body area networks, or sensor networks. By making every device a network, the network is the primitive building block of our architecture, allowing all types of networks to be integrated into a larger system.

As discussed above, our proposal of DMFs for SONs can be seamlessly applied to such concept. On the other hand, if a solution to network management in such a generic networking concept, if the cooperation between networks is done in self-organizing and distributed manner and the self-organization is done according to the principles enumerated in Chapter 9 and the control- and management intelligence of the network is represented by a common control space, then this solution conforms to the DMF for SONs presented in section 9.2.

Such a networking concept is the Ambient Networks [133][B1][J7][C5]. In ANs all the control functions already identified and isolated by “All-IP” networks are extended into the Ambient Control Space (ACS), which embraces a well defined set of control functions required to guarantee the co-operation between networks. The Ambient Control Space (ACS) is responsible for the control and management functionalities of the network. As on can see, the ACS is the correspondent of the common control and management plane, as introduced in presented in section 9.2. There are minimal prescriptions how the ACS is realized, or what functionality it supports, just that it

belongs to a network, not to a node. In the followings we present the ACS, as the main architectural element that enables the DMF for SONs.

The ACS provides a domain-structured, edge-to-edge view of the network control and encompasses all control functions within a certain network domain. The ACS consists of a set of control functions supporting multiradio access, connectivity, mobility, smart media routing context management, security, (see Fig.2.4) [135][84][J5].

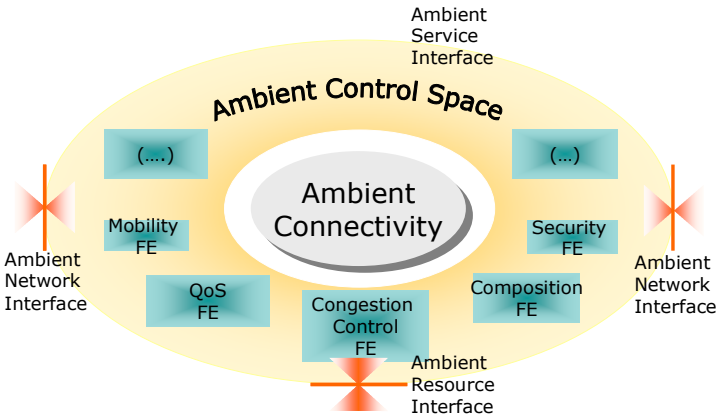

Figure 38. The Ambient Control Space

The ACS consists of two kinds of components: the actual control functions (the boxes in the control space) and the control space framework functions (not explicitly shown; implemented by the ellipse surrounding the connectivity plane.) The control space framework comprises all functions necessary to allow the control functions to plug into the control space, execute their control tasks and coordinate with other functions present in the control space.

The modules of the ACS that implement a specific control or management task are called Functional Entities (FE) [133]. The ACS raises a complex set of interdependent FEs, such as the basic functions for management, security and connectivity, as well as control functions for supporting mobility, multi-radio, resource management, composition control and smart media routing.

ACS exposes three interfaces: the Ambient Resource Interface (ARI) for communication with connectivity resources, the Ambient Service Interface (ASI) for interaction with services and applications and the Ambient Network Interface (ANI) for communication with other networks [137]. Recall that ANI realizes the functions of the SON Interface introduced earlier and depicted in Figure 29.

As the ANs around the world interact, they form and re-form new domains resulting in complex coordination and cooperation among them. This interaction process among ANs is generally termed as composition and we give a definition of it below.

Nevertheless, the control plane is dealing with operational aspects of functional entity (FE) interactions. Thus there is the need of the management plane component of the ACS, as the element that represents (and enforces) the strategy according to which the

short term, operational control functions are organized in the ACS, and implicitly in the AN [C4][C5]. In the following sub-chapter we introduce the DMF, a management framework that manages the ACS.

10.2.2 Ambient Network Management as a DMF

We introduced the AN as a global heterogeneous and dynamic network, and we proposed that its management should be managed as a complex interaction of self-organizing network domains. This proposal is the DMF for SONs, tailored to the ANs. The particularity of the ANSs compared to the SONs discussed in this and the previous Chapter is the ACS. As matter of fact the ACS offers intelligence and proper interfaces to the network and as such is an enabler that makes ANs self-organization capable, preparing them for DMFs. Starting from the ACS concept we proposed the *Ambient Network Management Framework* (*ANMFw*), which organizes the management of the ANs. The ANMFw is realized through Functional Entities (FEs), therefore we proposed new FEs, which sustains the self-organizing operation and autonomous management of the ANs. Two cooperating ANs form a new AN and this formation is realized by the proposed FEs of the *ANMFw* [J4][J6][C9].

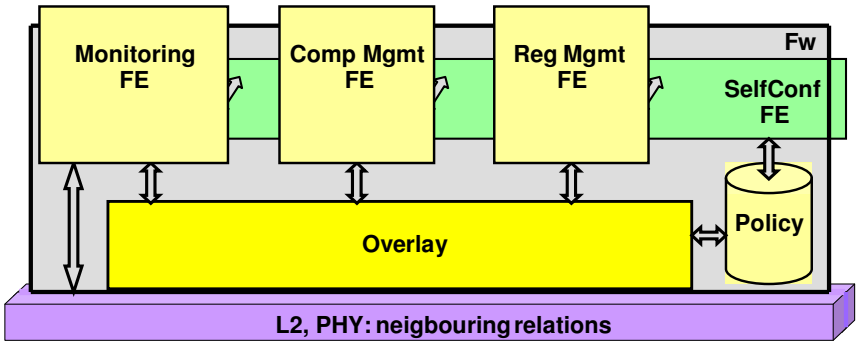

Figure 39. The Functional Elements of the Ambient Networks Management Framework

The schematic structure of the ANMFw is presented in Figure above, where the components have the following tasks:

Policy framework - enables the autonomous cooperation of the ANs. The composition strategies of the networks are handled and driven by distributed policy rules contained in each of the components of the network. We discuss the policy framework in section 10.2.3 and in Appendix D.

(Management) **Overlay** – abstracts any below L3 details of the network [142][143].

Composition Management FE – orchestration of network composition, ACS and FE management [C6][142].

Registries Management FE – multiple use: service discovery, overlay organization, lookup (routing) [142][143].

Monitoring FE – it is required to generate the triggers required to cope with the dynamic nature of the ANs. The tasks of the Monitoring FE should be executed as described by others in [111]**.**

10.2.3 Use Cases and self organizing mechanism

We have used Use Cases to specify the typical application scenarios. It has been drafted with a strong support from world leading telecommunication equipment vendors and telecom operators, thus these use cases reflect real business needs (see Appendix E) [85]. These stakeholders are both pulling (telecom operators place orders to vendors) as push (operators offer novel services to their customers, vendors offer new equipment and software to the telecom operators) the market, thus these use cases are self-realizing predictions. These use cases are not exhaustive – other use cases may be viable, as well. What we say is that these use cases are based on real needs.

Based on these use cases we proposed a generic policy system (see Appendix D) [C4][C8]. The use case can be implemented with a restrictive set of policies, where combination of more policies always result in the similar or less restrictive policy set. Also the property set based utility has the same properties. The utility function *U()* defined in 10.1.2 conforms to these use cases, too. A prototype based evaluation of this policy framework is described in the next section.

10.2.4 Performance evaluation of the prototype testbed

Based on the DMF for SONs architecture proposed in this Dissertation we implemented a peer-to-peer platform [C6][C9]. The platform removes the burden of handling hierarchical network structures from management functions. The platform consists of two main parts, the core and several modules. The core implements the most important and basic tasks (e.g. message delivery, overlay maintenance). Using the services of the core of the platform modules (applications) can be defined. These modules offer higher level services.

Modules implement several management functionalities. In order to add flexibility to our platform, some basic functionality has been implemented as modules. The superpeer election function and the policy negotiation and maintenance functionalities all are implemented as modules. If a new algorithm is requested by the operator or owner of a peer then this need can be satisfied in a flexible manner. Additionally, each management function is implemented in separate modules. Instant and automated network composition is one of the most important features of ambient networks. Every incoming composition requests is queued while composition negotiation is realized for earlier requests. Thus an overlay may run only one composition process at a time [C6][C8].

If a peer is responsible for inter-domain management of multiple ANs (that is, it has been elected as super-peer), the management platform will run a separate instance of this module per overlays. Thus the modules are only responsible to manage their own overlay. Of course, this asks for a careful design of the message-flow among different overlays within the same peer, but we provide a variety of messaging APIs to ease this task.

The monitoring application has a Graphical User Interface (GUI) that enables the user to follow the state of the network it is joined. The GUI allows the user to navigate among the overlay hierarchy levels. A snapshot of the GUI is presented in Figure 40 below. It shows three devices A, B and C (white circles) composed in one AN. The green circles represent the network interfaces of the nodes. Nodes A and B form the absorbed train network (blue rectangle), while the hotspot C is a self-contained AN (white rectangle). These two ANs are composed through gatewaying into one common AN. The common AN (pink rectangle) is represented as an overlay of the composed ANs. No matter from which node of the common AN monitors the user the topology, she/he will be able to "navigate" through the whole network. In this application we did not implemented any security constrains. In a real life application however it would be necessary to restrict the access on certain parts of the global AN. E.g., a non-authorized user would not be able to "zoom-in" into the hotspot network and monitor the other nodes attached to it.

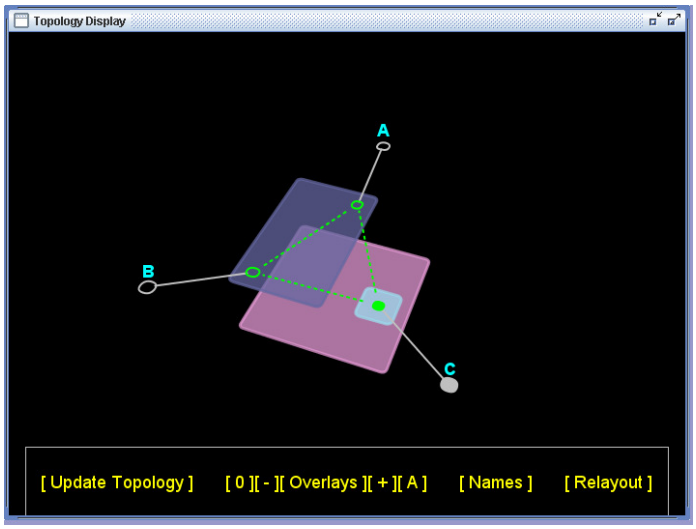

Figure 40. The GUI of the Ambient Networks monitoring application

We measured the delay from the detection of a new AN to the completion of the composition process [C6][143]. During this period three mechanisms are invoked, as follows. First, the enqueing of the composition requests are performed. Then the decision on what type of composition to follow is taken. Finally, the new super-peers are elected and (in case of gatewaying) a new overlay is instantiated. We used two laptops with IEEE 802.11b network cards as two peers. At the start of the experiment, each laptop (peer) formed its own network, thus they acted as super peers, as well. Then they were allowed to detect each other, and then they initiated the composition process. The policy rules were set in such a way that the resulted composition was an absorption or gatewaying, depending on the experiment [C8][C9].

Table 2. Performance of the composition process

Composition type	Average delay [sec]	Stdandard deviation [%]
Absorption	3.686	20.77
Gatewaying	4.140	18.67

We repeated each experiment ten times. We present the resulted average delays and the standard deviations (as percent of the average delay) in Table 2. We analyzed the process in more detail and we detected that the enqueueing of composition intentions took 2.943 for gatewying and 2.803 for absorption in average. These are quite large values (approximately 75% of overall delays), and in the later parts of the project we will try to reduce this overhead through a more efficient queueing and processing of these requests. Note that the 'pure' composition-related delay is less than 1 second for the absorption and 1.2 seconds for the gatewatying case. The difference between absorption and gatewaying is caused by the longer decision process and the initialization of the services in the new overlay. This latter takes 0.193sec in average.

10.3 Numerical Evaluation

10.3.1 The bottom up approach and clustering

In this section we compare our hierarchical structure against a flat clustering algorithm.

Clustering based approximation

We approximated our bottom-up algorithm using a hierarchical clustering algorithm, a method widely used in data analysis and data mining [133]. During this approximation we group neighbors with the same property sets together and provide a hierarchy, as explained below. It is just an approximation, because it requires a full knowledge (in terms of connectivity and available properties in our case) about the network. Initially, each node has its own cluster and all of these clusters are member of a working cluster set. At each round, we select from this set the subset of neighboring clusters with the largest number of common properties. If there are multiple subsets with the same largest number of common properties, then we choose the largest one from these subsets. Clusters of the selected subset are removed from the working cluster set and are assigned a new parent cluster. This parent cluster is inserted into the working cluster set, and the cycle restarts until either the working cluster set consists only of one single cluster or there are no more clusters with common properties. Thus the output of the algorithm is a cluster hierarchy with one or more root nodes [143].

Optimality of overlay hierarchy for property set based models

We will compare the estimated maintenance cost of the resulted hierarchy of against the estimated maintenance cost of an equivalent flat cluster. The flat clusters are generated by an algorithm which maintains separate clusters for each property alone. We expect that our hierarchical approach, approximated by the property set based algorithm presented in the previous section above achieves a lower maintenance cost because it aggregates multiple properties into the same group.

It is important to note that our property set based algorithm doesn't offer full functionality of per property flat clustering. For some properties, one single cluster created by the classical (benchmark) clustering may appear as multiple clusters in the hierarchy created by our property set based algorithm. In order to make a fair comparison, it is necessary to introduce a complementary clustering for each property grouping into one cluster all the overlays in the hierarchy whose nodes would belong to the same per property cluster (see Figure 41).

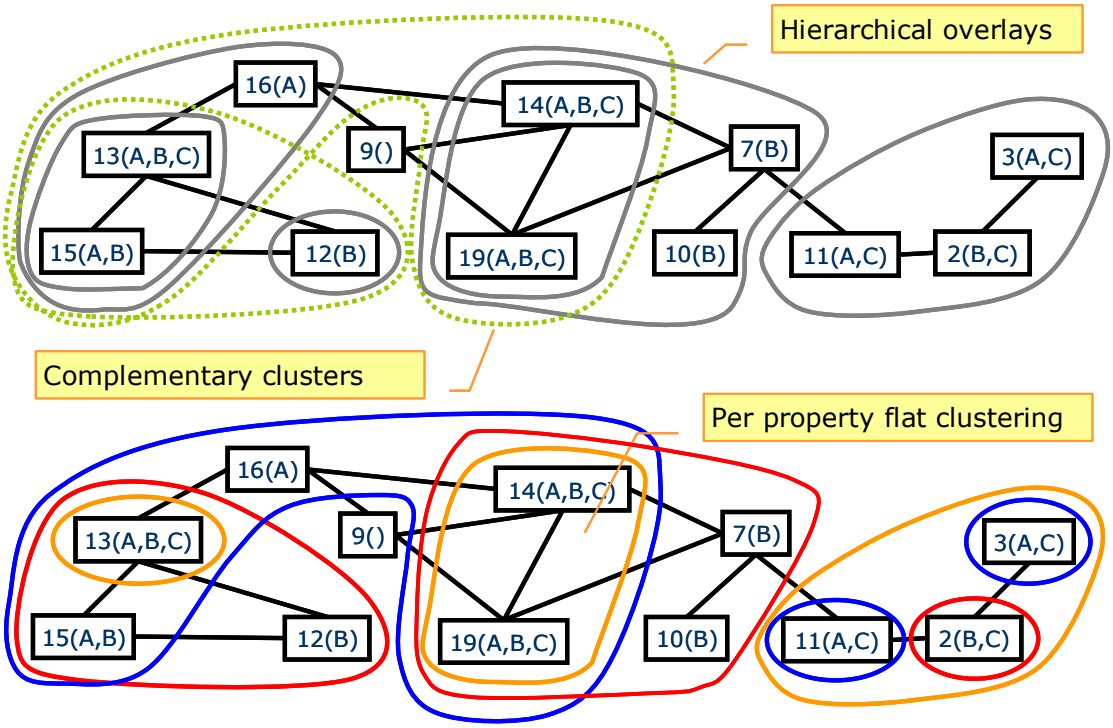

Figure 41. Evaluate optimality of overlay hierarchy (Example)

We evaluate the achieved gain of maintenance cost as the ratio of estimated maintenance cost of flat clustering versus the joint maintenance cost of property set based (i.e., hierarchical) clustering and of the complementary clustering (as just explained). As a rough first approximation, we assumed that group maintenance cost is linearly proportional to the number of group members both for per property clusters, overlays in the hierarchy and also for complementary clusters. Thus numerically:

$COST_{flatClustering} = \sum_{property}$ clusterSize

$COST_{hierarchicalOverlays} = \sum_{hierarchy}$ clusterSize + $\sum_{complementary}$ clusterSize

$COST\_Savings = (COST_{flatClustering} - COST_{hierarchicalOverlays}) / COST_{hierarchicalOverlays}$

We illustrated the clusters in Figure 41, where numbers represent node IDs while letters correspond to properties of these nodes. Now if we calculate the maintenance costs, we will have:

$COST_{flatClustering} = (5+1+1)+(3+4+1)+(1+2+3) = 21$

$COST_{hierarchicalOverlays} = (2+2+1+2+3+3)+(2+2) = 17$

$COST\_Savings = (21 - 17) / 17 = \underline{23.5\%}$

This means that if we apply our property set based hierarchical clustering algorithm to build a SON graph instead a flat clustering of the nodes we achieve 23.5% cost saving.

Simulation results

We conducted simulation experiments to evaluate the cost savings introduced above. We have generated 1000 peers and connected them in a pseudo-random manner. The property sets were randomly assigned to the peers, as well. We used a system-wide parameter *p* which was the probability that of two different peers would have the same property sets.

Simulation results (see Figure 42) have shown that optimality of hierarchy created by the property set based self-organization algorithm depends largely on the following factors

- physical network topology (i.e., average connectivity degree)
- distribution of properties

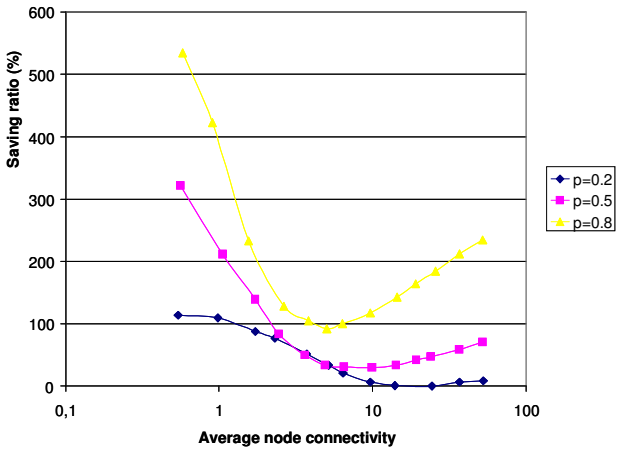

Figure 42. Evaluation of property set based self-organization algorithm

Simulation results in Figure 42 show that the property set based hierarchical algorithm performs significantly better than the flat clustering based reference algorithm if

- average node connectivity is high
- and/or average presence probability of properties are high

The rationale behind these results is that all of the above factors increase the number of common properties per group and this implies smaller maintenance costs than using per property flat clustering. Figure 42 shows high savings for very low average connectivity values too. However, these are not real savings, since in these cases most clusters consist of one single node.

Based on the investigations presented in this section we can say that the distributed hierarchical structures result in better structures than flat structures. The simulations also showed that the particular self-organization rules have a significant impact on the efficiency of the overall system. Therefore in the following section we investigate in more detail the behavior of the hierarchical structure (i.e., the SON graph) when the compositions are done according to the bottom-up algorithm.

10.3.2 Signaling overhead of the Bottom up approach

We have implemented a simulation model and with extensive simulations we have shown that the proposed algorithm scales well with the number of nodes in the network. The simulator and the simulation parameters are described in Appendix F.

We used the values measured in the prototype for each composition type. We randomly connected the nodes. The properties (self organizing attributes) for each node was randomly selected, in total we used 100 properties. We differentiated between scenarios not on the total number of properties, but on the resulted graph. More exactly, we differentiated among simulation scenarios based on the proportion of gateways from the total composition operations. The higher the probability of common properties was set, the less gatewaying type of composition occurred [C4][C6].

We present the results for two very different scenarios. In the first one lots of gatewaying compositions occur during a simulation run, 30% in average. Note that this means that on the average very few nodes form a common SON at the lower layers, because practically the member count of a SON increases only after absorptions.

We also present the results for a situation, where very few gatewaying compositions happen. If we the ratio of gatewaying compositions among all compositions is around 1% (practically below 2%), then in the larger networks basically one large overlay is created and only a few smaller SONs are directly connected to it which means that the hierarchical structure resembles a flat two-layer hierarchy. Since our goal was to investigate the formation of hierarchical structures, we selected a higher threshold for the ratio of gatewaying compositions. Therefore we chose the scenario where 5% of the compositions were of gatewaying type.

In our simulations the 30% gatewaying scenarios on average resulted in twice as deep hierarchies than the 5% gatewaying ones. There was not such a significant difference in the average number of SONs at a given hierarchy level, but the most populated hierarchy level in the 5% gatewaying scenario had several times more SONs than the most populated hierarchy in the 30% gatewaying scenario. This is an expected behavior, because more gatewaying results in a vertical growth of the tree.

From the overall hierarchy point of view we obtain two distinct scenarios just by setting the ratio of gatewaying compositions among all compositions, as expected. This is enough for us to asses the behavior of the system. The detailed reaction of SONs as the policy rules or policy sets are changed should be the topic of a different research.

The simulation results presented in this section are the average computed from ten simulations run with different seeds. The standard deviation in each case was significant, but did not exceed the10% of the average values.

Figure 43 presents the total configuration time of the SON graphs as a function of number of nodes in the system for the 30% gatewaying scenario.

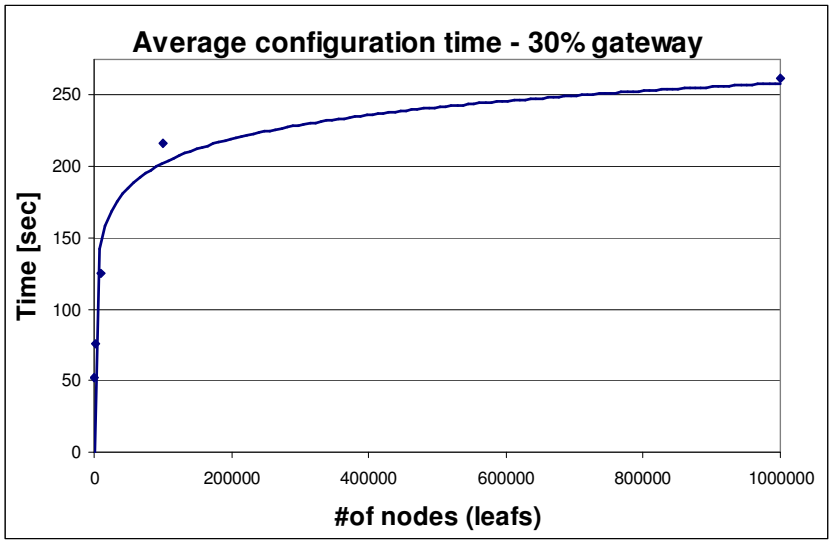

Figure 43. Scalability of the bottom-most composition algorithm (30% gatewaying scenario)

We present the simulation values for the 5% gatewaying scenario below in Figure 44. In both cases the logarithmic trend line fits well with the simulated configuration times.

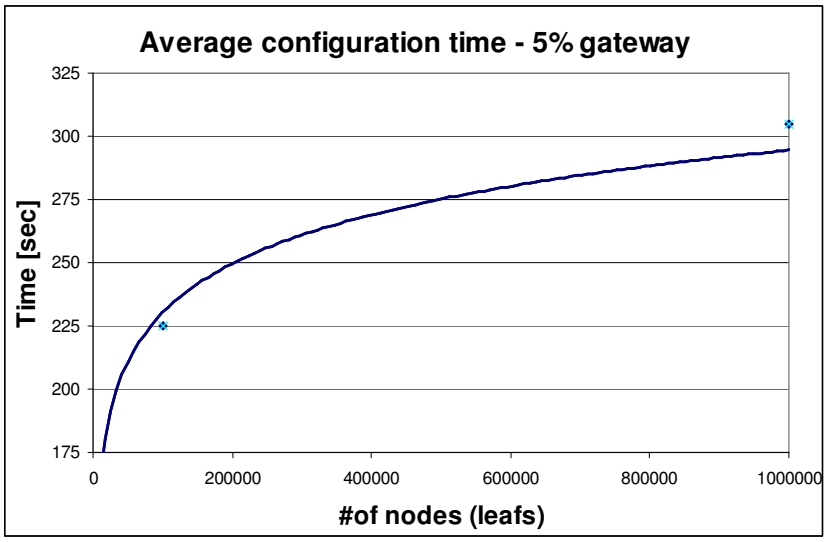

Figure 44. Scalability of the bottom-most composition algorithm (5% gatewaying scenario)

As we can see, the more gateways there are, the faster the composition processes are terminated (avg. config time). On the other hand, as the number of absorptions increase, the number of average messages handled by a node is increased (see Figure 45). Also, as more absorptions happen, the message load of the most loaded SON increases (see Figure 46): more absorptions mean larger overlays, meaning that the compositions started by more peers are to be handled by its superpeer.

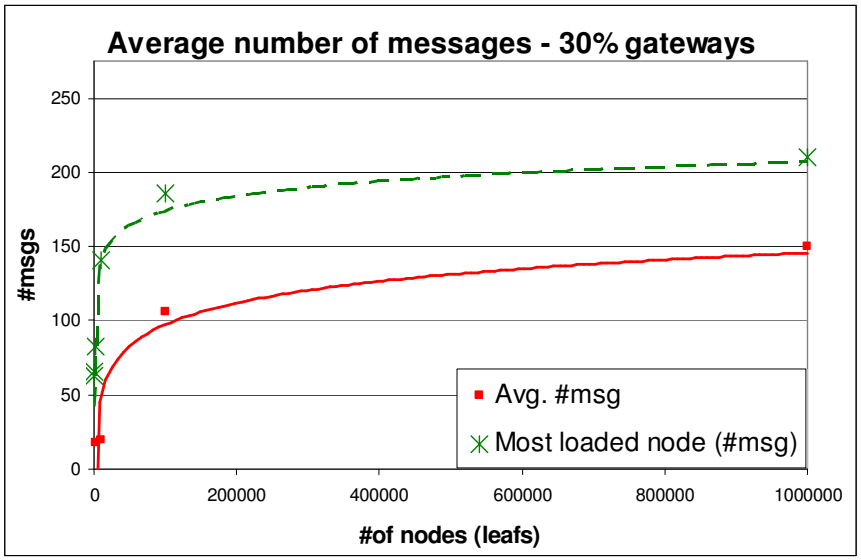

Figure 45. Message load of the bottom-most composition algorithm (5% gatewaying scenario)

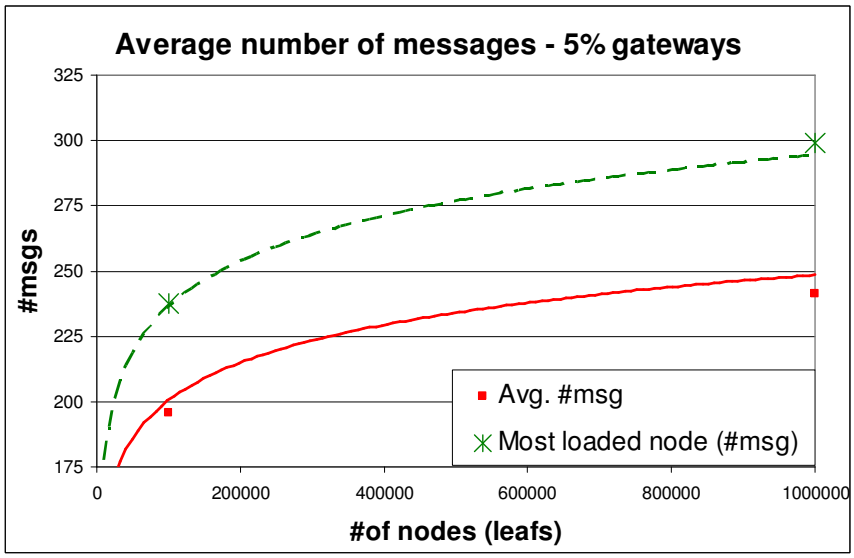

Figure 46. Message load of the bottom-most composition algorithm (5% gatewaying scenario)

Still we can state that the proposed bottom up algorithm is scalable, as the cost of the compositions is increasing slower than the number of member nodes, even in the case of very different composition profiles in the network. We fitted the logarithmic trend on the results for both scenarios to illustrate this (see Figure 45 and Figure 46).

10.3.3 Simulation under churn

The algorithm ensures the creation of the hierarchical management structure in SONs. Nevertheless the robustness of our proposals can be tested only in churn conditions. During churn a peer leaves or joins the system. Note that in our case a join (or arrival) is a much worse situation than a leave, as the former *always* results in composition, while

the latter *only if* the SON that was left behind is partitioned. In a stationary system, on the average, the number of leaving and joining peers is equal. This means that as time evolves the arrivals gradually replace the original members [C5][J6][C8].

According to [86] it is possible to express the cost of the maintenance of a network under churn without using the churn rate. The authors proposed to use the percentage of the original nodes of the networks replaced during churn. Following this logic, we tested the behavior of our system by evaluating the time required to replace 10% and 25% of the original members of the system. Table 3 and Table 4 present the times required to react to churns.

As presented in these tables, the dynamism does not add a very high cost to the original configuration time.

The reason is that since the SON is distributed most of the extra compositions happen in parallel. Only compositions on the heaviest loaded SON will have to wait until all the composition requests are answered. This is the main advantage of our proposal in terms of performance.

Table 3. System performance (30% gatewaying scenario)

#nodes	orig [s]	10% [s]	25% [s]	10% [% increase]	25% [% increase]
100	52.13	53.83	55.75	3.26	6.94
1000	76.3	81.22	86.83	6.45	13.8
10000	125.16	131.36	142.12	4.95	13.55

Table 4. System performance (5% gatewaying scenario)

#nodes	orig [s]	10% [s]	25% [s]	10% [% increase]	25% [% increase]
100	45.43	48.76	49.79	7.33	9.6
1000	102.07	103.55	108.36	1.45	6.16
10000	152.88	155.88	164.24	1.96	7.43

We have also expressed the differences between the times required to react to churn and the original configuration times normalized to the original SON configuration times. As we can see, the scenario with more gateways requires more *extra* time. The reason for this is, that in the graphs where there are more gateways, there are also more possibilities to send the composition request one level up in the hierarchy tree. This adds extra time to the process.

We have also tested the time required to reach the so called *half life* of the network [86], when 50% of the nodes are changed compared to the original configuration. The results are presented for the 5% gatewaying scenario in Table 5. This time we only present the relative overhead compared to the original configuration times.

Table 5. Overhead of reaching the half life of the network (5% gatewaying scenario)

#nodes	50% [% increase]
100	16.53
1000	26.9
10000	35.92

As we can see, the overhead is less than half of the total configuration time, which is due to fact that local interactions happen locally and are not queued at a single SON.

During our simulations we varied several parameters (e.g., the ratio of gatewaying, the churn rate) and we evaluated the time to create the SON graph. In practice this can be done in the opposite direction. Operators might set forth a threshold configuration time and guarantee (or allow) only that level of dynamism that can be supported by the network within this threshold. A different (softer) approach would be to monitor their users and if the trend is moving towards the violation of the threshold, they might take action to change the self-organizing behavior of their users (e.g., motivate/enforce to change the policies).

Chapter 11 Summary

We proposed a Distributed Management Framework (DMF) for self-organizing networks (SON) to determine how two or more management domains (authorities) should interact with each other to establish cooperation. We proposed two types of network-interaction methods, called absorption and gatewaying. We proposed a hierarchical peer-to-peer (p2p) overlay network structure that enables the realization of the management plane functions. We mapped the p2p overlays to the structure of the distributed management framework. We gave a formal model to describe the DMF structure, proposing the SON graphs. The SON graphs together with the connectivity graph are able to capture the dynamics of the network, as we gave the description of the absorption and gatewaying processes in our formal model. We gave a utility function and proposed and described an algorithm and that governs the self-organizing process of SONs.

We proposed the Ambient Networks as a practical realization of SONs, extending the Ambient Network Control Space with self organizing functionalities. We proposed the Ambient Network Management Framework, which organizes the management of the Ambient Networks. We proposed new Functional Entities, which sustain the self-organizing operation and autonomous management of the Ambient Networks. Ambient Networks was a joint research effort of world leading vendors and telecom operators. Our management solutions were integrated in the general Ambient Networks architecture [B1]. The Control Plane within the ACS proposed by research partners was presented as a novel concept to the public body governing the standardization of worldwide public mobile networks [89]. We have aligned our Ambient Network Management Framework with this control plane, proposing the integration of the control and management planes within the same ACS [C7]. We built a prototype testbed that implemented the Ambient Networks and we integrated it with the prototypes and functional elements developed by our research partners [C9].

We fed the performance parameters observed by means of measurement from the prototype into a simulator and we analyzed the performance of our algorithm in dynamic network environment, showing that the algorithm is scalable.

Part III Summary

Chapter 12 Conclusions

Scalable mechanisms offering new value added functions in the heterogeneous and dynamic services future internets will have growing importance. Cooperation of self-adapting elements, mechanisms and protocols are required to support these trends. We have identified two particular areas, domain-wide relative service differentiation for better quality of servce and dynamic, inter-domain network cooperation, as important elements of the path towards Future Internet.

We have proposed a network-wide proportional service to assure a predictable differentiation among flow classes. Our approach discards extra traffic at the ingress routers thus avoids congestion collapse. This results in lower queues in the core network and lower packet delays. We also assured a core stateless architecture, which results in simpler router mechanism and more scalable network. The novelty of the proposal is the interpretation of differentiated services applied to flow aggregates at network level combined with the idea of discarding the extra traffic at the ingress routers. We proposed an analytical description of the QoS parameters of the flows in this network. We proposed two algorithms to assure the network-wide proportional services in the network and have shown by means of extensive simulations that the resulted behavior of the flows conforms to proposed service. Our research can be continued by adapting the proposal to new emerging services and network architectures.

We also proposed an inter-domain network composition framework, which is able to provide distributed management support to self-organizing networks. The importance of the solution is that it offers a scalable and robust solution to manage the interactions of heterogeneous and dynamic networks. The novelty of the proposal is that it proposes a hierarchical structure based on peer-to-peer (p2p) overlays to achieve its goals. We proposed a heuristic algorithm to self-organize interacting networks into our hierarchical overlay structure based on their cooperation types. We built a prototype testbed to validate our solutions and we showed with the use of extensive simulations that the proposed solution scales well with the number of nodes of the network. We also verified that node churns are handled efficiently. A potential future research direction is the proposal of alternative distributed composition algorithms and a combined optimization of the proposed hierarchical structure to both network and service management tasks.

Our results confer scalability to the networks and adapt to changes in traffic and topology. Therefore they pave the way towards future networks, enabling the introduction of new services and business models. In what follows we discuss the applicability of our results.

12.1 Applicability of the results

Nowadays several network operators simply overprovision their capacities to cope with the growing traffic demands. On the other hand, peer-to-peer file exchange and streaming media applications already generate the majority of internet traffic. The growing number of legal enterprises offering online media content will challenge these business models. Further on, the spread of the Internet of Things model will increase the number of networked end-hosts, but with a distinct QoS profile. Also, the high speed cellular access networks will generate significant amount of traffic from/to mobile users. All these factors lead to a diversified demand of services with a fast growing traffic volume. Therefore both vendors and operators are seeking for more cost-effective and QoS agnostic solutions to cope with these changes. Our proposals enable operators to achieve these goals, our results in network service differentiation and self-organization can be deployed in their networks. Feasibility of the deployment depends on the business strategies of the operators, and is much likely to favor the application of our results in the case of green field investments or/and especially during the migration to future networks (e.g., 3GPP cores, IPv6 islands).

The advantages of the proposed mechanisms would allow the operators to deploy cheaper devices, save operational (management) costs and marketing costs (promises on QoS levels are more easily communicated). The implementation of the proposals does not require development of new mechanisms in routers. The network-wide proportional services can be implemented by defining a new signaling and control logic over existing router mechanisms, as detailed in the dissertation. We have demonstrated the feasibility of the Distributed Management Framework through a prototype implementation, which can serve as the basis of its implementation in a vendor specific enterprise system. This would not imply the development of new mechanisms but it should be adapted to the particular data management implementations.

The results in network-wide proportional service (Part I) enable the network operator to deploy QoS traffic control only at the ingress, which confers scalability to their network. This is a challenging issue in new global networks. Additionally, using the proposals, the operators achieve network wide QoS, not only per hop. Further advantage of the proposal is that the queues within the network domains, at the core routers are much less occupied. This can be used either to deploy cheaper routers (with lower memory need) or to gain competitive edge on the service market (offering better services). The new “All IP” core networks are a typical case of operator controlled domain with ingress filtering, and the proposal fits naturally in this architecture, as the control is deployed at the ingress, only.

The result of our distributed management framework (DMF) (Part II) served as a basis of an EU funded research project, where our partners were major vendors and operators of the global telecom market. As such the output of this work was aligned to the standardization work on the next public wireless networks. Particularly, the proposed DMF is the management plane extension of current control-plane 3GPP composition. In Future Internet the proposed network self-organizing methods can serve as the basis for the dynamic attachment of the smaller networks (e.g., personal area networks – PAN) or newly installed islands in the migration process from IPv4 to IPv6 Internet.

Chapter 13 Bibliography

13.1 Published Results

[B] BOOK CHAPTERS

[B1] A. Galis (ed.), R. Szabó (ed.), M. Brunner (ed.), H. Abrahamsson, J. Andres, E.A. Asmare, M.A. Callejo, L. Cheng, M. Erdei, A. Gonzalez, A. Gunnar, P. Kersch, Z.L. Kis, B Kovács, G. Molnár, J. Nielsen, G. Nunzi, R. Ocampo, Cs. Simon, S. Schuetz, R. Stadler, A. Wagner, K. Zimmerman, "Chapter 12 - Towards Ambient Networks Management". In: N Niebert, A Schieder, J Zander, R Hancock (eds.), Ambient Networks – Co-operative Mobile Networking for the Wireless World. ISBN 978-0-470-51092-6, pp. 231-260. Chichester: John Wiley & Sons, 2007.

[J] JOURNALS

[J1] Cs. Simon, B. Tordai, M. Nafornita, "End-to-end Proportional Services for TCP Flows", Scientific Bulletin of the Politehnica University of Timisoara – Transactions on Electronics and Communications, Vol. 49(63):(2), pp. 377-382, 2004.

[J2] Cs. Simon, M. Nafornita, "Network-wide Proportional Services", Scientific Bulletin of the Politehnica University of Timisoara – Transactions on Electronics and Communications, Vol. 53(67):(2), pp. 161-166, 2008.

[J3] M. Brunner, A. Galis, L. Cheng, J.A. Colás, B. Ahlgren, A. Gunnar, H. Abrahamsson, R. Szabo, Cs. Simon, J. Nielsen, S. Schuetz, A.G. Prieto, R. Stadler, G. Molnar, "Ambient Networks Management Challenges and Approaches", Lecture Notes in Computer Science, Vol. 3284, pp. 196-216, 2004.

[J4] B. Kovács, Cs. Simon, "»Ambient« hálózatok - áttekintés", Híradástechnika Vol. LX:(7), pp. 39-44, 2005.

[J5] M. Brunner, A. Galis, L. Cheng, J. A. Colás, B. Ahlgren, A. Gunnar, H. Abrahamsson, R. Szabo, Cs. Simon, J. Nielsen, S. Schuetz, A. G. Prieto, R. Stadler, G. Molnar: "Towards Ambient Networks Management", Lecture Notes in Computer Science, Vol. 3744, pp. 215-229, 2005.

[J6] L. Cheng, R. Ocampo, K. Jean, A. Galis, Cs. Simon, R. Szabo, P. Kersch, R. Giaffreda, "Towards Distributed Hash Tables (De)Composition in Ambient Networks", Lecture Notes in Computer Science, Vol. 4269, pp. 258-268, 2006.

[J7] Cs. Simon, P. Kersch, R. Szabó, "Ambiens hálózatok", Híradástechnika, Vol. LXIII:(3), pp. 46-47, 2008.

[C] CONFERENCES

[C1] Cs. Simon, A. Török, I. Moldován, O. Pop, J. Bíró, K. Ishibashi, A. Koike, "Proportional Services for IP Networks", in Proc. of High Speed Networking

Lab International Workshop 2001, pp. 52-59, Balatonfüred, Hungary, May 2001.

[C2] Cs. Simon, A. Vidács, I. Moldován, A. Török, "End-to-End Relative Differentiated Services for IP Networks", in Proc. of High Speed Networking Lab International Workshop 2002 , pp. 72-75, Budapest, Hungary, May 2002.

[C3] Cs Simon, A. Vidács, I. Moldován, A. Török, K. Ishibashi, H. Ishii, "End-to-End Relative Differentiated Services for IP Networks", in Proc. of 7th IEEE Symposium on Computers and Communication, pp. 783-788, Taormina, Italy, July 2002.

[C4] R. Szabo, P. Kersch, B. Kovacs, Cs. Simon, M. Erdei, A. Wagner, "Dynamic Network Composition for Ambient Networks: a Management View", In Proc. of Eurescom Summit 2005, pp. 35-42, Heidelberg, Germany, April 2005.

[C5] L. Cheng, R. Ocampo, A. Galis, R. Szabo, Cs. Simon, P. Kersch, "Self-management in Ambient Networks for Service Composition", in Proc. of IFIP International Conference on Intelligence in Communication Systems, pp. 67-76, Montreal, Canada, October 2005.

[C6] Cs. Simon, R. Szabó, P. Kersch, B. Kovács, A. Galis, L Cheng, "Peer-to-peer Management in Ambient Networks", In Proc. of the 14th IST Mobile and Wireless Communications Summit, pp. 537:1-5, Dresden, Germany, June 2005.

[C7] Cs. Simon, P. Poyhonen, D. Zhou, "Controlling and Managing Compositions in Ambient Networks", in Proc. of 15th IST Mobile and Wireless Communications Summit, pp. 331-337, Myconos, Greece, June 2006.

[C8] A. Bria, J. Markendahl, R. Rembarz, P. Pöyhönen, Cs. Simon, M. Miozzo, N. Akhtar, R. Jennen, "Validation of the Ambient Networks System Architecture", in Proc. of International Conference on Communications and Networking in China 2007, pp. 784-791, Shanghai, China, August 2007.

[C9] Cs. Simon, R. Rembarz, P. Pääkkönen, H. Perkuhn, C. Bento, N. Akhtar, R. Agüero, T. Katona, P. Kersch, "Ambient Networks Integrated Prototype Design and Implementation ", in Proc. of 16th IST Mobile IST Mobile and Wireless Communications Summit, pp. 522:1-5, Budapest, Hungary, July, 2007.

[R] CITATIONS

[R1] The detailed list of publication of the author (with more than 70 records) and the list of articles that cited his works (more than 30 independent citations) are available at:
http://mycite.omikk.bme.hu/search/slist.php?lang=0&AuthorID=10000591

13.2 References

[1] R. Braden, D. Clark, S. Shenker, “Integrated Services in the Internet Architecture: An Overview”, IETF RFC 1633, June 1994.

[2] S. Blake et al., “An Architecture for Differentiated Service”, IETF RFC 2475, December 1998.

[3] C. Dovrolis, P. Ramanathan, “A Case for Relative Differentiated Services and the Proportional Differentiation Model”, IEEE Network, Vol. 13:(5), pp. 29:(4), September-October 1999.

[4] C. Dovrolis, P. Stiliadis, P. Ramanathan, “Proportional Differentiated Services: Delay Differentiation and Packet Scheduling”, ACM SIGCOMM Computer Communication Review, Vol.29:(4), pp. 109-120, 1999.

[5] C. Dovrolis, D. Stiliadis, P. Ramanathan, “Proportional Differentiated Services, Part II: Loss Rate Differentiation and Packet Dropping”, in Proc. of IEEE/IFIP International Workshop QOS’00, 2000.

[6] J. Case, M. Fedor, M. Schoffstall, J. Davin, "Simple Network Management Protocol", IETF RFC 1157, May 1990.

[7] J. Rubio-Loyola, J. Serrat, A. Astorga, A. Fischer, A. Berl, H. de Meer, G. Koumoutsos, “A viewpoint of the network management paradigm for Future Internet networks”, IFIP/IEEE International Symposium on Integrated Network Management-Workshops IM '09, pp. 93–100, 1-5 June 2009.

[8] J. Case, R. Mundy, D. Partain, B. Stewart, “Introduction and Applicability Statements for Internet Standard Management Framework”, IETF RFC 3410, December 2002.

[9] Th. Zahariadis, D. Papadimitriou, H. Tschofenig, S. Haller, P. Daras, G. D. Stamoulis, M. Hauswirth, “Towards a Future Internet Architecture”, Lecture Notes in Computer Science, Vol. 6656, pp. 7-18, 2011.

[10] R. Chaparadza, “The Self-Managing Future Internet powered by the current IPv6 and Extensions to IPv6 towards IPv6++”, in Proc. of the 2nd International workshop on Management of Emerging Networks and Services MENS 2010, Miami, FL, pp. 1-5, December 2010.

[11] Y. Chen, C. Qiao, M. Hamdi, D. Tsang, “Proportional differentiation: a scalable QoS approach”, IEEE Communications Magazine, Vol. 41:(6), pp.52–58, 2003.

[12] P.J. Argibay-Losada, A. Suárez-González, C. López-García, M. Fernández-Veiga, "A new design for end-to-end proportional loss differentiation in IP networks", Computer Networks: The International Journal of Computer and Telecommunications Networking archive, Vol. 54:(9), pp. 1389-1403, June 2010.

[13] J. Crowcroft, P. Oechslin, "Differentiated End-to-End Internet Services using a Weighted Proportional Fair Sharing TCP", SIGCOMM Comput Commun Rev., Vol.28:(3), pp. 53-69, 1998.

[14] H. Su, M. Atiquzzaman, “End-to-End QoS for Differentiated Services and ATM Internetworking”, in Proc. of 9th International Conference on Computer Communications and Networks, Las Vegas, NV, pp.557-562, October 2000.

[15] Y.E. Sung, C. Lund, M. Lyn, S.G. Rao, S. Sen, "Modeling and understanding end-to-end class of service policies in operational networks", in Proc. of the ACM SIGCOMM 2009 Conference on Data communication, Barcelona, Spain, pp.219-230, 2009.

[16] D.H. Lorenz, A. Orda, D. Raz, Y. Shavitt, “Efficient QoS partition and routing of unicast and multicast”, IEEE/ACM Transactions on Networking, Vol.14:(6), pp.1336–1347, 2006.

[17] A. Orda, A. Sprintson, “A scalable approach to the partition of QoS requirements in unicast and multicast”, IEEE/ACM Transactions on Networking 13 (5), pp.1146–1159, 2005.

[18] Per Hop Behaviors Based on Dynamic Packet States, <draft-stoica-diffserv-dps-00.txt>, IETF draft, work in progress, 2000..

[19] J. MacKie-Mason, H. Varian, “Pricing the interenet”, Public Access to the Internet, B. Kahin and J. Keller (ed.), 1994.

[20] M. Odlyzko, “Paris Metro Pricing: The Minimalist Differentiated Services”, in Proc. of 7th International Workshop on Quality of Service (IWQoS), London, UK, pp. 159-164, May 1999.

[21] D. Ros, B. Tuffin, “A mathematical model of the Paris Metro Pricing scheme for charging packet networks”, Computer Networks Vol. 46:(1), pp. 73–85, 2004.

[22] C-K. Chau, Q. Wang, D-M. Chiu, “On the Viability of Paris Metro Pricing for Communication and Service Networks”, in Proc. of IEEE INFOCOM 2010, San Diego, CA, pp. 1–9, March 2010.

[23] H. Zhou, P. Fan, X-G. Xia, K. B. Letaief, “Achieving Network Wide Proportional Fairness: A Pricing Method”, in Proc. of IEEE Wireless Communications and Networking Conference (WCNC), pp. 1-6, April 2010.

[24] Ch. Li, Sh. Tsao, M. Ch. Chen, Y. Sun, Yueh-Min Huang, “Proportional Delay Differentiation Service based Weighted Fair Queueing”, in Proc. of 9th International Conference on Computer Communications and Networks (ICCN), Las Vegas, USA, pp. 418-423, October 2000.

[25] H. Saito, Cs. Lukovszki, I. Moldovan, “Local Optimal Proportional Differentiation Scheduler for Relative Differentiated Services”, in Proc. of 9th International Conference on Computer Communications and Networks (ICCN), Las Vegas, USA, pp. 418-423, October 2000.

[26] N. Christin, J. Liebeherr, T. Abdelzaher, "Enhancing class-based service architectures with adaptive rate allocation and dropping mechanisms", IEEE/ACM Transactions on Networking, Vol.15:(3), pp. 669–682, 2007.

[27] J. Liebeherr, N. Christin, "JoBS: Joint buffer management and scheduling for differentiated services", in Proc. of 9th International Workshop on Quality of Service (IWQoS), Karlsruhe, Germany, pp. 404–418, 2001.

[28] A. Striegel, G. Manimaran, "Packet scheduling with delay and loss differentiation", Computer Communications, Vol. 25:(1), pp. 21–31, 2002.

[29] P. Hurley, J.-Y.L. Boudec, P. Thiran, M. Kara, "ABE: Providing low delay service with best effort", IEEE Network, Vol.15:(3), pp. 60–69, 2001.

[30] R.R.-F. Liao, A.T. Campbell, "Dynamic core provisioning for quantitative differentiated services", IEEE/ACM Transactions on Networking, Vol.12:(3), pp. 429–442, 2004.

[31] T. Soetens, S. D. Cnodder, O. Elloumi, "A relative bandwidth differentiated service for TCP micro-flows", in Proc. of IEEE/ACM Int'l Symposium on Cluster Computing and the Grid, pp. 602-609, 2001.

[32] N. Christin, J. Liebeherr, "Marking Algorithms for Service Differentiation of TCP Traffic", Computer Communications, Vol.28:(18), pp. 2058-2069, 2005.

[33] J. Aweya et al., "Weighted proportional window control of TCP traffic," International Journal of Network Management, Vol.11, pp. 213-242, 2001.

[34] Y. Xue, K. Chen, K. Nahrstedt, "Distributed end-to-end proportional delay differentiation in wireless LAN", in Proc. of IEEE International Conference on Communications (ICC), Paris, France, pp. 4367-4371, August 2004.

[35] W.K. Chai, M. Karaliopoulos, G. Pavlou, "Providing Relative Service Differentiation to TCP Flows over Split-TCP Geostationary Bandwidth on Demand Satellite Networks", Lecture Notes in Computer Science, Vol.4517, pp.17-29, 2007.

[36] I. Stoica, S. Schenker, H. Zhang, "Core-Stateless Fair Queing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks", IEEE/ACM Transactions on Networking, Vol.11:(1), pp. 33-46, 1998.

[37] I. Stoica, H. Zhang, "Providing Guaranteed Services Without Per Flow Management", ACM SIGCOMM Computer Communication Review, Vol.29:(4), pp. 81-94, 1999.

[38] G. Cheng, K. Xu, Y. Tian, N. Ansari, "Core-stateless proportional fair queuing for AF traffic", in Proc. of IEEE Global Telecommunications Conference GLOBECOM2004, pp.732-736, December 2004.

[39] L. Zhang, "Virtual Clock: A new traffic control algorithm for packet switching networks", in Proc. of IEEE/ACM SIGCOMM1990, Philadelphia, PA, pp. 19-29, September 1990.

[40] Clark, W. Fang, "Explicit Allocation of Best Effort packet Delivery Service", IEEE/ACM Transactions on Networking, Vol. 6:(4), pp. 362-373, August 1998.

[41] M. Goyal, A. Durresi, P. Misra, Ch. Liu, R. Jain, "Effect of Number of Drop Precedences in Assured Forwarding", in Proc. of IEEE Global Telecommunications Conference (GLOBECOM), Rio de Janeiro, Brazil, pp. 188-193, December 1999.

[42] L. Su, R. Zheng, J. C. Hou, "An Active Queue Management Scheme for Internet Congestion Control and Its Application to Differentiated Services", in Proc. of 9th International Conference on Computer Communications and Networks (ICCN), Las Vegas, NV, pp. 62-68, October 2000.

[43] S. Floyd, V. Jacobson, "Random Early Detection gateways for Congestion Avoidance", IEEE/ACM Transactions on Networking, Vol.1:(4), pp. 397-413, 1993.

[44] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, F. True, "Deriving traffic demands for operational IP networks: methodology and experience", IEEE/ACM Transactions on Networking, Vol.9:(3), pp. 265-279, 2001.

[45] Cisco Netflow, available at http://www.cisco.com/warp/public/732/netflow/index.html.

[46] R. Caceres, N.G. Duffield, A. Feldmann, J. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. Kalmanek, B.Krishnamurthy, D. Lavelle, P.P. Mishra, K.K. Ramakrishnan, J. Rexford, F. True, J.E. van der Merwe, "Measurement and Analysis of IP Network Usage and Behavior", IEEE Communications Magazine, Vol.38:(5), pp. 144-151, May 2000.

[47] N. Duffield, M. Grossglauser, "Trajectory sampling for direct traffic observation", IEEE/ACMTransactions on Networking, Vol.9:(3), pp. 280–292, June 2001.

[48] Y. Zhang, M. Roughan, N. Duffield, A. Greenberg, "Fast accurate computation of large-scale IP traffic matrices from link loads", SIGMETRICS Performance Evaluation Review, Vol.31:(1), pp. 206-217, 2003.

[49] A. Dhamdhere, L. Breslau, N. Duffield, C. Ee, A. Gerber, C. Lund, S. Sen, "Flowroute: inferring forwarding table updates using passive flow-level measurements", in Proc. of the 10th annual conference on Internet measurement IMC2010, Melbourne, Australia, pp. 315-321, November 2010.

[50] K. Papagiannaki, N. Taft, A. Lakhina, "A distributed approach to measure IP traffic matrices", in Proc. of 4th IEEE/ACM SIGCOMM Internet Measurement Conference, Taormina, Italy, pp. 161-174, October 2004.

[51] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, C. Diot, "Traffic matrices: balancing measurements, inference and modeling", ACM SIGMETRICS Performance Evaluation Review, Vol.33:(1), pp. 362-373, 2005.

[52] S. Uhlig, B. Quoitin, J. Lepropre, S. Balon, “Providing Public Intradomain Traffic Matrices to the Research Community”, ACM SIGCOMM Computer Communication Review, Vol.36:(1), pp.83-86, 2006.

[53] U. Hofmann, I. Miloucheva, Th. Pfeiffenberger, “INTERMON – Complex QoS/SLA analysis in large scale Internet”, in Proc. of Winter International Synposium on Information and Communication Technologies (WISICT), Cancun Mexico, pp. 1-13, January 2004.

[54] Y. Zhang, M. Roughan, C. Lund, D. Donoho, “An information-theoretic approach to traffic matrix estimation”, in Proc. of IEEE/ACM Conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), Karlsruhe, Germany, pp. 301-312, August 2003.

[55] A. Soule, A. Nucci, R. Cruz, E. Leonardi, N. Taft, “How to identify and estimate the largest traffic matrix elements in a dynamic environment”, SIGMETRICS Performance Evaluation Review, Vol.32:(1), pp. 73-84, 2004.

[56] G. Varghese and C. Estan, “The measurement manifesto”, Computer Communications Review, vol. 34:(1), pp. 9–14, 2004.

[57] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, F. True, “Deriving traffic demands for operational IP networks: methodology and experience”, IEEE/ACM Transactions on Networking, Vol.9:(3), pp.265-279, 2001.

[58] A. Gunnar, M. Johansson, T. Telkamp, “Traffic matrix estimation on a large IP backbone: a comparison on real data”, in Proc. of 4th ACM SIGCOMM Internet Measurement Conference, Taormina, Italy, pp. 149-160, October 2004.

[59] The OpenFlow Switch Specification. Available at http://OpenFlowSwitch.org.

[60] A. Tootoonchian, M. Ghobadi, Y. Ganjali, “OpenTM: Traffic Matrix Estimator for OpenFlow Networks”, Lecture Notes in Computer Science Vol.6032, pp.201-210, 2010.

[61] D. Jiang, X. Wang, L. Guo, “An optimization method of large-scale IP traffic matrix estimation”, AEU - International Journal of Electronics and Communications, Vol.64:(7), pp. 685-689, 2010.

[62] D. Jiang, X. Wang, L. Guo, H. Ni, Z. Chen, “Accurate estimation of large-scale IP traffic matrix”, AEU - International Journal of Electronics and Communications, Vol.65:(1), pp. 75-86, 2011.

[63] D. Jiang, Z. Xu, Z. Chen, Y. Han, H. Xu, “Joint time–frequency sparse estimation of large-scale network traffic”, Computer Networks, Vol.55:(15), pp.3533-3547, 2011.

[64] B. Zhou, D. He, Z. Sun, W. Hock Ng, "Network Traffic Modeling and Prediction with ARIMA/GARCH", in: Modeling and Simulation Tools for Emerging Telecommunication Networks, pp. 101-121, 2006.

[65] Q. Meng, Y. Chen, “Small-time scale network traffic prediction based on local support vector machine regression model”, Chin. Phys. B, Vol. 18, pp. 2112–2194, 2009.

[66] M. Dashevskiy, Z. Luo, "Network Traffic Demand Prediction with Confidence", in Proc. of IEEE Global Telecommunications Conference, (GLOBECOM), New Orleans, OL, pp. 1-5, December 2008.

[67] H.-L. Sun, Y.-H. Jin, Y.-D. Cui, S.-D. Cheng, “Network traffic prediction by a wavelet-based combined model”, Chin. Phys. B, Vol. 18 (11) pp. 47–60, 2009.

[68] F.H.T. Vieira, G.R. Bianchi, L.L. Lee, “A network traffic prediction approach based on multifractal modeling”, J. High Speed Netw., Vol. 17:(2) 83–96, 2010.

[69] Y. Chen, B. Yang, Q. Meng, “Small-time scale network traffic prediction based on flexible neural tree", to appear in: Applied Soft Computing, available at: doi:10.1016/j.asoc.2011.08.045, 2011.

[70] C. Albuquerque, B. J. Vickers, T. Suda, “Network Border Patrol”, in the Proc. of 19th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2000, vol.1., Tel Aviv , Israel, pp. 322 - 331, March 2000.

[71] R. Pletka, A. Kind, M. Waldvogel, S. Mannal, “Closed-loop congestion control for mixed responsive and non-responsive traffic”, IEEE Global Telecommunications Conference GLOBECOM2003, San Francisco, CA, pp. 4180-4185, December 2003.

[72] N. Benameur, S. Ben Fredj, F. Delcoigne, S. Oueslati-Boulahia, J. W. Roberts, “Integrated Admission Control for Streaming and Elastic Traffic”, Quality of Future Internet Services – Lecture Notes in Computer Science, Vol. 2156, pp. 69-81, 2001.

[73] O. Pop, T. Máhr, T. Dreilinger, R. Szabo, “Vendor-Independent Bandwidth Broker Architecture for DiffServ Networks”, in Proc. of Polish-Czech-Hungarian Workshop on Circuit Theory, Signal Processing and Telecommunications Networks (PCHW), Budapest, Hungary, September 2001.

[74] G. Stattenberger, T. Braun, "Performance of a Bandwidth Broker for DiffServ Networks", in Proc. of the 13th Fachtagung Kommunikation in verteilten Systemen (KiVS03), Leipzig, Germany, pp. 93-105, February 2003.

[75] A. Terzis, J. Ogawa, S. Tsui, L. Wang, L. Zhang, “A Prototype Implementation of the Two-Tier Architecture for Differentiated Services”, in Proc. of 5th IEEE Real-Time Technologies and Applications Symposium RTAS1999, Vancouver, Canada, pp.1-8, June1999.

[76] Committed Acces Rate, Cisco press, available online at http://www.cisco.com/warp/public/732/Tech/car/

[77] W. Stevens, “TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms”, IETF RFC 2001, January 1997.

[78] J. Postel, Transmission Control Protocol, IETF RFC 793, September 1981.

[79] W. Feng, D. Kandlur, D. Saha, K. Shin, “BLUE: A New Active Queue Management Algorithms”, Project Report CSE-TR-387-99, University of Michigan, April 1999.

[80] W. Feng, Kang G. Shin, D.D. Kandlur, D. Saha, “The Blue active queue management algorithms”, IEEE/ACM Transactions on Networking, vol.10:(4), pp. 513–528, 2002.

[81] G. Da-gang, “A New Adaptive BLUE Algorithm“, in Proc. of the 1st Electrical and Control Engineering ICECE 2010, pp. 2601-2605, June 2010.

[82] F. Wu-Chang; D.D. Kandlur, D. Saha, K.G. Shin, “Stochastic fair blue: a queue management algorithm for enforcing fairness”, in Proc. of the 20th Annual Joint Conference of IEEE ComSoc (INFOCOM), Anchorage, AK, pp. 1520-1529, April 2001.

[83] Cs. Simon et al., “Enabling autonomicity in future networks”, in Proc. of the 2nd International workshop on Management of Emerging Networks and Services MENS 2010, Miami, FL, pp. 637-641, December 2010.

[84] N. Niebert, A. Schieder, H. Abramowicz, G. Malmgren, J. Sachs, U. Horn, C. Prehofer, H. Karl, “Ambient Networks: an Architecture for Communication Networks Beyond 3G”, IEEE Wireless Communications, Vol. 11:(2), pp. 14-22, April 2004.

[85] J. Nielsen, A. Galis, H. Abrahamsson, B. Ahlgren, M. Brunner, L. Cheng, J. Colas, Cs. Simon, A. G. Prieto, A. Gunnar, G. Molnar, R. Szabo, “Management Architectures and Approaches for Ambient Networks”, in Proc. of the 12th Wireless World Research Forum (WWRF), Toronto, Canada, November 2004.

[86] D. Liben-Nowell, H. Balakrishnan, D. Karger, “Analysis of the evolution of peer-to-peer systems”, in Proc. of 21st Symposium on Principles of distributed computing (PODC), Monterey, CA, pp. 233-242, July 2005.

[87] R. Szabo, Cs. Simon, “Distributed Inter-domain Management: Domain Composition”, 19th meeting of the Network Management Research Group (NMRG) of the Internet Research Task Force (IRTF), Stockholm, Sweden, January 2006.

[88] R. Campos, C. Pinho, M. Ricardo, J. Ruela, P. Poyhonen, C. Kappler, "Dynamic and automatic interworking between personal area networks using composition", in Proc. of IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications PIMRC, Berlin, Germany, pp. 947 - 951, September 2005.

[89] 3GPP TR 22.980, "Network composition feasibility study", 3rd Generation Partnership Project, Technical Specification Group Services and System Aspects (Release 10), March 2011.

[90] H.G. Hegering, S. Abeck, B. Neumair, "Integrated Management of Networked Systems", Morgan Kaufmann, 1999.

[91] A. Galis, “Multi-Domain Communication Management”, Boca Raton, Florida, CRC Press, 2000.

[92] TeleManagement Forum, "Enhanced Telecom Operations Map—The Business Process Framework for the Information and Communications Services Industry", Release 6.0 GB 921. November 2005.

[93] A. Clemm, "Network Management Fundamentals", Indianapolis, IN, Cisco Press, 2007.

[94] S. Aidarous and T. Pevyak (eds), “Telecommunications Network Management: Technologies and Implementations”, IEEE Press, 1997.

[95] J. Schönwalder, “Network Management by Delegation”, in Proc. of 8th Joint European Networking Conference, pp. 931:1-9, Edinburgh, UK, May 1997.

[96] B. Jennings, S. van der Meer, S. Balasubramaniam, D. Botvich, M. O. Foghlu, W. Donnelly, “Towards Autonomic Management of Communications Networks”. IEEE Communications Magazine, Vol.45:(10), pp. 112-121, October 2007.

[97] S. van der Meer, W. Donnelly, J. Strassner, B. Jennings, M. Foghlu, “Emerging Principles of Autonomic Network Management”, in Proc. of 1st IEEE International Workshop on Modelling Autonomic Communications Environments, Dublin, Ireland, pp. 1-5, October 2006.

[98] Y. Cheng, R. Farha, M. S. Kom, A. Leon-Garcia, J. Won-Ki Hong, “A Generic Architecture for Autonomic Service and Network Management”, Computer Communications, Vol.29:(18), pp.3691-3709, November 2006.

[99] L.B. Ruiz, J.M. Nogueira, A. Loureiro, “MANNA: A Management Architecture for Wireless Sensor Networks”, IEEE Communications Magazine, Vol.41:(2), pp.116-125, 2003.

[100] Madeira. Project homepage. http://www.celtic-madeira.org/index.html.

[101] J. Rubio-Loyola, A. Astorga, J. Serrat, W. K. Chai, L. Mamatas, A. Galis, S. Clayman, A. Cheniour, L. Lefevre, A. Fischer, A. Paler, Y. Al-Hazmi, H. de Meer, "Platforms and Software Systems for an Autonomic Internet", in Proc. of IEEE Global Telecommunications Conference GLOBECOM 2010, Miami, FL, pp. 1-6, December 2010.

[102] F. Sestinim, “Situated and Autonomic Communication – an EC FET European Initiative”, ACM SIGCOMM Communication Review, Vol.36:(2), pp.17-20, 2006.

[103] C. Jelger, C. Tschudin, S. Schmid, G. Leduc, “Basic Abstractions for an Autonomic Network Architecture”, in Proc. of 1st IEEE Workshop on Autonomic and Opportunistic Communications (WoWMoM AOC’07), Helsinki, Finland, pp. 1-6, June 2007.

[104] Autonomic Network Architecture (ANA) project homepage: http://www.ana-project.org.

[105] 4WARD project homepage, http://www.4ward-project.eu

[106] Stanford Clean Slate Program, http://cleanslate.stanford.edu

[107] A. Gupta, "Network Management: Current Trends and Future Perspectives", Journal of Network and Systems Management, Vol.14:(4), pp. 483-491, 2006.

[108] Jiang, Y.C., Xia, Z.Y., Zhong, Y.P., Zhang, S.Y., "Defend mobile agent against malicious hosts in migration itineraries", Microprocessors and Microsystems, Vol.28:(10), pp. 531–546. 2004.

[109] S. Murphy, E. Lewis, R. Puga, R. Watson, R. Yee, "Strong Security for Active Networks", in Proc. of IEEE Open Architectures and Network Programming OPENARCH, Anchorage, AK, pp. 63-70, April 2001.

[110] M. Burgess, F.E. Sandnes, "A Promise Theory Approach to Collaborative Power Reduction in a Pervasive Computing Environment", Lecture Notes on Computer Science, Vol. 4159, pp. 615-624, 2006.

[111] M. Dam and R. Stadler, "A generic protocol for network state aggregation", in Proc. of Radiovetenskap och Kommunikation (RVK), Linkoping, Sweden, pp. 411-417, June 2005.

[112] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, F. Zambonelli, "A Survey of Autonomic Communications", ACM Transactions on Autonomous and Adaptive Systems, vol.1:(2), pp.223-259, 2006.

[113] N. Agoulmine, S. Balasubramaniam, D. Botvich, J. Strassner, E. Lehtihet, W. Donnelly, "Challenges for Autonomic Network Management", in the Proc. of 1st Conf. on Modeling Autonomic Communication Environment (MACE), Dublin, Ireland, pp. 1-20, October 2006.

[114] M. Ripeanu, I. Foster, A. Iamnitchi, "Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design", IEEE Internet Computing Journal (special issue on peer-to-peer networking), vol. 6:(1), 2002

[115] E. Lua, J. Crowcroft, M. Pias, R. Sharma, S. Lim, "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes", IEEE Communications Survey and Tutorial, pp. 72-93, March 2004.

[116] giFT project, "The FastTrack Protocol", available at: http://cvs.berlios.de/cgi-bin/viewcvs.cgi/gift-fasttrack/giFT-FastTrack/PROTOCOL

[117] G. Kan, "Gnutella", in: A. Oram (ed), "Peer to Peer: Harnessing the Benefits of Disruptive Technologies", Chapter 8, O'Reilly, Sebastopol, CA, pp. 94-122, March 2001.

[118] M. Kleis, E. Lua, X. Zhou, “Hierarchical Peer-to-Peer Networks using Lightweight SuperPeer Topologies”, in Proc. of 10th IEEE Symposium Computers and Communications (ISCC), Cartagena, Spain, pp. 143–148, June 2005.

[119] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, "A scalable content-addressable network", in Proc. of 2001 IEEE Conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), pp. 161-172, October 2001.

[120] D.P. Rowstron, “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems”, Lecture Notes in Computer Science, Vol. 2218, pp. 329–350, 2001.

[121] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, H. Balakrishnan, “Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications”, in Proc. of the ACM conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), San Diego, USA, pp. 149-160, August 2001.

[122] B.Y. Zhao, L. Huang, J. Stribling, S.C. Rhea, A.D. Joseph, J.D. Kubiatowicz, “Tapestry: a resilient global-scale overlay for service deployment“, IEEE Journal of Selected Areas in Communications, Vol. 22:(1), pp. 41-53, 2004.

[123] P. Maymounkov, D. Mazieres, “Kademlia: A Peer-to-peer Information System Based On the XOR Metric”, in Proc. of the 1st International Workshop on Peer-to-Peer Systems IPTPS ’02, Cambridge, USA, pp. 53-65, March 2002.

[124] M. Cai, M. Frank, “RDFPeers: A Scalable Distributed RDF Repository based on a Structured Peer-to-Peer Network”, in 13th International Conference on World Wide Web, New York, NY, pp. 650-657, May 2004.

[125] G. Jesi, A. Montresor, O. Babaoglu, “Proximity-Aware SuperPeer Overlay Topologies”, in Proc. of 2nd IEEE International Workshop on Self-Managed Networks, Systems & Services (SelfMan), Dublin, Ireland, pp. 74-83, June 2006.

[126] A. Mizrak, Y. Cheung, V. Kumar, S. Savage, “Structured SuperPeers: Leveraging Heterogeneity to Provide Constant-Time Lookup”, The IEEE Workshop on Internet Applications (WIAPP), San Jose, CA, pp. 104-111, June 2003.

[127] M. Adler, R. Kumar, K. Ross, D. Rubenstein, T. Suel, D. Yao, “Optimal Peer Selection for P2P Downloading and Streaming”, in Proc. of IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Miami, FL, pp. 1538-1549, March 2005.

[128] B.J. Overeinder, E. Posthumus, F.M.T. Brazier, “Integrating peer-to-peer networking and computing in the AgentScape framework”, in Proc. of 2nd IEEE International Conference on Peer-to-Peer Computing, pp. 96-103, 2002.

[129] P. Kersch, R. Szabó, "DHT routing analysis in a logarithmically transformed space", Peer-to-peer networking and applications, Vol.1:(1) pp. 64-74, 2008.

[130] L. Cheng, K. Jean, R. Ocampo, A. Galis, P. Kersch, R. Szabo, “Secure Bootstrapping of Distributed Hash Tables in Dynamic Wireless Networks”, in Proc. of IEEE International Conference on Communications (ICC), Glasgow, UK, pp. 1917-1922, June 2007.

[131] Z.L. Kis, R. Szabo, “Scalable Merger of Chord-rings”, International Journal of Communication Networks and Distributed Systems, Vol.4:(4) pp. 376-388, 2010.

[132] R. Moskowitz, P. Nikander, P. Jokela, T. Henderson, “Host Independent Protocol”, IETF RFC 5201, April 2006.

[133] G. Karypis, E. Han, V. Kumar, "Chameleon: Hierarchical Clustering Using Dynamic Modeling", Computer Journal, Vol.32:(8), 1999.

[134] N. Niebert (ed), T. Petersen, P. Poyhonen, M. Barachi, Cs. Simon, J. Sachs, J. Lundsjo, B. Tharon, T. Rinta-Aho, F. Meago, M. Kapman, “Common Ambient Networks Use Cases”, Ambient Networks project report IST-2002-507134-AN/R1-8, December 2005.

[135] J. Nielsen et al. (incl. Cs. Simon),"Requirements and Proposals for Networks of the Wireless World: WWRF WG3 (CoNet) Whitepaper", pp. 1-81., Wireless World Research Forum, WG3 Communication architectures (CoNet), 2005.

[136] B. Ohlman (ed) et al. (incl. Cs. Simon), “Ambient Networks Framework Architecture”, Ambient Networks public deliverable D1-5, IST–2002-507134-AN/WP1-D05, December 2005.

[137] N. Niebert, M. Prytz, A. Schieder, N. Papadoglou, L. Eggert, F. Pittmann, C. Prehofer, “Ambient networks: a framework for future wireless internetworking”, in Proc. of IEEE 61st Vehicular Technology Conference (VTC 2005-Spring), pp. 2969-2973, June 2005.

[138] J.M. Kwon, S. Fahmy, “Active Queue Management (AQM) algorithm implementations in the ns-2 network simulator”, available at: http://www.cs.purdue.edu/homes/fahmy/software/aqm/ - last updated: July 2005.

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

[140] Abilene Network, available at: http://www.internet2.edu/pubs/200502-IS-AN.pdf, February 2005.

[141] R. Hulsermann, M. Jager, “Evaluation of a Shared Backup Approach for Optical Transport Networks”, in Proc. of 28th European Conference on Optical Communication ECOC 2002, Copenhagen, Denmark, September 2002.

[142] M. Ronai (ed) et al. (incl. Cs. Simon), “Integrated Design for Context, Network and Policy Management”, Ambient Networks Phase 2 public deliverable D10-D.1., FP6-CALL4-027662-AN P2/D10-D.1, December 2006.

[143] A.G. Prieto, R. Stadler (ed.s), et al. (incl. Cs. Simon), “Ambient Network Management - Proof of Concepts and Evaluations”, Ambient Networks public deliverable IST-2002-507134-AN/WP8/D8-3, December 2005.

Abbreviations

3GPP – 3rd Generation Partnership Project
ACS – Ambient Network Control Space
AN – Ambient Network
ANI – Ambient Network Interface
AQM – Active Queue Management
CBQ – Class Based Queuing
DMF – Distributed Management Framework
FE – Functional Entity
FIFO – First In First Out
INM – In-Network Management
IP – Internet Protocol
ISP – Internet Service Provider
LSA – Link State Advertisement
MIB – Management Information Base
NBP – Network Border Patrol
OSPF – Open Shortest Path First
P2P – Peer-to-Peer
PAN – Personal Area Network
PHB – Per-Hop Behavior
QDP – Quality Differentiation Parameter
QoS – Quality of Service
RED – Random Early Drop
SON – Self-Organizing Network
TCP – Transmission Control Protocol
UDP – User Datagram Protocol
WFQ – Weighted Fair Queuing

Acknowledgements

I would like to thank

- to my parents for everything, to Zsolt for the good example and to Gyöngyi for her support.
- to József Bíró, Tamás Henk, Róbert Szabó, Sándor Székely and Attila Vidács for their supervision and advices during my research.
- to Ericsson and EU for funding much of my work.
- to my colleagues at the Department of Telecommunications and Media Informatics for help and encouragement.
- to my students and my co-authors.

Appendix A – Basic queuing mechanisms for flow shaping

In this case study we tested the effect of basic shaping mechanisms on non-responsive (UDP) and responsive flows (TCP). If more flows compete for the (bottleneck) capacity of the outgoing link, a FIFO queuing in the router drops proportionally from the incoming flows, therefore proportional differentiation can be achieved this way. For UDP flows we expect that this mechanism works well, but not for TCP flows. TCP tries to adapt to the available bandwidth, therefore the assumption that at bottleneck the FIFO queuing will drop proportionally from flows is not valid any more, in ideal case TCP flows will share equally the bandwidth.

An alternative solution is to police or shape the flows at network ingress points. We police non-responsive flows to enforce the differentiation in an active manner at the ingress routers, but this differentiation would be done anyway by the FIFO queues. We use shaping for responsive flows in order to enforce the differentiation on them, a necessary step to achieve the required differentiation, which can not be achieved by FIFO queues.

We used OPNET simulation tool to validate our assumptions. The test network has two clients, two servers and four routers (see Figure 47). The clients generate constant rate (CBR) traffic to the servers, Client 1 to Server 1 (64kbps) and Client 2 to Server 2 (32kbps). The bottleneck is the link between node_2 and node_3, 64kbps. In this network node_2 has FIFO queuing, while the ingress routers (node_0 and node_1) are prepared to police/shape the traffic. Note that if these mechanisms at the ingress routers are not activated, the flows arrive in the FIFO queue at node_2, therefore they will be adapted to the bottleneck there. On the other hand, if the mechanisms are active, then they already shape/police the flows to the desired bandwidth capacity at the ingress points, the node_2 – node_3 link will not be a bottleneck anymore. As for the rate shaping we have used committed access rate (CAR) [76] at node_0 and node_1. CAR is a Cisco proprietary mechanism that enforces a given output rate by dropping the excess packets.

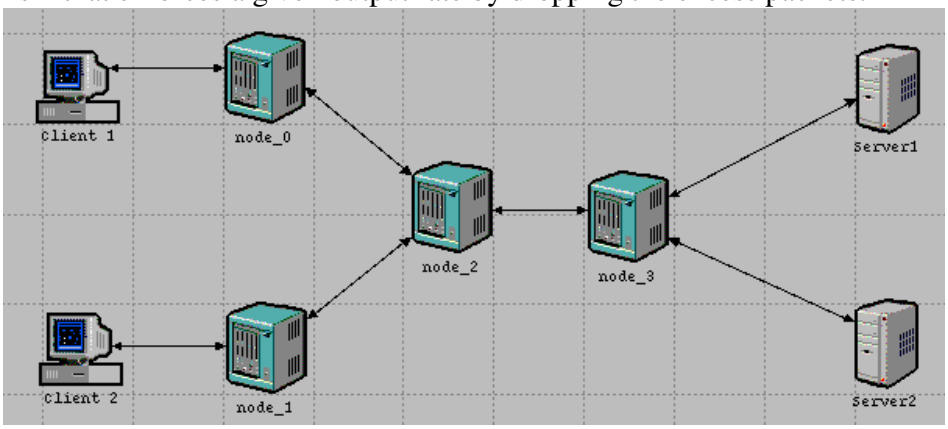

Figure 47. OPNET Simulation scenario

We have considered four scenarios for testing. First, we have used CAR shaping at ingress in order to achieve proportional shares at the bottleneck. Therefore, a rate of

42.666kbps was admitted for the flow **A** from *Client 1* to *Server 1*, and a rate of 21.333 for flow **B** from *Client 2* to *Server 2*. We will show in the lower part of the graphs the offered load and the upper graphs the throughput after the bottleneck for both flows.

A.1 Non responsive flows

Figure 48 shows the results of simulations with enabled CAR. Note that in the first 5 minutes of the simulation flow **A** did not start yet, but the flow cannot use the bandwidth because of policing.

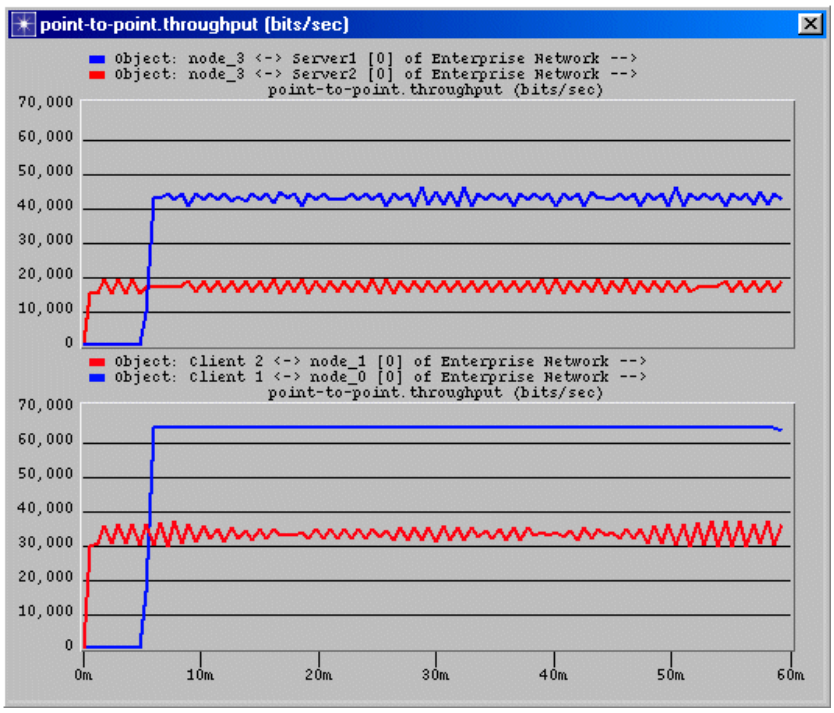

Figure 48. UDP flows with CAR

Figure 49 shows the results with CAR shaping disabled for UDP flows. As we have expected, traffic was proportionally dropped from both flows. Note that compared to the CAR scenario, in the first 5 minutes the flow B has used the excess bandwidth, because it is not restricted at the ingress.

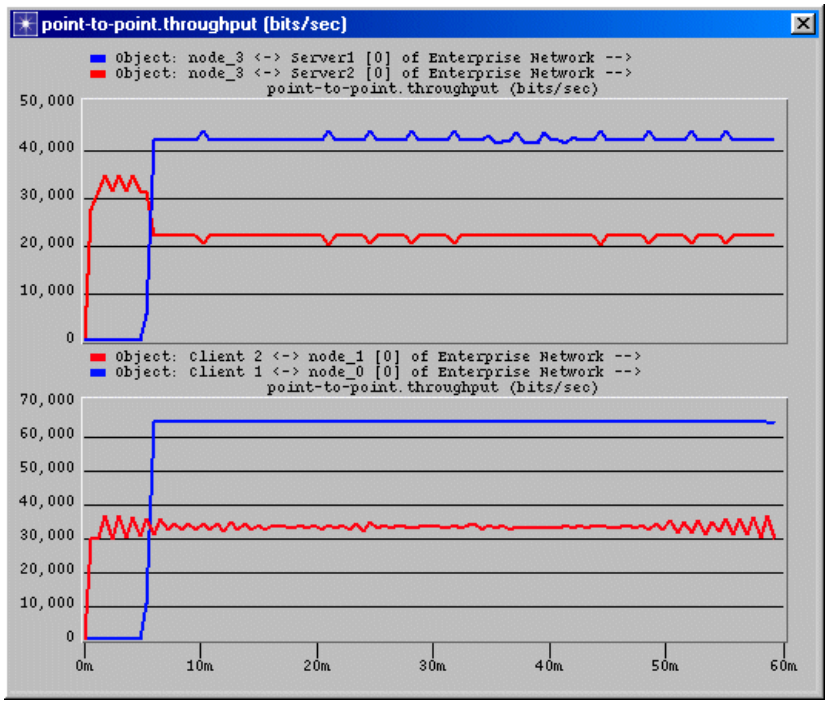

Figure 49. UDP flows without CAR (only FIFO at bottleneck)

As we can see, in both cases the ratio of the resulted throughputs is the desired one. Based on this simple scenario with one bottleneck in the network we should expect that there is no need for dedicated queuing mechanisms to assure differentiation among non-responsive flows. Simple FIFO queues can preserve the differentiation ratio existing between the flows prior entering the bottleneck link.

A.2 Responsive flows

For TCP flows, burstiness can be observed at graphs. Using CAR shaping (Figure 50) each flow gets its share of bandwidth as we set. The offered load is similar to the bottleneck load, since TCP reacts to the losses by reducing its bandwidth. Nevertheless, the CAR shapes TCP flows just with packet drops and does not take into account the inner mechanisms of TCP [77]. That is the reason why on short scale the achieved differentiation is not acceptable.

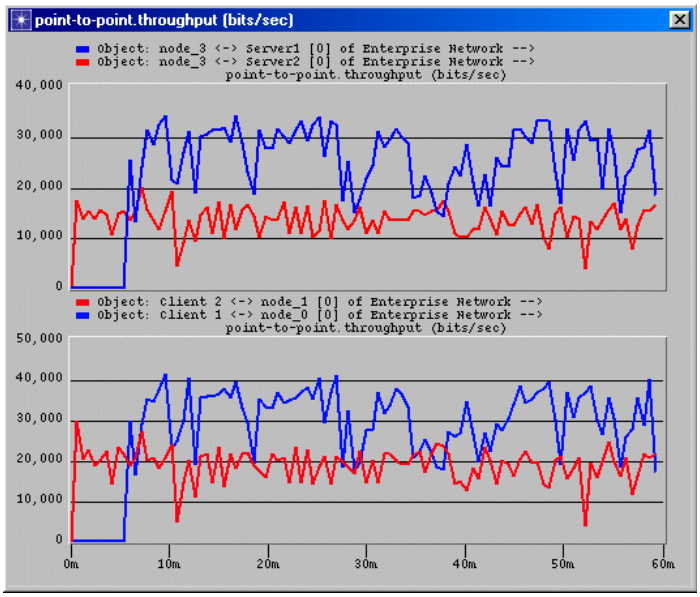

Figure 50. TCP flows with CAR

If we do not use rate shaping for TCP flows, the ratio between the throughputs of the competing flows does not depend on the offered load. The flows will share the bottleneck capacity equally, (Figure 51), independent on the offered load. If a number of TCP flows share the same bottleneck, the capacity will be divided equally. Note that because there are no “active” packet drops induced by the queues, TCP can adapt to the available capacity and is less bursty compared to the previous case.

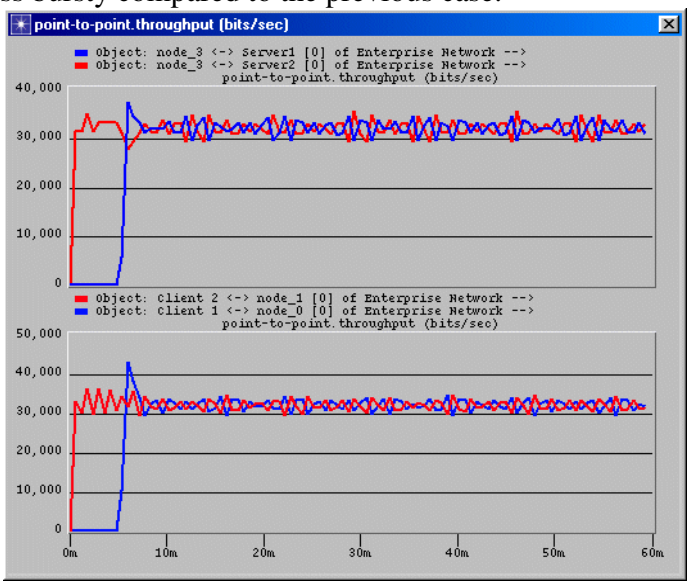

Figure 51. TCP flows without CAR (only FIFO at bottleneck)

Based on these investigations we can say that responsive flows can not be shaped with the above mechanisms.

Appendix B – The advantages of BLUE active queue management

Simple queuing mechanism cannot assure the desired policing/shaping of responsive flows at the ingress, as shown in Appendix A. We have selected the BLUE active queue management mechanism [138] to handle responsive (TCP) flows at the ingress. In the first part of this section we compared the behavior of BLUE against FIFO to verify if it provides a better solution indeed. In the second part of this section we compared the BLUE against a different active queue management solution to motivate our selection.

B.1 The advantages of BLUE over FIFO

First, we have verified that the proposed BLUE AQM behaves better than the simple FIFO. In the ingress nodes the BLUE active queue management implementation of [138], to avoid network congestion, and to assure fairness in terms of throughputs for the different micro-flows.
Two simulations were performed, one with a FIFO queue serving the bottleneck link and a different one, when the BLUE AQM is deployed at the bottleneck link. The scenario with the BLUE mechanism enforces an arbitrarily selected bandwidth on the aggregated TCP macro-flow. Since, originally, BLUE has been proposed for smooth congestion control, additionally the second scenario will result in much balanced TCP dynamics, as well.
A numerical comparison is given in Table B-1, which shows the average throughput of the 20 flows, sharing the same bottleneck link. The FIFO queuing results in a two order of magnitude higher standard deviation among micro flows, which means that the FIFO discipline does not assure fairness among micro flows within the same class.

TABLE B-1
Flow throughputs and dispersion

	FIFO	BLUE
Average	21077.56	20916.2
Standard deviation	27364.55	804.7

Additionally, we simulated a third scenario, where the path delays between the sources and node 0 (the input to the BLUE queue) were randomly distributed. This scenario simulates the case of the so called asymmetric TCP connections, where the round trip times (RTT) of the competing flows are different and the regular FIFO queues may introduce differences (unfairness) in terms of transfer rates. The links connecting the sources to the bottleneck link has been randomized with the values ranging from 0 to 2.5ms. Compared to the standard deviation value (804) for the symmetric case, it yielded a standard deviation value of 1205. This proves that BLUE successfully ameliorates this phenomena, since the resulted standard deviation has the same order of magnitude than the one obtained for the symmetric links (second column).
As a conclusion, after evaluating the results, one can tell that the BLUE outperforms FIFO, by shaping the bandwidth of the aggregated flows to the target value (in this case the bottleneck capacity of the link), also providing fairness in the bandwidth allocation of

individual micro-flows. And this was our purpose, to find a corresponding shaping mechanism to assure fairness within the flow-classes. There are no starving flows; the micro-flows share equally the available bandwidth.

B.2 The advantages of BLUE over RED

As we stated earlier, we surveyed the literature to find an acceptable AQM mechanism. The most referenced such mechanism is the RED [43] and its variants. Nevertheless, we did not opt for this AQM family, because it achieves the flow shaping with losses, instead of much finer methods. Packet drops in TCP results in abrupt throughput changes and even timeouts, which might lead to severe unfairness. Since we shape at flow level, we have no means to observe and react to such events if they appear. That is the main reason for selecting a different AQM. We did a set of simulations to verify that BLUE is dropping packets only in extreme congestion situations. An illustrative example is shown in Figure 52 below, which depicts a typical packet loss profile (expressed in lost kilobits) experienced by ingress routers deploying BLUE in our simulations presented in section 4.3.6.

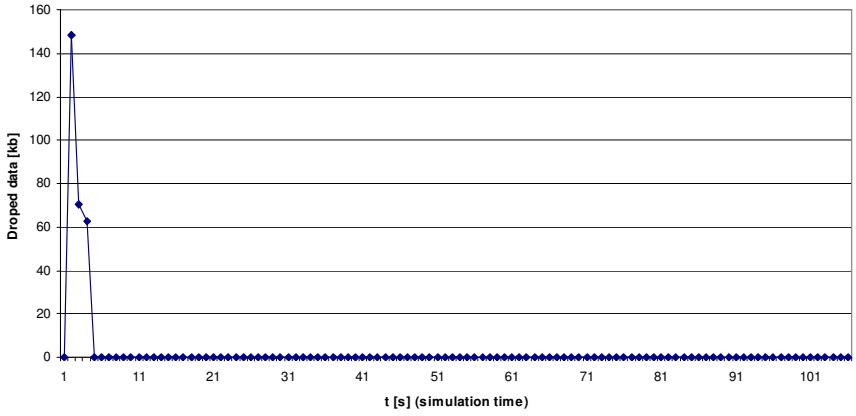

Figure 52. Data loss at BLUE AQM

We examined the shaping qualities of the BLUE AQM mechanism, by evaluating the losses (drops) at the ingress routers. We have found that losses appear just in the very beginning of the data transmission, when the TCP flows quickly increase their bandwidth in slow start. These losses appear because the shaping BLUE mechanism needs time to inform the sources to reduce their rates, and packet drop is the most efficient way to achieve this. The BLUE shaping mechanism is working well, with tolerable, small losses in the beginning of TCP traffic generation, and without losses afterwards. The fact that a BLUE based system is able to adapt to the dynamics of a variable TCP traffic has been extensively tested and published in [81][82].

Appendix C – Simulation scenarios

Scenarios with multiple bottlenecks and complex topologies are required for algorithm evaluation. We implemented the network wide proportional services architecture in a simulator tool. The simulator used is *ns2*, a widely used simulation tool in the IP-research community. The simulator is available at [139]. Complete documentation is also available at this location. The simulator has discrete event driven architecture, and is written in the combination of C++ and OTCL programming languages, following the object-oriented programming paradigm. It has a detailed implementation of different flavor TCPs and several traffic generators.

In *ns2*, each simulation scenario is configured and started based on configuration file, written in a script-language [139]. This script will configure the variables in the simulator and will schedule the events start and stop times during the simulation time.

C.1 Network topology

Through the simulations, we have tested our algorithms in three different topology scenarios. For each of the network scenarios we had two types of traffic patterns, one with static flows and one with variable load flows, as detailed later.

Based on our simulation experiences the interpretation of the results in the case of more classes and paths is quite difficult. Therefore, for illustrative purposes, we have selected the simplest network topology (demo scenario) with only four flows and graphs showing the results of such scenarios were presented in Chapter 5. As can be seen in the case of varying TCP flows (see section 5.1.6), the interpretation of results is difficult in such case, as well.

We built our simulated networks using the following types of nodes:

traffic generators: these nodes generate the traffic flows, which enters into the domain through one of the ingress nodes. Besides the traffic generation function, these nodes have no more functionality. In the graphical interface, traffic generators are represented with red rectangles.

core routers: these nodes could be ingress, egress or normal core nodes. The algorithms could be implemented in some of these nodes. In the graphical interface, they are represented with black circles.

traffic sinks: routers where the flows are terminated. In the graphical interface, they are represented with blue circles.

Demo scenario

The first network topology can be found in Figure 53. As we can see we had four traffic generators, all of them generated both low and high priority traffic flows. Because the results simulated in this network are used as an illustration, we give the colour codes for each flow, as well.

Parameters for the simple network scenario:

- Traffic source-destination pairs:
  Flow D 5 – 3, drawn in dark blue in the figures
  Flow B 8 – 4, drawn in red in the figures
  Flow A 7 – 3, drawn in yellow in the figures
  Flow C 6 – 4, drawn in dark green in the figures
- all flows have 1 MBps sending rate
- link 2 - 4 has 0.75 MBps link speed and 1ms link delay
- link 1- 3 has 2 MBps link speed and 1ms link delay
- the rest of the links have 3 MBps link speed and 2ms link delay

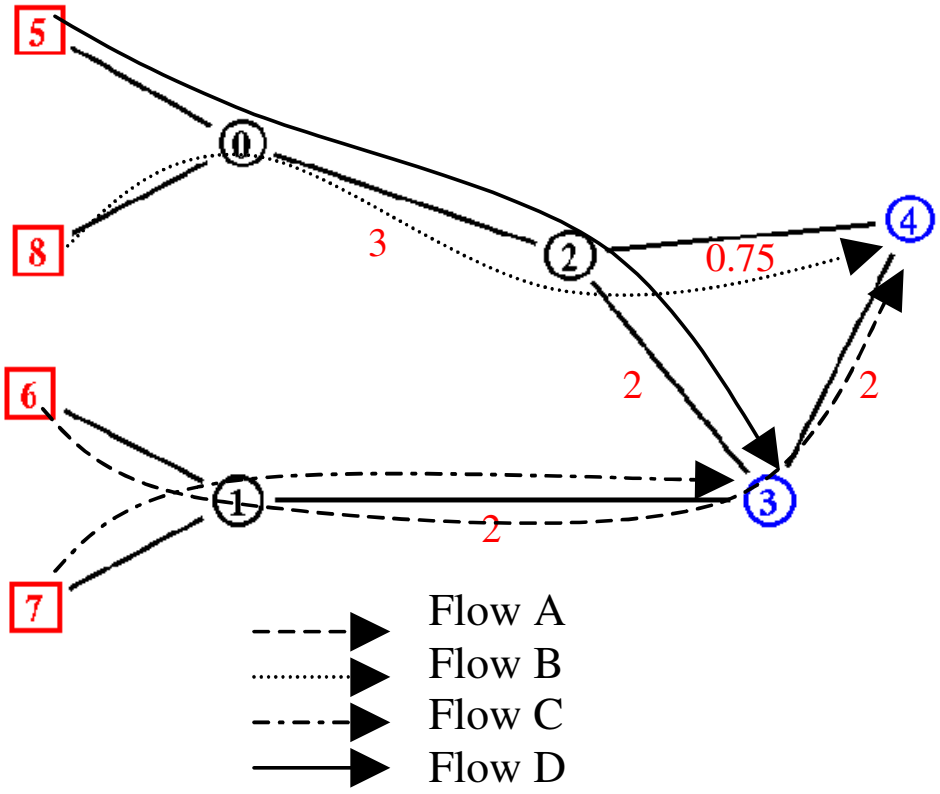

Figure 53. Demo network scenario

US backbone scenario

Based on the US backbone reference scenario of the Abilene network ([140], also used in [12]) we generated a more complex topology. We kept the 9 core routers (12 with the additional three sinks) and the 14 simulated links of the original network, but we reordered the internal links. In this case we had 6 traffic generators, all of them generated both low and high priority traffic flows. The resulting topology is displayed in Figure 54.

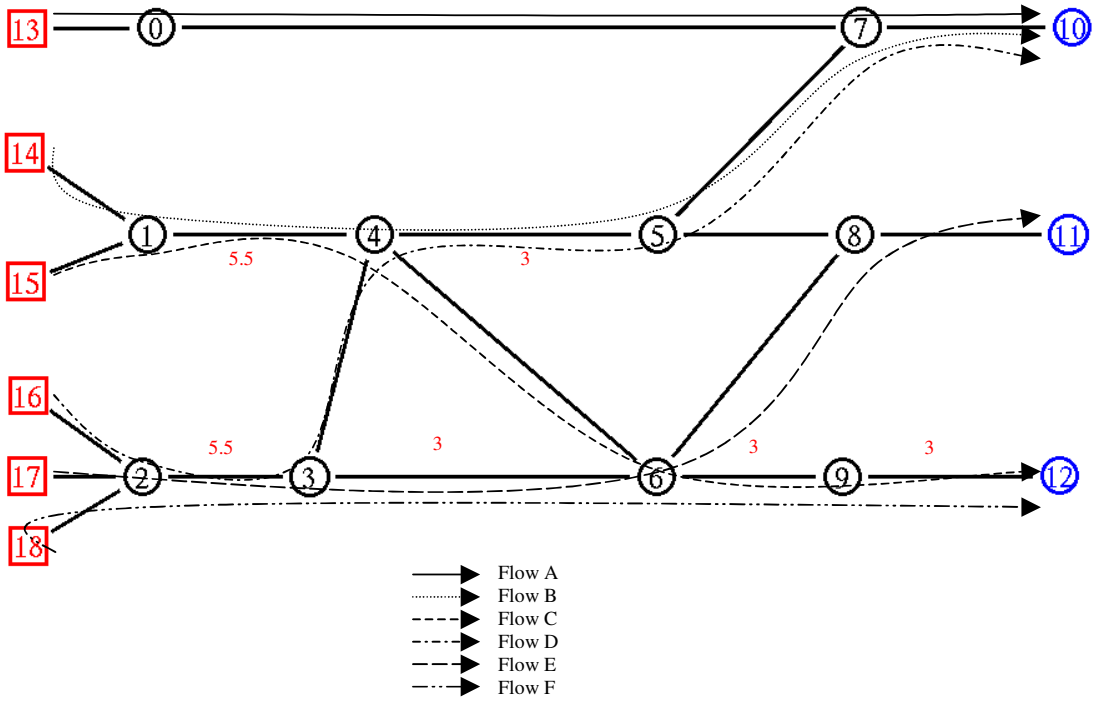

Figure 54. An US national backbone network scenario

Parameters for the 9 core node network scenario:

- Traffic source-destination pairs:

  Flow A: 13 – 10

  Flow A: 14 – 10

  Flow A: 15 – 12

  Flow A: 16 – 10

  Flow A: 17 – 11

  Flow A: 18 – 12
- all flows have 1 MBps sending rate
- links 1 - 4 and 2 – 3 have 5.5 MBps link speed
- the rest of the links have 3 MBps link speed

European network scenario

A 15 node scenario was generated with multiple bottlenecks and loops (Figure 55), modeling a national backbone with 18 routers and 22 links (resembling the hypothetical

German backbone network with 17 routers and 26 links in [141]). The scenario also contains a flow with no bottleneck (Flow C).

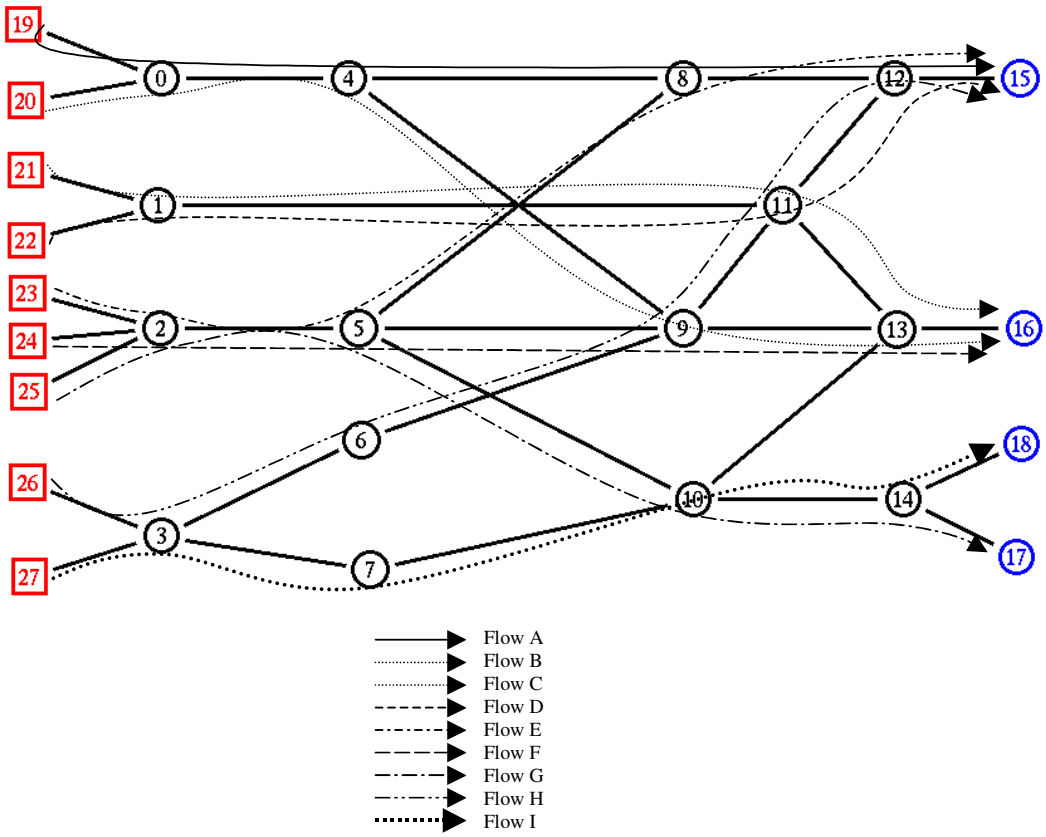

Figure 55. The European national backbone core node scenario

C.2 Simulating the UDP flows

In order to test our implementations we used two traffic patterns. As a first step, we have chosen the simplest possible CBR traffic. This was suitable whether the proportionality conditions for the high and low priorities traffic flows are met.

However, this is a static case, but we need to test the behavior of the network in dynamic conditions, as well. For this purpose, we have generated a mix of CBR traffics, simulating the behavior of the aggregated micro flows. The average bandwidth of each aggregate is set to 1 Mbps, and the bandwidth of each micro-flow is exponentially distributed with mean of 32 kbps (note that the simulated link capacities vary between 0.75Mbps and 8Mbps). In our simulation micro-flows join and leave the aggregate flow according to a Poisson process with the same arrival and departure rate (i.e., $\alpha_{arr} = \alpha_{dep} = 1/4\ sec^{-1}$). This emulates the situation when users start and stop IP streams over the network.

C.3 Simulating the TCP flows

In our preliminary investigations of the properties of BLUE AQM, as presented in Appendix B, we found that already 20 micro flows in a flow result in a complex enough scenarios, meaning that the aggregated flow hides the individual properties of a micro flow, therefore it models the real-life flow aggregation in a backbone network. We bundled such this order of magnitude of micro flows in each flow, as follows.

For the static scenario we generated 50 TCP streams for the high-class flow *A* and 60 streams for the low-class flow A (50/60 micro flows in flow *A*). For the rest of the flows these values were 40/30 micro flows for flow *B*, 30/50 micro flows for flow *C*, and 60/40 micro flows for flow *D*. In the case of the second scenario we also had the 40/30 and 30/50 micro flows for flows E, and F, respectively.

In order to generate variable traffic load in the TCP scenarios, we modified the number of the above enlisted TCP streams. We changed the number of streams at one second intervals. The numbers of micro flows within a certain class were changed by stopping an existing or starting an additional TCP stream. In order to decide whether to stop, start or leave unmodified the TCP streams, we relied on the random generator of the Linux OS.

C.4 Queuing infrastructure

Our algorithms suppose that at on every node the flows with the same traffic class are treated equally. We organized these two classes (high and low class) in two different DropTail (FIFO) queues. The bandwidth of the router had to be shared between two different flow classes and the corresponding queues had to be controlled by a common mechanism for both of the flows. That is the reason why we used Class based Queuing (CBQ) in the network nodes. The weights of the CBQ's can be modified during simulation, in order to achieve the proper bandwidth share between the flows. Note that the ns2 CBQ discipline offer a strict separation of their separate queues.

In the case of TCP flows, the queuing discipline for each flow was BLUE instead of DropTail (FIFO). That means that in the case when we simulated a background UDP flow and a high and low class TCP flows over each path, then a FIFO and two BLUE queues policed/shaped the traffic at each ingress.

Appendix D – Policy framework for network compositions

During the Ambient Networks projects we have proposed a policy framework that is able to govern the self organizing behavior of the domains. This policy framework enables the autonomous cooperation of the SONs [143][142]. The composition strategies of the networks are handled and driven by distributed policy rules contained in each of the components of the network. Although this work was done with ANs in mind, the framework is a generic one, because the ANs expose the SON properties required in section 9.1.1. We already discussed this property of ANs in section 10.2.2. We present this policy framework in this section.

D.1 Policy Data model

A policy is defined by three elements:

- profile,
- statements,
- rules.

Profile

The profile is a set of properties of the peer considered to be relevant during negotiation with other peers. Each property is represented by a key-value pair. Neither the keys nor the values are restricted in either type or value, they are arbitrary strings. A small set of predefined keys may exist, but most keys are defined by the peers.

Examples:

- “memory-capacity” = “10M”
- “company-meeting-participant” = “yes”
- “nationality” = “Hungarian”

Statements

Statements express relations between the profile of the peer and the profile of a prospective negotiation partner. The three parts of the statement expression are:

- key of a property in the profile of the negotiation partner,
- relational operator,
- key of a property in the profile of the peer or a constant.

Examples:

- “software-version” >= “software-version”
- “company-meeting-participant” = “yes”

Rules

The rules are parameters of the negotiation algorithm. Rules consist of two parts:

- condition,

- qualified set.

The condition is a boolean expression. This expression contains references to statements and boolean operators (AND, OR, NOT). By substituting the profile information of the negotiation partner into the statements, the condition can be evaluated to either true or false. The condition controls whether the rule is applied during negotiation or not.
The qualified set is a set of qualified references to statements. The qualified set qualifies each referenced statement with one of the following qualifiers:

- MUST,
- SHOULD,
- DON'T CARE,
- SHOULD NOT,
- MUST NOT.

The qualifiers define the logical value of the evaluated statement required for successful negotiation. Table D-1 defines the meaning of the qualifiers.

TABLE D-1
Qualifiers of the policy framework

Effect on the outcome of the negotiation		Statement value	
		True	false
Qualifier	MUST	None	failure
	SHOULD	None	none
	DON'T CARE	None	none
	SHOULD NOT	None	none
	MUST NOT	Failure	none

The qualifiers SHOULD and SHOULD NOT are equivalent to the DON'T CARE qualifier during negotiation, they are treated differently by the maintenance algorithms, however. In order to ensure that the policy is consistent with itself, contradictions between rules are detected when new rules are created. Rules contradicting with existing rules cannot be added to the policy. An example of a policy data set of a node (peer) is depicted in Figure 56.

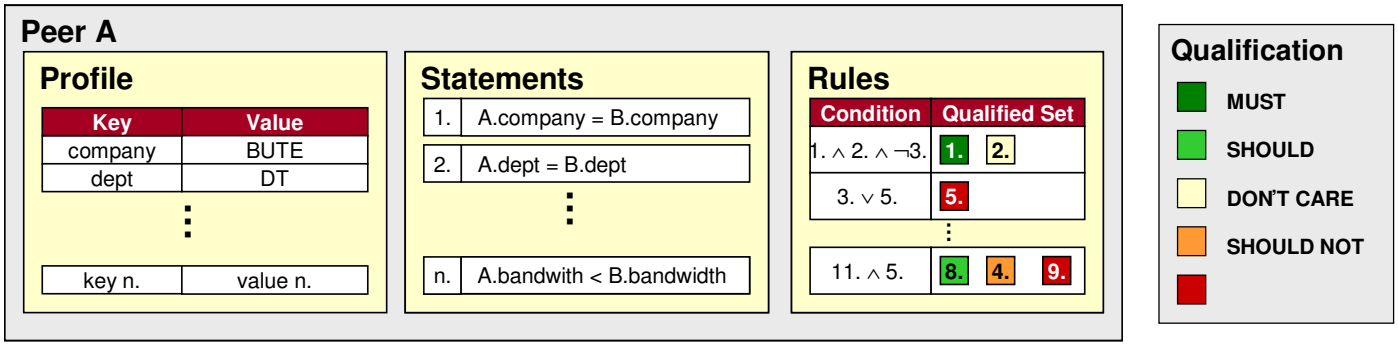

Figure 56. The data model of the policy framework

Negotiation algorithm

The negotiation algorithm is used to decide if two policies are mutually acceptable. The steps of the algorithm are the following:

1. The qualification of each policy statement is set to DON'T CARE.
2. The peers exchange their profiles.
3. For each rule the condition is evaluated. If it evaluates to true, the qualifiers in the qualified set are applied to the statements.
4. The peers compare the qualified statements.
5. If there is a conflict, conflict resolution is attempted.
6. The negotiation is considered successful if there are no conflicts.

D.2 SON policy

The policy of an SON is an aggregate of the policies of the individual peers. The way peer policies form the SON policy is determined by the maintenance algorithms. The SON policy is used whenever the SON is required to present itself as a single entity from the policy perspective. An example would be policy negotiation between a peer and an SON to determine whether the peer joins the SON or not. The SON policy may be stored either by the superpeer of the SON or in a storage distributed across the SON (a DHT for example), and this depends on the design of the management plane of the ACS.

SON policy interpretation

The SON policy contains the same elements as a peer policy (profile, statements and rules). The properties in the profile, the statements and the rules are augmented with cardinalities, that determine the importance of the element. The cardinality is the number of peers whose policy also includes the given element. This implies that there are two options for handling an SON policy during negotiation.
The negotiation algorithm is generalized to make use of the cardinalities. In a peer policy all elements have a cardinality of one.
The SON policy is transformed into a peer policy before the negotiation takes place. This involves eliminating unimportant elements, and resolving conflicts between elements by a conflict resolution algorithm, for example by simple majority decision.

Maintenance

The goals of policy maintenance are the following:

- merge the policy of a peer joining the SON into the SON policy,
- update the SON policy when a peer leaves the SON, and
- converge the SON policy towards the essence of the policies of the SON member peers.

Merging algorithm

The merging algorithm is designed to fulfill the first and third requirements. The steps of the algorithm are the following:

1. The elements of the peer profile are added to the SON profile. New elements are added a cardinality of 1, cardinalities of existing elements are incremented by 1.
2. The profile is reduced by removing conflicting profile properties. To this end the ratio of the cardinalities of the conflicting properties is calculated, this results in a value between 0 and 0.5. Depending on the value the property elimination happens in one of two ways:
   - If the ratio is smaller than a threshold, the property with the smaller cardinality is discarded (it is not significant),
   - otherwise both properties are discarded (they are conflicting).
3. The significance of statements is determined based on their cardinality and the cardinalities of the properties in them. Conflicting statements are discarded based on the method used for profile properties.
4. Rules are simplified. This involves the following steps:
   - The significance of the rules is determined based on the significance of the statements in them.
   - Nonexistent or insignificant statements are removed from the conditions.
   - Nonexistent or insignificant statements are removed from the qualified sets.
   - Conflicts between rules are resolved based on the qualifiers in the qualified sets. Two rules are potentially conflicting if there is such an evaluation of their conditions that they become both applicable. In this case conflict resolution is done for all statements that appear in the qualified set of both rules with different qualifiers. Table D-2 shows the result of conflict resolution between the qualifiers.

TABLE D-2
Policy conflict resolution

Resolution	MUST NOT	SHOULD NOT	DON'T CARE	SHOULD	MUST
MUST	rule discarded	*	MUST	*	N/A
SHOULD	*	*	SHOULD	N/A	
DON'T CARE	MUST NOT	SHOULD NOT	N/A		
SHOULD NOT	*	N/A			
MUST NOT	N/A				

*: Multiple alternatives, depends on the implementation of the algorithm.

   - Rules that have become empty or insignificant are discarded.

Appendix E – Use Cases for Self Organizing Networks

In this section we describe the use cases drafted within the Ambient Networks projects [C8][84]. Similarly to the logic of the previous section, we consider that even if this work was done with ANs in mind, the results can be used in a generic way, because the ANs expose the SON properties required in section 9.1.1. We already discussed this property of ANs in section 10.2.2.

The goal of defining use cases was to identify those typical interactions that a SON should react to. We chose the method of drafting real life use cases based, which, based on the experiences of our partners in the AN projects, who are the leading vendors of telecommunication solutions and operators of telecommunication networks, cover the behavior of the users [C8][84] [135]. Starting from these use cases we can observe the main characteristics of the networks interaction, which governed our research and which finally led to the definition of the utility function as described in Section 10.

We proposed a generic scenario, called the Rock Express [135]. We follow a rock band's Summer 2015 European Tour during which they use a special rock train for travel between gigs, as a concert stage and also to host exclusive interviews and present material to special guests and fans that pay to travel with the band. Multiple ANs are set up between different actors on board the trains as well as between actors on and off the train. Temporary ANs will also be set up at the concerts to facilitate information sharing and content distribution between the band and the audience and with friends not able to attend the concert. Potentially, all these ANs will be using different access technologies, and end users will be minimally affected when their connection is transferred from one access technology to another, or when new traffic is added to already restricted resources.

Starting from the Rock Express scenario we drafted a specific scenario with well defined sub-scenarios, named use cases. This scenario focuses on the train as the common public meeting point, where several types of networks interactions. This environment has the two most important properties attributed to the SONs under study: heterogeneity and dynamicity. Whereas it still keeps these important features of future networking environment, it allowed us to identify the actors and the interactions of the SONs [C8][134].

E.1 Scenario description

Bob is on his way to the railway station by bus with active sessions as he is synchronizing his large mailbox between the server and his PDA. At the same time, Bob is also watching some news via streaming on his PDA and since there were no real-time needs, Bob's news application on his PDA selected Bob's best-effort subscription, which is cheaper than requiring stringent QoS (note: the best effort subscription is selected by using context information on QoS parameters). When Bob arrives at the railway station, with the active Internet connection, his Personal Area Network (PAN) automatically adapts to surrounding networks and detects a WIMAX hotspot in the station, which is then taken into use to be able to use faster and cheaper Internet connection.

Another user at the railway station, John, switches on his PAN, which discovers the advertisements sent out by Bob's PAN, which is acting as a relay for the WIMAX. John connects to the WIMAX hot spot network (unaware that Bobs PAN acts as a relay) and starts a session.

Once Bob's PAN starts to relay traffic between WIMAX hotspot and other users in the station area, his PAN can be seen as an access extension and users whose traffic has been relayed are not aware of relay node(s) since they provide the same set of services than the WIMAX hotspot.

Alice, a working mate of Bob's arrives to the station as well. They both will travel from Aachen to Bruxelles to attend a meeting. Their gadgets form two PANs. Then they meet at the station and they embark a coach(train). At this time they will join the network of the coach, as it is operated by the railway company (e.g., a German one) they have a frequent traveler agreement. The train network is a stand alone moving network. Later on, a locomotive of other railway company (e.g. Belgian one) is connected to the train, thus the network of the train will be 'extended' (gatewayed composition). As the train arrives to the destination, the locomotive is connected to the station's network (hot-spot). Since the station's network is operated by a third party ISP, this connection also results in a gatewayed composition. Finally, Alice and Bob disembark from the train and head to the office (meeting rooms).

E.2 Network compositions in the proposed scenario

E.2.1 Users connecting to the public mobile network

Figure 60, represents an overlay configuration from PAN's perspective, where the PAN and UMTS networks are composed through gatewaying. We supposed that the UMTS operator's infrastructure is forced by the operator to form a stand-alone management domain.

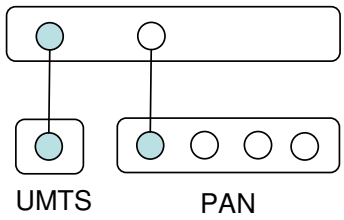

Figure 57. The UMTS-PAN overlay while Bob's PAN is connected to the UMTS network.

If other users are also attached to the UMTS service, they will also compose with the UMTS network through gatewaying. Figure 61 illustrates such a scenario, when besides the PAN two more users have UMTS access.

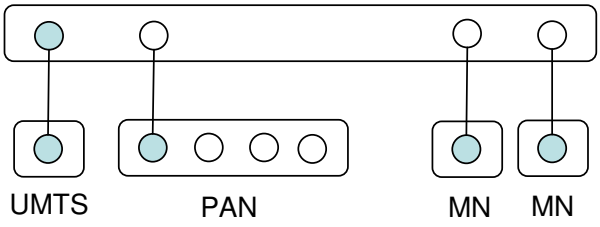

Figure 58. The overlay with all attached devices.

E.2.2 Users connecting to the hotspot of the station

The radio network SON (a WLAN hotspot and the related RN_AN, as presented in Figure 60) should be prepared to advertise over its SON Interface the mobility services of the WLAN. Note that, as already explained in the first part of this Appendix, Bob is advertising the WLAN access and the rest of the actors will join the hotspot only through this Bob+hotspot SON (the RN_AN). This means that RN_AN should be able (or: should be configured) to provide access to other SONs. As John's PAN is detected by Bob's PAN there is a decision to be made: which SON should join John – RN SON or Station SON. Due to the required connectivity and mobility access criteria John will join the Station SON. Therefore the two SONs negotiate, agree and realize the composition process.

Figure 60 presents the management overlays after John has decided to join the Station SON. The Figure depicts the situation when not only John, but also other users (represented by the Other_MN node) decide to join the Station SONs. In this case these users form an intermediate overlay (named MN_AN) making the SON of the station (named Station_AN) less populated, thus increases the scalability of the network.

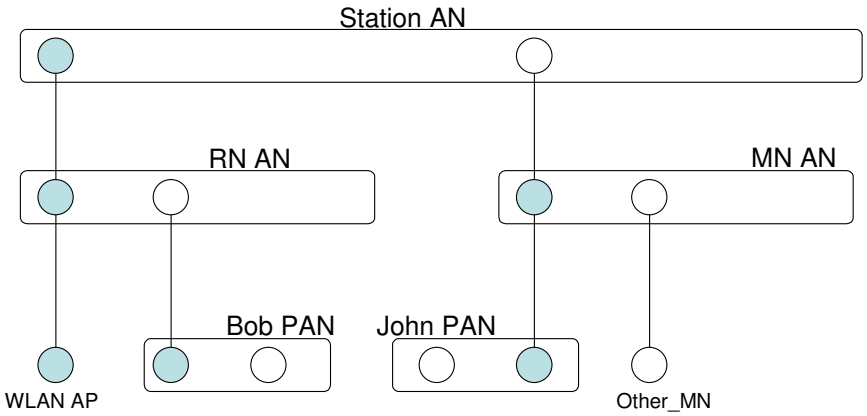

Figure 59: The WLAN hotspot hierarchy of the train station.

E.2.3 Forming the train network

As new SONs are joining the train, the compositions should reflect the new interactions. This leads to several composition procedures. First, the passengers form the Routing Group (a moving network), which leads to a composition. This network is composed with the coach's SON (Coach AN). Note that although the coach and its passengers are moving together, Alice and Bob preferred to form their own network first. Then, as the locomotive is attached to the train, a gatewayed composition leads to the formation of the Train_AN. When the train is connected to the UMTS network, it will have a gatewayed composition. Figure 63 reflects this final structure of the Global SON.

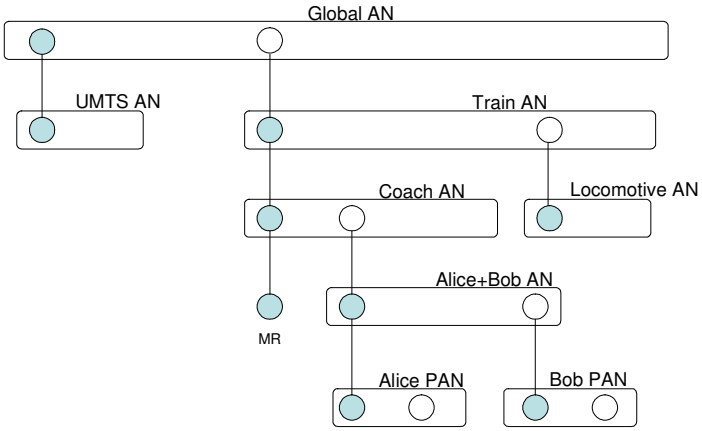

Figure 60: The composed SONs on the train.

E.2.4 Passengers leaving the train

This scenario depicts the situation created after the train arrives to the destination and the passengers leave the train at that (foreign) station.

Figure 64 shows the composition structure as both Alice and Bob left the train and their common A+B SON. This state is reached through multiple steps, as follows. The Train SON de-composes from the UMTS SON and compose with the WLAN SON. The A+B SON leave (de-compose) the Train SON. Alice and Bob leave the A+B SON. A+B SON is discontinued, both Alice and Bob join the WLAN SON.

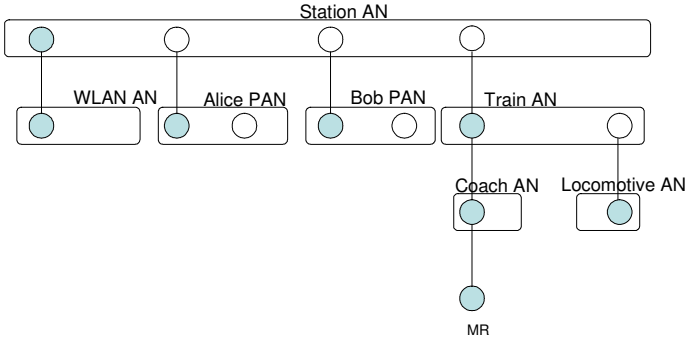

Figure 61: The composed SONs at the station.

Appendix F – Simulating Environment to Investigate Self Organizing Networks

We have simulated the behavior of SONs deploying the bottom-up composition algorithm presented in section 10.1.2. The results of the simulation based investigations are presented in section 10.3.

We implemented an event driven simulator which maintains the layer 2 topology of the nodes and the SON graph. The topology is represented as a connectivity matrix. The SON graph follows the super peer approach: each set of nodes in a SON at a given level of hierarchy is represented by its superpeer on the higher level of the hierarchy. Superpeer is elected randomly from the SON members.

We simulated only the composition process, meaning that data plane events were not simulated. The interaction of two SONs during composition, as given in the algorithm referenced at the beginning of this section, is a combination of three events: absorption, gatewaying and the so called *send up* (see also section 10.1.2). The first two represent the absorption and gatewaying types of composition, while the third one simulates the event when after an unsuccessful negotiation the composition is retried at a higher hierarchy level in the SON graph.

The duration of each event has been established to be equal to the values measured during the performance evaluation of our testbed prototype, as given in section 10.2.4.

We assigned to each of the above three events a cost in terms of messages. These messages cover the costs of composition negotiations and realization, and are based on our implementations [143]. In the case of both the gatewaying and the absorption processes we assosciated 5 messages to the node that initiated this process and 4 messages with the other peer. The send up event costs one message both parties involved. We log the number of messages for each node during the simulation.

If two concurrent events occur at a SON, then we enqueue the one arrived later, similarly as implemented in the prototype.

We have to specify the maximum number of neighbors a node might have. Based on this the simulator randomly selects the number of connections, and builds the connectivity matrix. Similarly, we have to specify the total number of properties in the network and the minimum and maximum numbers of policies a node would have. Then the simulator randomly assigns for each node a number of properties out of the available properties between these two bounds. After the simulation run, based on the logs we compute the ratio between the number of total gatewaying and absorption events. We tested several combinations of these parameters and finally we have selected two sets of parameters, which yield 30% and 5% of gatewaying events. We did our investigations using these two sets of parameters, as presented in section 10.3.

Appendix G – Citations of own publications

[J3] M. Brunner, A. Galis, L. Cheng, J.A. Colás, B. Ahlgren, A. Gunnar, H. Abrahamsson, R. Szabo, Cs. Simon, J. Nielsen, S. Schuetz, A.G. Prieto, R. Stadler, G. Molnar, “Ambient Networks Management Challenges and Approaches”, Lecture Notes in Computer Science, Vol. 3284, pp. 196-216, 2004

> [R1] G Dimitrakopoulos, K Tsagkaris, V Stavroulaki, A Katidiotis, N Koutsouris, P Demestichas, V Merat, S Walter, “A Management Framework for Ambient Systems Operating in Wireless B3G Environments”, Mobile Networks and Applications Vol.13:(6) pp. 555-568, 2008.

[J4] B. Kovács, Cs. Simon, "»Ambient« hálózatok – áttekintés”, Híradástechnika Vol. LX:(7), pp. 39-44, 2005

> [R2] Róbert Szabó, “Internet: siker(!), korlátok(!?) és jövő?”, Magyar Tudomány, Vol.168: (7) pp. 873-879, 2007.

[J6] L. Cheng, R. Ocampo, K. Jean, A. Galis, Cs. Simon, R. Szabo, P. Kersch, R. Giaffreda, “Towards Distributed Hash Tables (De)Composition in Ambient Networks”, Lecture Notes in Computer Science, Vol. 4269, pp. 258-268, 2006.

> [R3] Fatna Belqasmia, Roch Glithob, Rachida Dssoulia, “An overlay network for autonomous information discovery in the post-composition registries of ambient networks”, Journal of Network and Computer Applications, Vol.34: (2), pp. 697-707, 2011.
>
> [R4] Liquori Luigi, Tedeschi Cédric, Vanni Laurent, Bongiovanni Francesco, Ciancaglini Vincenzo, Marinkovic Bojan, “Synapse: A Scalable Protocol for Interconnecting Heterogeneous Overlay Networks”, Lecture Notes in Computer Science, Vol.6091, pp. 67-82, 2010.
>
> [R5] Popi C, Festor O, “A scheme for dynamic monitoring and logging of topology information in Wireless Mesh Networks”, in Proc. of IEEE Network Operations and Management Symposium, 2008 (NOMS 2008), Salvador, Brazil, pp. 759-762, April 2008.
>
> [R6] José Carlos Rufino Amaro, “Co-operação de Tabelas de Hash Distribuídas em Clusters Heterogéneos” Tese de Doutoramento em Informática, Escola de Engenharia, Universidade do Minho, Portugal, 287 p. 2008.
>
> [R7] Francois J, State R, Festor O, “Towards malware inspired management frameworks”, in Proc. of IEEE Network Operations and Management Symposium, 2008 (NOMS 2008), Salvador, Brazil, pp. 105-112, April 2008.

[C3] Cs. Simon, A. Vidács, I. Moldován, A. Török, K. Ishibashi, H. Ishii, "End-to-End Relative Differentiated Services for IP Networks", in Proc. of 7th IEEE Symposium on Computers and Communication, pp. 783-788, Taormina, Italy, July 2002.

[R8] Ruiru Chen, "Dynamical Congestion Control Strategies for a Network of Multi-Agent Systems Subject to Differentiated Services Traffic" PhD Thesis, Concordia University (Dept. of Electrical and Computer Engineering, Montreal, Quebec, Canada, 412 p. 2011.

[R9] Smit J J, Ferreira H C, "Scheduler performance evaluation and the effect of aggregation on QoS in a DiffServ enabled network", in Proc. of 7th AFRICON Conference in Africa (AFRICON 2004), Gaborone, Botswana, pp. 323-328., September 2004.

[C6] Cs. Simon, R. Szabó, P. Kersch, B. Kovács, A. Galis, L. Cheng, "Peer-to-peer Management in Ambient Networks", In Proc. of the 14th IST Mobile and Wireless Communications Summit, pp. 537:1-5, Dresden, Germany, June 2005.

[R10] Guangyu Shi, Jian Chen, Hao Gong, Lingyuan Fan, Haiqiang Xue, Qingming Lu, Liang Liang, "SandStone: A DHT based Carrier Grade Distributed Storage System", In Proc. of 2009 International Conference on Parallel Processing (ICPP-2009), Vienna, Austria, pp. 420-428., September 2009.

[R11] Adriano Fiorese, Paulo Simões, Fernando Boavida, "Assessment of multi-domain network management through P2P" Technical Report of Informatic Engineering Department, University of Coimbra, Portugal, 2008.

[R12] Ramy Farha, "Autonomic service architecture for next generation networks" Doctoral Dissertation, University of Toronto Toronto, Ont., Canada, ISBN 978-0-494-57867-4 2008.

[C9] Cs. Simon, R. Rembarz, P. Pääkkönen, H. Perkuhn, C. Bento, N. Akhtar, R. Agüero, T. Katona, P. Kersch, "Ambient Networks Integrated Prototype Design and Implementation ", in Proc. of 16th IST Mobile and Wireless Communications Summit, pp. 522:1-5, Budapest, Hungary, July 2007.

[R13] Jukka Makela, Kostas Pentikpousis, Vesa Kyllonen, "Mobility Trigger Management: Implementation and Evaluation", International Journal of Communications, Network and System Sciences, Vol.2:(3), pp. 211-221, 2009.

[R14] F Belqasmi, R Glitho, R Dssouli, "Ambient network composition", IEEE Network, Vol. 22:(4), pp. 6-12, 2008.