FLAMINGO

European Seventh Framework Network of Excellence

http://www.fp7-flamingo.eu/

WP6 — Automated Configuration and Repair

Deliverable D6.2 — Second Year Report on Automated Configuration and Repair

University of Twente, The Netherlands (UT)
Institut National de Recherche en Informatique et Automatique, France (INRIA)
University of Zurich, Switzerland (UZH)
Jacobs University Bremen, Germany (JUB)
Universität der Bundeswehr München, Germany (UniBwM)
University Politecnicà de Catalunia, Spain (UPC)
iMinds, Belgium (iMinds)
University College London, United Kingdom (UCL)

Project funded by the European Union under the
Information and Communication Technologies FP7 Cooperation Programme
Grant Agreement number ICT-FP7 318488

Document Control

Title:	D6.2 — Second Year Report on Automated Configuration and Repair
Type:	Public
Editor(s):	Gabi Dreo Rodosek
E-mail:	gabi.dreo@unibw.de
Doc ID:	D6.2
Delivery Date:	31.10.2014
Author(s):	Anthéa Mayzaud, Anuj Sehgal, Gaëtan Hurel, Gabi Dreo, Christos Tsiaras, Anna Sperotto, Daniel Dönni, Daphne Tuncer, Marinos Charalambides, Mario Flores, Jeroen Famaey, Mario Golling, Maxim Claeys, Niels Bouten, Nikolay Melnikov, Radhika Garg, Rashid Mijumbi, Ricardo Schmidt, Frank Tietze, Rick Hofstede, Sebastian Seeber, Steven Latré, Corinna Schmitt, Abdelkader Lahmadi, Jair Santanna, Stefano Petrangeli, Peter Hillmann, Bram Naudts, Javier Rubio-Loyola, Sofie Verbrugge

For more information, please contact:

Dr. Aiko Pras
Design and Analysis of Communication Systems
University of Twente
P.O. BOX 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893778
Fax: +31-53-4894524
E-mail: <a.pras@utwente.nl>

Legal Notices

Executive Summary

The Future Internet, Internet of Things or Internet of Anything are emerging trends that pose new challenges to the management of a complex, networked environment. Due to the complexity, management needs to be part of the functionality of the managed objects instead of a thought after. To cope with this, it is necessary to think of a new management paradigm, namely Management-by-Design. Furthermore, a necessity of the Management of the Future Internet is the automation of tasks and processes, especially due to the trend towards a myriad of networked devices with a broad range of capabilities (from a simple dumb sensor to the smart space, from isolated clouds to Inter-Clouds) and the dramatically increased complexity of smart networked environments. WP6 addresses this automation aspect.

Deliverable D6.2 reports on achievements reached within WP6 in its second year, focusing on enablers, like knowledge processing, information modeling, and learning techniques. With respect to the S.M.A.R.T. (Specific, Measurable, Achievable, Relevant, Timely) objectives (Section B.1.1.5 of the Description of Work) we claim the full achievement of our second year targets for WP6. Due to the tight collaboration of this work package with other work packages like WP1 and WP5, that has even been more intense in the second year, several PhD students, although not financially paid by FLAMINGO but jointly supervised, are directly contributing to the WP6 specific objectives. As a scientific output, we can report a total of 50 papers that have been already published in Y2, and several other papers that are currently under review. This scientific output is exceeding the objective initially set in the DoW of 20 papers. We point to Deliverable D8.2 for details.

The work package specific objectives center around the following three tasks, namely (i) to develop innovative architectural approaches for automated configuration and repair (Task 6.1), (ii) to identify enablers for these new architectures (Task 6.2) and (iii) to analyze the applicability of the developed approaches to selected application domains (Task 6.3). Key achievements of WP6 in the second year, as specified in the DoW and as documented in D6.2, are summarized below:

Task 6.1: Architectures Whereas the focus of the first year was building an inventory of architectures and approaches in the area of automation, and the development of a first architecture in the area of intrusion detection systems, Y2 was focused on the further refinement of this architecture as well as developing a new one for software defined networking (SDN). Due to the changes in the DOW by adding SDN as a new application area, another architecture for extrusion detection in this field has been developed. Both architectures can be seen as building blocks, addressing various aspects of the FLAMINGO integrative architecture for automated configuration and repair.

Task 6.2: Enablers An inventory of enablers and approaches in the field of network and service management has been set up in Y2. An overall objective of this task also is to provide guidelines for using enablers in application domains. Therefore, besides just collecting enablers and approaches, a taxonomy has been developed, and the identified approaches were classified with respect to this taxonomy. Furthermore, the specifics of enablers and application domains have been identified and matched.

Task 6.3: Application Domains A detailed enabler-centric description of four application areas, namely (i) wireless sensors networks, (ii) cloud-based services, (iii) content-aware networking and (iv) SDN was in the focus of the second year. The objective behind this effort is to identify specifics of enablers that are best applicable for a specific application domain.

To summarize, we claim that all S.M.A.R.T as well as work package specific objectives in the second year have been fully achieved.

Contents

1 Introduction

Another essential aspect for the Management of the Future Internet is to automate management actions to which WP6 is devoted to. Deliverable D6.2 reports on achievements of WP6 in the second year of the project. Therefore, S.M.A.R.T(Specific, Measurable, Achievable, Relevant, Timely) objectives as well as WP6-specific objectives are addressed.

The first S.M.A.R.T. objective is the integration of PhD students. For a detailed list of the fully integrated PhD students, we refer to Deliverable D8.2. Many PhD collaborations within the consortium, which started during the first year are still ongoing. In addition, several new collaborations started in the second year. We refer to Section 4 for more details.

The second S.M.A.R.T. objective refers to the scientific output of the project. In Y2, the research work packages published in total 50 papers at major conferences and in journals. We report this summarized number in all research WP deliverables due to the tight research integration of WP5, WP6 and WP7 which is also manifested in the joint publications. Monitoring data addressed in WP5 is used as input for the automation approaches dealt within WP6. In addition, WP6 is able to reconfigure the monitoring systems to adjust them with respect to the automated actions. Furthermore, WP7 verifies that monitoring (WP5) and automated (WP6) actions are all performed within the boundaries of the economic, legal and regulative constraints. For a detailed list of the FLAMINGO published and submitted papers, we refer to Deliverable D8.2.

The work package activities are structured around three focal points: (i) the development of architectures, (ii) the identification of enablers such as knowledge description and discovery, learning techniques, algorithms, and (iii) the analysis of application domains to validate the developed concepts.

Deliverable D6.2 is structured as follows: Achievements related to S.M.A.R.T and WP-specific objectives are summarized in Section 2. Section 3 includes details about architectural aspects (Section 3.1 to 3.4), enablers (Section 3.5) and application domains (Section 3.6 and 3.7) investigated within Y2 of WP6.

Regarding the architectures the deliverable presents the results from the architectural analysis started last year, taking into account the first draft of the FLAMINGO Automation Architecture in the Area of Intrusion Detection Systems. Due to the update of the description of work - adding the application domain SDN - another architecture in this area for extrusion detection has been developed.

Since PhD collaborations form the basis of the research work done in FLAMINGO, Section 4 presents the PhD contributions and describes each ongoing collaboration in detail.

The overall joint research of FLAMINGO is sketched in the Framework to Support the Combination of Policy-Based and Semantic-Based Approaches described in Section 5.

Section 6 concludes the deliverable.

2 Objectives and Activities

This section presents an overview of the S.M.A.R.T. objectives for WP6. For each S.M.A.R.T. objective, we indicate how it was achieved in the reported year of the project. WP6-specific objectives summarize the activities that have taken place among the consortium members in Y2.

2.1 S.M.A.R.T. Objectives

To meet the S.M.A.R.T. objectives, WP6 has been active in the following aspects.

- **Integration of PhD students** – *The Description of Work (Section B.1.1.5) states that "after 9 months each research WP will have identified at least two fully integrated Ph.D. students, which means that these students will be jointly supervised and financially paid by FLAMINGO".*
  In the first year of the project, seven PhD students have joined FLAMINGO as fully integrated PhD students. In the second year, 7 more Ph.D. students have joined the NoE. These students, their affiliations and the co-supervising institutions are listed in D8.2. Collaborations are a cornerstone of research within FLAMINGO. All PhD students are encouraged to work jointly together within the project, contributing different aspects of the research. Following this, no student can be mapped only to one WP. It is important that collaborations are not only taking place between fully integrated PhD students, but also among students that are not financially paid by FLAMINGO but jointly supervised. Detailed information on the integration of PhD students can be found in Section 4.1.

- **Scientific Output** – *The Description of Work (Section B.1.1.5) states that "after 18 month at least 20 scientific papers will be submitted / published".*
  In the first year, the project had exceeded the expected number of publications. In the second year the research work packages published 50 papers at major conferences as well as in journals, and with this exceed the expected number of papers. Furthermore, several other papers are currently under review. The strong collaboration between the WPs is based on the intensive PhD collaborations described in Section 4.1. There is a special intense collaboration between WP5 and WP6 since the monitoring data (WP5) builds the basis for numerous approaches addressed in WP6. Because several publications address aspects of WP5 and WP6 it is difficult to assign papers to only one WP. Therefore, a joint list of papers is included in Deliverable D8.2.

  Tables 1 and 2 report the result of collaborations with other European projects and institutions. Within the FLAMINGO consortium the published papers co-authored by more than one FLAMINGO member are listed in Table 3. Partners also targeted top conferences and journals in the network management field and high-end conferences and journals in the field of networking and measurements as suggested by the reviewers during the first evaluation. To address this, papers have been published at IEEE INFOCOM 2014, Internet Measurement Conference (IMC 2014) and ACM SIGCOMM Computer Communication Review (CCR).

Table 1: FLAMINGO publications in collaboration with other EU projects and institutions.

Authors	Title	Venue	EU project/ institution
R. de O. Schmidt, R. Sadre, A. Sperotto, H. van den Berg, and A. Pras	A hybrid procedure for efficient link dimensioning	Elsevier Computer Networks	Aalborg University
R. de O. Schmidt, R. Sadre, N. Melnikov, J. Schönwälder, and A. Pras.	Linking network usage patterns to traffic Gaussianity fit	IFIP Networking 2014	Aalborg University
F. Francois, N. Wang, K. Moessner, S. Georgoulas, and R. de O. Schmidt	Leveraging MPLS Backup Paths for Distributed Energy-Aware Traffic Engineering.	TNSM	University of Bristol, University of Surrey
I. Drago, R. de O. Schmidt, R. J. Hofstede, A. Sperotto, M. Karimzadeh, B. R. H. M. Haverkort, and A. Pras	Networking for the Cloud: Challenges and Trends	PIK - Praxis der Informationsver-arbeitung und Kommunikation	Politecnico di Torino, Mobile Cloud Networking
R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, and A. Pras	Flow Monitoring Explained: From Packet Capture to Data Analysis with NetFlow and IPFIX	IEEE Communications Surveys & Tutorials	Masaryk University, ETH, Politecnico di Torino, Aalborg University
G. van den Broek, R. van Rijswijk-Deij, A. Sperotto, and A. Pras	DNSSEC meets real world: dealing with unreachability caused by fragmentation	IEEE Communications Magazine	SURFnet BV
R. van Rijswijk-Deij, A. Sperotto, and A. Pras	DNSSEC and Its Potential for DDoS Attacks	ACM IMC 2014	SURFnet BV
C. Tsiaras, A. Sehgal, S. Seeber, D. Dönni, B. Stiller, J. Schönwälder, and G. Dreo Rodosek	Towards Evaluating Type of Service Related Quality-of-Experience on Mobile Net- works	WMNC 2014	SmartenIT
C. Tsiaras, S. Liniger, and B. Stiller	Automatic and on-demand Mobile Network Operator (MNO) selection mechanism demonstration	NOMS 2014 (Demo paper)	SmartenIT
C. Tsiaras, S. Liniger, and B. Stiller	An automatic and on-demand MNO selection mechanism	NOMS 2014	SmartenIT
C. Tsiaras, and B. Stiller	A Deterministic QoE Formalization of User Satisfaction Demands (DQX)	LCN 2014	SmartenIT
R. Mijumbi, J.L. Gorricho, and J. Serrat	Contributions to Efficient Resource Management in Virtual Networks	AIMS 2014	EVANS

Table 2: FLAMINGO publications in collaboration with other EU projects and institutions (ctd.).

Authors	Title	Venue	EU project/ institution
R. Mijumbi, J. Serrat, J.L. Gorricho, M. Claeys, F. De Turck, and S. Latré	Design and evaluation of learning algorithms for dynamic resource management in virtual networks	NOMS 2014	EVANS, University of Antwerp
R. Mijumbi, J.L. Gorricho, J. Serrat, M. Claeys, J. Famaey, and F. De Turck	Neural Network-based Autonomous Allocation of Resources in Virtual Networks	EUCNC 2014	EVANS
M. Claeys, S. Latré, J. Famaey, and F. De Turck	Design and evaluation of a self-learning http adaptive video streaming client	IEEE Communications Letters	University of Antwerp
N. Bouten, M. Claeys, S. Latré, J. Famaey, W. Van Leekwijck, and F. De Turck	Deadline-based Approach for Improving Delivery of SVC-based HTTP Adaptive Streaming Content	QCMan 2014	University of Antwerp
J. T. Araújo, R. Landa, R. G. Clegg, and G. Pavlou	Software-defined network support for transport resilience	NOMS 2014	ALIEN
J. T. Araújo, R. Landa, R. Clegg, G. Pavlou, and K. Fukuda	A longitudinal analysis of Internet rate limitations	INFOCOM 2014	NII International Internship Program
P. Porambage, C. Schmitt, A. Gurtov, and S. Gerdes	PAuthKey: A Pervasive Authentication Protocol and Key Establishment Scheme for Wireless Sensor Networks in Distributed IoT Applications	International Journal of Distributed Sensor Networks	SmartenIT, Oulu University and Aalto University
M. Claeys, D. Tuncer, J. Famaey, M. Charalambides, S. Latré, F. De Turck, and G. Pavlou	Towards Multi-tenant Cache Management for ISP Networks	EUCNC 2014	University of Antwerp
M. Karimzadeh, A. Sperotto, and A. Pras	Software Defined Networking to Improve Mobility Management Performance	AIMS 2014	Mobile Cloud Networking
C. Schmitt,T. Kothmayr, B. Ertl, W. Hu, L. Braun, and G. Carle	Tiny IPFIX: An efficient application protocol for data exchange in cyber physical systems.	Elsevier Computer Communications	TU Munich and TU Berlin

Table 3: Publications authored by multiple FLAMINGO partners.

Authors	Title	Venue	FLAMINGO partners
R. de O. Schmidt, R. Sadre, N. Melnikov, J. Schönwälder, and A. Pras	Linking network usage patterns to traffic Gaussianity fit	IFIP Networking 2014	UT, JUB
A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder	A Study of RPL DODAG Version Attacks	AIMS 2014 **Best Paper Award**	INRIA, JUB
C. Tsiaras, A. Sehgal, S. Seeber, D. Dönni, B. Stiller, J. Schönwälder, and G. Dreo Rodosek	Towards Evaluating Type of Service Related Quality-of-Experience on Mobile Networks	WMNC 2014	UZH, JUB, UniBwM
A. Sehgal, A. Mayzaud, R. Badonnel, I. Chrisment,and J. Schönwälder	Addressing DODAG Inconsistency Attacks in RPL Networks	GIIS 2014	JUB, INRIA
M. Golling, R. Hofstede, and R. Koch	Towards Multi-layered Intrusion Detection in High-Speed Backbone Networks	CyCon 2014	UniBwM, UT
R. Mijumbi, J. Serrat, J.L. Gorricho, M. Claeys, F. De Turck, and S. Latré	Design and evaluation of learning algorithms for dynamic resource management in virtual networks	NOMS 2014	UPC, iMinds
R. Mijumbi, J.L. Gorricho, J. Serrat, M. Claeys, J. Famaey, and F. De Turck	Neural Network-based Autonomous Allocation of Resources in Virtual Networks	EUCNC 2014	UPC, iMinds
M. Claeys, D. Tuncer, J. Famaey, M. Charalambides, S. Latré, F. De Turck, and G. Pavlou	Towards Multi-tenant Cache Management for ISP Networks	EUCNC 2014	iMinds, UCL
M. Golling, R. Koch, P. Hillmann, R. Hofstede, and F. Tietze	YANG2UML: Bijective Transformation and Simplification of YANG to UML	CNSM 2014	UniBwM, UT

2.2 Work Package Specific Objectives

Inside FLAMINGO each work package has its own defined objectives. This section reports on the WP6-specific objectives and the achievements during the second year.

2.2.1 Ongoing Objectives

OBJECTIVE 1 - To integrate European research in the area of automated configuration and repair: In cooperation with WP3 and WP5, WP6 has taken part in several activities at European level in the area of automated configuration and repair. WP6 supported the organization of the 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2014). Deliverable D3.3 provides additional information about this event. Furthermore, the number of 22 publications with other European partners undermines joint activities with different European projects. WP6 was also active in the organization of EuCNC workshops *Management of Large Scale Virtualized Infrastructures: Smart Data Acquisition, Analysis and Network and Service Management in the Future Internet* [1] and *Mobile Cloud Infrastructures and Services (MCIS)*[2]. In addition, during these sessions Filip De Turck (iMinds) and Marinos Charalambides (UCL) contributed with presentations. Furthermore, a TNSM Special Issue on "Efficient Management of SDN and NFV-based Systems" is organized by Filip De Turck.

OBJECTIVE 2 - To create and maintain articles within Wikipedia and other online systems in this area: The ongoing research generated valuable insights that have been contributed to Wikipedia. WP6 contributed to Wikipedia for example the following pages SOFTWARE DEFINED NETWORKING [3], NETFLOW[4], and SFLOW[5]. Detailed information about Wikipedia editing within FLAMINGO can be found in Deliverable D2.2.

OBJECTIVE 3 - To develop an inventory of approaches for automated configuration and repair: In Y2 WP6 has focused on building an inventory of enablers. Details about the inventory are available in Section 3.5. In addition, to the inventory several other databases have been set up. The collaboration **UT-UniBwM-IDS** built up a database of IDS message exchange protocols which support their collaboration work on a multi-layered intrusion detection system. **INRIA-JUB-RPL** has continuously updated their database of attacks on RPL networks. This database supports also the collaboration **INRIA-JUB-Distr** which develops a distributed monitoring architecture in the area of IoT. Furthermore, the collaboration **INRIA-UniBwM-Cloud** started building a database of network attacks specific for SDN environments.

OBJECTIVE 4 - To specify guidelines about the applicability of approaches for automated configuration and repair to specific application domains: Section 3.6 reports on limitations in application domains regarding the applicability of functions and methodologies, especially in respect to enablers, used within FLAMINGO. The collaboration **iMinds-UPC-NetVirt** applied three machine learning techniques to address resource allocation problems. Especially, they investigated the limitations of resource allocation in virtual networks. The application domain content-aware-routing was addressed in **UCL-iMinds-Cache** by investigating requirements to develop a

[1] http://www.eucnc.eu/?q=node/113
[2] http://www.eucnc.eu/?q=node/114
[3] http://en.wikipedia.org/wiki/Software-defined_networking
[4] http://en.wikipedia.org/wiki/NetFlow
[5] http://en.wikipedia.org/wiki/SFlow

proactive mechanism to efficiently manage the utilization of network resources. Requirements for cloud-based services have been investigated in collaboration **UT-UniBwM-IDS**.

OBJECTIVE 5 - To develop new architectures for automated configuration and repair approaches across administrative boundaries: During Y2 the collaboration **UCL-iMinds-Cache** developed and evaluated an approach for a Video-on-Demand use case that can be used across administrative boundaries. Using this approach, the average total bandwidth usage inside the ISP network is reduced substantially. The collaboration **UT-UniBwM-IDS** developed a first ”automated” intrusion detection architecture that can be used in different administrative boundaries.

OBJECTIVE 6 - To develop information models, algorithms, learning techniques and knowledge description approaches as enablers for automated configuration and repair: The description and investigation of enablers including information models, algorithms, learning techniques and knowledge description approaches are a core part of D6.2 and are reported in Section 3.5. Several collaborations have addressed this objective. **UCL-UT-Man** developed an algorithm to balance the load between several line cards achieving at the same time energy gains. The collaboration **iMinds-UT-QoS** developed a distributed algorithm to be able to divide resources among various HTTP Adaptive Streaming (HAS) clients. Each client performs a local optimization and shares the local network information and the local decision with different distributed agents to automatically react to changes in the network environment. **UZH-JUB-UniBwM-M2** uses information models to collect and analyze results of mobile measurements where in addition algorithms to compute the MOS value are applied. Reinforcement Learning, Artificial Neural Networks and Neuro-fuzzy Algorithms are applied in the collaboration **iMinds-UPC-NetVirt** to address resource allocation in virtual networks. To solve service chaining and composition problems, **INRIA-UniBwM-Cloud** is using graph theory to verify the path that a packet traversed.

OBJECTIVE 7 - To evaluate automated configuration and repair approaches as being part of the autonomic control loops: Cache Management approaches (**UCL-iMinds-Cache**) were further developed in Y2 by focusing on the application of further Integer Linear Program (ILP) approaches ([1], [2]) to efficiently manage the utilization of network resources which is part of a control loop for ISP content delivery services. Furthermore, the resource management approach developed in **UCL-UT-Man** was evaluated by showing that substantial energy gain can be achieved without significantly compromising the balance of the network in terms of load [3].

OBJECTIVE 8 - To apply policy-based and semantic-based approaches for automated configuration and repair: The collaboration **UCL-iMinds-Cache** addressed policy- and semantic-based approaches. In Y2 the collaboration evaluated an approach that recalculates the optimal content placement and routing paths every time a change in the request pattern occurs. This approach can serve as a benchmark of which performance can be achieved using proactive content placement in a multi-tenant scenario. The results using the theoretical approach are presented in [2]. In addition, this approach has been relaxed to perform the content placement and server selection at fixed time points. The evaluation can be found in [1].

OBJECTIVE 9 - To propose and study automated configuration and repair in the context of the management of clouds (especially Inter-Clouds): Management of clouds, especially Inter-Clouds has been addressed during Y2 in the collaboration **UT-UniBwM-IDS** by using cloud-based

solution requirements for cloud-based services. Furthermore, architectural approaches specific for this application domain have been partially developed. Mobile cloud security is a topic that has been addressed by an approach described in [4]. In addition, a method to automatically repair/remediate vulnerabilities is proposed in [5], which is also applicable in cloud environments.

OBJECTIVE 10 - To apply the developed approaches to several application domains such as of (i) wireless sensor networks, (ii) cloud-based services, (iii) content-aware networking and (iv) software defined networking: Wireless sensor networks are addressed by the collaborations **INRIA-JUB-RPL**, **UniBwM-JUB-RPL** and **INRIA-JUB-Distr**. The first two focus on investigations regarding security aspects of RPL and propose solutions to overcome possible shortcomings. These solutions will probably be evaluated on IEEE 802.15.4 + 6LoWPAN platforms. The last collaboration broadens the scope to a more general consideration in the area of Internet of Things (IoT). The application domain of cloud-based services is addressed by **UT-UniBwM-IDS** in the development of architectural approaches for a multi-layered intrusion detection system. Cache management approaches developed by **UCL-iMinds-Cache** focusing on multi-tenant content placement and server selection refer to the application domain of content-aware networking. The application domain of SDN is addressed by the collaboration **iMinds-UPC-NetVirt** where the control plane of an SDN enabled device is used to efficiently manage resources in virtualized networks by dynamically adjusting the virtual network (VN) to substrate network (SN) mappings based on the current network status. In addition, the collaboration **INRIA-UniBwM-Cloud** investigates recently available SDN-based mechanisms for delivering security in different network scales.

2.3 Tasks and Objectives Mapping

S.M.A.R.T objectives related to WP6 (Section 2.1) and WP6-specific objectives (Section 2.2) are summarized in Table 4. For each of the addressed objectives, Table 4 indicates if the objective has been achieved (S.M.A.R.T. objectives) or if there are WP activities that are contributing to the objective (WP6-specific objectives). For the WP6-specific objectives, Table 4 shows to which of the tasks in the DoW the objective is contributing to. Finally, the table acts as a guide for the reader to locate the sections of this deliverable that provide additional information on a specific objective. Furthermore, Table 5 presents a summary of all objectives and their progress (Y1 to Y2).

Table 4: Objectives and tasks.

Objective	Task 6.1	Task 6.2	Task 6.3	Status	Additional Material
S.M.A.R.T. Objective 1				Achieved	Section 4.1, D 8.2
S.M.A.R.T. Objective 2				Achieved	D 8.2
WP Objective 1				Ongoing	Section 4.1
WP Objective 2				Ongoing	D 2.2
WP Objective 3	X			Ongoing	Section 3.2
WP Objective 4			X	Ongoing	Section 3.6
WP Objective 5	X			Ongoing	Section 3.2
WP Objective 6		X		Ongoing	Section 3.5
WP Objective 7	X	X		Ongoing	Section 3.5
WP Objective 8	X		X	Ongoing	Section 3.5
WP Objective 9	X		X	Ongoing	Section 3.5
WP Objective 10			X	Ongoing	Section 3.5

Table 5: Progress in Y2

Objective	Y1 activities	Y2 activities
S.M.A.R.T. Obj 1	7 Ph.D.	14 Ph.D.
S.M.A.R.T. Obj 2	37 papers	50 papers
WP Objective 1	AIMS; Dagstuhl IM; CNSM; Coll. EU level	Dagstuhl; AIMS; EuCNC MCIS; Coll. EU level; TNSM ”Efficient Mgmt. SDN/NFV”
WP Objective 2	not addressed	wikipedia: SDN, NetFlow, sFlow
WP Objective 3	inventory of architectures, IDS exchange protocols	inventory of enablers, RPL attacks, SDN network attacks
WP Objective 4	guidelines for cloud-based services, content-aware routing	limitations in cloud-based services, content-aware routing with respect to enablers
WP Objective 5	architecure for cloud-based security services	architecure for VoD, automated intrusion detection
WP Objective 6	enablers for IDS, RPL security, network virtualization, QoE, QoS, traffic estimation	enablers for line card load balancing, HAS resource allocation, ressource allocation with machine learning
WP Objective 7	adaptive, energy-aware resource managent	ILP in cache management, network resource utilization mgmt.
WP Objective 8	content placement in CDNs according to policies	proactive content placement in multi-tenant scenarios
WP Objective 9	inter-cloud security systems, VoIP security in cloud scenarios	architectural approaches for cloud-based security, mobile cloud security
WP Objective 10	RPL on IEEE 802.15.4 + 6LoWPAN, TelosB, WSN, content placement in CDNs	Internet of Things, cloud-based multi-layered intrusion detection, virtualized networks in SDN, SDN-based security mechanism

3 Automated Configuration and Repair

3.1 Results on Automated Configuration and Repair Architecture

Since automated configuration and repair architectures were in the focus of the first year of FLAMINGO, this section reports on results not yet presented in Deliverable D6.1. Table 6 summarizes all architectures and their respective properties, addressed in Y1, including the developed FLAMINGO Automation Architecture in the Area of Intrusion Detection Systems. The progress of the deployment of this architecture is described in Section 3.3. As seen in Table 6, the proposed architecture needs to be further developed with respect to degree of automation, type of action (reactive/proactive) and applicability. Further steps in the deployment of this architecture are described in 3.2.

Table 6: Comparison of architectures in the area of automated configuration and repair, not yet included in D6.1

		ANA	ANEMA	CASCADAS	FOCALE	SELFMAN	SCAP	DACoRM	FLAMINGO (D6.1)
Time Horizon	Operational	X	X	X	X	X	X	X	X
	Tactical								
	Strategical								
Support of Fault Management	High								
	Medium	X	X	X	X	X	X	X	X
	Low								
Support of Configuration Management	High								
	Medium	X	X	X	X	X	X	X	X
	Low								
Support of Accountability Management	High								
	Medium	X	X	X			X	X	X
	Low				X	X			
Support of Performance Management	High								
	Medium	X	X	X	X	X	X	X	X
	Low								
Support of Security Management	High								X
	Medium	X	X	X			X	X	
	Low				X	X			
Management Layer	Enterprise				X				
	Service								
	Application								
	System			X					
	Network	X	X			X	X	X	X
	Element								
Applicability	Generic/ Universal	X			X				
	Multi-Domain					X			X
	Specific Domain		X	X			X	X	
Degree of Automation	Autonomous Autonomic Self*	X		X	X				
	Automated		X			X	X	X	X
	Manual								
Action	Proactive								
	Reactive	X	X	X	X	X	X	X	X
Memory	History								
	Recent Developments	X	X						X
	Current State			X	X	X	X	X	
Degree of Hierarchy	Hierarchy		X	X	X		X		X
	No (Peer)	X				X		X	

3.2 FLAMINGO Integrative Architecture for Automated Configuration and Repair

One of the main goals of WP6 until the end of the project is the development of the FLAMINGO integrative architecture for automated configuration and repair. Therefore, the consortium works on an integrative approach with a main focus on inter-domain aspects by discussing models and specifications for automated configuration and repair architectures across domain boundaries. To achieve the overall goal, our strategy focuses on examining architecture aspects from different application domains. The architecture presented last year in D6.1 (Section 5 - First Draft of a FLAMINGO Automation Architecture in the Area of Intrusion Detection Systems) addressed inter-domain aspects which clearly address the application domain cloud computing. The huge amount of data to be processed and exchanged in cloud environments was the reason to use mainly flow based monitoring as a basis for various security analysis and intrusion detection.

SDN as a new technology and an emerging key aspect for the Future Internet are considered as a necessity to be investigated as well within the consortium and represented in the update of the DoW. The ability to directly develop new algorithms inside the network, due to the usage of SDN, is a big step ahead in the area of automation where humans take care of a few higher-level decisions while adequate systems are responsible for a couple of lower-level decisions and actions. Therefore, in Y2 we developed an architecture focusing on SDN. Like in the last year, we started developing the architecture with a bottom-up approach. With respect to the strong integration of WP5 and WP6 and following the interdependence between monitoring and configuration and repair, WP6 extends in Y2 the FLAMINGO Monitoring Architecture to integrate aspects of SDN and support extrusion detection systems. Details about the FLAMINGO Automation Architecture in the Area of Extrusion Detection Systems are described in Section 3.4. Figure 1 describes the development of the FLAMINGO Integrative Architecture in the next years.

Within the next years of FLAMINGO we will focus on additional application domains to study each domain requirements since these play a major role in developing a generic integrative architecture for automated configuration and repair. After investigating the requirements and limitations of each application domain, an integrative approach facing automation, inter-domain aspects and generalizability can be approached. Furthermore, similar research questions will center around protocols for the information and configuration exchange across domain boundaries.

3.3 FLAMINGO Automation Architecture in the Area of Intrusion Detection Systems, Progress in Y2

This section resumes the architecture proposed in the Deliverable D6.1 shortly and highlights the progress achieved throughout the second year of FLAMINGO initiated through the active collaborations in WP5 and WP6 (see Figure 2). Since automated configuration and repair is a main aspect in WP6 and therefore of D6.2, these aspects build the core of this section.

Monitoring data is the basis for each configuration or repair action. These monitoring data, from which a knowledge base as well as information models can be derived is used as input for learning techniques, correlation models and of course forms a knowledge base to store annotated information. The collaboration **UT-INRIA-Flowid** collects monitoring data from mobile devices and is focused on the paths where the malicious traffic gets routed through. In Y2 the focus was on improving the collection and analysis model to fit in an IETF Internet-Draft describing a set of information elements for IPFIX metering process location. As a result the flow exporter, developed in this collaboration, emerged to be more convenient in exchanging data with other components and across domain boundaries. In addition, in the area of mobile devices the collaboration **UZH-JUB-UniBwM-M2** collects mobile network statistic data to determine the Quality of Experience (QoE).

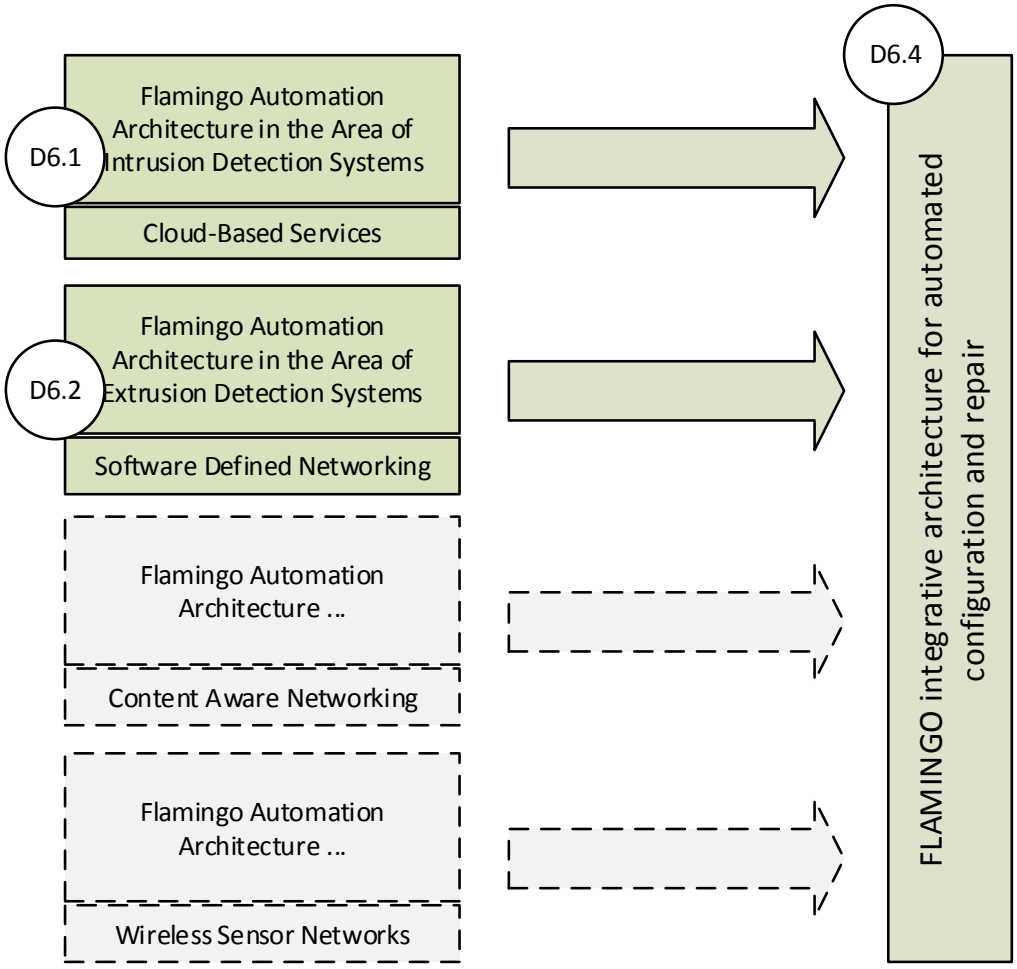

Figure 1: WP6 architecture progress

The application collecting this data is among one of the core developments in Y2. A monitoring architecture to collect data in IoT environments is developed by **INRIA-JUB-Distr**. The focus of this collaboration is to monitor events and network flows passively on resource constrained nodes. Additional monitoring data is obtained through a monitoring architecture developed by **UT-UniBwM-IDS** and **JUB-UT-Pattern**. In Y2 both collaborations extended their monitoring capabilities to collect packet-level traces respective traffic flow data from various distributed locations.

A first approach is a Real-Time IDS which is capable of real-time intrusion detection using a set of easily measurable metrics, for example multiple source-IP-addresses accessing a single destination-IP-address typically indicating DDoS attacks. In comparison, with traditional, ”heavy-weight” sensors, this Real-Time IDS is a comparatively ”light-weight” sensor, since we are not dealing with a dedicated device, but rather elude resources on the router (hampering him to fulfill his ”core business”). The next step includes the involvement of more ”heavy-weight” sensors. A first example of such an IDS is based on SSHCure, a flow-based IDS specific for SSH attacks, which implements an algorithm for near real-time detection of ongoing attacks and allows the identification of compromised attack targets. In case the flow-based IDS reports an alert, an additional IDS is used to verify/falsify the results of the flow-based IDS, for example, the well-known IDS

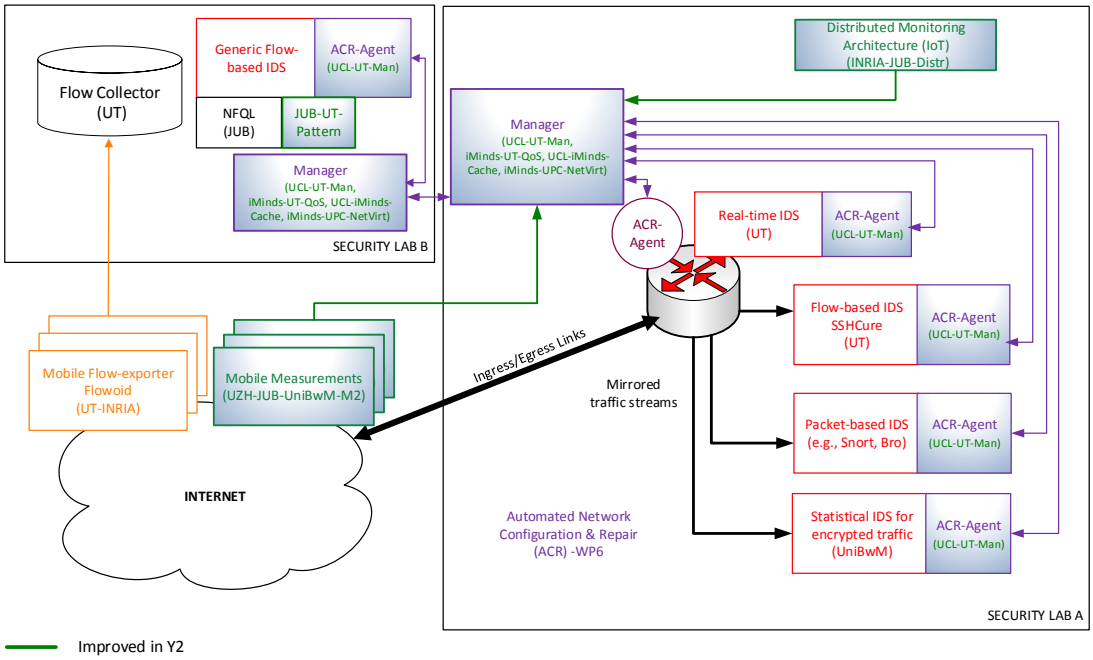

Figure 2: FLAMINGO Automation Architecture in the Area of Intrusion Detection Systems

snort[6] in addition to Bro[7]. These IDSs have to be managed to allow an adaptation of a single IDS based on detections from other IDSs, supported by a couple of knowledge description approaches, learning techniques and algorithms. This functionality is consolidated in a centralized manager component per site, which evaluates the output of the IDSs to define an overall goal for the intrusion detection process at a glance. The definition of this goal results in piecewise configuration actions for the IDSs. The basis for this architecture approach builds the work done in collaboration **UT-UniBwM-IDS**. In the current state the collaboration is collecting first datasets to be used for testing the prototype developed in the second year of FLAMINGO. Details about this approach and why it is used can be found in Deliverable D6.1.

By establishing a “rule-of-thumb” to estimate whether traffic is Gaussian distributed or not, based on conclusions from data analysis such as hosts behavior and applications usage, the collaboration **JUB-UT-Pattern** supports our manager. In Y2 the collaboration worked on the influence of traffic bursts that compromise traffic Gaussianity fit. The Network Flow Query Language (NFQL), developed by JUB, acts as a possible part of the architecture by providing functions that allow pattern querying in a flow message stream.

The management framework DACoRM described in D6.1 supports our architecture by embedding a level of intelligence to edge nodes. These edge nodes are realized by our ACR-Agents to improve self-management functionality. Therefore, ACR-Agents are organized into a management substrate through which they synchronize reconfiguration actions. DACoRM is developed among others through the collaboration **UCL-UT-Man** with it’s Y2 focus on evaluation using real topologies and traffic traces, where they focused on the influence of topology characteristics and routing protocols. Since configuration actions are covered by DACoRM the configuration itself needs to be adjusted and edited in an automated and later in an autonomous manner. The collaboration **iMinds-UT-QoS** developed a distributed algorithm that is able to divide resources among various

[6] http://www.snort.org/
[7] http://www.bro.org/

client instances, e.g. several kinds of intrusion detection systems with limited bandwidth. Since one of the missing aspects of the FLAMINGO architecture in the comparison (see Table 6) is the action (reactive/proactive), the collaboration **UCL-iMinds-Cache** considers this issue by the development of a proactive mechanism to efficiently manage the utilization of network resources. The network virtualization approach, emerged out of the collaboration **iMinds-UPC-NetVirt**, supports our architecture and applies machine learning based techniques. Finally, the collaboration **INRIA-UniBwM-Cloud** compares among other activities SDN-based mechanisms for delivering security to well established approaches in traditional networks. As a consequence the knowledge base of both architectures can be combined in order to distinguish between mitigation strategies against attacks.

During the lifetime of the project, the active PhD collaborations described in Section 4.1 and the planned collaborations shown in Figure 25 will contribute to the core architecture. Nevertheless, an overall goal of the collaborations is to establish well defined interfaces and standardized protocols to exchange configuration information and detection events. Within 2015 we will further focus on the specification whether and how existing research approaches, particularly with respect to knowledge and information exchange management, can be used for our first part of the FLAMINGO integrative architecture for automated configuration and repair, respectively needed to be extended.

3.4 FLAMINGO Automation Architecture for Extrusion Detection Systems

As described in Section 3.2 two architectures have been addressed within Y2. The first one was presented in Y1 and continuously improved in year 2. The second one focuses on the application domain SDN, introduced through the update of the description of work. This section aims to provide an overview of the FLAMINGO Automation Architecture in the Area of Extrusion Detection and how the established and ongoing collaborations are involved.

Since monitoring data is the basis to build a data base, from which a knowledge base as well as information models can be derived. These can be used as an input for correlation methods, learning techniques and knowledge description, what in turn builds the basis for the corresponding automated actions. Since our architecture developed in Y1 uses an already established monitoring architecture, that is described in D5.1, this also builds the basis for our second architecture. As a counterpart to our already developed architecture the second one (see Figure 3), that is described here, focuses on extrusion detection.

Considering that traffic needs to be investigated at the same scale, similar systems can be found in our second architecture as well, but with different configuration options set. Therefore, the collaborations involved in monitoring activities are the same like in our first architecture, but with special focus on SDN capabilities. **INRIA-UniBwM-Cloud** investigates SDN-based mechanisms for delivering security in networks, they provide insights about the different NetFlow data collection process in SDN. In addition, **JUB-UT-Pattern**, while working on packet-level traffic traces, observes the differences of collecting these traces in traditional networks and SDN enabled networks. The overall architecture developed by **UT-UniBwM-IDS** for multi-layered intrusion detection is in a first step of this architecture applied, but with different configurations due to the fact of detecting extrusions instead of intrusions. The evaluation, if this approach is the most suitable, also for extrusion detection, is planned as a further step. Reasoned by the distribution of detection systems an agent that configures and collects monitoring data (ACR-Agent) needs to be deployed to all involved components similar to the first architecture. The configuration and synchronization of these agents is done by distributed algorithms (**iMinds-UT-QoS**) and DACoRM, the management framework described in D6.1 and maintained by **UCL-UT-Man**. Furthermore, the SDN technology builds a basis for our architecture, that allows a much faster and data-driven reconfiguration of our architecture

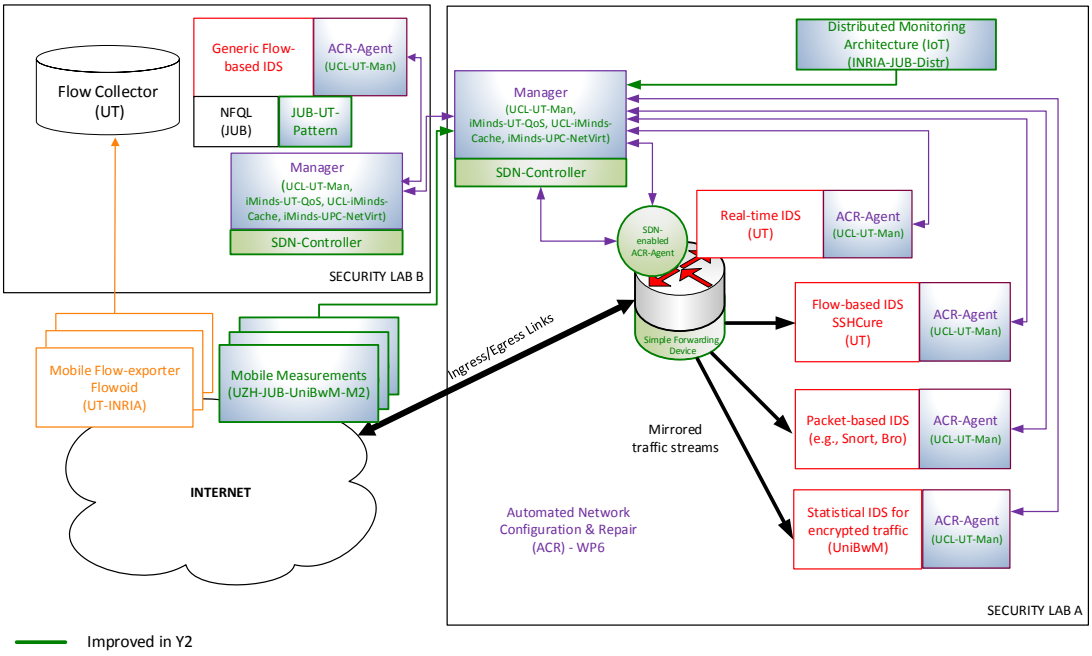

Figure 3: FLAMINGO Automation Architecture in the Area of Extrusion Detection Systems

in respect of detection systems, investigating the traffic, and possible mitigation strategies. The collaboration **iMinds-UPC-NetVirt** investigates management principals in SDN enabled networks. Their focus lies on resource management, especially in the configuration of the SDN control plane to efficiently manage resources in virtualized networks by dynamically adjusting the virtual network to substrate network mappings based on network status. Therefore, they extend an SDN controller to monitor the resource utilization, as well as the average load of links and switches to pro-actively add or remove flow rules from the switches [6].

In addition, the central manager approach from our first architecture is extended in a sense, that an SDN controller is added that takes care of reconfiguration actions (e.g. with OpenFlow) on simple forwarding devices to steer the traffic quickly, taking into account recent monitoring events and possible detected attacks. Due to the fact that SDN states a relatively new technology not all existing architectures could be included within this second architecture. During the lifetime of the project, the active PhD collaborations described in Section 4.1 and the planned collaborations shown in Figure 25 will contribute to the core architecture.

Within 2015 we will further focus on the specification how existing research approaches, can be extended to use the advantages of SDN to form our first part of the FLAMINGO integrative architecture for automated configuration and repair, as a protocol and technology independent architecture, using standardized interfaces and methodologies.

3.5 Enablers

Enablers in common are technologies, algorithms and also tools that support a defined part of a problem solution. Basic mathematical functions or basic data (ground truth data) are not considered as enablers. This can be seen as an input for an enabler or a process which is supported by an enabler. Also, tools and algorithms which build these aforementioned data can not be seen as an enabler.

In FLAMINGO WP6 relation to enablers we focus mainly on enablers which can support automation of configuration and repair. Enablers in this category must not directly be involved in automated configuration and repair, but they should be included in such a process or a pre-(processing) step of an automated configuration and repair process. For evaluation purposes we include enablers in a category named experimentation. This category includes enablers necessary to evaluate new developed approaches in our application domains. Therefore, also enablers like network testbeds and simulation environments are included. Also, pre-programmed rules or a simple logic can be seen as an enabler, but these approaches are not very suitable for dynamic and changeable environments which are in the focus of FLAMINGO. Thus, these could only be precursors of enablers considered in FLAMINGO.

3.5.1 Taxonomy

Based on questionnaires and discussions within the FLAMINGO consortium a taxonomy for the Future Internet management enablers was set up. The goal of developing this taxonomy is to define a structure where the enablers of research done in WP5 and WP6 can be mapped to. This helps researchers to identify which enablers are currently used and which experiences exist with the research work packages.

In a first step we collected enablers in the research area of network and service management within FLAMINGO. Later, during various iterative discussions we defined preferred terms under which we summarize parts of our list of enablers. This step iterated several times to come up with a taxonomy of FLAMINGO key enablers (see Figure 4). In the following sections all these enablers are described and linked to the respective collaborations which are working with these enablers. Therefore, we discuss the enablers in the order in which they occur in the taxonomy.

3.5.2 Knowledge Description

Knowledge Description builds the basis for representing and organizing knowledge that is used as a basis for decisions and reasoning in algorithms. It includes the representation of knowledge in a way that all necessary information as well as their relations are stored consistent. In addition, it supports the algorithms and expert systems in getting necessary data in a standardized way. A knowledge base can include a variety of kinds of knowledge like domain knowledge, control knowledge, explanatory knowledge and system knowledge [7]. The following paragraphs give an overview about knowledge description techniques used within FLAMINGO.

Knowledge Representation Knowledge Representation describes a representation of expertise, wisdom or rules-of-thumb, often described by rules containing ”if-then-else” conditional statements or cases containing various fact patterns. Knowledge bases focus on domains, also called narrow issues, within a particular fact situation. They may also consist of representative objects within a sub-class(rules against hearsay) and class(rules of evidence) of information.

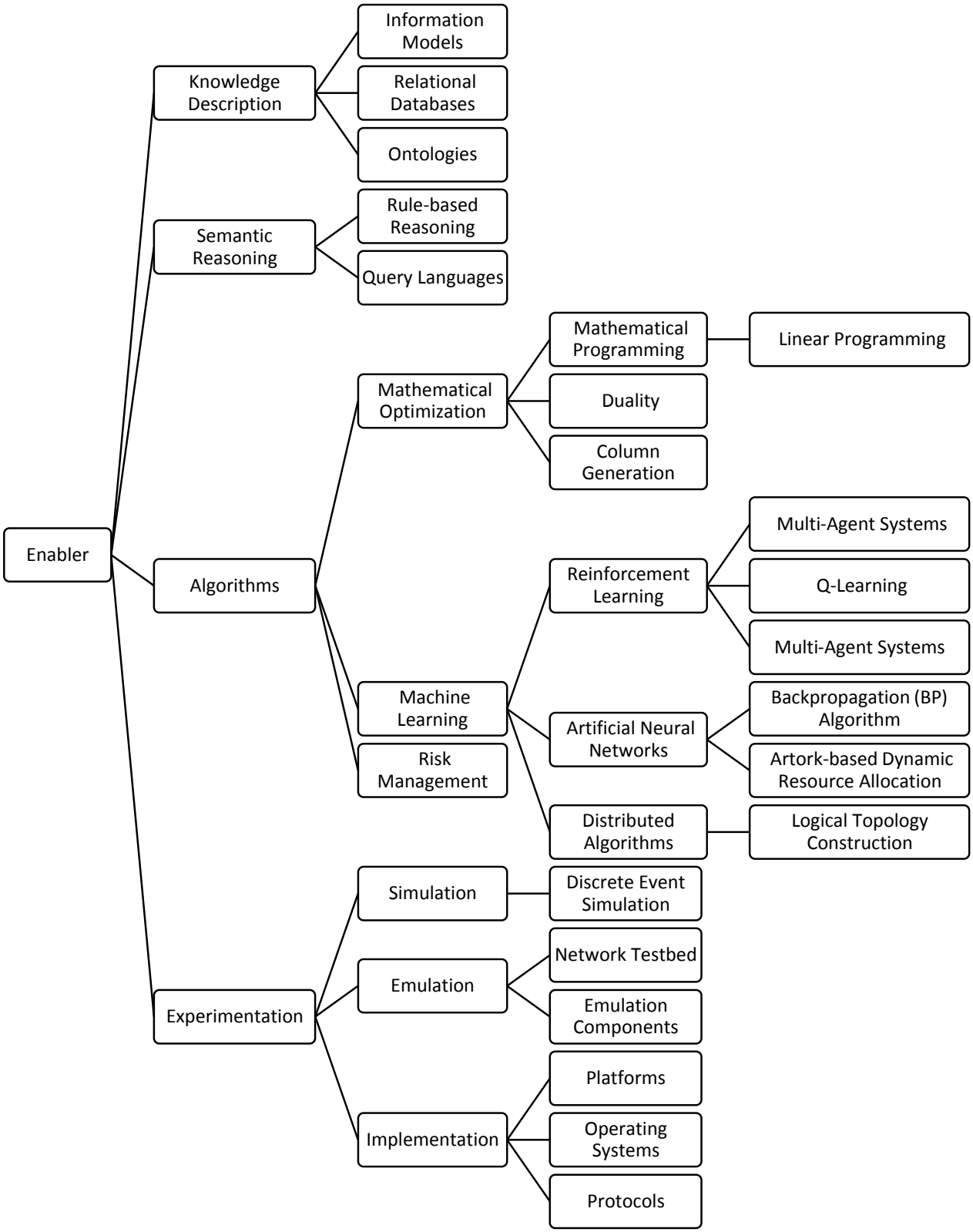

Figure 4: Taxonomy of FLAMINGO key enablers

Information Models A representation of concepts and the relationships, constraints, rules, and operations to specify data semantics for a chosen domain of discourse is called information model. In general it specifies relations between kinds of things, but may also include relations with individual things. It can provide sharable, stable, and organized structure of information requirements or knowledge for the domain context [8]. Information models are used in [9] where a context exchange process is proposed that automates the context communication between nodes. This scalable approach is able to quickly react to local context updates, while maintaining a high level of expressivity to define relationships between federation partners.

Relational Databases A database is called a relations database if it stores information about both the data and how it is related [10]. This information is represented in two dimensional tables to preserve relational structuring. A physical and logical implementation of a relational database including hardware and software is called Relational Database Management System (RDBMS). This is used to control access, reading, writing, modifying, and processing the information stored in the databases. The data is formally described and organized according to each database's relational model (database schema), according to the design. A relational database is a set of tables containing data fitted into predefined categories. Each table contains one or more data categories in columns. Each row contains a unique instance of data for the categories defined by the columns. The structured query language (SQL) is one program interface to a relational database.

Ontologies According to [11] the raw term ontology already exists for a long time in philosophy where it refers to the subject of existence. In the context of knowledge management, it is referred as the shared understanding of domains, which can be seen as a set of entities, relations, functions, axioms and instances.

OWL ontologies are used to represent network monitoring information. OWL reasoning and SWRL rules are applied by autonomic elements to dynamically generate information filtering rules based on the information needed by their management algorithms. Subsequently, OWL reasoning, SWRL, and SPARQL are used to automatically determine if a specific piece of semantic monitoring information satisfies one or more filter rules and information is forwarded only to those autonomic elements that need it.

OVAL has been introduced by the NIST agency, and stands for "Open Vulnerability and Assessment Languages". Within the FLAMINGO Automation Architecture in the Area of Extrusion Detection Systems 3.4 it will be used for describing vulnerable states and perform related configuration assessments in a standardized way.

Semantic Reasoning

Semantic reasoning is used to infer logical consequences from a set of axioms or sets of asserted facts. The concept of a semantic reasoner, which is the algorithm or piece of software where semantic reasoning is applied, uses an inference engine to generalize, by providing a set of mechanisms to work with. Means of an ontology language are used to specify the inference rules. A commonly used concept used by reasoners to perform reasoning is first-order predicate logic.

Rule-based Reasoning Rule-based reasoning uses ”if-then-else” rule statements. Rules are simply patterns in this case. An inference engine searches for patterns in the rules that match respective patterns in the data.
If rules start with data or facts and look for rules which apply to the facts until a specified goal is reached, they process data-driven reasoning or forward-chaining. Backward-chaining or goal-driven reasoning is also possible when they start with a goal and look for rules which apply to this goal until a conclusion is reached.

Query Languages A simple classification of query languages is possible according to whether they are database query languages or information retrieval query languages. The main difference is that database query languages give factual answers while in contrast an information retrieval query language attempts to find documents containing information that is relevant to the inquiry. **Information Retrieval Query Language** is used to retrieve an information from a database by an interpretation of the most suitable result according to the query. In contrast to a **Database Query Language** the semantic of the query is not defined by a precise rendering of a formal syntax.

3.5.3 Algorithms

An algorithm describes in a specific language what steps (e.g. formulas, procedures) need to be done by an executing entity and in which order to solve a defined problem. A computer program can be seen as an elaborate algorithm. In mathematics and computer science, an algorithm usually means a small procedure that solves a recurrent problem. The following paragraphs give an overview of the different kinds of algorithms, which were used and developed within FLAMINGO.

Mathematical Optimization Simplex method is used to solve the ILP, modeling the problem of cache capacity allocation, content placement and routing. Also for modeling the problem of QoE optimization subject to bandwidth limitations using packet-based sampling. The Levenberg-Marquardt non-linear optimization algorithm is used to fit theoretical cumulative request distribution curves to the real request distribution of a video. Based on this fit, the future requests of the video are predicted and used by a predictive cache replacement strategy. We apply optimization theory to formulate a set of mathematical programs that are aimed at achieving a one-shot virtual network embedding. We then use column generation to propose a heuristic that ensures a near optimal solution with significant improvement in computation complexity.

Mathematical Programming Mathematical programming, and especially linear programming, is one of the best developed and most used branches of operational research [12]. It involves the use of mathematical models, particularly optimizing models [13], to assist in taking decisions. The objective is usually to achieve optimum allocation of limited resources among competing activities, under a set of constraints imposed by the nature of the problem being studied. These constraints could reflect financial, technological, marketing, organizational, or many other considerations. In broad terms, mathematical programming can be defined as a mathematical representation aimed at programming or planning the best possible allocation of scarce resources. A mathematical program is made up of four main components [14]:

- **Variables** (also known as decision variables): These represent values that can be adjusted or controlled, for example the bandwidth of a substrate link, the CPU demand of a virtual node, etc. Usually, the aim of a mathematical program is to find the values of the variables that provide the best match of the objective function.

- **Objective function**: This is a mathematical expression that combines the variables to express the goal of the mathematical program. It may represent an average resource allocation or InP profit. It is usually required to maximize or minimize this function.
- **Constraints**: These are mathematical expressions that combine the variables to express limits on the possible solutions. For example, they may express the fact that the maximum number of virtual links that can be mapped on a substrate link must not have total bandwidth demand that exceeds the total free bandwidth capacity of the substrate link.
- **Variable bounds**: Only rarely are variables in a mathematical problem permitted to take on any value from negative infinity to positive infinity. Instead, the variables usually have bounds, for example, 0 and 1 may bound a variable that indicates whether or not a given virtual node is mapped onto a substrate node.

When the mathematical representation uses linear functions exclusively (i.e. when all the mathematical expressions for the objective function and the constraints are linear), we have a linear-programming model [12]. In the next section, we formulate a linear program and use it to explain the concepts of duality and hence column generation.

Linear Programs A linear program is the most common formulation of an optimization/mathematical problem. It involves a *minimization* or *maximization* of an objective function over some domain. The objective function is linear, and the domain, or feasible set, is defined by linear constraints [15]. Equations (1) - (4) show a generic example of a linear program [16].

$$\textbf{minimize} \quad \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j t_j \tag{1}$$

$$\textbf{subject to} \quad e_j t_j + \sum_{i=1}^{m} a_{ij} x_i \geq g_j \quad , \quad 1 \leq j \leq n \tag{2}$$

$$f_i x_i + \sum_{j=1}^{n} b_{ij} t_j \geq h_i \quad , \quad 1 \leq i \leq m \tag{3}$$

$$x_i \geq 0,\ t_j \geq 0 \quad , \quad 1 \leq i \leq m,\ 1 \leq j \leq n \tag{4}$$

As can be observed in the formulation (1) - (4), the variables are x and t, (1) is the objective function, (2) and (3) are constraints and (4) are variable bounds. Therefore, the linear program above has $m + n$ constraints, and all its variables are positive.

Application of Linear Programs for Quality Optimization

Network providers are exploring how they can offer VoD HTTP Adaptive Streaming (HAS) services next to traditional TV services over their managed network environment. HAS services offer the same content at multiple qualities, each at their corresponding rate. This allows providers to perform QoE management by adjusting each sessions quality level, based on the current network utilization. At peak times, the consequences of an inadequate amount of resources in the network, can thus be anticipated by reducing the quality of individual streaming sessions, while still allowing admittance of all users.

Figure 5 gives an overview of the problem variables and assumptions. Let us consider an access network topology modeled as a graph, consisting of a set of nodes $\mathcal{N}$, which encompasses servers $\mathcal{S} \subset \mathcal{N}$, proxies $\mathcal{P} \subset \mathcal{N}$, and clients $\mathcal{C} \subset \mathcal{N}$. A set of edges $\mathcal{E}$ connects the nodes in a logical tree topology which

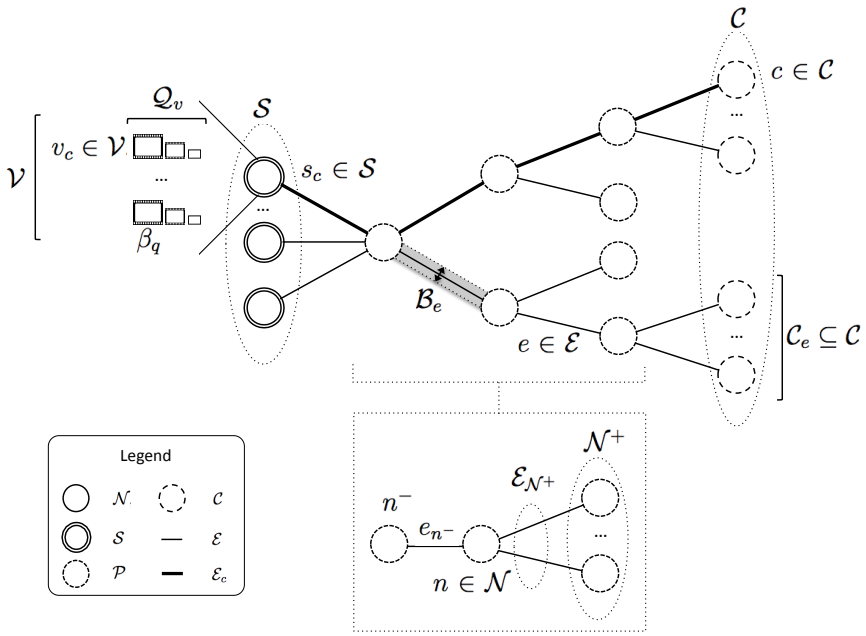

Figure 5: Graphical representation of variables and assumptions.

is typically used for video delivery networks[8]. Note that typical access networks are using a logical tree for their delivery, although the underlying physical network is not a tree due to replication concerns. Every node $n \in \mathcal{N}$ has an incoming edge $e_{n^-} \in \mathcal{E}$ connecting it to its predecessor $n^- \in \mathcal{N}$ and a set of outgoing edges $\mathcal{E}_{\mathcal{N}^+} \subseteq \mathcal{E}$ connecting it to its successors $\mathcal{N}^+ \subset \mathcal{N}$. Every edge $e \in \mathcal{E}$ has an associated bandwidth capacity $\mathcal{B}_e$ reserved for HAS traffic.
The servers host a set of videos $\mathcal{V}$. Every video $v \in \mathcal{V}$ has an associated set of quality representations $\mathcal{Q}_v \subseteq \mathcal{Q}$. Moreover, every quality representation $q \in \mathcal{Q}$ has a bit rate β_q. Every client $c \in \mathcal{C}$ has an associated origin server $s_c \in \mathcal{S}$, a unique delivery path $\mathcal{E}_c \subseteq \mathcal{E}$ from that server, and a video $v_c \in \mathcal{V}$. The set of clients that have an edge $e \in \mathcal{E}$ as part of their delivery path $\mathcal{E}_c$, is represented by $\mathcal{C}_e \subseteq \mathcal{C}$.
The problem consists of maximizing the QoE over all clients $c \in \mathcal{C}$, while adhering to the edge bandwidth constraints. The solution is characterized by a boolean decision matrix A. The element $a_{c,q} \in A$ is equal to 1 if quality $q \in \mathcal{Q}_{v_c}$ is selected for client $c \in \mathcal{C}$, and 0 otherwise. The decision variables are subject to the following two constraints:

$$\forall c \in \mathcal{C}, \forall q \in \mathcal{Q}_{v_c} : a_{c,q} \in [0,1] \tag{5}$$

$$\forall c \in \mathcal{C} : \sum_{q \in \mathcal{Q}_{v_c}} a_{c,q} = 1 \tag{6}$$

The above constraints state that the decision variables are boolean values and that only one quality representation can be selected per client.
According to *Padhye et al.*, the maximum achievable throughput B for a TCP connection subject to a round trip time RTT and maximum window size W_{max} can be approximated.

[8] An example is the Triple Play Service Delivery Architecture from Alcatel-Lucent (http://goo.gl/4aZVvf), which is used by over 50 operators worldwide (http://goo.gl/kHMY1b)

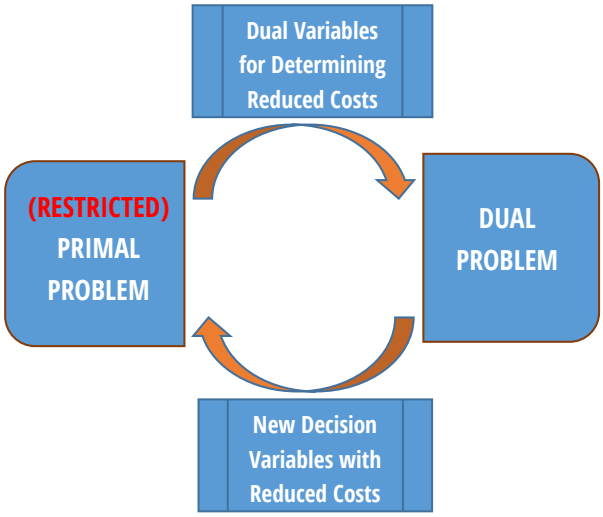

Figure 6: Interaction between Primal and Dual Problems in Column Generation

Duality Any given linear programming problem is referred to as a primal problem, and every primal problem has an associated linear program called the *dual problem* [15]. The dual problem provides an upper bound to the optimal value of the primal problem [17, 18]. The fundamental idea behind duality is that every feasible solution for the primal problem gives a bound on the optimal value of the objective function of the corresponding dual problem [19]. The duality theorem states that the objective function value of the dual at any feasible solution is always greater than or equal to the objective function value of the primal at any feasible solution [12].

Column Generation Many linear programs are too large to consider all their variables explicitly. Since most of the variables will be non-basic and assume a value of zero in the optimal solution, only a subset of variables need to be considered in theory when solving the problem [16]. Column generation takes advantage of this idea to generate only the variables which have the potential to improve the objective function, i.e. to find variables with negative reduced cost (assuming without loss of generality that the problem is a minimization problem). In order to use a column generation approach, the problem being solved is split into two subproblems: the primal problem and the dual problem. In Fig. 6, we represent the interaction between the primal and dual problems in column generation.

The main idea is to solve a restricted version of the program (the restricted primal problem) - which contains only a subset of the variables, and then (through the use of the dual problem) add more variables as needed [20]. Therefore, we start by solving a restricted primal problem, and from its solution, we are able to obtain dual prices for each of the constraints in the primal problem. This information is then utilized in the objective function of the dual problem. Then the dual problem is solved. If the objective value of the dual problem is negative, a variable with negative reduced cost has been identified. This variable is then added to the primal problem, and the primal problem is solved again. Re-solving the primal problem generates a new set of dual values, and the process is repeated until no negative reduced cost variables are identified. When the dual problem returns a solution with non-negative reduced cost, we can conclude that the solution to the primal problem is optimal [21]. In order to have the initial restricted set of variables, it is required to have an initial *feasible* solution to the primal problem.

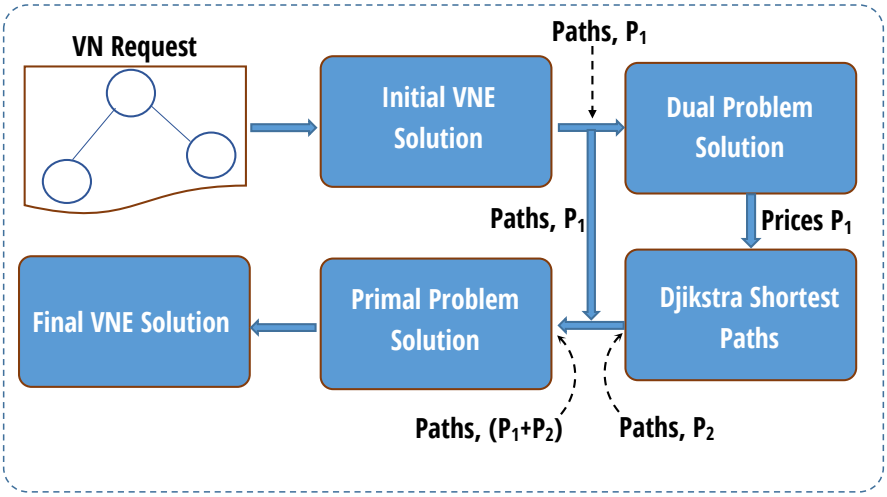

Figure 7: Column Generation-based Virtual Network Embedding

Column Generation-based Virtual Network Embedding

The proposed column generation approach is shown in Fig 7. We start by creating an initial set of paths (P_1) using a two stage node and link mapping. We then use these paths to solve a dual problem, and use the pricing problems (shortest path problems) to determine a set of paths (P_2) to add to the initial solution i.e. paths that can improve the initial solution. These paths are then used to solve a restricted primal problem to obtain a final solution. It can be noted that our proposal avoids the usual iteration required in a path generation approach where the primal and dual problems are solved sequentially, many times, instead preferring only to perform a single iteration.

Machine Learning Machine learning can be seen as a type of artificial intelligence that enables computers to learn without being explicitly programmed. An early definition describes machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed" [22]. Machine learning focuses on the development of computer programs that can teach themselves to grow and change when exposed to new data. Essentially, it is a method of teaching computers to make and improve predictions or behaviors based on various data. How this data is defined depends strongly on the problem to solve. In general machine learning is used as a generic term to define a variety of learning algorithms that produce a quasi learning from examples. The actual accuracy is determined by the quality of training and test data, that is provided to the learning algorithm. The so called convergence rate measures this data quality. The reason why example data needs to be provided is because the selected learning algorithm should be able to informatively by guidance make generalization. The algorithms can be classed into two main areas supervised learning (classification) and unsupervised learning (clustering) techniques. In addition, it is important to make an informed decision on how the separation of training and test data is done. Furthermore, the quality of the data provided to the learning algorithms needs to be described.

In the following paragraphs the learning algorithms used in FLAMINGO are described in detail.

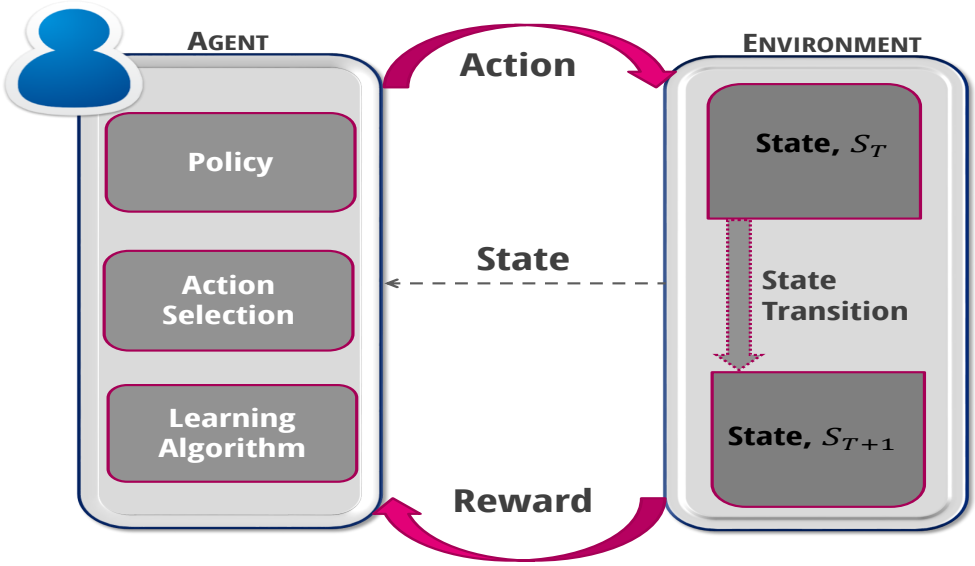

Figure 8: Agent-Environment interactions in Reinforcement Learning

Reinforcement Learning

Reinforcement Learning (RL) is a technique from artificial intelligence [23] in which an *agent* placed in an *environment* performs actions from which it gets numerical rewards. Fig. 8 shows an interaction between an agent and an environment in a typical RL scenario. For each learning episode [24], the agent perceives the current *state* of the environment and takes an *action*. The action leads to a change in the state of the environment (state transition), and the desirability of this change is communicated to the agent through a scalar *reward*, which is an evaluation of the desirability of the agents' action. In the next subsections, the three major components of a RL agent as shown in Fig. 8 are defined.

Dynamic Programming There are many learning algorithms in Reinforcement Learning (RL), all falling within three broad learning methods, which are: dynamic programming, Monte Carlo, and time difference methods [24]. Dynamic programming [25] uses the concepts of a dynamic system's state and of a value function to define a functional equation, which represents a controller for the system. It requires the knowledge of the probability of transiting from one state to another, and assumes that the measurements available on the system's state are detailed enough so that the controller can avoid reasoning about how to collect information about the state [26]. Problems with these characteristics are best described in the framework of Markovian Decision Processes (MDPs) [27]. Therefore, the dynamic programming approach is to transform the problem of finding a good controller into the problem of finding a good value function. However, Dynamic programming suffers from the curse of dimensionality, meaning that its computational requirements grow exponentially with the number of state variables [24, 28]. On the other hand, Monte Carlo [29] and time difference (TD) [30] methods are primarily concerned with how an agent ought to take actions in an environment to minimize the notion of long-term cost, that is, to obtain the optimal policy, when the state transition probabilities are not known in advance. Specifically, if state transition probability is not known, but a sample transition model of states, actions and costs can be built, Monte Carlo methods can be applied to solve the problem, while, if the only way to collect information

about the environment is to interact with it, TD methods are used. TD methods combine elements of dynamic programming and Monte Carlo ideas, they learn directly from experience which is a characteristic of Monte Carlo methods and they gradually update prior estimate values, which is common of dynamic programming [31].

Since for the problem of dynamic resource allocation in network virtualisation we do not have an estimate of the state transition probabilities, and because information about states, actions and rewards can only be obtained *over time* through interactions with both substrate and virtual networks, the natural reinforcement method of choice should come from TD methods. Since the most common TD learning algorithm is Q-learning [24, 32], the remainder of this Chapter will design and evaluate a Q-learning based algorithm for the opportunistic allocation of resources in network virtualisation.

Q-Learning This is a time difference [24] learning algorithm that gradually builds information about the best actions to take in each possible state. This is achieved by finding a *policy* that maximizes some long-term measurement of reinforcement. Therefore, the main objective of a Q-learning agent is to maximize the overall reward it achieves throughout the learning period. In general, the objective of a learning algorithm is to learn an optimal *policy*. In algorithm 1, the generic psuedocode of a Q-learning agent is given.

Policy A policy defines the learning agent's way of behaving at a given time. It is a mapping from each environment state to actions to be taken when in that state [24]. In Q-learning, a policy is composed of a *Q-value*, $Q(s, a)$ for each of the possible state-action combinations. $Q(s, a)$ is a measure of the desirability of each action, a while in the state, s. The learning process therefore involves continuously updating these values until they guide the agent to taking the best action while in any of the possible states [24].

Policy Update While using the Q-learning algorithm, after every learning episode, an agent updates its Q-values using the Q-learning rule in (7).

$$Q(s_p, a_p) \leftarrow (1-\alpha)Q(s_p, a_p) + \alpha\left\{r_p + \lambda \max_{a\in\mathcal{A}} Q(s_n, a)\right\} \tag{7}$$

where $Q(s_p, a_p)$ is the new value of state s_p corresponding to action a_p, r_p is the reward obtained from taking the action a_p while in state s_p and s_n is the next state resulting from taking the action a_p while in state s_p, implying that $Q(s_n, a)$ is the value associated with the action a of the state s_n. The parameters $0 \leq \alpha \leq 1$ and $0 \leq \lambda \leq 1$ are referred to as learning rate and discount factor respectively.

Learning rate: The learning rate (α) determines the extent to which newly acquired information overrides old information. If $\alpha = 0$ the agent does not learn anything, while a value of 1 for α would make the agent consider only the most recent information.

Discount factor: The discount factor (λ) models the importance that is attached to future rewards in comparison with immediate reward. Therefore, $\lambda = 0$ would make the agent short-sighted so that it would only give importance to current rewards, while a value close to 1 would make it strive for high rewards in the long-term, even if this means getting negative rewards in the short run.

Action Selection A reinforcement learning agent usually has one of two possible strategies with regard to how actions are selected; (1) *exploit* the knowledge that it has found for the current state s by taking the action a that maximizes

$Q(s, a)$ i.e. take the action that it already knows is the best in that state, or (2) *explore* by selecting a different action from the one that it currently thinks is best, with the objective of trying to learn if there is an action better than what it currently thinks is best.
Balancing the ratio of exploration and/or exploitation is a great challenge in RL since it influences the convergence (learning) time and the quality of learned policies. On one hand, too much exploration prevents the agent from maximizing the short-term reward because selected exploration actions may yield negative reward from the environment. But on the other hand, exploiting uncertain environment knowledge prevents agents from maximizing the long-term reward since selected actions may not be optimal [33]. Whether an agent explores or exploits its knowledge can generally be determined by its *action selection* criterion. The two most common action selection criteria are ϵ-greedy and softmax [24].

ϵ-greedy Action Selection In ϵ-greedy, a greedy action (i.e. action with the highest Q-value) is selected most of the time, and $-$ using a small probability, $\epsilon -$ a random action is chosen once in a while. This ensures that after many learning episodes, all the possible actions will be tried a high number of times, leading to an optimal policy. Although ϵ-greedy action selection is an effective and popular means of balancing exploration and exploitation in reinforcement learning, one drawback is that when it explores, it chooses equally among all actions. This means that it is as likely to choose the worst-appearing action as it is to choose the next-to-best action. In tasks where the worst actions are very bad, this may be unsatisfactory. The obvious solution is to vary the action probabilities as a graded function of their estimated values [24].

Softmax Action Selection Softmax differs from ϵ-greedy in the way the random action is selected. A weight is assigned to each of the actions depending on their estimated values. A random action is selected based on the weight associated with it, ensuring that worst actions are unlikely to be chosen. When using softmax, an agent takes a random action a while in state s with a probability $\mathcal{P}(a|s)$ which is commonly defined by a *Gibbs* or *Boltzmann* distribution [34] as shown in equation (8).

$$\mathcal{P}(a|s) = \frac{\exp\{Q(s,a)/\tau\}}{\sum_{\hat{a} \neq a} \exp\{Q(s,\hat{a})/\tau\}} \tag{8}$$

where τ is a positive parameter called the temperature. High temperatures ($\tau \to \infty$) cause the actions to be almost equiprobable, while lower temperatures ($\tau \to 0^+$) cause the probability to be dependent on the Q-values.
Both methods (ϵ-greedy and softmax) have only one parameter that must be set [24]. Whether softmax action selection or ϵ-greedy action selection is better is unclear and usually depends on the task being considered.

VDBE-Softmax Tokic et. al. propose the Value-Difference Based Exploration policy with Softmax action selection (VDBE-Softmax) [35, 36]. The VDBE-Softmax policy only performs exploration in situations when knowledge about the environment is uncertain, indicated by fluctuating Q-values during learning. The Softmax-method is extended by introducing a state-dependent exploration probability $\epsilon(s)$, being updated through the learning process based on the difference in learned Q-values before and after a learning step. Large differences indicate a high degree of uncertainty about the environment, shift-

ing the policy to exploration. When differences decrease and the agent gains certainty, exploitation becomes more probable. In case of exploration, the Softmax policy is used to avoid bad performance when many actions yield relatively high negative reward. In this way, the VDBE-Softmax policy takes into account the tradeoff between the ϵ-greedy and Softmax exploration policies.

The state-dependent exploration probabilities are initialized to $\epsilon(s) = 1$ for every state and updated after every learning step using the update rule shown in equation (9).

$$\epsilon(s) = \delta \times f(s_t, a_t, \sigma) + (1 - \delta) \times \epsilon(s) \tag{9}$$

In this equation, σ is a positive constant called inverse sensitivity and $\delta \in [0, 1)$ a parameter determining the influence of the selected action on the state-dependent exploration probability. The parameter σ influences $\epsilon(s)$ in a way that low values cause full exploration at small value changes while high values of σ require high value changes to perform a high level of exploration. The term $f(s_t, a_t, \sigma)$ is based on the difference Δ between the learned Q-values before and after a learning step and is calculated as shown in equation (10).

$$f(s_t, a_t, \sigma) = \frac{1 - e^{\frac{-|\alpha\Delta|}{\sigma}}}{1 + e^{\frac{-|\alpha\Delta|}{\sigma}}} \tag{10}$$

Q-Learning-based HTTP Adaptive Streaming Client In recent years, Over-the-Top (OTT) video delivery, where content is transported over the best-effort Internet, has gained a lot of popularity. For OTT video, HTTP Adaptive Streaming (HAS) is becoming the de facto standard. The general HAS concept is illustrated in Fig. 9. A HAS video file consists of multiple segments with a typical length of 2 to 10 seconds, encoded at multiple quality levels and resolutions. At the client side, the information about the video segments and quality levels is provided in the form of a manifest file. A standard HAS client requests a video segment on arrival of the previous segment. Based on the perceived network state and the information in the manifest file, a quality selection heuristic dynamically adapts the requested quality level. Each segment is downloaded in a progressive manner, while at the client side a buffer is used to bridge temporary anomalies such as a late arrival of a video segment during a small time window. Finally, the video segments, stored in the buffer, are played back as a single continuous video stream.

Transporting the video segments over HTTP provides both seamless interaction through firewalls and reliable delivery. On the other hand, the best-effort nature of the Internet makes these HTTP-based techniques prone to bandwidth fluctuations and network congestion. Given the detrimental impact on the Quality of Experience (QoE), the intelligence of the quality selection heuristic is crucial for the quality of HAS techniques.

Several large industrial players, including Microsoft[9], Apple[10] and Adobe[11] have commercial HAS implementations. MPEG, in collaboration with other standard groups, such as 3GPP, standardized the HAS interfaces and protocol data in Dynamic Adaptive Streaming over HTTP (DASH) in 2011 [37]. In this way, a common ground was established between the vast amount of available implementations. The

[9] http://www.iis.net/downloads/microsoft/smooth-streaming

[10] http://tools.ietf.org/html/draft-pantos-http-live-streaming-10

[11] http://www.adobe.com/products/hds-dynamic-streaming.html

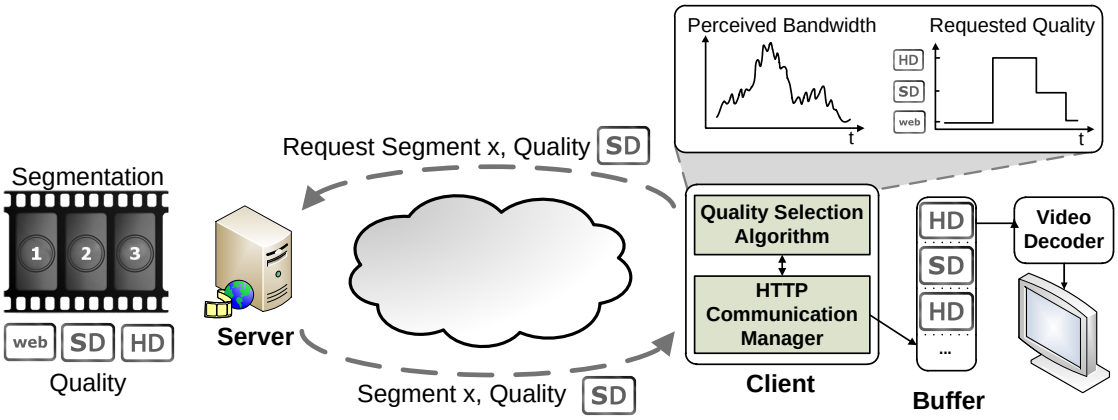

Figure 9: Schematic overview of the HTTP Adaptive Streaming concept.

bitrate adaptation heuristics are, however, not standardized, and thus implementation specific.

While current quality adaptation algorithms are hard-wired to fit specific network configurations [38], Claeys et a.l [39, 40] propose a Q-Learning-based HAS client to dynamically adjust its behavior by interacting with the network environment. Applying learning enables the client to adapt its behavior to network conditions that were not under consideration when designing the typical deterministic quality selection algorithms.

To apply reinforcement learning on the HAS use case, the available actions, the environmental state and the reward function have to be modelled. The agent's action is to select one of the available quality levels to request for the next video segment, given the perceived network state. A state definition is needed to model the behavior of the networking environment the agent interacts with. When defining the state definition, care has to be given to the introduced number of states. Too many states will slow down learning and introduce the danger of overfitting, while too few states hamper the ability to correctly model the environment, leading to unsatisfactory results.

The initial approach to a self-learning HAS client [41] consisted of an environment model with more than 2.5 million states, leading to issues in terms of convergence. Furthermore, the vast environment model made the client unapplicable in situations with variable bandwidth. In the learning agent proposed by Claeys et al. [39], the state is constructed by only two parameters, found to be essential to model the networking environment: the available bandwidth perceived by the client and the current client buffer filling level, i.e. the total duration of the segments stored in the client buffer. Both of them are continuous values, which need to be discretized to be modeled as state variables. The value ranges and number of discretization levels of these state elements are shown in Table 7. In this table, B_{max} denotes the maximum client buffer size in seconds while T_{seg} and N respectively denote the segment duration in seconds and the number of quality levels. BW_{max} is the highest possible throughput, e.g. the physical link capacity.

Since the agent's goal is to maximize the numerical reward, we want the reward function to be a measure of the QoE. There are three factors that impact the video quality as perceived by the user [42]: (i) the average segment quality level, (ii) the switching behavior of quality levels and (iii) video freezes, caused by buffer

Table 7: Proposed environmental state definition.

State element	Range	Levels
Buffer filling	[0 ; B_{max}]sec	$\frac{B_{max}}{T_{seg}} + 1$
Bandwidth	[0 ; BW_{max}]bps	$N + 1$

starvations. For each of these aspects, a simple linear reward component has been constructed, as shown in (11), (12) and (13).

$$\mathrm{R}_{\mathrm{quality}} = Q_k - N \tag{11}$$

$$\mathrm{R}_{\mathrm{switches}} = -1.0 * |Q_k - Q_{k-1}| \tag{12}$$

$$\mathrm{R}_{\mathrm{bufferfilling}} = \begin{cases} -100 & : B_k = 0 \\ B_k - B_{max} & : B_k > 0 \end{cases} \tag{13}$$

In these equations, Q_k and B_k respectively denote the quality level requested for segment k and the buffer filling at the time of that request. The components drive the agent to higher quality levels, less switches and higher buffer filling. Since the highest priority is to avoid video freezes at any time, a strong penalization is given to an empty buffer. The exact penalization value is of less importance, but has to be significantly higher than the other components of the reward function. The total reward function can be defined as specified in (14).

$$\mathrm{R} = \mathrm{R}_{\mathrm{quality}} + \mathrm{R}_{\mathrm{switches}} + \mathrm{R}_{\mathrm{bufferfilling}} \tag{14}$$

Claeys et al. show that the Q-Learning based HAS client is able to outperform the deterministic traditional Microsoft IIS Smooth Streaming algorithm by up to 13% in a mobile network environment. Furthermore, the Q-Learning-based client is shown to be very well suited to react to video shifts. Since previously obtained knowledge can be reused when learning on a new video sequence, the learning agent can be trained offline in a variable bandwidth environment before being applied in an online HAS client. After an offline training phase of about 200 episodes, the self-learning HAS client is able to instantly adapt to network and video changes for all considered scenarios, making it applicable in a practical environment.

Multi Agent Systems When more than one agent interacts with each other, the resulting system is called a multiagent system [43]. A multiagent system (MAS) can be defined as a group of autonomous, interacting entities sharing a common environment, which they perceive with sensors and upon which they act with actuators [44]. Depending on the application, the interaction between the agents in a MAS can either be cooperative or competitive. As stated by [45], multiagent systems are ideal for problems that require autonomous decision making capabilities. Specifically, multiagent systems are rapidly finding applications in a variety of domains, including robotics, distributed control, telecommunications, and economics [44]. Needless to mention, dynamic resource allocation in network virtualisation can be classified both under distributed control and telecommunications.

A Multi-Agent Q-Learning-based Framework for Achieving Fairness in HTTP Adaptive Streaming As shown previously in this Section, a Q-Learning-based HAS client, in charge of dynamically selecting the best quality level on the basis of its past experience, is able to outperform current HAS heuristics, achieving a better QoE even under dynamic network conditions. In a real scenario, multiple clients

Algorithm 1 $Q - Learning(States, Actions, \lambda, \alpha)$

Initialise: $States, Actions, \lambda, \alpha$
Initialise $q - values$, $Q(s, a)$, arbitrarily
repeat
  for Each Learning Episode **do**
    Choose a_p from s using policy derived from Q (e.g. ϵ-greedy)
    Take action a_p, observe r_p, s_n
    $Q(s_p, a_p) \leftarrow (1 - \alpha)Q(s_p, a_p) + \alpha \{r_p + \lambda \max_{a \in \mathcal{A}} Q(s_n, a)\}$
    $s_p \leftarrow s_n$
  end for
until Learning ends

simultaneously request content from the HAS server. Often, clients have to share a single medium and issues concerning fairness among them arise, meaning that the presence of a client has a negative impact on the performance of others. Moreover, a fundamental aspect we have to consider when dealing with learning in a multi-agent setting, is that the learning process of an agent influences that of the other agents. For example, when a client selects the $i - th$ quality level, it is using a portion of the shared bandwidth. This decision can have an impact on the performance of the other clients and thus also on their learning process. This mutual interaction can lead to unstable behavior (e.g. the learning process never converges) or unfair behavior (e.g., some agents control all the resources to the detriment of the other ones). Also traditional HAS clients present fairness issues. The main drawback here is that HAS heuristics are static and uncoordinated. This entails they are not aware of the presence of other clients nor can adapt their behavior to deal with it. Classical TCP rate adaptation algorithms are not effective in this case, since quality selection heuristics partly take over their role, as they decide on the rate to download.

We investigate here the aforementioned problems arising in a multi-client setting. Particularly, we present a multi-agent Q-Learning-based HAS client, which is able to achieve smooth video playback, while coordinating with other clients in order to improve the fairness of the entire system [46]. This goal is reached with the aid of a *coordination proxy*, in charge of collecting measurements on the behavior of the entire agent set. These information are then used by the clients to refine their learning process and develop a fair behavior. The single-agent Q-Learning client proposed by Claeys et al. [39] and described previously has been used as basis for our multi-agent algorithm.

The logical work-flow of our multi-agent algorithm is shown in Fig. 10. First, each client requests at the HAS Server the next segment to download at a certain quality. Based on this information, the coordination proxy estimates the reward the agents will experience in the future. This data is then aggregated into a *global signal*, representing the status of the entire set of agents. This global signal is then returned to the agents, which use this information to refine their learning process. In particular, the global signal informs an agent about the difference between its performance and that of the entire system. In this way, the agents can learn how to modify their behavior to achieve similar performance, i.e. fairness. In light of the above, the reward estimation and global signal computation have to be simple enough to avoid overloading the coordination proxy and maintain scalability.

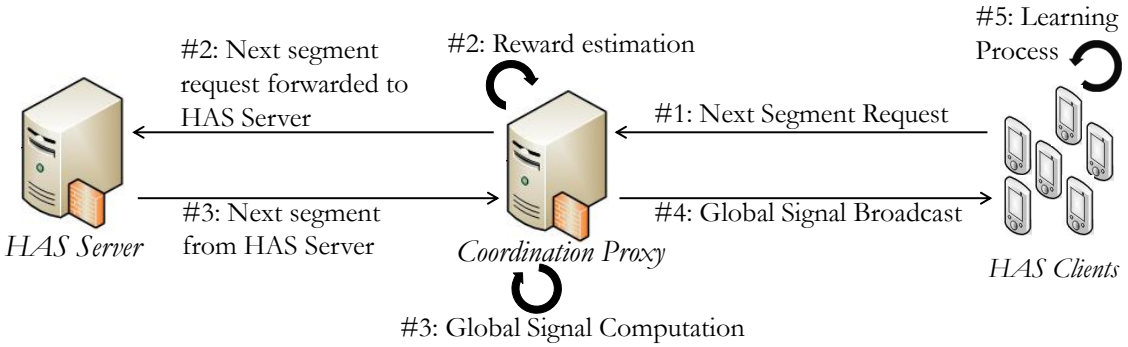

Figure 10: Logical work-flow of the proposed solution.

The coordination proxy can compute an estimate of the reward $r_i(k)$ the client will experience at the $k-th$ step, exploiting the information sent when requesting a new segment at a certain quality. In particular, for each client i, the coordination proxy can compute the following:

$$r_i^f(k) = -|Q_{max} - Q_i(k)| - |Q_i(k) - Q_i(k-1)| \tag{15}$$

$r_i^f(k)$ represents an estimate of the local reward the $i-th$ agent will experience at the next $k-th$ step. We refer here to an estimate since the buffer filling level term is lacking, as can be seen comparing $r_i^f(k)$ with reward shown in Eq. 14. The video player buffer filling level $b_i(k)$ is not accessible to the coordination proxy, being computed by the client only when the new segment will be received, i.e. when the $k-th$ step will be actually executed.
After the reward was computed for each client, the coordination proxy aggregates them into a global signal. This value represents the global status of the entire system and helps agents achieving fairness. The global signal is computed as the average of clients' estimated reward:

$$gs(k) = \frac{1}{N}\sum_{i=1}^{N} r_i^f(k) \tag{16}$$

The global signal can then be added as an HTTP header field and returned to the agents when downloading the next segment to play.
In order to enforce fairness among clients, the global signal is used inside the reward of the agents, as shown in Eq. 17:

$$r_i^{he}(k) = r_i(k) - \alpha \max(r_i^f(k) - gs(k), 0) - \beta \max(gs(k) - r_i^f(k), 0) \tag{17}$$

With $r_i(k)$ being the local reward reported in Eq. 14. It can be noted that in this formulation the coordination proxy acts as a macro-agent representing the behavior of the entire system. The reward reaches the maximum when $r_i^f(k) = gs(k)$, i.e. when the behavior of the agent matches that of the macro-agent. When the reward is far from the global signal, the punishment term operates to modify the agent policy. This way, agents' reward will converge to a similar value, i.e similar performance is achieved and, consequently, fairness.
In order to enforce the learning process, we also add the global signal $gs(k)$ to the agent state, in addition to the perceived bandwidth and video player buffer filling

level. This way, the agent can also consider the overall system behavior when requesting a new segment. For example, if $gs(k)$ is close to zero, i.e. the overall system performance is good, the agent will learn to select a quality level to maintain this condition. We discretize $gs(k)$ into three intervals, to represent the conditions when the agent set is performing bad ($gs(k) \approx 2 \times (1 - Q_{max})$), normal or well ($gs(k) \approx 0$).

We investigated the performance of the proposed multi-agent HAS client, in comparison with both the single-agent HAS client studied by Claeys et al. [39] and a traditional HAS client, the Microsoft ISS Smooth Streaming (MSS). Clients performance are expressed in terms of QoE, while fairness can be expressed as the standard deviation of clients' QoE. In the evaluated bandwidth scenario, we were able to show that our multi-agent HAS client achieves a better video quality and a remarkable improvement of fairness, up to 60% and 48% in the 10 clients case, compared to MSS and the Q-Learning-based client, respectively.

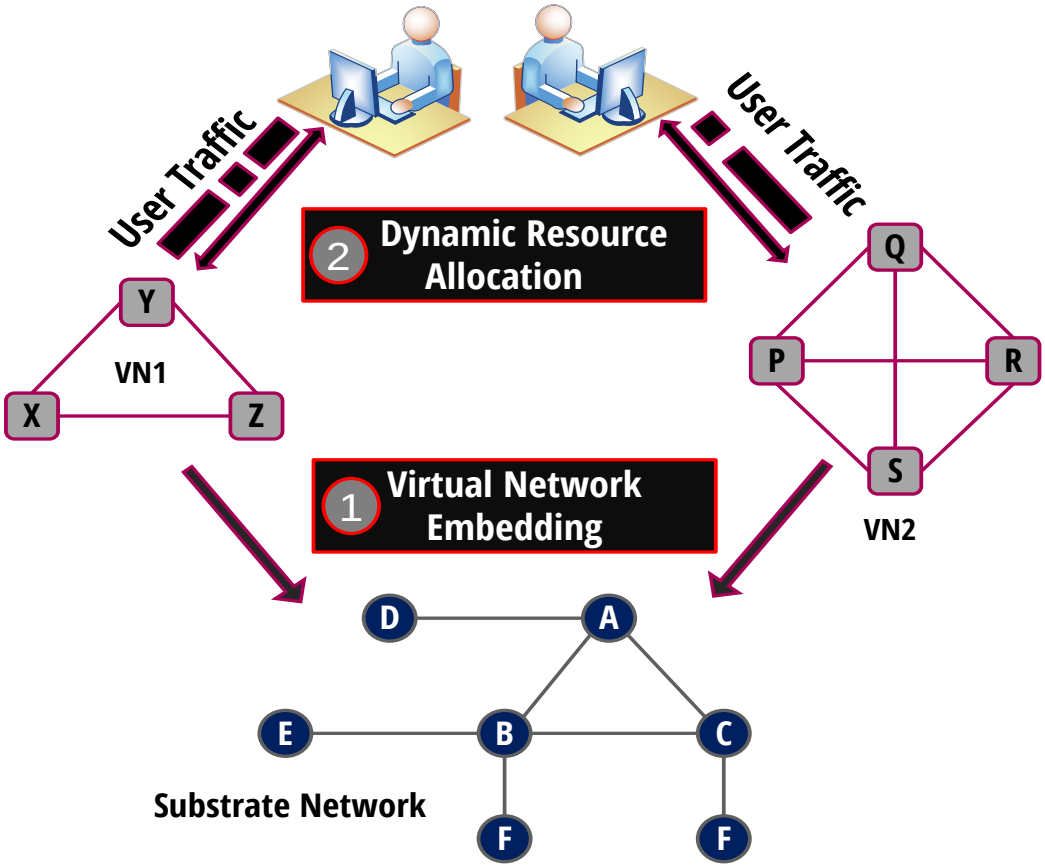

Figure 11: Dynamic Resource Allocation in Network Virtualisation

Dynamic Resource Allocation in Virtualised Networks The dynamic resource allocation problem considers that virtual network resource management does not stop at the embedding of virtual nodes and links to substrate nodes and paths. Instead, as illustrated in Fig. 11, after a successful virtual network embedding, there should be a lifecycle management of resources allocated/reserved for the mapped VN, aimed at ensuring efficient utilization of overall SN resources. Our consideration is that SPs reserve resources to be used for transmitting user traffic, and therefore, after successful mapping of a given VN, user traffic in form of packets is transmit-

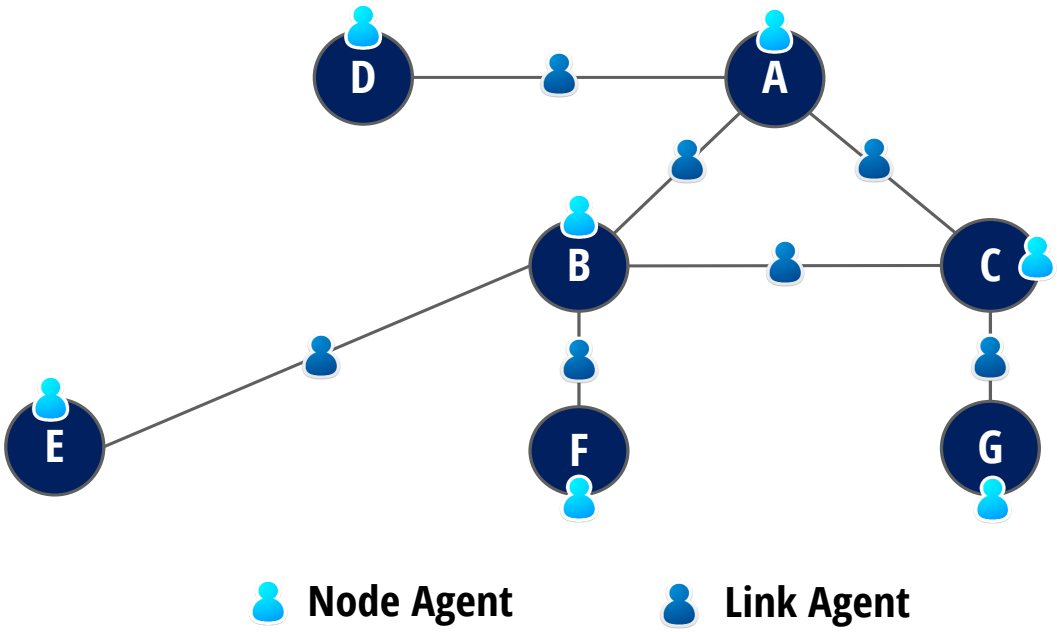

Figure 12: Substrate Network Modeling

ted over the VN. Actual usage of allocated resources is then monitored and based on the level of utilization, we dynamically and opportunistically adjust allocated resources. The opportunistic use of resources involves carefully taking advantage of unused virtual nodes and link resources to ensure that new VN requests are not rejected when resources reserved to already mapped VNs are idle. It is however a delicate trade-off which also ensures that the VNs always have enough resources to guarantee that QoS parameters are kept as established in the corresponding SLAs (or VN request specifications).

Reinforcement Learning-based Dynamic Resource Allocation

Virtual network embedding allocates resources to virtual nodes and links based on the specification in the VN requests. Stopping at the embedding stage would result in a static allocation in which a fixed amount of substrate network resources is reserved for each virtual link and node irrespective of actual utilization. The approach proposed in this Chapter is to dynamically adjust the resource allocation using RL. To this end, we represent the substrate network as a multiagent system shown in Fig. 12, in which each substrate node and link is represented by a node agent $n_a \in \mathcal{N}_a$ and a link agent $l_a \in \mathcal{L}_a$, where $\mathcal{N}_a$ and $\mathcal{L}_a$ are the sets of node agents and link agents respectively. The node agents manage node queue sizes while the link agents manage link bandwidths. The agents dynamically adjust the resources allocated to virtual nodes and links, ensuring that resources are not left under utilized, and that enough resources are available to serve user requests. We consider that each $n_a \in \mathcal{N}_a$ has information about the substrate node resource availability as well as the resource allocation and utilization of all virtual nodes mapped onto the substrate node. In the same way, we expect that each $l_a \in \mathcal{L}_a$ has information about substrate link bandwidth as well as the allocation and utilization of these resources by all virtual links mapped to it. In case a given virtual link is mapped onto more than one substrate link, then each of the $l_a \in \mathcal{L}_a$ agents coordinate to ensure that their allocations do not conflict.

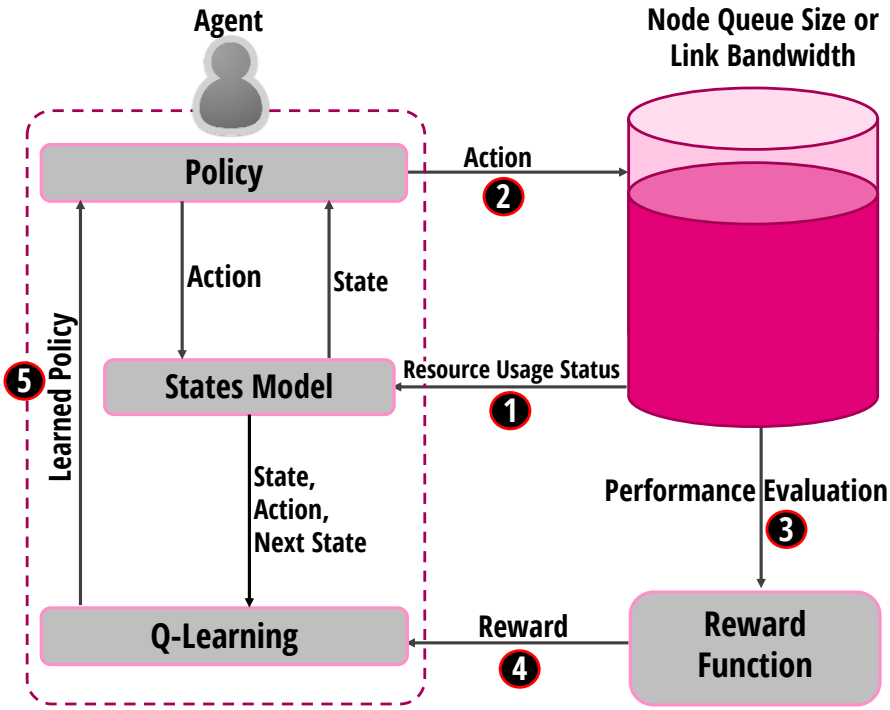

Figure 13: Learning System Modelling: Case of a single substrate node/link

While RL is well studied, its application to dynamic resource allocation in network virtualisation is not trivial. In particular, it requires that all aspects of the RL scenario represented in Fig. 8 are modelled in the context of a network virtualisation environment. In Fig. 13, we show the proposed RL model for a given substrate node/link. As can be noted, the learning is made up of five steps. The agent starts by getting a resource usage status, which could have information such as the percentage of substrate node/link resources available and the ratio of total virtual node/link demand currently allocated. With this information, the agent takes an action, which could involve increasing or reducing the amount of resources allocated to the virtual node/link. The virtual node/link is then monitored to evaluate its performance e.g. in form of link delays (in case of virtual links) or packet drops (in case of virtual nodes). This evaluation is communicated to the agent in form of a reward, and based on this, the agent adjusts its policy to ensure that its future resource allocation actions are better.

Artificial Neural Networks (ANN) ANNs are computational methodologies inspired by networks of biological neurons to perform multifactorial analyses [47]. A major strength of neural networks is their ability to handle high-dimensional input, as they do not suffer from the curse of dimensionality [48]. Therefore, although their complexity can make them difficult to harness, neural networks can be very efficient. In particular, they have excellent generalization capabilities, which can help to solve difficult reinforcement-learning tasks, especially when the dimension of the state space is high [49].

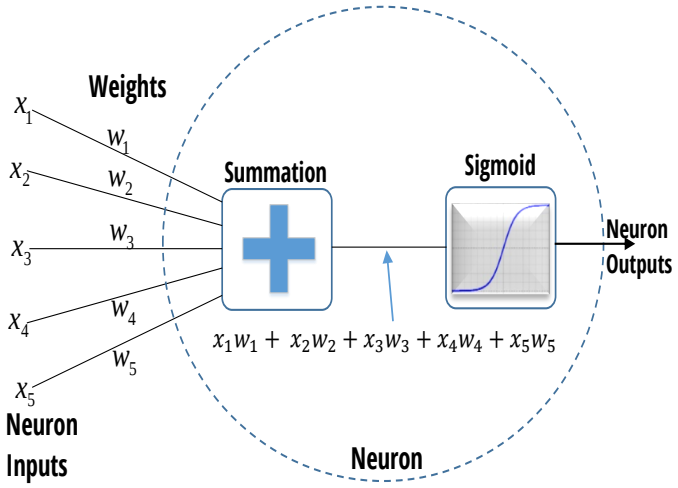

Figure 14: Structure of a Neuron

Structure of a Neuron Neural networks are made up of simple interconnected computing nodes known as *neurons*; which are connected by *links*. As shown in Fig. 14, a neuron operates as nonlinear summing device. It receives one or more inputs, which are first multiplied by *weights*[12] along each link, and then *summed* to produce a weighted sum. The weighted sum is then passed through an activation function (such as the logistic (or sigmoid) function [50]), which determines the input-output behavior of the neuron.

Neural Network Structure In ANNs, neurons are arranged in layers, with each layer consisting of one or more neurons. Fig. 15 shows the most commonly used structure of ANNs, which is made up of 3 layers; an input layer, a hidden layer and an output layer [51]. Each layer consists of one or more neurons. While in general information can flow in both directions (with feedback paths), what we consider in this Chapter are ANNs where information flow is from input to output. These are called feed forward neural networks. It is also important to note that the neurons in the input layer are passive (they do not modify the data) as each of them receives a single value as its input, and duplicates the value to its multiple outputs and sends it to all hidden nodes (in case of fully connected networks [52]). On the other hand, the nodes of the hidden and output layer are active since they actually modify the data at their inputs.

Learning in Neural Networks Before neural networks can be used for any task, they should be trained. The learning process of neural networks is represented in Fig. 16. Learning in ANNs consists of determining the proper set of connection weights to estimate a given training set. This requires that for every input, a desired/target output must be known to determine the error. The expected output is compared with an actual output to determine an error. This error is used to adjust the weights along the links of the neural network, with the objective of gradually minimizing the error. Training continues until a neural network produces outcome values that match the known outcome values within a specified accuracy level, or until it satisfies some other stopping criteria. The most popular method for learning in ANNs is called back-propagation (BP) [23].

[12] The weights are a set of predefined numbers.

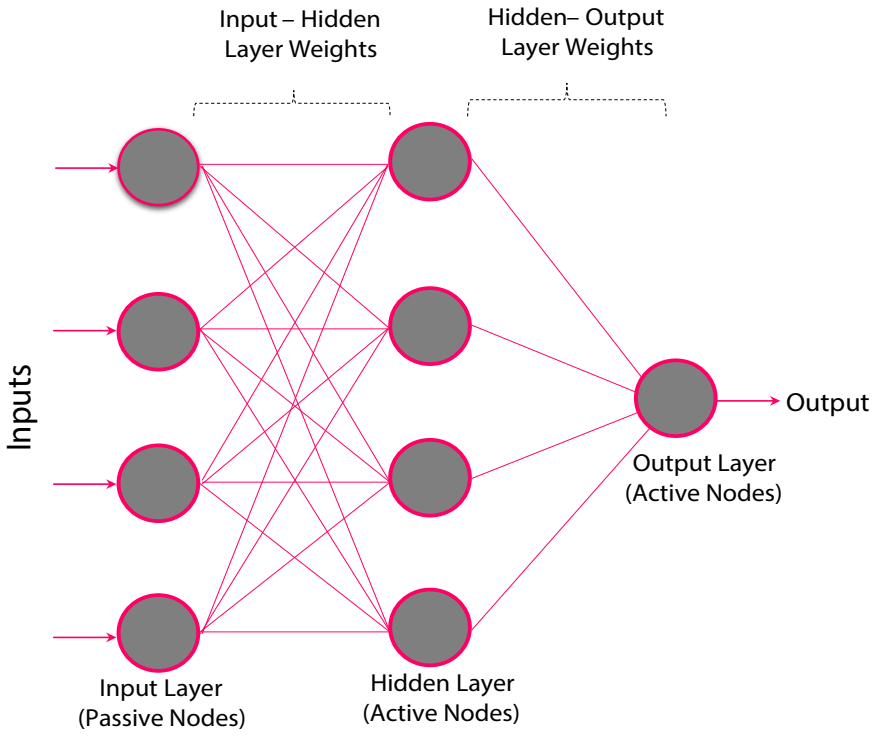

Figure 15: Example of a Neural Network

Back-propagation (BP) Algorithm Just like the algorithm's name, in BP, after an output is obtained, an *error signal* is determined, and "propagated backwards" from the output layer to the input layer. This is achieved by calculating the gradient of the error of the network regarding the network's modifiable weights [53]. This gradient is almost always used in a simple stochastic gradient descent algorithm to find weights that minimize the sum squared error. Often the term "back propagation" is used in a more general sense, to refer to the entire procedure encompassing both the calculation of the gradient and its use in stochastic gradient descent [54].
To illustrate the back propagation algorithm, Fig. 17 shows a block diagram with the processes involved. The weights between the input layer (i) and the hidden layer (j) are represented as w_{ij} while those between the hidden layer and the output layer (k) are represented as w_{jk}. Therefore, in BP, for every input, the output is determined (feed forward step). Using the target output, an error is determined. This error is then propagated backwards (error back propagation), first adjusting the weights between the output layer and hidden layer, w_{jk}, and then those between the hidden layer and the output layer, w_{ij}. In fact, for each set of weights, what the BP algorithm determines is the required change in weights Δw. This is determined using equation (18)[55] for weights w_{kj} as an example.

$$\Delta w_{kj} = \alpha \delta_j y_k \tag{18}$$

where Δw_{kj} is the amount by which weight w_{kj} should be changed to reduce the error. $0 \leq \alpha \leq 1$ is referred to as *learning rate*, and it determines how fast learning occurs. y_k is the output of the node in layer k. δ_j represents the product of the error with the derivative of the activation function [55]. Algorithm 2 shows the back propagation procedure in full.

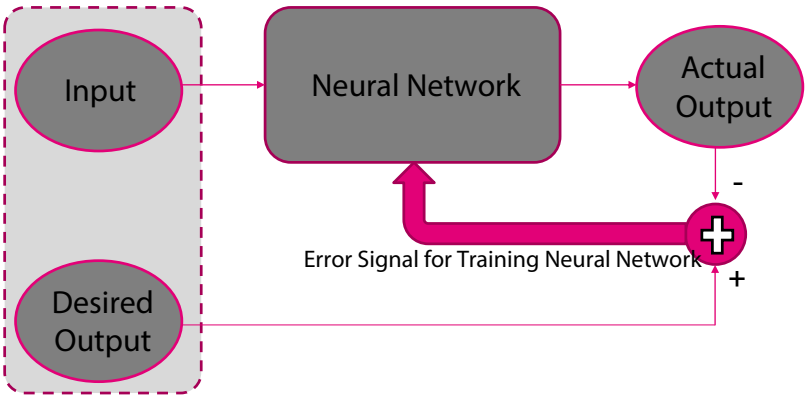

Figure 16: Neural Network Learning Model

Algorithm 2 $BackPropagation$

```
Initialize network weights (often with small random values)
repeat
    for Each training example do
        Feed Forward: Determine actual output
        Get desired output
        Determine error (i.e error = desired - actual) at the output layer
        Error back propagation: Compute Δw_jk for all weights from hidden layer to output layer
        Error back propagation: Compute Δw_ij for all weights from input layer to hidden layer
        update network weights
    end for
until The errors are acceptable
```

Artificial Neural Network-based Dynamic Resource Allocation The system model used for our proposal is shown in Fig. 18. As can be observed from the figure, there are three main components: the multi-agent system representing the substrate network, the ANN that represents the internal components of each agent, and the evaluative feedback block that produces the error signal. In the following subsections, each of these elements of the model is detailed.

Artificial Neural Network Our proposal uses a 3-layer ANN. An important design issue of any ANN is determining network topology, i.e. number of neurons in each of the network layers.

Input Layer We model the state of any virtual resource (node queue size or link bandwidth) v hosted on a substrate resource z, by a 3-tuple, $s = \left(R_a^v, R_u^v, R_u^z\right)$, where R_a^v is the percentage of the virtual resource demand currently allocated to it, R_u^v is the percentage of allocated resources currently unused, and R_u^z is the percentage of total substrate resources currently unused. Therefore, the input layer consists of 3 neurons, one for each of the variables R_a^v, R_u^v and R_u^z.

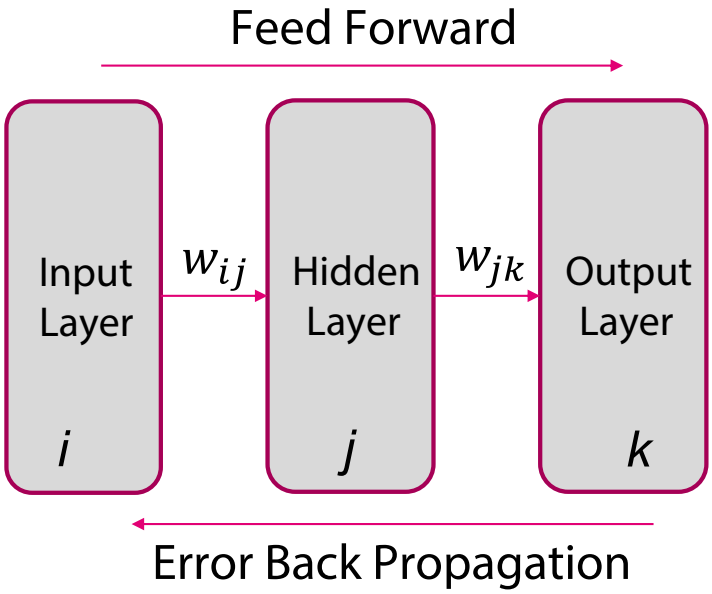

Figure 17: Neural Network Back Propagation

Output Layer During each learning episode, an agent should perceive the state s and give an output. This output, a, is a scalar that indicates which action should be taken to change the resource allocation for the virtual resource v under consideration. The action may be aimed at increasing (if it is positive) or reducing the resources allocated to any virtual node or link respectively. Therefore, the output layer consists of 1 neuron, representing the action, a. To illustrate the effect of an action, if a given virtual resource v has total allocated resources v_r and the agent action is a (where $-1 \leq a \leq +1$), then the resulting resource allocation is: $v_r = v_r + a \times v_t$, where v_t is the total initial demand of the virtual resource (as specified in the VN request before the VNE). It is worth mentioning that the agent only takes a given action if it does not violate resource allocation requirements, for example, at any point, $v_r \geq 0$ and $v_r \leq v_t$.

Hidden Layer The optimal number of neurons in a hidden layer of any ANN is problem specific, and is still an open research question [50]. In this Chapter, we determine this number by experimentation. We perform a search from the number of hidden layer neurons, $N_{HL} = 1$ to 15.

In order to achieve this, we need a test dataset. The dataset used for this purpose was saved from the q-table of a RL approach proposed in [56]. This q-table was a result of a learning system for a similar DRA task and it gives the state-action-values for the learning task. This dataset contains 512 entries, each showing the best action value in each of the possible 512 states.

With the above training set, a $10-$fold cross-validation was performed in Weka 3.6 [57], using the default parameters (learning rate, validation threshold, momentum, etc.) for the multilayer perceptron in Weka. Fig. 19 shows the average root mean square error (RMSE) values for 20 experiments. From the figure, the optimal number of neurons for the hidden layer is 4. The reason for choosing to use 4 neurons is not only to allow for a low RMSE, but also to avoid a possibility of over-fitting which could be caused by a network

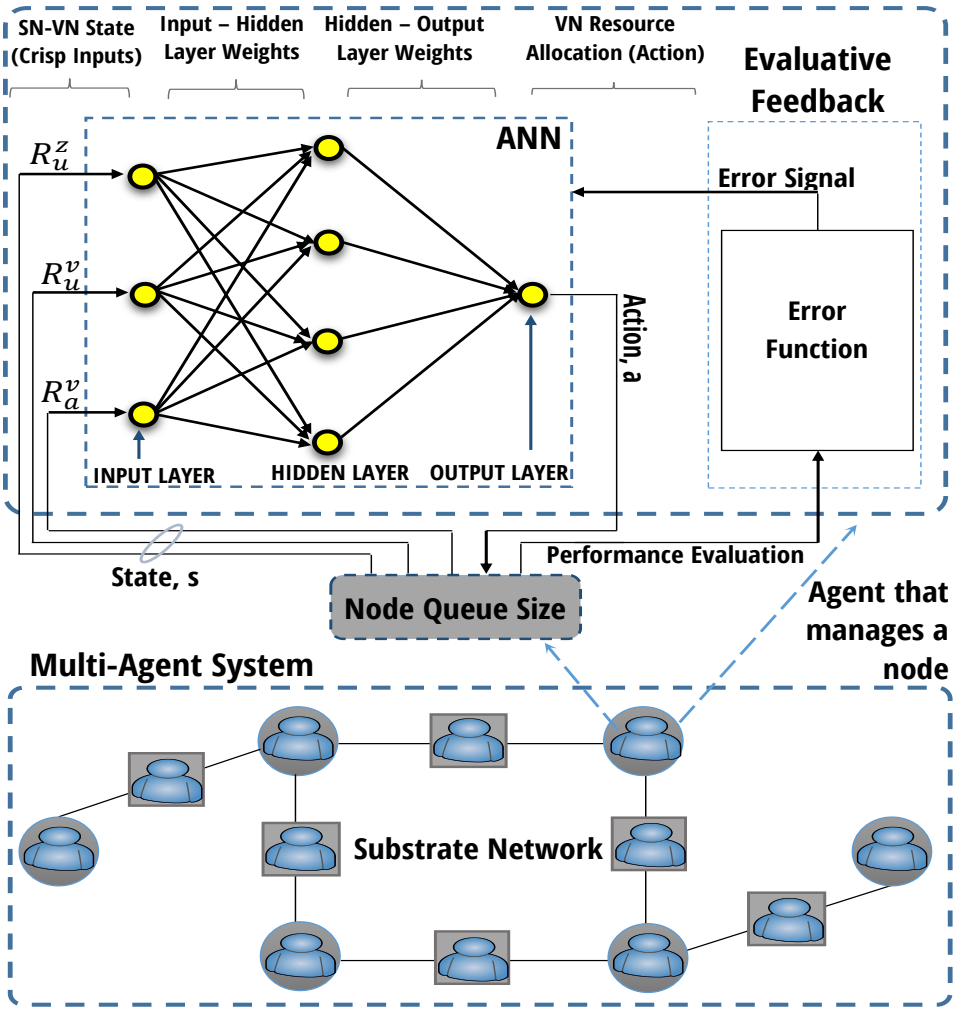

Figure 18: Artificial Neural Network-based Resource Allocation Model

with too many neurons. This experimentation also provides initial weights that are used to initialise the neural network, and hence avoid the slow learning characteristics (and hence slow convergence) of BP.

ANN Learning Since for the network virtualisation problem under consideration it is not possible to have a target output for each input, the training of the ANN is achieved by determining the error using a reinforcement learning-like approach. Therefore, our approach combines both ANNs and RL. One of the most remarkable achievements in the combination of ANNs and RL in machine learning research was its implementation as a backgammon player [58]. In addition, other combinations of ANNs and RL have been applied to many problems such as [59, 60, 49]. In these proposals, ANNs are used as function approximators for the RL policy. Our proposal differs from these works on two fronts: (1) we use RL to train the ANN rather than using ANNs to approximate the RL policy. This, remarkably, allows us the possibility to dispense with the need for training examples and/or target outputs usually needed for learning in neural networks[13], and (2) we apply the combination

[13]While we still use some training examples in our proposal (see Section 3.5.3), it is only aimed at guiding in the ANN

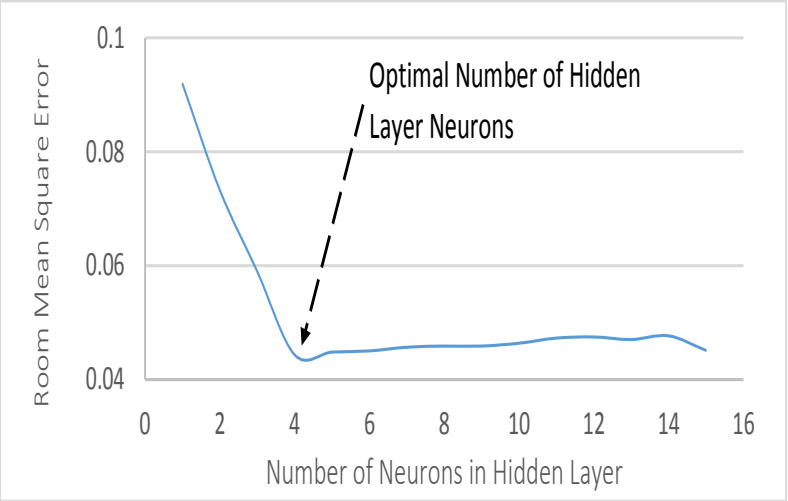

Figure 19: Variation of RMSE with Number of Neurons in Hidden Layer

ANN and RL to a network virtualisation environment.

Evaluative Feedback After each learning episode, the affected substrate and virtual nodes/links are monitored, taking note of average utilization of substrate resources, the delay on virtual links and packets dropped by virtual nodes due to buffer overflows. These values are fed back to the agent in form of a *performance evaluation*, used by the error function to produce an error signal, which is used by the BP algorithm to adjust the weights of the ANN, and hence improve future actions.

Error Function The error $e(v)$, is an indication of the deviation of the agent's actual action, from a target action. The objective of the error function is to encourage high virtual resource utilization while punishing $n_a \in \mathcal{N}_a$ for dropping packets and $l_a \in \mathcal{L}_a$ for having high delays. Good actions by an agent are characterized by an $e(v)$ equal or close to 0, while any deviations indicate undesirable actions. Therefore, the value of $e(v)$ gives the degree of desirability or undesirability of the agent's action, and is dependent on resources allocated to the virtual resources, underutilized resources, link delay in case of $l_a \in \mathcal{L}_a$ and the number of dropped packets in the case of $n_a \in \mathcal{N}_a$. The proposed error function is shown in (19).

$$e(v) = \begin{cases} \left(R_u^v + \alpha P_v\right) & \forall n_a \in \mathcal{N}_a \quad (19a) \\ \left(R_u^v + \beta D_v\right) & \forall l_a \in \mathcal{L}_a \quad (19b) \end{cases}$$

where α and β are constants aimed at ensuring that the magnitudes of the two terms in each of (19a) and (19b) are comparable. The values $\alpha = 0.05$ and $\beta = 40$ used in this Chapter, were determined by simulations. Therefore, to make these values comparable to $0 \leq R_u^v \leq 1$, we divide each value by 20 (multiply it by $\alpha = 0.05$). P_v is the number of packets dropped by node $n_a \in \mathcal{N}_a$ from the time the allocation action was taken, and D_v is the $extra$ delay encountered

structure design as well as ensuring a faster convergence of the algorithm (through problem specific weight initialisation) rather than as a requirement as would have been in a typical ANN learning scenario.

by a packet using $l_a \in \mathcal{L}_a$. The extra delay is calculated as the difference between actual delay and the theoretical delay. We define theoretical delay as the delay the virtual link would have if it was allocated 100% of its bandwidth demand. The actual delay is determined as the difference between when a packet is received at one end of the link, to when it is delivered to the other end. Once the error is determined, the ANN weights are adjusted using BP.

Distributed Algorithms A specific type of algorithm that uses various interconnected processors is called distributed algorithm. Therefore, the algorithm is responsible for the organization of running the different parts of the algorithm at the same on different processors. Each processor performs a different part of the overall algorithm at the same time and submits then the results. The following paragraph describes the Logical Topology Construction Algorithms for Decentralized Network Resource Management developed within FLAMINGO.

Logical Topology Construction Algorithms for Decentralized Network Resource Management Deliverable D6.1 presented DACoRM [61], an in-network resource management framework developed by UCL to support dynamic resource reconfiguration functionality in fixed backbone networks. In DACoRM, the decision-making process is distributed across a set of network nodes, so that each node is responsible for deciding on reconfiguration actions based on local feedback regarding the state of the network. The network nodes participating in the resource management process form a management substrate, which is a logical structure used to facilitate the exchange of information between distributed decision-making points. The DACoRM framework was used to support three different applications, namely adaptive traffic engineering [62][61], energy efficiency [3] and in-network cache management [63].

As described in Deliverable D6.1 section 3.3.7, the performance of the proposed management scheme, in terms of communication overhead, can be affected by the structure of the management substrate. The use of three structures was proposed to organize nodes in the management substrate, namely a full-mesh, ring and hybrid topologies. In the full-mesh model, each node is logically connected to every other node in the substrate, each node is logically connected to two other nodes only in the case of the ring. The hybrid model consists of a set of rings inter-connected in a fully-meshed fashion through proxy nodes, so that there exists exactly one proxy in each ring. The three structures are depicted in Figure 17 of Deliverable D6.1.

In order to optimize the structure of the ring and hybrid models, a set of structure construction algorithms has been developed [64]. Given a set of network nodes and the underlying physical topology, the proposed algorithms compute the structure of each model with the objective of minimizing the latency and volume of signalling messages among decision-making entities. The algorithms are the followings.

Ring Construction Algorithm The objective of the Ring Construction Algorithm is to determine the order according to which nodes are connected in the ring, so that the total delay is minimized. This problem is similar to the Travelling Salesman Problem (TSP). Although a number of approaches with near-optimal performance exists in the literature to solve the TSP, an approach based on the simple *Nearest Neighbors* tour construction heuristic has been developed. This relies on a sequential process where nodes are considered iteratively, so that the successor of each node i is the node j which is closest, in terms of distance, to node i.

Multiple Rings Construction Algorithm The objective of this algorithm is to partition nodes into clusters and compute the resulting sub-rings. The clusters are formed based on the proximity of nodes in terms of distance and the sub-rings are computed according to the Ring Construction Algorithm. The algorithm provides a set of clusters and the ring structure within each cluster.

Cluster Proxy Selection Algorithm

The objective of the Cluster Proxy Selection Algorithm is to determine which node to select in each cluster to act as a proxy, i.e. interface between the different sub-rings. The selection is based on the proximity with respect to the distance cost. The proxy node in a sub-ring is the one which is the closest, on average, to every other remote nodes in the substrate.

Risk Management In the context of the Internet of Things, security is a real challenge, however the constrained nature of this type of networks requires trade-off between security solutions and their cost. Risk management consists in dynamically activate or deactivate security mechanisms and can be considered as a good trade-off in RPL-based networks. The risk level can be defined as a combination of the potentiality of an attack, its consequences and the exposure of the network. Risk management is a process consisting in monitoring, prioritizing and controlling risks. For instance, when a high potentiality is observed, security mechanisms (being aware of their costs) can be activated to reduce the exposure and maintain the risk level to a low value.

The Risk management process is composed of two main activities: RISK ASSESSMENT and RISK TREATMENT. Risk assessment consists in quantifying the potentiality of attacks. For that, it is necessary to evaluate the performance of detection techniques (based on anomalies or known signatures) in RPL environments, and to identify the network nodes able to perform this activity. Risk assessment aims also at quantifying the consequences of successful attacks. The objective is to assess the relative importance of nodes in the RPL network, and to analyze how the attack against a given node may impact on the functioning of the overall network. The risk treatment activity consists then in selecting and applying the security mechanisms that are needed. The activities of suspicious nodes can be mitigated, or the nodes can be (partially) excluded from the RPL network. For instance, the number of requests from them may be restricted over time, or the considered RPL nodes may not be allowed to act as routers anymore. The selection of countermeasures takes into account the costs induced by their activation on the RPL network. As previously mentioned, this cost is often not negligible in such a critical environment.

3.5.4 Experimentation

The evaluation of proposed approaches and techniques is one of the most important aspects to validate solutions in research. The reliability and validity of approaches is a must in technical research. Significant results must be more than a one-off finding and be inherently repeatable so that other researchers are able to perform the same experiment, given the same conditions and generate the same results as well. Furthermore, this will strengthen findings and ensure that the scientific community will accept the hypothesis. If the results obtained meet all requirements of the scientific research method along with the experimental concept it is called valid. The following sections give an overview of experimental tools used within the consortium and highlight work done to ensure valid and reliable results.

Simulation A simulation reproduces the behavior of a system. The simulation can run on one single system or a network of (distributed) systems. To simulate a system the simulation uses abstract models. In the past simulations become a indispensable part of mathematical modeling in different sciences. Therefore, a simulation of a system is represented as the running of the system's model, that enables the initiator of a simulation to investigate and research new technology and to estimate systems' performance [65].

COOJA is the dedicated network simulator for the Contiki OS. A simulated Contiki Mote in COOJA is an actual compiled and executing Contiki system. The system is controlled and analyzed by COOJA. This is performed by compiling Contiki for the native platform as a shared library, and loading the library into Java using Java Native Interfaces (JNI). Several Contiki libraries can be compiled and loaded in the same COOJA simulation, representing different kinds of sensor nodes (heterogeneous networks). COOJA controls and analyzes a Contiki system via a few functions. For instance, the simulator informs the Contiki system to handle an event, or fetches the entire Contiki system memory for analysis. This approach gives the simulator full control of simulated systems. The security implications and risk mitigation strategies of the RPL protocol are studied by applying analytical and traffic modeling, along with statistical analysis. Based on this a risk management mitigation approach that uses heuristics to detect attacks is developed and implemented on the TelosB hardware platform, using the Contiki OS and simulated based on the COOJA tool. A thorough evaluation of the approach is performed. Initial investigation is performed into the applicability of a Trust Computing Module (TPM) in RPL based constrained networks. A key exchange mechanism, based on the TPM is developed and preliminary simulative evaluation is performed using COOJA. MSPSim is used to obtain performance insights, via COOJA, regarding usage of the developed scheme on TelosB hardware.

Discrete Event Simulation

ns3 [14] is a discrete-event network simulator for Internet systems, primarily targeted at research and educational use. ns-3 is built using C++ and Python with scripting availability and is split over a couple of dozen modules, containing one or more models for real-world network devices and protocols. For example, there are several TCP-implementations available, such as *TCP New Reno*, *TCP Westwood* and *TCP Tahoe*. The general process of creating a simulation in ns-3 can be divided into several steps:

Topology definition to ease the creation of basic facilities and define their interrelationships, ns-3 has a system of containers and helpers that facilitates this process.

Model development models are added to simulation (for example, UDP, IPv4, point-to-point devices and links, applications); most of the time this is done using helpers.

Node and link configuration models set their default values (for example, the size of packets sent by an application or MTU of a point-to-point link); most of the time this is done using the attribute system.

Execution simulation facilities generate events, data requested by the user is logged.

Performance analysis after the simulation is finished and data is available as a time-stamped event trace. This data can then be statistically analysed with tools like R to draw conclusions.

[14] http://www.nsnam.org

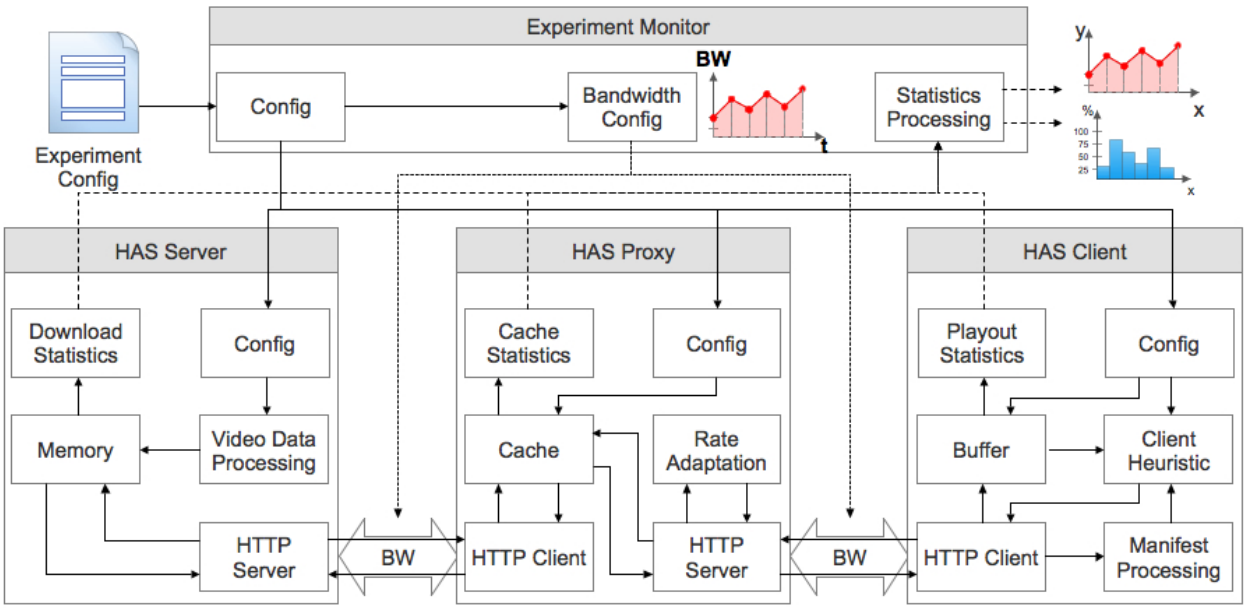

Figure 20: Overview of the NS-3 based HTTP Adaptive Streaming simulator framework.

Graphical Visualization raw or processed data collected in a simulation can be graphed using tools like Gnuplot, matplotlib or XGRAPH.

ns-3 allows you to develop and test a wide variety of services by implementing applications on top of the network protocols. For example, a HTTP adaptive streaming framework can be developed on top of ns-3, allowing to simulate the behavior of the various components in the delivery network and optimize the streaming client's behavior. In order to perform a realistic evaluation of the characteristics of the different quality selection approaches, a packet-based simulator was built upon ns-3. Figure 20 gives a conceptual overview of the work-flow of this simulator. Three main components exist: (i) a *HAS Server* taking realistic video statistics as input, such as the number of segments and their corresponding sizes (ii) a *HAS Proxy* acting as a broker towards the clients for the *HAS Server* and (iii) a *HAS Client* requesting videos at certain quality levels and outputting the related statistics concerning received quality, delay, perceived sMOS and quality switches. All these components communicate over HTTP using a ns-3 channel with configurable characteristics such as available bandwidth and end-to-end delays.

Although the possibilities of ns-3 are numerous, there are some shortcomings, such as the absence of selected acknowledgments (SACK) for the TCP protocols. This can be overcome by using the Network Simulation Cradle (NSC) [15], which is a framework for wrapping real-world network code into simulators, allowing simulation of real-world behavior at little extra cost. This work has been validated by comparing situations using a test network with the same situations in the simulator. To date, it has been shown that the NSC is able to produce extremely accurate results. NSC supports four real world stacks: FreeBSD, OpenBSD, lwIP and Linux. Emphasis has been placed on not changing any of the network stacks by hand. Not a single line of code has been changed in the network protocol implementations of any of the above four stacks. However, a custom C parser was built to programmatically change source code.

[15] `http://www.wand.net.nz/~stj2/nsc/`

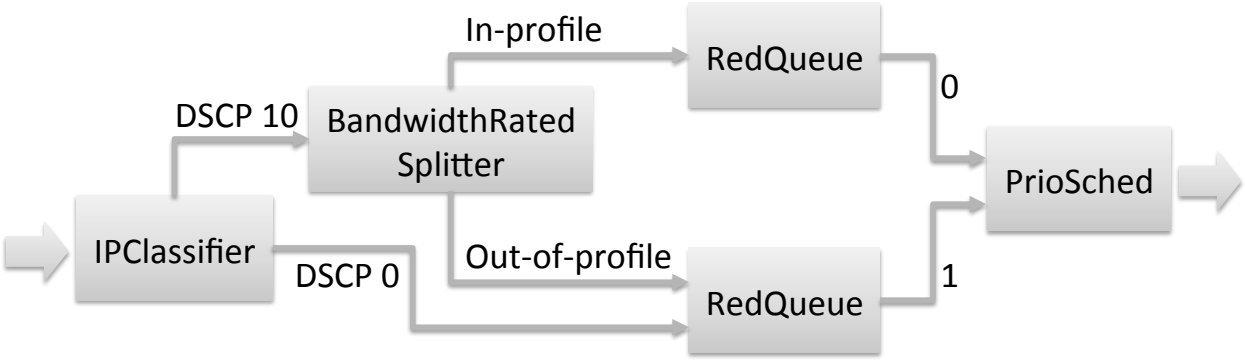

Figure 21: Click-implementation of AF behavior.

Other shortcomings of ns-3 can be overcome by using Direct Code Execution (DCE), which is a framework for ns-3 that provides facilities to execute, within ns-3, existing implementations of userspace and kernelspace network protocols or applications without source code changes. For example, instead of using ns-3's implementation of a ping-like application, you can use the real ping application. It also allows to use the Linux networking stack in simulations.

ns-3 also offers support for Click Modular Routers[16]. Click is a software architecture that allows building configurable routers, by using different combinations of packet processing units called elements. This flexibility provides a good platform for testing and experimenting with different protocols. ns-3 allows the integration of Click routers into your network simulations. For example, a DiffServ-enabled router could be implemented using click and integrated in a ns-3 simulation environment. DiffServ is a simple, scalable mechanism for classifying and managing IP network traffic. DiffServ applies the principle of traffic classification, placing each data packet into a limited number of traffic classes. For this purpose, the 6-bit DS Field of the IP header is used. DiffServ-aware routers each implement Per Hop Behavior (PHB), defining the packet forwarding rules for every traffic class. Next to the default PHB, which is typically best-effort traffic, the Assured Forwarding (AF)[66] and Expedited Forwarding (EF)[67] PHB are commonly used with DiffServ. While EF traffic is often given strict priority queuing above all other traffic classes, inducing low delay, the AF PHB, allows the operator to assure the delivery of this traffic class as long as it does not exceed some subscribed rate. When traffic exceeds the subscription rate, it faces a higher drop probability when congestion occurs. The AF PHB has been implemented in a Click router as shown in Figure 21. A combination of an IPClassifier, a BandwidthRatedSplitter, two Random Early Detection (RED) queues and a PrioSched module were used to implement this behavior. As long as the prioritized traffic is lower than the profile rate, the traffic is marked as in-profile, otherwise it is marked as out-of-profile. When congestion arises, the out-of-profile traffic is dropped more aggressively than in-profile traffic. In absence of congestion, the prioritized traffic is allowed to exceed its profile rate.

Emulation Emulation describes the duplication of functions of a system in another system, that is different from the first one, in a sense that the emulated behavior closely resembles the behavior of the real system. There exists an exact reproduction of the whole behavior. Computer programs (software) in electronic devices are often used to emulate or imitate another

[16] `http://read.cs.ucla.edu/click/`

program or device (e.g. commonly known printers are emulated to support a huge diversity of software). Hardware emulators take the form of a hardware device. They can be implemented completely in software or partly in hardware depending on the type and performance capability of hardware that needs to be emulated.

Network Testbed Network Testbeds give researchers an environment where they can develop, debug, and evaluate their approaches. They act as a platform for experimentation of distributed systems cooperating together via various types of network connection. A typical testbed could include software, hardware, and networking components, that are accessible remotely.

Openstack Openstack [68] is a free and open-source cloud operating system managed by the Openstack Foundation, and supported by a large community including over 6000 individuals and 200 compagnies. Widely used as an IaaS (Infrastructure as a Service) software, it enables enterprises and service providers to offer on-demand computing resources by provisioning and managing large networks of virtual machines. To this end, Openstack is composed of several core projects to provide computation (Openstack Compute), storage (Openstack Storage) and network (Openstack Network) as services. Openstack also provides additional modules (e.g. for identity or OS images management) as well as a web-based platform in order to offer administrative users a graphical interface for accessing, provisionning and automating cloud-based resources.

The Openstack Compute project (also named Nova) is the core component to instantiate and manage networks of virtual machines. These compute resources are directly accessible using APIs for developers or using web interfaces for administrators or regular users. The Openstack Compute architecture is designed to scale horizontally on commodity hardware without any software requirements, thus providing flexibility when designing public or private clouds. Furthermore, it provides the ability to integrate with legacy systems and third party technologies. OpenStack Compute is deployed on common supported hypervisors in a virtualized environment, such as KVM, XenServer and LXC.

The Openstack Storage project is composed of two subprojects named Cinder and Swift, respectively used for both block and object storage. The Cinder project provides persistent block storage to guest virtual machines (VMs), by allowing block devices to be exposed and connected to compute instances for expanded storage and better performance. Cinder can be well integrated with common enterprise storage platforms such as NetApp and SolidFire. On the other hand, the Swift project provides a fully distributed, API-accessible and cost effective storage platform that can be integrated directly into applications or used for backup, archiving and data retention.

The Openstack Network project (also named Neutron) is a pluggable, scalable and API-driven system for managing networks and IP addresses, thus providing network as a service between interface devices managed by other Openstack services (e.g. Nova). It ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self service, even over their network configurations. Furthermore, additional network services such as intrusion detection systems (IDS) virtual private networks (VPN) can be deployed and managed using an extension framework.

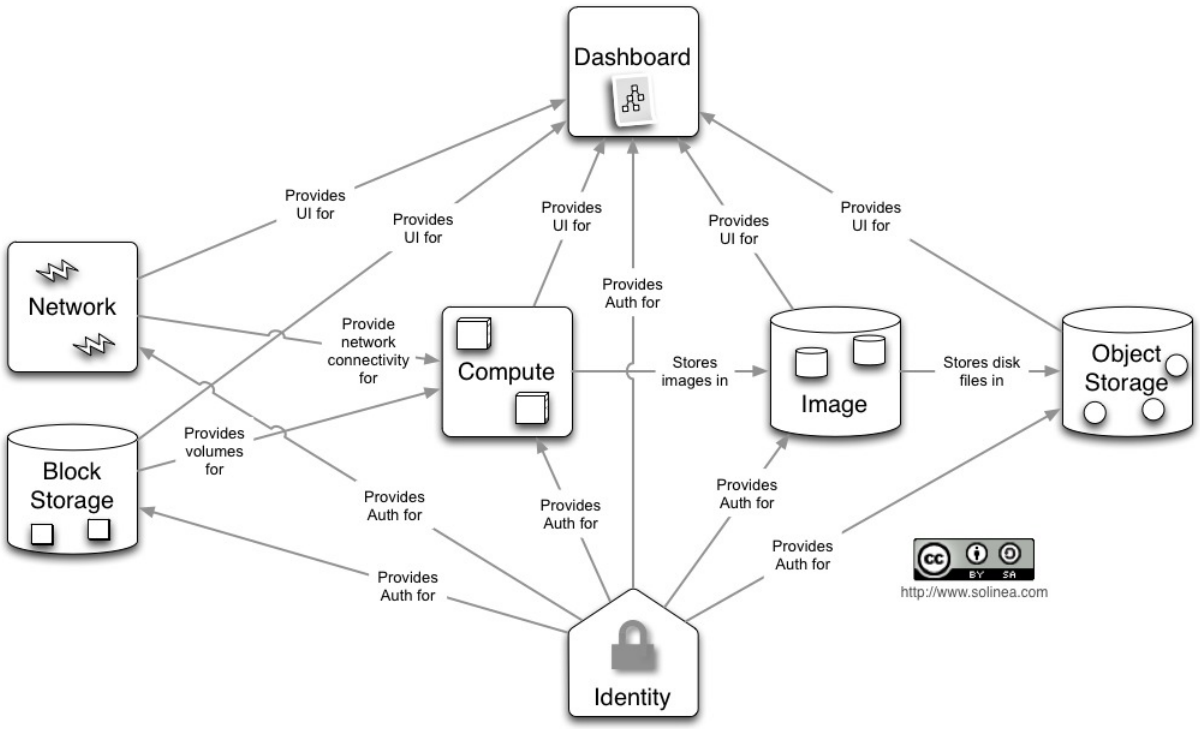

Figure 22: The Openstack architecture [68]

Mininet Mininet [69] provides a framework which is executable on a single laptop not matter how many devices a developer tries to simulate. The idea which builds the basis for this possibility, is the usage of lightweight virtualization. This allows the simulation of hundreds of devices without paying attention on resource consumption. Such an economic usage of resources is grounded on OS-level virtualization features where processes and virtual Ethernet pairs are put together in network namespaces. In addition these environments can be packaged after a simulation study to be exchanges among other interested parties.
Initial remarks of how to start using Mininet can be found on the Mininet webpage[17]. Subsequently a comprehensive Command Line Interface (CLI) is provided for defining and running an experimental environment in Mininet. By the help of emulated links, hosts, switches and controllers, which are part of the Linux OS lightweight virtualization capabilities, the environment will be set up. The interaction with the simulation network is supported by a network-aware command line interface provided by Mininet that is aware of simulated device names and network configurations. This is done by an automatic substitution of host IP addresses and host names.

Virtual Wall Large-scale network experiment scenarios may use a complex network setup that consists of many interconnected machines. Besides manually connecting these machines via network cables, some effort has to be dedicated to the installation of configuration files along with the necessary software packages. Such a setup does not always promote resource sharing and is often time consuming. The iMinds ilab.t Virtual Wall was created in order to solve these issues.

[17] `www.mininet.org`

The Virtual Wall is a test environment that automates experiment setup and configuration procedures. It is based on the Emulab [70] network testbed software. Emulab refers to both the hardware as well as open-source management software of a network testbed created at the University of Utah. The management software created for Emulab also runs at the core of the Virtual Wall. It gives researchers a range of environments in which to develop, debug and evaluate their network protocols and distributed systems. Using a set of management tools, a researcher can test distributed software deployments or evaluate system scalability aspects. The virtual wall infrastructure can simulate packet loss, jitter, traffic shaping and other unfavorable network conditions that occur in real-life scenarios. Some of its other features include:

- automated topology creation (e.g., circular, star, mesh, etc.),
- device configuration,
- operating system and software installations,
- fast experiment switching, and
- hardware virtualization.

Each Virtual Wall consists of multiple re-programmable server, client and router nodes. All server nodes are connected to a non-blocking 1.5 Tb/s VLAN Ethernet switch. The non-blocking aspect of the switch assures that no delays are introduced by the switch itself.

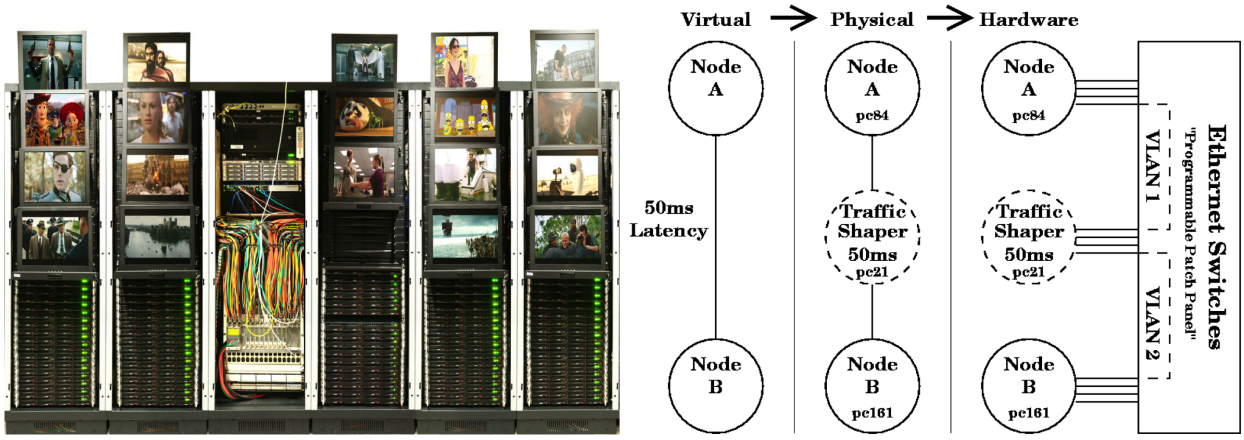

Figure 23: A photograph (left) and schematic overview (right) of the Virtual Wall testbed.

The left picture of Figure 23 shows the original Virtual Wall setup. It consists of 300 servers (organized into Wall 1 and Wall 2) interconnected by a central switch and 20 video displays for demo purposes. The right part of Figure 23 shows a sample experiment setup which consists of two nodes interconnected by a link with 50ms latency. This setup maps to a physical topology with three nodes, one of which introduces link latency. At the hardware level, all three nodes are connected through the central switch using two automatically configured VLANs. A user only needs to specify the logical topology as shown in the leftmost part of the right figure. The deployment and configuration of the traffic shaping node, as well as the configuration of the VLANs is performed automatically.

At present two different configurations of the Virtual Wall are available.

- **Wall 1**
  - 200 server nodes with the following specifications:
    - Dual CPU, dual or quad core
    - 4-12 GB of RAM
    - 1x160 GB HDD
    - 2-6 network interfaces
  - Central switch: Force 10 networks
    - 336x1 Gb/s ports
    - 8x10 Gb/s ports
    - 1.53 Tb/s backplane
  - 20 computer displays for demo purposes
- **Wall 2**
  - 1 super computational node
  - 1 super graphical node
  - 100 server nodes with the following specifications:
    - 12 cores, 2.4 GHz
    - 24 GB of RAM
    - 2-5 network interfaces
  - Central switch: Force 10 networks
    - 576x1 Gb/s ports
    - 8x10 Gb/s ports
    - 1.67 Tb/s backplane

CDN emulation on the Virtual Wall

To design and evaluate the developed CDN solutions, a three step approach was taken. In a first phase, a wide range of novel network functions (e.g., video-optimized caching algorithms and admission control mechanisms enabling adaptive content delivery) were simulated independently of each other. This allowed doing an elaborate parameter sweep and identifying the best performing solutions for the remainder of the work. In a second phase, a subset of the, in the first phase, proposed solutions were integrated and emulated on the iLab.t Virtual Wall testbed facility. These emulations serve a double goal: on one hand they allow confirming (or contradicting) the simulation-based experiments by evaluating their performance under more realistic conditions. On the other hand, the emulations allow preparing the software prototypes for a large-scale evaluation in a field trial, which is the third and final phase of the experimentation approach.

The emulations were carried out after the majority of the simulation experiments and before a field trial. Therefore, the results obtained through emulation were able to suggest improvements for additional experiments in the simulation phase as well as improve the devised solutions for deployment in the field trial. In this section, we briefly describe the main advantages of the emulations and how they have helped the experimentation in the other phases.

- **Performance evaluation under more realistic scenarios** In the simulation phase, several simplifying assumptions were made that allowed speeding up the time required to perform the simulations (thus effectively allowing to perform more detailed simulations), with the cost of a loss in realism. Once simulation found a good parameter setting for a particular algorithm, this setting

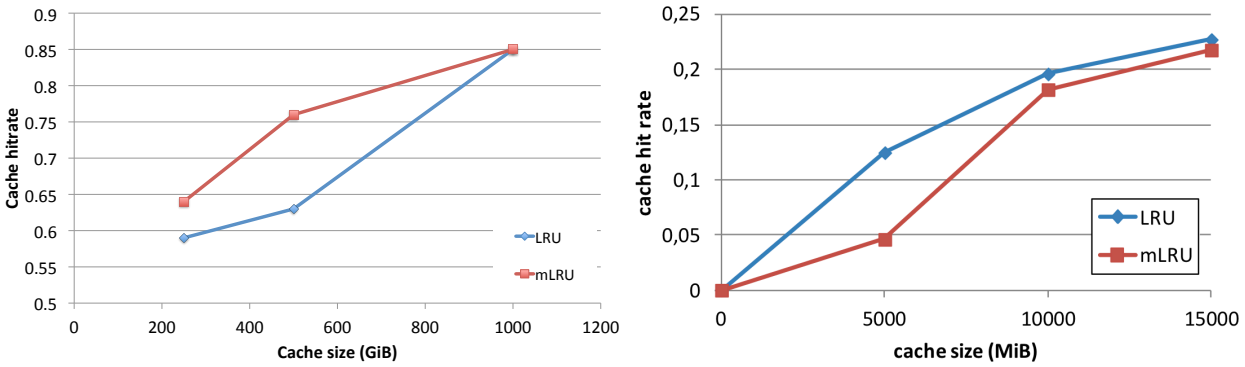

Figure 24: Comparison of cache algorithm performance obtained through simulation (left) and emulation (right)

was evaluated through emulation to confirm the obtained simulation results. In some cases, the emulation results showed significant differences due to the greater accuracy in the modelled environment (e.g., more bursty traffic). This allowed providing suggestions for improvements for the simulation environment and algorithmic contributions.

- **Integrated evaluation of CDN solutions** During simulation, each solution was evaluated independently of the others. In the emulations, the different solutions were integrated and their combined performance was evaluated. In some cases, this provided some important insights. This is illustrated in Figure 24, which shows the performance of the first version of a novel caching algorithm (i.e., modified LRU – mLRU) compared to the traditional LRU approach. While initial simulation-based results showed that mLRU performed best, the emulations indicated the contrary. This was because, in the emulations, it was combined with work on HTTP Adaptive Streaming (HAS). The use of HAS negatively influenced mLRU performance. These conclusions were used to suggest improvements to the design and configuration of mLRU.
- **Prototype testing for field trial deployment** Finally, in order to adequately prepare the field trial, several prototypes were first evaluated through emulation. In these experiments, the focus was on carrying out stress tests to identify possible bottlenecks in the software. As a result, several issues (e.g., memory leaks) were identified, which were resolved before the field trial phase was initiated

EmanicsLab The EmanicsLab environment was initiated in 2007 by the European Network of Excellence for the Management of Internet Technologies and Complex Services (EMANICS). MyPLC, the backend management infrastructure of PlanetLab builds the basis for EmanicsLab. Currently the network consists of 22 nodes at 11 sites across Europe. The infrastructure of EmanicsLab is used for research activities in the area of network and service management. Therefore, it is used for distributed flow collection and analysis, distributed intrusion detection and distributed monitoring and accounting. Further, detailed information about EmanicsLab can be found in D1.2.

Emulation Components Emulation Components describe separately available tools to support emulation experiments. These components do no necessarily belong to a specific emulation environment. They are usable in most varied experimentation scenarios.

MSPSim MSPsim is an instruction-level simulator, that simulates unmodified target firmware, to achieve accurate timing simulation for the MSP430 microcontroller. The simulator is easily extensible with peripheral devices making it possible to simulate various types of MSP430 based sensor nodes. MSPsim can show a graphical representation of the sensor board in an on-screen window. LEDs on the sensor board are displayed using the correct colors. Furthermore, it is intended to be used as a component in a larger sensor network simulation system supporting cross-level simulation. For this reason MSPsim is designed to run multiple instances of the simulator in a single process unlike other MSP430 simulators such as the GDB MSP430 simulator. In addition, MSPSim can simulate hardware peripherals such as sensors, communication ports, LEDs, and sound devices such as a beeper.

Initial investigation is performed into the applicability of a Trust Computing Module (TPM) in RPL based constrained networks. A key exchange mechanism, based on the TPM is developed and preliminary simulative evaluation is performed using Cooja. A model of the TelosB mote is used in MSPSim to obtain performance insights, via Cooja [71].

Security issues related to RPL are also studied using a Contiki implementation simulated in Cooja. The TelosB mote is emulated via MSPSim in Cooja for these studies. MSPSim was useful in obtaining an analysis of the effects of RPL DODAG version attacks [72] and in developing a risk management based strategy for mitigating DODAG inconsistency attacks [73].

tc The tc tool can be used to show and manipulate traffic control settings in linux based systems. Among other functions tc can prioritize packets, apply traffic shaping, limit bandwith and support an Active Queue Management (AQM). Using tc it is possible to decide which (and whether) packets to accept at what rate on the input of an interface and determining which packets to transmit in what order at what rate on the output of an interface. tc allows the user to have granular control over queues and the queuing mechanisms of a networked device. The power to rearranging traffic flows and packets with tc tools is tremendous and can be complicated, but is no substitute for adequate bandwidth.

An application that detects traffic shaping in mobile networks is developed for Android and evaluated in a controlled environment by using the tc package of Linux. This application is then extended to perform protocol based QoS evaluation in mobile networks. The generated data is represented in UML and stored in SQL databases. Linear programming, analytical modeling and statistical analysis are used in order to infer the QoS of a protocol based on the performance metrics obtained from a mobile network [74].

Implementation An Implementation represents a realization of an algorithm as a program, software component, or other computer system. Computer programming is used to translate algorithms into an Implementation. To execute these implementations platforms and operating systems are necessary. They are used to interpret the implementation and execute the algorithm.

Platforms Platforms are in a general sense environments where a piece of software is designed to run within, obey constraints and making use of given facilities. Typical hardware platforms in the area of wireless sensor networks, that are used in FLAMINGO are AVR Raven and TelosB.

AVR Raven The AVR Raven is a development platform consisting of two parts, the Raven board itself and the RZUSB stick, which is used to interface the IEEE 802.15.4 based network with regular IP networks. The RZUSB stick masquerades as a USB Ethernet interface when plugged into a computer.This USB Ethernet interface can then be used to access IEEE 802.15.4 networks. The AVR Raven board hosts two microcontrollers. The ATMega1284P is used as the primary microcontroller on which all the processing and interfacing with the radio, an AT86RF230, occurs. The second microcontroller, an ATMega3290P, is almost exclusively used to control an on-board LCD.

There is good support for the AVR Raven built into the Contiki operating system, thereby allowing for development of software for resource constrained devices by using a stable toolchain. The AVR Raven also provides more memory and flash storage than most other development platforms, thereby making it a good choice for prototyping solutions and studying their suitability for resource constrained environments. The on-board LCD is also a unique feature that allows for easy debugging of code during development. Furthermore, solutions relying on IEEE 802.15.4 and 6LoWPAN for communication can be interfaced with existing IP networks easily by using the RZUSB stick that enumerates as an Ethernet interface on computers.

The performance of standard IP networking management and monitoring protocols is investigated with constrained networks in mind. Implementation of SNMP, NETCONF and DTLS is carried out for the AVR Raven platform, using Contiki OS. Via analytical and traffic modeling, and statistical analysis it is determined that SNMP performs well in constrained environments, as long as security features are disabled. In fact, it is discovered that security and encryption are responsible for the bulk of computing resources occupied in constrained networks by management and monitoring protocols [75].

An implementation of multicast DNS (mDNS) and DNS Based Service Discovery (DNS-SD) is also performed for the AVR Raven platform using Contiki [76]. The suitability of using this combination for service discovery in constrained networks is studied.

TelosB The TelosB mote is a good representative of resource constrained devices. It uses the 16-bit MSP430F1611 microcontroller from the MSP430 family and provides access to 10 KB of RAM and 48 KB of flash memory to store binaries. It is also equipped with an IEEE 802.15.4 radio chip, thereby allowing it to integrate with 6LoWPAN networks as well. The TelosB platform is also a good choice for studying protocols developed for the resource constrained environments because not only is it representative of devices in this environment, but also because it is supported well by most operating systems targeting this area. As such, there is mature support for the TelosB in Contiki, TinyOS and RIOT.

The TelosB platform was used as the development target during studies on security aspects of RPL. Using Contiki firmware developed for the TelosB, the effects of DODAG inconsistency attacks in RPL networks were studied and a solution was designed to mitigate such attacks. The performance of the mitigation approach was also evaluated using TelosB motes emulated in the MSPSim software [73]. A

detailed analysis of the effects of RPL version number attacks was also performed by using specially designed firmware deployed to the TelosB nodes [72]. An implementation of multicast DNS (mDNS) and DNS based service discover (DNS-SD) has also been ported to the TelosB platform. Successful function of this protocol shows that mDNS and DNS-SD can be an effective approach for service and device discovery in resource constrained environments as well [76].

Operating Systems A piece of software that manages computer hardware, software resources and provides common services for additional computer programs is called Operating System. It is an essential component of the system software in a computer system, since all applications usually require an operating system in order to function well [77].

Android The Android mobile operating system developed by Google has seen rapid uptake in smartphones, tablets, smartwatches and portable computing devices. Based on a Linux kernel, Android allows development of software that can take advantage of many important operating system features without the need to obtain root access to the device. The large numbers of Android device users also makes it a good platform to target for developing applications that can be used for taking measurements in mobile networks. This platform is also a good target to implement algorithms that are developed during studies for quality of service (QoS) and quality of experience (QoE).

BonaFide [78], and its extension BonaFide+ [74], were developed as applications for the Android platform. The original BonaFide application was designed to perform traffic shaping detection in mobile networks, by emulating protocol based traffic and comparing its achievable bandwidth against random data flows [78]. This application has been further extended to take the form of BonaFide+, in order to collect QoS data in mobile networks and utilize this to infer protocol based QoE. This QoE is then shown as colors related to MOS values on a map, to allow a user to understand what their experience would be in a specific location based on the type of service they might use [74].

Contiki Contiki is an open source operating system for resource constrained devices that are typically used in the Internet of Things (IoT). The μIPv6 stack implemented by Contiki provides access to IPv6 networking on embedded devices. Using IEEE 802.15.4 radios, Contiki is able to work with the 6LoWPAN adaptation layer as well. Being an operating system targeted to the IoT area, it also contains implementations of various other IETF protocols developed for this area (e.g., RPL, CoAP, etc.). Due to its strong suite of available libraries, Contiki has also been ported to several hardware platforms, including the AVR Raven and TelosB. This makes it a good development platform for studying protocols suitable for the IoT.

The Contiki operating system has been used to develop implementations of network management protocols like SNMP and NETCONF for the resource constrained environment [75] and to study the suitability of these protocols for the IoT. Contiki was also used to study the security aspects of the RPL routing protocol [72]. The security research based on using an implementation of RPL in Contiki has led to the development of an attack mitigation strategy [73] and also a trusted computing approach for key-exchange and encryption of RPL messages [71]. Contiki has also been useful in evaluating the suitability of mDNS and DNS-SD for service and device discovery in resource constrained environments [76].

Protocols Protocols are used for communication purposes among digital systems to exchange data following standardized rules. Well-defined formats for exchanging messages are used to expect a response from a range of possible responses pre-determined for that particular situation.

Routing Protocol for Low power and Lossy Networks (RPL) The IETF ROLL (Routing Over Low power and Lossy networks) working group has proposed a new standard, RPL (Routing Protocol for Low power and Lossy Networks) based on IPv6 and specifically designed for strongly constrained environment. The devices are interconnected according to a topology structure which combines mesh and tree topologies: the Destination Oriented Directed Acyclic Graphs (DODAG). A DODAG graph is built from a root node which is the data sink of the graph. A network is composed of one or several DODAGs grouped into RPL instances which are associated to an objective function. An objective function computes the best path for a set of metrics or constraints. For instance, this function may minimize energy consumption or may simply compute the shortest path. These multiple instances enable the protocol to perform different optimizations, such as quality-of-service ones. A rank is associated to each node and corresponds to its location in the graph with respect to the root. The node rank is always increasing in the downward direction. The RPL packets may be forwarded according to three traffic patterns: point-to-point (P2P), multipoint-to-point (MP2P) and point-to-multipoint (P2MP) [72]. A set of new ICMPv6 control messages is defined to exchange RPL routing information. These routing information are exchanged according to timers based on the trickle algorithm. This algorithm optimizes the transmission frequency of control messages depending on the network state. It consists in increasing the frequency of messages when an inconsistency is detected [73]. This allows faster recovery. On the other hand, the frequency of messages may be reduced when the network shows stability. The RPL protocol defines loop avoidance and detection mechanisms as well as repair processes to ensure path maintenance.

Network Management Protocols Network management refers to a specific kind of protocols to support the operation, administration, maintenance, and provisioning of networked systems. The core tasks of a network management protocol cover command and control practices.

SNMP The performance of standard IP networking management and monitoring protocols is investigated with constrained networks in mind. Implementation of SNMP, NETCONF and DTLS is carried out for the AVR Raven platform, using Contiki OS. Via analytical and traffic modeling, and statistical analysis it is determined that SNMP performs well in constrained environments, as long as security features are disabled [75]. In fact, it is discovered that security and encryption are responsible for the bulk of computing resources occupied in constrained networks by management and monitoring protocols.

DACoRM The DACoRM framework presented in Deliverable D6.1 enables distributed decision-making entities to take network resource utilization reconfiguration decisions in a collaborative manner through a management substrate. As described in section 8, the substrate facilitates the communication between reconfiguration entities. These can, for instance, exchange information related to reconfigurations to perform or request assistance from each other when local reconfigurations are not possible.

In Deliverable D6.1, three management substrate structures were presented, i.e. full-mesh, ring and hybrid topologies, and a brief description of the communication model to use between substrate nodes in each structure was provided. The models of communication are depicted in Figure 17 of Deliverable D6.1. A more detailed description of the communication mechanisms and message sequences is presented in this section.

As can be observed in Figure 17 of Deliverable D6.1, the communication model used in the full-mesh topology follows a star structure centered on the node that initiates the communication [61]. The initiator (represented as a gray disc in the figure) sends a $REQUEST$ message to all its neighbors in the management substrate. Upon receiving a $REQUEST$ message, each neighbor node analyses the content of the message to decide whether it can provide a satisfactory reply to the request. In case it can, the node appends the required information to a $RESPONSE$ message and forwards it back to the initiator. Otherwise, a negative $RESPONSE$ message is returned.

The communication in the ring model relies on a hop-by-hop mechanism [61]. The initiator node sends a $REQUEST$ message to one of its neighboring nodes according to the communication direction followed. The initiator node then enters a listening period where it waits for the message to travel hop-by-hop through the ring until it reaches the initiator again. Upon receiving the $REQUEST$ message, the next hop node analyzes the content of the message, appends the required information as well as its identity to the message and forwards it to the next hop node.

The protocol for the communication between the substrate nodes organized into a hybrid structure supports two modes of communication [64].

- Local Sub-ring Communication: The first mode concerns the communication between nodes located in the same ring. This mode corresponds to the case where the node that initiates the communication needs to exchange some information with (an)other node(s) in the local sub-ring. In that case, the initiator node sends a local request in the form of a $LOCAL\_REQ$ message to one of its neighboring nodes according to the communication direction followed. The message then travels hop-by-hop through the ring until it reaches the initiator node again.
- Remote Sub-ring Communication: The second mode concerns the communication between nodes located in different rings, when for example the initiator node needs to retrieve information from a node located in a remote sub-ring. To do this, the initiator needs to first communicate with its local IE since this node acts as the interface to the other rings. The initiator starts by sending a remote request ($REMOTE\_REQ$) message directly to its IE node, which then forwards it to all the other IE nodes of the MS. Each IE is subsequently responsible for circulating a $LOCAL\_REQ$ message in its local ring. Upon receiving this message back, each IE analyzes its content and creates a remote response ($REMOTE\_RESP$) message that contains information about potential satisfactory replies from its ring. This is sent back to the original requesting IE, which forwards it to the initiator.

The characteristics of the three structures are summarized in Table 8 based on the findings presented in [79]. The total number of nodes in the substrate is noted N and the total number of sub-rings in the hybrid model is noted r.

Table 8: Comparison of the Proposed Management Substrate Topology Models.

Model	Full-mesh	Ring	Hybrid
Level of hierarchy	Flat structure	Flat structure	One level hierarchy
Number of neighbors	N-1	2	min $= 2$ max $= 2 + (r - 1)$
Communication model	Star fashion mechanism	Hop-by-hop mechanism	Star & hop-by-hop mechanisms
Communication delay	Driven by the geographical distance	Driven by the number of nodes	Driven by both factors

3.6 Application Domain Requirements to Enablers

To develop or adapt an enabler to a specific application domain, a couple of limitations emerge. To fasten the process of this development the following paragraphs give an overview on requirements and limitations for enablers in the application domains investigated by FLAMINGO.

Wireless Sensor Networks (WSNs) WSN design is influenced by many factors, e.g. fault tolerance, scalability, production costs, operating environment, sensor network topology, hardware constraints, transmission media, and power consumption. Many of these factors are already addressed by researchers (e.g. [80]), because they serve as a guideline to design a protocol or an algorithm for sensor networks. Within WP2 of FLAMINGO a course module entitled “Wireless Sensor Networks integrated in Internet of Things” addresses those design issues for gaining basic understanding of the challenge. Another module entitled “Secure Communication Applied to Internet-of-Things” addresses security issues that are challenging for constraint devices. Focusing on enablers specific for WSNs there can be identified, beyond the aforementioned factors, the following key factors:

- Hardware constraints (memory, power, computational capacity)
- Maximum transmission unit

Devices building a wireless sensor network should be handsome and long living at the same time. For the handsome reason the size of the devices is small from coin to match box size. This allows only limited space for hardware equipment including sensors, memory, radio unit, and power supply (usually battery powered). As a result enablers are memory, computational capacity, and energy. This means in order to ensure long lifetime the developed algorithms must be energy efficient and require less memory and computational capacity. Currently, in the informal Internet draft [81] constraint devices were specified and organized in classes based on memory resources. These devices can be grouped into the following three classes (where 1 KByte = 1024 Byte, RAM: supported data size, and Flash: supported code size):

- **Class 0** devices are sensor-like nodes, usually pre-configured, and have less than 10 KByte RAM respectively 100 KByte Flash. In general class 0 devices are not able to communicate directly and secure with the Internet.
- **Class 1** devices have about 10 KByte RAM respectively 100 KByte Flash. Compared to class 0 devices they are unable to talk easily to other Internet nodes employing a

full protocol stack (e.g., HTTP, TLS, or security protocols). Class 1 devices are able to provide support for security functions required on large networks, as it is scope of this proposal. Furthermore, those devices can be integrated as fully developed peers into an IP network.

- **Class 2** devices are more memory richer, usually having around 50 KByte RAM and 250 KByte Flash available. With this range they can support mostly same protocol stacks as used on notebooks or servers. But class 2 devices can still benefit from lightweight and energy-efficient protocols in order to increase lifetime.

Devices with capabilities significantly beyond class 2 must be mentioned, since they are also included in the wide device diversity in the IoT. Those devices can usually use existing protocols in an unchanged manner, but may be still constrained by a limited energy supply. Thus, it is required to look for light-weighted solutions for complex algorithms that need computing power, which is also rare on constraint devices. This fact is closely related to memory and energy resource of those devices and will have a direct impact on lifetime of the deployed system.

Another enabler is the supported maximum transmission unit (MTU) that depends on the radio hardware. For example, RF transceiver CC2420 [82] allows a MTU of 127 Byte on PHY-Layer but leaves only 102 Byte MTU available on MAC Layer. Depending on the supporting communication solution (e.g., IPv6), in combination with chosen operating system, the MTU becomes more challenging. Thus, different cryptographically algorithms are not feasible for such devices. For example, using X.509 certificates is challenging and can be used with class 2 devices only. Elliptic Curve Cryptography in contrast is possible for class 1 devices.
Sensor network design is influenced by many factors, e.g. fault tolerance, scalability, production costs, operating environment, sensor network topology, hardware constraints, transmission media, and power consumption. Many of these factors are already addressed by researches [83], because they serve as a guideline to design a protocol or an algorithm for sensor networks. Focusing on enablers specific for wireless sensor networks, hardware constraints and scalability are the major factors to look at, since complex algorithms (see Section 3.5.3) need computing power which is almost rare on constraint devices.

Cloud-Based Services Cloud Computing allows sharing of data-processing tasks, centralized data storage, and online access to computer services or resources based on internet computing in which large groups of remote servers are connected. According to [84] cloud computing can be seen as a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources. In addition, these resources can be rapidly provisioned and released with a minimal effort and interaction by the service provider. Furthermore, a couple of challenges exist in cloud-based services. The first challenge arising in cloud environments is privacy. The increased usage of cloud based services pushed the issue of privacy concerns of cloud computing services to the utmost importance. The reason for that is, a provider of such kind of services is in a position so that with the greater use of cloud computing services a plethora of data is accessible. Another issue concerns legal aspects in cloud computing. The problem of who is in possession of the data is important. The cloud company, as the possessor of data, has certain legal rights. If the cloud company is the custodian of the data, then a different set of rights would apply. Also security aspects are not negligible. Following commonly known service models of cloud based services the subsequent security aspects arise:

- **Software as a Service - SaaS** SaaS can be separated into a diversity of security issues. This paper aims to give a short overview without going into details. Network security issues in this delivery model is often provided via commonly known SSL encryption endpoints depended on the cloud provider, e.g. in the case of Amazon Web Services (AWS) significant protection against traditional network security issues, such as packet sniffing, MITM (Man-In-The-Middle) attacks, IP spoofing, port scanning, is provided. The data location problem, which is mostly tackled if business data is outsourced in the cloud, is addressed in [85]. Data integrity and data segregation is investigated in almost all cases via generally accepted techniques (e.g., XML based APIs, SOAP and REST services with transactions support and data validation techniques). On behalf of data access and authentication/authorization for accessing services, various approaches have been made by the authors of [86], [87], [88].

- **Platform as a Service - PaaS** PaaS tries to give control to consumers that build applications on platforms. So the provider should be aware of the security below the application level (e.g., host and network intrusion prevention). Most applications leverage the Enterprise Service Bus (ESB). In these cases the ESB has to be secured directly, e.g. via Web Service (WS) Security [89].

- **Infrastructure as a Service - IaaS** Customers using IaaS have to be aware of all possible security issues of their systems, except a security lack in the hypervisor of the environment is out of their scope. Since the rapid emergence of virtualization of everything in information society, the control over data regardless of its physical location raised to utmost interest. Several techniques which are applicable to IaaS scenarios are discussed in [90], whereas the question of responsibility (cloud provider or consumer) is not clearly answered and differs between all aforementioned service models. As an example, the Amazon Elastic Compute Cloud (EC2) offers infrastructure as a service that vendor responsibility for security is specified up to the hypervisor. The consumer is in charge for security that is related to the IT system including OS, applications and data.

Content-Aware Networking The content delivery market has mainly been dominated by large Content Delivery Networks (CDNs) such as Akamai and Limelight. However, CDN traffic exerts a lot of pressure on Internet Service Provider (ISP) networks. Recently, ISPs have begun deploying so-called Telco CDNs, which have many advantages, such as reduced ISP network bandwidth utilization and improved Quality of Service (QoS) by bringing content closer to the end-user. Virtualization of storage and networking resources can enable the ISP to simultaneously lease its Telco CDN infrastructure to multiple third parties, opening up new business models and revenue streams. The deployment of such a service raises, however, several technical challenges. In particular, a key issue to address concerns the development of efficient and scalable approaches for the management of the considered caching infrastructure. In this context, research efforts carried out by UCL and iMinds have been focusing on the design and performance evaluation of a novel proactive cache management system for ISP-operated multitenant Telco CDNs. This involves the design of an algorithm that can optimize content placement and server selection across tenants and users, based on predicted content popularity and the geographical distribution of requests.

Besides the reconfiguration logic represented by the actual cache management algorithm, the development of the proposed solution requires the support of several enablers. More specifically, in this scenario, the four following enablers were identified as key support tools and technologies: graph theory (shortest path problem), linear programming, heuristic and discrete event simulation.

As a first step, it is crucial to formally model the considered problem based on well-defined tools. In terms of modeling, two enablers have been used for this scenario. In order to model the caching and network infrastructure, graph theory principles and methods have been considered, one of them being the shortest path problem definition. While the investigated problem can be modeled with several degrees of freedom, the number of degrees of freedom directly affects the size of the problem space and, as a result, the feasibility of the proposed solution. As such, it is necessary to carefully select the main problem parameters. In this scenario, it was shown that optimizing the actual routing decision did not significantly influence the performance. The use of shortest path routing based on the Dijkstra algorithm was therefore considered.

In addition to the infrastructure modeling, a mathematical formulation of the cache management problem is also required to both rigorously define the problem space and derive its complexity. This is essential in order to theoretically estimate the feasibility of the solution. In this scenario, Integer Linear Programming (ILP) was used as the modeling tool.

The objective of the management approach is to optimize the use of network and caching resources. In real deployment, optimizing resources does not necessarily mean achieving the theoretical optimum since this may be associated with practically unfeasible solutions (due to the time-complexity of the algorithm, for example). As a result, heuristic methods are usually considered. The literature is rich of the application of heuristics to various domains and provides references describing the properties of different strategies, which can be used when developing a novel heuristic.

In order to evaluate the performance of the proposed approach in terms of resource utilization, discrete event simulation has been performed. The results of the evaluation provide insights on the performance gain which can be achieved for a wide range of conditions. The results could be further confirmed by testbed and finally real-world deployment.

Software Defined Networks (SDN) Software defined networking was developed to facilitate innovation and enable simple programmatic control of the network data-path. In contrast to conventional network structures, the control plane and the forwarding plane are separated. The communication between the controller and data planes of the forwarding devices is done primarily via the standardized OpenFlow protocol. With this protocol rules for queuing, modifying, forwarding and dropping messages could be applied directly in the forwarding devices. Each SDN enabled switch has a chain of flow tables where a set of flow entries is stored. A flow is described as a set of packets that match given properties and a flow entry adds information about the forwarding behavior of packets which match this patterns. If a packet does not match, it is queued, and an inquiry event is sent to the controller. To handle the queued packet the controller sends a new flow entry to the SDN-enabled switch so that subsequent packets in the same flow will be handled directly by the switch without further requests to the controller. In an OpenFlow network, switches come in two varieties: pure and hybrid. Pure OpenFlow switches have no legacy features or on-board control, and completely rely on a controller for forwarding decisions. Hybrid switches support OpenFlow in addition to traditional operation and protocols. Most commercial switches available today are hybrids.

In fact SDN tries to analyze data flows in order to reprogram the network nearly in real-time to guarantee the shortest delay or the highest bandwidth. This new network management approach offers completely new opportunities in dealing with resources and can help to implement flexible network structures.

Much of the current work on SDN examines or proposes solutions within the context of a single administrative domain which matches quite well SDN's logically centralized control model. However, environments whose administration is inherently decentralized, like the

Internet, call for a control plane that is logically distributed. This will allow participating autonomous systems (ASes) to be controlled independently by their own (logically centralized and possibly physically distributed) controller.

While controller-switch ("southbound") interaction is fairly well defined in protocols there is no standard for interactions between controllers and network services or applications ("northbound"). Additionally, the northbound API should allow applications to apply different policies to the same flow (e.g. forwarding by destination and monitoring by source IP). The work [91] proposes modularization to avoid that rules installed to perform one task do not override other rules. This is accomplished by means of an abstraction layer implemented with a language based on Frenetic. **INRIA-UniBwM-Cloud** investigates recently available SDN-based mechanisms for delivering security in different network scales. While mainly the recent Openflow-based solutions for mitigating such kind of attacks are emphasized, the collaboration also investigates some previous SDN attempts.

3.7 Usage of Enablers and their Requirements in Application Domains

The preceded description of FLAMINGO relevant application domains provides an overview of limitations and requirements in applications domains that researchers should be aware of when implementing new enablers in these domains. To make this information more valuable for FLAMINGO, some questionnaires regarding enablers used in application domains within the collaborations were posed to research WP participants.

Afterward, the analysis of the results should give an overview of the enablers used within an application domain. Table 9 gives an overview of the outcome. This table compacts the proposed taxonomy to a minimum in a sense that no valuable information gets lost, but improving the readability. Remains to mention that this questionnaire constitutes the usage of enablers within FLAMINGO rather than in general. Nevertheless, the results show that computationally expensive enablers, like a machine learning approach, are less suitable in the application domain of wireless sensor networks. Other datasets show that knowledge management is a must when dealing with huge amounts of data or using additional enablers that reason further decisions.

Furthermore, as a first step in a decision process about the most usable enabler, the functionality is at the forefront. Afterward, several decisions are possible. If a complex algorithm is inevitable in an application domain with less computational power possibilities architectural changes might be an alternative, to outsource the complex algorithms. Otherwise, if alternatives or architectural changes are not an option, the decision about the most suitable enabler is strongly influenced by the application domain in which the enabler has to be implemented. Therefore, within the FLAMINGO consortium we work jointly together to optimize already existing approaches to port them into additional application domains. Besides this, newly developed enablers are demanded to be application domain independent and thus usable in various scenarios. In addition, these independent enablers are superbly suitable candidates for the FLAMINGO integrative architecture.

Table 9: Assessment of Enablers used in FLAMINGO Collaborations

		Wireless Sensor Networks				Software Defined Networking			Content Aware Networking			Cloud Based Services			
		INRIA-JUB-RPL	UniBwM-JUB-RPL	UZH-JUB-UniBwM-M2	INRIA-JUB-Distr	iMinds-UPC-NetVirt	INRIA-UniBwM-Cloud	UCL-iMinds-Cache	iMinds-UT-QoS	JUB-UT-Pattern	UCL-UT-Man	UT-UniBwM-IDS	UT-INRIA-Flowoid
Knowledge Management													
Knowledge Representation	Information Models			x	x		x			x	x	x	x
	Relational Databases			x									
	Ontologies						x						
Semantic Reasoning	Rule-based Reasoning				x		x					x	
	Query Languages												
Algorithms													
Mathematical Optimization	Mathematical Programming			x		x		x	x				
	Duality			x		x		x	x				
Machine Learning	Reinforcement Learning					x							
	Artificial Neuronal Networks					x							
	Distributed Algorithms												
	Logical Topology Construction Algorithms										x		
Risk Management		x											
Experimentation													
Simulation	Discrete Event Simulation	x	x		x	x		x	x				
Emulation	Network Testbed	x		x	x		x			x			
	Emulation Components	x	x	x	x		x						
Implementation	Platforms	x	x	x	x								x
	Operating Systems	x	x	x	x		x	x	x		x	x	x
	Protocols	x	x										x

4 PhD Collaborations

The integration of PhD students is one of the S.M.A.R.T. objectives within this WP. Section 4.1 gives an overview of FLAMINGO collaborations that are ongoing, have been ended or are in the process of starting during this year. Furthermore, collaborations envisioned for the next years of FLAMINGO are shown.

In the overall FLAMINGO approach, monitoring (WP5) forms the basis for any automated configuration and repair action (WP6), while in parallel both activities are conducted within the boundaries of economic, legal and regulative constraints (WP7). Y2 leads to several collaborations between these three WPs. As done in Y1, we focus in reporting the collaborations between WP5 and WP6, given the strong interdependence between monitoring and automated configuration and repair.

A detailed description about the currently ongoing collaborations and the recently completed ones can be found in section 4.2. All fully integrated PhD students are listed in D8.2, including their co-supervisors and affiliation.

4.1 PhD Student Collaborations

The integration of PhDs into FLAMINGO allows valuable and fruitful joint research in the area of network and service management. The bottom-up approach started last year was continued to integrate experienced researchers as well as new researchers not necessarily paid by FLAMINGO. The maintenance of the architecture described in D6.1.(see Section 3.3) leads to several new collaborations. Table 11 summarizes the collaborations, the affiliations involved and their respective status. Each collaboration can have one of the following status: **ONGOING, ENDED, STARTED, PLANNED**. **ENDED** applies to collaborations started in Y1 of FLAMINGO and ended in Y2 because the research goals have been reached they have branched into new collaborations. A collaboration is called **ONGOING** if started during Y1 or Y2 and progress is already reported (e.g. measurement results, planned papers,...). **STARTING** collaborations are in the process of defining their topic, research interests and goal of the collaboration, and drafting a plan how to possibly reach their goal. The last type of collaborations with the status **PLANNED** have defined mutual interest in working jointly together, but didn't define a concrete topic.

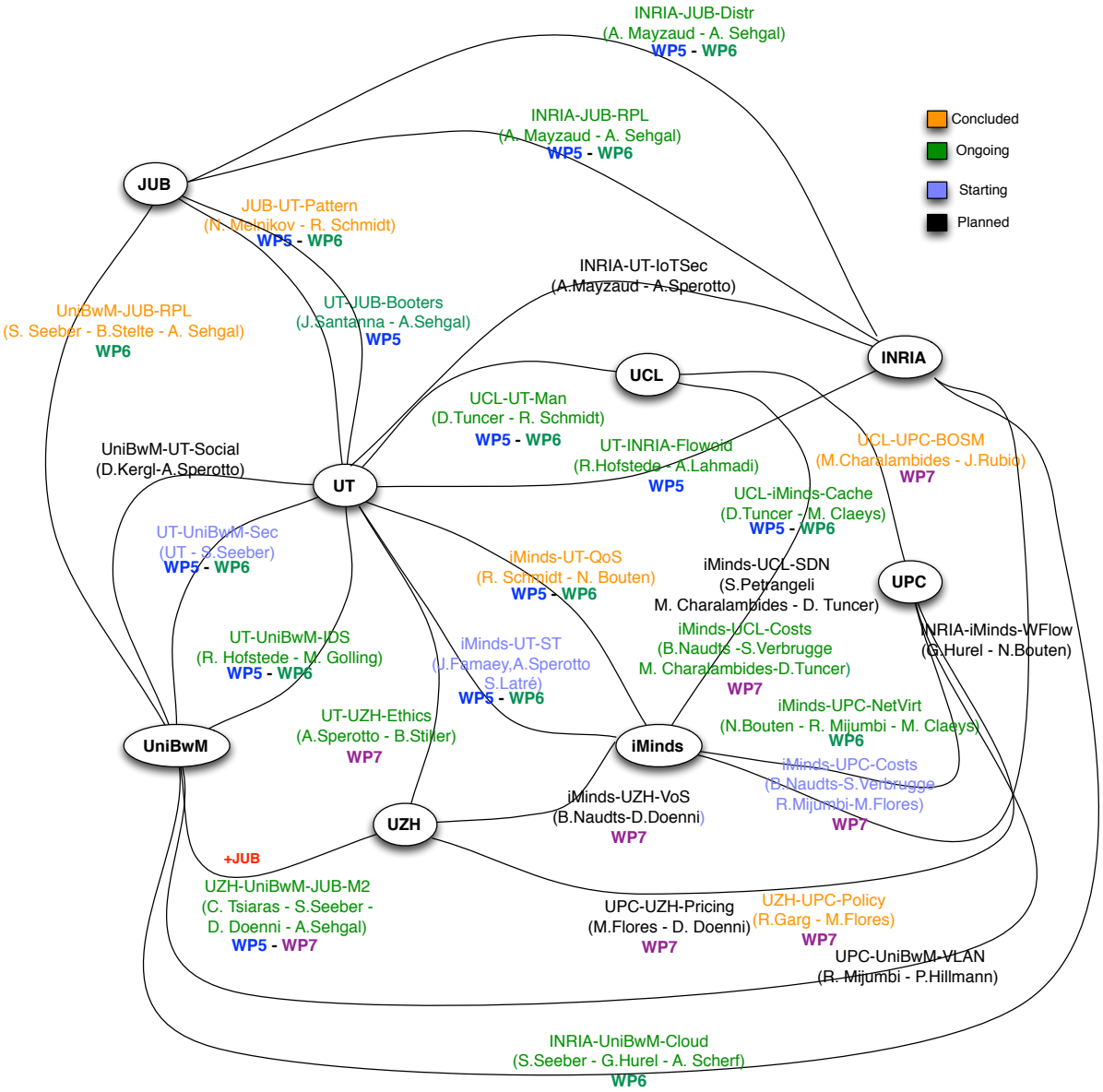

Figure 25: Overview of PhD collaborations

4.2 Description of the collaborations

This section presents the currently ongoing and recently ended collaborations between WP5 and WP6. Each collaboration description roughly follows the same structure. At first the topic of each collaboration is explained. Subsequently the progress and achievements in Y2 are highlighted. Depending on the status of a collaboration further steps are described. In the end each collaboration highlights the contribution to each WP.

4.2.1 Linking Network Usage Patterns to Traffic Gaussianity Fit (JUB-UT-Pattern)

The Gaussianity of traffic aggregates is a desirable characteristic in the domain of network traffic modeling due to the wide adoption of Gaussian models. Past works have extensively researched the Gaussian property of traffic aggregates, its advantages for proposing traffic models and how

Table 10: Overview of the FLAMINGO Collaborations, as in Figure 25

Acronym	Researchers	WPs	Status
INRIA-JUB-RPL	A. Mayzaud - A. Sehgal	WP5, WP6	Ongoing
INRIA-JUB-Distr	A. Mayzaud - A. Sehgal	WP5, WP6	Ongoing
INRIA-UniBwM-Cloud	S.Seeber - G.Hurel A. Scherf	WP6	Ongoing
UCL-iMinds-Cache	D.Tuncer - M. Claeys	WP5, WP6	Ongoing
UCL-UT-MAN	D.Tuncer - R.Schmidt	WP5, WP6	Ongoing
UT-UZH-Ethics	A.Sperotto - B.Stiller	WP7	Ongoing
UT-UniBwM-IDS	R. Hofstede - M. Golling	WP5, WP6	Ongoing
UT-INRIA- Flowoid	R. Hofstede - A. Lahmadi	WP5	Ongonig
UZH-UniBwM-JUB-M2	C. Tsiaras - S.Seeber D. Doenni - A.Sehgal	WP5, WP7	Ongoing
iMinds-UPC-NetVirt	N.Bouten - R. Mijumbi M. Claeys	WP6	Ongoing
iMinds-UCL-Costs	B.Naudts -S.Verbrugge M. Charalambides-D.Tuncer	WP7	Ongoing
JUB-UT-Pattern	N. Melnikov - R. Schmidt	WP5, WP6	Ended
UniBwM-JUB-RPL	S. Seeber - B.Stelte - A. Sehgal	WP6	Ended
UCL-UPC-BOSM	M.Charalambides - J.Rubio	WP7	Ended
iMinds-UT-QoS	R. Schmidt - N. Bouten	WP5, WP6	Ended
UZH-UPC-Policy	R.Garg - M.Flores	WP7	Ended
iMinds-UPC-Costs	B.Naudts - S.Verbrugge R.Mijumbi - M.Flores	WP7	Starting
iMinds-UT-ST	J.Famaey - A.Sperotto S.Latré	WP5, WP6	Starting
UT-UniBwM-Sec	TBD - S.Seeber	WP5, WP6	Starting
UT-JUB-Booters	J.Santanna - A.Sehgal	WP5	Starting
iMinds-UZH-VoS	B.Naudts-D.Donni	WP7	Planned
INRIA-iMinds-WFlow	G.Hurel - N.Bouten	WP5, WP6	Planned
INRIA-UT-IoTSec	A.Mayzaud - A.Sperotto	WP5, WP6	Planned
UPC-UZH-Pricing	M.Flores - D. Dönni	WP7	Planned
UniBwM-UT-Social	D.Kergl-A.Sperotto	WP5	Planned
UPC-UniBwM-VLAN	R. Mijumbi - P.Hillmann	WP6	Planned
iMinds-UCL-SDN	S.Petrangeli - M. Charalambides D. Tuncer	WP5, WP6	Planned

this can be disturbed by traffic bursts. Past works, however, rely on data measured quite a long ago, dating back to 90's and first half of 2000's. Therefore, this motivated the collaboration to further study the impact of network usage patterns on Gaussianity using more recent measurement datasets, i.e., linking traffic Gaussianity (or lack thereof) to network usage patterns. This knowledge is valuable to understand the limitations of current traffic models in presence of certain network traffic. Therefore, the aim of this collaboration was to find out the potential connections between traffic bursts and poor Gaussianity, and also to point out what sort of host and/or application are behind such disruptions.

The results show that Gaussianity fit can be directly linked to presence or absence of extreme traffic bursts. (In the study, an extreme traffic burst is defined, at any timescale, by a traffic peak in which

Table 11: PhD students involved in Ongoing WP/WP6 collaborations

Name	Affiliation	Collaborations	Acronym
Anthéa Mayzaud	INRIA	JUB	INRIA-JUB-RPL INRIA-JUB-Distr
Gaetan Hurel	INRIA	UniBwM	INRIA-UniBwM-Cloud
Rick Hofstede	UT	UniBwM, INRIA	UT-INRIA-Flowid UT-UniBwM-IDS
Ricardo Schmidt	UT	UCL	UCL-UT-Man
Jair Santanna	UT	JUB	UT-JUB-Booters
Mario Golling	UniBwM	UT	UT-UniBwM-IDS
Sebastian Seeber	UniBwM	UZH, JUB, INRIA	UZH-UniBwM-JUB-M2 INRIA-UniBwM-Cloud
Achim Scherf	UniBwM	INRIA	INRIA-UniBwM-Cloud
Rashid Mijumbi	UPC	iMinds	iMinds-UPC-NetVirt
Anuj Sehgal	JUB	INRIA, UT, UniBwM	INRIA-JUB-RPL INRIA-JUB-Distr UT-JUB-Booters UZH-UniBwM-JUB-M2
Christos Tsiaras	UZH	UniBwM, JUB	UZH-UniBwM-JUB-M2
Daniel Dönni	UZH	UniBwM, JUB	UZH-UniBwM-JUB-M2
Niels Bouten	iMinds	UPC	iMinds-UPC-NetVirt
Maxim Claeys	iMinds	UCL	iMinds-UPC-NetVirt

the traffic rate crosses a threshold given by the aggregate average rate plus three times the traffic rate standard deviation.) In addition, the results show that even in a more homogeneous network (i.e., hosts with similar access rates) one can identify extreme traffic bursts that might ultimately compromise Gaussianity fit. These bursts are typically caused by single, or very small group of hosts, and also by a small handful of applications (Applications studied by the collaboration were classified at the port level, e.g., HTTP in port 80 and HTTPS in port 443).

The collaboration contributes to WP5 since it is based on the collection of packet-level traffic traces from many different locations. This collaboration also contributes to WP6 since its results can be applied to management operations that directly or indirectly rely on Gaussian models.

4.2.2 Energy-aware Traffic Management (UCL-UT-Man)

Driven by the rising cost of energy and increasing environmental consciousness, some recent research efforts have been investigating the development of energy-saving resource optimization techniques for green operations. These aim at reducing the energy consumption of Internet Service Provider networks (and therefore operational expenditure) by adapting the configuration of network elements (i.e. line cards, interfaces or routers) according to traffic dynamics. While some efforts (e.g. [92]) have considered time-driven approaches by which network configuration profiles can be computed in a static fashion and applied according to the time period, others (e.g. [93]) have developed more adaptive approaches by which individual network devices can decide to enter sleep mode based on their awareness of current network conditions. Switching off entire links/routers, however, can adversely affect the network performance and subsequently cause degradation of service quality. In particular, this can disconnect the network topology, which, in

addition to possible packet losses under traffic variations, is not suitable for online reconfigurations given that the process of computing the routing configuration is not trivial.

In contrast to the majority of previous work that suggests switching off entire links, the adaptive energy-aware resource management approach developed [3] aims at reducing the energy consumption of core IP networks by applying the switching off strategy at the line card level. Exploiting the fact that many links in core networks are bundles of multiple physical cables, the proposed approach adapts the link capacity at run time by controlling the allocation of traffic in the network so that the load is distributed over a subset of router line cards, while unused ones enter sleep mode.

To achieve the energy-saving objective, the proposed approach relies on the DACoRM in-network management framework described in Deliverable D6.1 and presented in [61]. According to DACoRM, network edge nodes are embedded with a level of intelligence that enable them to realize self-management functionality and are organized into a management substrate through which they coordinate reconfiguration actions to control the traffic distribution in the network. More specifically, based on the path diversity provided by multi-topology routing (MTR), the network edge nodes adapt the volume of traffic routed over the multiple paths between any source-destination pair by adjusting the configuration of traffic splitting ratios. The main principle of the approach is to iteratively move the traffic load from less utilized line cards to more utilized ones that can accommodate this load and thus potentially fill-up their remaining capacity.

The performance of the energy management approach has been evaluated using real topologies and traffic traces, and the results show that substantial energy gain can be achieved without significantly compromising the balance of the network in terms of load [3]. Further evaluations have also been performed to investigate the influence of network topology characteristics (e.g. the number of nodes and degree of router connectivity) and MTR planes configuration on the performance of the adaptive scheme. In addition, the collaboration has investigated the influence of the routing protocol (e.g. multi-topology routing, Open Shortest Path First (OSPF), Equal-Cost Multi-Path (ECMP)) on network energy consumption.

The research issues related to the development of the resource management approach mainly fall within the scope of WP6.

4.2.3 Intrusion Detection Systems (UT-UniBwM-IDS)

This joint research activity is a collaboration between UT and UniBwM. Intrusion detection is nowadays commonly performed in an automated fashion by IDSes [94]. Several classifications for IDSs are common. One of these classifications focuses on the kind of data that is used for performing intrusion detection. The first class of IDSs mainly uses packet headers (flows) for intrusion detection. While these flow-based IDSs have a high-performance and are usually little privacy-intrusive, they are typically affected by a high number of undetected attacks (false negatives; see Figure 26).

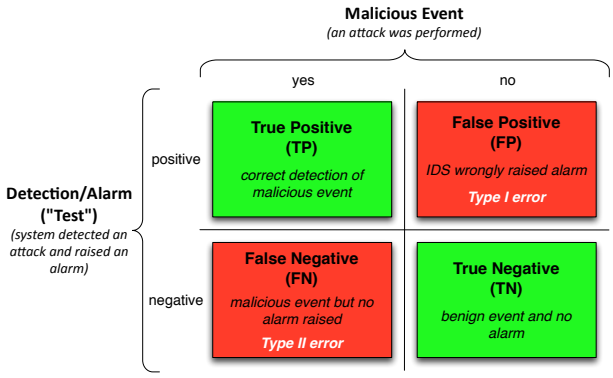

Figure 26: Categories of Alarms (”Confusion Matrix”).

In contrast to flow-based IDSs, payload-based IDSs are capable of performing extensive layer-7-detection (and, therefore, have a lower false negative rate), but at the expense of a much higher system requirements as well as a violation of privacy [95].

Given these observations, performing intrusion detection in high-speed networks is a challenging task. While many payload-based IDSs are working well at the backend of service provider networks, the backbone is often characterized by communication links with high-speed connections and thus requires well equipped IDS in order to be capable of handling 100 Gbps or more, for example [96]. Within this collaboration, it is planned to create a framework for distributed intrusion detection in high-speed networks by combining especially flow-based and payload-based intrusion detection. As already stated, in addition to monetary aspects, legal issues in general and privacy issues in particular are also important reasons why payload-based IDS are rarely deployed in high-speed networks today [95].

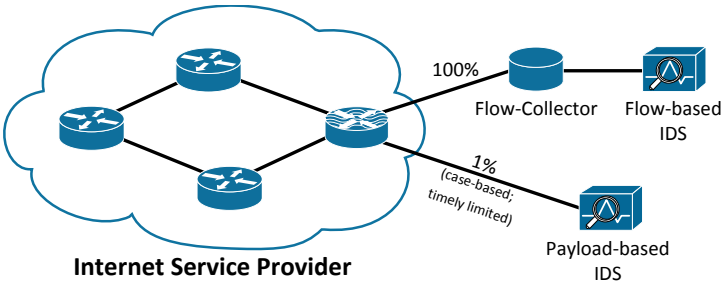

Figure 27: Simplified scenario for UT-UniBwM-IDS.

In order to overcome these disadvantages, this collaboration makes use of both approaches (flow-based and payload-based intrusion detection) in a multi-layered approach. As depicted in Figure 27, the approach is centered around the ideas that (i) the first detection layer comprises flow-based intrusion detection, which performs detection based on the entire packet stream (100%) and that (ii) depending on the result of the flow-based detection, the payload-based IDS is activated for a certain period of time to investigate the anomaly of the flow-based IDS in more detail (1%) – in order to verify or falsify the result of the flow-based IDS. As network attacks can last shortly and a switch has to be made from flow-based to packet-based detection, detection has to be performed in real-time.

All the ideas presented above are summarized in our architecture for multi-layered intrusion detection, shown in Figure 28, and published in [97]. The architecture features three main data streams:

A Flow meta-data that can be retrieved directly from the router's Command-Line Interface (CLI).

B Flow data, exported by means of Cisco's NetFlow [20] or the recent IETF standardization effort IPFIX.

C Full packet streams, potentially pre-filtered by the router upon instruction by the *Manager*.

Key characteristic of the *Real-Time IDS* is that it constantly analyses the full traffic stream, without any form of sampling or filtering. In a previous work, we have shown that a similar approach is able to mitigate DDoS attacks in near real-time [98]. Upon detection of such an attack, the *Real-Time IDS* can reconfigure the router to drop the attack traffic, to make sure that neither the network itself, nor the monitoring infrastructure is overloaded. In addition, the *Manager* is informed about the attack by means of a standardized message exchange format, such as the Intrusion Detection Message Exchange Format (IDMEF).

Besides the *Real-Time IDS*, also the *Flow-Based IDS* is constantly monitoring its input data stream. Given that flow export technologies, such as NetFlow and IPFIX, aggregate packets into flows, such an IDS is usually capable of monitoring the aggregated traffic using commodity hardware. An example of a flow-based IDS is SSHCure,[18] which detects SSH dictionary attacks and reports

[18] `http://github.com/sshcure/sshcure/`

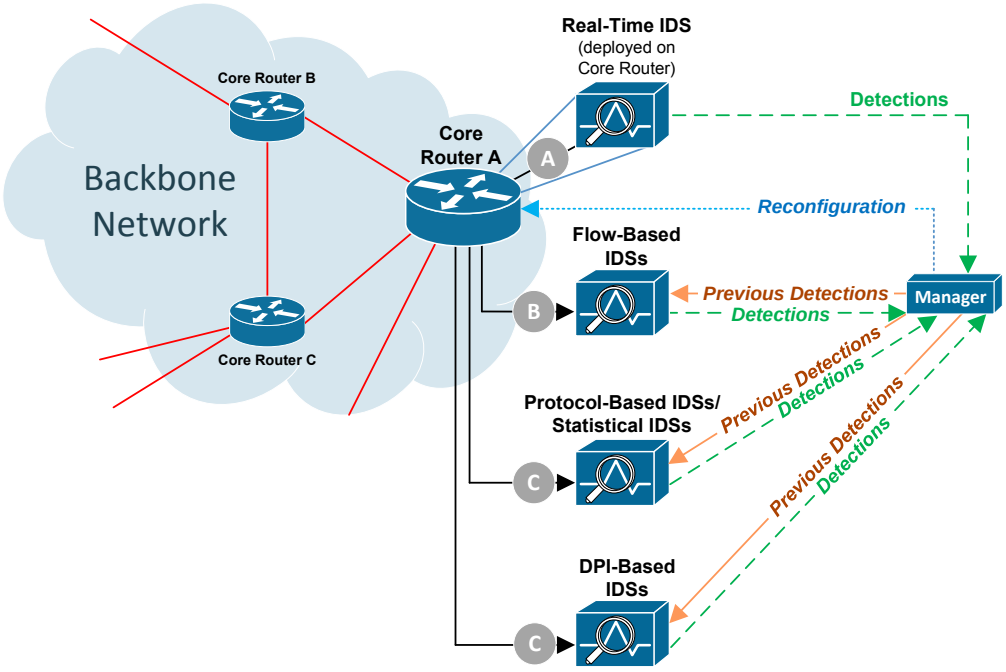

Figure 28: Architecture for multi-layered intrusion detection [97].

whether a host has been compromised [99]. The *Flow-Based IDS* may be informed by the *Manager* about previous detections, and reports its own detections to the *Manager* again. Although not supported by current IDSs, the main idea of forwarding previous detection results to IDSs is to give as much information as possible and so to make the process of intrusion detection as reliable as possible.

In situations where the *Manager* decides to initiate a more extensive analysis of an attack, the *Protocol-Based IDS* or *DPI-based IDS* can be activated and instructed. The *Manager* decides which IDS is most suitable for a particular attack. Before activating the other IDSs, the *Manager* has to reconfigure the router to pre-filter the traffic stream to only include the attack traffic. Analogously to the *Flow-Based IDS*, these IDSs report their detections to the *Manager*. If an attack has been detected, the router is instructed to drop the attack traffic. If an attack could not be confirmed, the *Manager* will not dispatch any investigation about that particular traffic to the various IDSs anymore.

As to the state of the multi-layered architecture, we can report that a prototype has been developed that is currently under validation. We are currently in the process of collecting the first datasets, which will be used for testing the operation of the prototype and, in a later stage, for performing a validation of the accuracy.

The collaborative work contributes mainly to WP5: Network and Service Monitoring by performing multi-layered IDS. Especially objective 8 - novel solutions for IDS is addressed with this collaboration. As the long-term goal of this collaboration is also to link different managers with each other, this addresses objective 3 ”to develop a generic distributed flow monitoring architecture”. Regarding this objective, in [100] an Evaluation of State of the Art IDS-Message Exchange Protocols was already performed.

For WP6, [100] also contributed to objective 3 ”to develop an inventory of approaches for automated configuration and repair”. By using cloud-based solutions (as described in [101, 102]),

requirements for cloud-based services have been investigated and architectural approaches specific for this application domain have already been partially developed (also addressing objective 9 “to propose and study automated configuration and repair in the context of the management of clouds (especially interclouds)”) As it is planned to develop an ”automated” architecture that can be used in different administrative boundaries, objective 5 ”to develop new architectures for automated configuration and repair approaches across administrative boundaries” is addressed as well.

4.2.4 Towards A Trust Comupting Architecture for RPL in Cyber Physical Systems (UniBwM-JUB-RPL)

Cyber Physical Systems (CPSs) are widely expected to be formed of networked resource constrained devices. In order to suit the constraints of such networks, the Internet Engineering Task Force (IETF) developed the Routing Protocol for Low power and Lossy Networks (RPL) and Low-power and Lossy Networks (LLNs). Security in CPSs is important for maintaining the integrity and privacy of data, while also improving network resiliency to attacks. Even though RPL provides support for integrity and confidentiality of messages, details regarding key management and signatures are not covered. Since complexity and size is a core concern in LLNs, off-loading the security features to a Trusted Platform Module (TPM) makes it possible to include sophisticated security provisions in an RPL implementation.

This collaboration developed mechanisms to use the security mechanisms of a TPM in order to secure the communication in an RPL network. The design of a trust establishment and key exchange mechanism around the implied trust of a TPM to provide keys for secure RPL nodes, was a main task of this research. With this approach, the usage of a TPM on Resource Constrained Devices reduces the processing load on the main processor. The goal of this examination is the prevention of the dissemination of misleading routing information, which can affect the availability of the whole network.

The collaboration fits within WP6, since it develops a mechanism, specifically, targeted to RPL networks, to secure the communication. This approach is applicable in RPL networks which are used in wireless sensor networks.

4.2.5 QoE-Driven In-Network Optimization for Adaptive Video Streaming Based on Packet Sampling Measurements (iMinds-UT-QoS)

HTTP Adaptive Streaming (HAS) services allow the quality of streaming video to be automatically adapted by the client application in face of network and device dynamics. A major obstacle for deploying HAS in managed networks, is the purely client-driven design of current HAS approaches, which leads to excessive quality oscillations, globally suboptimal behavior, and the inability to enforce management policies.

These challenges can be tackled by steering the quality selection from within the network. iMinds already deployed a distributed in-network management heuristic, which is able to reduce the number of switches with a factor 5 and increase the quality up to 30%. One of the shortcomings of this deployment, however, was the assumption of a static bandwidth for each link and the absence of cross-traffic. To overcome this issue, this collaboration uses sampled packet measurements to measure and predict the per-link throughput, which is available for HAS traffic. Using these predictions the in-network video quality adaptation can divide the resources among the different HAS clients based on a provider’s policy.

This work contributes to WP5 because real world traffic traces will be collected and used to validate the link dimensioning approach.

This work contributes to WP6 in the following. The goal of this collaboration is to develop a distributed algorithm/heuristic, which is able to divide the resource among the various HAS clients subject to a provider's policy. Using the measurements provided by WP5, each agent is able to perform a local optimization based on the throughput predictions. The different distributed agents share this network information and their local decisions with each other to be able to automatically react to changes in the network environment. Using the measurements and predictions on the current and future cross-traffic, the agents can make estimations on the residual bandwidth that can be shared among the different HAS sessions. These local estimations and the shared network information serve as input to the algorithm which limits the quality of the HAS sessions crossing the managed resources. This leads to a more stable quality selection at the client, since oscillations due to changed network environments are avoided.

4.2.6 Security of RPL Networks (INRIA-JUB-RPL)

The collaboration Security of RPL Networks between INRIA and JUB led to several sub-collaborations due to the existence of various types of attacks in RPL networks. In the following a summary of each individual topic is provided.

Mitigating DODAG inconsistency attacks is the first sub-collaboration. Fundamentally, RPL utilizes DODAGs, a directed graph like structure, to organize the routing topology in a network. The methodology used to detect and repair possible inconsistencies in DODAG can be manipulated by malicious nodes to harm the network.

The aim of this sub-collaboration is to develop methodologies to mitigate such attacks. An approach that dynamically adapts parameters of an adaptive threshold has been developed.

The second sub-collaboration is called **RPL Version number attacks**. Version numbers are used by the RPL DODAG root in order to keep track of the latest version of the topology. If a node detects that it is part of an older version, it is required to join the new version. However, due to the lack of security mechanisms, this method could be utilized by malicious nodes to attack the topology, and possibly even hijack nodes to join its own network.

The aim of this study is to evaluate the effectiveness of attacks based on manipulating version numbers, and also study the already proposed solutions. Based on the study a new approach that overcomes existing shortcomings would be developed.

Mitigating Black-hole and Sink-hole attacks is the last sub-collaboration. The goal is to develop a mechanism that mitigates black-hole and sink-hole attacks in RPL networks, by establishing inferred trust between neighbors.

This work is currently in the development phase, with network metrics contributing towards the trust metric already identified. An implementation of the preliminary approach is currently pending.

Monitoring of the RPL network and the identification of possible attacks in an RPL network contributes to WP5. The automated repair and mitigation of detected attacks in RPL networks contributes to WP6, as well as the application of developed approaches to wireless sensor networks, via the Internet of Things application area.

4.2.7 Distributed Monitoring Architecture for the Internet of Things (INRIA-JUB-Distr)

A generic distributed monitoring architecture is being designed for application in the Internet of Things (IoT) area. The goal of the architecture is to be able to monitor events and network flows passively without having any impact upon the resource constrained nodes that participate in such a network. This collaboration has been recently defined and is therefore in a preliminary stage.

The monitoring architecture will be applied to network and service monitoring in the IoT, and also towards anomaly detection and correction (including security aspects).

Developing a distributed monitoring architecture of the IoT infrastructure can be seen as a part of WP5. An automated repair of detected anomalies in the IoT and the application of developed approaches to wireless sensor networks, via the Internet of Things application area contributes mainly to WP6

4.2.8 Mobile Measurements (UZH-JUB-UniBwM-M2)

Quality-of-service (QoS) metrics have been traditionally used to evaluate the perceived quality of services delivered by mobile network operators. However, this metric is not suitable for evaluating the experience of an end-user. Experience of a user is quantified based upon activities such as speed of web page loading, quality of video streaming, or voice quality of Internet-telephony. Due to the temporal and geographical nature of mobile networks, the perceived experience of a user may change based on location and time. Mobile operators may prioritize certain services over others, leading to a service type dependent quality of experience (QoE).

This collaboration aims to develop mechanisms for evaluating activity and protocol based QoE. The aim is to have a method for obtaining a service specific QoE based on active measurements performed in mobile networks. Obtained QoE values will be mapped to mean opinion scores (MOS) and presented on a global map. This not only aids operators in identifying their users' QoE in specific locations, but can also assist users in identifying areas where thy might have coverage issues.
An approach to obtain QoE MOS values based on statistics (bandwidth, latency, signal strength, etc.) has been outlined [74].

This approach has led to the development of the BonaFide+ [103] application. With the initial development of the measurement application complete, measurement data is now being collected and approaches being defined to further refine the QoE calculation approach. Once enough data is collected, the results were analyzed and published. In the meantime, a larger measurement endpoint infrastructure is also being sought after. Assistance of the M-Lab project [104] is likely to be sought, while current endpoints are deployed on EmanicsLab [105] already. A website with information regarding the application, access to source code and collected data is currently being worked upon as well [106].

The aspect of an active collection of network metrics for service quality monitoring and the identification of possible traffic shaping in the network is part of WP5.

4.2.9 Cache Management (UCL-iMinds-Cache)

The rise of Internet-based over the top multimedia services has put immense strain on the resources of Internet Service Provider (ISP) networks. This has resulted in increasing operating

costs but also in decreasing revenues of traditional traffic forwarding services. As a consequence, ISPs have started exploring alternative business models and service offerings. This has lead to the deployment of Telco Content Delivery Networks (CDNs), which allow content to be cached deep inside the ISP network. For the operators, Telco CDNs reduce bandwidth demand on their backbone infrastructure and open up new business models. For the end-users, Quality of Service (QoS) can be significantly improved, as content is stored nearby and the ISP has full control over the network infrastructure. Furthermore, the advent of cloud computing and SDN technologies enable ISPs to virtualize their networks and by extension, their Telco CDN infrastructures. This opens up new business models and allows ISPs to dynamically offer virtual storage and content delivery services at the edge of the network, redeeming traditional CDN and content providers from installing additional hardware.

In this collaboration, iMinds and UCL have been investigating a cache management approach for multi-tenant caching infrastructures. The multi-tenant content placement and server selection problems were formally modelled by means of an Integer Linear Program (ILP) formulation. Solving this model determines where to store which content item of each tenant (content placement) and from which location to satisfy each request (server selection), with the objective of minimizing bandwidth consumption in the network while maximizing the cache hit ratio. The model can easily be adapted to consider other optimization criteria, such as cache hit ratio maximization or delivery delay minimization. All of these decisions are taken considering the content popularity and its geographical distribution. To compute a new configuration, the model requires predicted values concerning request distribution, for which a prediction strategy has been employed.

In the considered scenario, a large-scale ISP operates a limited capacity CDN service by deploying caching points within its network. Each network node is associated with caching capabilities, which enable a set of content items to be stored locally. The local caches can be external storage modules attached to routers or, with the advent of flash drive technology, integrated within routers. The ISP leases the caching space in its network to multiple content providers. Each content provider specifies the amount of caching capacity it wishes to lease for storing part of its content catalog.

The proposed approach has been thoroughly evaluated in a simulated environment on a Video-on-Demand (VoD) use case, for which a request trace of the VoD service of a leading European telecom operator was used. Its performance has been compared to a reactive caching approach, using the Least Recently Used (LRU) replacement strategy. The evaluations show that with the proposed ILP approach, the average total bandwidth usage inside the ISP network can be reduced by 12%-17%, depending on the amount of leased caching capacity. Furthermore, the average peak link usage can be improved by around 73%. Finally, evaluations showed that using the proactive ILP approach can lead to more balanced link load distribution inside the ISP network. All of these results are influence by the accuracy of the prediction of content requests, which is the current focus of the collaboration.

Aspects related to the analysis of the VoD traces fall within the scope of WP5. This research activity mainly falls within the scope of WP6. The development of proactive mechanisms to efficiently manage the utilization of network resources concerns WP6.

4.2.10 Management of Virtualized Networks (iMinds-UPC-NetVirt)

The management of virtualized networks is challenging task. Several technical challenges in terms of instantiation, operation, and management of virtual networks are either untouched or require further attention. Therefore, the collaboration is divided into sub-collaborations to focus on specific research topics without losing the overall management goals existent in virtualized networks. In the following each sub-collaboration is explained:

Machine Learning-based Virtual Network Resource Management Most current approaches to resource management in network virtualization allocate a fixed amount of resources to the virtual nodes and links for their entire lifetime irrespective of actual utilization. As Internet traffic is not static, this could lead to an inefficient utilisation of overall network resources, especially if a substrate network rejects requests to embed new VNs while reserving the resources for VNs that are lightly loaded.

In this collaboration, instead of allocating a fixed amount of resources to a given VN throughout its lifetime, we dynamically and opportunistically allocate resources to virtual nodes and links depending on the perceived needs. We use a demand-driven dynamic approach that allocates resources to virtual nodes and links using machine learning techniques. Therefore, after the initial virtual network embedding step, resources allocated to each virtual node and link are monitored and adjusted to reflect both actual resource need by the virtual network, and resource availability in the substrate network. We represent each substrate node or link as an agent. These agents are tasked to monitor the resource utilisation of all mapped virtual nodes and links, and comparing this with the available substrate resources, re-allocations are performed. The agents then monitor the network to determine its performance, for example through parameters such as packet delay and drop ratio, and based on these statistics, the agents use machine learning techniques to make better actions for future resource allocations. We have been able to apply three machine learning techniques to this problem: Reinforcement Learning [107], [108],[109], Artificial Neural Networks [110], [109] and Neuro-fuzzy Algorithms [111]

SDN-based Management of Virtual Network Resources In this collaboration, we use the SDN control plane to efficiently manage resources in virtualized networks by dynamically adjusting the virtual network (VN) to substrate network (SN) mappings based on network status. We extend an SDN controller to monitor the resource utilisation of VNs, as well as the average loading of SN links and switches, and use this information to proactively add or remove flow rules from the switches. The results are presented in [6].

Chaining of Service Functions in Network Function Virtualisation In this part of the collaboration the different network functions involved in multimedia services are identified and how they are chained to support the delivery of multimedia services. The different opportunities for virtualizing multimedia network functions will be determined and how this impacts the service function chains. We will determine which physical resource deployments are optimal for such virtual multimedia services (i.e. locations of datacenters, peering points). Furthermore, the optimal network function mappings will be determined based on the alternative service function chains and the resource availability at the substrate network. This work is still in progress and has not yet resulted into a submitted/accepted paper.

The overall research activity of this collaboration falls within the scope of WP6, since automated and dynamic reconfiguration of virtualized networks based on utilization information is a main goal.

4.2.11 Cloud Security (INRIA-UniBwM-Cloud)

The aim of the joint research activity “Cloud Security” between INRIA and UniBwM is to investigate recently available SDN-based mechanisms for delivering security in different network scales, ranging from home networks to datacenters. In the first step the scope of the study is focused on several well-known network attacks such as denial-of-service, information gathering and malware propagation.

While mainly the recent Openflow-based solutions for mitigating such kind of attacks are emphasized, the collaboration also explores some previous SDN attempts - such as ForCES and Active Networks - in this light. Furthermore, the proposed approaches will be compared with the ones that are found nowadays in traditional networks (i.e. non-SDN). In addition, several well-known Openflow controllers are evaluated to identify the most suited ones for implementing security solutions in SDN networks. As a result a framework to compare OpenFlow compatible SDN controllers regarding their security functions will be presented.

The active collection of network flow metrics and OpenFlow statistics for security monitoring builds the basis for this collaboration and therefore concerns WP5. Related to WP6 an automated mitigation and network (i.e. forwarding-plane) reconfiguration is done once a network attack has been detected.

4.2.12 Flowoid: a NetFlow/IPFIX Probe for Android-based Devices (UT-INRIA-Flowoid)

Analysis of the network behaviour of applications running on a smartphone device requires the collection of information about data leaving the device and where it is sent. Cisco’s NetFlow and the more recent IPFIX are flow export technologies that have seen a rapid adoption and widespread integration in many campus, enterprise and backbone networks. To be able to export and analyse mobile device characteristics (such as its location at the moment of certain network activity), the NetFlow and IPFIX protocols have to be extended. The flow exporter, flow collector and analysis application need to be aware of these extensions as well.

The work in this collaboration has been divided between INRIA and UT. On the one hand, INRIA is responsible for developing a flow exporter tailored to Android devices. On the other hand, UT is responsible for the flow collector and analysis application. The major achievements of the collaboration are:

- Development of a NetFlow and IPFIX metering process for Android devices;
- Extension of nfdump/Nfsen and SURFmap with location support;
- IETF Internet-Draft (ID) describing a set of information elements for IPFIX metering process location [112].

The following aspects of this collaboration fall within WP5. In this work, INRIA has developed Flowoid, a NetFlow and IPFIX metering process tailored to Android devices. The probe associates geolocation data with each observed network flow, consisting of the GPS coordinates of the mobile device, among others. This information is exported together with the traditional fields defined in the NetFlow and IFPIX: IP version, source and destination addresses, the number of exchanged bytes, the type of protocol, the number of exchanged packets, the source and destination ports, and the

duration of a flow. In addition, it contains seven additional fields that denote the identifier of the device: the identifier of the localization method, a timestamp, the integer part of the latitude, the decimal part of the latitude, the integer part of the longitude, and the decimal part of the longitude. UT has extended the state-of-the-art flow collector nfdump/NfSen[19] with location support, allowing us to analyse the flows exported by the Flowoid probe. In a previous work UT has developed a network monitoring tool based on the Google Maps API, named SURFmap [113],[20] which adds a geographical dimension to flow data and displays the data on a map. Since SURFmap [113] already supports network traffic geolocation (i.e. adding physical locations of hosts to network data), this tool has been extended to visualize the locations of devices on a map. This will allow us to visualize network traffic of mobile device with respect to devices locations.

Several technical aspects of this collaboration have been done in Y1. The main activity carried out in Y2 is related to improving the Internet Draft (ID) ([112]) based on community feedback. We will continue the collaboration regarding the improving of the ID in WP4. However, the collaboration regarding flow-based monitoring of Android devices, will be continued in WP5 and WP6 to develop a modular and flexible service to collect, store and analyze NetFlow data of running applications on the user's device.

The flow-based monitoring of network traffic produced by mobile devices, which is core to this collaboration, contributed to the monitoring activities of WP5. Relating the location information of a device to its network traffic can be beneficial to many network management and measurement applications, including traffic profiling, anomaly detection and provider-independent measurements. The developed approach allows us to better understand Android apps regarding their network flows and usage and it proves to be promising for the automated configuration of mobile networks that concerns WP6. When Metering Processes are running on devices with a (frequently) changing physical location, data analysis applications may need to be aware of these movements since they are likely to affect the behavior of the network in terms of routing, throughput, etc. Thus, configuration policies and actions could be applied to adapt the network according to the observed locations and maintain an acceptable quality of service of running applications on the user's devices. For example, knowing the location of a device at a moment of certain network activities could be used to dynamically reroute its traffic to closer data sources.

[19] `http://nfsen.sourceforge.net`
[20] `http://surfmap.sourceforge.net`

5 Framework to Support the Combination of Policy-Based and Semantic-Based Approaches

The core of FLAMINGO is the integration, dissemination and joint research. Developing new approaches in the area of network and service management, three main challenges arise. In order to make network and service management decisions (for instance security management, configuration management, cloud management or accounting), it is essential to have a good understanding of traffic flowing through the network, by monitoring traffic and network devices. These activities are part of WP5. Autonomic Networking builds upon self- and distributed management approaches. Such approaches require the development of new concepts, like coordination protocols, association strategies, information modeling, knowledge processing, and learning capabilities of managed objects. This is done by WP6. While aforementioned issues address technical dimensions rather than business, economic and legal aspects, WP7 covers investigations regarding policy management, economics, business incentives and legal constraints in a sense that all management actions are performed within the boundaries of economic, legal and regulative constraints.

Due to the tight cooperation between these three WPs, an appropriate monitoring and automatic configuration and repair management approach, inline with legal and regulative constraints, is conceivable. To realize such a valuable and joint research, we propose our FRAMEWORK TO SUPPORT THE COMBINATION OF POLICY-BASED AND SEMANTIC-BASED APPROACHES. Figure 29 shows the first overall design of our framework.

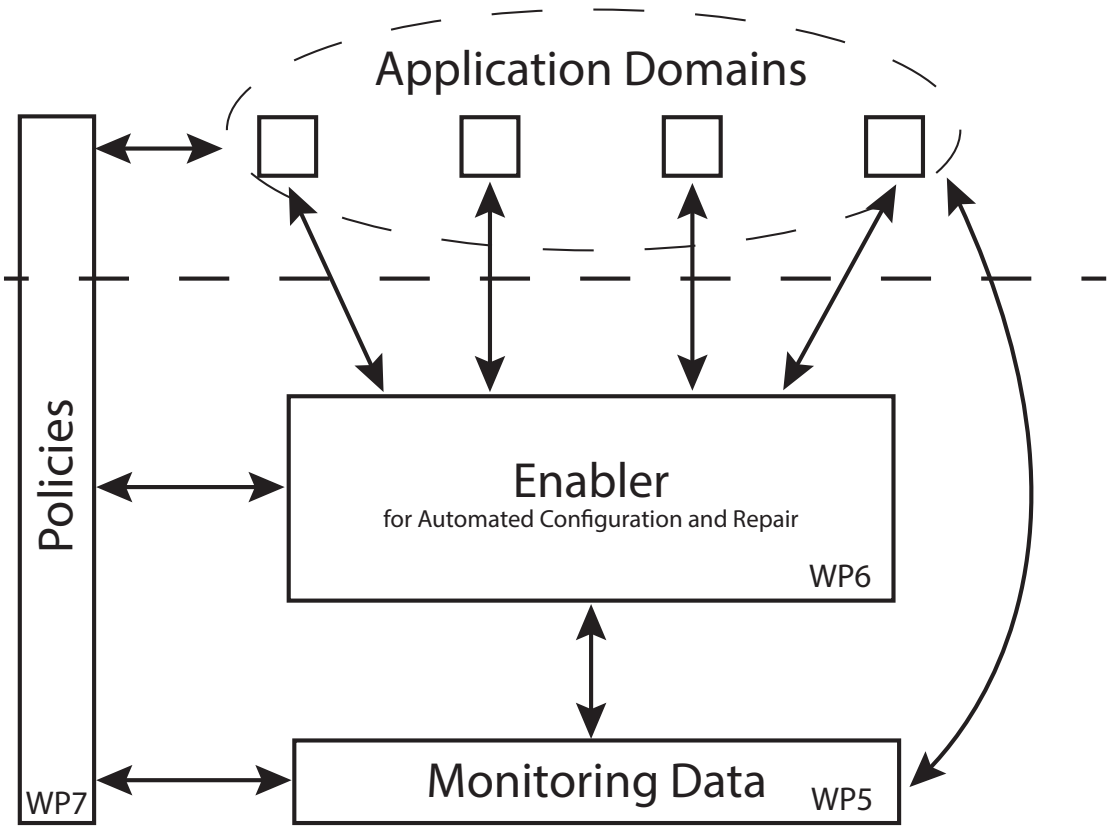

Figure 29: FLAMINGO Framework to Support the Combination of Policy-Based and Semantic-Based Approaches

On top of our framework the application domains build the environment for our developed approaches. Regardless if an approach originated from WP5 or WP6 it is definitively adopted to at least one application domain investigated by FLAMINGO. This is indicated in the Figure by the arrows from application domains to monitoring data and enabler respectively. A bi-directional exchange of information is realized, because monitoring data accrues in several application domains and builds a basis for rule based decisions. Since all enablers (detailed information about enabler used in FLAMINGO can be found in 3.5) need monitoring data as a basis to build a data base, from which a knowledge base as well as information models can be derived, they also exchange information and data in a bi-directional way.

This monitoring data is used as an input for correlation methods, learning techniques and knowledge description, what in turn builds the basis for the corresponding automated actions. In addition, enablers are used to gather more sophisticated monitoring data from various application domains to support management decisions. Since automatic monitoring and repair actions, data collection through monitoring, and application of approaches in various application domains, raises concerns in respect of legal and regulative constraints. All approaches have to be verified accordingly by WP7.

Views vary on whether or not an enabler or a process for gathering monitoring data in various application domains is inline with policies and constraints provided by WP7. Therefore, the communication and exchange of information between WP7 (Policies) and the other three levels (Application Domains, Enabler, Monitoring Data) is an essential task that should be performed continuously and in a bi-directional way. As a consequence enablers, applications and monitoring data collectors are modified to meet the criteria and policies provided by WP7.

As a next step towards the FLAMINGO integrative architecture for automated configuration and repair, the framework will be extended to standardize the process of exchanging regulative requirements, current deployment characteristics, and resulting configuration actions.

One example of many is the collaboration and the scenario Mobile Measurements **UZH-JUB-UniBwM-M2**. An Android application forms the basis for collecting data. Since the application runs on a mobile device, on that only constrained resources are available, it can be mapped to the application domain WSN. The monitoring data is stored in relational databases and analytical modeling algorithms are applied to gain valuable results.

The methodology of gathering measurement data, evaluating results and presenting the results in the application needs to be verified by policies. These policies arise through the application domain and jurisdiction where the data is stored and collected. As a consequence, only anonymized data about the network traffic and location of the user can be collected.

6 Conclusions and Outlook

Deliverable 6.2 describes the achievements of WP6 in the second year with respect to S.M.A.R.T. as well as work package specific objectives. This deliverable documents also the full achievement of these objectives. For more details about the achievement of the S.M.A.R.T. objectives, the reader is referred to Deliverable D8.2; the achievements of work package specific objectives in the second year have been reported in this deliverable.

The work package specific objectives in the second year have centered around (i) maintaining the architecture developed in the first year and developing an additional architecture considering the new application domain of SDN. (ii) building an inventory of enablers used within FLAMINGO and present applicability constraints taking into account the characteristics of the different application domains. In addition, D6.2 reports on two FLAMINGO Automation Architectures, based on the monitoring architecture, as presented in D5.1 but extended with automation aspects and the ability to support SDN. All WP6 specific objectives are in the status of ongoing research.

A strong integration of PhD collaborations (both, fully payed and not fully payed by FLAMINGO) is the basis for the immense and excellent scientific output, as well as the achieved results. The extreme, continuous success of PhD collaborations also in the second year, especially between WP5 and WP6, is a guarantee to obtain excellent research results in the next years as well, and to foster and extend the joint PhD collaborations.

Acknowledgments

This deliverable is based on input from the WP6 Partners of the FLAMINGO consortium. A particular acknowledgment goes to all the PhD students that have not only provided textual input, but that are working on a daily basis on the challenging research topics that we report.

7 Abbreviations

$3GPP$	3rd Generation Partnership Project
$6LoWPAN$	IPv6 over Low Power, Wireless Networks
ACE	Autonomic Communication Element
ACR	Automated Configuration and Repair
AE	Autonomic Element
AME	Autonomic Management Entities
ANA	Autonomic Network Architecture
$ANEMA$	Autonomic Network Management Architecture
ANM	Autonomic Network Management
ANN	Artificial Neural Networks
API	Application Programming Interface
AQM	Active Queue Management
AS	Autonomous System
AWS	Amazon Web Services
BP	Back-Propagation
$CASCADAS$	Component-ware for Autonomic, Situation-aware Communications and Dynamically Adaptable Services
CDN	Content Delivery Network
CLI	Command Line Interface
CPS	Cyber Physical Systems
CVE	Common Vulnerabilities and Exposures language
$CVSS$	Common Vulnerability Scoring System
$DACoRM$	Decentralised and Adaptive Network Resource Management Framework
$DASH$	Dynamic Adaptive Streaming over HTTP
DCE	Direct Code Execution
$DDoS$	Distributed Denial of Service attack
$DNS-SD$	DNS Based Service Discovery
$DODAG$	Destination Oriented Directed Acyclic Graph
DoW	Description of Work
DRA	Dynamic Resource Allocation
$ECMP$	Equal-Cost Multi-Path
EF	Expedited Forwarding
$EMANICS$	European Network of Excellence for the Management of Internet Technologies and Complex Services
ESB	Enterprise Service Bus
EU	European Union
FI	Future Internet
FN	False Negative
$FOCALE$	Foundation, Observation, Comparison, Action, Learning, rEason
FP	False Positive

GPS	Global Positioning System
HAS	HTTP Adaptive Streaming
$HTTP$	Hyper-text Transfer Protocol
$ICMP$	Internet Control Message Protocol
IDS	Intrusion Detection System
$IDMEF$	Intrusion Detection Message Exchange Format
$IETF$	Internet Engineering Task Force
ILP	Integer Linear Program
InP	Infrastructure Provider
$INRIA$	Institut National de Recherche en Informatique et Automatique
IoT	Internet of Things
IP	Internet Protocol
$IPFIX$	Internet Protocol Flow Information Export
ISP	Internet Service Provider
$ITU-T$	International Telecommunications Union - Telecommunications Standardization Sector
JNI	Java Native Interfaces
JUB	Jacobs University Bremen
LLN	Low-power and Lossy Networks
LRU	Least Recently Used
MAS	Multi-Agent System
$mDNS$	multicast DNS
MDP	Markovian Decision Processes
$MITM$	Man-In-The-Middle
MNO	Mobile Network Operator
MOS	Mean Opinion Score
$MP2P$	Multipoint-to-Point
$MPEG$	Moving Picture Experts Group
MTR	Multi-Topology Routing
MSS	Microsoft ISS Smooth Streaming
MTU	Maximum transmission unit
$NETCONF$	Network Configuration Protocol
$NFQL$	Network Flow Query Language
NFV	Network Functions Virtualization
$NIST$	National Institute of Standards and Technology
NSC	Network Simulation Cradle
$OSPF$	Open Shortest Path First
OTT	Over-the-Top
$OVAL$	Open Vulnerability and Assessment Language
OWL	Web Ontology Language
PHB	Per Hop Behavior
QoE	Quality-of-Experience
QoS	Quality-of-Service
RED	Random Early Detection
$RDBMS$	Relational Database Management System
$RMSE$	Root Mean Square Error
$ROLL$	Routing Over Low Power Lossy networks
RPL	Routing Protocol for Low power and Lossy Networks
$SACK$	Selective Acknowledgments
$SCAP$	Security Content Automation Protocol

SDN	Software-defined networking
SLA	Service Level Agreement
$S.M.A.R.T.$	Specific Measurable Achievable Relevant Timely
SN	Substrate Network
$SNMP$	Simple Network Management Protocol
$SPARQL$	SPARQL Protocol and RDF Query Language
SOA	Service-oriented architecture
SP	Service Provider
SSH	Secure Shell
$SWRL$	Semantic Web Rule Language
$P2P$	Peer-to-Peer
$P2MP$	Point-to-Multipoint
RL	Reinforcement Learning
TCP	Transmission Control Protocol
TD	Time Difference
TN	True Negative
$TNSM$	Transactions on Network and Service Management
TP	True Positive
TPM	Trusted Platform Module
TSP	Travelling Salesman Problem
$UniBwM$	Universität der Bundeswehr München
UCL	University College London
UDP	User Datagram Protocol
UPC	Universitat Politecnica de Catalunya
UT	University of Twente
UZH	University of Zürich
$VDBE$	Value-Difference Based Exploration
VoD	Video-on-Demand
VoS	Value of Service
VM	Virtual Machine
VNP	Virtual Network Provider
VPN	Virtual Private Networks
$WLAN$	Wireless Local Area Network
WP	Work Package
WSN	Wireless Sensor Network
XML	Extensible Markup Language

References

[1] M. Claeys, D. Tuncer, J. Famaey, M. Charalambides, S. Latre, G. Pavlou, and F. De Turck. Proactive Multi-tenant Cache Management for Virtualized ISP Networks. In *Network and Service Management (CNSM), 2014 10th International Conference on*, 2014.

[2] M. Claeys, D. Tuncer, J. Famaey, M. Charalambides, S. Latre, F. De Turck, and G. Pavlou. Towards multi-tenant cache management for isp networks. In *Networks and Communications (EuCNC), 2014 European Conference on*, pages 1–5, June 2014.

[3] M. Charalambides, D. Tuncer, L. Mamatas, and G. Pavlou. Energy-aware adaptive network resource management. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 369–377, May 2013.

[4] G. Hurel, R. Badonnel, A. Lahmadi, and O. Festor. Outsourcing Mobile Security in the Cloud. In *Proc. of the 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2014)*, pages 69–73. IFIP, 2014.

[5] Martín Barrère, Rémi Badonnel, Olivier Festor, et al. A sat-based autonomous strategy for security vulnerability management. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS)*, NOMS2014. IEEE, 2014.

[6] Rashid Mijumbi, J Serrat, J Rubio-Loyola, Niels Bouten, Filip De Turck, and Steven Latré. Dynamic resource management in sdn-based virtualized networks. In *1st International Workshop on Management of SDN and NFV Systems*, 2014.

[7] Vladan Devedzic. Knowledge modeling–state of the art. *Integrated Computer-Aided Engineering*, 8(3):257–281, 2001.

[8] Y Tina Lee. Information modeling: From design to implementation. In *Proceedings of the second world manufacturing congress*, pages 315–321. Citeseer, 1999.

[9] Steven Latré, Jeroen Famaey, and Filip De Turck. A semantic context exchange process for the federated management of the future internet. *International Journal of Network Management*, 24(1):1–27, 2014.

[10] Elias M Awad. *Systems analysis and design*. McGraw-Hill Professional, 1985.

[11] X.H. Wang, Da Qing Zhang, Tao Gu, and H.K. Pung. Ontology based context modeling and reasoning using owl. In *Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 18–22, March 2004.

[12] S. Bradley, A. Hax, and T. Magnanti. *Applied Mathematical Programming*. Addison Wesley, 1977.

[13] Olivier Chapelle, Vikas Sindhwani, and Sathiya S. Keerthi. Optimization techniques for semi-supervised support vector machines. *J. Mach. Learn. Res.*, 9:203–233, June 2008.

[14] John W. Chinneck. Practical Optimization: a Gentle Introduction. `www.sce.carleton.ca/faculty/chinneck/po.html`, 2001. Accessed: 2014-04-30.

[15] S. Lahaie. How to take the Dual of a Linear Program. `www.cs.columbia.edu/coms6998-3/lpprimer.pdf?`, 2008. Accessed: 2014-02-17.

[16] Wikipedia: Column Generation. `http://en.wikipedia.org/wiki/Column_generation`. Accessed: 2014-04-30.

[17] Cynthia Barnhart, Ellis L. Johnson, George L. Nemhauser, Martin W. P. Savelsbergh, and Pamela H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):pp. 316–329, 1998.

[18] Michel X. Goemans and David P. Williamson. Approximation algorithms for np-hard problems. *SIGACT News*, pages 144–191, 1997.

[19] Dimitri P. Bertsekas and Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2nd edition, September 1999.

[20] L. R. Ford, Jr. and D. R. Fulkerson. A suggested computation for maximal multi-commodity network flows. *Manage. Sci.*, 50(12 Supplement):1778–1780, December 2004.

[21] Jacques Desrosiers and MarcoE. Labbecke. A primer in column generation. In Guy Desaulniers, Jacques Desrosiers, and MariusM. Solomon, editors, *Column Generation*, pages 1–32. Springer US, 2005.

[22] Phil Simon. *Too Big to Ignore: The Business Case for Big Data*. John Wiley & Sons, 2013.

[23] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009.

[24] Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[25] Lucian Busoniu, Robert Babuska, Bart De Schutter, and Damien Ernst. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2010.

[26] Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2010.

[27] Gergely Neu, András György, Csaba Szepesvári, and András Antos. Online markov decision processes under bandit feedback. In *NIPS*, pages 1804–1812, 2010.

[28] Warren B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 2007.

[29] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer-Verlag, 1 edition, August 1999.

[30] Bertil Gustafsson, Heinz-Otto Kreiss, and Joseph Oliger. *Time dependent problems and difference methods*. Pure and applied mathematics. J. Wiley, New York, Chichester, Brisbane, 1995.

[31] Ana Marıa Galindo Serrano. Self-organized Femtocells: a Time Difference Learning Approach, 2012.

[32] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[33] Michel Tokic. Adaptive e-greedy exploration in reinforcement learning based on value differences. In *Proceedings of the 33rd Annual German Conference on Advances in Artificial Intelligence*, KI'10, pages 203–210, Berlin, Heidelberg, 2010. Springer-Verlag.

[34] L. P. Pitaevskii and E. M. Lifshitz. *Statistical Physics, Course of Theoretical Physics 5, 3rd Edition*. Oxford: Pergamon Press, January 1976.

[35] Michel Tokic. Adaptive ϵ-greedy exploration in reinforcement learning based on value differences. In *Proceedings of the 33rd annual German conference on Advances in artificial intelligence*, KI'10, pages 203–210. Springer-Verlag, 2010.

[36] Michel Tokic and Günther Palm. Value-difference based exploration: adaptive control between epsilon-greedy and softmax. In *Proceedings of the 34th Annual German conference on Advances in artificial intelligence*, KI'11, pages 335–346. Springer-Verlag, 2011.

[37] Thomas Stockhammer. Dynamic adaptive streaming over HTTP: standards and design principles. In *ACM Conference on Multimedia Systems (MMSys)*, 2011.

[38] Saamer Akhshabi, Ali C. Begen, and Constantine Dovrolis. An Experimental Evaluation of Rate-adaptation Algorithms in Adaptive Streaming over HTTP. In *ACM Conference on Multimedia Systems (MMSys)*, 2011.

[39] M. Claeys, S. Latré, J. Famaey, and F. De Turck. Design and evaluation of a self-learning http adaptive video streaming client. *Communications Letters, IEEE*, 18(4):716–719, April 2014.

[40] Maxim Claeys, Steven Latré, Jeroen Famaey, Tingyao Wu, Werner Van Leekwijck, and Filip De Turck. Design and optimisation of a (fa) q-learning-based http adaptive streaming client. *Connection Science*, 26(1):25–43, 2014.

[41] Maxim Claeys, Steven Latré, Jeroen Famaey, Tingyao Wu, Werner Van Leekwijck, and Filip De Turck. Design of a Q-Learning-based Client Quality Selection Algorithm for HTTP Adaptive Video Streaming. In *Workshop on Adaptive and Learning Agents (ALA)*, 2013.

[42] R.K.P. Mok, E.W.W. Chan, and R.K.C. Chang. Measuring the Quality of Experience of HTTP video streaming. In *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2011.

[43] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8(3):345–383, July 2000.

[44] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *Trans. Sys. Man Cyber Part C*, 38(2):156–172, March 2008.

[45] Gerald Tesauro, David M. Chess, William E. Walsh, Rajarshi Das, Alla Segal, Ian Whalley, Jeffrey O. Kephart, and Steve R. White. A multi-agent systems approach to autonomic computing. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '04, pages 464–471, Washington, DC, USA, 2004. IEEE Computer Society.

[46] Stefano Petrangeli, Maxim Claeys, Steven Latré, Jeroen Famaey, and Filip De Turck. A multi-agent q-learning-based framework for achieving fairness in http adaptive streaming. In *14th IFIP/IEEE Network Operations and Management Symposium (NOMS), Krakow, Poland, 2014*, 2014.

[47] J E Dayhoff and J M DeLeo. Artificial neural networks: opening the black box. *Cancer*, 91(8 Suppl):1615–1635, April 2001.

[48] A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *Information Theory, IEEE Transactions on*, 39(3):930–945, May 1993.

[49] Rémi Coulom. *Reinforcement Learning Using Neural Networks, with Applications to Motor Control*. PhD thesis, Institut National Polytechnique de Grenoble, 2002.

[50] Jeff Heaton. *Introduction to Neural Networks for Java, 2Nd Edition*. Heaton Research, Inc., 2nd edition, 2008.

[51] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, San Diego, CA, USA, 1999.

[52] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[53] Jing Li, Ji-hang Cheng, Jing-yuan Shi, and Fei Huang. Brief introduction of back propagation (bp) neural network algorithm and its improvement. In David Jin and Sally Lin, editors, *Advances in Computer Science and Information Engineering*, volume 169 of *Advances in Intelligent and Soft Computing*, pages 553–558. Springer Berlin Heidelberg, 2012.

[54] Wikipedia: Backpropagation. `http://en.wikipedia.org/wiki/Backpropagation`. Accessed: 2014-05-09.

[55] Raúl Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag New York, Inc., New York, NY, USA, 1996.

[56] Rashid Mijumbi, Juan-Luis Gorricho, Joan Serrat, Maxim Claeys, Filip De Turck, and Steven Latre. Design and evaluation of learning algorithms for dynamic resource management in virtual networks. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS)*, NOMS2014. IEEE, 2014.

[57] Stephen R. Garner. Weka: The waikato environment for knowledge analysis. In *In Proc. of the New Zealand Computer Science Research Students Conference*, pages 57–64, 1995.

[58] Gerald Tesauro. Temporal difference learning and td-gammon. *Commun. ACM*, 38(3):58–68, March 1995.

[59] Junfei Qiao, Ruiyuan Fan, Honggui Han, and Xiaogang Ruan. Q-learning based on dynamical structure neural network for robot navigation in unknown environment. In *ISNN (3)*, volume 5553 of *Lecture Notes in Computer Science*, pages 188–196. Springer, 2009.

[60] Shi chao Wang, Zheng xi Song, Hao Ding, and Hao bin Shi. An improved reinforcement q-learning method with bp neural networks in robot soccer. In *ISCID (1)*, pages 177–180. IEEE, 2011.

[61] D. Tuncer, M. Charalambides, G. Pavlou, and N. Wang. Dacorm: A coordinated, decentralized and adaptive network resource management scheme. In *Network Operations and Management Symposium (NOMS), 2012 IEEE*, pages 417–425, April 2012.

[62] D. Tuncer, M. Charalambides, G. Pavlou, and N. Wang. Towards decentralized and adaptive network resource management. In *Network and Service Management (CNSM), 2011 7th International Conference on*, pages 1–6, Oct 2011.

[63] D. Tuncer, M. Charalambides, R. Landa, and G. Pavlou. More Control Over Network Resources: An ISP Caching Perspective. In *Network and Service Management (CNSM), 2013 9th International Conference on*, 2013.

[64] Daphne Tuncer, Marinos Charalambides, Hisham El-Ezhabi, and George Pavlou. A hybrid management substrate structure for adaptive network resource management. In *The Sixth IEEE/IFIP International Workshop on Management of the Future Internet 2014 (ManFI2014)*, 2014.

[65] Steven Strogatz. The end of insight. *Brockman, John, What is your dangerous idea*, 2007.

[66] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group). RFC 2597, RFC Editor, Jun 1999. Updated by RFC 3260.

[67] B. Davie, A. Charny, J.C.R. Bennet, K. Benson, J.Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, and D. Stiliadis. An Expedited Forwarding PHB (Per-Hop Behavior). RFC 3246, RFC Editor, Mar 2002.

[68] The Openstack Project. `https://www.openstack.org/`. Accessed: 2014-07-03.

[69] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM, 2010.

[70] Jay Lepreau. Emulab - Network Emulation Testbed Home. [Online]. Available: `http://www.emulab.net/`. [Accessed: September 29, 2013].

[71] Sebastian Seeber, Anuj Sehgal, Björn Stelte, Gabi Dreo Rodosek, and Jürgen Schönwälder. Towards a trust computing architecture for rpl in cyber physical systems. In *2013 IFIP/IEEE International Conference on Network and Service Management (CNSM), Zürich, Switzerland, October 2013*, 2013.

[72] Anthéa Mayzaud, Anuj Sehgal, Rémi Badonnel, Isabelle Chrisment, and Jürgen Schönwälder. A study of rpl dodag version attacks. In *8th International Conference on Autonomous Infrastructure, Management and Security (AIMS), Brno, Czech Republic, June-July 2014*, 2014.

[73] A. Sehgal, A. Mayzaud, R. Badonnel, I. Chrisment, and J. Schönwälder. Addressing dodag inconsistency attacks in rpl networks. In *Addressing DODAG Inconsistency Attacks in RPL Networks, Montreal, Canada, September*, 2014.

[74] Christos Tsiaras, Anuj Sehgal, Sebastian Seeber, Daniel Dönni, Burkhard Stiller, Jürgen Schönwälder, and Gabi Dreo Rodosek. Towards evaluating type of service related quality-of-experience on mobile networks. In *7th IFIP Wireless and Mobile Networking Conference (WMNC), Vilamoura, Algarve, Portugal, May 2014.*, 2014.

[75] Anuj Sehgal, Vladislav Perelman, Siarhei Kuryla, and Jurgen Schonwalder. Management of resource constrained devices in the internet of things. *Communications Magazine, IEEE*, 50(12):144–149, 2012.

[76] A. Siljanovski, A. Sehgal, and J. Schönwälder. Service discovery in resource constrained networks using multicast dns. In *23rd European Conference on Networks and Communications (EuCNC), Bologna, Italy, June 2014*, 2014.

[77] Alexander Clemm. *Network management fundamentals*, volume 800. Cisco press Indianapolis, 2007.

[78] V. Bashko, N. Melnikov, A. Sehgal, and J. Schönwälder. Bonafide - a traffic shaping detection tool for mobile networks. In *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, pages 328–335. IEEE, 2013.

[79] D. Tuncer. *Engineering Self-Managed Adaptive Networks*. PhD Thesis. University College London, 2013.

[80] Holger Karl and Andreas Willig. *Protocols and architectures for wireless sensor networks*. John Wiley & Sons, 2007.

[81] Carsten Bormann, Mehmet Ersue, and A Keranen. Terminology for constrained node networks. *draft-ietf-lwig-terminology-00 (work in progress)*, 2013.

[82] AS Chipcon and RF Smart. Cc2420 preliminary datasheet (rev 1.2). *Copyright Each Manufacturing Comoany*, 2004.

[83] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393 – 422, 2002.

[84] Peter Mell and Tim Grance. Draft nist working definition of cloud computing. *Referenced on June. 3rd*, 15, 2009.

[85] Softlayer. Service level agreement and master service agreement. `http://www.softlayer.com/sla.html`. Last visited on 2014-07-09.

[86] David P Kormann and Aviel D Rubin. Risks of the passport single signon protocol. *Computer Networks*, 33(1):51–58, 2000.

[87] Kevin D Bowers, Ari Juels, and Alina Oprea. Hail: a high-availability and integrity layer for cloud storage. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 187–198. ACM, 2009.

[88] Matt Blaze, Joan Feigenbaum, John Ioannidis, and Angelos D Keromytis. The role of trust management in distributed systems security. In *Secure Internet Programming*, pages 185–210. Springer, 1999.

[89] Oracle. Wiring through an enterprise service bus. `http://www.oracle.com/technology/tech/soa/mastering-soa-series/part2.html`. Last visited on 2014-07-09.

[90] Subashini Subashini and V Kavitha. A survey on security issues in service delivery models of cloud computing. *Journal of Network and Computer Applications*, 34(1):1–11, 2011.

[91] Christopher Monsanto, Joshua Reich, Nate Foster, Jennifer Rexford, David Walker, et al. Composing software defined networks. In *NSDI*, pages 1–13, 2013.

[92] F. Idzikowski, S. Orlowski, C. Raack, H. Woesner, and A. Wolisz. Saving energy in IP-over-WDM networks by switching off line cards in low-demand scenarios. In *Proceedings of the 14th conference on Optical Network Design and Modeling (ONDM'10)*, pages 42–47, 2010.

[93] A. Coiro, F. Iervini, and M. Listanti. Distributed and adaptive interface switch off for internet energy saving. In *Proceedings of the 20th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–8, 2011.

[94] K. Scarfone and P. Mell. Guide to Intrusion Detection and Prevention Systems (IDPS). *NIST Special Publication*, 800(2007):94, 2007.

[95] M. Golling and B. Stelte. Requirements for a Future EWS-Cyber Defence in the Internet of the Future. In *Cyber Conflict (ICCC), 2011 3rd International Conference on*, 2011.

[96] GÉANT. Breakthrough GÉANT Network Marks Ten Years of Success: High Bandwidth pan-European Research Network Continues Advances with 100 Gbps Plans , November 2010.

[97] Mario Golling, Rick Hofstede, and Robert Koch. Towards Multi-layered Intrusion Detection in High-Speed Backbone Networks. In *Proceedings of the NATO CCD COE 6th International Conference on Cyber Conflict, CyCon'14*, 2014.

[98] Rick Hofstede, Václav Bartoš, Anna Sperotto, and Aiko Pras. Towards Real-Time Intrusion Detection for NetFlow/IPFIX. In *Proceedings of the 9th International Conference on Network and Service Management, CNSM'13*, pages 227–234, 2013.

[99] L. Hellemons, L. Hendriks, R. Hofstede, A. Sperotto, R. Sadre, and A. Pras. SSHCure: A Flow-Based SSH Intrusion Detection System. In *Proc. of the 6th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2012)*, volume 7279, pages 86–97. IEEE, 2012.

[100] R. Koch, M. Golling, and G. Dreo Rodosek. Evaluation of State of the Art IDS Message Exchange Protocols. In *International Conference on Communication and Network Security (CNS 2013)*, 2013.

[101] G. Dreo, M. Golling, W. Hommel, and F. Tietze. ICEMAN: An architecture for secure federated inter-cloud identity management. In *IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, May 2013.

[102] G. Dreo, M. Golling, and W. Hommel. MuSIC: An IT security architecture for inter-community clouds. In *IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, May 2013.

[103] Google Play: Bonafide+ Application. `https://play.google.com/store/apps/details?id=de.jacobs.university.cnds.bonafide.plus`. Accessed: 2014-09-22.

[104] Measurement Lab. `http://www.measurementlab.net/`. Accessed: 2014-09-22.

[105] Emanics Lab. `http://www.emanicslab.org`. Accessed: 2014-09-22.

[106] Bondafide. `http://www.bonafide.pw`. Accessed: 2014-09-22.

[107] Rashid Mijumbi, J Serrat, JL Gorricho, Maxim Claeys, Filip De Turck, and Steven Latré. Design and evaluation of learning algorithms for dynamic resource management in virtual networks. In *14th IEEE/IFIP Network Operations and Management Symposium (NOMS 2014)(submitted to)*, 2014.

[108] Rashid Mijumbi, Juan-Luis Gorricho, and Joan Serrat. Learning algorithms for dynamic resource allocation in virtualised networks. In *Management of Large Scale Virtualized Infrastructures: Smart Data Acquisition, Analysis and Network and Service Management in the Future Internet*, 2014.

[109] Rashid Mijumbi, Juan-Luis Gorricho, and Joan Serrat. Contributions to efficient resource management in virtual networks. In *IFIP 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS), 2014*, 2014.

[110] Rashid Mijumbi, JL Gorricho, J Serrat, Maxim Claeys, Jeroen Famaey, and Filip De Turck. Artificial neural network-based autonomous allocation of resources in virtual networks. In *To appear in proceedings of the European Conference on Networks and Communications (EUCNC)*, 2014.

[111] R. Mujumbi, J. Serrat, J. L. Gorricho, M. Shen, K. Xu, and K. Yang. A neuro-fuzzy approach to self-management of virtual network resources. In *Journal of Expert Systems With Applications (submitted to)*, 2014.

[112] O. Festor, A. Lahmadi, R. Hofstede, and A. Pras. Information Elements for IPFIX Metering Process Location (Internet Draft). http://tools.ietf.org/html/draft-irtf-nmrg-location-ipfix-01, July 2014.

[113] R. J. Hofstede and T. Fioreze. SURFmap: A Network Monitoring Tool Based on the Google Maps API. In *IFIP/IEEE International Symposium on Integrated Network Management (IM 2009)*, June 2009.