Self-Configuration and Monitoring of Service Specific Overlay Networks

Imad Abdeljaouad

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for a doctoral degree in Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

The constant growth in network communications technologies and the emergence of Service Specific Overlay Networks (SSONs), coupled with the rapid development of multimedia applications make the management of such technologies a major challenge. This thesis investigates the SSONs management problem and proposes an autonomic architecture, a self-organizing and self-adapting algorithm, and a utility function for monitoring the Quality of Experience (QoE) of IPTV streams in SSONs.

First, we examine the different issues stemming from the autonomic management of SSONs and identify the limitations of existing approaches. We then propose an architecture to ease the management of SSONs by incorporating autonomic computing principles to make SSONs acquire self-management capabilities. The proposed architecture introduces autonomic control loops that continuously monitor network components and analyze the gathered data. An Autonomic System (AS) is comprised of one or more Autonomic Managers (AM) which take control of managing other elements in the network. The proposed architecture highlights the different components of an AM and identifies its purpose. The distributed nature of the proposed architecture avoids limitations of centralized management solutions.

We then propose a scheme to allow AMs to emerge among the set of nodes in the network as the most powerful ones in terms of different factors, including processing capabilities and stability. Using a self-organizing and self-adapting distributed protocol, each node in the overlay selects an appropriate AM to report to so that sensed data is delivered error-free, and in a timely manner, while the load is distributed over the AMs.

Finally, we propose a utility function to monitor the quality of IPTV streams by predicting QoE based on statistical Quality of Service (QoS) information. The proposed function is simple and does not require high processing power. It allows the QoE of IPTV users to be monitored in real-time by the AMs, so that quality degradations are accurately identified and adaptation mechanisms are triggered at the right moment to correct issues causing degradations.

Theoretical analysis and simulations studies are presented to demonstrate the performance of the proposed schemes.

Dedications

To my mother Fatima, to my father Ahmed, to my wife Naouar, and last but not least, to my dear brothers Sami Kamal and Tarik Mounir.

Acknowledgements

I would like to offer my gratitude and humble acknowledgment to all the wonderful people that supported me throughout the course of my Ph.D. program, both at the academic and personal levels.

First of all, I would like to offer my profound thanks to my supervisor, Prof. Dr. Ahmed Karmouch, for his kind supervision, continued support, guidance and mentoring. Without his constructive advices and continuous motivation, it would not have been possible to complete this work. I would also like to thank my doctoral committee for their precious time, effort, and constructive feedback.

I would like to thank Cistech Limited for their support and the constructive feedback that helped improve the quality of this thesis.

I have had the opportunity to work with exceptional lab mates and am grateful for their encouragement and friendships. I would like to thank Ismaeel Al Ridhawi, Yousif Al Ridhawi, Heli Amarasinghe, Houda Rachidi, and all my colleagues at the Intelligence for Mobile Autonomic and Cognitive Networks Laboratory for their cooperation and support during my research.

My deepest thanks go to my parents for their love, support, and encouragements. I would also like to thank my wife for her consolation and moral support. Special thanks go to my brothers, my parents and brothers in-law, to all members of my family and to all my friends.

Contents

List of Tables

List of Figures

Acronyms

ACL	Autonomic Control Loop
ACS	Ambient Control Space
ADMA	Autonomous Decentralized Management Architecture
AF	Assured Forwarding
AI	Artificial Intelligence
AM	Autonomic Manager
AN	Ambient Network
ANEMA	Autonomic Network Management Architecture
ANSI	American National Standards Institute
AS	Autonomic System
ATIS/IIF	Alliance for Telecommunications Industry Solutions/IPTV Interoperability Forum
AutoI	Autonomic Internet
BE	Best Effort
BPD	Battery Powered Device
BPI	Base Performance Index
CAN	Content Addressable Network
CM	Context Manager
CMIS/CMIP	Common Management Information Service/Protocol
CORBA	Common Object Request Broker Architecture
CoS	Class of Service
CQ	Continual Queries
DE	Decision Element
DHT	Distributed Hash Table
DiffServ	Differentiated Services
DIS	Device Information Server
DNA	Distributed Network Agent

DRAMA	Dynamic Readdressing and Management for the Army
DSCP	DiffServ Code Point
DSL	Digital Subscriber Line
E2E	End-to-End
EF	Expedited Forwarding
EM	Energy Manager
EP	Energy Profiler
ESR	Efficient, Scalable and Robust
ETSI	European Telecommunications Standards Institute
FER	Frame Erasure Rate
FOCALE	Foundation, Observation, Comparison, Action and Learning Environment
FPS	First Person Shooter
FR	Full Reference
FTP	File Transfer Protocol
GANA	Generic Autonomic Network Architecture
GAP	Goal Achievement Point
GOP	Group of Pictures
HDTV	High Definition TV
IETF	Internet Engineering Task Force
I-frame	Intra-coded Frame
IMO	Information Management Overlay
INM	In-Network Management
IntServ	Integrated Services
IPTV	IP TeleVison
ISP	Internet Service Provider
ITU-T	International Telecommunication Unit-Telecommunication Standardization Sector
iTV	Interactive TV
KP	Knowledge Plane

MAPE	Monitor, Analyze, Plan Execute
MC	Media Client
MIB	Management Information Base
MOON	Manageable Open Overlay Network
MOS	Mean Opinion Score
MP	Media Port
MPLS	Multi-Protocol Label Switching
MS	Media Server
MSE	Mean Squared Error
MUI	Multimedia User Interface
NR	No Reference
OB	Overlay Builder
OCS	Overlay Control Space
OSI	Open Systems Interconnection
OSL	Overlay Support Layer
P2P	Peer to Peer
PAM	Potential Autonomic Manager
PEAQ	Perceptual Evaluation of Audio Quality
PESQ	Perceptual Evaluation of Speech Quality
PHB	Per-Hop-Behavior
PI	Performance Index
PM	Performance Manager
PolM	Policy Manager
PSNR	Peak-Signal-to-Noise-Ratio
QH	QoS Handler
QoE	Quality of Experience
QoS	Quality of Service
RF	Route Finder
RM	Resource Manager
RON	Resilient Overlay Network

RR	Reduced Reference
RSVP	Resource Reservation Protocol
SATO	Service-Aware Transport Overlays
SGMP	Simple Gateway Management Protocol
SHE	Super Head End
SLA	Service Level Agreement
SMART	Smart Media Routing and Transport
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SON	Service Overlay Network
SSIM	Structural SIMilarity Index
SSON	Service Specific Overlay Network
STB	Set-Top-Box
UFM	User Feedback Manager
VHO	Video Hub Office
VoD	Video on Demand
VoIP	Voice over IP
VQEG	Video Quality Experts Group
VQM	Video Quality Metric
VSO	Video Switching Office
WSDL	Web Service Description Language

Chapter 1

Introduction

1.1. Overview

The area of network communications has seen remarkable change in the last decade through different stages. The first stage was through the introduction of the Internet, which completely revolutionized the way network services are provided. The second stage arose from the development and deployment of cellular and wireless technologies, which enabled the supply of data and telephony services anytime, anywhere. Wireless cellular technologies have considerably changed users' behavior, allowing access to different data and multimedia services from anyplace. The third stage constitutes of upgrading the traditional telephony system into more sophisticated and high speed technologies such as DSL (Digital Subscriber Line). These technologies have enabled the emergence of Triple Play Services to homes (data, telephony and TV over IP). Currently, we are witnessing another stage which consists of the emergence of new customized and advanced services such as P2P-based applications and social networking.

Behind this ideal view of technology advancement, operators and service providers have to deal with many hidden issues. In fact, users have become more focused on the services themselves rather than the high-speed network. They no longer care about the technologies being used to deliver services, but became more concerned about the received quality and the cost. On the other hand, network architectures and protocols are of great importance to operators as they drive their ability to react to customers' needs. Increased customers' demands, resolving problems, and increasing capacity are examples of tasks that require operators to constantly upgrade their network equipment, integrate new services, components, and technologies without interrupting the operation of running services. These and other management tasks suffer from the increased equipment

complexity and heterogeneity, which requires more expertise and more efforts from highly skilled people to maintain the infrastructure. This traditional way of networks management has been used for a long time, relying on the use of primitive tools and experts to troubleshoot and configure networks and services. Its main drawbacks consist of the high cost associated with the need for highly trained administrators to identify problems and correct them, as well as the high amount of time it takes to perform these tasks. As a result, more sophisticated approaches relying on software agents [1], active agents [2] and policy-based systems [3] were proposed with the goal of reaching higher response times at lower costs by automating the management tasks. Nevertheless, network management systems could not cope with the continuous increase in size and complexity of network components and services. One of the weaknesses of these systems was their static nature that translates into their lack to adapt their decisions to specific situations and context. The introduction of overlay networks [4] to hide the heterogeneity and complexity of physical devices resulted in the emergence of new types of services customized to customers' needs. Overlays implement value-added functionalities and are constantly growing, pushing for an efficient management solution to offer services at the best quality and reduced cost.

An overlay network is a virtual network composed of a set of nodes connected via virtual links. They are built on top of different real networks. Nodes are connected via virtual or logical links that may be possibly mapped to many physical links in the underlying network. Overlay networks are flexible and able to offer new complex services that cannot be offered by current networks. They increase reliability and robustness and offer end-to-end QoS guarantees. They can be subdivided into two main categories: Peer-to-Peer (P2P) networks and Service Overlay Networks (SONs). P2P networks offer a mixture of features including redundancy, anonymity, scalability, permanence, trust and authentication. P2P overlay networks are famous for their applications of content search and retrieval. SONs, on the other hand, generically provide the necessary QoS for different multimedia applications including Voice over IP (VoIP), IPTV and online gaming. The notion of SONs was proposed as an effective means to solve some of the issues, mainly

QoS, to ease the creation and deployment of valued Internet services. End users directly pay the SON provider that provides end-to-end QoS support. SON providers buy resources, such as bandwidth, from infrastructure providers by means of service level agreements (SLAs) to build a logical end-to-end service delivery platform on top of existing transport networks. Recently, IEEE decided to create a new working group, IEEE NGSON WG [5], for the development of SONs. Network operators, service and content providers as well as end-users are intended to profit from NGSON. When the client requests a specific service, a path, called an SSON, is formed on the fly between the node providing that service and the client. Nodes that take part of delivering the service provide capabilities that enable them to modify it according to user requirements. SSONs allow a seamless integration of next-generation multimedia services within intelligent networks. The highly dynamic nature of SSONs, as well as their characteristics push for the need of more sophisticated management approaches to overcome the limitation of traditional management schemes. The co-existence of multiple SSONs could degrade their performance and that of the underlying network as well, if left unmanaged.

Autonomic computing principles [6] represent an emerging solution to ease networks and services management tasks by moving the management functionalities to the network itself without or with minimal human involvement. Autonomic systems were inspired by the nervous system of the human body. Their goal is to predict, diagnose, repair and avoid any faults and malfunctions that could take place in the functionalities of underlying networks without human intervention. An autonomic system makes decisions on its own based on high-level policies set by administrators. It constantly checks and optimizes its status and adapts itself to changing conditions. The use of autonomic computing principles for overlay networks management aims at overcoming challenges faced during the delivery of multimedia services using overlay networks, mainly the heterogeneity and highly dynamic nature of these networks.

In this thesis, we address the problem of SSONs self-management. This chapter presents an overview of different aspects of the problem and the motivation behind our

work. Then, it presents the contributions of the proposed work compared to other works in the literature. Finally, an outline of the organization of the thesis is given.

1.2. Service Specific Overlay Networks

SSONs represent a new trend in multimedia delivery over the Internet. They overcome the challenges of traditional multimedia delivery over the Internet by using overlay networks to create a virtual end-to-end path and provide a specific service with QoS guarantees. The notion of SON was first introduced in 2003 [7] as an effective means to address some issues of the current Internet, mainly end-to-end QoS, and to ease the creation and deployment of emerging QoS-sensitive services. A SON is mainly composed of service gateways that perform specific data and control functions. In [8], Schmid et al. define the concept of SSONs as part of the ambient networks project. SMART creates an SSON for each type of service or group of services. SMART is composed of three basic components: Media Ports (MPs), a Media Client (MC) and a Media Server (MS).

MSs constitute sources of media, i.e. media servers where actual multimedia files are stored. MCs are clients that request media files from MSs. MPs are overlay nodes that

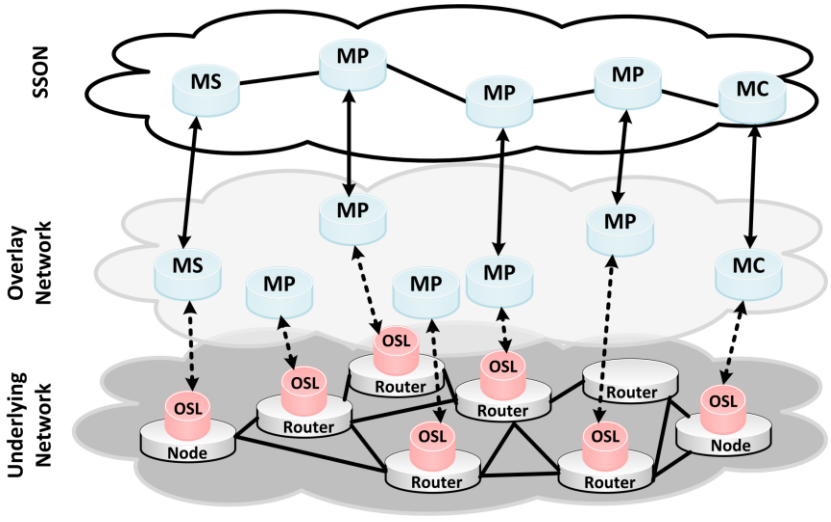


Figure 1: SMART and SSON Architecture

perform service specific functions. Examples of services that can be offered by an MP include caching, transcoding, translating, buffering etc... SMART enables services to be created on the fly and to be customized depending on users' needs and the current state of the network. It also allows the services to be adapted to changing network conditions and users' requirements. SMART represents a key architecture to overcome today's broad range of heterogeneous access networks, terminals, users, protocols, applications and services. SSONs can be used to stream the same multimedia file to different users with different preferences, such as audio languages. Instead of replicating the same video with different audio languages for each user, a MP offering a splitter service is used to split the multimedia file into video and audio streams. The audio stream is then passed on to MPs that offer translation services to translate it to the appropriate language and forward it to the right MC. In this example, the use of SSONs would significantly save bandwidth and processing power, and take the burden of media conversion and storage away from service providers and end-users.

SSONs are created for the sole purpose of offering a particular service and thus have a life cycle. Their operation requires cooperation and coordination among the overlay nodes to discover MPs and configure them appropriately. Their benefits come at the expense of added complexity and overhead. Added packet headers and redundant work at the overlay layer introduce additional overhead on the network. Moreover, overlays tend to overload the network due to the increase in traffic they transport. This leads to high resource consumption. Overlays are generally designed independently for different applications. For instance, a node can be part of two overlay networks: a content storage and retrieval network, and media transport. This causes them to affect each other negatively since each overlay has its own requirements, which could be contradictory in some cases. Overlays tend to compete in transporting traffic and end up creating bottlenecks and congestion as well as negatively affecting the underlying network performance. For these reasons, they need to be efficiently managed in order to reduce management cost and complexity while preserving operation at the best possible state.

1.3. SSONs Management Challenges

The management tasks of SSONs can be divided into two main categories. The first category consists of the actual overlay nodes present in the network, namely MSs, MPs, and MCs. The second category comprises the different services created and provided to users by means of SSONs. The creation of an SSON goes through different steps. First, a discovery process is needed to learn about the different MPs present in the network, the services they offer, and their QoS. A path is then formed between the MS and the MC, containing only those MPs that can honour the MC's request in terms of technical, QoS, and QoE requirements. Once the SSON has been established, proper monitoring of the service QoS and QoE is fundamental to ensure high customer satisfaction that represent the main source of revenue of service providers. The dynamic nature of overlay networks in terms of constantly changing resources, network conditions, nodes churn and movement, make traditional network management schemes useless as the gathered monitoring information becomes obsolete in a short period of time. For instance, overlay nodes may fail unexpectedly and links might experience congestion, which causes routing information to change over time. The co-existence of different SSONs and the numerous users and applications requirements also affect the choice of the overlay routing path. In addition, overlay nodes do not have a global view of the network infrastructure and the set of services running on the network. They are limited to knowledge acquired by each node which differs from one node to the other. The lack of control over the physical infrastructure, and the lack of knowledge of critical physical network information cause overlays to inefficiently use network resources. This is coupled with the indirect or even mis-communications between overlay and underlay networks which lead to inaccurate analysis such as mismatch between topologies and incorrect end-to-end probing results, generating redundancy and a large amount of overhead. Moreover, their high decentralization makes resource coordination hard to achieve. Finally, the open nature of overlays poses security and privacy issues, and makes resource sharing and collaboration

between end nodes challenging. These characteristics of overlay networks make their management using traditional schemes impractical.

Traditional network management schemes suffer from increased complexity, cost and response time. Complexity rises from heterogeneous devices making up the network and the fact that networks belong to different autonomous entities governed by different rules and policies. Network administrators need to be highly trained to understand management reports and know where changes should be made. This cost inefficient solution increases response time since experts will have to dig into logs with thousands of lines, identify faults and find out how to correct the problem. More sophisticated solutions tend to use new technologies (such as software agents) to automate the management process and reduce cost and response time. However, these solutions cannot cope with the continuous increase in size and complexity of network components and services, especially for the case of SSONs. These automated management solutions suffer greatly from their static nature which fails in the presence of today's highly dynamic networks. Users' mobility and continuous changes in networks' characteristics require more sophisticated solutions that can cope with these dynamics without putting more burdens on the network and the administrators. Several initiatives to push the research forward into innovative management solutions were recently launched. Although they use different names, they actually converge to the same direction: the creation of a new generation of intelligent independent equipment, networks and services with self-management capabilities. The most common terminologies found in the literature relate to the difference in the focus of these approaches and include Autonomic Communication, Autonomic Networks Management, Self-Managed Networks and Autonomic Networks. Their ultimate goal is to make network components manage themselves eliminating human intervention, or at least minimizing it.

1.4. Motivation

Existing research work has focused more on improving traditional management solutions by automating their operation. The ultimate goal was to ease the management tasks performed by administrators. Setting up measurement points in different parts of the network, remotely configuring devices, and filtering gathered information are basic examples of common tasks performed by administrators. A major limitation of existing frameworks stems from their built-in static configurations that are set by administrators into different devices of the network. They lack the flexibility required for constantly changing networks, as is the case for overlays. They may also not be adequate to handle changes in underlying physical networks. Another effect of the nature of static configuration is the resort of network administrators to estimate network traffic and user's requirements when setting up network devices. Such estimates might not be accurate and cause the network to run inefficiently, far from its optimal state. In addition, when service degradation happens, customers expect the QoS to be re-established in a timely manner. The long time needed for administrators to analyze logs and troubleshoot problems makes it hard to respond promptly to these degradations. Even adaptive management techniques, both at the network and application layers, lack flexibility to support applications with different requirements and fail to build upon past experiences from the impact of previously embraced strategies. Not to mention their dependence on decisions made by human administrators.

In addition, expert administrators have to deal with low-level management and configurations of the different devices in the network. They need to troubleshoot different components and read hundreds, if not thousands, of logs to identify the problem. They then have to manually configure the devices using basic command-line or GUI-based management interfaces. These complicated procedures and the lack of experts in the IT world make the management tasks tedious and time consuming for administrators which causes the cost of such management systems to increase. As for SSONs, the creation of a single SSON involves the discovery of appropriate services that offer the QoS set by the customer. A path is then established by selecting suitable nodes that will be used to deliver

the media files from the server to the end user. These nodes should be configured appropriately to meet the customer's QoS and be at his/her expectations. A SSON is created for each media session. This means that a number of SSONs will have to co-exist and compete for resources. If left un-managed, they might end up negatively affecting each other. This will not only degrade their performance but will also have an impact on the underlying network. Network reconfiguration happening as a response to customize a user's request can only be accomplished by a network operator. This task can take as long as a couple of days to be completed and result in excessive delays. QoS negotiation should not be limited to connection time as it limits the user's behavior to opt for a better service quality during the lifetime of the session.

As a consequence, system administrators and service providers are looking for IT systems that can manage themselves in an autonomic manner mimicking the behavior of the human nervous system. This idea has already been studied in the field of Artificial Intelligence (AI). Today's advanced technologies and innovations allow for new types of solutions to be proposed under the shade of operators' requirements that have become more precise than the general case AI considered in solving the problem. The first company to introduce the autonomic computing term with the goal of building a new generation of intelligent computer systems was IBM in 2001 [6]. Autonomic computing refers to the ability of components to self-manage based on high-level objectives. More specifically, the term autonomic was adopted from the field of neuroscience, the scientific study of the nervous system. The nervous system is divided into two subsystems. Organs under voluntary control (generally muscles) are controlled by the somatic nervous subsystem while the autonomic nervous system, which is not subject to voluntary control, manages individual organ function and homeostasis [9]. Under autonomic computing principles, system administrators and experts will not have to deal with low level management and configuration anymore but would instead focus on the high level management process.

An autonomic system proposes a self-regulation of an entity based on internal rules and policies and as a response to its external environment. Autonomic systems adapt their

decisions and learn from their behaviour and that of others. They also interact with other entities and adjust their operation as a result of the state of their environment. The use of autonomic computing principles in the management of SSONs has many advantages. The main goals consist of reducing the complexity and hiding the heterogeneity of devices with minimal user intervention. The abovementioned limitations of existing management solutions and the advantages of autonomic computing principles represent strong motivations towards the development of a strong autonomic management solution that enables systems to manage themselves with minimal involvement of humans.

1.5. Thesis Overview

The proposed research work tackles the issue of SSON management from an autonomic point of view. The ultimate goal is to incorporate the autonomic computing principles into the management tasks of SSONs to make the network manage itself efficiently without human intervention. Overlay nodes need to become aware of themselves and their environment to coordinate management tasks.

The proposed work is divided into a global autonomic architecture for autonomic management, a highly decentralized self-organization and self-adaptation algorithm to efficiently organize autonomic managers, and a QoE prediction scheme to monitor the QoE of IPTV sessions running in the network.

The proposed architecture defines the different components and their interaction to achieve system wide performance. The architecture becomes challenging in the case of dynamic networks in terms of resources, equipment and users.

All overlay nodes in the proposed architecture are potential AMs. An AM implements the autonomic MAPE (Monitor, Analyze, Plan, and Execute) loop as part of its autonomic operation. Monitoring consists of gathering data through sensors and organizing it in a proficient way for easy retrieval. The different types of overlay nodes have more important functions to perform besides the management tasks. Only a subset of AMs will

have enough resources to perform management tasks alongside serving customers and performing service specific functionalities. We propose an election scheme by which AMs emerge from within the set of overlay nodes as the most powerful, stable nodes in the network. Each AM will be responsible for managing one or more elements of the network. The problem of finding the optimal selection of elements a manager should manage is NP-hard. We provide a highly decentralized algorithm to solve this problem without incuring high overhead on the network, and that takes into consideration the dynamic nature of nodes in terms of mobility and churn.

Finally, we propose a utility-based approach to predict QoE of IPTV users running their services on an overlay network. Basically, we provide a QoE mapping to utility in order to model the quality of the IPTV session. This information is to be used to detect quality degradations when congestion or any other network failure happens, and to trigger adaptation procedures to bring the quality back to acceptable levels.

1.6. Summary of Contributions

The following presents a synopsis of the major contributions planned for dissertation:

- An autonomic architectural design for autonomic management of SSONs. The architecture presents the different components, their functionalities and interaction. It is designed to cope with the challenges stemming from the characteristics of SSONs, mainly increased complexity and high heterogeneity of network resources and equipment as well users' mobility.

- An AM election scheme and AM selection algorithm for the self-organization of AMs to efficiently manage network elements. The algorithm is based on a modified regret matching procedure for network elements to select an AM to manage it. The proposed algorithm copes with network's changing conditions

and provides a near-optimal selection that improves system wide performance in a highly distributed manner, and under limited overhead.

- A utility-based function for predicting QoE of IPTV over an overlay network. The utility is a function of QoS parameters (losses and delays) at the application layer as opposed to packet loss ratio and delay at the network layer that has been used in previous research. We demonstrate the effectiveness of our scheme to other schemes that use packet loss ratio and present our modified utility function. We also show the accuracy of the proposed function in predicting users' QoE based on statistical losses at the application layer.

1.7. Organization of the Thesis

The remainder of the thesis is organized as follows. Chapter 2 defines the subject and its scope. We briefly draw the lines around which we base our work. Chapter 3 presents the related works and discusses the previous approaches to autonomic networks management. We also identify the issues and drawbacks of previous frameworks and the different proposed solutions in the literature.

Chapter 4 outlines the architecture upon which we base SSONs management. The different components making up the architecture are presented along with their interactions. We describe the functionalities of each component in detail and show how it contributes to solve the management problem in an autonomic way.

Chapter 5 presents a novel AM election scheme and an algorithm for self-organization of SSON nodes to gather monitoring information in an efficient, adaptive and highly distributed manner. This introduces an autonomic behaviour of nodes to efficiently gather sensed information while incurring limited overhead on the network.

Chapter 6 presents a novel scheme for monitoring the QoE of an IPTV stream by predicting the quality and modeling it in terms of utility. Simulation results show the performance of our proposed scheme and its effectiveness.

Finally, Chapter 7 concludes the research thesis and provides future directions.

Chapter 2

Background

The internet was not initially designed to transport multimedia applications. Its creation was tied to the sole goal of allowing network administrators to communicate. The evolution of new services that require special treatment from the Internet other than transporting data changed the way researchers and industry professionals look at the Internet. Major changes had to be brought up to the Internet architecture in order for it to cope with the rapid changes and evolvement of services, mainly multimedia services.

This chapter presents an overview of the definitions and semantics used in this thesis. It will identify the context and scope of our work.

2.1. Introduction

Recent technological advances have pushed the production of highly complex and efficient equipment forward at lower prices. The Internet is nowadays transporting a variety of traffic that has different characteristics. We are witnessing a high and continuous rise of Internet services and the multimedia world. Devices with high processing capabilities made it easy and realistic to deliver audio and video in real time. Video conferencing and many other applications have now become a reality. Although it was designed as a best effort network, researchers and industrial companies made enormous progress in allowing the Internet to offer a great deal of quality sensitive services over its un-reliable infrastructure. A Service Oriented Architecture (SOA) has been recently proposed to support interoperability among different platforms and guarantee scalability. An essential element in SOAs is service composition, which led to the introduction of new complex services that were not otherwise available.

2.1.1. Service Composition

Service composition is the act of combining two or more basic services to develop new complex customized ones by discovering, integrating and executing them. The composition process is driven by customer's needs for higher level solutions at the lowest prices possible. The goal of service providers is to supply composite services from basic services at the server based on the technical and QoS requirements of users. Service composition can be done in design time or at runtime. The former has drawbacks especially when nodes are mobile and are constantly changing their configuration due to movement, nodes' failures or changes in network conditions. The latter approach can cope with changes in the network including QoS degradation and is able to replace component services if the requirements change.

Most of the time, the same service is offered by different providers. However, only one should be picked up by the system based on different criteria including the QoS required by the client. These requirements could change during the lifetime of a session, at which point a service selection adaptation is needed based on these changes. For these reasons, proper management of today's composite services have become more complicated and time consuming. The use of autonomic computing principles to ease management tasks and reduce cost is a promising solution for both service and infrastructure providers.

2.1.2. Autonomic Computing Principles

The incorporation of autonomic computing principles in computing systems was first introduced by IBM in 2001 [6]. This new vision towards an autonomic behaviour of computing systems has drawn much interest from researchers. The approach defines basic self-managing concepts to achieve a certain degree of self-governance. The evolution of the system management process has been carried out in different stages towards the final adoption of autonomic concepts and technologies. At first, operators had to monitor and manually configure each element during its lifetime. The introduction of system management technologies allowed system operators to monitor multiple elements at the

same time, using management consoles equipped with configuration interfaces. Systems based on analytical solutions were then developed to analyze gathered information, identify and predict potential problems and issues. The system operator is then given a set of appropriate solutions to choose from and deploy, in order to correct or avoid malfunctions. With the technological advances in the field of AI, researchers identified a need and potential for solutions that eliminate human intervention. Systems that were able to gather monitored data and predict situations were enhanced to react automatically to various situations without human intervention. This is established through a better understanding of the environment by making systems acquire the proper knowledge to identify the situation and make decisions at lower levels. The final stage consists of achieving the ultimate goal of autonomic management. The only human intervention at this point consists of human operators feeding the system high level policies and objectives. The system takes on the tasks of interpreting these high level goals and responding accordingly to achieve them. Human operators concentrate on identifying high level business goals and put their entire trust in the system to efficiently manage itself.

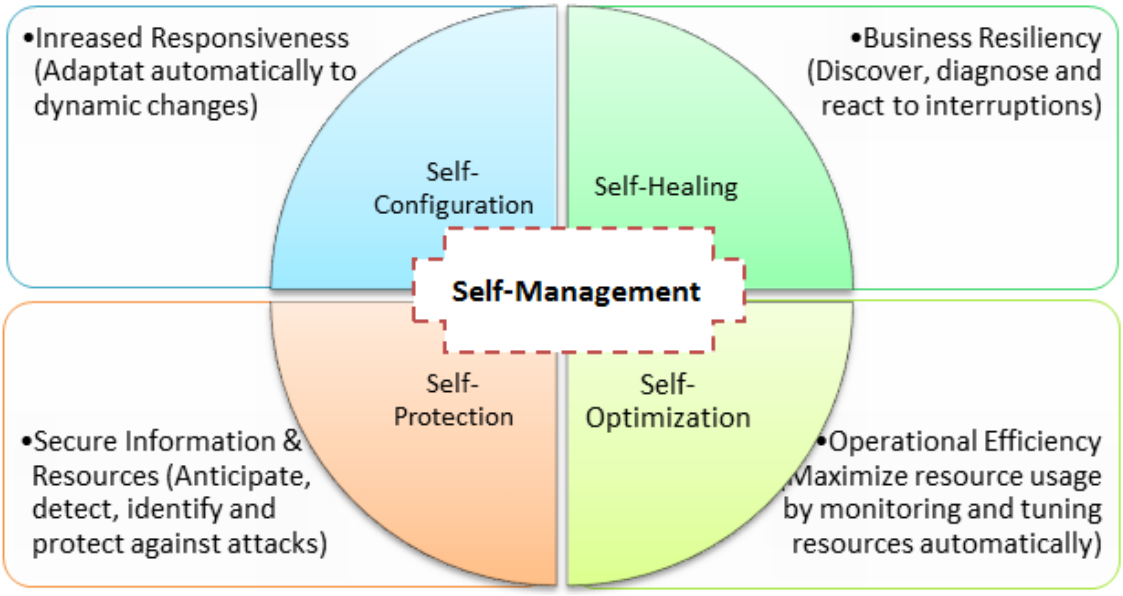


Figure 2: IBM Self-Management Aspects

IBM breaks self-management into four main aspects presented in Figure 2, referred to as self-CHOP: self-configuration, self-healing, self-optimization and self-protection.

A system implementing these concepts is said to be autonomic. Self-management is the essence of autonomic computing. It is intended to free human operators from the burden of knowing all the details of system operation and maintenance, and to provide a reliable system that runs at peak performance. Like the human nervous system, autonomic systems will maintain and adapt their operation in the presence of changing conditions and requirements including malicious or innocent hardware and software failures. Self-monitoring and self-adapting attributes make the autonomic system self- and environment-aware. They enable it to collect the proper knowledge and make the right decisions on its own. The self-CHOP properties that make up the four aspects of self-management are explained as follows:

- Self-Configuration allows new entities to be introduced into the system without service interruption and with minimal human intervention. The configuration process takes place under well-defined high level objectives and policies set by administrators. When a new entity joins the system, it shall do so seamlessly by making its presence and functionality public to other components that will reconfigure themselves if needed.

- Self-Healing provides elements with the necessary functionalities to detect, diagnose and repair any problems resulting from bugs, or failures in software or hardware. Using the acquired knowledge, an analyzer would examine different logs and data to identify the root cause and match it to a solution that will bring the system to a functional state.

- Self-Optimization requires elements to continuously look for various ways to improve their operation by identifying opportunities to increase their efficiency at lower costs. The ultimate goal in self-optimization is to efficiently maximize

resources utilization to meet the needs of end-users while reducing the cost of operation.

- Self-Protection enables autonomic systems to anticipate attacks and protect against them based on early reports from sensors. Self-protection defends the system against large-scale problems arising from malicious attacks or cascading failures that could not be corrected using self-healing methods.

An autonomic system is composed of multiple interactive autonomic elements. An autonomic element consists of an AM that controls and manages one or more (managed) elements. The interaction between the AM and the managed element(s) is achieved using sensors and effectors. Sensors gather information from the managed elements and send them to the AM for further processing and context retrieval. They can also be used to send asynchronous messages such as notifications when certain events happen. Effectors are used to configure the managed element(s) and change the different aspects of their states to control their behaviour. AMs run a control loop that implements the lifecycle for managed element(s) and autonomically adapts their state to achieve the high level goals set by system operators. This loop is referred to as MAPE: Monitor, Analyze, Plan, Execute and is illustrated in Figure 3.

The monitor function allows the AM to collect, aggregate, filter and report details about its managed element(s). The analyze function permits the analysis of collected information by the AM in order to understand the state of the managed element(s). It requires the use of complex models and algorithms to effectively build a global understanding of the state of the elements. Once the situation has been determined, the AM must identify and plan a set of actions necessary to accomplish the high level objectives. Finally, the execute function allows the AM to change the actions of the managed element(s) with the aid of effectors. Because autonomic systems manage themselves, they are inherently distributed. The knowledge base sits at the center of the MAPE loop, and

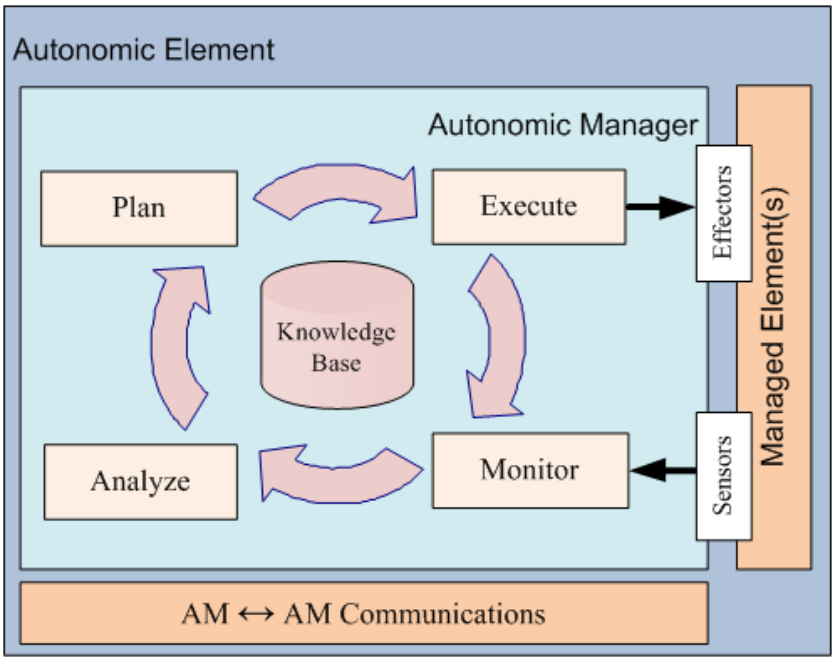


Figure 3: Autonomic Element Architecture

contains information and statistics gathered by different sensors. It holds data that has been processed and converted to knowledge that will be analyzed by the analysis function.

The autonomic computing initiative opened different perspectives for the design of new systems capable of managing themselves in different areas. Just as autonomic computing involves the self-management of computing systems, autonomic networking [10] was proposed to study new protocols and network architectures and develop networks able to function with a certain degree of freedom, and to adapt efficiently to changing network conditions. Under today's highly complex and heterogeneous networks and devices, autonomic networking represents a key solution towards efficient networks and services management. The emergence of service composition techniques and overlay networks played a role in the evolution of highly sophisticated services that are user centric but demand special handling by networks. Such services include VoIP, video conferencing, online gaming, IPTV and other multimedia applications that are sensitive to congestion, delays and errors.

2.2. Multimedia Services in the Internet

The incorporation of multimedia and multimedia communication by educational and industrial sectors in design, management, training and learning environments mark a major turning point. This important and continuously expanding area involves different technologies, including compression, computer networks, and the delivery of multimedia over these networks. New standards and technologies for multimedia and multimedia communication are being developed every year and it has become challenging to cope with the pace of these rapidly evolving technologies. The amount of information involved in the transmission of audio and video is significant. Multimedia information must be first compressed before it can be transported through different networks in order to reduce communication delay and the amount of bandwidth consumed. To ensure a reasonable quality at the receiving end, several restrictions, such as delay and jitter, have been identified. Therefore, communication networks are being enhanced to support multimedia communication capabilities. They are being constantly improved to provide a superhighway for smooth multimedia information transport and delivery. Different QoS management schemes have been proposed in the literature to enable best effort networks to deliver services at the required quality mainly by introducing some kind of priority between different types of traffic.

2.2.1. Traditional Technologies for Multimedia QoS Management

Multimedia data, as opposed to generic computer data, is characterized by particular properties and necessitates special requirements to be met by the delivery system, especially real-time data. Matters get more complicated when the intended audience is heterogeneous in terms of receivers and access technologies. For example, IPTV services must be delivered according to certain QoS guarantees, to a set of users using devices with different screen sizes, compression schemes and connected through heterogeneous access networks. Hence, multimedia services are often required to be adapted and customized according to users' needs. The traditional way to deal with this is through an end-to-end

basis using different methods at the server side. Under this architecture, content is modified at the server side and transported over communication networks. The sole responsibility of these networks is delivering packets from the source to the destination under specific constraints, to ensure an acceptable service quality. This creates bottlenecks at the servers and poses scalability issues. Another method is to move the processing functions to the end user's device. However, due to device limitations in terms of processing power, memory and battery, this method was quickly discarded. Another solution that we will discuss later involves moving the processing functions to the network itself by defining different network-side functionalities to support multimedia delivery. Regardless of which method is used, the main goal of multimedia delivery systems is to respect the QoS requirements of the applications and efficiently manage network resources.

Due to the high volume of multimedia data and the best-effort service offered by the Internet, communication networks encountered several challenges trying to respect the QoS requirements imposed by different applications. Several QoS management models were proposed in the literature to support QoS over the Internet. The most established ones that were defined by international organizations include the OSI QoS Framework [11], the International Telecommunication Unit-Telecommunication Standardization Sector (ITU-T) model [12], Integrated Services (IntServ) model [13] and Resource Reservation Protocol (RSVP) [14], Differentiated Services (DiffServ) model [15], and Multi-protocol Label Switching (MPLS) [16].

Many QoS solutions that were proposed in the literature failed to be deployed mainly because of the autonomous nature of different networks making up the Internet. Autonomous systems are governed by different policies that make them hard to manage. Moreover, achieving an end-to-end QoS guarantee for a certain flow became hard to accomplish in the presence of high heterogeneous and complex devices. In an attempt to hide this complexity and heterogeneity, and to support QoS in the Internet on an end-to-end basis, overlay networks have been proposed in the literature.

2.2.2. Multimedia Delivery over Overlay Networks

Overlay networks are virtual networks that are built on top of physical networks using logical links and connections. They allow the creation of new services that are not available in existing networks. For example, they can be used to increase security and robustness and to provide new services for mobile users. They can be incrementally deployed without involving ISPs and they do not need any equipment set up or infrastructure changes [4] [17]. Overlay networks can be divided into two main sub-categories: P2P networks and SONs. P2P networks offer a mixture of features including redundancy, anonymity, scalability, permanence, trust and authentication. SONs, on the other side, are created for the sole purpose of providing QoS support to a specific service. P2P networks are also referred to as application specific overlay networks whereas SONs are generic overlay networks, in the sense that they are able to support different types of applications.

Examples of application specific overlay networks include multicasting [18] [19], content distribution networks [20], and P2P file sharing [21]. Resilient Overlay Networks (RON) [4] allows distributed Internet applications to detect and recover from path outages and periods of degraded performance. However, RON suffers from scalability issues and does not scale for more than 50 nodes. Many application layer multicasting schemes work on the problem of placing nodes strategically to create a multicasting service at the overlay. Overcast [17] is an application-level multicasting system that provides a scalable and reliable single-source multicast. It builds efficient data distribution trees that adapt to the network changing conditions while utilizing the network bandwidth efficiently. Other works that exploit the use of overlays to build efficient multicast trees for multimedia streaming include [22], [23] and [24]. Overlay networks have also been proposed for content search and retrieval, including multimedia content delivery. For instance, [25] presents a framework for provisioning enhanced network-based multimedia distribution services to heterogeneous receivers. The heterogeneity lies in clients' processing capabilities and characteristics including network conditions and media representation of

the content. An overlay network is formed between sources and destinations by relaying multiple cooperating content-aware intermediaries to distribute the media. The overlay network provides efficient content-based programmability for the provisioning, routing and management of media streams. Another framework that provides multimedia distribution service based on P2P networks was proposed in [26]. The framework uses P2P techniques to self-organize hosts with the same network conditions in the same groups, in order to build a topology-aware overlay. Two distributed heuristic replication schemes based on the topology-aware overlay were also proposed to improve media delivery quality and provide service availability. SDNet [27] is a distributed storage-assisted data-driven overlay network that provides VCR-like functionality in P2P environments. SDNet is based on the integration of two networks: DONet, an enhanced data-driven overlay network used for the routine video distribution, and a multi-way tree to support efficient Video on Demand (VoD) operations. MarconiNet [28] is an overlay network based on IETF protocols to help provide signalling, session announcement, session description, stream control, media delivery, and feedback control for content distribution using application-layer techniques. It helps build a flexible mobile content distribution overlay network that supports multi-layered security and payment. Other generic content-delivery networks based on the use of overlays were proposed in the literature and have gained much interest due to their fast search and retrieval characteristics using Distributed Hash Tables (DHTs). Content Addressable Network (CAN) [29], Chord [21], Pastry [30], and Viceroy [31] self-organize based on participating nodes' IP addresses, or use the node's stored data as the organizing content, to provide fast content search and retrieval techniques and algorithms.

As opposed to the abovementioned overlay networks that were designed for a single specific application, SONs provide generic service types networks where services are created based on users' requirements and needs. SONs bring more advantages to multimedia delivery and support, by adapting the delivery functions to network load and conditions, and supporting user mobility and preferences.

Smart Media Routing and Transport (SMART) [32] framework has been developed to achieve these goals in the context of Ambient Networks (ANs). SMART aims at enabling network customization for services running over legacy networks. It introduces the concept of network-side functions or media processors that enhance media services by providing different services including caching, adaptation, security etc.... The development of a smart multimedia delivery framework enables a transparent incorporation of network-side functions in the end-to-end path. It also uses the advantages of value-added services inherent to ANs such as mobility management, the use of context information and QoS support. Finally, it allows flow routing and prompt adaptation of media delivery based on network, user and service context and policies.

SMART targets a highly heterogeneous and mobile environment where different networks, terminals, network interfaces, users, signalling/transport protocols, applications, and services will co-exist and cooperate to provide customized high-level solutions and services. As a result, some streams may need to be cached, adapted, transcoded, split, translated, secured or transformed in a certain way, based on some requirements, before they can be delivered to the end user. SMART architecture [8] allows a transparent integration of next-generation multimedia services into ANs.

2.2.2.1 SMART Acrchitecture

SSONs represent a key concept in the SMART architecture. A SSON is a virtual overlay network that is created for every media delivery request. The novelty of this approach lies in the fact that appropriate overlay-level routing paths that meet specific requirements can be configured. Also, network-side media processing functions can be transparently integrated into end-to-end delivery paths to allow services to be customized.

The separation of media flows through the creation of independent SSONs allows fine-grained adaptation and added flexibility to be provided for each media stream. The use of overlays for each service introduces a new level of abstraction that allows the establishment of a generic solution, completely independent of the underlying

infrastructure, and more capable to be deployed in today's highly heterogeneous environments. Overlays also add the flexibility of defining routing limitations which result in a generic, easily extensible, framework for context-aware routing of media.

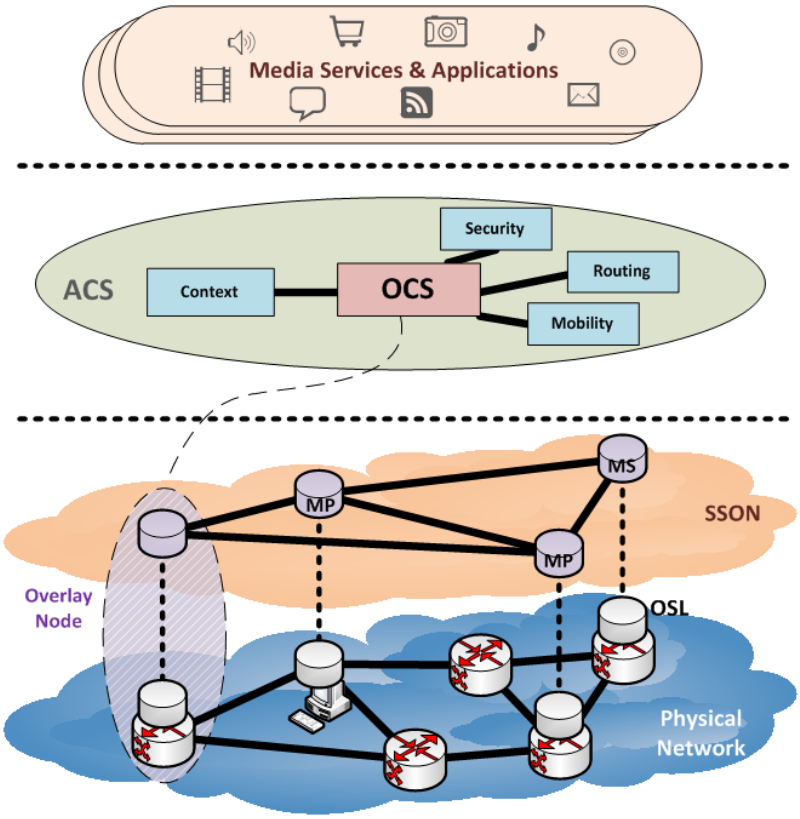


Figure 4: SMART Architecture [8]

Figure 4 illustrates the SMART architecture in the context of ambient networks. SMART functionalities require the optimal selection of overlay nodes (called ONodes in SMART terminology) that will be used for the creation of an SSON, and their configuration, including possible adaptations of media and the overlay network topology. These functions are based on services' and users' requirements and preferences, and hence are controlling actions that can be grouped in a control plane. The transport and handling of

data is performed at the user plane where actual media is transformed and adapted as instructed by the media routing.

The Overlay Control Space (OCS) constitutes the main control entity in SMART. It makes up the media delivery functional area of the Ambient Control Space (ACS). The OCS represents the decision point for all SSONs. It is responsible of the creation, management (including adaptation and re-configuration functions), and termination of SSONs based on users' requests and requirements. The main management tasks performed by the OCS are primarily related to the control of addressing and routing mechanisms at the overlay layer. Hence, the decisions made by OCS involve appropriate configuration of routing tables that will choose which ONodes are going to be used for each SSON and what processing capabilities will each ONode perform as part of the SSON establishment.

On the other side, the user plane consists of four main entities: Overlay Support Layer (OSL), media clients, media ports and media servers. The OSL is implemented in every ONode and realizes basic overlay functionalities needed for the transport and handling of data. It resides on top of the underlying network and builds virtual overlay links to all other OSL entities. The sole purpose of the OSL is the transport and processing of packets on the overlay. Application modules that implement the behaviour of a MC, MP or MS reside on top of the OSL and help in the procedure of media routing. MCs represent clients that receive media data and pass it on to the appropriate application. MPs are media ports that perform specific functions to customize and adapt media according to network conditions and user's requirements. MSs are data sources where media is stored. The same ONode can perform different roles in an SSON by activating different modules. For example, an ONode can be a MS and a MC at the same time in the context of P2P applications.

2.2.2.2 Media Processing Functions

The definition of services in the context of SMART is generic, ranging from simple requests for information such as web browsing, to more complex services including

mobility, content adaptation and media splitting features. The forming of these services might require numerous stages of transformation and adaptation of media before it is delivered to the end point. An example of such transformations is the encoding of a video to fit into the user's device screen. The idea of moving transformation and adaptation functionalities to the network side provides a novel solution to overcome the drawbacks of server-side and client-side adaptation. It allows the network to achieve the best QoS possible by choosing optimal MPs, configuring them appropriately and constantly adapting their functions. Additional intelligence can be embedded into the network using SMART functionalities. Examples of such intelligence include media splitting, synchronization, smart caching, smart media routing and adaptation to meet users' needs and requirements. Media processing functions are offered by MPs present in the network. Due to constantly changing network conditions, and the unstable nature of overlay networks in terms of nodes joining and leaving, SSONs must be carefully managed in order to ensure that network resources are optimized, and users get services at the best possible quality.

2.2.2.3 SSONs

An SSON is defined through the set of MPs making up the path between the MS and the MC to deliver a service, in addition to the virtual links connecting them. An SSON is created for every media delivery service or group of services. This setting allows them to be independently configured to meet the exact requirements of service providers and users. The SSON lifecycle involves initial service requirements negotiation. An SSON request is then sent to the network to setup a path from the MS to the MC. If needed, MPs are discovered and then selected appropriately according to the requested service. Overlay routing tables need to be configured to optimize the overlay path. MPs are also configured to perform service specific functions. During the delivery process, an SSON might need to be adapted according to network conditions and changes in the requirements. Fast adaptation is required in cases where nodes or links fail unexpectedly. On the other hand, slow adaptation might take place when the requirements change or nodes are added or

moved in the network. When the media delivery is over, SSONs must be torn down to release resources.

SSONs are an effective mechanism to support end-to-end QoS guarantees for the delivery of multimedia content. Real-time applications such as online gaming, VoIP and video streaming can be easily supported on heterogeneous devices using SMART architecture. However, creating an SSON and setting up MPs to deliver the media under certain QoS constraints does not guarantee that the quality of the service will not be affected during the lifetime of the session. The end-to-end QoS must be continuously monitored to trigger adaptations when it drops below acceptable values. This is very important to service providers because it allows them to keep their customers happy, make them loyal and attract new ones. Video applications, and more specifically IPTV, are very sensitive to quality degradations and require careful monitoring and management to ensure that customers receive the best quality possible and avoid customer churn. They have strict requirements that must be met by the network and maintained throughout the multimedia session to ensure that the required QoS and QoE of users are preserved.

2.3. IPTV

The IPTV system allows television services to be delivered over the IP protocol suite, instead of being delivered through traditional networks (terrestrial, satellite and cable). There are many advantages to IPTV over traditional linear-programming TV but the most important one is its ability to address users individually. IPTV services may be classified into three main categories: live TV constitutes the current traditional way people view TV, time-shifted TV allows users to replay a TV show that was broadcast hours or days ago, and VoD provides users with a catalogue of videos to browse and choose one to watch. IPTV is still on-going standardization efforts from different bodies including ITU, IETF, European Telecommunications Standards Institute (ETSI) and Alliance for Telecommunications Industry Solutions/IPTV Interoperability Forum (ATIS/IIF).

2.3.1. Definition

Many different definitions for IPTV exist in the literature. The ITU focus group on IPTV defines it as “multimedia services such as television/video/audio/text/graphics/data delivered over IP-based networks managed to support the required level of QoS/QoE, security, interactivity and reliability” [33]. Another definition that is given by ATSI describes IPTV as “the secure and reliable delivery to subscribers of entertainment video and related services. These services may include, for example, Live TV, VOD and Interactive TV (iTV). These services are delivered across an access agnostic, packet switched network that employs the IP protocol to transport the audio, video and control signals. In contrast to video over the public Internet, with IPTV deployments, network security and performance are tightly managed to ensure a superior entertainment experience, resulting in a compelling business environment for content providers, advertisers and customers alike” [34].

IPTV can be broadly defined as the means of delivering enhanced video applications to a television through a broadband connection that uses an IP transport network. The sheer number of possible IPTV applications and usage scenarios explains the interest and investments that have been put into it by different industrial and research bodies.

2.3.2. Applications of IPTV

The use of IPTV enables differentiated products and services to be offered to individuals. It provides a cost saving solution in terms of equipments and system deployment and implementation. Its flexibility to offer customized services makes it an important source of revenue to content and service providers. IPTV applications are numerous; we will cite only a few promising and interesting ones.

- Targeted advertising allows a type of ad to be placed so as to reach consumers based on different traits (such as demographics, age or observed behaviour), as

opposed to the traditional advertising method utilized in traditional TV systems, where the same ad is broadcasted to all viewers.

- Private video recorder allows users to record requesting programs and provide them when the program ends.
- Time Shifted TV delays the broadcast of TV programs to a later time or day. IPTV-commerce enables the delivery of e-commerce services on a TV.
- On-screen caller ID is another interactive IPTV application that merges video content and telephony services.

The success of IPTV is mainly due to the numerous applications available for a variety of different devices as well as its successful integration with existing networks and solutions. IPTV systems are made up of different components, ranging from content servers where multimedia content is stored into different end user devices that play the content.

2.3.3. IPTV Architectures

The global IPTV architecture identifies four main domains: the consumer domain represents end users that select and consume content and pay the bills; the network provider domain connects the consumer domain to the service provider domain; this latter is responsible for providing services to consumers; finally, the content provider domain owns content or licenses to distribute it. IPTV was initially implemented as digital TV over terrestrial and satellite networks. The growth of networking and storage technologies, coupled with advancements in coding and the high increase in bandwidth, enabled the transport of TV over a wide variety of fixed and mobile networks.

A typical IPTV architecture is comprised of the following main components:

- Super Head End (SHE): SHE hosts live video feeds and real-time encoding of video broadcasts as well as VOD content. It also includes back-end systems

such as the subscriber database. A video operator usually has one or two SHEs residing in the core of the transport network.

- Video Hub Office (VHO): VHOs contain a mix of devices including VOD servers, network routers that connect the distribution network to the network core and encoders for local TV stations. VHOs are located in metropolitan areas and serve between 100,000 to 500,000 homes. An operator typically maintains a few dozen regional VHOs.

- Video Switching Office (VSO): VSOs host routers that aggregate and multiplex video service with other services such as VoIP and data. A typical VSO would serve a neighbourhood in a dense area or a small city.

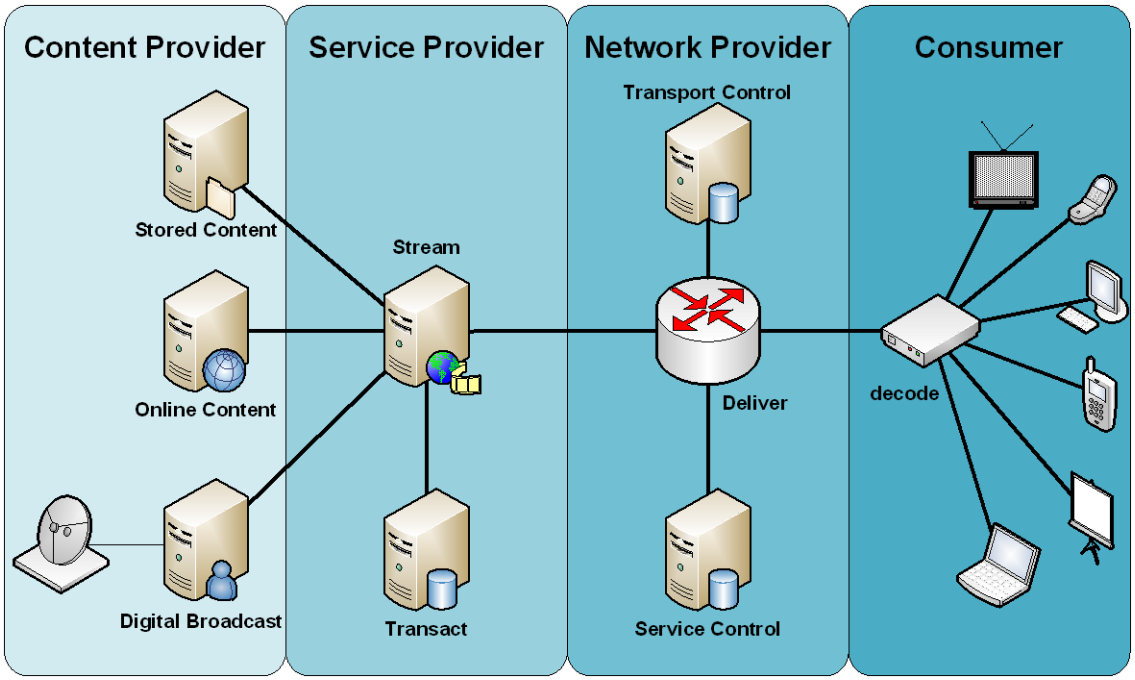


Figure 5: IPTV Components [143]

2.3.4. IPTV Requirements

Video services offered by IPTV place extra demands on the network. With the introduction and deployment of High Definition TV (HDTV) and other forms of video standards, IPTV users are expecting a high level of video quality in terms of higher resolution, aspect ratio and frame rate. This means that a home with IPTV service requires a significantly higher amount of bandwidth than one with data services only. The added bandwidth is mainly due to the fact that video is delivered through constant streams to the end user's set-top-box (STB). For instance, the MPEG2 compression standard consumes around 3.75 Mbps while the new MPEG4 standard consumes only 2 Mbps both providing the same high-quality image. HDTV, on the other hand, consumes between 6 Mbps to 15 Mbps depending on the encoding rate. Additionally, VOD implies the use of unicast, per-user channels that increase bandwidth consumption in the network as opposed to broadcast TV channels that are delivered using IP multicast. The amount of users a network can handle is limited by its resources.

Successful services lead to a rapid increase in the number of subscribers. Admission control is essential to IPTV services in order to ensure no packets are dropped and the

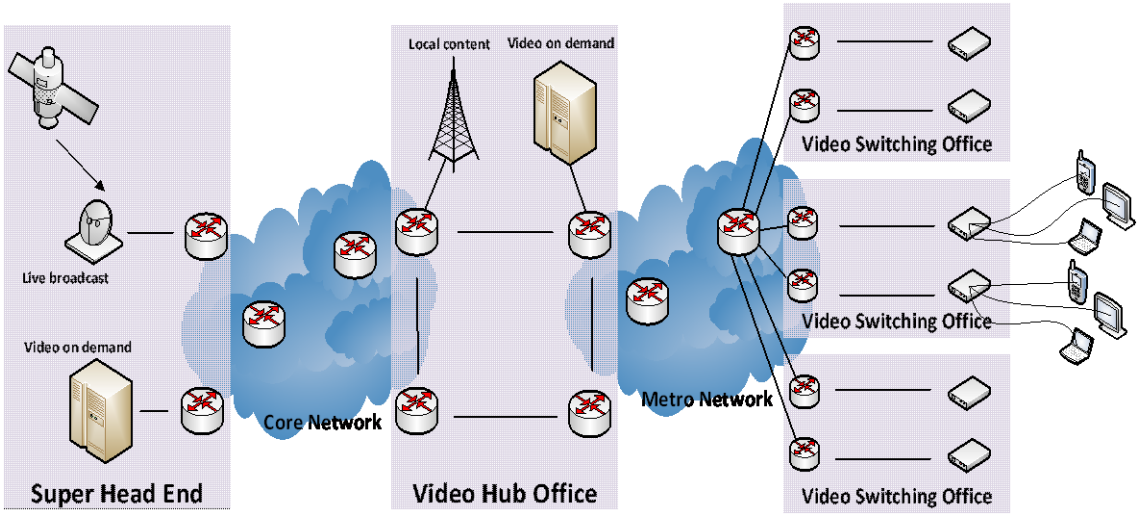


Figure 6: Basic IPTV Network Architecture [144]

experience of users is as expected. The network must implement mechanisms that grant permission to new video sessions only if they will not cause the quality of running sessions to deteriorate. Channel zap time or channel change time is the time it takes the system to terminate a channel and start playing another one. Although subscribers will not use channel zap time as the only metric to purchase TV service, it greatly affects their satisfaction with the service. The IPTV system should minimize channel latency as much as possible to ensure satisfied customers and good service.

In addition, service availability represents one of the most important requirements of IPTV. Broadcast IPTV consists of multicasting video streams to different users interested in viewing a certain channel. When a multicast source is lost, hundreds to thousands of users could be affected. IP multicast must be optimized to minimize bandwidth usage and service latency. Transparent service restoration is also needed in the network to minimize service down time. IP multicast redundancy is a good alternative to increase availability because it allows easy switching from one source to the other. On the other hand, VOD is user-centric and hence the loss of a stream does not affect other users. However, proper management of VOD streams is important in order to avoid random packet losses due to over subscriptions. It is therefore necessary to use different QoS schemes for VOD and broadcast IPTV to increase service availability.

During the lifecycle of a service like video, the bandwidth that the service will use depends on different parameters such as the density of the area, the overall time the service will be offered and so on. The service lifecycle should be supported by the network in order to accommodate more users without considerable upgrades. The network must be able to scale easily to any rapid growth of subscribers, and since the service could change over time, the network must be flexible to address these changes without any effects on provisioning. For instance, additional bandwidth should be provided easily when needed, as is the case of services and channels that become popular over a period of time.

QoS is very crucial when assessing IPTV because of its impact on the quality of video streams. Packet loss constitutes one of the metrics that greatly affect IPTV quality. Although the loss of a negligible number of packets might not drastically affect a viewer's experience, if the event occurs too often or lasts for a long period of time, it will degrade image quality and impact the user's QoE. Some STBs are equipped with error correction algorithms that conceal missing information or simply ask for it to be retransmitted. However, the limited processing power of STBs and other portable devices makes it hard to implement highly efficient algorithms that can cope with errors in IPTV streams. In general, STBs have very limited resources. This constitutes a challenge to handle degradations of QoS parameters. Jitter for instance is an important factor that must be considered, because different STBs have different buffer sizes ranging from 150ms to 5s. It is also essential to keep the end-to-end delay bound to a minimum value in order to avoid degradations in the QoE. Finally, it is important to consider the co-existence of other types of traffic that will be transported alongside IPTV. Providing a good IPTV experience should not happen at the expense of degrading the quality of other services such as VoIP. When multiple services run on the same network, reliable scheduling and congestion avoidance methods should be employed to keep the different services running at acceptable QoS and offer customers an acceptable level of QoE.

2.4. QoS and QoE Requirements for Multimedia Services

Multimedia services have different QoS and QoE requirements. The introduction of QoS architectures like IntServ and DiffServ was supposed to improve the best-effort service provided by the Internet. The notion of QoS was initially aiming at the level of satisfaction that a user gets from using a service. It then became a more technical issue, focusing on monitoring and improving network key performance parameters such as delay, jitter and packet loss. However, end users usually do not react to technical performance. What matters for them is the experience they are able to obtain. Based on this insight, we have noticed an important move concerning service quality. A new sense of interpreting end-to-end quality has emerged, with the aim of putting the end user at the end of the

communication chain. The notion of QoE has appeared to describe quality as perceived by the end user instead of as calculated by the network in terms of technical parameters.

2.4.1. QoS vs. QoE: Definition

QoS is defined by ITU-T as "the collective effect of service performance which determines the degree of satisfaction of a user of the service" [35]. It is very important to support QoS nowadays, as it defines the requirements that should be met when a service is provided. This is because users are not interested in how a particular service is provided but in the quality level they perceive. QoS refers to the ability of communication systems to provide a better service to selected users by means of prioritizing their traffic over heterogeneous networks. It is essential to ensure that this prioritization does not negatively affect other users by degrading their quality. Intuitively, QoS represents the set of technical parameters to be guaranteed to the users, namely bandwidth, delay, jitter and packet loss rate.

On the other hand, QoE is defined by ITU-T as "the overall acceptability of an application or service, as perceived subjectively by the end-user" [36]. QoE is a subjective measure of a customer's experience. It is a concept comprising all the elements of a user's perception of the network, its performance and how both meet the user's expectations. QoE is determined from several interacting factors; these include cost, reliability, usability, utility and fidelity [37]. Both QoS and QoE measure how well the system meets the expectations of the user, but QoS differs from QoE in the fact that it focuses on measuring performance from a network perspective. For example, QoE looks at user-perceived degradations such as voice or video artefacts, whereas QoS focuses at network related effects such as delay or jitter. QoE is directly related to QoS but the challenge for a service provider is to accurately map QoS at the network level to the QoE at the user level, and be able to control it. It is important to note that measurements taking place in individual nodes might show acceptable QoS, but the end user may be experiencing unacceptable QoE. For instance, QoS measured at the network layer might not always map to the correct QoE.

Although the network might report the loss of a small (yet acceptable) number of packets belonging to a video frame, the implications could be worst if other frames depend on the corrupted one for decoding. In cases like this, QoS fails to accurately map the right level of QoE users are experiencing.

QoS requirements for multimedia applications have been the focus of study of different standardization groups. Recommendations Y.1541 [38], F.700 [39] and G.1010 [40] represent major findings by the ITU.

2.4.2. QoS Requirements for Multimedia Services

ITU classifies applications into eight groups according to their tolerance to the amount of delay and errors. Error tolerant applications include conversational voice and video, voice and video messaging, streaming audio and video, and fax. Error intolerant applications include command/control, transactions based applications, messaging, downloading (FTP), and background applications (web-browsing, email…). ITU identifies applications within four main categories based on their delay tolerance. Interactive applications do not tolerate delays beyond one second and include conversational video and audio. Responsive applications have delays that do not exceed two seconds such as web browsing and e-commerce. Timely applications require delays less than ten seconds while non-critical ones can tolerate longer delays. Conversational services [40] [41] is the only class of services with characteristics that are determined by the perception of users. Overall, the transfer time should be low and the temporal relation between information must be kept. The second class of multimedia services, the QoS expectations of users for Interactive services [41], involves a request-response model of the end user. The destination expects an answer back in reasonable times. Round trip propagation delay and low information loss are key characteristics of this class. Streaming services [41] is a unidirectional class with high continuous utilization meaning there are few idle or silent periods. It is characterized by having no firm bound for delay and delay variations.

QoS has been used to objectively measure the service delivered by the vendor, where measurements are usually not related to the user but to the media. Because of that, QoS fails to capture the user experience and how satisfied he/she is with the service. The shift from QoS to QoE put the focus more on the users rather than on the media being delivered. Although QoE is comprised of different factors, some of which are hard to quantify, many methods based on QoS parameters have been proposed. In the next section, we will briefly outline the most important ones.

2.4.3. QoE Requirements for Multimedia Services

The parameters that define QoE are usually service dependent. For instance, zapping time is relevant for TV services whereas in voice or gaming, it does not make any sense. QoE is affected by three broad parameters. First is the quality of the video or audio stream at the source, which depends on the type of codec used. Second is the QoS which encompasses the delivery of content over heterogeneous networks. Third is the perception of the user, i.e. his expectations of the service. While the first two categories are easy to capture, human perception is not. This is why it is usually quantified by ways of Mean Opinion Score (MOS) [42], which represents the result of assessment of some test panel. MOS covers a scale from 1 to 5, where 1=bad, 2=poor, 3=fair, 4=good and 5=excellent. A MOS value of 3.5 represents the minimum acceptable quality. In general, QoE can be measured using three possible methodologies:

- No Reference (NR): this model does not need to use the original stream or source file to predict QoE. It simply monitors several QoS parameters in real-time and use them to calculate a QoE value.

- Reduced Reference (RR): this model uses limited knowledge of the original stream and combines it with real time measurements to predict QoE.

- Full Reference (FR): this model uses the original reference stream and compares it with the received stream to calculate the corresponding QoE value.

The E-model (ITU-T Rec. G.107) is one of the most important examples of a NR QoE model for voice conversation. It uses end-device characteristics and transport parameters to predict QoE. An example for a FR model is the Perceptual Evaluation of Speech Quality (PESQ) model (ITU-T Rec. P.862). It uses both the original and the received streams and compares them to determine which differences result in annoying artifacts. The Perceptual Evaluation of Audio Quality (PEAQ) model (ITU-R BS.1387) represents the standards for measuring audio QoE, as opposed to PESQ which cares for speech quality in particular. PEAQ is a FR model that analyzes the audio signal sample by sample after temporarily aligning both the source and received signals. The G-model [43] was developed for predicting QoE of First Person Shooter (FPS) games just like the E-model was developed for speech. Wattimena et al. [43] performed subjective tests with famous FPS games such as Unreal Tournament, Counter Strike and Quake III and IV. The G-model uses network measurements of delay and jitter to calculate QoE as a MOS value.

On the other hand, the most commonly used QoE metrics for video include Peak-Signal-to-Noise-Ratio (PSNR), Video Quality Metric (VQM) [44] and Structural Similarity Index (SSIM) [45]. PSNR calculates QoE using mean squared error (MSE) of the original signal to the received one. The ratio is output in terms of dB and can be thought of as the ratio between the maximum possible power of the source stream and the power of distorting noise affecting its quality. PSNR is a popular method to assess the difference in quality between two videos despite being inaccurate, as was shown in [46].

VQM [44] was developed to objectively measure the perceived quality of video in terms of perceptual effects of impairments, including blurriness, noise, block and color distortions. It then joins them into one metric by means of linear combination. VQM has shown a high degree of correlation with subjective video quality scores as reported by the Video Quality Experts Group (VQEG) [47] in their Phase II validation tests. As a result, ANSI (ANSI T1.801.03-2003) and ITU-T (ITU-T J.144, ITU-R BT.1683) have decided to adopt VQM as a standard for video quality measurement.

SSIM [45] is a method to measure similarity between two images, and hence needs the source and destination streams. It assesses three terms: luminance, contrast and structure. SSIM is based on the fact that the human vision system focuses more on extracting structural information rather than errors happening in the viewing field. It is regarded as one of the most accurate metrics used for calculating video QoE.

ITU issued standards for measuring video QoE of standard definition TV: ITU-T Rec. J144 and ITU-R BT.1683 as FR models, and ITU-T Rec. J.249 as a RR model. ITU standards for multimedia include a FR model in ITU-T Rec. J.247, and a RR model in ITU-T Rec. J.146.

2.5. Summary

This chapter discussed background details pertinent to address the issue of SSONs autonomic management. The use of QoS schemes is not enough to guarantee a good user experience. There are different factors affecting video QoE, many of which are hard to quantify. The use of QoS parameters to predict QoE has been a successful approach followed by many researchers due to its simplicity and the availability of QoS measurements schemes. The increased complexity and heterogeneity of computer networks pose a challenge to the management of services. It is hence necessary to be able to monitor network resources and users' QoE to guarantee proper functioning of equipment and acceptable users QoE at a reduced cost. In this work, we refer to IPTV QoE from a picture quality perspective. Audio quality, synchronization issues and other parameters that affect QoE, such as freshness of content, are out of scope. In the next chapter, we discuss the major contributions in the field of networks management in general and autonomic networks management in particular.

Chapter 3

State-of-the-Art in Autonomic Networks Management

This chapter presents a survey of current research efforts related to the management of SSONs and is organized as follows. Traditional schemes for networks management are presented in Section 3.1. Efforts towards automating the management tasks of computer networks are outlined in Section 3.2 followed by those inspired by IBM's autonomic computing principles. Section 3.3 cites major works that use overlay networks to manage computer networks and important contributions for overlay networks management. Section 3.4 identifies key contributions towards SSONs management which includes solutions that tackle different aspects of the IBM MAPE control loop. Section 3.5 concludes the chapter with a summary of existing contributions and some open issues.

3.1. Traditional Networks Management

The management of communication networks involves the control and organization of network functionalities and the behaviour of its different components, with the purpose of reaching a set of predefined high-level objectives and goals (e.g., maximize resource utilization and reduce operational cost). Historically speaking, many network management schemes have been proposed, some of which were adopted by standardization bodies. Management functionalities were divided into five major categories referred to as FCAPS. FCAPS stands for Fault, Configuration, Accounting, Performance and Security, the classes where network management tasks are defined. The focus of previous research works was to develop simple protocols to automate network management. The goal was to collect monitoring information and store it into a database for later access by administrators, who

will inspect these high volumes of data and then perform different service and component configuration using interfaces.

3.1.1. SNMP

Simple Network Management Protocol (SNMP) has achieved high acceptance since its creation in 1988 to manage network elements in the ever growing Internet. SNMP came to replace its predecessor Simple Gateway Management Protocol (SGMP), and was meant to be replaced on the long term by a solution based on the Common Management Information Service/Protocol (CMIS/CMIP). It is based on a manager/agent model that consists of a manager, an agent, a management information base (MIB), managed devices and the network protocol. The SNMP manager and the SNMP agent communicate through a set of commands and an SNMP MIB [48]. Human administrators were responsible for manually interpreting and analyzing the monitored data, which is of significant size. They then have to make decisions to change the system configuration in order to achieve high-level objectives.

3.1.2. Web-based Network Management

Web-based network management [49] and the common object request broker architecture (CORBA) [50] are two other major contributions towards easing the network management tasks. They were proposed to overcome the problems of traditional network management solutions, namely cost, limited support to third-party RDBMSs, and platform-dependency, meaning that some traditional systems force their customers to support a Unix system. The use of web services and distributed object technologies, CORBA in particular, for network management overcome the aforementioned issues of SNMP. However, they too suffer from many issues. The main problem arises from the added complexity and overhead of making even a simple change, such as adding a new network variable to the MIB, which requires manual intervention, changes in its relation to other variables and in already existing programs. Furthermore, the performance of these systems relied strongly on human expertise.

3.2. Automatic and Autonomic Networks Management

As communication networks grew in size and complexity, more sophisticated management solutions had to be developed in order to minimize the involvement of human experts in the management tasks. Traditional schemes were not only costly, but were also time consuming because of the time it takes humans to process and act upon huge amounts of monitored data.

3.2.1. Automatic Networks Management

Software agents [1], active networks [2], and policy languages [3] represent major research efforts to automate the management of communication networks. Agents, being stationary or mobile, can be used to perform certain tasks on behalf of the human administrator by automating it. An advantage of using agents for network management is their extensibility. Agents can be added or removed from the network in a dynamic way, which makes it flexible to change the computation required for a specific management task. Mobile agents can also be very beneficiary to networks management [51]. They are efficient, save resource consumption and reduce network traffic due to the small size of code they usually run, and the fact that they are deployed only when needed. They are also robust and fault tolerant in cases when the delegation entity fails, interact with real-time systems to prevent delays, and can be used in heterogeneous environments.

On the other hand, the active networks technology is used to dynamically insert active programs at runtime in order to control network equipment. The high number of managed elements and their large distribution is one of the major problems for networks management. Active networks can alleviate this problem by distributing the management centers inside the network. They are also flexible permitting the replacement of protocols and services as needed by the network. The processing and time constraints of centralized management schemes can be easily overcome using active networks by distributing the load among different parts of the network. A major drawback of active networks is that already existing systems had to be redesigned in order to program components to perform

advanced operations and computations. This also comes at added complexity to the devices management functionalities, which incurs more overhead and delays. Both active networks and mobile agents technologies are based on executing code remotely on other devices. This poses security risks and represents one of the main reasons why they were not adopted in commercial solutions.

Policies allow changing the behaviour of network components at runtime without altering their operation. They simplify the management tasks by defining the behaviour of different equipment in response to changes in the network environment. They basically represent rules set by network administrators to guide the actions of network components. These rules are converted into component-specific policies that are stored into a repository and enforced when needed. Policies have the advantage of allowing network entities to be configured on the fly without affecting system operation.

Although these technologies add a great deal of automation to the management of communication networks, they still rely on the intervention of human experts. Human operators were faced with the trouble of having to describe and program precise functionalities of different (most probably) heterogeneous equipment. They have to develop appropriate policies, and keep modifying this behaviour as a response to the continuously changing environment conditions. Another limitation of these approaches is their reactive nature, in a sense that human intervention relies on changes in performance indicators for system malfunction or components failure. Unfortunately, these approaches are unable to cope and adapt with the ever growing business objectives and diverse users' requirements. The incorporation of autonomic computing principles in the management of computer networks constitutes a promising solution to the abovementioned problems and limitations.

3.2.2. Autonomic Management of Communications Networks

As we previously stated, the notion of autonomic computing was inspired from the human autonomic nervous system that regulates different entities of the body without involving the human sub consciousness. By incorporating the principles of autonomic

computing into the management tasks of communication networks, researchers and industrial implementers aim at freeing human administrators from the burden of low-level device configuration, so that they focus more on business logic. Some of the important efforts towards the autonomic management of networks and services include the Foundation, Observation, Comparison, Action and Learning Environment (FOCALE) architecture [52]. FOCALE alleviates problems caused by the heterogeneity of network components with different functionalities and programming models. The architecture was designed to address a number of issues, such as data integration from different sources and the dynamic adaptation of resources and services in the network. The basic idea is to use the different knowledge engineering techniques available to manage new and legacy network components in a seamless manner. Ontologies were used as a shared vocabulary for semantically equivalent components and policies. Systems and components within the FOCALE architecture operate with the help of a self-learning control loop. Vendor-specific data is gathered from network components and then analyzed to ensure that services are running in an appropriate manner. If not, managed resources are reconfigured and re-analyzed to ensure that they are running at the desired state.

The Generic Autonomic Network Architecture (GANA) [53] proposes a reference model to enable protocols, nodes and networks to provide autonomic management. A Decision Element (DE) represents the main autonomic concept in GANA. It implements a control-loop and manages a number of elements assigned to it. The self-* autonomic properties are realized by the different DEs present in the network. GANA defines a hierarchical control loop framework that establishes four levels of abstraction for the design of DEs, managed entities and the corresponding control loops. These are protocol-level, abstracted functions level, node level and network level DEs. DEs at higher levels manage those at lower levels all the way to the managed elements in order to force them to take specific decisions to enforce a desired behaviour or reach a desired state.

The Autonomic Network Management Architecture (ANEMA) [54] defines a management model where objectives fed by human operators are converted into policies

and enforced into network components using a utility-based approach. These utility functions are described in terms of goal policies to achieve the desired utility. The architecture is made up of two layers: the objective definition layer and the objective achievement layer. The former is mainly composed of an objective definition point where administrators input their high level requirements and experts introduce high level management tasks. The objective achievement layer, on the other hand, contains a set of autonomic entities called Goal Achievement Points (GAPs). Each GAP tries to achieve objective high-level requirements using the corresponding network utility functions optimization models and the specified goal policies. A GAP runs IBM's MAPE control loop and communicates with other GAPs in its environment.

The Autonomic Internet (AutoI) project architecture [55] is another example of hierarchical approaches that define several management levels. The proposed architecture introduces two important aspects. First, it allows different heterogeneous domains to interact and cooperate to achieve certain system wide goals. Second, it enables the autonomic management of virtualized resources and services which makes it flexible and extensible allowing for example user mobility and service migration. The AutoI architecture consists of two-levels: the lower-level is responsible for autonomic management of a domain while the upper-level provides some sort of autonomous coordination between lower-level components; it also allows different domains owned by different operators to regulate their configuration in an automatic manner to achieve network wide goals. The architecture uses policy-based adaptation methods to deal with changes in the network. The main drawback of AutoI is its monitoring scheme which is not scalable because it is based on a centralized controller.

Dynamic Readdressing and Management for the Army (DRAMA) [56] is a policy-based management tool for MANETs. The main goal of DRAMA is to enable MANETs to adjust their behaviour without human involvement, i.e. in an autonomic manner. The system is organized into hierarchical clusters where managers on top of each cluster or domain cooperate to manage the network with the help of policies. Some of DRAMA's

disadvantages are that it does not constitute a complete solution for network management and planning automation. Moreover, it does not specify how distributed coordination is achieved among the different system components. Finally, the proposed centralized structure of the highest management level poses scalability and efficiency concerns, and represents a single failure point.

Context-Awareness for the Autonomic Management of MANETs [57] is another platform to autonomically manage MANETs. The authors propose the use of a policy-based network management approach along with context awareness to enable the self-management of MANETs. The proposed model consists of a three-tier hierarchical architecture. The policy-based approach is organized in a hybrid manner, both hierarchical and distributed, and forms hyper-clusters. The proposed solution is not fully autonomic because it only deals with self-configuration and does not address the remaining self-management properties (self-optimization, self-protection and self-healing). The majority of network components are sensors and effectors with the simple role of gathering data, processing it, and sending it to the cluster head to enforce policies. The control loop is in fact formed between nodes with different hierarchies rather than between inner components of each node. The major contribution of the architecture is the use of context-awareness for the autonomic management of MANETs.

The Unity architecture [58] is an IBM proposition towards the self-management of distributed computing systems. Unity is an agent-based and service oriented architecture that adopts three self-management properties: self-configuration, self-healing and self-optimization. The different components present in the system are all autonomic elements that cooperate to optimize the performance of the system and fulfill SLAs. Each autonomic element manages itself and interacts with other elements to exchange monitoring information and provide other services.

The Autonomous Decentralized Management Architecture (ADMA) [59] is another architecture that incorporates autonomic principles into the management of MANETs in a

distributed manner. It differs from previous architectures in the use of P2P techniques which suppresses disadvantages of centralized schemes. The main goal however remains the same, that of enabling some degree of self-configuration of MANETs. The nodes cooperate together to achieve high level goals set by network administrators. ADMA combines policy-based management with autonomic principles in a distributed manner. However, it suffers from some disadvantages. Coordination between different nodes is limited to the duplication of policies and collection of monitored data. The operation of the network depends on the policies pre-defined by human administrators, which are prone to human errors and might bring inconsistency into the network. Besides, there is no focus on the plan and analyze components of the autonomic control loop.

The 4WARD project [60] is another FP7 European project that aims at redesigning networking models towards a clean slate framework for network of the future. One of its proposals is In-Network Management (INM), which represents a distributed autonomic architecture where management functionalities are embedded within the network. One of the strong aspects of INM was its support for migration of existing networks moving the Internet towards a pure INM system. A service-oriented architecture helped in achieving this goal where future enhancements of the INM architecture can be seen as a new service that is either a composition of already existing services or consists of a new enabled one. The proposed solution introduces two entities, INM-enabled and non INM-enabled entities. Non INM-enabled entities are network elements that do not support the INM architecture, and are hence managed by INM-enabled entities. This helps to ease the integration and gradual deployment of INM into the Internet. A great deal of attention was given to the monitoring and self-adaptation parts of the ACL. INM suffers from two main issues: security and cooperation between the different network entities to achieve a network-wide autonomic behaviour.

In addition to the aforementioned autonomic network management architectures, there have been substantial efforts to use overlay networks techniques in the management of communications networks, taking advantage of their inherent self-configuration

property. In addition, the introduction of service overlay networks is providing end-to-end QoS over the Internet and facilitating the creation and adoption of new and complex services. IEEE spotted a need for SON standardization and formed a working group [5] to develop and standardize a framework for next generation SONs, which describes their different capabilities and potential issues. In the following section, we will briefly outline major contributions that use overlay networks techniques for autonomic network management, as well as recent proposals towards the autonomic management of SONs.

3.3. Overlay Networks Management

The use of overlay networks and P2P techniques in particular for the autonomic management of communications networks was mainly tied up to solve the problems of self-configuration and self-adaption. Overlay networks are characterized by being highly dynamic and have unpredictable behaviour due to the freedom of nodes that join and leave the network. They build an abstraction layer on top of physical resources which eliminates heterogeneity issues. Moreover, they are scalable and organize themselves to offer specific services in an efficient manner. For instance, protocols for data storage/retrieval, such as Chord [21], use distributed hash tables to organize keys and nodes in the network in order to provide efficient data search services.

3.3.1. Overlay Networks for Networks Management

Overlay networks have been used to increase routing robustness in cases where links fail unexpectedly. For instance, RONs [4] allow nodes in the network to find the best paths for data delivery in terms of latency. This property enables network nodes to run in an optimal setting in terms of low latency, which is one step towards achieving self-optimization. They also allow fast recovery from faults and links failures which can be seen as a form of self-healing. On the other hand, there have been substantial research efforts towards the use of structured overlay networks for data monitoring and knowledge management.

In [61], data from RON experiments is analyzed to study the difference in using overlay routing instead of IP routing when IP routes fail or when there are better routes in the overlay. Two mechanisms were studied: in the first one, nodes probe each other and measure the loss rate, and then select the best path based on the average loss rate. In the second mechanism, data is sent over multiple overlay paths to increase the redundancy and avoid faults and routes failures.

In [62], the authors study the effectiveness of different overlay networks failure detection and recovery mechanisms to overcome IP-layer path anomalies and provide users with improved routing services. The performance of overlay networks is studied in terms of failure detection and recovery, the stability of the network and the overhead incurred. In particular, different IP-layer path failure characteristics, overlay topologies, detection and restoration factors are investigated to study their effect on performance of the overlay network. The authors show that highly connected overlays are useful for failure detection but are counterproductive for failure recovery, and propose the use of a different topology with much smaller node degrees.

The authors of [63] propose the use of overlay networks to detect failures in MANETs. The basic idea is to introduce detectors nodes that periodically sense each other's presence using heartbeat messages. Detectors are also responsible for sensing other non-detector nodes. MANET nodes form a cooperating overlay network where they exchange data to detect nodes failures and autonomously adapt to changing network conditions.

An information monitoring system called PeerCQ (Continual Queries) was proposed in [64]. PeerCQ forms a P2P information monitoring network using a large set of heterogeneous nodes. The architecture focuses on optimizing hot spot queries, which are information requests that are issued by a large number of users. Hot spot queries optimization reduces the overall number of duplicate messages and improves system utilization. The advantages of PeerCQ include management at no extra cost, no hardware or

connection costs and scalability. PeerCQ represents an example that uses P2P techniques to achieve the first task of IBM MAPE loop, which is information and resources monitoring.

The GANA architecture is another example that uses P2P techniques for the sole goal of implementing self-monitoring functions [65]. GANA monitoring elements are responsible for traffic monitoring coordination, information sharing and dissemination, topology discovery, resource discovery, node state monitoring, providing context aware information, fault and conflict management, accounting and registration to network services. P2P techniques are applied in GANA to implement a distributed storage and retrieval database for monitoring information.

BGPmon [66] is an approach for monitoring the Border Gateway Protocol using a publish/subscribe overlay network. BGPmon suppresses unnecessary functionalities such as route selection and data forwarding, and focuses instead on monitoring. The overlay network is used to provide real-time access to monitoring data. Data exchange between different entities of the overlay is XML-based. Different nodes or monitors in BGPmon coordinate and peer up with each other to form an overlay network so that data is easily accessible and exchanged.

Moreover, a global network monitoring model based on overlay networks was proposed in [67]. The overlay is formed by defining representative measurement points that will carry out monitoring and capture an overall overview of the network state. The authors propose to remove redundancy using metrics composition and introduce a simple active measurement methodology. The proposed scheme contributes to a scalable, robust and reliable end-to-end monitoring. The main goal of this work is the creation of a collaborative network monitoring overlay based on cooperation between measurement points that are strategically located in the network. This work, however, represents initial key points towards a global management solution, and does not detail the selection and operation of the measurement points and how data aggregation is done.

A personal network is a concept that extends personal area networks by including remote personal nodes, such as home or office nodes. To reduce the complexity of their management, a system based on P2P publish-subscribe naming was proposed in [68]. The architecture is composed of announcers, subscribers and resolvers connected together in a P2P network. Connectivity between remote personal nodes is usually done via tunnels that are dynamically established. The dynamic nature of nodes due to cluster mobility and roaming makes them change their attachment point and thus overlay tunnels must be re-established in a dynamic way. The locating system is comprised of the P2P naming system, where announcers announce their location and connection information, subscribers perform name subscriptions, and resolvers track changes and update subscribers.

An information management platform based on overlay networks is presented in [69]. Information Management Overlay (IMO) regulates information flow based on the network state and its properties. It consists of a controller that receives QoS requirements from a management application and sets up collection and aggregation points appropriately in the network. Collectors collect data using sensors and filter them before forwarding them to the aggregation points. These latter select relevant data only, aggregate it and then filters it before sending it to the IMO controller. Data filtering and aggregation allows the system to save bandwidth and incurs less overhead at the expense of accuracy. The localisation of information aggregation points is based on the use of policies.

A distributed network management framework based on a P2P overlay network was proposed in [70] and [71]. The overlay network serves to overcome the disadvantages of centralized management, mainly single point of failure, scalability issues and the limited view of the central management unit. The proposed Distributed Network Agent (DNA) is a software that runs on network components and performs tasks in two phases. The first phase consists of performing local tests to ensure it is free from local errors, while the second phase consists of performing distributed tests. DNAs form an overlay network with the goal of maintaining connectivity to the overlay and being able to find other random DNAs in acceptable time. These goals can be easily accomplished with the use of DHTs.

In [72], the authors present an autonomic P2P system of statefull service containers to autonomically adapt the execution of services. They propose the use of control loops to perform adaptation with the help of performance information, service migration and SLA management tools. The uses of P2P techniques solve scalability and decentralization management issues. The authors use Pastry [30] to build the P2P overlay because it fits with their design requirements. Load information is collected by inspecting the P2P routing packets and finding resources using a computed search tree.

Efficient, Scalable and Robust (ESR) [73] is an overlay based on P2P networks to enable autonomic communication. The main idea is to exploit P2P overlay networks to overcome the challenges of autonomic communication. The authors identify three key requirements for a successful P2P system that is compatible with AC. A good system should be able to locate information and route it at high efficiency. It should also scale very well and be robust. Finally, it should prevent malicious nodes and behaviour to affect the overall system performance.

A model for autonomous and decentralized services in MANETs is presented in [74]. MANETs are characterized by having a dynamic topology, unreliable communication, complexity due to the lack of a centralized control and a loose control of nodes. The proposed approach aims at the autonomic provision of services. Focus is given to topology formation, adaptation to changes in the network, scalability and stability of the network. It defines different layers of operation to overcome the following challenges: efficient utilization of available resources in a dynamic environment, independency from the underlying infrastructure, reliability under a dynamic network, reduced management complexity and increased flexibility for developers. The architecture consists of four different layers. The neighbor-to-neighbor layer sits at the bottom and is responsible for delivering upper-layer frames between neighbors. It also maintains routing cache for the routing layer to keep up with the dynamic creation and destruction of neighboring links. The routing layer routes upper layer frames from a node to another one where it is assumed that nodes are not aware of how they can reach each others. On top of that is the topology

maintenance layer. It is responsible of forming a virtual topology of the nodes participating in the overlay, a Chord ring was used. Finally, a DHT layer uses the ring topology to maintain a distributed hash table where information can be stored and retrieved in an efficient manner.

On the other hand, the work presented in [75] proposes a routing protocol that is able to handle networks complexity without any human intervention. This work differs from other related works in the fact that it does not build upon any existing P2P protocol or routing scheme. The proposed routing scheme is capable of efficiently routing messages in a random graph where nodes have random addresses. Inspired by the operation of DHTs, the routing protocol distributes knowledge in the whole network so that any route can be resolved in an efficient time $O(\log n)$. Hence, the routing table size of nodes is also limited to $O(\log n)$ entries regardless of how addresses are assigned. Unlike other schemes, the proposed routing approach does not require any underlying routing protocol.

3.3.2. Autonomic Management of Service Overlay Networks

The properties and characteristics of overlay networks, including self-organization and adaptation to network changes, made of them an important tool for achieving autonomic management of communication networks. We have discussed many approaches and schemes, based on existing overlay networks and P2P systems, to manage communications networks in an autonomic way. However, most published works focus mainly on the monitoring and adaptation aspects of IBM's MAPE loop. A global solution that incorporates all aspects of autonomic computing for the management of networks is still far from being perfect. On the other hand, overlay networks have been proposed mainly to offer a specific service that cannot be offered using the current Internet. SONs are overlay networks that are created to deliver a specific service. SONs suffer from added complexity because of the dynamic changes in the environment, the heterogeneous devices making up the overlay, and the need to offer services at the required QoS. We briefly outline major contributions and findings.

A policy-based architecture was proposed in [76] and [77] to automate the management of overlay networks. The architecture is context-aware and forms a policy layer on top of the overlay layer to ease the management of overlay networks. The policy layer is composed of policy decision points and policy enforcement points, which cooperate to adapt the operation of the overlay network based on the needs and requirements of users, service providers and the network. Policies are generated at run time based on the gathered context. This allows prompt self-adaptation of the overlay network based on the monitored data.

In [78], a self-organizing algorithm that composes autonomic entities into a service specific overlay network is proposed. The idea is to enable network entities to self-organize in an autonomic manner and compose new services without any human intervention or configuration. The following requirements for SSON composition are addressed: decentralization, efficiency, robustness, dynamicity and distributed operation. The proposed scheme is highly efficient compared to other composition schemes. SORD [79] is an extension to [78] that adds resiliency to the resource discovery process. It allows resources to be located with more efficiency and accuracy. The approach is based on the creation of optimal chordal rings to overcome the problems of efficiency and large message overhead found in traditional resource discovery approaches. It allows resources to be found without querying a central entity or involving human administrators in configuring network entities. SORD is targeted for SMART but can be used for any kind of application such as ad hoc and P2P networks.

In order to resolve problems of overlay links failure, [80] proposes the use of a backup path at the overlay to overcome links failures and provide quick recovery. The backup path is calculated based on the probability of correlation between physical links in the underlying network. The ultimate goal of this work is to find routes for the backup path that minimize the joint path failure probability between the initial path and its backup. In addition, different bandwidth allocation algorithms for both paths are studied in terms of their performance. Using backup path algorithms improve the resiliency of overlay

networks and allow fast recovery from unexpected failures. In [81], the issue of overlay network partition is addressed and the authors study how many nodes it takes to partition the network and for how long it will be broken regardless of the topology. The authors use an iterative method to detect cut vertices which result in the partition of the overlay. The recovery process is proactive and consists of creating additional overlay links around nodes that have been detected as cut vertices.

An architecture for creating, configuring, adapting, contextualizing and tearing down Service-Aware Transport Overlays (SATOs) is proposed in [82]. This work tackles the problem of selecting nodes that will collect and aggregate a specific type of context as part of the overlay monitoring process. The proposed solution relies on the use of DHTs by hashing the type of context for a key. Then, nodes responsible for that particular key are selected as those responsible for collecting and aggregating that type of context. This solution makes use of the advantages of DHTs, namely lookup time, load balancing, decentralization and robustness. An entity named ConCoord is responsible for coordinating the collection and aggregation of context information and providing it to SATO managers. The authors present two types of possible adaptations to changes in the network and illustrate how the tear-down process is achieved under the proposed architecture.

The autonomic management of service overlay networks under the scope of the ambient networks project was also discussed in [83] and [84]. The authors propose the use of on-demand creation of SSONs. The proposed self-management approach aims at creating, configuring, adapting and tearing down SSONs. The management architecture consists mainly of an overlay management functional entity responsible of creating and adapting SSONs. Context is gathered by a service context functional entity and fed back to the management entity. The authors suggest the use of DHTs to dynamically select nodes that become responsible for collecting and analyzing the collected information. Sensors register to a local context coordinator which runs an algorithm to select appropriate nodes to build the SSON based on applications requirements. Unfortunately, this work presents a

generic architecture and does not detail the operation of its different components and how different technologies can be used to achieve the different management goals.

An important application of overlay networks is their support for multicasting. Overlay multicast solved scalability and complexity issues of IP multicast. In [85], the authors propose a mechanism to detect failures in overlay multicast systems within one heartbeat interval with the cooperation of other nodes. The idea of cooperative failure detection speeds up the detection process. First, the expected detection time, the probability of false positive and the overhead of failure detection mechanisms are analyzed to study their tradeoffs. Then, the proposed cooperative failure detection scheme is evaluated and compared to other mechanisms to prove its effectiveness and fast detection property.

A model for parameter configuration to detect failures in overlay networks was proposed in [86]. Close attention was given to detection time and the overhead generated, which represents the cost of detection. There is a clear tradeoff between detection time and overhead. A small detection time with a small probability of false positive involves high overhead due to the exchange of a large amount of messages and the sharing of information between nodes.

In [87], the authors evaluate different tree overlay networks adaptation strategies for the self-management of nodes in order to connect them in a tree topology. The evaluation is based on a defined self-organized goal and four metrics: connectedness, connectivity, instability and robustness. The focus of this work is on tree overlay networks, where agents are connected to a high number of children. Eight different strategies to connect peers together in a tree topology were proposed and evaluated based on the abovementioned metrics.

An autonomic platform for delivering services in service-oriented networks was proposed in [88]. The platform presents a self-optimizing solution that aims at balancing the goals of maximizing the business value derived from processing service requests and resources utilization. It mainly consists of a utility-based cooperative service routing

protocol that propagates prices based on the degree of congestion among nodes. This enables a dynamic routing of service requests. A utility function is used to determine the utility of using resources on a particular path and an optimization problem is formulated to maximize it. The utility takes into consideration the associated price due to congestion.

On the other hand, [89] proposes two mathematical programming models for the user assignment to access nodes, traffic routing, and dimensioning of the capacity reserved on overlay links in SONs. Two schemes were proposed for the user assignment and routing optimization problems. The first aims at reducing the cost of the network while ensuring that all users are covered. The second aims at maximizing the network profit by choosing which users will be served, based on the generated revenue from their subscription to a SON service and the cost incurred.

Another work that studies the optimization of bandwidth provisioning in SONs is presented in [90]. The problem is formulated mathematically for a single-link topology and then generalized to any type of topology by proposing an approximate solution. The problem consists of finding optimal link bandwidth that maximizes revenue under certain QoS constraints.

Finally, there has been some limited work towards the monitoring of service overlay networks. In [91], the authors present MONET, a framework for monitoring SONs. MONET proposes that only a subset of overlay links are to be monitored as opposed to having each node monitor its own links. The main goal is to overcome scalability problems in large networks. The main challenge for monitoring is the amount of overhead generated when high accuracy is needed. The main idea is based on cooperating overlay nodes that share measurement information and deduce the performance of some links without the need to directly measure them. With proper knowledge of the IP topology and information sharing, overlay nodes can measure the performance to any other node with minimal overhead. Using additive and multiplicative metrics, measurements can be deduced from the measured subset of overlay links. On the other hand, an algebraic approach that

selectively monitors a set of independent paths to infer the characteristics of other paths in the network was proposed in [92]. In addition to selecting a subset of links, the authors also propose algorithms to adapt the set of links to be included in the monitoring process as nodes join and leave the network, or when links are broken or created.

Manageable Open Overlay Network (MOON) [93] is an event monitoring and aggregation infrastructure that minimizes the monitoring latency and aggregation cost. The proposed solution takes into consideration the large-scale geographical and distribution of overlay nodes. MOON clusters nodes based on their geographical location and organizes them in a way that information is disseminated with minimal cost and latency. Nodes are divided into three types and form clusters. Overlay nodes form different hierarchical clusters where information is disseminated from the bottom of the cluster to its head. Issues such as how the system responds to nodes failures, including when a cluster node fails, and communication performance were not addressed in MOON.

3.4. Autonomic SSONs Management

SSONs constitute a promising approach to the creation of services on the fly and their delivery over the Internet. The different MPs present in the network have different capabilities and offer a wide range of services. Proper operation of SSONs relies on different factors such as accurate discovery of MPs and the ability to compose services with minimum overhead and in a timely manner. Most important is the ability of the SSON to keep up with the dynamic changes in the overlay networks which include not only nodes and link failures, but also changes in the QoS offered by the MPs. The SMART architecture does not specify how SSONs are created and managed or how they are adapted dynamically according to the network state. There are many challenges in supporting autonomic computing principles for the management of SSONs. We outline some key challenges which are currently the focus of extensive efforts in the research community.

- Information reflection and collection: resource discovery mechanisms are essential for information reflection and collection and are a primary building

block for SSONs. The performance and scalability of SSONs are greatly affected by these mechanisms due to the large amount of overhead generated by flooding messages.

- Lack of centralized control: overlay networks are known for their robustness to failures and changes in the network thanks to their distributed operation that does not rely on a centralized entity. The use of a central management entity is out of the question in the case of SSONs. A central management solution suffers from single point of failure, scalability and congestion.

- Malicious behavior: the growth of SSONs and the lack of centralized control make cooperation between overlay nodes essential to preventing malicious and selfish behaviour such as free riding. Malicious nodes might take advantage of naive systems to report falsified information, get better quality, or get offered more resources at the expense of other nodes and users.

- Dynamic environment: overlay networks are known for their highly dynamic environment due to the loose requirements on nodes' presence. A node is free to join and leave the network at will, and is free to participate in offering its services or not. Moreover, nodes fail unexpectedly which complicates matters more, because the system must seamlessly adapt to these conditions in a timely manner. Another aspect of dynamic behaviour is the change in nodes' resources due to different factors, such as over provisioning.

- Heterogeneity: another characteristic of overlay networks is their heterogeneity in terms of hardware, storage capacity, processing power and the software installed. The management solution should hide the details and differences of components and allow for generic and abstract operations to avoid added complexity that stems from heterogeneity.

- No knowledge of underlying network infrastructure: SSONs are usually created without knowledge of the underlying physical network. Service providers usually do not have access to the underlying infrastructure and hence the management solution should use whatever knowledge available at the overlay. Knowledge of the IP network topology can be used to detect and recover from errors, however even when such information is not available, the overlay network should keep functioning at high performance and at minimum cost.

- Security: the security requirements for autonomic operation of SSONs include traditional security objectives, such as authentication, confidentiality and integrity. The lack of a central entity makes security hard to achieve. In the absence of a central entity that assigns identifiers, it is extremely hard to achieve a high level of trust among participating nodes of the overlay.

An autonomic management solution for the management of SSONs should take into consideration the aforementioned properties and characteristics of SSONs. A novel management architecture should conform to IBM's control loop that consists of monitoring, analysis, adaptation and execution components to achieve specific goals set by human administrators.

3.5. Summary

This chapter discussed the different schemes for network management, ranging from traditional solutions to autonomic architectures that attempt to add a degree of self-management to communication networks. Overlay networks were used to overcome different challenges facing networks management including the increased heterogeneity and complexity of devices. With the emergence of SONs, overlay networks are used to build a delivery path for transporting media across nodes in the network based on some QoS/QoE guarantees. Because of the characteristics of overlays, namely node churn, movement and their highly dynamic behavior, it would be impractical to manage them using traditional network schemes. Moreover, the co-existence of multiple SSONs with

different requirements poses another challenge to their management. Nevertheless, it is necessary to come up with a global management solution for these overlays to guarantee their efficiency and ensure optimal operation without wasting resources at the underlying physical network.

The fundamental purpose of autonomic management is to allow systems to achieve certain goals with no or minimal human intervention. This is achieved through cooperation and self-organization of different system components. An SSON consists of a mixture of nodes that differ in terms of hardware, software components, storage and processing capability. For these reasons, some nodes will have limited capabilities and will not be able to take part of the management process. Other nodes might be selfish and prefer not to waste their resources in running algorithms that consume CPU power.

Moreover, the management solution must be robust against nodes movement, failures and churn. Changes in topology and nodes' resources during the operation of an SSON should also be taken care of without, or with minimal service disruption. These and other issues, such as security and scalability, make the management of SSONs challenging. However, an autonomic solution will be greatly beneficial to the operation and deployment of SSONs, because it provides management at reduced cost and increased efficiency. In this work, we first propose a distributed architecture for the autonomic management of SSONs. We present the different components and modules included in the architecture as well as their functionalities. We then propose a distributed algorithm for the election and selection of AMs among the set of available nodes of the network. The algorithm allows nodes to self-organize to build an efficient management structure to gather and store data in an effective distributed way. Finally, we show how the architecture can be used to monitor the QoE of IPTV streams using utility functions. The proposed function enables service providers running on overlay networks to monitor IPTV QoE of their customers with minimal overhead and in a light manner. The function is independent of the underlying physical network and relies on measurements at the overlay to predict IPTV QoE. It also

provides a way to adapt it to users' needs depending on criteria such as price, and is shown to work for different types of videos.

Chapter 4

An Architecture for Autonomic Overlay Networks Management

4.1. Introduction

The emergence of the service centric paradigm of the current Internet due to the introduction of a high number of new services, in particular multimedia services, has raised concerns over traditional ways that treat networks as "dumb pipes". Charging customers flat rates for using network resources does not guarantee the return on investment for network operators. This on its turn holds back network operators from upgrading and expanding their equipment to offer better QoS to different services. On the other hand, the lack of expertise of network operators and the non-cooperation from service providers who refuse to disclose crucial information about their services restrict the emergence of new services as well as their flexibility. A promising solution consists of separating network infrastructure management and service delivery to decrease the management complexity. In parallel to service delivery, there has been considerable development in the network hardware which has made networks infrastructure more heterogeneous. Hence, a single service is more likely to be delivered through a wide variety of transmission media for an End-to-End (E2E) service delivery. This increased heterogeneity represents a challenge to service providers in meeting customers' QoE expectations. Network failures have also become more common in modern networks. Overlay networks [4] were introduced to address the above issues. SSONs in particular represent a promising solution for the creation and delivery of new complex service through the composition of already existing ones. This chapter proposes an autonomic management architecture for the discovery, configuration and management of services in heterogeneous networks.

4.2. Overlay Architectures

The proposed architecture is based on the essence of autonomic computing to achieve a certain degree of self-management with minimal or no human intervention. As previously discussed, autonomic computing refers to self-management characteristics of distributed computing systems, which adapt to unpredictable changes while hiding the details and complexity of the system from operators and users. The vital goal of autonomic computing is the development of systems that are able to perform self-management. Self-management refers to self-CHOP aspects that are achieved by means of a control loop. This latter monitors and analyzes the system for faults, and then plans and executes adaptation strategies. The behaviour of autonomous systems is driven by high level policies that are business goals, which enable it to make its own decisions, optimize its status and optimally adapt to changing conditions. An autonomous system is generally modeled with two main control loops (global and local) that implement self-monitoring capabilities via sensors and self-adjustment capabilities aided by effectors.

The architecture illustrated in Figure 1 was proposed in previous works [78] [79]. Figure 7 illustrates the network scenario where the proposed architecture is applicable. An application that originates from a MS is offered as a composition of different services provided by different MPs to finally deliver the composed service to the client, MC. AN SSON is formed by selecting optimal nodes to represent MPs that will take part of the composition process. When a request arrives, a single SSON is built for each user and therefore the maximum QoS guarantee needs to be provided for each request. The construction of SSONs makes use of MPs which implement network side functions having the ability to modify content and services. Examples of services offered by MPs include caching, adaptation and synchronization [94]. An SSON is composed of three different types of nodes: multiple MPs, a MS, and a MC. MCs are the actual clients that request media services. MSs are sources of media flows from which MCs receive the desired media services. MPs offer a particular type of service to users. SSONs are created to offer new services originating at the MS by composing existing ones based on the technical, QoS and

QoE requirements of the client. In this chapter, we propose an architecture that has provisions to further improve node selection and service compositions besides providing means for autonomic management of the whole overlay, which includes the complete ACL.

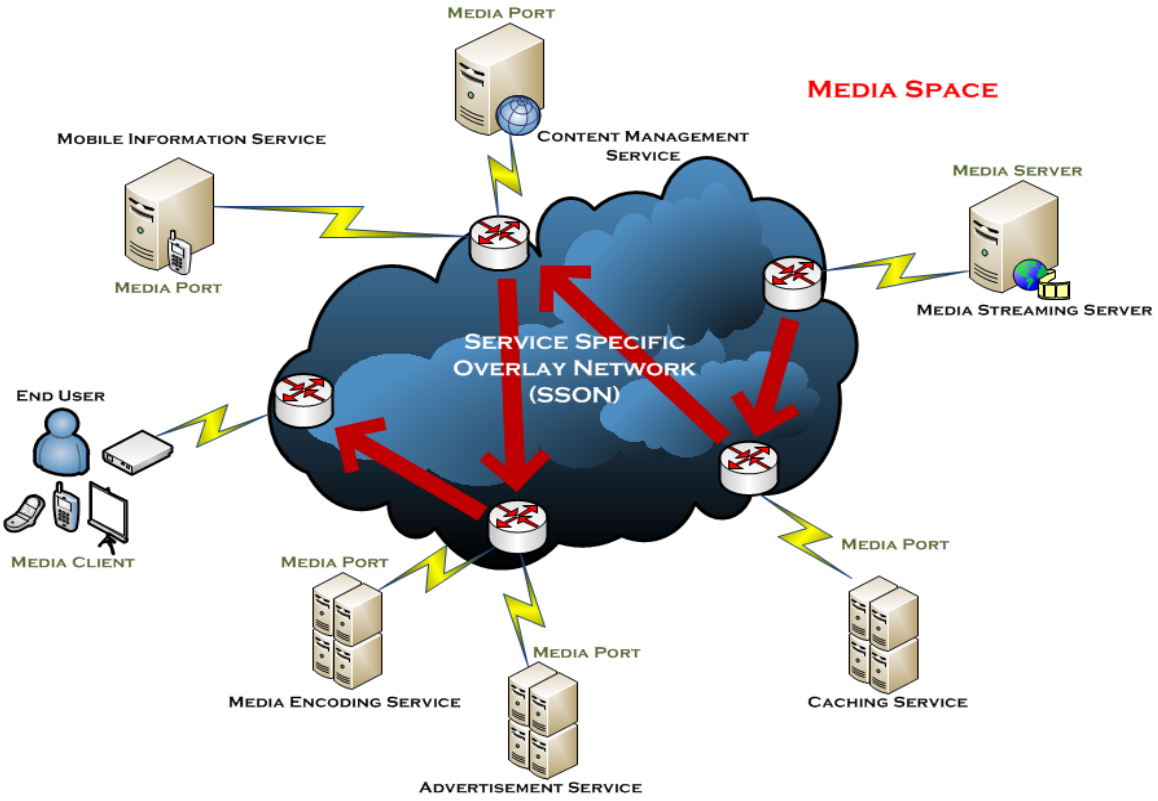


Figure 7: Network Scenario for Multimedia Services [95]

4.3. Proposed Architecture

This section outlines and describes the different functionalities, modules and components of the proposed autonomic architecture to manage media delivery in overlay networks. Some of the components of the proposed architecture were briefly discussed in our published work [95], and will be elaborated more herein. In addition, this section explains how the design goals are met by the proposed architecture. The use of overlay networks eases the separation of services management from the underlying hardware complexities and heterogeneities. It also enables the seamless management of different

networks and the realization of the previously mentioned business model. In this architecture, a fully decentralized management approach is adopted in order to meet the goal of scalability, which also introduces global and local control loops to facilitate the management tasks of both networks and services. Figure 8 demonstrates the proposed autonomic overlay management architecture. As it can be seen, the scope of the architecture is restricted to the management of overlay and SSON only. Each SSON forms a media space consisting of a single MS where the media resides, a number of MPs that alter the media according to the requirements, and a MC where the media is played.

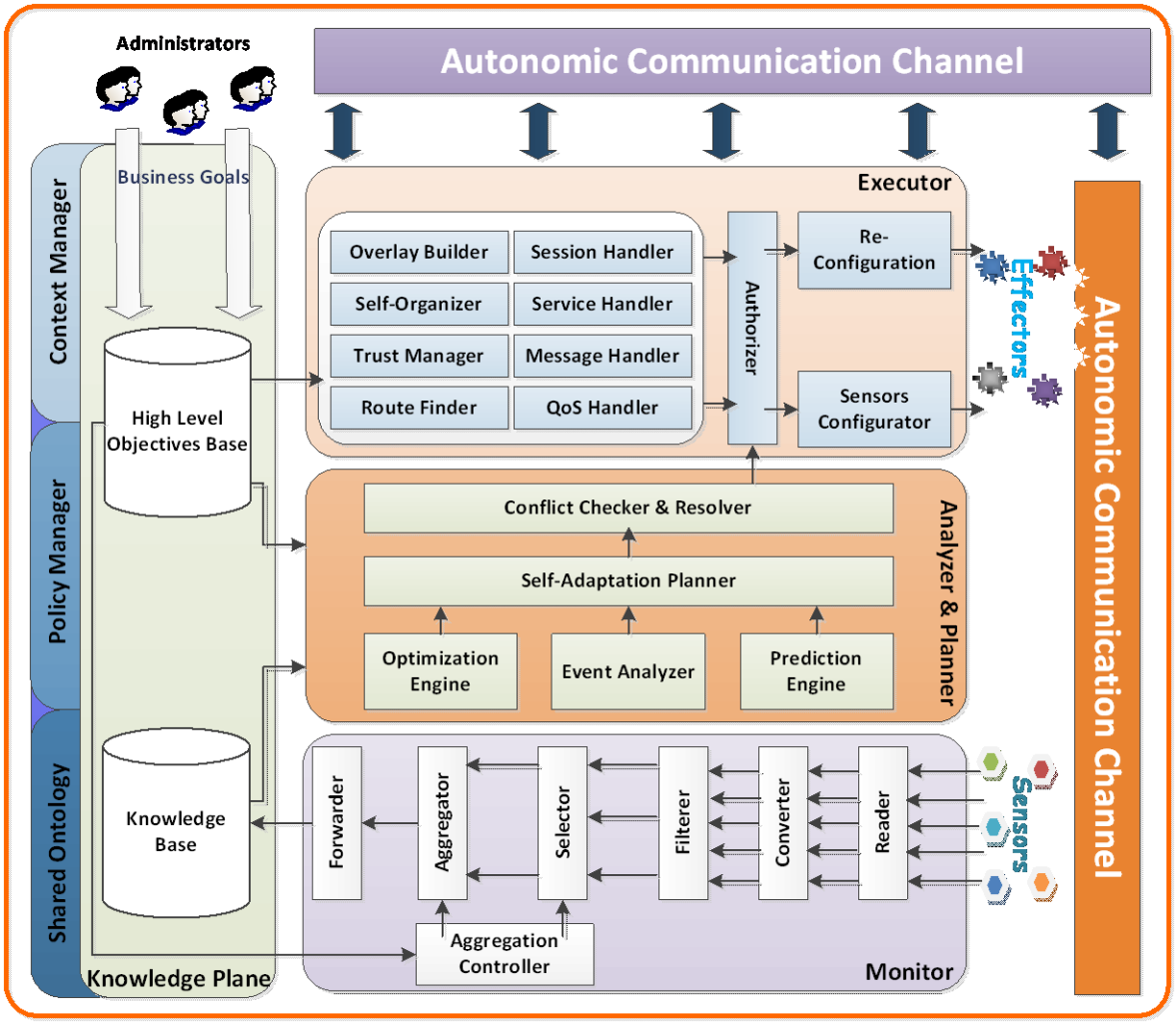


Figure 8: Autonomic Overlay Management Architecture

Each MP is an autonomic component that consists of an AM. Therefore, each MP's functionality completes a local ACL. Therefore, there is no central entity in the architecture. Sensors present in each MP provide the necessary data to the managers. AMs analyze this data to monitor and update the state of the managed elements. The status and the location information of each MP in the network are stored in a decentralized DHT. When an MP is looking for a specific service, it queries the DHT to locate MPs offering that particular service.

The overlay network is composed of nodes organized in both structured and unstructured topologies. In structured overlays, nodes are organized following specific criteria and algorithms, which leads to overlays with specific topologies and properties. They typically use dynamic hash table-based indexing, such as in the Chord system [21]. Unstructured P2P networks, on the other hand, do not organize themselves according to algorithms or optimization of network connections. Unstructured overlay networks can be easily constructed with nodes forming random overlay links. However, they greatly suffer when a peer is looking to acquire some data. The query is flooded into the network to find the node holding the needed data. If the service clients are looking for is not popular, then it is less likely that it will be found in a node close to the requester. As a consequence, this operation will result in the exchange of a high number of messages that would eventually consume significant bandwidth in a wireless network besides the operational overhead. Additionally, since there is no correlation between the requested service and the corresponding MP, there is a high chance that the search will be unsuccessful. The use of structured overlay networks overcomes many of the limitations of unstructured networks. DHTs are used in structured overlays to distribute content in the network and offer efficient lookup and retrieval times. The main disadvantage of structured overlays is the high number of messages exchanged between peers to stabilize the network, maintain connectivity and restore links that fail when nodes leave the network unexpectedly. This represents a major limitation for highly dynamic mobile environments where nodes leave the network at will and new nodes join the network constantly.

4.3.1. Media Port

A MP offers a particular type of service to users like caching, synchronization video encoding/decoding and content insertion. Our work does not assume a specific service description. Services can be described using standard Web Service Description Language (WSDL) [96] for example, and extended with semantic meta-data. For simplicity, a MP service can be described using a service identification ID, an input I, an output O, and the function f that the service provides. Using this simple representation, a service S always receives I and produces O as a result of applying f on I. Each service used incurs a cost, and each MP provides one or more services. We assume that the media end points do not alter the media flow to avoid overloading servers with too many requests or clients with limited capabilities and processing power. Therefore, they are described using their I and O only. For a MC, I refers to the possible input format, and O refers to the content output channel (e.g., Display). For a MS, I refers to the content input, and O refers to the encoding scheme. Using this simple description scheme, a MC requesting content from a MS can be served directly only if the input I of the client is compatible to the output O of the server. In the case of non-compatibility, a MP (or perhaps more than one) has to be inserted between the MS and the MC to establish the media delivery. Given an input media I and a requested output media O, the problem is to find a set of services (or MPs) that transforms I into O and minimizes or maximizes a cost criterion. The result is that the MPs are chained to process the media flow.

MPs can be described according to their input and output ports: single, splitters or joiners [97]. Single MPs have only one input port and one output port. They take a media flow as input and transform it into a different output flow according to the service function that they offer. Splitters have one input and several outputs. They take one media flow as an input and produce a number of output flows. A splitter might for example take a video as an input and produce audio and video as an output. Joiners have several inputs that they merge into one output. Similar to the previous example, a joiner might take an audio and a video flow as input and produce a video flow as its output. Services can therefore be

independent, or partially or completely composed. Independent MPs can perform a service without the help from the other MPs. Partially composed MPs consist of at least two MPs where the output and inputs of the first can be composed with some of the inputs and outputs of the second. They are partially composed in the sense that they need the other MPs to provide a complete service. Completely composed MPs are made up of at least two MPs where all the output and input ports of the first are composed with all the input or outputs ports of the second. Also, completely composed are MPs where all the outputs of the first are composed with some of the inputs of the second and where the remaining inputs are composed with all the inputs of a third MP. In other words, completely composed MPs are those that provide a complete service.

Each overlay node is a potential MP. Each one of them is equipped with an AM. As illustrated in Figure 8, an AM handles both the overlay's and the SSON's functionalities. An AM consists of a Monitor, an Analyzer and Planner, and an Executor.

4.3.2. Autonomic Manager Monitor

The monitor is responsible for collecting necessary measurement of the environment that are of significance to the operation of SSONs. Sensors retrieve any required information including common operations such as CPU and memory usage, the number of packets and bytes coming in and out of network interfaces, and statistics of storage on a node. Sensors should be able to run independently, making it possible for them to gather and report data at different rates. They can also be turned on and off if for some reason the AM responsible for them deems appropriate.

The reader collects all the measurements from sensors either at regular intervals or as an event coming from the sensor. The converter is responsible of transforming raw data into a common measurement object. This object contains information about the sensor, a timestamp and the data retrieved from the sensor. The filterer takes the information from the converter and filters them before forwarding them to the selector. The goal of filtering is to reduce the amount of measurement and hence reduce the overhead. This can be

achieved by only sending values that have significantly changed since previous measurement. Filtering can be used to trade accuracy with overhead [98].

The selector, aggregator and aggregation controller work altogether to achieve certain goals, mainly reduce the amount of final measurement data that will be stored in the knowledge base. The aggregation controller indicates when to perform an aggregation, what will be aggregated and how the aggregator should proceed with an aggregation. The when specification causes the aggregator to run in intervals; waking up every X seconds to perform an aggregation. The what specification defines the time interval for aggregation such as "the last 10 seconds". Finally, the how specification represents a function to aggregate the data. Different functions can be used, such as sum and average. The what specification from the aggregation controller is fed to the selector. The aggregator aggregates any selected data the selector presents using the how specification from the aggregation controller and hands over the result to the forwarder.

The forwarder inserts the final aggregated data into the knowledge base by forwarding it to the node responsible for storing it. Using this approach, we can make use of the advantages of DHTs, mainly efficient storage and lookup times, to efficiently insert and retrieve data. The distribution of data across nodes in the network eliminates a single point of failure and distributes the load over the network. Moreover, when nodes that store data fail, we do not lose all the gathered information as would be the case with a centralized approach. Replication and maintenance mechanisms such as [99] can be used to increase system resiliency. The specifications of the aggregation controller are determined from the high level objectives fed by administrators. The information is gathered and processed by the monitor for presentation to the analyzer and planner components. It is then analyzed for any problems, misconfigurations, and possible optimizations in order to improve the operation of services.

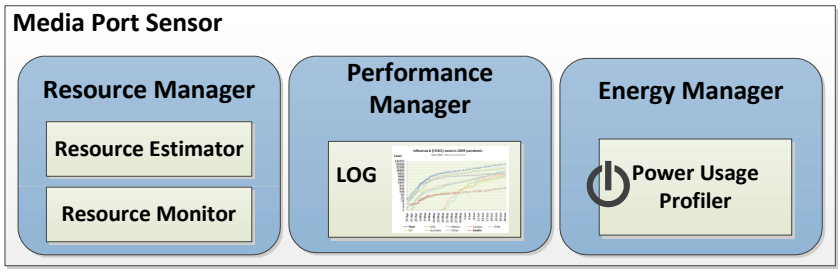


Figure 9: Media Port Sensor

Figure 9 illustrates a sensor, which consists of a Resource Manager (RM), a Performance Manager (PM) and an Energy Manager (EM). A RM senses the usage of resources within a MP such as storage space, processing power and main memory. RM is used to obtain the availability of resources in a MP. PM keeps track of the performance level of the services a MP provides. An EM on the other hand evaluates the energy consumed by a MP based on resource usage data and Energy Profiler (EP). EP is a table that gives the energy consumption for different levels of resource usages. The sensory information is stored in the Context Manager (CM).

4.3.3. Autonomic Manager Analyzer and Planner

The gathered data is analyzed by the analyzer based on high level objectives to deduce relevant information. Basically, it examines the current performance, detects relevant events and predicts the future state of the network. The optimization engine strives to improve the performance of services and the network as a whole. The event analyzer discovers pertinent events that indicate degradation of QoS and QoE. The prediction engine predicts the future state of the network to minimize or avoid service disruption and correct problems beforehand. These components report directly to the self-adaptation planner which constructs adaptation plans based on the situation identified by the analysis process. Once a plan has been calculated by the planner, it is checked against any conflicts that could arise from its execution. The conflict checker examines the impact of the plan on the operation of other SSONs, and makes a decision on either passing the plan to the executor

or to the conflict resolver. This latter attempt to rectify the situation by altering the plan, and then passing it to the executor.

4.3.4. Autonomic Manager Executor

The executor reconfigures the managed elements based on the output of the analyzer and planner components, and communicates with other AMs via the autonomic communication channel. Before the executor can perform any re-configurations, including sensors re-configurations (such as turning them on and off), it checks with the authorizer to make sure that it has the proper privileges to do so. In addition to performing re-configurations on managed elements, the executor is also made up of other components for self-management. These components ensure proper operation of the overlay by building overlay links, managing neighbors' relations, finding new routes and handling service and message requests received from neighbors.

4.3.4.1 Self-Organizer

The self-organizer aims at building the DHT-based overlay network for information storage and retrieval. It is also responsible for organizing the nodes in a fully distributed manner to improve the collection and access to gathered information as well as prompt service and resource discovery for service composition.

4.3.4.2 QoS Handler

QoS Handler (QH)'s responsibility is to evaluate the QoS (service delay) levels of services a MP provides based on the inputs from RM and PM. QoS of each MP is evaluated

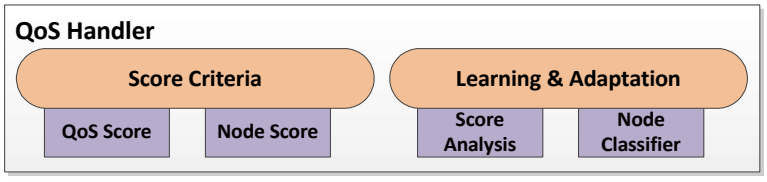


Figure 10: QoS Handler

based on the score assigned to the respective overlay node, based on a number of criteria which includes stability of the nodes, load on the node, energy efficiency, QoS levels and security [100]. A virtual chord like structure is maintained based on the scores given to overlay nodes, where nodes with high scores are in the center and the lower the score is, the farther the node is from the center of the chord structure. The SSON is built by optimally selecting MPs that form the path from MS to MC and also facilitate service composition.

4.3.4.3 Overlay Builder

This module, as illustrated in Figure 11, is responsible for positioning the MP in the overlay and hence building the overlay network in a distributed manner. The formation of the overlay network is performed in three stages: initialization, sub-Chord formation and extension, and promotion-based stabilization and overlay management as discussed in [100].

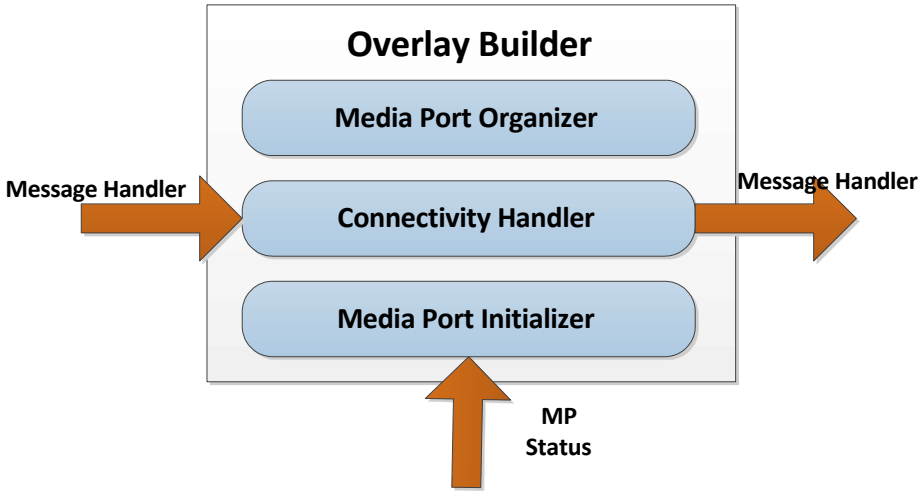


Figure 11: Overlay Builder

4.3.4.4 Route Finder

RF is responsible for finding the best possible route from MS to MC connecting a number of MPs to facilitate effective service composition. In the work presented in [79],

search for MPs to form a composition was performed based on three assumptions: The location of the MC is known, the location of the MS is known, and the search for MPs is performed in the direction emanating from the MC towards the MS only. Location information is provided by the CM. Using this, any node's RF receiving a query will forward the query to local nodes only if it is within a certain degree α. The angle α represents the maximum search scope for possible component services. This value depends on the locations of MC and MS. This approach, in general, avoids sending queries to regions of the network where answers are not likely to be found. However, in wireless mobile networks, the dynamic movement of nodes at the network layer may result in an unequal distribution of MPs at the overlay level. We may end up with high density regions where a large number of diverse MPs exist while the other regions may have them sparsely located. In the latter, it is highly likely that the search degree must be continuously increased until all component services are found.

In recent work [100], the limitations of the previous solution is overcome using two methods. The first is by providing a hybrid lower overlay structure. The second is by dynamic node promotion and demotion according to the node score. These two methods result in a circular MP space consisting of MPs and MSs that is further reorganized by the MP Organizer module residing in Overlay Builder (OB) of MPs based on the types of services provided. As a result, MP initializer modules of the nodes arriving in the network must first determine the overlay region they must join.

4.3.4.5 Authorizer

The Authorizer module is responsible for authorizing the other MPs for receiving and providing services. This is performed mainly to penalize free riders and nodes with malicious behavior. It is based on the trust the respective node has earned from the past negotiations. If the trust score is above a predetermined threshold, then the communication with the corresponding MP is authorized. Such a communication is disallowed if the threshold trust level is not met. Trust score is assigned based on the history, i.e., the more a

node has provided the required services in the past, the higher the score is. It is evaluated as follows:

$$Q_{trust} = \frac{Number\ of\ Service\ Deliveries}{Number\ of\ Requests}. \quad (1)$$

The authorization condition is given as follows:

$$A(x) = \begin{cases} 1, if\ Q_{trust} \geq T_{trust} \\ 0, \quad otherwise \end{cases}. \quad (2)$$

where, T_{trust} is the threshold trust value. More advanced trust schemes based on building reputation for different nodes have been proposed in the litterature and can be used instead of (1). Examples of trust management schemes in overlay networks can be found in [101].

4.3.4.6 Session Handler

The Session handler module is responsible for establishing an overlay session to initiate an end user application process, maintaining it and terminating it. Session handler's main task is to uphold the end-to-end QoS levels of the applications. In cases where there is a drop in the QoS levels, session handler indicates that to Service Handler which does the necessary service specific configurations to keep the QoS at the desired level. This module is also responsible for triggering Route Finder (RF), when there is a failure in the established path, which in turn finds an alternate path.

4.3.4.7 Service Handler

The Service Handler manages the services MS and each MP provide by appropriately configuring them to meet users' requirements. This includes altering the resolution of the video provided.

4.3.5. Knowledge Plane

Knowledge Plane (KP) is responsible for providing the autonomic computing with the required knowledge. The knowledge to form KP in a distributed manner is provided by

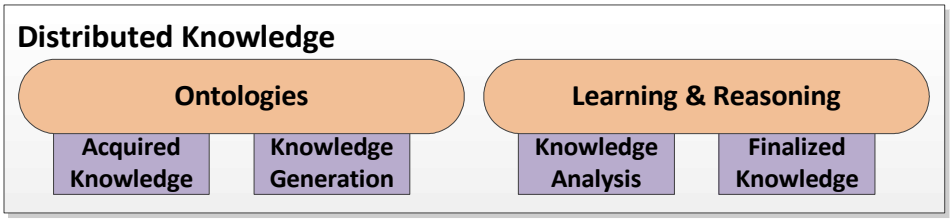


Figure 12: Distributed Knowledge

the context information, ontologies and the set of policies. An ontology is a "formal, explicit specification of a shared conceptualization" [102]. In the context of autonomic network management, [52] define ontology as: "a formal, explicit specification of a shared, machine-readable vocabulary and meanings, in the form of various entities and relationships between them, to describe knowledge about the contents of one or more related subject domains throughout the life cycle of its existence". An ontology provides a shared vocabulary, which can be used to model a domain that is, the type of objects and/or concepts that exist, and their properties and relations. KP infers or generates new knowledge from the acquired knowledge, which then goes through a thorough analysis for conflict detection and removal to maintain consistency. Extensive analysis produces the finalized knowledge.

4.3.5.1 High Level Objectives Base

The high level objectives base is a repository where high level objectives (business goals) are stored. One of the main goals of autonomic management is the elimination or minimization of user intervention. Users and administrators input high level business objectives in the form of policies. The repository serves as ground for initial configuration of network components and directs the operation of other autonomic elements. As depicted in the proposed architecture, the high level objectives are used by the monitor to set up the aggregation controller, by the analyzer and planner to understand what is desired, and by the executor to lead the operation of its different components.

4.3.5.2 Knowledge Base

The knowledge base stores the aggregated collected data, referred to as knowledge. Overlay networks are characterized by being highly dynamic, heterogeneous and experience random churn. Since there is no central entity in an overlay network, a centralized knowledge base would not be practical and would suffer from many disadvantages, mainly a single point of failure. We propose the use of DHTs to build and maintain a knowledge base distributed over the overlay. An election scheme is used to choose the best nodes available in the overlay to become AMs and form a DHT-based storage/retrieval system. A knowledge base can vary in complexity from a plain database into more complex expert systems with built-in reasoning methods. The details of a knowledge base are out of the scope of this paper. However, Samaan et al. [103] provide a summary of the requirements and characteristics of an autonomic knowledge base.

4.3.5.3 Context Manager

The Context Manager (CM) architecture is illustrated in Figure 13. The responsibility of CM is to provide network state information from the monitored data. CM is equipped with the necessary intelligence to obtain service level status information from

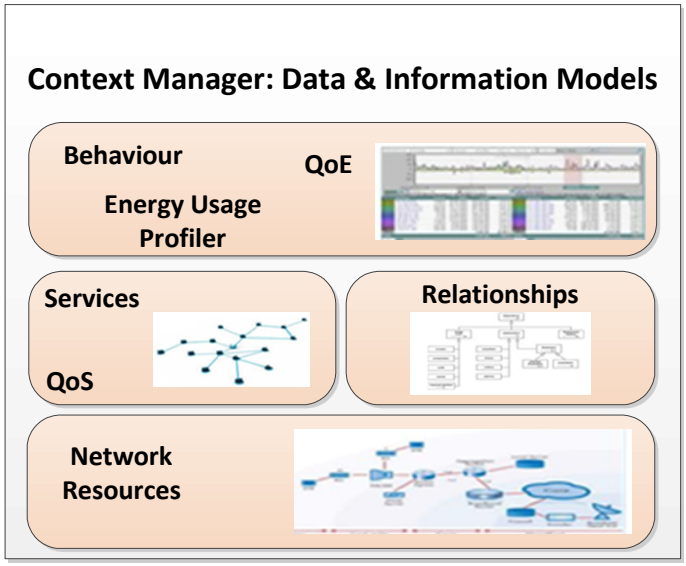


Figure 13: Context Manager

the device level status information. This is processed by gradual inference of statuses at different levels. At the very bottom level, CM acquires the status of network resources from the monitor. It then gathers the status of services provided by MPs, e.g. QoS, and their inter-relationships. Finally, the composite service status is obtained from the status of individual services. Some examples of composite service are QoE and energy usage.

4.3.5.4 Policy Manager

Policy Manager (PolM) is responsible for setting and enforcing a set of policies required to manage the network system in a desirable manner. PolM consists of Simulator Manager and Configuration Manager modules.

The Simulator Manager: The simulator manager is tasked with achieving three major goals: maintaining an accurate network simulation model that closely resembles the physical network, fine-tune the simulator to achieve a synthesized network performance that accurately reflects the expected network performance and finally, minimizing the difference between the desired network performance and objectives, and the actual measurements. To achieve these goals, the manager is aided with two types of policies to control the behavior of both the simulator and the overlay scheme, namely the simulation configuration policies and the actual policies. The former type of policies is employed to configure the run-time simulator with the required simulation scenarios, while the latter controls the selection of adaptive policies. The manager also maintains meta-data describing the static information related to the underlying physical network and the possible configuration scenarios of the simulator. All the necessary network and service quality parameters are collected from the context manager and the network to measure the performance of the network and the overlay scheme.

To maintain scalability, the manager estimates the network and overlay performance using a number of traffic classes: conversational, streaming, emergency, interactive and background. Each traffic class is further divided into three different sub-classes each with a different priority: high, medium and low. These classes are also used as

the basis for defining various overlay configuration policies. The MP Sensors scan the environment to detect any changes. Examples of performance evaluation measurements are: throughput, jitter, end-to-end delay and error rate. Indications of performance degradation trigger a set of simulation configuration policies that execute a number of simulation runs testing one or more new policies. Obtained simulation results are used as indicators to either perform further simulations or to communicate the new policies to the configuration manager.

The Configuration Manager: The Configuration Manager maintains a list of currently active configuration policies communicated by the simulation manager and used to configure the overlay network. Once the simulation manager communicates a new set of overlay configuration policies to the Configuration Manager, the new policies are automatically mapped to configure the running overlay scheme.

4.3.6. Multimedia User Interface

Multimedia User Interface (MUI) resides in MC (e.g., STB) to facilitate a user's interaction with the application in terms of selecting the desired application and content and providing ratings on his/her experience and content. Typically a MUI is equipped with a display screen (e.g. TV) and a controller (e.g. infrared remote control) to control the application.

4.3.6.1 User Feedback Manager

User Feedback Manager (UFM) is a software that resides in the MC to enable users to provide ratings on the content and applications. Users are provided with the rating option in a pre-determined range (e.g., 1-10) on MUI, where they can cast their ratings via the controller. Users' ratings are taken into account to determine the popularity of content and reconfigure the application specific settings to provide the level of service that is as close as possible to users' expectation. For example, the popularity $P(C_i)$ of content C_i based on user rating is given by:

$$P(C_i) = \frac{\sum_{j=1}^{m} r_j}{m}. \quad (3)$$

where, m is the number of users that have provided their ratings and r_j is the rating given by the user j. User ratings on the service quality are taken into consideration to increase the popularity of content and optimize the future use of network resources using caching or replication schemes. Popularity of content can be calculated based on an interval in order to get relevant and up-to-date ratings.

4.4. IPTV Use Case Scenario

This section shows how the architecture can be deployed in a modern IPTV Network Scenario. Let us consider the scenario of delivering VoD only, for simplicity. The architecture is however, applicable to the other multimedia services such as live streaming as well. Figure 14 illustrates the proposed architecture for an IPTV environment. Service delivery is initiated by the users from the end user devices (laptop, PDA, STB), where users request for video content from the list of advertised videos on MUI. We assume that the MC is aware of the nearest MS to send the requests to. Once the nearest MS is known, MC attempts to establish a path from MC to MS. The establishment of the path is performed as explained in the previous section. In case of a VoD request, the location of the content is

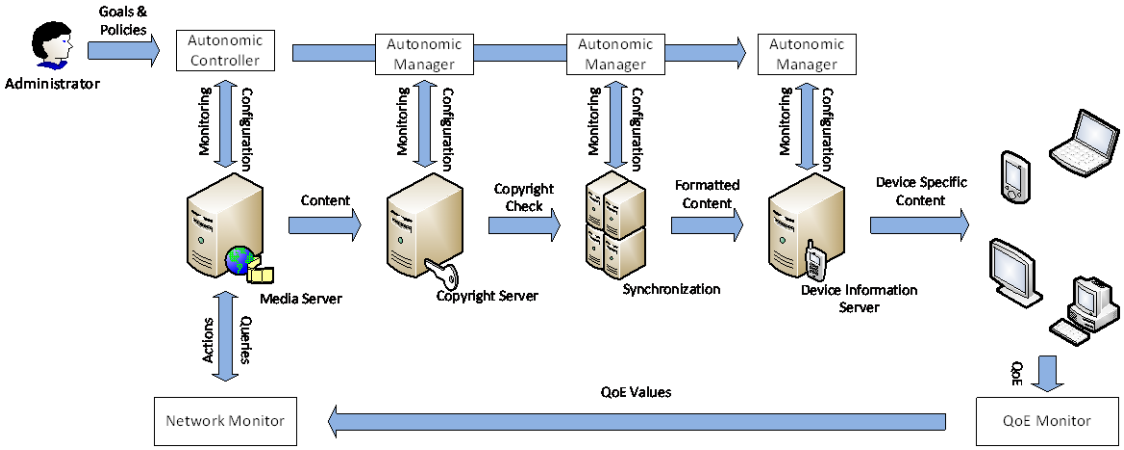


Figure 14: IPTV Network Management

found in the same way an MP offering a particular service is found. This mechanism replaces the customary way of querying a server, thereby eliminating the centralized approach. The type of content is taken into account to narrow the search. Now, this server is the MS of the SSON and begins forming a complete SSON by finding the appropriate MPs to from the path from MS to MC (the client device). MPs perform caching, synchronization and encoding functionalities to deliver the desired type of client device compatible video to MC. These services are managed and configured by the AMs managing the respective MPs. Device Information Server (DIS) identifies the type of device (for e.g., the type of mobile phones can be identified from the identifying number of each phone), and performs the appropriate encoding to deliver the device compatible version of the video.

QoE Monitor is a software supplied by the service provider that resides on the client device, which evaluates the QoE of applications as perceived by the end user. This is performed as explained in the previous section. These QoE levels are then fed to the AM as a feedback. Managers adjust policies appropriately to deliver the desired QoE to the end users. Refined policies are then fed to all the relevant AMs of MCs to adjust their behavior appropriately for the new set of policies. The configurations with respect to the SSON are then fed to overlay builder which does the necessary amendments to the SSON. An example of such an amendment is rerouting video streams due to congestion or failure. The business goals are fed to the managers by the administrators. Feedback control loop established here guarantees that the actual result is as close as possible to the desired result.

4.5. Summary

This chapter proposes an autonomic architecture to support the delivery of media services on the overlay, where SSONs are built to deliver user requested services with QoS guarantee. Different components are described to allow seamless creation of SSONs to provide customized services based on users' needs and requirements, as well as the current state of the network. Shared ontologies are used to give the semantics of the network

components whereby tackling the heterogeneity issues stemming from network infrastructure. The proposed architecture is highly distributed solving scalability issues, and relies on DHTs to store and access knowledge in an efficient manner. The different autonomic managers present in the network self-organize to divide the management load. In the next chapter, we present an election scheme for the election of AMs among the set of available nodes making up the overlay network. We also present a highly distributed algorithm for assigning nodes to AMs under certain criteria so that the gathered data reaches AMs in a timely manner and error free.

Chapter 5

Self-Organizing and Self-Adapting Overlay Networks for Autonomic Management

5.1. Introduction

SONs generally provide the QoS needed for applications such as online gaming, multimedia streaming and VoIP. For some applications, the number of nodes making up an overlay network may reach millions: Skype claims 30 million users at peak times [104]. The nodes' increased heterogeneity and mobility, as well as the fact that nodes join and leave the network at will, pose challenges to service providers in meeting customers' QoS and QoE expectations. The management of such dense and highly dynamic environments requires a comprehensive solution that provides both constant monitoring of network resources and efficient algorithms for the detection and recovery of anomalies. The use of autonomic computing principles to achieve a degree of network self-management with little or no user intervention represents a promising solution for network and service providers. In the notion of autonomic computing, each AM is responsible for managing one or more elements. A node reports information gathered by its sensors to the AM with the goal that data are delivered intact under minimal delays. Receiving information in a timely manner enables the managers to build an accurate view of the current network state. We propose a game theoretic approach to solve the AM selection problem that can arise. Overlay nodes self-organize to form a management architecture where each node selects an AM to manage it with the goal of efficient network performance. To do this, we use a modified regret matching scheme. The advantages are that the scheme is highly distributed and incurs very

limited overhead. More importantly, it adapts the selection dynamically to the time-variant state of the network.

5.2. Autonomic Management of Overlay Networks

The Service Specific Overlay Networks architecture is illustrated in Figure 1. We recall that SSONs are composed of three types of nodes: Media Servers, Media Ports , and Media Clients. MCs request media services from MSs, the sources of media flows. Every MP offers a particular type of service. MPs have the flexibility to modify content and services such as caching, adaptation and synchronization. SSONs are built by composing services based on the technical, QoS, and QoE requirements of MCs. An SSON is built for each user when a request arrives by selecting nodes that optimally represent MPs. Figure 3 illustrates the Autonomic Control Loop (ACL) introduced by IBM. The idea is that autonomic systems manage themselves automatically, as the human nervous system does, by controlling complexity based on high-level objectives. This self-management consists of the four previously discussed principles, collectively referred to as self-CHOP: self-Configuring, self-Healing, self-Optimizing and self-Protecting. Sensors and effectors are essential to the architecture of any AS. An AM creates an ACL by monitoring behavior with sensors and then analyzing it against past and current data. The manager then plans the required actions and uses effectors to execute them. An AM can be responsible for managing more than one element. Information gathered by sensors is sent to monitors for further processing (such as filtering and aggregation) before it is made available in the knowledge base. The knowledge base is distributed among overlay nodes by Distributed Hash Tables with their advantages of lookup time, decentralization, robustness and load balancing.

Every overlay node in Figure 1 implements the ACL in Figure 3 to provide autonomic management of the whole network. This does not mean that each node is active in the management tasks. A smart phone, for example, might not have enough storage and power resources to store and process sensed data. A battery-powered device might reduce

its lifetime if included in the management process, and this could cause it to stop offering its services. The number of active AMs at any particular time is therefore limited to a subset of nodes. A node must then pick an AM to manage it that will result in the lowest end-to-end transmission delay, processing load and link error ratio. A centralized solution would provide the optimal selection of AMs. But it would be very expensive to compute with a large number of nodes. It would also be inefficient due to the dynamic nature of the network in which nodes join, leave and move at will. In addition, the computation of a new solution would be necessary for any slight change in the characteristics of nodes and links. An alternative solution is for a node to pick the closest AM as its manager. Although this solution is simple and easy to implement, it does not necessarily give good results. For example, if an AM is located in a highly dense area, it can become overloaded. To overcome these problems, we propose the use of a game theoretic approach using a modified regret matching procedure. Our proposed scheme is highly decentralized, incurs very little overhead and adapts to changing network conditions.

A number of studies have been carried out to manage networks and services using AC principles. Strassner et al. [52] proposed the FOCALE architecture that uses models and ontologies to represent knowledge. A model-based translation module is located in the middle of the architecture to overcome heterogeneity at the network stack. Additional modules address different aspects of autonomic network management. Liakopoulos et al. proposed the GANA architecture [53] as an approach for implementing self-monitoring functions in autonomic networks. GANA uses P2P techniques to improve performance monitoring and resource management. Each node monitors its local state and communicates with its peers. A hash function is used to store information based on the IP addresses of nodes. Tang et al. proposed techniques and algorithms to create an optimal event monitoring and aggregation infrastructure called MOON [93]. MOON aims to minimize the monitoring latency and event aggregation cost while taking into consideration the large-scale geographical and network distribution of overlay nodes. Nodes are clustered based on their geographical location and organized in such a way that monitoring

information is disseminated with minimal time and cost. Castro et al. [67] proposed initial work towards the formulation of a network monitoring overlay. They define representative measurement points, remove redundancy using metrics composition and introduce a simple active measurement methodology. Chen at al. [92] proposed an algorithm to select a subset of overlay links, to monitor losses on them and, from this data, infer the loss rate on other links. Leitao et al. [105] proposed Hybrid Partial View (HyParView), a fully decentralized membership protocol that builds and maintains an unstructured overlay network. The protocol was designed such that load is distributed in the system and every component is monitored by at least one monitor at any time.

Previous work assumes that all nodes have enough resources to take part in the autonomic management. However, a node may correspond to a portable battery-powered device with very limited resources. In this case, the node may be unable to process data at the required speed. The management tasks could also drain the device's battery if CPU-hungry algorithms that consume power are running. The previous solutions also ignored the dynamic nature of overlay networks in terms of the arrival, departure and movement of nodes. Our highly distributed self-organizing algorithm is proven to converge to the set of correlated equilibrium as time goes. According to [106], a correlated equilibrium is "nothing other than a Nash equilibrium where each player may receive a private signal before the game is played (the signals do not affect the payoffs; and the players may base their choices on the signals received). This may be equivalently described as follows: Assume that the signal of each player i is in fact a "play recommendation" $s^i \in S^i$, where the N-tuple $s = (S^i)_{i \in N}$ is selected (by a "device" or "mediator") according to a commonly known probability distribution. A correlated equilibrium results if each player realizes that the best he can do is to follow the recommendation, provided that all the other players do likewise". The actions of each node represent an optimal reaction to its environment and the actions of other nodes. The algorithm also adapts to the time-varying state of the network, including the mobility and the churn of nodes. Each node is autonomous and does not need to exchange any information with other nodes. The overhead incurred by the

algorithm is very limited. Before presenting the proposed manager selection algorithm, we first describe how AMs are chosen from amongst the set of nodes to be the most powerful and stable ones.

5.3. Promoting and Discovering Autonomic Managers

We assume that wach node has an address necessary for message exchange. In addition, nodes have a profile containing additional relevant information for the definition of an overlay network, such as Node ID and geographical location. We assume that end-to-end delay between two nodes is directly proportional to geographical distance. Given a set of AMs and based on this information, a node can calculate the geographical distance to all AMs and select the top l AMs that constitute smaller end-to-end delay.

In order for service nodes to select an AM, they obviously need to get hold of the list of active AMs present in the network. Since AMs form a DHT structure to store the retrieved information, we propose to insert the list of AMs in the DHT structure itself and make it available through querying the DHT protocol. Nodes that become AMs update the list with their IP address. Likely, when an AM leaves the network or when a node finds out that their corresponding AM crashed, it updates the list accordingly. AMs are chosen from the list of available nodes based on different criteria of available resources including processing power, storage space, high speed broadband connections and high node degree. However, the most important criterion is node stability. It is a combination of both node's mobility and uptime time. Choosing stable nodes to be AMs has many advantages. First, the AM list is rarely updated, saving bandwidth and overhead of constantly adding and removing AMs. Second, the information gathered and stored by AMs remains intact for a long period of time. When an AM leaves the network, it should first transfer the data it stores to the new AM responsible for it. By choosing stable nodes to be AMs, we minimize the number of times this procedure occurs, and hence save bandwidth. In order to protect the gathered data in cases where an AM leaves the network (intentionally or

unintentionally), we propose the use of replication and maintenance mechanisms such as [99].

The promotion of a regular node to become an AM is based on different factors representing the characteristics of overlays. In fact, for overlay networks to be fully autonomic, they must be able to store the gathered knowledge in a distributed manner. We propose the use DHTs to store and retrieve data because of their advantages, namely scalability, decentralization and resiliency to node churn. Many DHT-based protocols for content storage and retrieval have emerged in the literature (e.g. Chord [21], CAN [29], Kadmelia [107], Pastry [30] and Tapestry [108]). Network variations in terms of unexpected nodes arrivals and departures pose a limitation on the performance and availability of DHTs. At best, node churn degrades the performance of DHTs due to the reorganization of keys, which consumes bandwidth and CPU. At worst, it causes some data to be lost. It has been shown in [109] that the use of DHTs becomes viable only when peers stay up for days. Having unstable nodes promoted to AMs not only affects the DHT protocol, but could also negatively disturb the management of deployed SSONs, causing service degradation and customer dissatisfaction.

Only stable and powerful nodes are promoted to AMs. The capabilities of overlay nodes range from powerful servers to smart phones with limited resources. AMs responsible for managing other elements must be powerful enough to run sophisticated algorithms in a timely manner. As nodes join the network, powerful ones are identified and promoted to AMs in a distributed manner. AMs then form a DHT structure based on the protocol used to store and retrieve gathered data and inferred knowledge. We decided to use Chord as the DHT protocol for illustration purposes only; any other DHT protocol can be used. We divide nodes in the overlay into three types:

- Autonomic Managers (AMs) are nodes with highly available resources, currently active in the management process. As opposed to traditional networks

management schemes where managers are static and pre-assigned, the number of AMs should depend on the network and service load.

- Potential Autonomic Managers (PAMs) are nodes that have the capabilities to become AMs but are not needed for the time being. As load increases, PAMs are invited to join the AMs group in order to minimize and distribute the management load. PAMs represent the next best nodes in terms of stability and available resources after the set AMs.

- Battery Powered Devices (BPDs), on the other hand, are nodes that are battery powered. Such nodes constitute laptops and smart phones with limited resources, and hence do not perform management functionalities in order to maximize their lifetime and avoid draining their battery.

AMs and PAMs have the following characteristics:

- Are stable and have a long life span. According to [107], [110], [111] and [112], the longer a node stays in the system, the more likely it will remain online for longer periods of time than newly arriving nodes. Stability is directly related to the node's arrival time.

- Have an adequate level of available resources to be able to manage one or more autonomic elements. Basically, they should have a low load, low mobility probability [113], high CPU and memory capacity, high storage space, and high node degree with broadband links.

We introduce a new metric called the Performance Index (PI) that is a function of all of the above. The more powerful a node is, the higher its PI. Nodes with high PIs are promoted to AMs and start collecting and storing data from the managed elements using the DHT protocol. The number of AMs present in the network at a certain time depends on the total number of nodes being managed as well as the number of deployed SSONs. By choosing only stable nodes to be AMs, we minimize the churn and failure rates of AMs and

ensure that the data they collect and store will remain intact for long periods of time. This helps reduce the overhead of maintaining the DHT-based structure and avoids wasting bandwidth for transferring data from leaving AMs to new AMs. PAMs differ from AMs in two ways. First, they are on standby mode waiting for a chance to become AMs. This can happen in two cases: as more nodes join the network or as new SSONs are created, the management load increases calling for more AMs. In this case, an overloaded AM sends a request to the PAM with the highest PI under its management authority. This latter becomes an AM and joins the DHT-based structure as a successor of its ex-AM. This enables the join process to be quick and use minimum message exchange. Nodes then connect to the new AM using the algorithm described next. Similarly, when the management load decreases, the AM switches back to becoming a PAM in order to release its resources for other purposes. PAMs play another important role: they are queried on-demand by their corresponding AMs to perform certain non time-critical tasks. This allows AMs to focus on real-time management of currently deployed SSONs, such as monitoring the QoE of a live IPTV stream. PAMs also serve as back-up storage of their corresponding AMs in order to avoid losing the management data in case of unexpected AMs failure (which is implausible).

When the overlay network is initially created, the first nodes joining declare themselves as AMs and start forming the DHT-based structure. Newly arriving nodes self-organize and select the AM that will be responsible for managing them. It is possible that initial AMs are not the most powerful nodes in the overlay. In fact, as new powerful nodes (PAMs) join and become more stable, their PI would become higher than current AMs. In order to allow a PAM to take over the management tasks of its AM, we introduce a new metric: the Base Performance Index (BPI). BPI is a fixed score that is calculated from fixed parameters of a node (such as storage capacity, CPU, memory etc…). When a PAM has a higher BPI than its AM, it is only a question of time for its PI to surpass its AM's PI, and hence switch roles with it. The reason we introduce BPI is to lower the overhead of having all PAMs periodically query their AMs for PI updates, in a time where they do not stand a

chance of switching roles. Hence, PAMs with a higher BPI than their AM are the only ones which query their AM for PI updates. A simple request to switch roles with an AM is sufficient for a PAM to be promoted. Alternatively, an overloaded AM simply sends a join request to one of its PAMs to promote it. As previously mentioned, the list of available AMs in the network can be made available through the DHT protocol being used. That way, nodes can easily and quickly fetch this list and select an AM to manage them.

5.4. Problem Formulation

The overlay network consists of two different types of nodes: regular nodes and AMs. Regular nodes are nodes that are incapable of managing themselves and/or other nodes mainly due to hardware constraints, i.e. they do not have enough processing power and resources to run sophisticated algorithms in a timely manner. AMs, on the other hand, are top powerful nodes in the network. An AM is responsible of managing one or more elements (regular nodes) in the network. Hence, every node will be managed by one single AM.

Consider a set of AMs $M = \{M_1, M_2, \dots, M_s\}$ and a set of regular nodes $U = \{u_1, u_2, \dots, u_n\}$, so that $s \ll n$. The $delay$ function $D: (U \times M) \rightarrow \mathbb{R}^+$captures the network delay between a node and an AM. The $error$ function $E: (U \times M) \rightarrow \mathbb{R}^+$ captures retransmission delays due to errors in the link between a node and an AM.

Consider an assignment $\lambda : U \rightarrow M$ that maps each node to a single AM. We assume that each node u that selects an AM m incurs a unit of load on m. Denote the load on m as $\mathcal{L}(m) \triangleq |\{u: \lambda_u = m\}|$. Define a monotonous increasing $processing\ delay$ function, $\delta_m = \mathbb{N} \rightarrow \mathbb{R}^+$, which captures the delay incurred on AM m by every processed node. The different AMs can have different processing capabilities which means that they can have different $processing\ delay$ functions. The service delay $\Delta(u, \lambda)$ of a node u according to the assignment λ is the sum of all delays:

$$\Delta(u, \lambda) \triangleq D\big(u, \lambda(u)\big) + \delta_{\lambda(u)}\big(\mathcal{L}\big(\lambda(u)\big)\big) + E\big(u, \lambda(u)\big). \quad (4)$$

The cost associated with an assignment λ is therefore the maximum delay incurred on a node:

$$\Delta^{Max}\big(\lambda(U)\big) \triangleq max_{u\in U}\,\Delta(u,\lambda). \quad (5)$$

The AM selection problem consists of finding an assignment λ^* that minimizes $\Delta^{Max}\big(\lambda^*(\mathrm{U})\big)$. Such an assignment is called optimal. This problem is NP-hard. Our goal is therefore to derive an algorithm that finds a sub-optimal solution of the problem within acceptable time, while adapting to network changes.

5.5. Autonomic Manager Selection Game

We propose a fully distributed algorithm that is proven to converge to the set of correlated equilibrium as $\rightarrow \infty$. The algorithm is based on a game theoretic approach where actions of each node represent its optimal reaction to its environment and the actions of other nodes. Each node is autonomous and does not need to know other nodes' actions, which introduces very limited exchange of information and reduces the overhead. The algorithm also adapts to the time-varying state of the overlay, including nodes' mobility and churn.

Game theory has been used to construct many distributed algorithms for different fundamental problems. We consider the AM selection problem as a game where players perform a modified regret matching procedure based on their own history [106]. All regular nodes represent players in the game. The available AMs represent different actions of the players. Each action is associated with a payoff the player receives. When a node selects a specific AM to manage it, there is an associated payoff or utility that comes with that choice. The payoff function is mainly made up of two terms: the end-to-end delay and the error rate of the link between a node and its associated AM. The end-to-end delay consists of both the transfer and processing delays. Processing delay models the load imposed on an AM since overloaded AMs will experience a high delay processing queries. The goal of each player is to maximize its payoff, meaning that it should choose the best AM in terms

of minimum delay, error rate and load. The game finishes when equilibrium is reached. Equilibrium means that all players decide to not change their actions in subsequent rounds and stick to a final choice.

In this game, players are denoted by a set $P = \{1,2, \dots, n\}$, where n is the total number of regular nodes. All the available AMs make up the action space of each player, denoted by a set $S^i = \{1,2, \dots, s\}$, where s is the number of the available AMs per player i. However, we decide to limit the action space of players to the top l AMs in terms of end-to-end delay (which we assume is a function of geographical distance). Upon fetching the list of AMs, each player ranks them according to the distance and selects the first l AMs to constitute its set of actions. This decision was made to allow the game to have small and steady convergence times regardless of the number of players and actions.

The game is played in discrete time $t = 1,2, \dots$. Denote by $s_t^i \in S$ the choice of player i at time t. The payoff of player i in period t is denoted $U_t^i \left(s_t^i, s_t^{-i}\right)$. According to [106], the only thing that player i needs in order to play the game is his own payoffs and actions. He does not need to know the game being played, the total number of participating players, their choices and what their payoff functions are. The only information available to each player i is merely his set of available actions S^i to choose from. We construct the following payoff function for each node $i \in P$ at round t according to the constraints defined in [106] and [114]:

$$U_t^i \left(s_t^i, s_t^{-i}\right) = w \times \left(100 - Err_i(t)\right) - Delay_i\,(t). \tag{6}$$

In (6), $Err_i(t)$ represents the error ratio of the link between node i and its selected AM s_t^i in terms of the ratio of packets lost, corrupted or unsuccessfully reassembled as a response to probing the AM. $Delay_i\,(t)$ represents the average two-way delay, w is a weight factor to balance the effect of the two parts, and s_t^i and s_t^{-i} represent the AM selected by node i and the AMs selected by all the other nodes except node i at round t. The proposed function serves two purposes. First, $Err_i(t)$ reflects the quality of the link

between node i and its selected AM at time t. Second, $Delay_i\ (t)$ reflects the quality of the link in terms of end-to-end transmission delay and the load on the selected AM at time t. When an AM is overloaded, it would take it more time to process requests due to the increases in its processing delay. This leads to an increase in the overall value of $Delay_i\ (t)$. Both $Err_i(t)$ and $Delay_i(t)$ can be calculated by node i without requiring information from other nodes, which significantly eases the decentralization of the algorithm and reduces overhead. A simple mechanism to acquire such information is the PING utility. PING does not incur much overhead and provides the round trip time and packet loss rate of the link between two nodes.

5.6. Modified Regret Matching Based Autonomic Manager Selection Algorithm

To play the game, each node performs a modified regret matching procedure [106]. Regret matching simply means that if player i has taken action j at time t, the probability of playing a different action k at time $t + 1$ will be proportional to the "regret for not having played k instead of j". The regret is defined as the increase in the average payoff that would have resulted if the player has chosen action k in the past, each time action j was chosen (and everything else remained the same). The implementation of regret matching requires a player to observe all past actions that were chosen by all the players. The player would also have to compute what his payoffs would have been if his actions in the past were different from what they really were. In [106], the authors show that this level of complexity is not really needed to obtain correlated equilibria. The regret-matching procedure is modified in a way that play probabilities are determined from actual realizations only. A player does not need to know the game he is playing, neither does he need to know his own payoff function nor the others player's payoffs. This procedure is a reinforcement procedure meaning that receiving a relatively high payoff for playing action j at round t will tend to increase the probability of playing the same action in the next round $t + 1$. To summarize, each player plays a repeated game where he only knows his set of available actions and observes his own actual realized payoffs. He knows nothing about other players (neither their actions nor their payoffs). The modified regret matching

procedure is an adjustment to a regret matching procedure [114] introduced in 2000 by the same authors. The goal of the two works is the same: defining play probabilities for the next round based on the regrets of the past history of plays. The original regret matching procedure is modified to allow players to calculate play probabilities based on the previous received payoffs only. Thus eliminating the need for a player to know other players' actions or payoffs, and making it perfect for situations that require no interaction or exchange of information between players. As we previously mentioned, nodes in the overlay represent players and AMs represent the set of actions for each player. Proof that the modified regret matching procedure converges to the set of correlated equilibrium of the game almost surely as $t \to \infty$ can be found in [106].

The average regret of node i from $j \in S^i$ to $k \in S^i$ at round t is defined as follows:

$$Q_t^i(j,k) = max\{C_t^i(j,k), 0\}. \tag{7}$$

where:

$$C_t^i(j,k) = \frac{1}{t} \sum_{\tau \le t : s_\tau^i = k} \frac{p_\tau^i(j)}{p_\tau^i(k)} U_\tau^i - \frac{1}{t} \sum_{\tau \le t : s_\tau^i = j} U_\tau^i. \tag{8}$$

Equation (7) represents an estimate of the average regret of node i at round t for not having selected AM k every time AM j was selected in the past. It measures the difference of the average payoff over the periods when k was selected and the periods when j was selected. p_τ^i denotes the play probabilities at round t, (hence $p_\tau^i(j)$ is the probability that i chose j at round τ); Play probabilities depend only on $U_1^i, U_2^i, \ldots U_t^i$. This means that node i does not need to exchange any information with other nodes. Thus, the overhead of running the algorithm is very low.

Each node i selects AM j at round t, then the probability (9) of switching to another AM k at round $t+1$ is approximately proportional to the average regret from j to k; with the remaining probability (10), the same AM j is selected again. AMs with larger regrets at

Algorithm 1
AM Selection Algorithm Based on Modified Regret Matching for Node $i \in P$

1: fetch the l closest AMs from the list of AMs in the DHT-structure
2: randomly select an initial AM s_1^i
3: set $p_1^i(j) = \frac{\delta}{m^i}$ for every $j \in S^i$
4: **for** each round $t = 1,2,$ ... **do**
5: update the list with the l closest AMs
6: remove old AMs from S^i
7: set $p_1^i(j) = \frac{\delta}{m^i}$ for new AMs in S^i
8: set $err = 0, delay = 0$
9: **while** round t is not over
10: probe AM s_t^i
11: update err and $delay$ based on probe reply
12: end while
13: use equation (6) to compute $u_t^i(s_t^i, s_t^{-i})$
14: use equations (7) and (8) to compute $Q_t^i(j,k)$
15: use equations (9) and (10) to compute p_{t+1}^i
16: use play probabilities $p_{t+1}^i$to select an AM for the next round
17: end for

the current round will be selected with a higher probability at the next round. The regrets will be reduced over time, as will the average regret of any two AMs for each node.

$$p_{t+1}^i(k) = \left(1 - \frac{\delta}{t^\gamma}\right) min\left\{\frac{1}{\mu} Q_t^i(j,k), \frac{1}{m^i - 1}\right\} + \frac{\delta}{t^\gamma m^i}. \quad (9)$$

$$p_{t+1}^i(j) = 1 - \sum_{k \in S^i : k \neq j} p_{t+1}^i(k). \quad (10)$$

where $j = s_t^i$ is the choice of node i in period t. At $t = 1$, we choose p_1^i arbitrarily such that $p_1^i(j) \geq \delta / m^i$ where m^i represents the number of actions of i. Other variables are

chosen as follows [106]: $0 < \delta < 1$, $0 < \gamma < 0.25$, $\mu > 2M^i(m^i - 1)$ for all i, where M^i is an upper bound on $|u^i(s)|$ for all $s \in S$.

In [106], S. Hart et al. prove that when the game stabilizes, players do not regret their dynamic selection of actions according to the modified regret-based matching procedure. In our case, each node attains a certain reasonable payoff (i.e., the lowest error ratios, the lowest end-to-end delay and a load-balanced AM) against the previous variations of nodes' movements, network topology and AM load.

Overall, the local sub-optimal performances of all nodes lead to significant network performance. With no coordination or cooperation between nodes, the modified regret matching based algorithm achieves the best result possible.

5.7. Simulations and Results

5.7.1. Discovering and Promoting AMs

5.7.1.1 Simulations Setup

We performed simulations using OverSim [115], an open source overlay and P2P network simulator based on the discrete event simulator OMNET++ [116]. Chord was used as the DHT-protocol to store and retrieve data because of its efficiency in lookup times. We extended the original Chord protocol and call it AutonomicChord in our simulations. A sample application based on simple request-response type messages was used utilizing the KBR protocol [117]. KBR provides efficient routing to keys from a large identifier space. In the simulations, we varied the size of the network from 18 to 1200 nodes, assuming that 20% of the nodes are BPDs with a mean life time of 100s and 30% of the nodes are AMs. The rest of the nodes are PAMs on standby mode, which get deployed when the load increases. Simulations run for a total of 1000s. The same Chord settings were set for AutonomicChord.

5.7.1.2 Results and Analysis

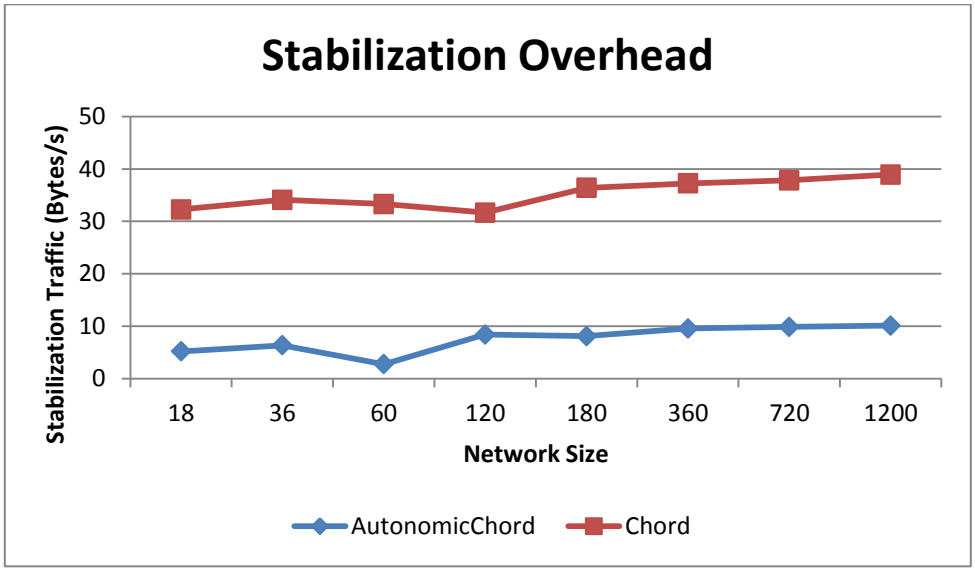


Figure 15: Stabilization Overhead

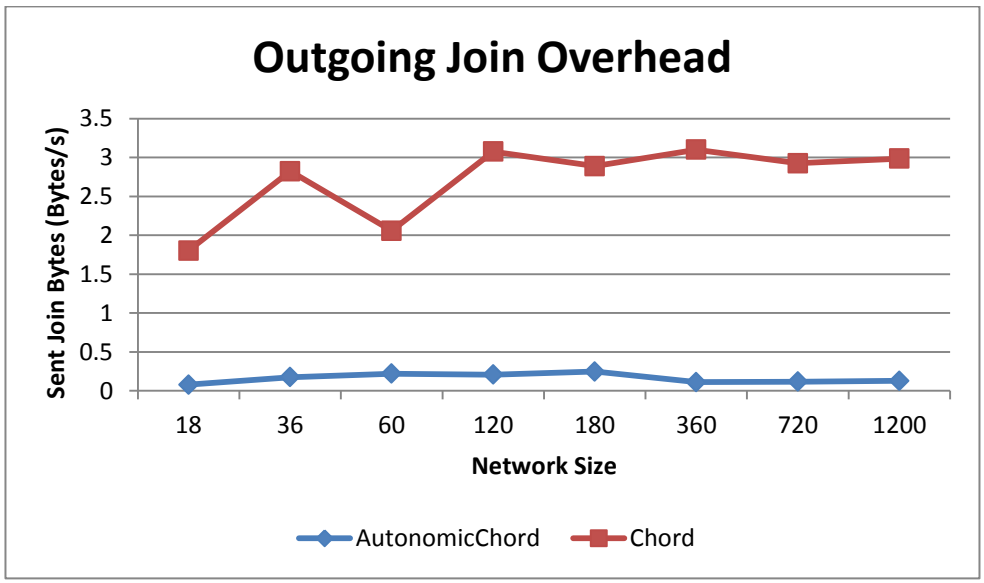


Figure 16: Join Overhead

Figure 15 and Figure 16 compare the stabilization and join overhead of both protocols. Since AMs in AutnomicChord are stable nodes with very low churn probability, the network requires very small message exchange to stabilize. Finger table, successor and predecessor entries are rarely altered thanks to nodes' stability. However, when unstable BPDs are made part of the overlay structure as in Chord, they break the structure when they leave, causing more message exchange to stabilize the structure.

Joining a Chord structure with N nodes and K keys requires O(K/N) keys exchange and results in extra join overhead. The same applies to AutnomicChord. However, the exclusion of BPDs and the on-demand introduction of PAMs in the proposed overlay results in a significant decrease in the join overhead as illustrated in Figure 16.

Figure 17 presents a comparison of the mean sent/received overlay maintenance traffic between nodes in both networks. For Chord, the high number of nodes included in the network increases the number of messages that need to be exchanged to maintain links to the successor, predecessor, and fingers. Moreover, BPDs cause all these entries to be invalidated as they die and leave the network, which in its turn results in more stabilization

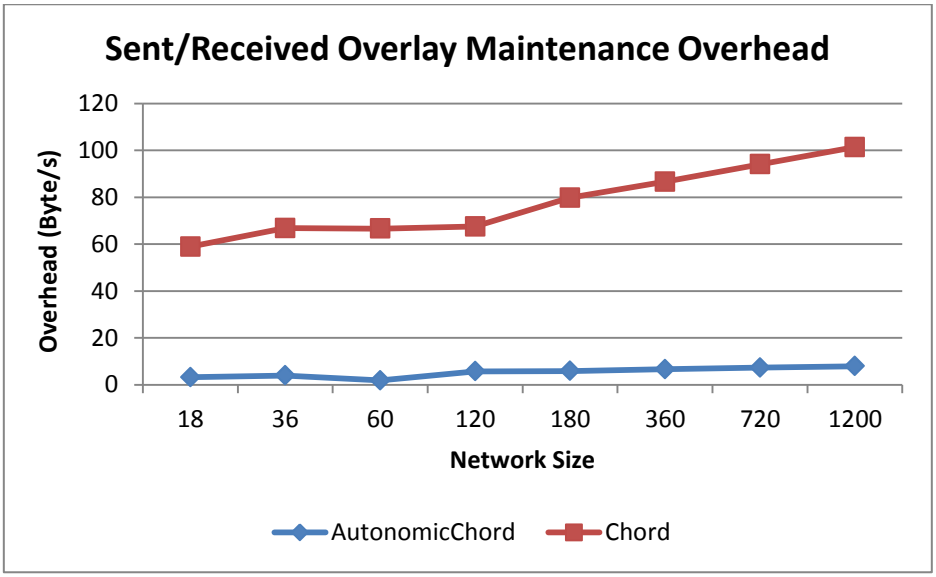


Figure 17: Mean Sent/Received Maintenance Overhead

delay and overhead. The reduced number of nodes (AMs) present within the AutonomicChord structure, as well as their stability helps keep successor, predecessor and finger entries intact. This explains the very low maintenance overhead experienced by AutonomicChord compared to the normal Chord.

Finally, Figure 18 shows the mean one-way hop count, which is the mean number of hops it takes a message to reach its destination. We see a slight improvement of AutonomicChord over Chord due to the fact that the total number of nodes in AutonomicChord is reduced to the needed AMs, whereas in Chord all nodes form the structure which leads to a waste of resources in low load scenarios.

Using the proposed scheme, overlay nodes self-organize in an autonomic manner to form a DHT-structure composed of powerful, stable AMs. Each regular node then queries the DHT for the list of AMs present in the network and performs the proposed game theoretic approach to select a manager that will be responsible to manage it. We evaluate the performance of the proposed AM selection algorithm next.

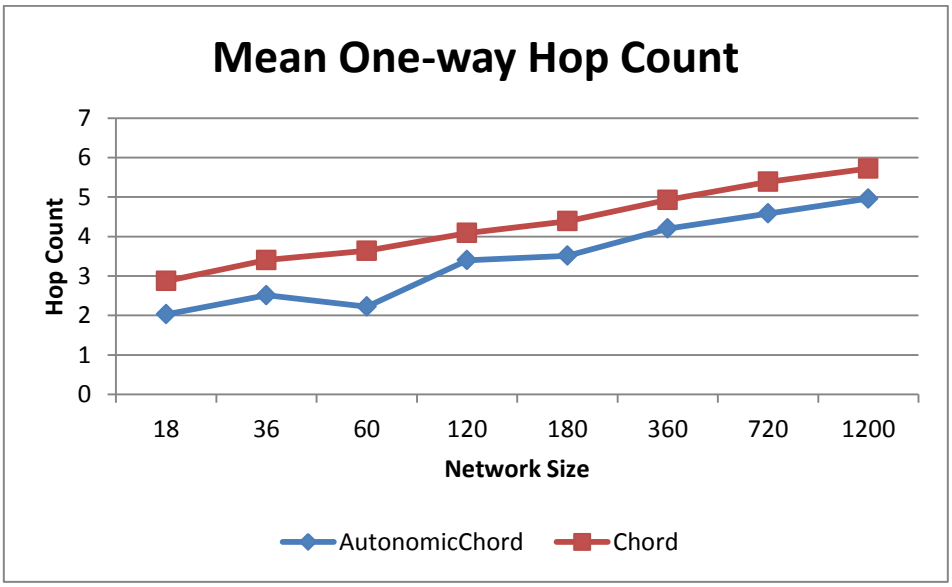


Figure 18: Mean One-way Hop Count

5.7.2. Autonomic Manager Selection

5.7.2.1 Simulations Setup

OverSim [115] was also used to model overlay networks and was extended to implement our algorithm. We compared our scheme to a simple solution where each node selects the AM that is closest in terms of geographical distance (which we assume is directly proportional to end-to-end delay).

The network setup places nodes in a 2-dimensional Euclidean space. Delay between any two nodes is assumed to be proportional to the distance between them. The arrival and departure of nodes from the network was modeled using a lifetime-based churn generator based on a uniform distribution. Nodes are deployed randomly (given random coordinates) in a 200x200km field. Nodes' mobility consists of changing their coordinates to new random values. Churn and mobility take place every 10 seconds when enabled unless otherwise specified, while simulations run for 1000 seconds. Simulations were repeated 30 times and the average as well as the 95% confidence levels was recorded. During each simulation run, AMs were randomly chosen among the available set of nodes. The percentage of AMs present in the network was set to 20% that of the total number of nodes in the network and the parameter l to 10, unless otherwise specified. We chose the following parameters for regret calculation: $w = 0.01$, $\delta = 0.9$, $\gamma = 0.24$ and $\mu = 2\left(m^i - 1\right)$.

We evaluate the proposed algorithm in terms of four metrics: the load gain, equilibrium time, error ratio gain and delay gain. Load gain represents the standard deviation of the load on the AMs. A perfectly load balanced system has zero standard deviation. The load gain corresponds to the difference in the load distribution between the proposed algorithm and choosing the closest AM solution. Equilibrium time is the time, in terms of game rounds, it takes the algorithm to stabilize. Note that for the "cloest AM

solution", nodes simply fetch the list of AMs from the DHT and select the closest AM (in terms of distance, that we assume models end-to-end delay) to be its manager. Error ratio gain and delay gain represent the increase achieved by the algorithm in terms of link error rate and end-to-end delay between a node and its corresponding AM compared to choosing the closest AM.

5.7.2.2 Scalability

In this section, we evaluate the performance of the algorithm in terms of network size. Basically, we would like to study if the algorithm scales well to larger networks. Figure 19 illustrates the performance of the proposed algorithm compared to the simple solution of choosing the closest AM in terms of load gain for the three considered scenarios. Results clearly show that the proposed algorithm outperforms the closest AM solution by more than 50% gains. We also note than it does so under node churn and mobility. Another important aspect is that load gains are almost the same for different network sizes meaning that the algorithm scales well in terms of load distribution. When choosing the closest AM in terms of distance (end-to-end delay), AMs that are located in

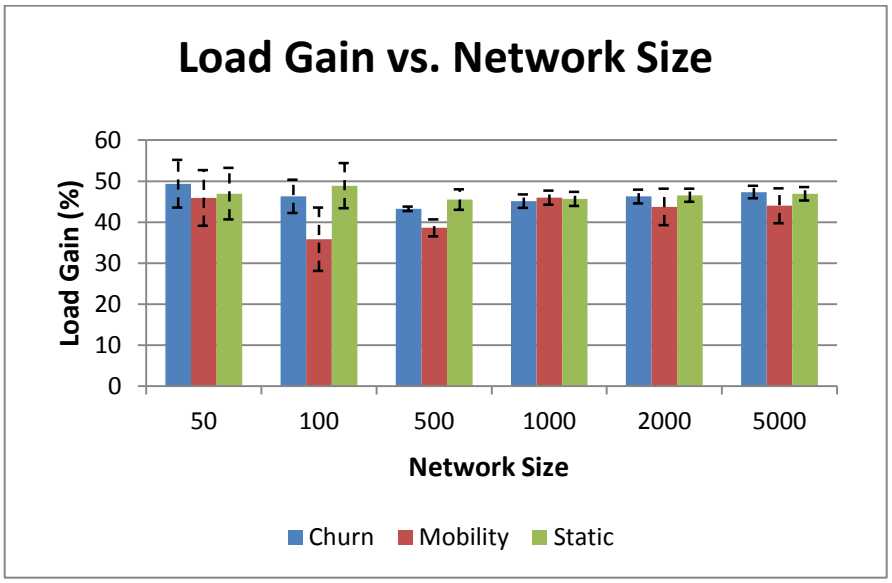


Figure 19: Load Gain vs. Network Size

highly dense areas will end up serving a high number of nodes. This will cause the load to be unevenly distributed among AMs in the network. On the other side, using the proposed algorithm enables nodes to select an AM based on three criteria: error ratio, end-to-end delay, and load. As more nodes select a specific AM, its processing delays goes up which causes its probing replies to be delayed. As a consequence, the nodes' payoffs of choosing that AM decreases, increasing their probability of switching to another (hopefully less loaded) AM in the next round. As the game proceeds, the load on the AMs gets evened and that explains the high gains achieved in Figure 19.

Figure 20 and Figure 21 show the error and delay gains, respectively, achieved using the proposed algorithm. For the case of delay, we see that the algorithm achieves high gains that go up to 28%. When choosing the closest AM, nodes tend to minimize the end-to-end delay of communication between them and their chosen AM. However, the high number of nodes that connect to an AM add extra processing delays which affect the communication link between nodes and their corresponding AMs. Although choosing the closest AM is simple and straightforward; it is not always beneficial as the results show,

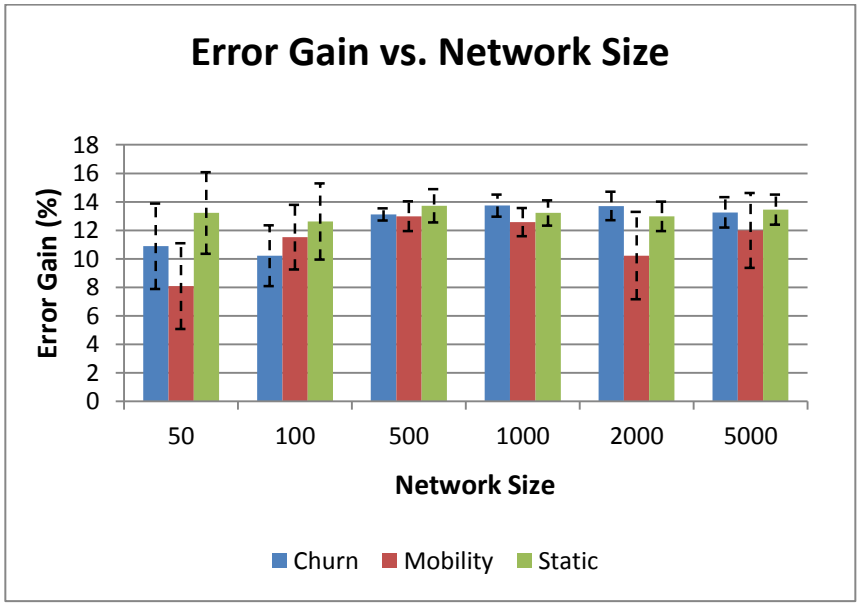


Figure 20: Error Gain vs. Network Size

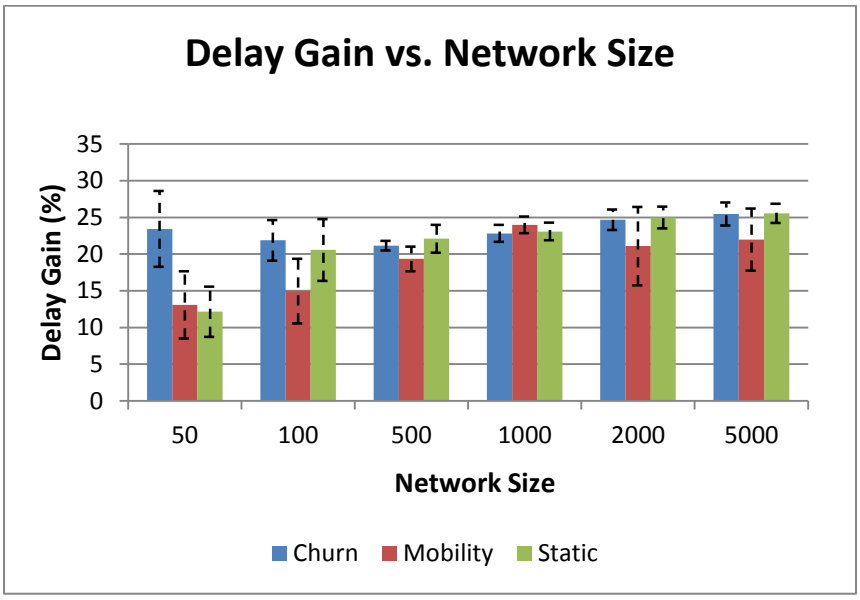


Figure 21: Delay Gain vs. Network Size

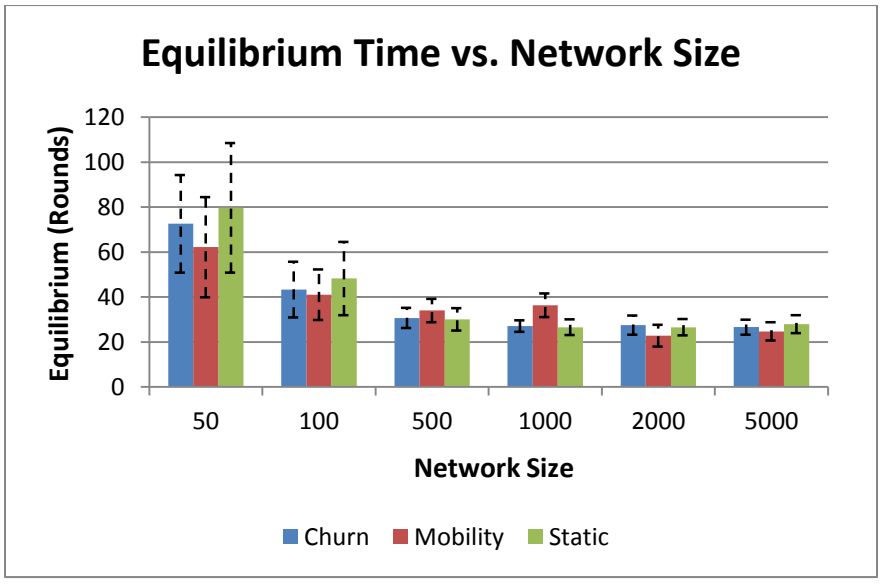


Figure 22: Equilibrium vs. Network Size

especially in highly dense areas. Concerning error gains, results show that the algorithm achieves gains ranging from 5% to 16% in different network scenarios. Again, we clearly

notice that the algorithm scales well for large networks and that the gains are almost the same for all network sizes.

Figure 22 shows the equilibrium time of the algorithm as a function of network size. The equilibrium time, or convergence time, is the number of rounds it takes nodes to choose an AM and refrain from changing their decision in future rounds. We notice that it takes the algorithm more rounds to converge for smaller networks while it takes a less, almost constant, number of rounds as the network size grows. This behavior can be explained by the fact that more AMs are present in larger networks, giving nodes more choices and helping them diverge from choosing the same AM in a small amount of time. This increases the payoff of nodes and makes the algorithm converge quickly compared to smaller network sizes. Figure 22 also shows that the algorithm adapts well in the presence of mobility and churn. Assuming each rounds counts for one second, the algorithm takes 100 seconds in the worst case to converge, while it takes only 18 seconds even for a large network. In a highly dynamic network where nodes move, join, and leave at will, the algorithm should be run continuously in order to cater for these changes. However, in more static scenarios, it is safe to stop the algorithm after a node keeps selecting the same AM for a number of rounds. Our simulation results show that nodes can safely stop the algorithm when their AM selection does not change for at least 10 consecutive rounds. The overhead introduced by scalability consists of added message exchange between nodes and AMs which we study in the following section.

5.7.2.3 Percentage of Autonomic Managers

The number of AMs present in the network at any certain time depends on both the number of nodes and the number of SSONs in operation. As management load increases, more nodes will be invited to take part of the management process. To study how the proposed algorithm behaves in such cases, we perform the same set of simulations for 500 nodes while varying the percentage of AMs present in the network. The results are drawn in Figure 23, Figure 24, Figure 25 and Figure 26. As the percentage of AMs increases, we

notice that the gains as well the equilibrium time decrease. This behavior is expected. When there are few AMs in the network, nodes will experience a low payoff regardless of the AM they choose because each AM will have a high number of nodes connected to it. As the game proceeds, nodes will be switching AMs in each round hoping to find a less overloaded one. It will take some time for a node to realize that all AMs are overloaded and that it should select the best one it can find. This also explains the high gains achieved under such circumstances. On the other hand, as the percentage of AMs increases, the number of AMs surpasses the number of nodes. This means that not only AMs will not be overloaded, but some of them will not even be managing any nodes. That is why it takes only very few rounds for the algorithm to converge in these cases. The same analysis is true for gains which are also decreasing as the percentage of AMs increases.

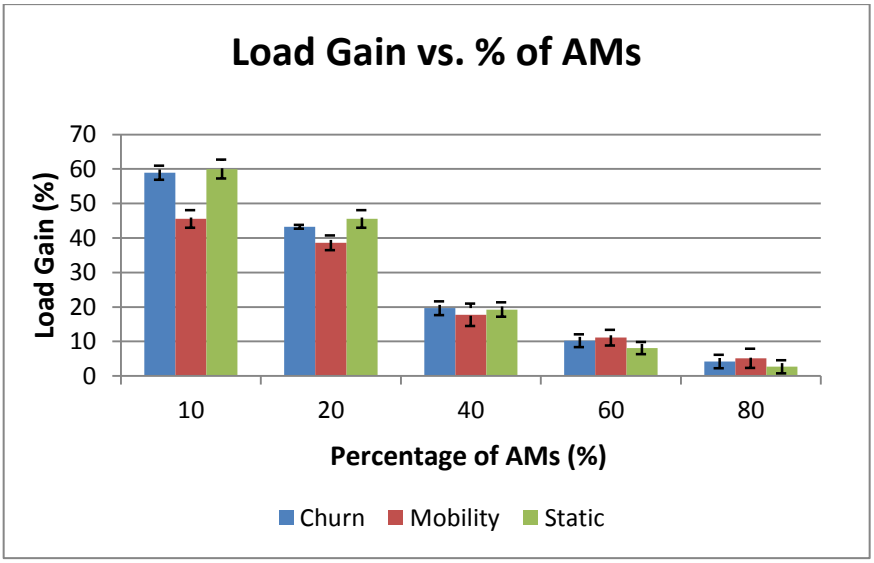


Figure 23: Load Gain vs. % of AMs

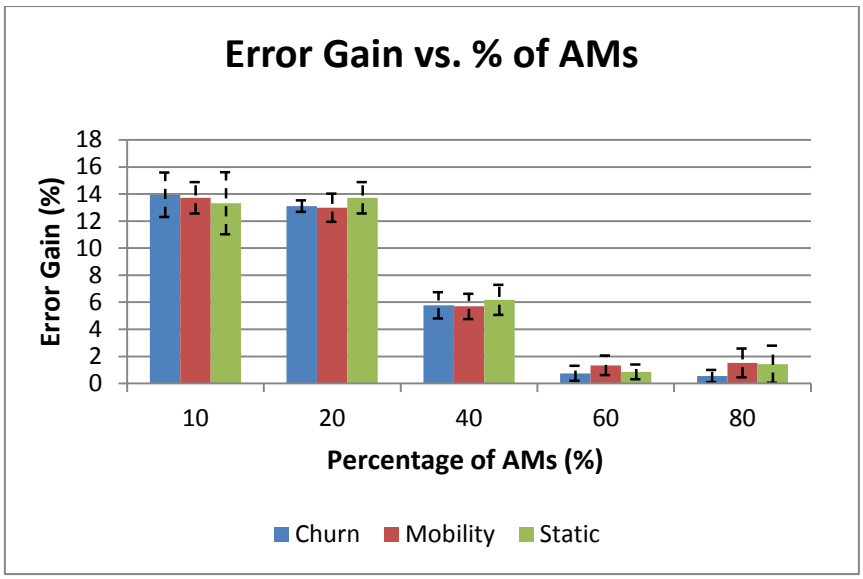


Figure 24: Error Gain vs. % of AMs

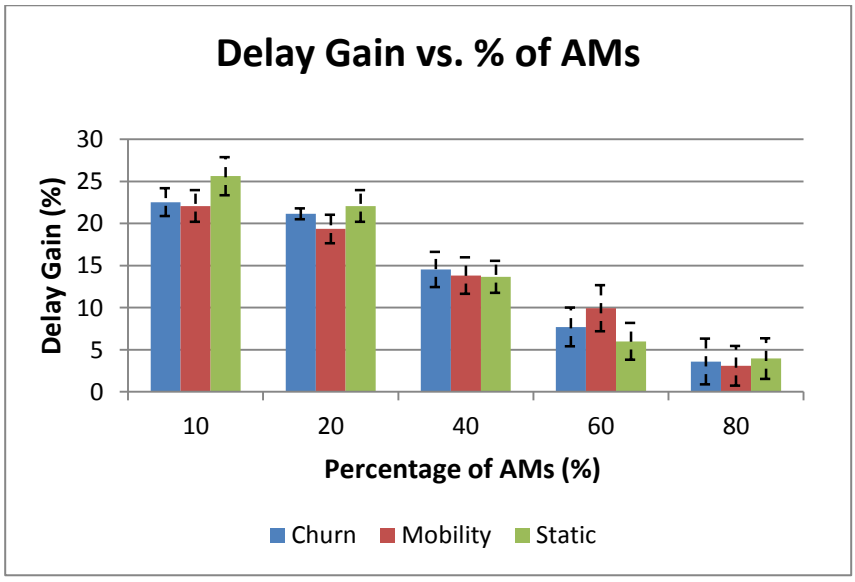


Figure 25: Delay Gain vs. % of AMs

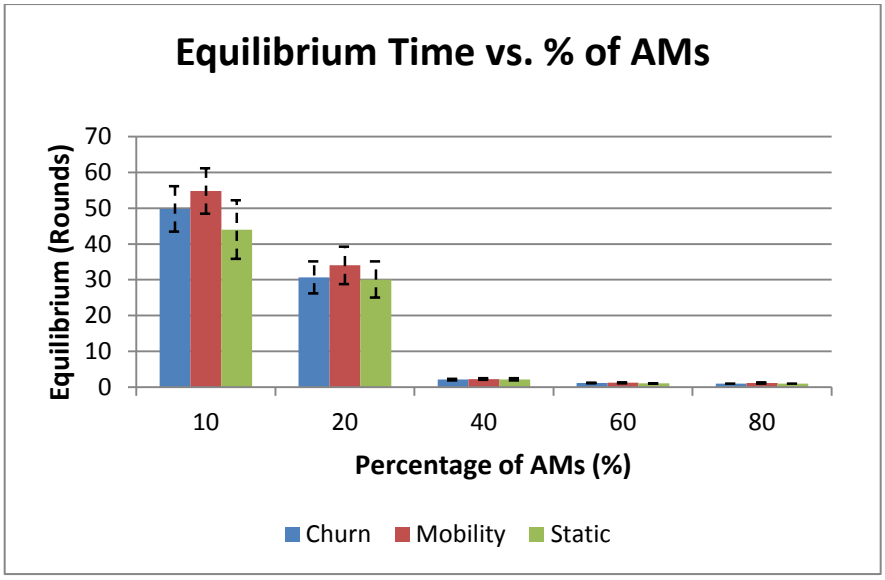


Figure 26: Equilibrium vs. % of AMs

5.7.2.4 Churn and Mobility Rate

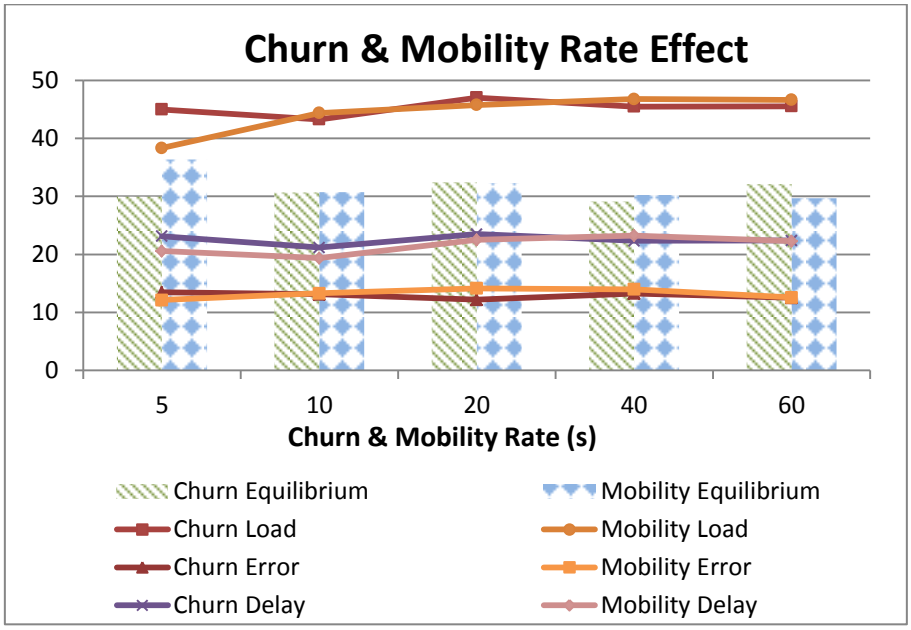


Figure 27: Churn and Mobility Rate Effect

In Figure 27, we study the effect of churn and mobility rates on the performance of the algorithm. Simulations of a network of 500 nodes were run using the same default values while the churn and mobility rates were modified from 5 seconds to 90 seconds. A randomly chosen node is created, removed, or moved to a random position every rate period. The figure shows the average of the 30 simulation runs. As the results illustrate, the algorithm performs well under different churn and mobility rates. We notice very slight differences in the gains achieved as well as the equilibrium time in both scenarios.

5.7.2.5 Size of Actions Set

In order to study the effect of the parameter l, the size of the actions set, on the performance of the algorithm, we carried simulations with 500 nodes and 20% AMs while varying l. In the proposed algorithm, the parameter l represents the number of AMs that represent the set of actions of a node. Although the network might comprise of say 100 AMs, we propose that nodes do not need to include all AMs in their actions set. In fact, as Figure 28 shows, doing so causes the algorithm to converge slowly and does not incur additional gains in load, delay and errors. We calculate l as a function of the percentage of

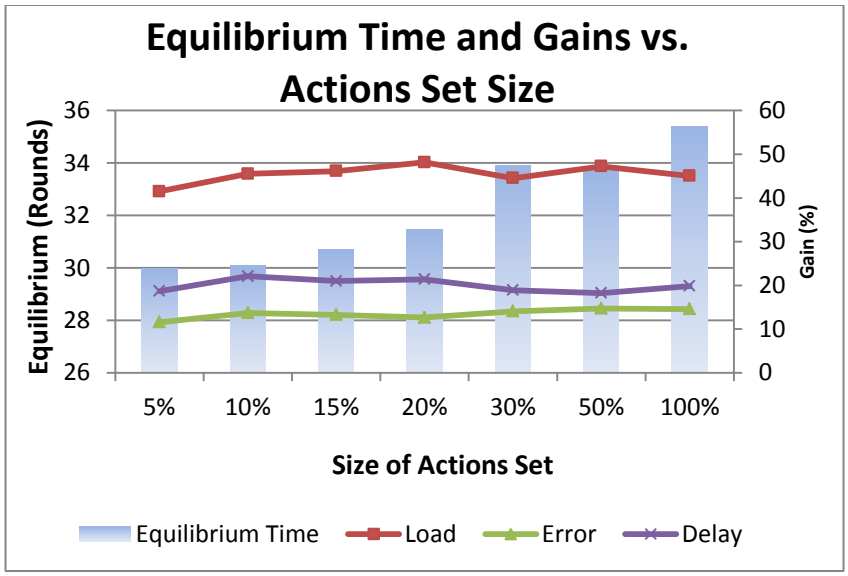


Figure 28: Actions Size Set Results

AMs in the network. In Figure 28, we see that the best gains are achieved when only 10% of the AMs make up the actions set. In fact, when including less than 10% AMs, nodes narrow their choices of selecting a better AM which might be far away but might have better error rates and load. On the other hand, when too many AMs are included in the actions set, it takes the algorithm more rounds to converge because it has a higher probability to switching to new AM as time goes by. We conclude that the action set size l should be set to 10% the number of AMs present in the network, requiring nodes to store only a few number of entries to run the algorithm. As mentioned earlier, the total number and list of AMs present in the network can be fetched by querying the DHT protocol in use.

5.7.2.6 Overhead of the Algorithm

Figure 29 draws the overhead generated by the algorithm as a function of the percentage of AMs in the network. We assume that PING is used to measure links' characteristics between nodes and AMs and a PING request holds 32B of data. We note that for a network size of 10000 nodes with 10% AMs, the overhead is less than 280 KB/s. As the percentage of AMs increases, the overhead decreases and is less than 1 KB/s for a small network of 100 nodes with 80% AMs. Note that increasing the game round period

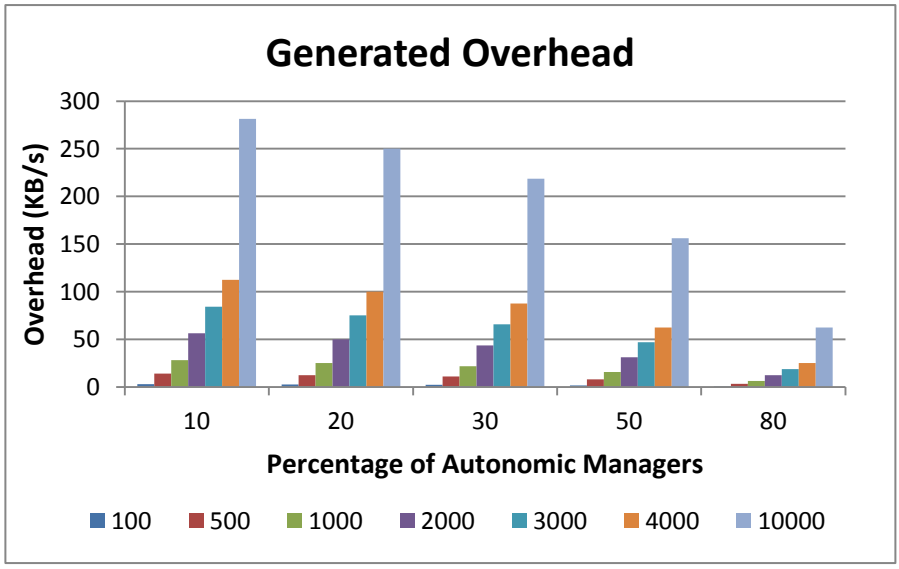


Figure 29: Algorithm Overhead

from 1 second to x seconds also cuts down the overhead by x times. For example, if the game is played in rounds of 2 seconds, the overhead decreases to 140 KB/s for a network of 10000 nodes and 10% AMs.

5.7.2.7 Optimal Solution

In Figure 30, we compare the proposed algorithm and the solution of choosing the closest AM to the optimal solution. The optimal solution was calculated using IBM's ILOG CPLEX Optimizer [118]. We run 10 simulations of a static scenario of 500 nodes and 20% AMs, and calculate the difference between the optimal solution and our proposed algorithm, as well as choosing the closest AM solution. We notice that in some runs the proposed algorithm is only 14% far from the optimal solution. In all simulation runs, our proposed algorithm performs better than selecting the closest AM and outperforms it by a 30% difference. We also note that our proposed algorithm is always more than 74% closer to the optimal solution, which supports our previous statement that the proposed algorithm achieves sub-optimal results with limited overhead and interaction.

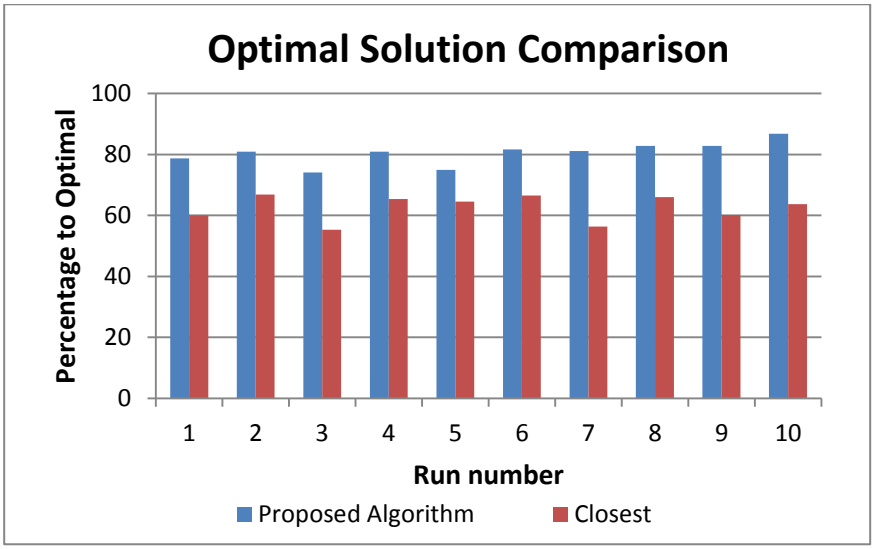


Figure 30: Optimal Solution Comparison

5.8. Summary

In this chapter, we proposed an AM election scheme and a self-organizing algorithm to ease the management of SSONs. We propose the use of DHTs to store the gathered information for efficient retrieval. Due to technological constraints, we argue that some nodes will not have the necessary capabilities to perform the self-management functionalities. Hence, they must rely on other, more powerful, nodes to manage them. The decision on which AM will manage which nodes is important as management data should be exchanged reliably and in a timely manner, under the dynamic nature of overlay networks. First, we proposed a scheme to select AMs among the set of available nodes. Stable nodes with higher processing capabilities are promoted to AMs to take part of the management process while other nodes perform a regret-based procedure to select an AM. The proposed fully distributed algorithm enables nodes to choose the best available AM in terms of delay, load and error ratio. The algorithm has the property of self-adapting to constantly changing network conditions, and introduces very limited overhead. Using extensive simulations, we show that the proposed algorithm outperforms the simple solution of choosing the closest AM, and balances the load among AMs. We also studied the performance of the algorithm under different scenarios of churn and mobility. Results show that the algorithm achieves high gains compared to the closest AM selection solution, and reaches equilibrium in reasonable times while introducing very limited exchange of data in the network. Finally, we compare the performance of the proposed algorithm to the optimal solution and notice that it gets 86% closer to the optimal solution.

Chapter 6

A Utility Function for Predicting IPTV QoE Over an Overlay Network based on Losses and Delays

6.1. Introduction

Different ways exist to deliver digital TV services to customers. However, satellite and cable networks are switching to transporting TV over the IP-based communication network. IPTV offers more services than the traditional linear-programming TV service, due to its ability to address users individually. These new services include time-shifted TV, network personal video recorder, VoD among others. Unfortunately, IP was designed to provide best effort service meaning that the network does not guarantee data delivered or a given QoS level or priority. Proposed protocols for QoS support over IP networks, such as IntServ and DiffServ, have not seen wide acceptance mainly because they require changes to all routers in the network or are not scalable. Moreover, the introduction of a new service into the current infrastructure without or with minimal changes to the existing infrastructure poses a challenge to IPTV service providers. As a result, IPTV streams might experience degraded quality along the communication path during the lifetime of the media session. To overcome these problems, overlay networks were proposed. An overlay network is a virtual network made up of virtual nodes and logical links built on top of an existing network infrastructure. Overlays can be used to increase reliability and robustness in routing, to offer QoS guarantees and to provide a wide range of new services [4]. They are designed independently, usually extend to more than one Internet Service Provider (ISP), and experience dynamic changes to the network topology and conditions. Even with the use of

overlays, the quality of an IPTV stream must be monitored to ensure satisfied customers and avoid customer churn.

Assuring QoE for IPTV users has become a top priority for IPTV service providers. To keep the business going, IPTV service providers must monitor the QoE of their users and ensure that it does not degrade to unsatisfactory levels. Since IPTV service providers deliver their services using overlay networks, they cannot access the physical infrastructure. The only option they have is to monitor QoE at the application layer by setting up appropriate sensors in relevant points of the overlay to gather important data and analyze it.

In this chapter, we propose a utility function that maps statistical losses and delays at the application layer into QoE. We use IPTV as the service running on the network because of its wide use and the impact of video quality on customer satisfaction. Our utility function provides a way to predict the user experience and has a tunable parameter to control the user degree of sensitivity to losses/quality degradation. It is essential to point out that the overall user QoE depends on many metrics, some of which are not directly related to the quality of the multimedia content (such as freshness, start-up time and response time) [119]. However, such metrics are not studied in detail in the networking area and are difficult to quantify. On the other hand, many researchers focused on the impact of QoS parameters on QoE [120], [121]. We decided to use byte loss rate and delays as QoS criteria because jitter can be compensated for using a buffer at the receiver.

Figure 31 represents the IPTV monitoring architecture for overlay networks. The node architecture is applied to both the Video Service Office and Set-Top Box depending on where the adaptation will take place. For instance, if the VSO is responsible for performing the adaptation, the STB only gathers the amount of received bytes and the delay, and reports them back to the VSO. In this case, the VSO measures the sent bytes for each STB and stores all the information in the knowledge base. Utility is then calculated for each STB and a check for QoE requirements violation is done. The VSO can use any

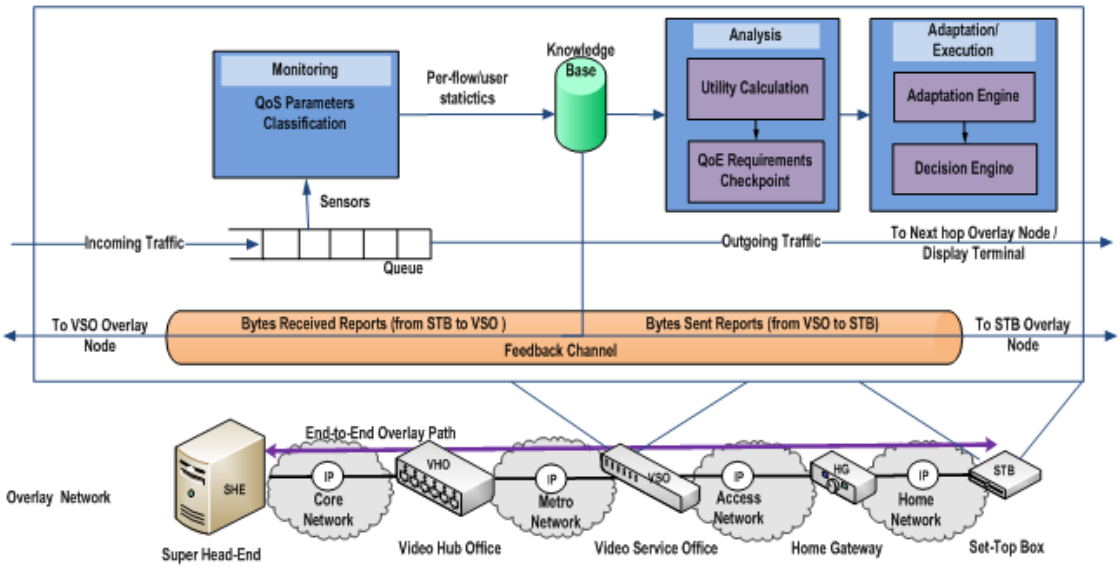


Figure 31: IPTV Overlay QoE Monitoring Architecture

adaptation scheme to increase the quality of the video (e.g. change codec) when it goes below unacceptable values. On the other hand, when adaptation is to take place at the STB, the VSO measures the sent bytes and sends them to different STBs. The STBs in this case calculate the utility, predict the QoE and adapt accordingly if needed. Note that any other network entity can perform this task given that gathered statistical data is made available to it.

6.2. The Theory of Utility Function

6.2.1. Utility Functions Review

Utility theory was originally proposed by Von Neumann and Morgenstern [122] as a way to measure investors' preferences for wealth and the amount of risk they are willing to assume in the hope of attaining greater wealth. It describes the "want satisfying" capacity of a commodity [123], identified by utility functions, each with different properties and characteristics. As a simple example, let X be the set of all possible consumable packages. The consumer's utility function U(X) assigns a satisfaction score to each package in the set. If $\mathrm{U(x)} > U(\mathrm{y})$, then the consumer clearly favors x to y.

Utility refers to satisfaction only. Economists distinguish between two main classes of utility: cardinal and ordinal. Cardinal utility measures utility with specific numbers that provide an ethically or behaviourally significant quantity. Ordinal utility compares utility as a result of the consumption of two sets of goods; ranking is the only thing that matters. The relative satisfaction between two states is expressed with a continuous utility function. Utility functions have been used in the literature for different purposes. In the next section, we briefly describe some of its important uses.

6.2.2. Classification of Utility Functions

A number of utility functions have emerged since Scott Shenker [124] introduced their use in 1995 for admission control in the Internet. He discusses the different shapes of utility functions needed for elastic, hard real-time, delay-adaptive and rate-adaptive real-time applications. His work inspired many researchers who proposed different mathematical functions in order to calculate the utility of various types of traffic as a function of bandwidth [125], [54], [126] , [127] and [128]. Meshkova et al. [129] use utility functions for network management. Different types of functions are proposed for class-based queuing and power-aware load-balancing. Xiao et al. [130], Nguyen-Vuong et al. [131] and Pal et al. [132] use a sigmoid utility function to achieve other goals such as power control, access network selection in heterogeneous wireless networks, and admission control and bandwidth allocation schemes that measure user satisfaction. Finally, Fielder et al. use a linear function in [133] and [134] to measure the utility of network throughput in the presence of losses. In general, utility functions can be classified as linear, logarithmic, exponential and sigmoid. More details about the characteristics of and differences between these types can be found in [131].

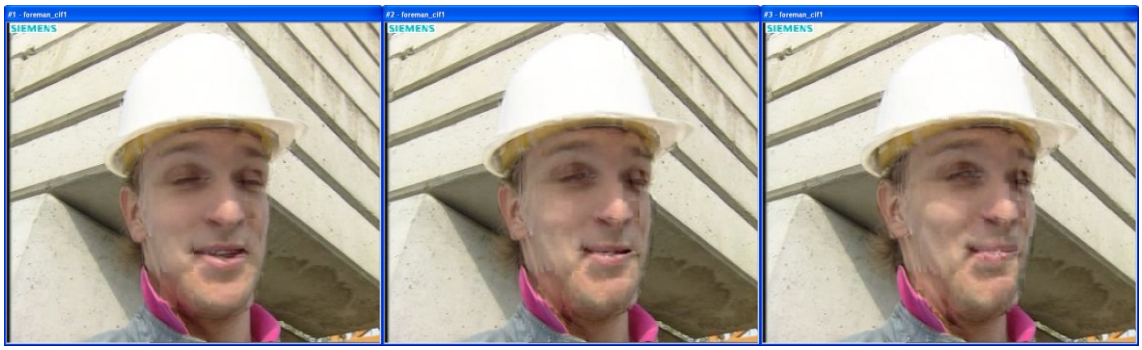
As discussed in the previous section, utility-based microeconomic models have been applied in different fields to model the customer's decision-making process. The use of utility functions is just a mathematical way of saying that we are trying to attain certain goals. Our goal is to achieve a high end-user QoE. We use a sigmoid utility function in our

work because it estimates the user's satisfaction with respect to the perceived QoS [135] as we show mathematically in the next section.

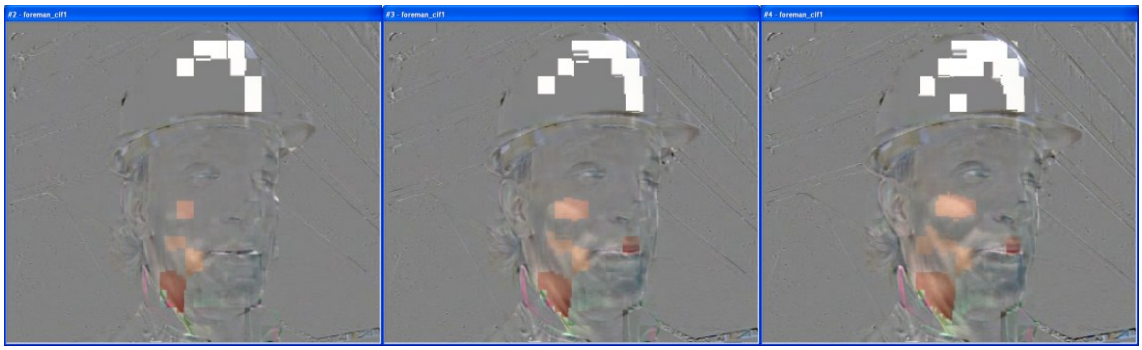
6.3. Utility Function for Predicting QoE

It is essential to distinguish between upward and downward criterion utility functions. An upward criterion function means that the utility increases as the value of that criterion increases. With a downward criterion, the utility decreases as the value of the criterion increases. In our work, we consider only the downward function, since our criteria are byte loss rate and delay.

There are many reasons for choosing byte loss ratio and delay (at the application layer) as opposed to packet loss ratio and packet delay as a criteria. For instance, when packets belonging to an intra-coded frame (I-frame) are lost, the receiver might be unable



0 packet lost in I-frame | 0 packet lost in P-frame | 1 packet lost in P-frame

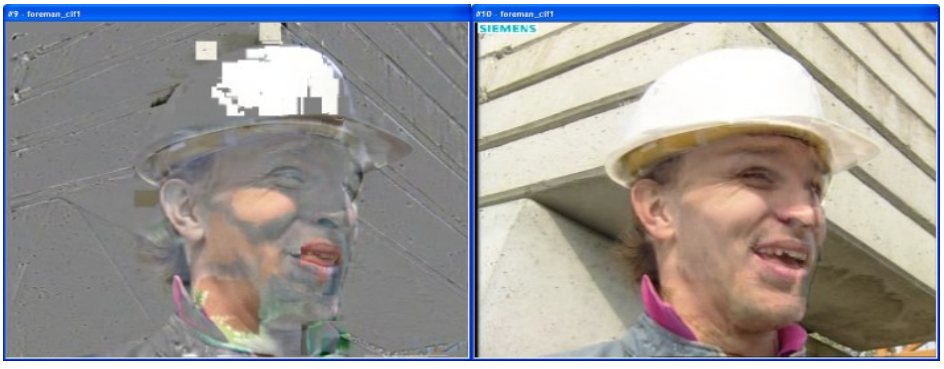


1 packet lost in I-frame | 0 packet lost in P-frame | 0 packet lost in P-frame

Figure 32: Effect of packet loss on video quality

to decode the whole group of pictures (GoP) and not only the erroneous I-frame (because of inter-dependencies among frames). From a receiver's point of view, the amount of data lost in this case is more than just a single packet of a frame but rather the whole GoP as Figure 32 illustrates. Moreover, IPTV service providers do not have access to the underlying network infrastructure to gather packet-level statistical information. Finally, the use of error-correction schemes at different layers can cancel the effects of packet loss at the receiver and therefore using packet loss ratio as criterion would produce inaccurate results. For example, when some packets in an I-frame are lost, they might be recovered using redundant data or any application-level error correction scheme. In this case, no loss will be recorded at the application layer and the frame will be decoded successfully. Since our approach makes use of error rate at the application layer, it explicitly takes into account error correction schemes by using byte loss ratio after any error correction schemes have been applied.

Figure 33 shows the effects of packet delay on video quality. In both cases, (a) and (b), a single packet experiences the same amount of delay. However, the effect on the quality is tremendous. This is because when a packet is delayed, it causes the frame to which it belongs to be delayed; if this packet happens to belong to an I-frame, all frames in the GoP will be delayed since they all depend on an error free I-frame to be successfully decoded. On the other hand, when a packet that belongs to a P-frame is delayed, a smaller



(a) packet belonging to an I-frame delayed (b) packet belonging to a P-frame delayed

Figure 33: Effect of packet delay on video quality

number of frames are affected since only a subset of the GOP depends on the P-frame. Hence, using packet delays as a measure to predict QoE could lead to making inaccurate conclusions.

6.3.1. Mathematical Analysis

Table 1 defines mathematical symbols used herein. Consider a utility function of a downward parameter $0 \leq x_i < \infty$ where x_i is the amount of the monitored resource. Due to technological constraints and/or user's preferences, this parameter will always have upper and lower limits and so will the utility ($x_\alpha \leq x_i \leq x_\beta$). It is more convenient and meaningful to scale the utility to the interval $[0,1]$ so that its value could be expressed as a percentage. According to [131], the utility level should not change drastically given a very small change to a criterion value (product's characteristic) and the marginal utility should be regular. The utility function should therefore be twice differentiable on the interval $[x_\alpha, x_\beta]$. Given that our utility is for a downward criterion, an increase in the value of x_i

Table 1: Table of Notations

Symbols	Meanings
x_α	value below which user QoE does not change
x_i	criterion
x_m	inflection point
x_β	value above which user QoE does not change
$u(x)$	utility of x
ζ	sensitivity parameter
D_i	delay index
$missedFrames$	number of frames that missed their playback deadline
$totalFrames$	total number of received frames
$Delay$	average delay of displayed frames
$Buffer$	receiver buffer size

implies a lower user utility, with the utility function as a decreasing function of x_i. However, when x_i goes below a certain threshold and the utility becomes close to 1, user behavior is indifferent to the decrease of x_i. This means that the improvement of utility disappears for a decrease of x_i if the latter is less than the minimum required. Similarly, the decline of utility disappears when x_i reaches a certain threshold after which user satisfaction does not change. To illustrate with an example, consider the case of video quality in terms of byte loss rate. If the user is experiencing a MOS score of 5 with an loss rate of $x\%$, then further decrease in the loss rate to a value less than $x\%$ will not change the way the user perceives the quality (which for the user is always excellent). In terms of utility, further decrease in error rate in this example does not yield an increase in the utility for the user. This behaviour follows the law of diminishing marginal utility, i.e., $lim_{x_i \to x_\alpha} u^{'}(x_i) = 0$. The effect of diminishing marginal utility implies the convexity of $u(x_i)$ for x_i greater than a given value x_c and its concavity for x_i less than a given value x_v. These requirements for a decreasing utility function can be mathematically expressed as follows:

$$u'(x_i) \leq 0. \tag{11}$$

$$\exists x_c : u''(x_i) > 0, \forall x_i \geq x_c. \tag{12}$$

$$\exists x_v : u''(x_i) < 0, \forall x_i \leq x_v. \tag{13}$$

Of the different utility functions mentioned in the previous section, only sigmoid functions satisfy conditions (11), (12) and (13). Two of the widely used sigmoid functions are:

$$u_1(x) = \frac{1}{1+e^{\zeta (x_m - x)}}. \tag{14}$$

$$u_2(x) = \frac{(x/x_m)^{\zeta}}{1+(x/x_m)^{\zeta}}. \tag{15}$$

For (14) and (15), we see that $u_1(x_m) = u_2(x_m) = 0.5$. The value of x_m corresponds to the point of inflection. In fact, x_m *and* ζ respectively represent the center and steepness of the utility function. We use the parameter ζ to model the sensitivity of the user. In addition to being twice differentiable and satisfying requirements (11), (12) and (13), the sigmoid utility function is redefined to satisfy the following:

$$u(x) = 1 \; \forall x \leq x_\alpha. \tag{16}$$

$$u(x) = 0 \; \forall x \geq x_\beta. \tag{17}$$

$$u(x_m) = 0.5 \; for \; any \; x_m. \tag{18}$$

We use the following utility function, as it satisfies all our requirements, given a range for a downward criterion $x, x_\alpha \leq x \leq x_\beta \leq \infty$, and a middle point x_m:

$$u(x) = \begin{cases} 1 & x < x_\alpha \quad (a) \\ 1 - \frac{1}{1+e^{\zeta \frac{x_m - x}{x_m - x_\alpha}}} & x_\alpha \leq x \leq x_m \quad (b) \\ \frac{1}{1+e^{\gamma \frac{x - x_m}{x_\beta - x_m}}} & x_m < x \leq x_\beta \quad (c) \\ 0 & x > x_\beta \quad (d) \end{cases} \tag{19}$$

where:

$$\zeta = \max\left(5 \, , 4 * \frac{x_m - x_\alpha}{x_\beta - x_m}\right). \tag{20}$$

and

$$\gamma = \zeta * \frac{x_\beta - x_m}{x_m - x_\alpha}. \tag{21}$$

The parameter ζ represents the adjusted steepness parameter or sensitivity parameter.

Clearly, (19b) and (19c) resemble (14). The reason for choosing a function that resembles (14) and not (15) is because of the properties of the exponential function. The rapid growth of the exponential function gives a suitable way to model the irritation of users. Using the steepness parameter, it is easy to control the curve of this function.

Theorem 1: The proposed utility function (19) satisfies requirements (16), (17), (18), is twice differentiable, concave in the interval $[x_\alpha, x_m]$ and convex in the interval $(x_m, x_\beta]$ with an inflection point at x_m.

Proof: First, conditions (16), (17) and (18) are clearly satisfied by (19a), (19b), (19c) and (19d).

For $x_\alpha \leq x \leq x_m$:

$$u'(x) = -\frac{\zeta\, e^{\zeta\left(\frac{x_m - x}{x_m - x_\alpha}\right)}}{\left(1+e^{\zeta\left(\frac{x_m - x}{x_m - x_\alpha}\right)}\right)^2 (x_m - x_\alpha)}. \tag{22}$$

since $x_m - x_\alpha > 0, e^x > 0 \; and \; \zeta > 0$, then $u'(x) < 0$.

For $x_m < x \leq x_\beta$:

$$u'(x) = -\frac{\gamma\, e^{\gamma\left(\frac{x - x_m}{x_\beta - x_m}\right)}}{\left(1+e^{\gamma\left(\frac{x - x_m}{x_\beta - x_m}\right)}\right)^2 (x_\beta - x_m)}. \tag{23}$$

since $x_\beta - x_m > 0, e^x > 0 \; and \; \gamma > 0$, then $u'(x) < 0$.

Hence condition (11) is satisfied over the interval $[x_\alpha, x_\beta]$.

From (22) and (23), we conclude that:

$$\lim_{x \to x_m^-} u'(x) = -\frac{\zeta}{4\,(x_m - x_\alpha)} = \lim_{x \to x_m^+} u'(x). \tag{24}$$

Hence $u(x)$ is continuously differentiable and thus twice differentiable.

We calculate the second derivative of $u(x)$ as (25) and (26) for the intervals $[x_\alpha, x_m]$ and $(x_m, x_\beta]$ respectively:

$$u''_\alpha(x) = -\frac{9\, e^{3\left(\frac{x_m - x}{x_\beta - x_\alpha}\right)}\left(e^{3\left(\frac{x_m - x}{x_\beta - x_\alpha}\right)} - 1\right)}{\left(1 + e^{3\left(\frac{x_m - x}{x_\beta - x_\alpha}\right)}\right)^3 (x_\beta - x_\alpha)^2}. \tag{25}$$

$$u''_\beta(x) = \frac{9\, e^{3\left(\frac{x_m - x}{x_m - x_\beta}\right)}\left(e^{3\left(\frac{x_m - x}{x_m - x_\beta}\right)} - 1\right)}{\left(1 + e^{3\left(\frac{x_m - x}{x_m - x_\beta}\right)}\right)^3 (x_m - x_\beta)^2}. \tag{26}$$

It can be inferred that $u''_\alpha(x) \leq 0$; hence $u''_\beta(x) \geq 0$. This implies that our function is concave in the interval $[x_\alpha, x_m]$ and convex in the interval $(x_m, x_\beta]$ with an inflection point at x_m

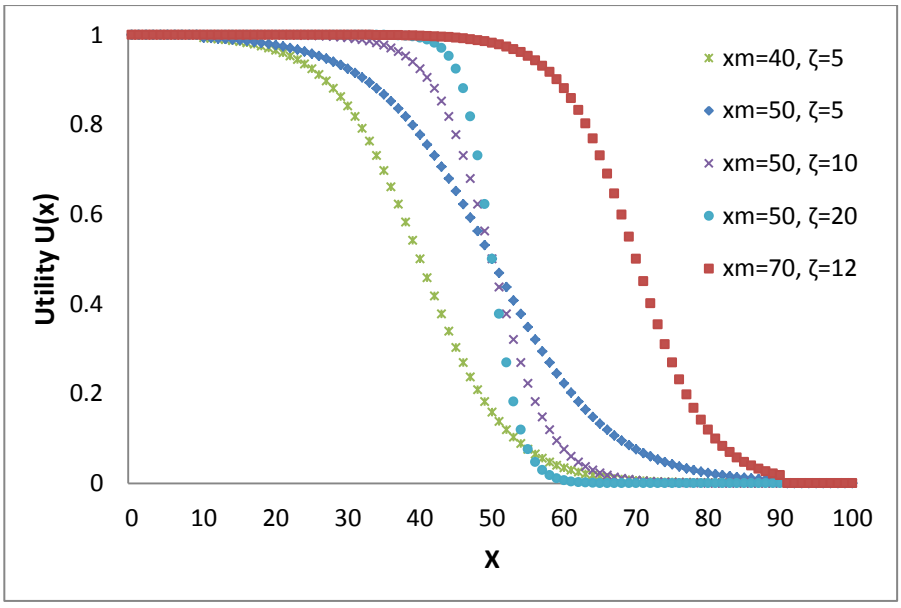


Figure 34: Utility Function Characteristics

.

Figure 34 illustrates the form of the utility function for different values of $x_m$and ζ while $x_\alpha = 10$ and $x_\beta = 90$. The parameter x_α represents the minimum error rate at which the user starts noticing the errors (or rather, starts caring about the quality) while x_β represents the maximum error rate after which the user's experience does not change and is judged to be too bad. Finally, x_m represents the amount of error rate at which the utility is equal to 0.5. These parameters depend on the type of content as we show later. The user controls the sensitivity threshold ζ that determines the shape of his utility. The smaller the value of ζ the higher the sensitivity. Maximum value ζ can get is $10 \cdot \zeta_{init}$ where, ζ_{init} is the initial value calculated in (20). Sensitivity in this context is associated with a cost the user is willing to pay to get a better service. If we consider the three curves in Figure 34 with $x_m = 50$ and different ζ values, we notice that for $x = 39$, $U(x) = 0.79\ for\ \zeta = 5$ while $U(x) = 0.93\ for\ \zeta = 10$ and $U(x) = 0.99\ for\ \zeta = 20$. This implies that the user with $\zeta = 5$ experiences a low utility value for the same value of x compared to users with higher ζ. Moreover, if we associate a cost function with the value of the utility, then sensitive users will be paying less money than less sensitive users, as is illustrated in Figure 34. Thus, users control the sensitivity parameter and can trade quality to the price they are paying. This utility-based pricing scheme gives users control and flexibility over their budget. Service providers will be obliged to improve the quality of more sensitive users over those with higher ζ values. Moreover, more sensitive users should be charged a higher base rate in order to preserve the fairness among different users.

6.3.2. The Delay Index

A delayed packet causes the frame to which it belongs to be buffered until that packet is received. Since each frame has a deadline constraint associated with it, if the frame is not available at the client when the decoding process attempts to display it, it will miss its deadline or be skipped if the buffer is full. In the worst case, if the frame in question is an I-frame, all subsequent GOP frames (depending on it) will be unsuccessfully decoded until the delayed packet reaches the client, resulting in more frames missing their

deadlines and in buffer overflows. In the simplest case where there are no frame dependencies, packet delays cause visual impairments.

In order to cater for these situations, we propose a new metric called the delay index, D_i:

$$D_i = 100 * \left(\frac{missedFrames}{totalFrames} + \frac{Delay}{Buffer}\right). \quad (27)$$

D_i is made up of two terms: the fraction of frames that missed their playback deadline, and the average delay of displayed frames divided by the buffer size. This latter represents the state of the buffer (how full it is) as a ratio. Note that when the buffer overflows due to excessive delays, packets buffered at the start are discarded to make room for incoming ones. These discarded frames are recorded as missed frames. Values in (27) are recorded over an observation period that we set to 5 seconds in our simulations. However, one can change this value to trade accuracy and overhead. Since both terms in (27) are ratios, we multiply their sum by 100. The two parts can take a maximum of 1 but both cannot be equal to 1 at the same time. This is because $missedFrames \leq totalFrames$ and when $Delay > Buffer$, the buffer overflows and causes frames to be discarded (recorded as missed frames and not included in delay calculations). Hence if all frames missed their deadline, i.e. $\frac{missedFrames}{totalFrames} = 1$, $Delay = 0$ and $D_i = 100$. On the other hand, if no frame missed the deadline, i.e. $\frac{missedFrames}{totalFrames} = 0$, the average delay would not exceed the buffer size, hence $\frac{Delay}{Buffer} \leq 1 \; and \; D_i \leq 100$. As our simulation results show, any delay index value above 80 corresponds to unacceptable quality. As opposed to using packet delays, the proposed delay index implicitly takes into account the effect of frame type, since a delayed I-frame would cause other frames that depend on it to be further delayed or miss their deadline.

6.3.3. Pricing Function

Providing QoS differentiation in the Internet is closely related to the implementation of appropriate pricing and charging schemes. Service providers have been adopting pricing schemes that charge customers according to their QoS requirements. When QoS degrades, customers are still charged the same amount. This leads to unsatisfied customers and negatively affects the service provider's reputation and business. While service providers cannot predict the drop in QoS, they can and should keep their customers happy by charging them a fair price based on the quality drop. It makes more sense to charge for QoE instead of QoS. Utility functions work perfectly in this situation. Many QoS-based pricing schemes and solutions have been proposed in the literature [136], but Reichl et al. [137] describe forthcoming transitions from QoS to QoE pricing.

For the sake of simplicity, we propose a straightforward QoE cost function $C(u)$, a function of the user's utility u as in (28) and (29). Here, the values P_{max}, P_{med} and P_{min} represent the maximum, medium and minimum prices for using the service, respectively.

For $0.5 \leq u \leq 1$:

$$C(u) = 2u(P_{max} - P_{med}) + 2P_{med} - P_{max}. \quad (28)$$

For $u < 0.5$:

$$C(u) = P_{min}. \quad (29)$$

6.4. Simulations and Results

6.4.1. Scenario and Parameters

Using the network simulator NS-2 [138], Evalvid framework [139] and MSU Video Quality Measurement Tool [140], we run a set of simulations with different byte loss ratios in order to quantify the relationship between error rate and QoE in terms of the MSSIM [141] index. We measure the byte loss ratio as the difference between bytes sent by the

VSO and the bytes received at the STB. By varying the packet error ratio at the link between VSO and STB, we introduce packet loss at the network layer which translates into byte loss at the application layer. Only error-free frames which can be decoded are taken into consideration when adding the amount of bytes received at the STB. This means that when a frame is erroneous, we drop it as well as all the other frames that depend on it for decoding. For monitoring purposes, the sender and receiver agree on a specific interval on which measurements will be taken and exchanged. The calculated utility can be used as feedback to different adaptive streaming and error correction schemes to improve the user's QoE. Similarly, we introduce delays at the network layer that translate into frame delays at the application layer.

In order to select appropriate values for the utility function parameters (x_α, x_m and x_β), we experimented with two different sets of standard video sequences [142]: the first one consists of 'talking head' type where scene change and motion are very limited. The second set consists of 'animation' clips rich in graphics, motion and scene change. All sequences have a frame rate of 25 fps and are in QCIF format encoded using MPEG4. However, the Tax and Penguins sequences are different. The Tax sequence has a frame rate and resolution of 30 fps and 320x240, respectively. The Penguins sequence has 24 fps and a resolution of 624x336. For each video sequence, we record the bytes loss rate and the corresponding MSSIM.

6.4.2. Results

6.4.2.1 Byte Loss Ratio

It is worth mentioning that we have run simulations based on different packet loss ratios and found no clear correlation between packet loss ratio and MSSIM for the same set of video sequences. This strengthens our choice for byte loss ratio at the application layer as opposed to packet loss ratio at the network layer. The results of the packet loss based simulations are illustrated in Figure 35 and Figure 36.

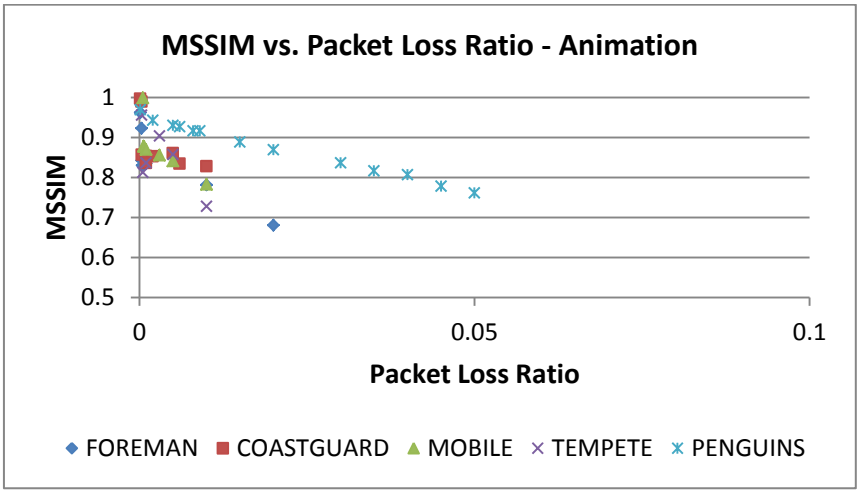


Figure 35: MSSIM vs. Packet Loss Ratio – Animation Video Type

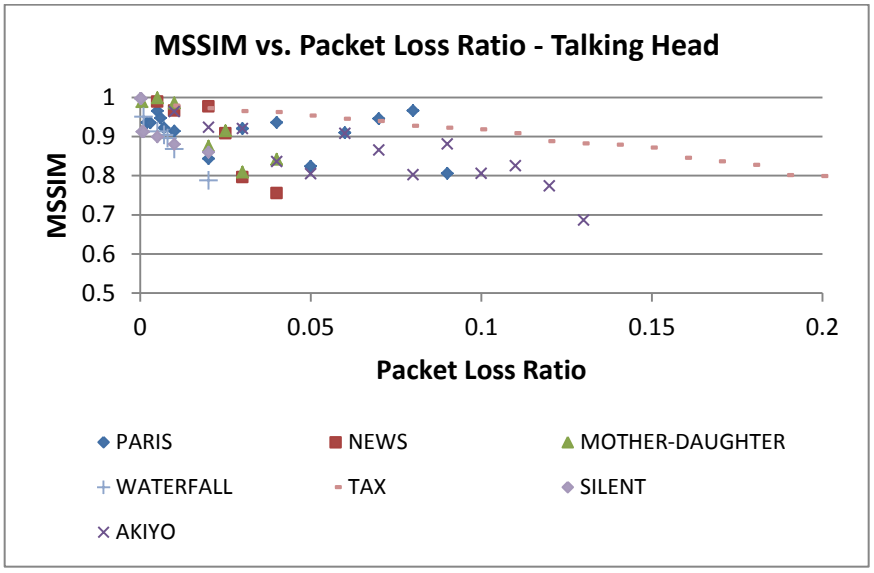


Figure 36: MSSIM vs. Packet Loss Ratio – Talking Head Video Type

Figure 37 and Figure 38 plot the MSSIM in terms of byte loss ratio. We can spot a clear difference between how the two different sets of sequences behave to byte loss ratio. For instance, we notice that talking head type sequences reach a MSSIM of 0.9 at about 65% byte loss. However, animation type sequences reach the same MSSIM value for only 8% byte loss. According to [141], an MSSIM of 0.89 represents the threshold between a MOS score of 3 (Fair: slightly annoying) and 2 (Poor: Annoying). As the MSSIM value

increases beyond 0.89, the quality improves towards excellent and vice-versa. Hence, we suggest mapping an MSSIM value of 0.89 into our proposed utility function as 0.5 utility.

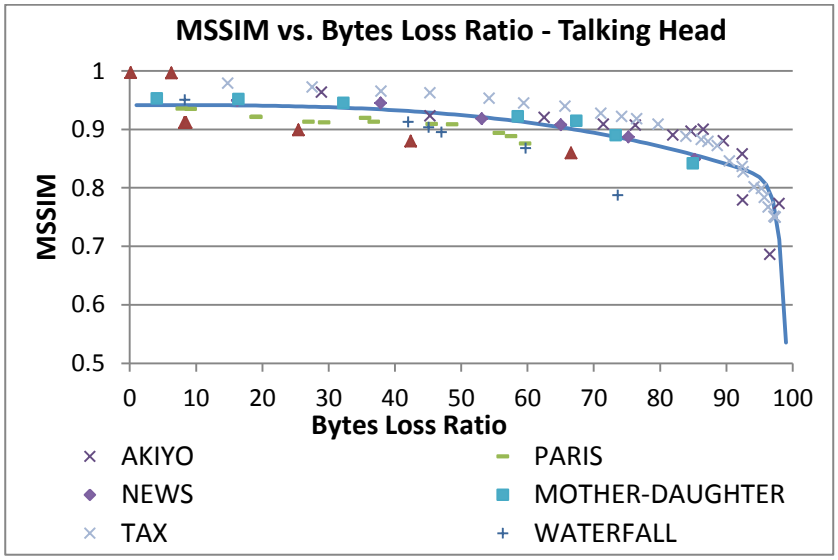


Figure 37: MSSIM vs. Byte Loss Ratio – Talking Head Video Type

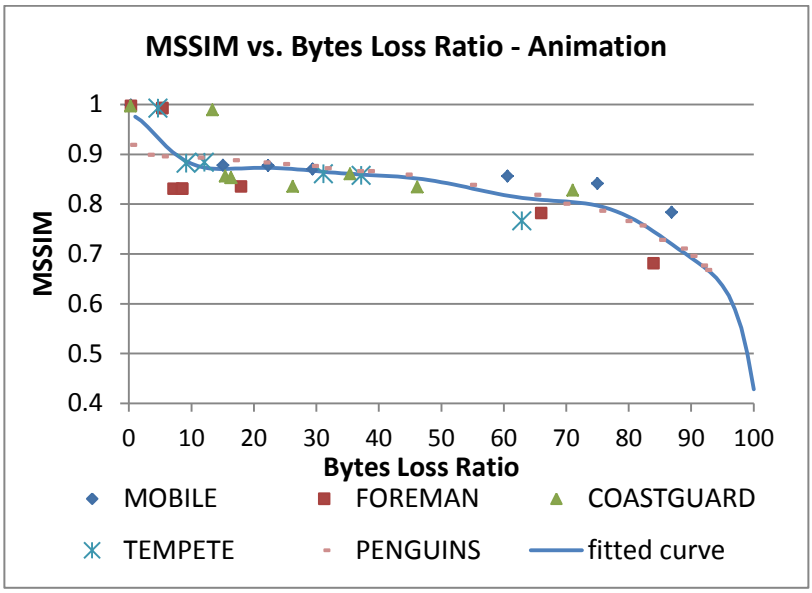


Figure 38: MSSIM vs. Byte Loss Ratio – Animation Video Type

A utility value below 0.5 represents unacceptable quality, while a value above 0.5 represents better quality (up to excellent quality as a utility of 1). As we discussed earlier, users will react differently to the change in the quality. We suggest price as a parameter to the sensitivity of users. Therefore, the sensitivity parameter ζ introduced in our utility serves to control the shape of the function for each user.

Based on the results obtained in [141], we choose the values of x_α, x_m and x_β as follows:

- $x_\alpha = 10, x_m = 71$ and $x_\beta = 80$ with a default $\zeta = 31$ for talking head type sequences
- $x_\alpha = 0, x_m = 8.5$ and $x_\beta = 10$ with a default $\zeta = 22.67$ for animation type sequences.

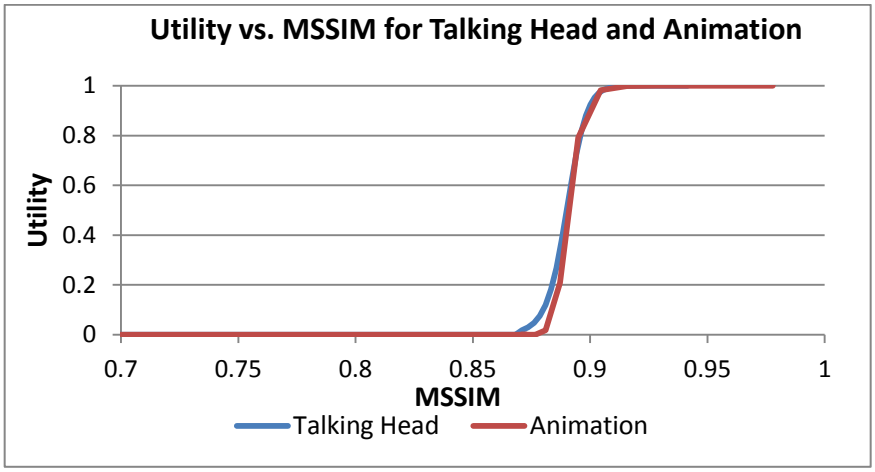


Figure 39: Utility vs. SSIM for Talking Head and Animation Types (Bytes Loss)

We plot utility vs. MSSIM for the default calculated sensitivity ζ in Figure 39. We calculated the correlation R between the plots in Figure 39 as 0.998 and the mean square error as 0.0002. Next, we perform delay based simulations on the same set of video sequences.

6.4.2.2 *The Delay Index*

In order to study the effect of frame delays on MSSIM, we change the receiver buffer size in Evalvid in order to simulate a wide range of receivers. Basically, we start with a buffer size of 300ms and double it for each simulation to a maximum of 5000ms or 5 seconds. Results for Animation type videos are depicted in Figure 40, Figure 41, Figure 42, Figure 43 and Figure 44. While results for Talking Head type videos are shows in Figure 45, Figure 46, Figure 47, Figure 48 and Figure 49.

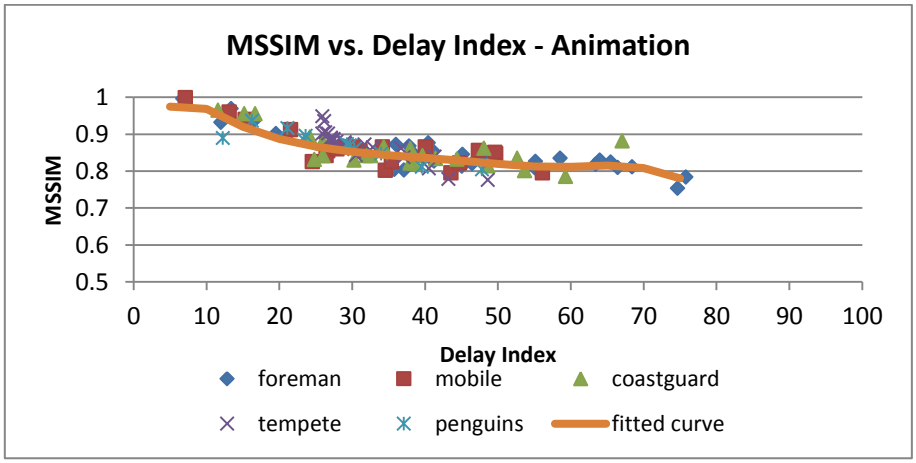


Figure 40: MSSIM vs. Delay Index for Animation - 300ms Buffer

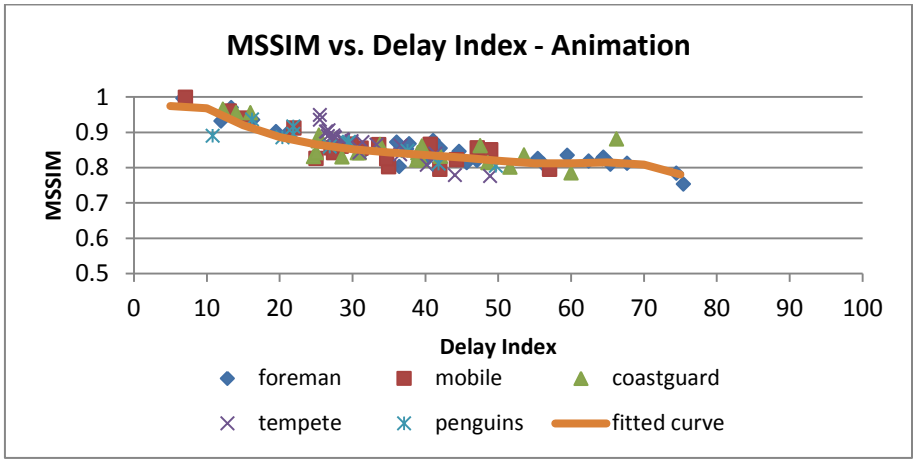


Figure 41: MSSIM vs. Delay Index for Animation - 600ms Buffer

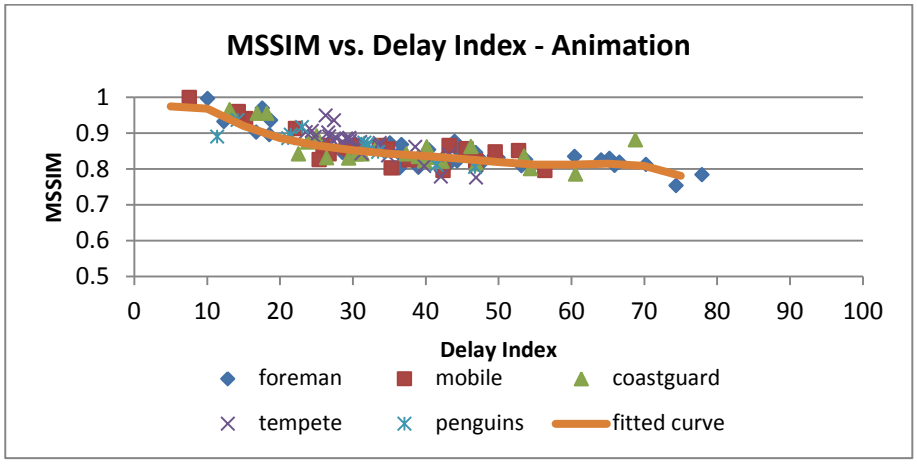


Figure 42: MSSIM vs. Delay Index for Animation - 1200ms Buffer

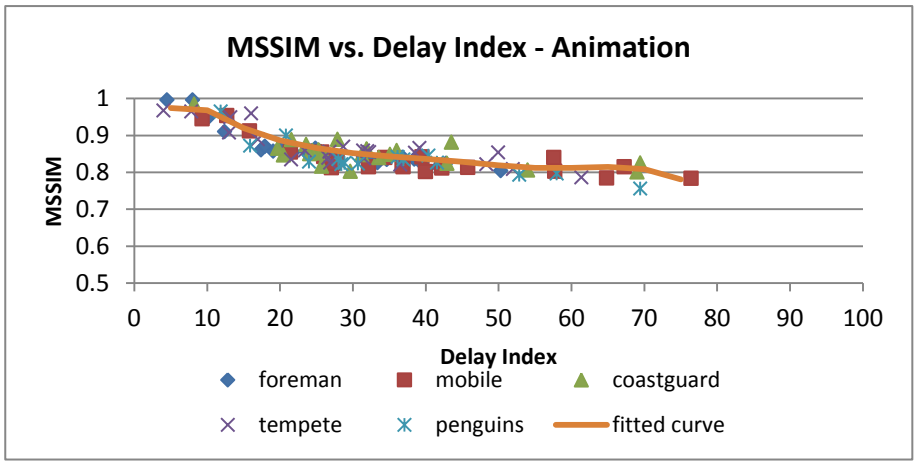


Figure 43: MSSIM vs. Delay Index for Animation - 2400ms Buffer

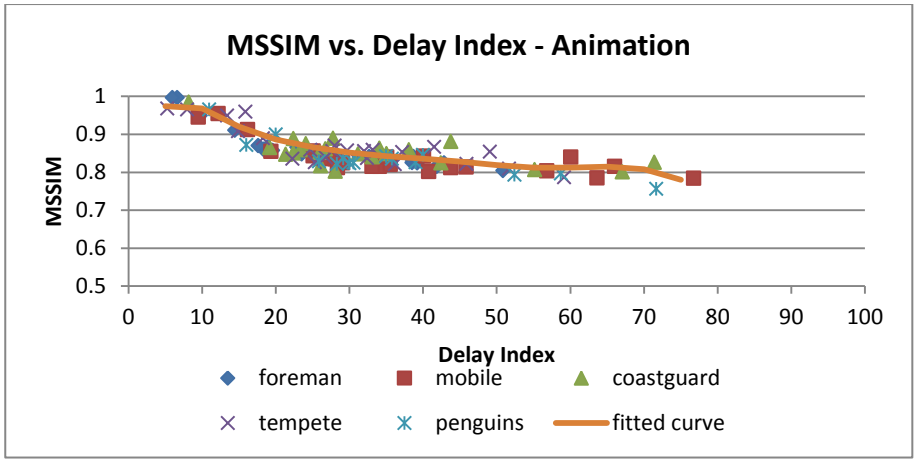


Figure 44: MSSIM vs. Delay Index for Animation - 5000ms Buffer

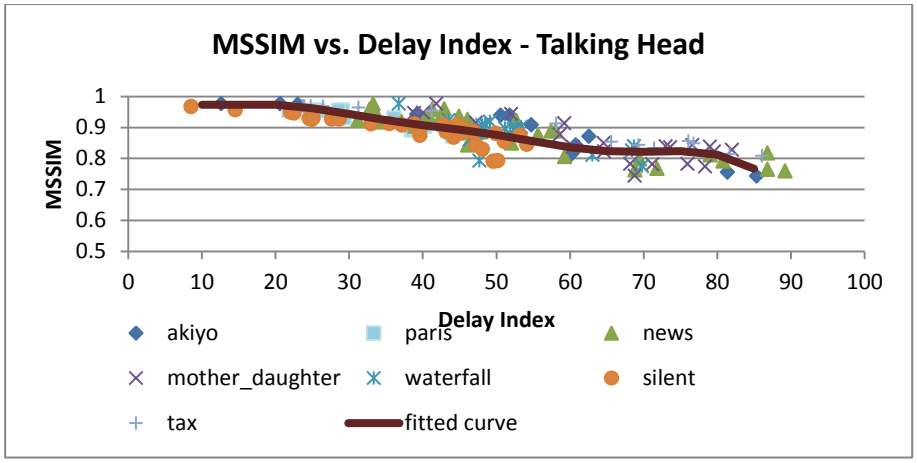


Figure 45: MSSIM vs. Delay Index for Talking Head - 300ms Buffer

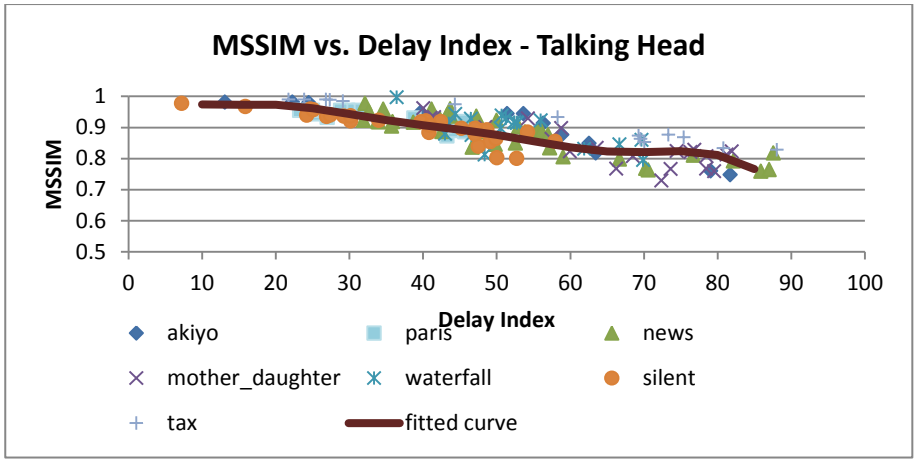


Figure 46: MSSIM vs. Delay Index for Talking Head - 600ms Buffer

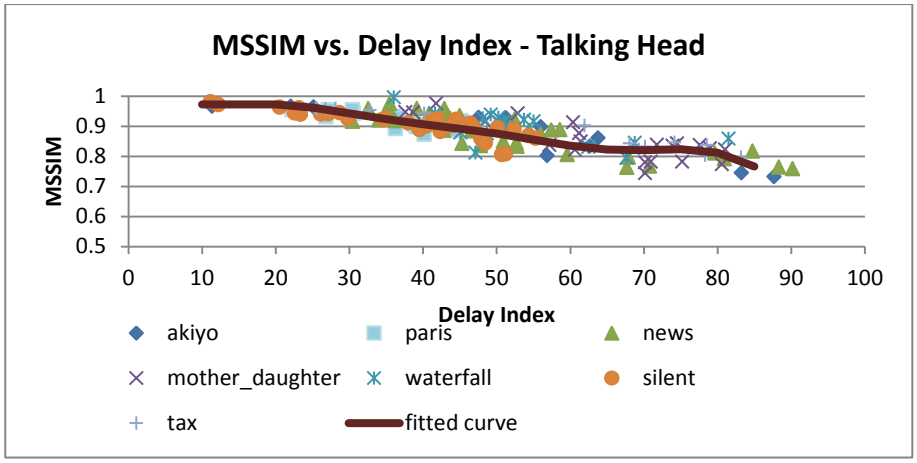


Figure 47: MSSIM vs. Delay Index for Talking Head - 1200ms Buffer

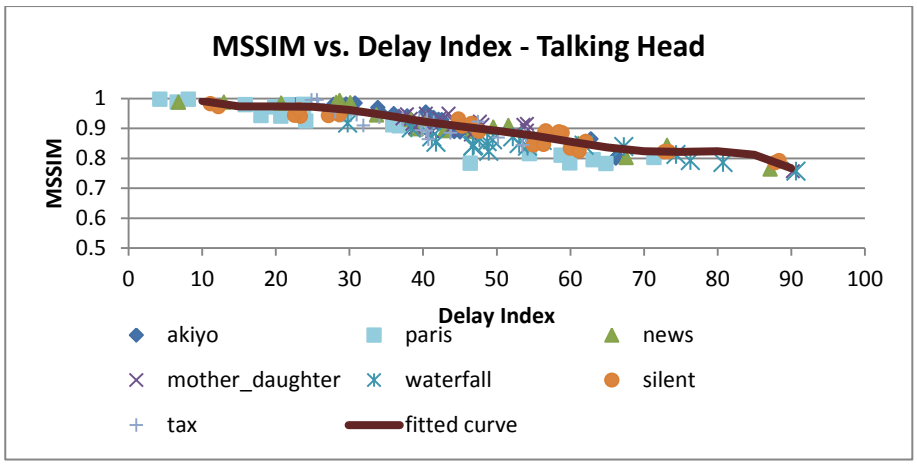


Figure 48: MSSIM vs. Delay Index for Talking Head - 2400ms Buffer

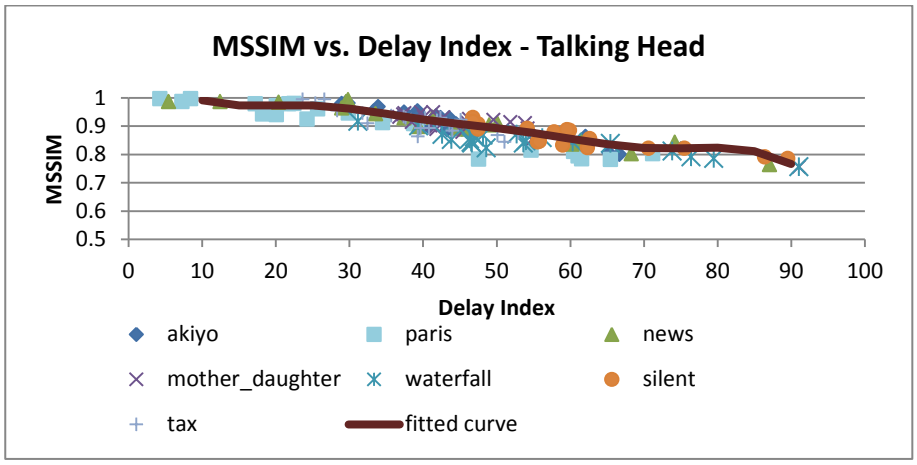


Figure 49: MSSIM vs. Delay Index for Talking Head - 5000ms Buffer

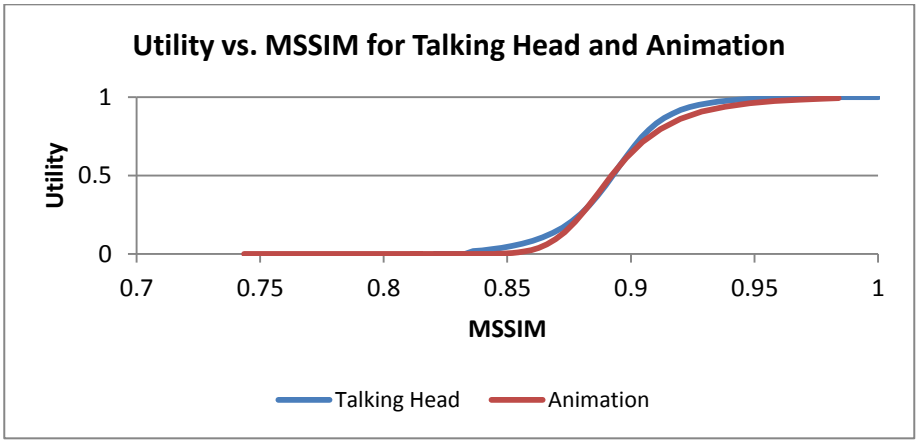


Figure 50: Utility vs. SSIM for Talking Head and Animation Types (Delay Index)

Following the same analogy used for byte loss, we choose the following values for the delay index:

- $x_\alpha = 25, x_m = 50, x_\beta = 65$ with a default $\zeta = 6.67$ for talking head type video sequences.
- $x_\alpha = 8, x_m = 19, x_\beta = 40$ with a default $\zeta = 5$ for animation type video sequences.

Finally, we calculated the correlation and mean square error between utility and MSSIM for previously chosen values as in Figure 50. The correlation and the mean square error were calculated respectively as 0.997 and 0.0003 for the delay index, respectively. This proofs the effectiveness of our utility function in calculating the same utility value for the same MSSIM value of the different types of videos we experimented with. Our proposed utility gives a very good approximation for MSSIM.

6.5. Summary

In this chapter, we have presented a new method of monitoring and predicting the QoE of IPTV users in overlay networks using a utility-based approach. The proposed solution is very light and simple and does not require high processing or memory consumption. The proposed utility function takes error rate and delay at the application layer as criteria, because the amount of data lost and the delays at the application layer give a more accurate view of the quality of the video stream as opposed to using statistical information at the network layer.

We performed simulations, using ns-2 and Evalvid, to assess the accuracy of our scheme. Results show that our utility-based approach maps utility to video quality accurately and offers users the flexibility to balance quality and price. The results of the simulations show that the use of byte loss rate and delay as criteria is effective, reliable and accurate in estimating the QoE of IPTV users in terms of MSSIM.

Chapter 7

Conclusion and Future Research Directions

This chapter outlines the conducted research work. The focus of the work has been the autonomic management of SSONs. Section 7.1 provides a summary of research contributions in the area of autonomic SSONs management. A summary of future research directions is then given in Section 7.2.

7.1. Thesis Contributions

The first step to solving the autonomous management problem was a literature review of the major works and contributions in the fields of networks management particularly overlay networks management, and the notion of autonomic computing. The construction of SSONs on the fly was discussed in previous work by Al-Oqily et al. [76], [78] and [79]. Once SSONs are created, they must be continuously monitored to ensure that the required QoS and QoE are being met. Based on the different characteristics and requirements of SSONs, we propose an architecture to manage SSONs in an autonomic way. The architecture uses a fully-decentralized approach to gather monitoring information from different parts of the network and analyze it to identify potential problems and degradations. AMs are essential elements in every autonomic architecture. They are responsible for managing one or more elements in the network by configuring sensors, sharing the acquired knowledge and analyzing it to take appropriate decisions. A summary of the main contributions of current research work is presented as follows:

- An autonomic architecture to support the delivery of media services in overlay networks based on users' QoS and QoE requirements. The heterogeneity of

hardware and software components stemming from network infrastructure is tackled using shared ontologies that give the semantics of the network components. Different components have been described to allow the creation and maintenance of SSONs with minimal human intervention. Policies that describe high level goals are fed to the system by humans and are continuously checked against the state of the network to optimize resource usage and increase users' QoS and QoE.

- An AM election scheme and a selection algorithm based on a game theoretic approach have been presented. Current management schemes assume that management nodes have enough resources and are willing to take part of the management process. This assumption is not always true, especially for the case of SSONs where nodes have different storage and processing capabilities. Hence, only a subset of overlay nodes will be willing to manage other elements in the network. The proposed scheme allows the promotion of powerful stable nodes to become AMs. Then the proposed algorithm ensures that overlay nodes select the most appropriate managers that will receive sensed information intact and in a timely manner. We also show through simulations that our scheme adapts to changing network conditions including node churn and mobility, converges in a few rounds, and is 86% close to the optimal solution.

- A loss and delay-based utility function for the prediction of IPTV QoE has been presented. The proposed utility function uses the loss ratio and frame delay at the application layer as criteria to calculate the utility value that represents a mapping to QoE. The function is simple and does not require high processing power which makes it easy to implement on any device no matter its processing power and capability. The predicted utility value is used to monitor the level of quality IPTV users are experiencing and to ensure that it does not go below the required levels as indicated by high level goals fed to the system by human administrators. Through simulations, we show the effectiveness of the proposed

utility function in predicting IPTV QoE for different types of standard video sequences with different characteristics.

7.2. Future Research Directions

The main focus on future research work can be summarized into three key directions as follows:

- The proposed architecture opens many directions to explore, especially the Analyzer and Planner, Knowledge Base, and Executor components. The utility function proposed is one example of analyzing application statistical data and converting it into something meaningful, i.e. QoE. With the introduction of service composition schemes, new applications will emerge into the Internet. Analyzing the quality of these applications against any degradation is essential to ensure high customer satisfaction. On the other hand, optimization and prediction schemes that analyze the state of the network as a whole, and of individual elements, to predict future behavior and optimize the operation of the system need to be studied further. Shared ontologies, policy and context management are other important parts of the architecture that we plan to further investigate for building an efficient knowledge base.

- Concerning the proposed AM election and selection algorithm, we plan to study a way to divide the management of different existing SSONs among the set of AMs. As each MP can be part of many SSONs, we intend to investigate cooperation schemes to assess how each SSON will be managed, and how different AMs should cooperate to make individual decisions that will not affect the overall system operation.

- Another focus of future research work is to validate the proposed utility function using real-life experiments. We plan to implement the proposed function in an SSON system and test human subjects on the perceived QoE in terms of utility and

MOS. We also plan to run more simulations with different video sequences (both standard and non-standard), having different characteristics such as resolution.

Bibliography

[1] A. Karmouch, "Mobile software agents for telecommunications," *IEEE Communications Magazine*, vol.36, no.7, pp.24-25, Jul. 1998.

[2] R. Boutaba and A. Polyrakis, "Projecting advanced enterprise network and service management to active networks," *IEEE Network Magazine*, vol.16, no.1, pp.28-33, Jan./Feb. 2002.

[3] S. Calo and M. Sloman, "Guest editorial: Policy-based management of networks and services", *J. Network and Systems Management*, vol. 11, n. 3, pp. 249–252, 2003.

[4] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proceedings of the eighteenth ACM symposium on Operating systems principles*, 2001, pp. 131-145.

[5] IEEE Standards Associaction. (2011, Sep.). IEEE NGSON WG. [Online]. Available: http://grouper.ieee.org/groups/ngson

[6] J.O. Kephart and D.M. Chess, "The vision of autonomic computing," *IEEE Computer Society*, vol.36, no.1, pp. 41- 50, Jan. 2003.

[7] D. Zhenhai, Z. Zhang, and Y.T. Hou, "Service overlay networks: SLAs, QoS, and bandwidth provisioning," *IEEE/ACM Transactions on Networking*, vol.11, no.6, pp. 870- 883, Dec. 2003.

[8] S. Schmid, F. Hartung, M. Kampmann, S. Herborn, and J. Rey, "SMART: Intelligent multimedia routing and adaptation based on service specific overlay networks," in *Proceedings of the Eurescom Summit*, 2005, pp. 69-77.

[9] S. Bakewell. (1995). The autonomic nervous system. Addenbrooke's Hospital, Cambridge. [Online]. Available: http://www.nda.ox.ac.uk/wfsa/html/u05/u05_010.htm

[10] B. Melcher and B. Mitchell, “Towards an autonomic framework: Self-configuring network services and developing autonomic applications”, *Intel Technology Journal*, vol. 8, issue 4, Nov. 2004.

[11] OSI-QoS, “Quality of service basic framework – Outline,” International Standards Organization ISO/IEC JTC1/SC21/WG1 N1145, UK, 1994.

[12] L. Hui-Lan and I. Faynberg, “An architectural framework for support of quality of service in packet networks,” *IEEE Communications Magazine*, vol.41, no.6, pp. 98-105, June 2003.

[13] R. Braden, D. Clark, and S. Shenker, “Integrated services in the internet architecture: an overview,” RFC 1633, June 1994.

[14] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, “Resource reservation protocol (RSVP)--Version 1 functional specification,” RFC 2205, IETF, Sep. 1997.

[15] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, “An architecture for differentiated service,” IETF Network Working Group RFC 2475, Dec. 1998.

[16] T. Nadeau, C. Srinivasan, and A. Farrel, “Multiprotocol label switching (MPLS) management overview,” IETF Network Working Group RFC 3814, Sep. 2003.

[17] J. Jannotti, D. K. Gifford, and K. L. Johnson. “Overcast: Reliable multicasting with an overlay network”, in *Proceedings of the 4th conference on Symposium on Operating System Design & Implementation,* vol. 4, 2000.

[18] C. Yang-hua, S.G. Rao, S. Seshan, and Z. Hui, “A case for end system multicast,” *IEEE Journal on Selected Areas in Communications*, vol.20, no.8, pp. 1456- 1471, Oct. 2002.

[19] Z. Beichuan, S. Jamin, and Z. Lixia, “Host multicast: A framework for delivering multicast to end users,” in *Proceedings of IEEE INFOCOM 2002*, *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, vol.3, pp.1366-1375, 2002.

[20] Akamai Technologies. (2011). [Online]. Available: http://www.akamai.com

[21] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of ACM SIGCOMM,* 2001.

[22] W. Feng, X. Yongqiang, and L. Jiangchuan, "mTreebone: A collaborative tree-mesh overlay network for multicast video streaming," *IEEE Transactions on Parallel and Distributed Systems*, vol.21, no.3, pp.379-392, Mar. 2010.

[23] T. Wanqing, J. Xing, C.J. Sreenan, and M.W. O'Brien, "Performance analysis for overlay multicast on tree and M-D mesh topologies (II)," in *Proceedings of IEEE International Conference on Communications, ICC '08*, pp.419-423, 2008.

[24] S. Chen, B. Shi, S. Chen, and Y. Xia, "ACOM: Any-source capacity-constrained overlay multicast in non-DHT P2P networks," *IEEE Transaction on Parallel and Distributed Systems*, vol. 18, no. 9, pp. 1188–1201, Sep. 2007.

[25] N. R. Manohar, A. Mehra, M. H. Willebeek-LeMair, and M. Naghshineh, "A framework for programmable overlay multimedia networks," *IBM Journal of Research and Development,* vol. 43, pp. 555-578, Apr. 2000.

[26] X. Zhe, Z. Qian, Z.Wenwu, Z. Zhensheng, and Z. Ya-Qin, "Peer-to-peer based multimedia distribution service," *IEEE Transactions on Multimedia*, vol.6, no.2, pp. 343-355, Apr. 2004.

[27] X. Changqiao, G.M. Muntean, E. Fallon, and A. Hanley, "Distributed storage-assisted data-driven overlay network for P2P VoD services," *IEEE Transactions on Broadcasting*, vol.55, no.1, pp.1-10, Mar. 2009.

[28] A. Dutta and H. Schulzrinne, "MarconiNet: Overlay mobile content distribution network," *IEEE Communications Magazine*, vol.42, no.2, pp. 64- 75, Feb. 2004.

[29] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of* ACM *SIGCOMM*, vol. 31, 2001.

[30] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pp. 329-350, 2001.

[31] D. Malkhi, M. Naor, and D. Ratajcza, "Viceroy: A scalable and dynamic emulation of the butterfly," in *Proceedings of the twenty-first annual symposium on Principles of distributed computing (PODC '02)*, pp. 183-192, 2002.

[32] N. Niebert, A. Schieder, H. Abramowicz, G. Malmgren, J. Sachs, U. Horn, C. Prehofer, and H. Karl, "Ambient networks: An architecture for communication networks beyond 3G," *IEEE Wireless Communications*, vol.11, no.2, pp. 14-22, Apr. 2004.

[33] ITU-T Newslog. (2009, Feb.). New IPTV standard supports global rollout. *International Telecommunication Unit*. [Online]. Available: http://www.itu.int/ITU-T/newslog/New+IPTV+Standard+Supports+Global+Rollout.aspx

[34] Alliance for Telecommunications Industry Solutions. (2005, Jul.). ATIS IPTV exploratory group report and recommendation to the TOPS council. *Alliance for Telecommunications Industry Solutions*. [Online]. Available: http://www.atis.org/tops/IEG/ATIS_IPTV_EG_RPT_final.pdf

[35] ITU-T. (2008, Sep.). Recommendation E.800: Terms and definitions related to quality of service. *International Telecommunication Unit*. [Online]. Available: http://www.itu.int/rec/T-REC-E.800-200809-I

[36] ITU-T Study Group 12. (2006, Sep.). Recommendation P.10/G.100: New Appendix I – Definition of Quality of Experience (QoE). *International Telecommunication Unit*. [Online]. Available: http://www.itu.int/rec/T-REC-P.10-200701-S!Amd1

[37] Nokia. (2004). Quality of experience (QoE) of mobile services: Can it be measured and improved? *Nokia Corporation*. [Online]. Available: http://www.nokia.com/NOKIA_COM_1/About_Nokia/Press/White_Papers/pdf_files/whitepaper_qoe_net.pdf

[38] ITU-T. (2011, Dec.). Recommendation Y.1541: Network performance objectives for IP-based services. *International Telecommunication Unit*. [Online]. Available: http://www.itu.int/rec/T-REC-Y.1541-201112-P

[39] ITU-T. (2000, Nov.). Recommendation F.700: Framework Recommendation for audiovisual/multimedia services. *International Telecommunication Unit.* [Onilne]. Available: http://www.itu.int/rec/T-REC-F.700-200011-I

[40] ITU-T. (2001, Nov.). Recommendation G.1010: End-user multimedia QoS categories. *International Telecommunication Unit*. [Onilne]. Available: http://www.itu.int/rec/T-REC-G.1010-200111-I

[41] 3GPP. (2003, June). 3GPP TS 22.105: Services and service capabilities. *3rd Generation Partnership Project*. [Onilne]. Available: http://www.3gpp.org/ftp/Specs/html-info/22105.htm

[42] ITU-T. (1996, Aug.). Recommendation P.800: Methods for subjective determination of transmission quality. *International Telecommunication Union*. [Onilne]. Available: http://www.itu.int/rec/T-REC-P.800-199608-I

[43] A. F. Wattimena, R. E. Kooij, J. M. Van Vugt, and O. K. Ahmed, "Predicting the perceived quality of a first person shooter: the quake IV G-model," in *Proceedings of the 5th ACM SIGCOMM Workshop on Network and System Support for Games*, vol. 42, 2006.

[44] M.H. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *IEEE Transactions on Broadcasting*, vol. 50, no. 3, pp. 312- 322, 2004.

[45] Z. Wang and Q. Li, "Video quality assessment using a statistical model of human visual speed perception," *J. Opt. Soc. Amer. A - Opt. Image Sci. Vis*., vol. 24, no. 12, pp. B61–B69, 2007.

[46] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electronics Letters*, vol.44, no.13, pp.800-801, 2008.

[47] The Video Quality Experts Group. [Online]. Available: www.its.bldrdoc.gov/vqeg/

[48] J. Galvin and K. McCloghie, "Administrative model for version 2 of the simple network management protocol (SNMPv2)," RFC 1445, Apr. 3, 1993.

[49] G. Pavlou, P. Flegkas, S. Gouveris, and A. Liotta, "On Management Technologies and the Potential of Web Services," *IEEE Communications Magazine*, vol.42, no.7, pp. 58- 66, Jul. 2004.

[50] Object Management Group. (2002, Apr.). Meta Object Facility (MOF) Specification, Version 1.4. *Object Management Group*. [Online]. Available: http://www.omg.org/spec/

[51] H. Hoang To, S. Krishnaswamy, and B. Srinivasan, "Mobile agents for network management: when and when not!," in *Proceedings of the 2005 ACM symposium on Applied computing (SAC '05),* Lorie M. Liebrock (Ed.), pp. 47-53, 2005.

[52] B. Jennings, S. van der Meer, S. Balasubramaniam, D. Botvich, M. O. Foghlu, W. Donnelly, and J. Strassner, "Towards autonomic management of communications networks," *IEEE Communications Magazine*,vol.45, no.10, pp.112-121, Oct. 2007.

[53] L. Hsin-Nan, W. Kuen-Pin, C. Jia-Min, S. Ting-Yiand, and H. Wen-Lian, "GANA-A genetic algorithm for nmr backbone resonance assignment," in *Proceedings of IEEE Computational Systems Bioinformatics Conference*, pp. 218- 219, 2005.

[54] H. Derbel, N. Agoulmine, and M. Salaiin, "ANEMA: Autonomic network management architecture to support self-configuration and self-optimization in IP networks," *The International Journal of Computer and Telecommunications Networking*, vol. 53, no. 3, pp. 418-430, 2009.

[55] (2009, Feb.). Autonomic Network Architecture (ANA) FP7 project. *European Union Information Society Technologies Framework Programme 6.* [Online]. Available: http://www.ana-project.org.

[56] R. Chadha, Y.-H. Cheng, J. Chiang, G. Levin, S.-W. Li, and A. Poylisher, "Policy-based mobile ad hoc network management for DRAMA," *MILCOM Journal*, vol. 3, pp. 1317–1323, Dec. 2004.

[57] A. Malatras, A. M. Hadjiantonis, and G. Pavlou, "Exploiting context awareness for the autonomic management of mobile ad hoc networks," *Journal of Network and Systems Management*, vol. 15, pp. 29–55, Mar.2007.

[58] D. M. Chess, A. Segal, and I. Whalley, "Unity: Experiences with a prototype autonomic computing system," in *Proceedings of the First International Conference on Autonomic Computing. IEEE Computer Society*, pp. 140–147, 2004.

[59] M. Ayari, Z. Movahedi, F. Kamoun, and G. Pujolle, "ADMA: Autonomous decentralized management architecture for MANETs - A simple self-configuring case study," in *Proceedings of International Wireless Communications and Mobile Computing Conference (IWCMC), AutonomicWireless Networking Workshop*. L, 2000.

[60] (2008). 4WARD FP7 project. [Online]. Available: http://www.4ward-project.eu

[61] D. G. Andersen, A. C. Snoeren, and H. Balakrishnan, "Best-path vs. multi-path overlay routing," in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (IMC '03)*, pp. 91-100, 2003.

[62] L. Zhi, Y, Lihua, M. Prasant, and C. Chen-Nee, "On the analysis of overlay failure detection and recovery," *The International Journal of Computer and Telecommunications Networking*, vol. 51, no. 13, pp. 3828-3843, 2007.

[63] D. Ben Khedher, R. Glitho, and R. Dssouli, "A novel overlay-based failure detection architecture for MANET applications," in *Proceedings of the 15th IEEE International Conference on Networks*, pp.130-135, 2007.

[64] B. Gedik and L. Liu, "A scalable peer-to-peer architecture for distributed information monitoring applications," *IEEE Transactions on Computers*, vol.54, no.6, pp. 767- 782, June 2005.

[65] A. Liakopoulos and A. Zafeiropoulos, "Autonomic monitoring and resource management using P2P techniques," in *Proceedings of Terena Networking Conference TNC*, 2009.

[66] H. Yan, R. Oliveira, K. Burnett, D. Matthews Z. Lixia, and D. Massey, "BGPmon: A real-time, scalable, extensible monitoring system," in *Proceedings of Cybersecurity Applications & Technology Conference For Homeland Security*, pp.212-223, 2009.

[67] V. Castro, P. Carvalho, and S. Rito Lim, "Toward a scalable and collaborative network monitoring overlay," in *Proceedings of the Third international conference on Traffic monitoring and analysis (TMA'11)*, Jordi Domingo-Pascual, Yuval Shavitt, and Steve Uhlig (Eds.). Springer-Verlag, Berlin, Heidelberg, pp. 176-180, 2011.

[68] W. Louati and D. Zeghlache, "Personal overlay networks management using a P2P-based publish/subscribe naming system," in *Proceedings of the 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pp.95-99, 2008.

[69] L. Mamatas, S. Clayman, M. Charalambides, A. Galis, and G. Pavlou, "Towards an information management overlay for emerging networks," in *Proceedings of IEEE Network Operations and Management Symposium (NOMS)*, pp. 527-534, 2010.

[70] A. Binzenhfer, K. Tutschku, B. auf dem Grabem, M. Fiedler, and P. Carlsson, "A P2P-based framework for distributed network management," in *Proceedings of wireless systems and network architectures in next generation Internet*. *Lecture notes in computer science*, vol 3883. Springer, Berlin, pp. 198–210, 2006.

[71] A. Binzenhöfer and K. Tutschku, "DNA - A P2P-based Framework for distributed network management," in *Proceedings of KiVS Kurzbeiträge und Workshop*, pp. 135-138, 2005.

[72] C. Reich, K. Bubendorfer, and R. Buyya, "An autonomic peer-to-peer architecture for hosting stateful web services," in *Proceedings of the 8th IEEE International Symposium on Cluster Computing and the Grid CCGRID '08.*, pp.250-257, 2008.

[73] D. Li, H. Liu and A. Vasilakos, "An efficient, scalable and robust p2p overlay for autonomic communication," in *Autonomic Communication*, A. V. Vasilakos, M. Parashar, S. Karnouskos and W. Pedrycz, Eds. Springer US, pp. 327-350, 2009.

[74] P. Gouvas, A. Zafeiropoulos, A. Liakopoulos, G. Mentzas, and N. Mitrou, "Integrating overlay protocols for providing autonomic services in mobile ad-hoc networks", *IEICE Communications, Special Issue on: Implementation, Experiments, and Practice for Ad Hoc and Mesh Networks*, vol. E93-B, no. 8, August 2010.

[75] K. Kutzner, C. Cramer, T. Fuhrmann, "Towards autonomic networking using overlay routing techniques," *ARCS*, pp. 222-23*,* 2005.

[76] I. Al-Oqily and A. Karmouch, "Automating overlay networks management," in Proceedings of the 21st International Conference on Advanced Information Networking and Applications, AINA '07, pp.386-393, 2007.

[77] I. Al-Oqily and A. Karmouch, "Policy-based context-aware overlay networks," in *Proceedings of the First International Global Information Infrastructure Symposium,* pp. 85-92, 2007.

[78] I. Al-Oqily and A. Karmouch, "A self-organizing composition towards autonomic overlay networks," in *Proceedings of the IEEE Network Operations and Management Symposium, NOMS*, pp.287-294*,* 2008.

[79] I. Al-Oqily and A. Karmouch, "SORD: A fault-resilient service overlay for mediaport resource discovery," *IEEE Transactions on Parallel and Distributed Systems*, vol.20, no.8, pp.1112-1125, Aug. 2009.

[80] W. Cui, I. Stoica, and R. Katz, "Backup path allocation based on a correlated link failure probability model in overlay networks," in *Proceedings of the 10th IEEE International Conference on Network Protocols*, pp.236–245, 2002.

[81] T. Qiu, E. Chan, and G. Chen, "Overlay partition: iterative detection and proactive recovery communications," in *Proceedings of IEEE International Conference on Communications*, pp. 1854 -1859, 2007.

[82] B. Mathieu, M. Song, M. Brunner, M. Stiemerling, M. Cassim, A. Galis, L. Cheng, K. Jean, R. Ocampo, Z. Lai, and M. Kampmann, "Autonomic management of context-aware ambient overlay networks," in *Proceedings of the Second International Conference on Communications and Networking in China, CHINACOM '07*, pp. 727 -733, 2007.

[83] B. Mathieu, M. Song, A. Galis, L. Cheng, K. Jean, R. Ocampo, M. Brunner, M. Stiemerling, and M. Cassini, "Self-management of context-aware overlay ambient networks," in *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 749 -752, 2007.

[84] K. Markus, M. Bertrand S. Martin, C. Mirko, G. Alex, J. Kerry, O. Roel, B. Kazimierz, A. Muhammad, A. Kamal, P. Mark, B. Bryan, and R. Teemu, "Dynamic adaptable overlay networks for personalized service delivery," in *Proceedings of M2NM,* 2007.

[85] M. Yang and Z. Fei, "A cooperative failure detection mechanism for overlay multicast," *J. Parallel Distrib. Comput*., vol. 67, no. 6, pp.635-64, June 2007.

[86] J. Cao, J. Su, Y. Wang, and Z. Sun, "A Fine-grained parameter configuration model for failure detection in overlay network systems," in *Proceedings of International Conference on MultiMedia and Information Technology*, pp.580-585, 2008.

[87] E. Pournaras, M. Warnier, and F. M. T. Brazier, "Adaptation strategies for self-management of tree overlay networks," in *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing (GRID)*, pp.401-409, 2010.

[88] R. D. Callaway, M. Devetsikiotis, Y. Viniotis, and A. Rodriguez, "An autonomic service delivery platform for service-oriented network environments," *IEEE Transactions on Services Computing*, vol.3, no.2, pp.104-115, Apr.-June 2010.

[89] A. Capone, J. Elias, and F. Martignon, "Routing and resource optimization in service overlay networks," *Comput. Netw*., vol. 53, no. 2, pp. 180-190, Feb. 2009.

[90] K. Park and C. Choi, "Optimization driven bandwidth provisioning in service overlay networks," *Computer Communications*, Vol. 31, no. 14, pp. 3169-3177, Sep. 2008.

[91] Z. Li, L. Yuan, and P. Mohapatra, "An efficient overlay link performance monitoring technique," *IFIP Networking*, pp. 513-524*,* 2006.

[92] Y. Chen, D. Bindel, H. Song, and R.H. Katz, "Algebra-based scalable overlay network monitoring: Algorithms, evaluation, and applications," *IEEE/ACM Transactions on Networking*, vol.15, no.5, pp.1084-1097, Oct. 2007.

[93] Y. Tang, E. Al-Shaer, and B. Zhang, "Toward globally optimal event monitoring & aggregation for large-scale overlay networks," in *Proceedings of the 10th*

IFIP/IEEE International Symposium on Integrated Network Management, pp.236-245, 2007.

[94] W.T. Ooi and R. van Renesse, "The design and implementation of programmable media gateways," in Proceedings of the 16th Int'l Workshop Network and Operating System Support for Digital Audio and Video (NOSSDAV '00), 2000.

[95] Y. Al Ridhaw, I. Abdeljaouad, G. Kandavanam, and A. Karmouch, "An architecture for autonomic management of overlay networks," *2nd IEEE Workshop on Multimedia Communications & Services (MCS 2011), IEEE Globecom*, Houston, Texas, USA*, 2011*.

[96] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, (2001, Mar.). Web services description language (WSDL) 1.1. [Online]. Available: http://www.w3.org/TR/wsdl

[97] S. Herborn, Y. Loez, and A. Seneviratne, "A distributed scheme for autonomous service composition," in *Proceedings of the First ACM Int'l Workshop Multimedia Service Composition (MSC '05)*, pp. 21-30, 2005.

[98] A. Cooke, A. Gray, L. Ma, W. Nutt, J. Magowan, P. Taylor, R. Byrom, L. Field, S. Hicks, and J. Leake, "R-GMA: An information integration system for grid monitoring," in *Proceedings of the 11th International Conference on Cooperative Information*, pp. 462-481, 2003.

[99] S. Monnet, S. Legtchenko, P. Sens, and G. Muller, "RelaxDHT: A churn-resilient replication strategy for peer-to-peer distributed hash-tables," *ACM Trans. Auton. Adapt. Syst.* 7, 2, Article 28, 18 pages.

[100] Y. Al Ridhawi, G. Kandavanam, and A. Karmouch; "A dynamic hybrid service overlay network for service compositions," in *Proceedings of WORLDCOMP'11*, pp. 389-395, 2011.

[101] D. Chopra, H. Schulzrinne, E. Marocco, E. Ivov, "Peer-to-Peer Overlays for Real-Time Communication: Security Issues and Solutions," *IEEE Communications Surveys & Tutorials*, Vol.11, No.1, January 2009.

[102] T. R. Gruber, "A translation approach to portable ontology specifications," in *Proceedings of Knowledge Acquisition*, vol. 5, pp. 199-220, 1993.

[103] N. Samaan and A. Karmouch, "Towards Autonomic Network Management: an Analysis of Current and Future Research Directions", *IEEE Communications Surveys and Tutorials*, vol.11, no.3, pp.22-36, 3rd Quarter 2009.

[104] (2012, Feb.). About skype. Skype Corporation. [Online]. Available: http://about.skype.com

[105] J. Leitao, L. Rosa, and L. Rodrigues, "Large-scale peer-to-peer autonomic monitoring," in Proceedings of the 3rd IEEE Workshop on Distributed Autonomous Network Management Systems, in Conjuction with GLOBECOM, pp. 1-5, 2008.

[106] S. Hart and A. Mas-Colell, "A reinforcement procedure leading to correlated equilibrium," in *Proceedings of Economics essays: a Festschrift for Werner Hildenbrand*, pp. 181–200, 2001.

[107] P. Maymounkov and D. Mazi, "Kademlia: A Peer-to-Peer information system based on the XOR metric," in *Proceedings of Revised Papers from the First International Workshop on Peer-to-Peer Systems (IPTPS '01)*, pp. 53-65, 2001.

[108] B.Y. Zhao, H. Ling, J. Stribling, S.C. Rhea, A.D. Joseph, and J.D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," *IEEE Journal on Selected Areas in Communications*, vol.22, no.1, pp. 41- 53, Jan. 2004.

[109] R. Rodrigues and C. Blake, "When multi-hop peer-to-peer lookup matters," in *Proceedings of the Third international conference on Peer-to-Peer Systems (IPTPS'04)*, Geoffrey M. Voelker and Scott Shenker (Eds.). Springer-Verlag, Berlin, Heidelbe, 2004.

[110] M. Bishop, S. Rao, and K. Sripanidkulchai, "Considering priority in overlay multicast protocols under heterogeneous environments," in *Proceedings of IEEE INFOCOM*, pp. 1–13, Apr. 2006.

[111] F. Bustamante and Y. Qiao, "Friendships that last: Peer lifespan and its role in p2p protocols," in *IEEE WCW*, pp. 233–246, Sept. 2003.

[112] S. Saroiu, P. Gummadi, and S. Gribble, "A measurement study of peerto-peer file sharing systems," in *MMCN*, pp. 156–170, Jan. 2002.

[113] N. Samaan and A. Karmouch, "A mobility prediction architecture based on contextual knowledge and spatial conceptual maps," *IEEE Transactions on Mobile Computing* , vol.4, no.6, pp. 537- 551, Nov.-Dec. 2005.

[114] S. Hart and A. Mas-Colell, "A simple adaptive procedure leading to correlated equilibrium," Econometrica, 68 (2000), pp. 1127–1150.

[115] I. Baumgart, B. Heep, and S. Krause, "OverSim: A flexible overlay network simulation framework," in *Proceedings of IEEE Global Internet Symposium*, pp.79-84, 2007.

[116] A. Varga. (2011). OMNeT++ user manual, Version 4.2. [Online]. Available: http://omnetpp.org/doc/omnetpp/Manual.pdf

[117] F. Dabek, B. Zhao, P. Druschel, J. Kubiatowicz, and I. Stoica, "Towards a common API for structured peer-to-peer overlays," in *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, Volume 2735/2003, pp. 33–44, 2003.

[118] IBM. 2012. IBM ILOG CPLEX Optimizer. http://www-01.ibm.com/software/integration/optimization/cplex-optimizer .

[119] R. Serral-Gracià, E. Cerqueira, M. Curado, M. Yannuzzi, E. Monteiro, and X. Masip-Bruin, "An overview of quality of experience measurement challenges for video applications in IP networks," in *Proceedings of International Conference on Wired/Wireless Internet Connections*, Sweden, pp. 252-263, 2010.

[120] N. Staelens, S. Moens, W. Van den Broeck, I. Mariën, B. Vermeulen, P. Lambert, R. Van de Walle, and P. Demeester, "Assessing quality of experience of IPTV and video on demand services in real-life environments", *IEEE Transactions on broadcasting*, vol. 56, pp. 458 -466, 2010.

[121] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *IEEE Network Magazine*, vol. 24, no. 2, pp.36-41, Mar. 2010.

[122] J. Von Neumann and O. Morgenstern, "Theory of games and economic behaviour," Ed. Princeton: Princeton University Press, 1944.

[123] T. R. Jain and V.K. Ohri, "Principles of Microeconomics," V.K. Publications: New Delhi, 2009-10.

[124] S. Shenker, "Fundamental design issues for the future internet," *IEEE Journal on Selected Areas in Communication*, vol. 13, pp. 1176-1188, 1995.

[125] Z. Cao and E.W. Zegura, "Utility max-min: An application-oriented bandwidth allocation scheme," in *Proc. IEEE Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. INFOCOM '99*, vol.2, pp.793-801, 1999.

[126] N. Lu and J. Bigham, "An Optimal bandwidth adaptation algorithm for multi-class traffic in wireless networks," in *Proceedings of the 3rd international Conference on Quality of Service in Heterogeneous Wired/Wireless Networks*. *QShine '06*, vol. 191, 2006.

[127] V. Rakocevic, "Dynamic bandwidth allocation in multi-class IP networks using utility functions," PhD Thesis, pp. 90-100, Queen Mary, University of London, UK, 2002.

[128] L. Breslau and S. Shenker, "Best-effort versus reservations: A simple comparative analysis," in Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication SIGCOMM '98, 1998.

[129] E. Meshkova, J. Riihijarvi, A. Achtzehn, and P. Mähönen, "On utility-based network management," Tech. Rep. RWTH/iNETS/Meshkova, June 2010.

[130] M. Xiao, N. B. Shroff, and E. K. P. Chong, "A utility-based power-control scheme in wireless cellular systems," *IEEE/ACM Transactions on Networking*, vol.11, no.2, pp. 210-221, 2003.

[131] Q. Nguyen-Vuong, Y. Ghamri-Doudane, and N. Agoulmine, "On utility models for access network selection in wireless heterogeneous networks," in *Proceedings of IEEE Network Operations and Management Symposium, NOMS*, pp.144-151, 2008.

[132] S. Pal, S.K. Das, and M. Chatterjee, "User-satisfaction based differentiated services for wireless data networks," in *Proceedings of IEEE International Conference on Communications*, vol.2, pp. 1174- 1178 , 2005.

[133] M. Fiedler, S. Chevul, O. Radtke, K. Tutschku, and A. Binzenhöfer, "The network utility function: A practicable concept for assessing network impact on distributed services," in *Proceedings of the 19th International Teletraffic Congress (ITC19)*, 2005.

[134] M. Fiedler, K. Tutschku, S. Chevul, L. Isaksson, and A., Binzenhöfer, "The throughput utility function: Assessing network impact on mobile services," in *Proceedings of the 2nd EuroNGI IA.8.2 Workshop*, Springer (LNCS 3883), 2005.

[135] G. D. Stamoulis, D. Kalopsikakis, and A. Kyrikoglou, "Efficient agent-based negotiation for telecommunications services," in *Proceedings of IEEE Global Telecommunications Conference, GLOBECOM '99*, 1999, vol.3, pp. 1989-1996.

[136] L. A. DaSilva, "Pricing for QoS-enabled networks: A survey," in *Proceedings of IEEE Communications Surveys & Tutorials*, vol. 3, no. 2, pp. 2-8, 2000.

[137] P. Reichl, "From charging for quality of service to charging for quality of experience," in *Annales des Télécommunications*, vol. 65, no. 3-4, pp. 189-199, 2010.

[138] DARPA and NSF. (2012, Feb.). The network simulator - ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns

[139] J. Klaue, B. Rathke, and A. Wolisz, "EvalVid - A framework for video transmission and quality evaluation," in *Proceedings of the International Conference on Modeling Techniques and Tools for Computer Performance Evaluation*, pp. 255-272, 2003.

[140] MSU Video Group. (2012, Feb.). *Video Filtering and Compression*. [Online]. Available: http://www.compression.ru/video/index.htm

[141] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600-612, Apr. 2004.

[142] Arizona State University. (2012, Feb.). YUV video sequences. *Arizona State University*. [Online]. Available: http://trace.eas.asu.edu/yuv/

[143] Hughes Systique. IPTV. [Online]. Available: http://wiki.hsc.com/wiki/Main/IPTV

[144] Cisco. Optimizing video transport in your IP triple play network. [Online]. Available: http://www.cisco.com/en/US/prod/collateral/routers/ps368/prod_white_paper0900aecd80478c12.html

[145] I. Abdeljaouad and A. Karmouch, "Self-Organizing and Self-Adapting Autonomic Managers for Service Specific Overlay Networks Management," Submitted to *ACM Transactions on Autonomous and Adaptive Systems*.

[146] I. Abdeljaouad and A. Karmouch, "Self-Organizing Autonomic Overlay Networks for Monitoring IPTV Quality of Experience Using Utility Functions," Submitted to *Computer Communication*.

[147] I. Abdeljaouad and A. Karmouch, "Utility Function for Predicting IPTV Quality of Experience Based on Delays in Overlay Networks," accepted by *2013 IEEE Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, USA, 11-14 Jan. 2013.

[148] I. Abdeljaouad and A. Karmouch, "Self-Organizing Overlay Networks for Autonomic Manager Selection," accepted by *3rd IEEE International Workshop on SmArt COmmunications in NEtwork Technologies ('ICC'12 WS - SaCoNet-III')*, *IEEE ICC 2012*, Ottawa, ON, Canada, 10-15 June 2012.

[149] Y. Al Ridhaw, I. Abdeljaouad, and A. Karmouch, "Service Specific Overlay Composition and Reconfiguration," *2012 International Conference on Multimedia Computing and Systems (ICMCS),* pp.479-484, 10-12 May 2012.

[150] I. Abdeljaouad, G. Kandavanam, and A. Karmouch, "A loss-based utility function for predicting IPTV quality of experience over an overlay network," *IEEE Globecom,* Houston, Texas, USA*, 2011*.

[151] Y. Al Ridhaw, I. Abdeljaouad, G. Kandavanam, and A. Karmouch, "An Architecture for Autonomic Management of Overlay Networks," *2nd IEEE*

Workshop on Multimedia Communications & Services (MCS 2011), IEEE Globecom, Houston, Texas, USA*, 2011*.