UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Danny Alex Lachos Peréz

MUDED: Integrating Networks with Applications through Multi-Domain Exposure and Discovery Mechanisms

MUDED: Integrando redes a aplicações por meio de mecanismos de descoberta e exposição em ambientes de multiplos domínios

Campinas

2021

Danny Alex Lachos Peréz

MUDED: Integrating Networks with Applications through Multi-Domain Exposure and Discovery Mechanisms

MUDED: Integrando redes a aplicações por meio de mecanismos de descoberta e exposição em ambientes de multiplos domínios

Thesis presented to the Faculty of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Doctor, in the area of Computer Engineering.

Tese apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutor em Engenharia Elétrica, na Área de Engenharia de Computação.

Supervisor: Prof. Dr. Christian Rodolfo Esteve Rothenberg

Este exemplar corresponde à versão final da tese defendida pelo aluno Danny Alex Lachos Peréz, e orientada pelo Prof. Dr. Christian Rodolfo Esteve Rothenberg

---

Campinas

2021

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

Lachos Peréz, Danny Alex, 1983-
L118m MUDED: integrating networks with applications through multi-domain exposure and discovery mechanisms / Danny Alex Lachos Peréz. – Campinas, SP : [s.n.], 2021.

Orientador: Christian Rodolfo Esteve Rothenberg.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Redes de computadores. 2. Arquitetura de redes. 3. Virtualização de redes. I. Esteve Rothenberg, Christian Rodolfo, 1982-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: MUDED: integrando redes a aplicações por medio de mecanismos de descuberta e exposição em ambientes de multiplos domínios
Palavras-chave em inglês:
Computer networks
Network architecture
Virtualização de redes
Área de concentração: Engenharia de Computação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora:
Christian Rodolfo Esteve Rothenberg [Orientador]
Jéferson Campos Nobre
José Ferreira de Rezende
Luiz Fernando Bittencourt
David Fernandez Cruz Moura
Data de defesa: 25-02-2021
Programa de Pós-Graduação: Engenharia Elétrica

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-3738-7320
- Currículo Lattes do autor: http://lattes.cnpq.br/5466177320244302

COMISSÃO JULGADORA – TESE DE DOUTORADO

Candidato: Danny Alex Lachos Peréz (RA: 153840)

Data da Defesa: 25 de Fevereiro de 2021

Título da Tese: "MUDED: Integrating Networks and Applications through Multi-domain Exposure and Discovery mechanisms".

Título em outro idioma: "MUDED: Integrando redes a aplicações por meio de mecanismos de descoberta e exposição em ambientes de multiplos domínios".

Prof. Dr. Christian Rodolfo Esteve Rothenberg (FEEC/UNICAMP) - Presidente

Prof. Dr. Jéferson Campos Nobre (UFRGS) - Membro Titular

Prof. Dr. José Ferreira de Rezende (UFRJ) - Membro Titular

Prof. Dr. Luiz Fernando Bittencourt (IC/UNICAMP) - Membro Titular

Dr. David Fernandes Cruz Moura (CTEX) - Membro Titular

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de PósGraduação da Faculdade de Engenharia Elétrica e de Computação.

I dedicate this PhD thesis to Evelin, my amazing wife, whose unconditional support for me and our children made it possible to complete this work; and to our children Alexander and Eitor, who are the greatest treasure that God has given us.

I also dedicate this work to my beloved mother, Mrs. Hilda Perez, who has been a lifelong inspiration for me. Thank you for your sacrifices, supports, advice, and prayers.

This dissertation is also dedicated to my siblings Veronica, Rossio, and Daniel, whose unyielding love and encouragement have inspired me to complete this work. Last, but certainly not least, I dedicate this thesis to my sweet nephews and nieces Aidric, Zoe, Aeowyn, and Sebastian.

Acknowledgements

First of all, I would like to thank my advisor, Prof. Dr. Christian Rothenberg. Without his innovative ideas, thoughtful encouragement, and careful supervision, this thesis would never have taken shape. I would therefore like to express my gratitude for being a great instructor and for the confidence in being able to work closely with his stellar group of students and collaborators.

At the University of Campinas (Unicamp), I have had the opportunity to learn from the best professors. I would like to acknowledge Prof. Dr. Carlos Alberto de Castro Junior, Prof. Dr. Guilherme Pimentel Telles, Prof. Dr. Tiago Fernandes Tavares, Prof. Dr. Marco Paulo Amorim Vieira, and Prof. Dr. Romis Ribeiro de Faissol Attux for sharing their knowledge and experience, which served as a constant motivation throughout this work. I would also take the opportunity to thank all collaborative activities, during this research work, with experts in the academy and industry such as Q. Xiang and R. Yang (Yale University), B. Ohlman, M. Santos and P. Gomes (Ericsson), F. Boten (T-Mobile), L. Contreras (Telefonica), S. Randriamasy (Nokia), Kai Gao (Sichuan University), and J. Zhang (Tongji University).

I would like to thank the committee members Jéferson Campos Nobre, José Ferreira de Rezende, Luiz Fernando Bittencourt, David Fernandes Cruz Moura, and Barbara Martini for the recommendations and insightful criticism they offered me in order to improve the final version of my thesis.

I had the good fortune to meet very talented people within the INTRIG and LCA groups. My special thanks go to Rony, Samuel, Samira, Gyanesh, Alex, Luis, Raphael, Anderson, Hirley, Ramon, Alaelson, Raza, Tariql, Nathan, Suneet, David, Claudio, Talita, Javier, Elias, Rodrigo, Roberto, Mariana, Ranyeri, Aldo, Wallace, Vitor, Raphael V. and Amadeu, Mariana, Lino, and Celso for their friendship and support throughout these years.

I would like to express special gratitude for the financial and technical support received from the Ericsson Innovation Center (Brazil), which allowed me to carry out this research.

"If your dreams do not scare you, they do not big enough."
—Ellen Johnson Sirleaf, *Nobel Peace Prize 2011*

"Discipline, sooner or later, will overcome the intelligence."
—Japanese proverb

Abstract

The evolving Internet application landscape is envisioned to adopt technologies such as SDN, NFV, and MEC to provide softwarized services, requiring resource orchestration across multiple networks managed by different technological and administrative domains. In such multi-domain settings, the collaboration between networks and applications provides opportunities to both applications to improve their performances and network service providers to increase their business offerings. Although many systems are proposed to support such collaborations, they are point or incremental solutions. In this work, we propose the exploration of a more integrated architecture with huge possibilities taking a network-application integration (NAI) approach. Specifically, we explore the NAI possibilities in two concrete aspects: application-aware networking and network-aware applications. We review recent progress in these two aspects and identify the key barriers in systematically realizing such a deep integration. To address these barriers, we propose a generic multi-domain NAI exposure and discovery framework, called MUDED. Through different systematic analysis and demonstrated prototypes, the MUDED's key components showcase: the maturing of the IETF ALTO protocol on the road to becoming a generic NAI possibilities discovery and exposure mechanism; and the optimized network view generation to simplify the network service placement and management.

Keywords: network-application integration; multi-domain; application-aware networking; network-aware applications; ALTO; network inventory.

RESUMO

Imagina-se que o cenário de aplicações da Internet adote tecnologias como SDN, NFV e MEC para fornecer serviços "softwarizados", exigindo orquestração de recursos em várias redes gerenciadas por diferentes domínios tecnológicos e administrativos. Em tais configurações de múltiplos domínios, a colaboração entre redes e aplicações proporciona oportunidades para tanto as aplicações melhorarem seu desempenho quanto os provedores de serviços de rede aumentarem suas ofertas de negócios. Embora muitos sistemas sejam propostos para suportar tais colaborações, eles são soluções pontuais ou incrementais. Neste trabalho, propomos a exploração de uma arquitetura mais integrada utilizando uma abordagem de integração de aplicações e rede (NAI por suas siglas em inglês). Especificamente, exploramos as possibilidades da NAI em dois aspectos concretos: rede ciente de aplicações e aplicações ciente de rede. Revisamos o recente progresso nesses dois aspectos e identificamos as principais barreiras para realizar sistematicamente tal integração. Para superar essas barreiras, propomos um framework genérico de exposição e descoberta de NAI em múltiplos domínio, denominado MUDED. Através de diferentes análises sistemáticas e protótipos demonstrados, os principais componentes do MUDED mostram: o amadurecimento do protocolo IETF ALTO de se tornar um mecanismo genérico de descoberta e exposição das possibilidade NAI; e a geração de uma visão de rede otimizada para simplificar o posicionamento e gerenciamento do serviço de rede.

Palavras-chaves: integração de aplicações-rede; multi-domínio; rede ciente de aplicações; aplicações ciente de rede; ALTO; inventário de rede.

List of Figures

List of Tables

Acronyms

3GP-DASH 3GPP Dynamic Adaptive Streaming over HTTP.

3GPP 3rd Generation Partnership Project.

5GEx 5G Exchange.

5GS 5G System.

ABR adaptive bitrate.

ALTO Application-Layer Traffic Optimization.

ANE Abstract Network Element.

ANI Abstracted Network Inventory.

AppD Application Descriptor.

BSS Business Support Systems.

CDN Content Delivery Network.

CDNI CDN Interconnection.

CO Central Orchestrator.

DO Domain Orchestrator.

E2E End-to-End.

ECN Explicit Congestion Notification.

GST Generic Network Slice Template.

HTTP Hypertext Transfer Protocol.

IEC International Electrotechnical Commission.

IETF Internet Engineering Task Force.

IFA Interfaces and Architecture.

IMS IP Multimedia Subsystem.

INT Inband Network Telemetry.

ISO International Organization for Standardization.

JTC Joint Technical Committee.

LIS Location Information Server.

LNI Logical Network Inventory.

LO Local Orchestrator.

MCS Modulation and Coding Scheme.

MdO Multi-domain Orchestrator.

MEC Multi-access Edge Computing.

MOS Mean Opinion Score.

MoWIE Mobile and Wireless Information Exposure.

MVC Multiview Video Coding.

NAI Network-Application Integration.

NEF Network Exposure Function.

NF Network Function.

NFV Network Function Virtualization.

NGMN Next Generation Mobile Networks.

NS Network Service.

NSD Network Service Descriptor.

NSP Network Service Provider.

OSS Operations Support Systems.

P2P Peer-to-peer.

P4P Provider Portal for Applications.

PCC Path Computation Client.

PID Provider-Defined Identifier.

QoE Quality of Experience.

QoS Quality of Service.

SDI Software-Defined Internetworking.

SDN Software-Defined Networking.

SF service function.

SFC Service Function Chaining.

SINR Signal to Inference plus Noise Ratio.

SLA Service Level Agreement.

SLO Service-level Objective.

SO Service Orchestrator.

SP Service Provider.

SSE Server-Sent Events.

SVC Scalable Video Coding.

SXP SFC eXchange Platform.

TP Transit Provider.

VL Virtual Link.

VNF Virtual Network Function.

WAN Wide Area Network.

WG Working Group.

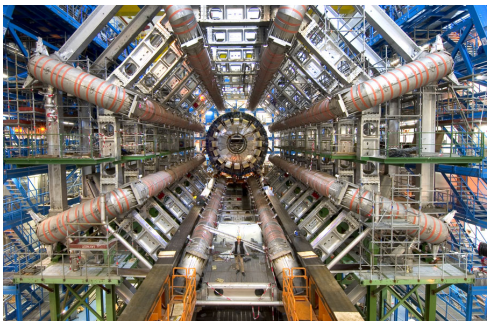
Contents

1 Introduction

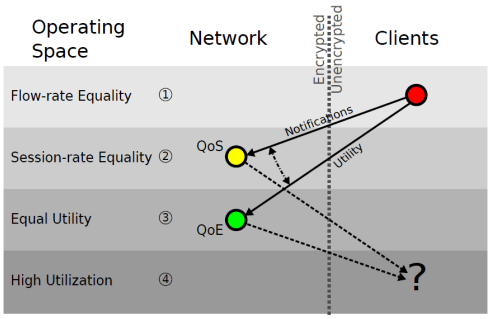
Traditionally, networks and applications have coexisted using the general-purpose and best-effort model of the internet. This network-application interaction has worked successfully over the years, with applications using the underlying network infrastructure and the network providing sufficient quality of service to the application's users (GAO *et al.*, 2020a). However, the best-effort approach is challenged due to the ever-growing demand for more complex applications with stringent application-specific requirements such as service guarantees, substantial resources, speed, reliability, etc. Data-intensive science applications (*e.g.*, colliders (See Fig. 1a), telescopes, and light sources), for instance, rely on networks as one of the key components of their infrastructure for local and global interconnection of laboratories, sites, and data centres (AL., 2020). Another unexpected but evident example is the current COVID-19 pandemic, with many institutional applications taking advantage of the network infrastructure to share data quickly and support collaborative efforts from multiple communities and disciplines such as medicine, health, genomics, and disaster mitigation (Pacific Wave, 2020a; Pacific Wave, 2020b). In the case of adaptive applications (*e.g.*, DASH), they may only achieve flow-rate equality when a network view is absent (JIANG *et al.*, 2012). The network, given a global view, is in the best position to achieve QoS- or QoE-level fairness (See Fig. 1b). Therefore, any allocation of resources with a goal beyond flow-based fairness is only possible when networks and applications collaborate (CHEN *et al.*, 2016). Flexible inter-domain routing (XIANG *et al.*, 2018; GUPTA *et al.*, 2015; MARQUES *et al.*, 2009) and End-to-End (E2E) network services (ALLIANCE, 2015; HALPERN; PIGNATARO, 2015; KATSALIS *et al.*, 2016; ETSI, 2020) are also emerging applications that construct complex data flows between users in the network.

1.1 Background and Motivation

The collaboration between networks and applications increases the quality and hence the business offering of the former, and the performance of the latter (XIE *et al.*, 2008; FERGUSON *et al.*, 2013a; SCHMIDT *et al.*, 2013). Consequently, different systems and mechanisms have been proposed to support such collaboration. However, they are point or incremental solutions with various limitations. For example, network providers and applications have considered different selfish solutions (OTT, 2005; KARAGIANNIS *et al.*, 2005; MADHYASTHA *et al.*, 2006; KUZMANOVIC; KNIGHTLY, 2006). However, such solutions are largely application/network-oblivious, making the interaction between them inefficient. Other solutions follow a "best-effort" (LEE *et al.*, 2014; SOULÉ



(a) Large Hadron Collider (LHC)[1]



(b) DASH applications[2]

Figure 1 – Network & Application interaction examples.

et al., 2014) or "blackbox-request" (CAMPANELLA *et al.*, 2006; ZHENG *et al.*, 2005) approach. Despite such proposals provide better network-application collaboration, they are typically implemented under the scenario of a single network, operated by a single commercial entity with their own bespoke applications (*e.g.*, Hadoop MapReduce, Google Search, Facebook).

At the crossroads, upcoming 5G applications such as virtual and augmented reality, on-line medicine, self-driving vehicles, gaming and others, are likely to take advantage of technologies such as Software-Defined Networking (SDN), Network Function Virtualization (NFV), Multi-access Edge Computing (MEC) to provide softwarized network services, decomposed into Virtual Network Functions (VNFs), usually instantiated in distributed resources which are available across multiple domains with different technology and/or administration (SUN *et al.*, 2018) (See Fig. 2). Such network services (or simply services) demand stringent requirements such as on-demand application deployment, information on deployment topology and network capabilities, scalability, and security. As a result, much of the existing work does not directly apply, and there is a need for two-way network-application interaction.

In this work, we propose to explore a more integrated and coherent architecture that takes a deep Network-Application Integration (NAI) approach. Specifically, we explore the possibilities of NAI in two concrete aspects: application-aware networking and network-aware applications. The first one allows applications to specify diverse requirements for the network infrastructure. The second one allows networks to expose underlying network information available to applications.

Despite the possibilities of NAI, systematically realizing it is non-trivial and There are still significant barriers to achieving such deep integration:

(B1) Network information exposure. Applications are lacking of visibility of avail-

[1] Figure source: <https://phys.org/>
[2] Figure source: <https://dl.acm.org/doi/pdf/10.1145/2940136.2940144>

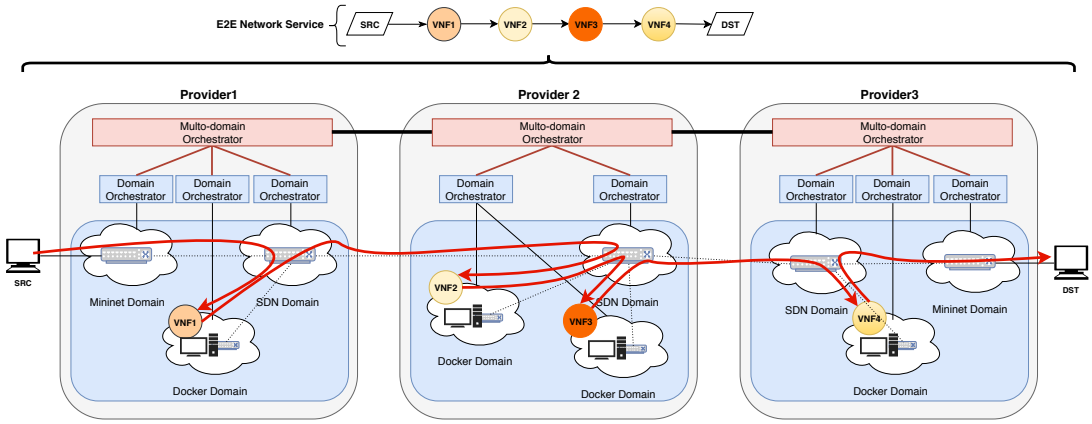


Figure 2 – An example for E2E Network Services across multiple domains (Multi-technology and Multi-administration)

able and shared network resources (*e.g.*, bandwidth of shared resources for a set of flows), resulting in poor performance. Existing network resource exposure mechanisms, including graph-based (HINDMAN *et al.*, 2011; VERMA *et al.*, 2015) and one-big-switch-based (XIE *et al.*, 2008; ALIMI *et al.*, 2014) representations, either expose all sensitive information, or fail to capture the resource sharing between virtual flow requests.

(B2) Network information discovery. The lack of a generic, flexible mechanism for applications to specify and discover the network information they need for NAI, from the network. Existing solutions (*e.g.*, (XIE *et al.*, 2008; ALIMI *et al.*, 2014; PUJOL *et al.*, 2019)) provide application interfaces to discover E2E cost information of different packet spaces. However, this information is derived from the network's fixed resource allocation (*e.g.*, fixed-route assignment) to the corresponding packet spaces, and applications are not provided the flexibility to discover additional network resources (*e.g.*, on-demand routing) that can satisfy their needs (*e.g.*, waypoint routing).

(B3) Optimized network view. Resource-filtering mechanisms may effectively reduce the redundancy in the network view. However, the number of available configurations would increase exponentially with both the network size and the number of services, which can be computationally expensive and time-consuming. To deal with scalability requirements, several solutions have been proposed for network view creation and abstraction (SONKOLY *et al.*, 2015; XIANG *et al.*, 2018; FIORANI *et al.*, 2015; FIORANI *et al.*, 2016; SOENEN *et al.*, 2016). However, none of the existing approaches take service requirements into account to generate an optimized network view representation.

To fill this gap, we propose MUDED, a **MU**lti-**D**omain generic NAI possibilities **E**xposure and **D**iscovery framework. The MUDED system resorts in two main mechanisms. First, a standardized discovery and exposure mechanism based on the IETF

Application-layer Traffic Optimization (ALTO) protocol (ALIMI *et al.*, 2014) to allow two-way network-application interaction. Second, a novel method (PEREZ *et al.*, 2020a) for constructing abstracted network information to address the scalability challenges when processing large amounts of data in multi-vendor, heterogeneous technology environments.

1.2 Hypothesis & Research Questions

The previous introduction argues the core motivating aspects of this work aligned with the following proposed hypothesis:

"Application and network integration is a key component for multi-domain settings, which for widely use should consider generic and standard mechanisms satisfying the ever so important features of NAI possibilities exposure and discovery, along with addressing the scalability and performance concerns".

The validation of such assertion takes place by pursuing three objectives:

(O1) Bridging the gap between networks and applications by showing the possibilities of NAI in two concrete aspects: application-aware networking and network-aware applications.

(O2) Demonstrate the maturing of the ALTO protocol to be used as a standard mechanism for exposing and discovering NAI possibilities in settings traversing multiple domains.

(O3) Provide a novel abstraction mechanism to generate service-optimized network inventory views among different domains to deal with scalability and performance requirements.

The research questions below aim to explore each of the objectives, resulting in the confirmation of this working hypothesis. Those research questions complement each other in order to establish a common ground and create the line of thought throughout this thesis:

For softwarized environments, instead of operating in the development of somewhat isolated and incremental solutions, we propose the exploration of a more integrated and coherent architecture. Such an architecture takes a deep NAI approach to accommodate a variety of applications and domain sciences and a variety of administrations and technologies. The design of a single coherence system considers both application-aware information to the network and network awareness in applications. Such research perspective is addressed in Chapter 2, "Deep Network & Application Integration", which poses the following research question:

Research Question #1: What are the possible means for a much stronger network & application collaboration than current mainstream networking?

Deep Network & Application Integration. In this chapter, we conduct a systematic review of the large variety of possibilities in designing and implementing NAI by application-aware networking and network-aware applications. We also present the design and evaluation of MUDED, a generic multi-domain NAI possibilities exposure and discovery framework.

Instead of building from scratch, we aim to design a multi-domain NAI framework that leverages the maturing of protocols and interfaces such as the IETF ALTO protocol. ALTO already introduces basic mechanisms (*e.g.*, modularity, dependency) and abstractions (*e.g.*, map services) for applications to improve their performance (LACHOS *et al.*, 2019). However, the current ALTO base protocol is not designed for a multi-domain setting of exposing network information. Towards such resolution, throughout Chapter 3 of this work, "Multi-domain Information Exposure & Discovery using ALTO", we pose the following research question:

Research Question #2: How to expose and discover multi-domain NAI possibilities using the IETF ALTO protocol?

Multi-domain Information Exposure & Discovery using ALTO. In this chapter, we identify what network information the multi-domain applications need and the benefit of using it. We then discuss the current ALTO design issues for gathering such multi-domain information along with basic mechanisms to be considered to allow ALTO to expose network information across multiple domains.

On the other hand, as the deployment size and heterogeneity complexity of softwarized multi-domain networks increase, one of the foremost challenges for management systems is how to handle the scale of network service provisioning effectively. A common system engineering principle to deal with scalability requirements is to introduce proper abstraction mechanisms that reduce the discovery time of network resources while simplifying their management. As Chapter 4 showcases, aiming "ANI: Abstracted Network Inventory", we pursue:

Research Question #3: How to effectively handle the scale and complexity of multi-domain environments to create proper abstract network views?

ANI: Abstracted Network Inventory. In this chapter, we propose the Abstracted Network Inventory (ANI) component to generate service-optimized network views over the same network inventory. ANI implements a novel abstraction method where network service requirements are used as an input to generate an optimized abstract network inventory representation, called Logical Network Inventory (LNI).

1.3 Thesis Approach & Contributions

In general, this research's main goal involves the systematic study of NAI that, for wide use, it needs to consider multi-domain standard mechanisms for exposing and discovering NAI possibilities, including mechanisms for providing an optimized network view to address the scalability concerns. We attain such a goal by the approaches and the contributions described in the next subsections, each defining an activity in the development of this work. Such activities are mapped to chapters of this work, as illustrated in Figure 3.

All the contributions are briefly explained and summarized as follows:

1.3.1 Deep Network & Application Integration

- **Main objective:** We review the possibilities in designing and implementing NAI through application-aware networking and network-aware applications (suggesting answers for the research question #1). we also propose the design of MUDED, a multi-domain generic framework for NAI possibilities exposure and discovery.
- **Main Contributions:**
  1. A systematic review of the large variety of possibilities in designing and implementing NAI by application-aware networking and network-aware applications.
  2. We present the design of MUDED, a NAI possibilities exposure and discovery system for network service placement in multi-technology and multi-administrative scenarios.
  3. A prototype implementation of MUDED[3], following the 5GEx project architectural design[4].
- **Related Publication:** From the systematic analysis to detailed design and experimental validation, the topics above have been addressed in the following publications, including two best paper awards and a contribution in an European funded initiative:
  - **D. LACHOS**, C. ROTHENBERG. "MUDED: Integrating Networks with Applications through Multi-Domain Exposure and Discovery Mechanisms". In *2020 Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN): Doctoral Symposium (NFV-SDN'20 Doctoral Symposium)*, Madrid, Spain. Nov 2020. **(Best Paper Award)**

[3] <https://github.com/intrig-unicamp/alto-based-broker-assisted-mdo>
[4] <https://doi.org/10.1002/ett.3085>

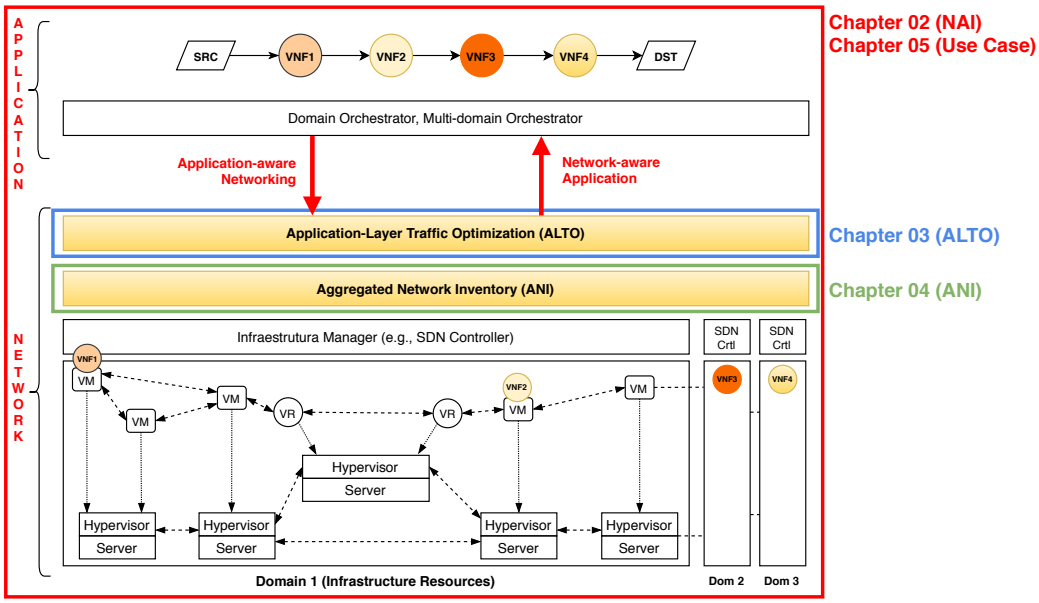


Figure 3 – Scope of the thesis chapters towards Network and Application Integration (NAI) through multi-domain exposure and discovery mechanisms.

  - **D. LACHOS**, Q. XIANG, C. ROTHENBERG, S. RANDRIAMASY, L. CONTRERAS, B. OHLMAN. "Towards Deep Network & Application Integration: Possibilities, Challenges, & Research Directions". In *ACM SIGCOMM'20 I Workshop on Network Application Integration*. Aug 2020.
  - **D. LACHOS**, C. ROTHENBERG. "ALTO-based Broker-assisted Multi-domain Orchestration". *IETF*, draft-lachosrothenberg-alto-brokermdo-04. Jul, 2020.
  - **D. LACHOS**, C. ROTHENBERG. "Multi-domain E2E Network Services". *IETF*, draft-lachosrothenberg-alto-md-e2e-ns-02 . Jul, 2020.
  - **D LACHOS**, C. ROTHENBERG. "Multi-domain Orchestration leveraging the Application-Layer Traffic Optimization Protocol". In *V Workshop Pre-IETF (V-WPIETF) - XXXVIII Congresso da Sociedade Brasileira de Computação (CSBC 2018)*. Brazil. Jul, 2018. **(Best Paper Award)**
  - **D. LACHOS**, C. ROTHENBERG, R. SZABÓ. "Broker-assisted Multi-domain Network Service Orchestration". In *IEEE Wireless Communications and Networking Conference (WCNC'18 Students)*, Spain. April 2018.
  - Deliverable 3.7 "Software Prototype Documentation and User Manual". *5GEx project contribution*. Dec 2017. <https://tinyurl.com/y3w348bj>.

- **Explored In:** Chapter 2 and Chapter 5

1.3.2 Multi-domain Information Exposure & Discovery using ALTO

- **Main Objective:** The current ALTO base protocol is not designed for a multi-domain setting of discovering and exposing network information. Via an analytical investigation of the ALTO framework, we elaborate key design requirements of ALTO for exposing multi-domain information along with a set of mechanisms to design a multi-domain ALTO framework, thus removing barriers **(B1)** and **(B2)** and suggesting an answer to the investigated research question #2.

- **Main Contributions:**

  1. We identify what network information the emerging multi-domain applications need and the benefit of using it.
  2. We give a systematic review of the ALTO design issues for multi-domains settings of exposing network information.
  3. We summarize envisioned solutions and on-going efforts to design a multi-domain ALTO framework.
  4. Different IETF document proposals on track towards standardization to support some of the envisioned mechanisms for ALTO to support multi-domain[5].

- **Related Publication:** All the contributions lead to the following publications, which include several standardization proposals at the IETF ALTO working group:

  – **D. LACHOS**, C. ROTHENBERG, Q. XIANG, R. YANG, B. OHLMAN, S. RANDRIAMASY, L. CONTRERAS, KAI GAO. "Multi-Domain Information Exposure using ALTO: The Good, the Bad and the Solution". In *ACM/IRTF Applied Networking Research Workshop 2020*. Jul 2020.

  – **D. LACHOS**, C. ROTHENBERG, Q. XIANG, R. YANG, B. OHLMAN, S. RANDRIAMASY, F. BOTEN, L. CONTRERAS, J. ZHANG, K. GAO. "Multi-domain Information Exposure using ALTO". *IETF*, draft-lachos-alto-md-info-exposure-00. Jul, 2020.

  – **D. LACHOS**, C. ROTHENBERG, Q. XIANG, R. YANG, B. OHLMAN, S. RANDRIAMASY, F. BOTEN, L. CONTRERAS. "Supporting Multi-domain Use Cases with ALTO". In *ACM/IRTF Applied Networking Research Workshop - ANRW'19*, Montreal, Canada. Jul 2019.

  – **D. LACHOS**, C. ROTHENBERG, Q. XIANG, R. YANG, B. OHLMAN, S. RANDRIAMASY, F. BOTEN, L. CONTRERAS, J. ZHANG, K. GAO. "Supporting Multi-domain Use Cases with ALTO". *IETF*, draft-lachos-alto-multi-domain-use-cases-01. Jul, 2020.

[5] <https://www.ietf.org/proceedings/108/slides/slides-108-alto-alto-re-charter-overview-00>

  - **D. LACHOS**, Q. XIANG, C. ROTHENBERG, R. YANG. "Multi-domain Service Function Chaining with ALTO". *IETF*, draft-lachos-sfc-multi-domain-alto-01. Jul, 2020.
  - L. CONTRERAS, **D. LACHOS**, C. ROTHENBERG. "Use of ALTO for Determining Service Edge". *IETF*, draft-contreras-alto-service-edge-02. Nov, 2020.

- **Explored In:** Chapter 3

1.3.3 ANI: Abstracted Network Inventory

- **Main Objective:** Suggesting answers for the proposed research question #3 and remove barrier **(B3)**, we propose the ANI component that implements a novel method (i) receiving services requirements from a catalog, (ii) receiving a network representation from a network inventory, and (iii) processing those inputs to generate a service-optimized abstract network view, referred to as LNI.

- **Main Contributions:**

  1. We propose the ANI as a novel component that allows the creation of service-optimized network inventory views in distributed environments. To the best of our knowledge, ANI is the first approach that uses service requirements as an input to generate an abstract network inventory.
  2. We formally define a network model and problem statement along with the development of three algorithms to generate an LNI given the capacity-related resources and requirements of a network inventory and network service, respectively.
  3. We evaluate the proposed algorithms through extensive experiments using random and real-world topologies[6]. Results show significant benefits of using an LNI to simplify the service management and placement.

- **Related Publication:** The contributions above can be summarized as follows, including a work as per patent application:

  - **D. LACHOS**, C. ROTHENBERG, M. SANTOS, P. GOMES. "ANI: Abstracted Network Inventory for Streamlined Service Placement in Distributed Clouds". In *6th IEEE International Conference on Network Softwarization (NetSoft'20)*. Jun, 2020.

[6] <https://github.com/intrig-unicamp/ani>

  - M. SANTOS, **D. LACHOS**, P. GOMES, C. ROTHENBERG, A. VIDAL. "Technique for Simplifying Management of a Service in a Cloud Computing Environment". PCT Patent Application Serial No. PCT/EP2019/058274. Filing Date: 02 Apr, 2019. Publication Number: WO/2020/200427. Publication Date: 08 Oct, 2020. Available from Internet: <https://patentscope.wipo.int/search/en/detail.jsf?docId=WO2020200427>

- **Explored In:** Chapter 4

1.3.4 Further Contributions, Results & Collaborative Activities

The complete effort around this work incorporates a number of collaborative activities referenced to the corresponding scientific article, patent, blog post and/or demo indicating the co-authors. The list of publications is shown below:

- APNIC Blog. "Multi-domain information exposure using ALTO: The good, the bad and the solution". Available from Internet: <https://blog.apnic.net/2020/10/08/multi-domain-information-exposure-using-alto-the-good-the-bad-and-the-solution/>. Oct 08, 2020.
- APNIC Blog. "ALTO: Application-Layer Traffic Optimization protocol". <https://blog.apnic.net/2020/10/07/alto-application-layer-traffic-optimization-protocol>. Oct 07, 2020.
- C. ROTHENBERG, **D. LACHOS**, N. SARAIVA, R. ROSA, R. MUSTAFA, T. ISLAM, P. GOMES. "Intent-based Control Loop for DASH Video Service Assurance using ML-based Edge QoE Estimation". In *6th IEEE International Conference on Network Softwarization (NetSoft'20) - Demo Session*, Ghent, Belgium. Jun, 2020.
- C. ROTHENBERG, **D. LACHOS**, N.SARAIVA, P. GOMES. "Method, System and Network Node for Generating a Network Service Monitoring Model". PCT Patent Application Serial No. PCT/SE2020/050271. Filing Date: 16 Mar, 2020.
- N. SARAIVA, N. ISLAM, **D. LACHOS**, C. ROTHENBERG. "Policy-Driven Network Traffic Rerouting Through Intent-Based Control Loops". In *Anais do XXIV Workshop de Gerência e Operação de Redes e Serviços*. Sep, 2019.
- N. SARAIVA, **D. LACHOS**, R. ROSA, M. SANTOS, C. ROTHENBERG. "Network Service Orchestration: A Survey". In *The Computer Communications Journal*. May, 2019.

During the development of this thesis proposal, collaborations with industrial actors like Telefónica, Ericsson, Nokia, and T-Mobile have also been obtained, including

a couple of short visits to leading institutions such as Yale University. Besides, several successful grant proposals were also prepared for attending different conferences and workshops, such as IRTF/ACM Applied Networking Research Workshop (ANRW'19), Latin American Student Workshop on Data Communication Networks (LANCOMM'19), IETF-104 (2019), and ONAP Academic Summit (2018). Finally, getting involved in the peer review process at conferences (such as IMC'18 Shadow PC[7]) and journals (such as IEEE Transactions on Industrial Informatics[8]) were also other rewarding results during this work.

1.4 Outline

The remainder of this thesis is organized as follows. Chapter 2, "Deep Network & Application integration", proposes the exploration of an integrated architecture taking a network-application integration (NAI) approach. Specifically, it explores the NAI possibilities in two concrete aspects: application-aware net-working and network-aware applications. Next, it presents the design of MUDED, a generic multi-domain NAI possibilities exposure and discovery framework, based on two novel components such as the IETF Application-Layer Traffic Optimization (ALTO) protocol and the Abstracted Network Inventory (ANI) component. Chapter 3, "Multi-domain Information Exposure & Discovery using ALTO", identifies the benefits of using ALTO for multi-domain information exposure and discovery and discusses the ALTO design issues for gathering it. Besides, it also elaborates key design requirements to realize the proposal of providing multi-domain information by ALTO services. Chapter 4, "ANI: Abstracted Network Inventory", proposes the Abstracted Network Inventory (ANI) component to generate service-optimized network views called Logical Network Inventory (LNI). It also provides a formal definition of the network model along with the development of three algorithms to build an LNI. Chapter 5, "MUDED Use Case: 5GEx Information Exchancge", describes a MUDED prototype implementation into the 5G Exchange (5GEx) project architectural design, along with a functional and performance evaluation. Finally, we present our conclusions with remarks for future goals and activities in Chapter 6, "Conclusion & Future Work".

[7] <https://conferences.sigcomm.org/imc/2018/shadow/>
[8] <https://publons.com/researcher/3833094/danny-alex-lachos-perez/peer-review/>

2 Deep Network & Application Integration

2.1 Introduction

A fundamental problem in the Internet architecture is that applications and networks are designed with different objectives (XIE *et al.*, 2008). On the one hand, network applications (*i.e.*, network resource consumers) aim to optimize the application's utility (*e.g.*, maximize throughput, robustness, etc.). On the other hand, network providers (*i.e.*, Internet service providers or ISPs) aim to optimize the network's utility (*e.g.*, minimize inter-domain cost, minimize MLU - Maximum Link Utilization, etc.). With this inefficient interaction, getting a feedback between networks and applications is extremely limited.

In order to fill this gap, network providers and applications have considered different selfish solution approaches (See Fig. 4a). ISPs, for example, attempt to improve the application issues through an infrastructure upgrade, usage-based charging model, rate limiting, or termination of services (OTT, 2005). Meanwhile, applications attempt to improve the network efficiency having flexibility in shaping communications patterns as well as having flexibility to adapt to network topologies and conditions (KARAGIANNIS *et al.*, 2005; MADHYASTHA *et al.*, 2006; KUZMANOVIC; KNIGHTLY, 2006). However, the poor network-application cooperation in this approach does not allow to improve both network and application utility.

Other existing solutions adopt either a "best-effort" (LEE *et al.*, 2014; SOULÉ *et al.*, 2014) or "blackbox-request" (CAMPANELLA *et al.*, 2006; ZHENG *et al.*, 2005) approach (See Fig. 4b). In the first one, solutions allow applications to submit complete network requirements, and the network computes and enforces the optimal resource allocation for applications. In the second one, applications submit the amount of network resources needed, and the network returns success or failure based on the resource availability. Such solutions are quite good but nevertheless typically implemented for bespoke applications, such as big-data (Spark/Hadoop MapReduce), search (Google), social-network (Facebook). Besides, either "best-effort" or "blackbox-request" approach has limitations on the privacy/scalability, or has inefficiency in finding the optimal resource allocation for applications, respectively.

In this chapter, we propose to explore a more integrated and coherent architecture that takes a deep network-application integration (NAI) approach (See Fig. 4c). Specifically, we explore the possibilities of NAI in two concrete aspects: application-aware networking and network-aware applications. The first one allows applications to specify diverse requirements for the network infrastructure. The second one allows networks

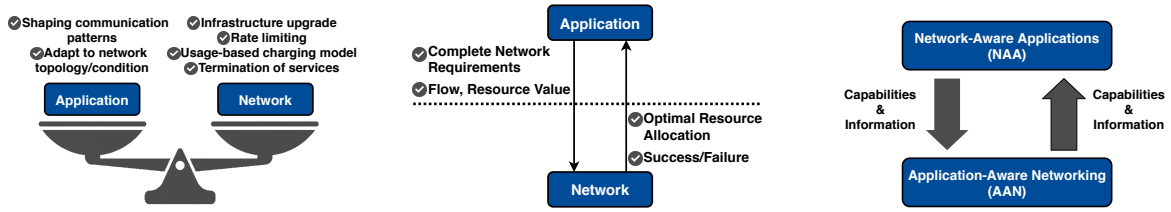


Figure 4 – Different approaches for the interaction of networks and applications: (a) selfish point, (b) best-effort/black-box approach, and (c) network-application integration (NAI) Approach.

to expose underlying network information available to applications. We review recent progress in these two aspects and present the design of a generic NAI possibilities exposure and discovery framework, called MUDED. MUDED resorts in two mechanisms. First, an ALTO-based discovery and exposure mechanism to allow two-way network and application interaction. One, in the "top-down" direction, is related to the discovery expression by the application of the required properties and characteristics needed to be supported by the network. The other one, in the "bottom-up" direction, is related to the network information exposure that can be processed and consumed by the application. Second, a novel abstraction method directed to the ANI component to generate service-optimized network views or LNIs.

The main contributions of this chapter are organized as follows. Section 2.2 and 2.3 provide a systematic overview of NAI possibilities by application-aware networking and network-aware applications, respectively. Section 2.4 presents the design of the MUDED framework, including the three key design points: (i) resource discovery language, (ii) resource information exposure, and (iii) abstracted network inventory. Finally, we conclude the chapter in Section 2.5.

2.2 Possibilities of NAI: Application-Aware Networking

Applications have varying needs for network latency, bandwidth, packet loss, etc. However, such applications' requirements are often unknown to the network due to applications and networks are decoupled. Thus, one concrete aspect of NAI is adding application knowledge to the network so that applications can express finer granularity requirements.

There are substantial possibilities in designing and implementing NAI by application-aware networking. For example, the network infrastructure can provide better support for applications introducing different capabilities. Table 1 shows a set of transport differentiation capabilities for applications and the newer trend where applications can also provide in-network computation or in-network storage.

Table 1 – Application-aware networking: possibilities.

Example Capability	Network Support
Provide Transport Differentiation	
• At app-level granularity	
Create different networks/slices/QoS	Classification; Scheduling
• At sub-app granularity	
Scheduling each packet according to app-level deadline (PERRY *et al.*, 2014); Distinguish application-level structures (*e.g.*, I frame vs P frame); Co-flow schedule	Classification; Network State; Scheduling
• Cross-app/protocol dependency	
Identify full dependency (*e.g.*, DNS->handshake->...)	Classification; Network state; Scheduling
Provide In-Network Storage/Compute	
• Application state inside the network	
Key-Value Store	Programmable networking
• Application compute inside the network	
Paxos algorithms	Programmable networking

2.2.1 Standard Proposals

In this section, we review current initiatives on standardization bodies to express characteristics or properties of applications/services to be achieved by the network.

- **ETSI NFV-IFA.** The Interfaces and Architecture (IFA) Working Group (WG) in ETSI NFV describes the ETSI NFV architecture, information models, functional requirements, and normative Network Service Descriptors (NSDs) (ETSI, 2016). In particular, the NSD is used by an NFV orchestrator to deploy a Network Service (NS) instance. Basically, the NSD is a deployment template describing a NS as a composition of VNFs and Virtual Links (VLs). Both VNFs and VLs contain a set of attributes to assist the resource instantiation for functions and links, respectively.

  The VNFs and VLs have their corresponding descriptors to specify list of constraint (*e.g.*, QoS properties) and monitoring (*e.g.*, VNF scaling) parameters. Monitoring parameters can also be specified at the NS level.

- **GSMA & 3GPP.** Evolving network scenarios (*e.g.*, 5G) are bringing new challenges in terms of performance and capabilities for traditional operator networks (GAO *et al.*, 2020a). One of those challenges is the ability to partitioning one common network infrastructure into multiple independent virtual E2E networks, an approach referred to as network slicing. Each slice is then used as a dedicated network by vertical customers to allocate resources in all the distinct network segments (*i.e.*, core, transport, and access networks).

In the 3GPP architecture (3GPP, 2017), management systems will use a 5G slice template (known as Generic Network Slice Template (GST) (ASSOCIATION, 2019a)) to request a E2E slice. The GST acts as a generic framework for applications to communicate demands for network capabilities, and its general structure includes optional, conditional, and mandatory attributes to implement slices, along with the expected Service-level Objectives (SLOs) and scalability characteristics.

- **TeleManagement Forum (TM Forum).** In order to facilitate the interaction between network providers and applications, the TM Forum has been working to develop a set of open APIs to request services with the required management functionality (TM-FORUM, 2020).

  The Service Catalog Management API (TMF633), for example, allows the management of the entire lifecycle of the catalog elements which are available through services that an operator offers to the customers. The Service Ordering API (TMF641) specifies a model definition to request a service order including operations such as creating, updating, retrieving, and filtering. The Service Inventory API (TMF638) provides a standardized mechanism to query and manipulate a customer's service instances. Finally, the Service Activation and Configuration API (TMF640) enables the creation, modification, and termination of service instances. Such lifecycle management actions include the collection of monitoring data through the definition of threshold/periodic-based rules.

- **ETSI Multi-access Edge Computing (MEC).** The Application Descriptor (AppD) (ETSI, 2019), specified by the ETSI MEC, is used to describe the application rules and requirements of a MEC application, including application lifecycle management. Specifically, the AppD contains a description of minimum computation, storage, and network resources, along with other aspects such as a description of traffic and DNS rules.

  The deployment of MEC in a NFV environment is also possible (ETSI, 2018). Therefore, NSDs and AppDs can co-exist to allow mobile edge components and applications to be instantiated in a virtualized infrastructure.

  Currently, there is no common agreement on the model/interface of an intent, especially since the diversity of intent stakeholders, targets, network scopes, and infrastructure. Based on such parameters, the NMRG is working to clarify the definition of an intent, including an intent classification (LI *et al.*, 2020).

2.2.2 Research contributions

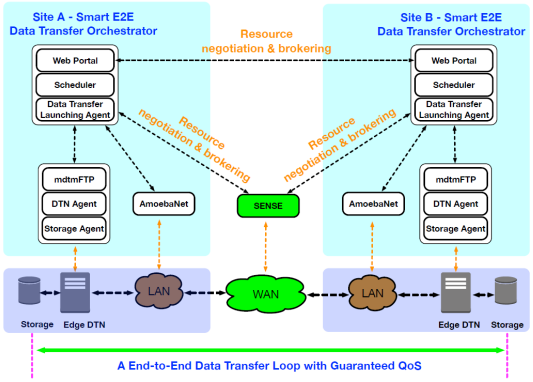
Several research activities have been proposed exploring the possibilities of adding application knowledge to the network layer (YANG *et al.*, 2017; SCHMIDT *et al.*, 2013;

LI *et al.*, 2020b; FERGUSON *et al.*, 2013b; ZHAO *et al.*, 2018):

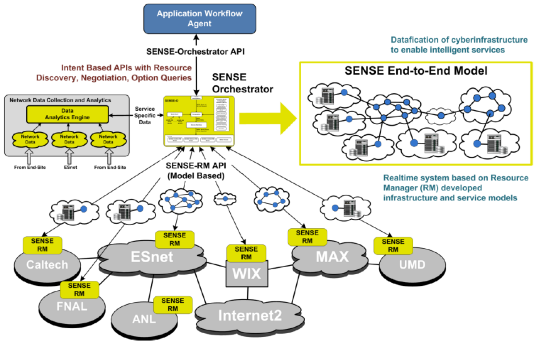
- Magellan (YANG *et al.*, 2017), for instance, is a programming environment for users to specify a global packet/in-network processing logic which is expressed in a general-purpose language. Then, Magellan automatically generates both datapaths in every single network device and runtime for control plane.
- Schmidt *et al.* (SCHMIDT *et al.*, 2013) introduce *Socket Intents* as a proactive, application-expressed approach for multi-access network connectivity. *Socket Intents* allow applications to share information, in a generic way, about their communication patterns such as preferences (*e.g.*, bandwidth optimization), characteristics (*e.g.*, expected packet rates), expectations (*e.g.*, paths availability), and resiliences (*e.g.*, handle certain error cases).
- Application-aware IPv6 Networking (APN6) (LI *et al.*, 2020b) proposes a framework for using IPv6 extensions header to convey the service/application requirements along with the packet to the network. The application awareness introduced by APN6 can benefit different use cases, such as application-aware SLA guarantee, application-aware network slicing, and application-aware network measurement.
- Ferguson *et al.* (FERGUSON *et al.*, 2013b) introduce the concept of *participatory networks* in which the network provides a configuration API for applications to control a software-defined network. The proposed API, called PANE API, is used in different use case applications (Ekiga, SSHGuard, ZooKeeper, and Hadoop) in which information from applications benefits network flexibility, configuration, and performance.
- An SDN-based application-aware networking approach to improve the users' quality of experience by optimizing the performance of DASH video streaming is presented in (ZHAO *et al.*, 2018). The proposed architecture provides a fine-grained way of controlling network devices with adaptive traffic engineering to a current network condition.

2.2.3 Real deployment examples

- BigData Express (LU *et al.*, 2018) is a data transfer service for big data science. It provides an application-aware SDN-enabled network service to program networks with fast provisioning of multi-domain E2E network paths at run-time and with guaranteed QoS (See Fig. 5a). BigData Express is currently deployed in several research institutions, including UMD, FNAL, StarLight, KISTI, KSTAR, and Ciena.



(a) BigData Express[1]



(b) SENSE[2]

Figure 5 – Application-aware Networking: Real deployment examples.

- The SDN for E2E Networked Science at the Exascale (SENSE) (MONGA *et al.*, 2018) is another system providing an intuitive intent-based interface to allow applications to express high-level service requirements. A multi-institution testbed (See Fig 5b) has been deployed at DOE Laboratories and Universities facilities, including Caltech, Fermilab, UMD, NERSC, among others.

2.3 Possibilities of NAI: Network-Aware Applications

Applications running over networks face challenges due to the lack of network state and information. Applications can benefit from network information exposure to make them more flexible in terms of rate adaptation, transmission time, server/path selection, among others. Therefore, the other side of designing and implementing NAI is network-aware applications, and there are many possibilities as well.

Table 2 illustrates that applications have possibilities to conduct transport selection capabilities based on network state (*e.g.*, packet loss, Inband Network Telemetry (INT)), performance metrics (*e.g.*, throughput, max reservable Bandwidth), capability information (*e.g.*, delivery/acquisition protocol), and locality (*e.g.*, servers location and paths). Besides, if network can provide programmability support, then applications can also use that support to conduct network compute selection.

2.3.1 Standard Proposals

- **3rd Generation Partnership Project (3GPP).** There are different standardization activities within the 3GPP to make possible for applications to obtain network

[1] Figure source: <http://grp-workshop-2019.ucsd.edu/presentations/4_WU-GRP-2019-BigData_Express.pdf>

[2] Figure source: <https://ieeexplore.ieee.org/document/8648795>

Table 2 – Network-aware applications: possibilities

Example Capability	Network Support
Conduct Transport Selection	
• Time adaption	
Bandwidth time window	Network state; Capability information
• Server/Path adaption	
e.g., Which servers to use in multiple replicas	Network state; Capability information
• Rate adaption	
Congestion control (reacting to packet loss/delay/ECN bitdelay, ECN bit (RAMAKRISHNAN *et al.*, 2001)/ INT (KIM *et al.*, 2015)); Adaptive streaming; Lower-than-best-effort (*e.g.*, LEDBAT); Multipath TCP.	Network state; Capability information
Conduct Network Compute Selection	
• Network function instantiation and invocation	
e.g., Function as a service (FaaS)	Programmable networking

information to improve both application performance and potentially network efficiency. For example, the system architecture for the 5G System (5GS) (3GPP, 2020) describes a Network Exposure Function (NEF) to expose capabilities and events with a secure provision (according to the network policy) and translation capacities between internal and external network functions. In addition to the previous generic network performance information, a QoS notification control is introduced so that the network can indicate to applications if the required bit rate cannot be provided. Additional information such as which bit rate needs to be selected is also introduced with a QoS profile so that applications can adapt their bit rates according to those supported by the network.

Furthermore, the 3GPP Dynamic Adaptive Streaming over HTTP (3GP-DASH) document (3GPP, 2019) provides an architectural overview to deliver continuous media content over Hypertext Transfer Protocol (HTTP). Specifically, DASH-aware network elements provide detailed network throughput information to a DASH server, so that the server selects chunks to realize bandwidth usage that is either equal or below to the available network bandwidth.

- **Internet Engineering Task Force (IETF).** One maturing example of network-aware application protocols is ALTO (ALIMI *et al.*, 2014). The IETF ALTO protocol exposes network state and capabilities to support efficient construction of diverse network-aware applications models, such as CDN model, swarm model, dataflow/streaming model, etc. Network information is exposed as abstractions (*e.g.*, network/cost maps) to protect the information privacy and improve the scalability

(See Chapter 3 for more details about the IETF ALTO protocol).

Another standard contribution is the Explicit Congestion Notification (ECN). ECN (RAMAKRISHNAN *et al.*, 2001) is an Internet standard track protocol, in the transport layer, to indicate fast congestion notification to the endpoints. The ECN has been supported by the 4G radio station (eNB - evolved Node Base station) to provide congestion encountered information to IP Multimedia Subsystem (IMS) applications to perform the adaptive bitrate. Another areas of experimentation for the ECN include, among others, congestion response differences, congestion marking differences, and TCP control packets and retransmissions (BLACK, 2018).

- **ISO/IEC JTC.** The Joint Technical Committee (JTC) of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) develops technical specifications within the field of audio, picture, multimedia, and hypermedia information coding (SubCommittee 29[3]). The MPEG-DASH standard (23009, 2017), for instance, is a specification widely used for the application to detect network congestion based on the current throughput and buffering states, and adaptively select the next segment of video streaming with acceptable bitrate to avoid the re-buffering (ZHANG *et al.*, 2020). Besides, the MPEG-DASH specification includes some other additional features such as switching and selectable streams, ad-insertion, Scalable Video Coding (SVC) and Multiview Video Coding (MVC) support, quality metrics for reporting the session experience, among others.

2.3.2 Research contributions

There are different proposals introducing the benefits of network awareness for applications (XIE *et al.*, 2008; YANG *et al.*, 2020; ZHANG *et al.*, 2020; XIANG *et al.*, 2020a):

- Provider Portal for Applications (P4P) (XIE *et al.*, 2008) is a framework to enable a better cooperation between network providers and network applications. P4P iTrackers accelerate the content distribution and optimize the utilization of ISP network resources. Specifically, each network provider can maintain an iTracker for its network, and appTrackers then can request it network information to make peer selection. As a variant, trackerless systems can also interact with an iTracker. In this case, P2P clients query the iTracker directly to make local decisions to select their peers.

- (MAO *et al.*, 2017; MAO *et al.*, 2020) implement AI-based adaptive bitrate (ABR) systems (Pensieve and ABRL, respectively). Unlike traditional ABR algorithms that

[3] <https://www.iso.org/committee/45316.html>

use fixed heuristics, AI-based ABR algorithms use system models generated from performance observations or past decisions in order to know the network dynamics. With this knowledge, applications can estimate or predict the overall users' QoS/QoE. AI models use different network information as an input, such as network-level metrics (*e.g.*, bandwidth, packet loss, delay, etc.), QoE parameters (*e.g.*, Mean Opinion Score (MOS), download time, etc.), among others. Pensieve and ABRL have proven that all different information that can reflect the performance are useful to the AI-based ABR.

- Delivering functions over networks (YANG *et al.*, 2020) helps to provide information of multiple resources (computing resource, link/path, storage, radio resources) and network services (software as a service, AI as a service, encoding/decoding as a service, function as a service, content as a service) for a distributed edge computing platform. The proposed system supports multi-domain scenarios and scheduling to minimize the computational latency.

- Mobile and Wireless Information Exposure (MoWIE) (ZHANG *et al.*, 2020) provides on demand and periodic network information (*e.g.*, Signal to Inference plus Noise Ratio (SINR), Modulation and Coding Scheme (MCS)) from network to applications and advocate an integrated in-band/out-band transport. Applications can then use such network information to adapt key control knobs such as media codec scheme, encapsulation, and application logical function to ensure users' Quality of Experience (QoE). In addition, the MoWIE's experimental evaluation includes three case studies: (i) MoWIE-assisted TCP Optimization, (ii) MoWIE-assisted video streaming optimization, and (iii) MoWIE-assisted cloud gaming.

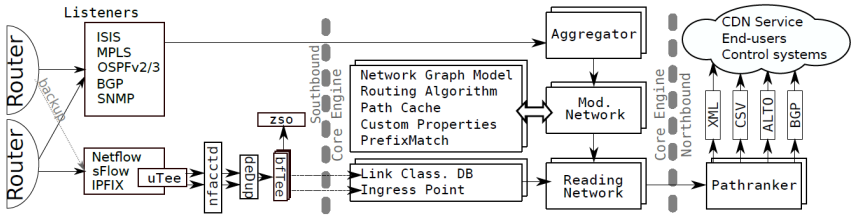
- (XIANG *et al.*, 2020a) formulates a Software-Defined Internetworking (SDI) model to extend the intra-domain SDN capabilities to generic inter-domain SDN. Specifically, SDI exposes a programmable interface to allow applications to control their inter-domain routes, just as a traditional SDN switch exposes Openflow to allow applications to select their next hops.

2.3.3 Real Deployment Examples

- Comcast, a large cable broadband Internet Service Provider (ISP) in the U.S., deployed a P4P-based open framework (YANG *et al.*, 2009). Specifically, P4P iTrackers are used to allow P2P networks to optimize traffic within each ISP while improving P2P download performance for P2P clients. The results of the trial (See Fig 6a) showed that P4P iTrackers can improve the speed of downloads to P2P clients as well as localizing P2P traffic within the Comcast network.

Swarm	Global Avg bps	Change	Comcast Avg bps	Change
Random (Control)	144,045 bps	n/a	254,671 bps	n/a
P4P Fine Grained	162,344 bps	+13%	402,043 bps	+57%
P4P Generic Weight	163,205 bps	+13%	463,782 bps	+82%
P4P Coarse Grained	166,273 bps	+15%	471,218 bps	+85%

(a) P4P-based Framework (Comcast)[4]



(b) Flow Director (Benocs)[5]

Figure 6 – Network-aware applications: Real deployment examples.

- Another much larger deployment is Flow Director (PUJOL *et al.*, 2019), the first-ever ISP-hyper-giant collaboration system. Flow director has three main functions (See Fig 6b): (i) it collects data to determine the state of the ISP's network, then (ii) it computes the best mapping, and finally (iii) it communicates this information with the cooperating hyper-giant in near-real time via multiple protocols (*e.g.*, ALTO, BGP, etc.). Flow Director starts with the ALTO protocol but goes further, designing, building, rolling-out, and operating a large-scale system that enables automated cooperation between one of the largest eyeball networks and a leading hyper-giant.

2.4 MUDED System Design

The preceding discussion exposes huge possibilities of NAI. However, there still exists a major barriers in systematically realizing such a deep integration. Different NAI possibilities are lacking of generic and standard mechanisms for exposure and discovery of NAI possibilities (barriers **B1** and **B2**), and existing realizations are complex point solutions, and could raise scalability concerns (barrier **B3**). In this section, we first present the design of a systematic framework for multi-domain NAI possibilities exposure and discovery (MUDED).

The MUDED architecture aims to deal with barriers **(B1)**, **(B2)**, and **(B3)** both at a multi-operator level and a multi-domain single network operator level. Figure 7 highlights the scope of MUDED by presenting a logical interworking architecture. At the

[4] Figure source: <https://doi.org/10.17487/RFC5632>
[5] Figure source: <https://dl.acm.org/doi/pdf/10.1145/3359989.3365430>

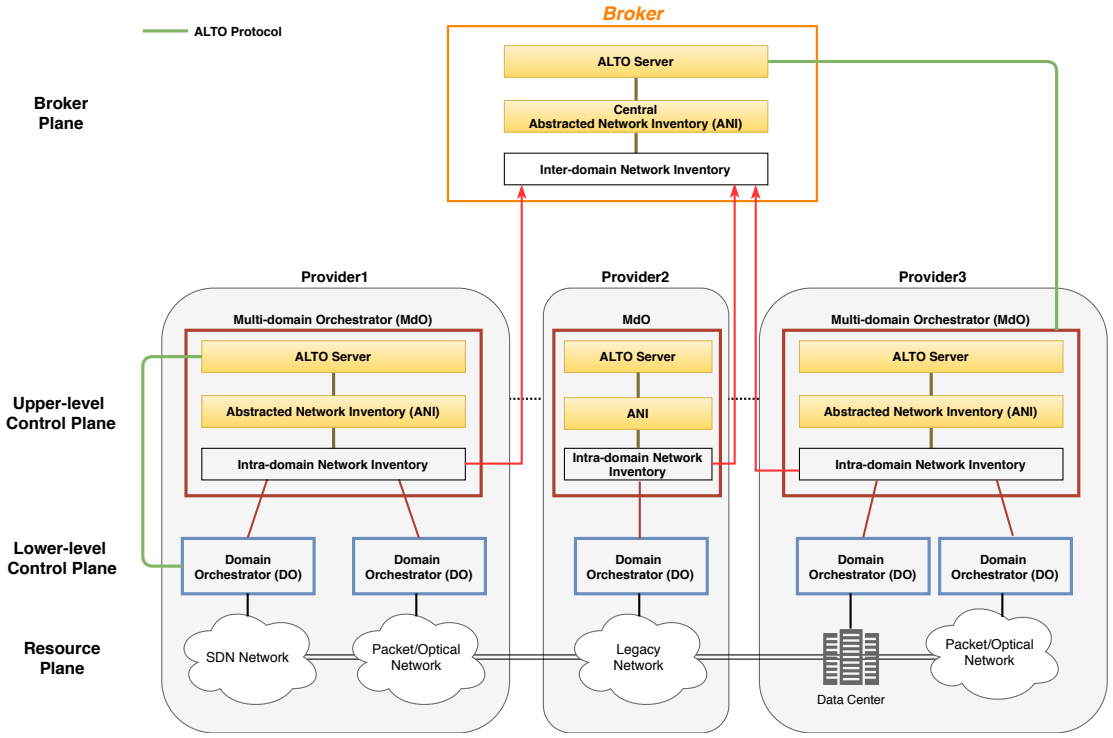


Figure 7 – MUDED: Integrating Networks with Applications through Multi-Domain Exposure and Discovery Mechanisms

lower-level control plane, a Domain Orchestrator (DO) is responsible for various resource domains, featuring physical and virtual, software and hardware components (at the resource plane). A set of DOs is assumed to be managed by a Multi-domain Orchestrator (MDO) at the upper-level control plane. MdOs, in turn, coordinate resource and/or service orchestration at multi-domain level, where multi-domain may refer to multiple DOs or multiple administrative domains.

For multiple DOs (same administrative domain), an MdO receives resource information from each DO and builds an intra-domain network view. The Abstracted Network Inventory (ANI) is the component that implements the new method of aggregating the intra-domain network inventory, providing a summarized view of the resources. From this centralized and aggregated information, the ALTO server component creates and provides abstract maps with a simplified view, yet enough information about network infrastructure. After, a DO (as an ALTO client) sends ALTO service queries to the ALTO server. This server provides aggregated multi-domain information exposure as set ALTO base services defined in (ALIMI *et al.*, 2014), *e.g.*, Network Map, Cost Map and ALTO extension services, *e.g.*, Property Map (ROOME *et al.*, 2020), Multi-Cost Map (RANDRIAMASY *et al.*, 2017), Path Vector (GAO *et al.*, 2020b).

For multiple administrative domains, we are considering a centralized approach where a broker entity (at the broker plane level) is working on the top of the MdOs as a coordinator of the information exchange among the involved administrative domains.

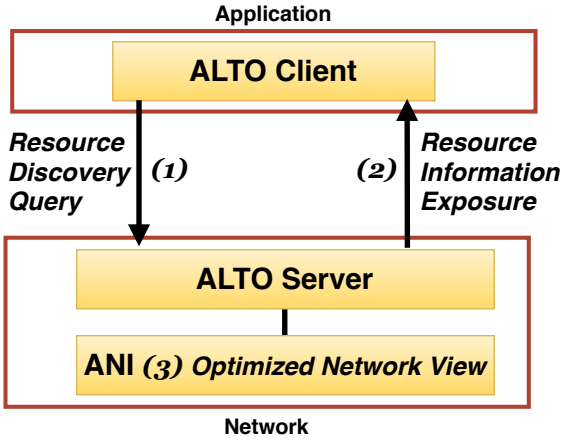


Figure 8 – MUDED Framework: Three key design points.

The broker entity also contains a central ANI component and a central ALTO server component. In this case, each intra-domain network inventory in each domain is in charge of sending the network view of the local domain to the inter-domain network inventory, in order to maintain an updated view of the available resources. From this centralized information, the central ANI component implements the method for network inventory aggregation. The central ALTO server then creates an abstracted inter-domain network view and service set, which can be easily consumable by any MdO (ALTO client) by using ALTO-based REST APIs.

The MUDED framework consists in three novel design points: a declarative resource query language, a resource information exposure, and an abstracted network inventory. In particular, first, MUDED utilizes two declarative languages for applications to express their intents on discovering resources in the network. Second, the MUDED framework uses generic mathematical programming constraints as a unified, compact representation of network information. Third, MUDED incorporates a novel mechanism to create a service-optimized network inventory view over the same network infrastructure.

As shown in Figure 8, for the first and second design points, we are leveraging the maturing of NAI protocols such as ALTO to discover and adequately expose multi-domain resource and topology information (See Chapter 3 for more details). In case of the third design point, we propose the ANI component to deal with the scalability issues (See Chapter 4 for more details).

The following subsections particularize each design point:

2.4.1 Resource Discovery Language

MUDED introduces two declarative languages that allow applications to express flexible resource discovery intents. The first one is to use a representation based on the Network Function Forwarding Graph (NFFG) to specify a set of E2E service requirements

for an ALTO server in order to obtain candidate resources (domains) and candidate paths (PEREZ; ROTHENBERG, 2020; PEREZ; ROTHENBERG, 2018). The second proposed discovery mechanism is to use a SQL-style language for an ALTO client to express its requirement on available resources (XIANG *et al.*, 2020b; LACHOS *et al.*, 2020).

- **NFFG-based Language.** A NFFG is a graph of logical links connecting virtual or physical Network Functions (NFs) nodes for the purpose of describing traffic flow between those NFs (UNIFY D3.2a, 2015). As an ALTO client, each DO/MdO will send ALTO queries to the ALTO server following such NFFG format.

  In order to support the NFFG-based language functionality, MUDED introduces some extensions to the ALTO base protocol. Next, we present a non-normative overview to extend the ALTO Filtered Cost Map (GAO *et al.*, 2020b).

```
1   object {
2     NFFG sg;
3   } ReqFilteredCostMap;
4
5   object {
6     JSONString nfs<1..*>;
7     JSONString saps<1..*>;
8     NextHops sg_links<1..*>;
9     REQs reqs<1..*>;
10  } NFFG;
11
12  object {
13    JSONNumber id;
14    JSONString src-node;
15    JSONString dst-node;
16  } NextHops;
17
18  object {
19    JSONString id;
20    JSONString src-node;
21    JSONString dst-node;
22    JSONNumber sg-path<1..*>;
23  } REQs;
```

  The ALTO Server must allow the request input to include a service graph (sg) with a formatted body as a NFFG object. The NFFG object is based on the formal NFFG specification defined in (UNIFY D3.2a, 2015). Annex A gives the full tree representation of the NFFG model defined in YANG. Specifically, a NFFG object contains: (i) *nfs*: a list of network functions; (ii) *saps*: a list of service access points or endpoints; (iii) *sg_links*: service graph links representing logical connections between network functions, endpoints or both; and (iv) *reqs*: E2E requirements as a list of IDs of service graph links.

Afterward, the ALTO server response for each *sg* link in each E2E requirement must be encoded as an array of arrays, where sub-arrays indicate potential candidate paths calculated as the per-domain topological distance corresponding to the amount of traversing domains. Moreover, the ALTO server must provide path vector information along with the associated ALTO Property Map information (*e.g.*, entry points of the corresponding foreign domains), in the same body of the response. Chapter 5 (MUDED Use Case) and Annex B give more examples of the ALTO filtered cost map query and the corresponding response.

- **SQL-style Language.** It uses a resource-filtering design, which allows applications to define predicates on packet spaces (*i.e.*, different sets of flows), and predicates on resources (*i.e.*, particular resource attributes that applications are interested in discovering). Leveraging the equivalence between relational algebra and first-order logic, the language uses SQL-style semantics, which are familiar to both application and network engineers.

  Specifically, the resource discovery intent is expressed as (i) a sequence of *flow specification statements*; (ii) a sequence of *resource requirement statements*; and (iii) a *query statement*. In the flow specification statement is defined a flow or a set of flows about which the application (DO/MdO) wants to get the resource information. With the resource requirement statement, applications express the basic requirements on resources provided by the network for a set of flows. Finally, the application uses a SQL-style query to assemble the previous flow specification and resource requirements and express the resource discovery intent.

  Figure 9 gives an example to illustrate how resource discovery intents are specified using the SQL-style language. Consider a case where an application has two flows to transmit and wants to find the bandwidth the network can provide for those two flows. The first flow is an FTP flow from source IP 10.0.0.1 to destination IP 10.0.0.2, and the second flow is a HTTP flow from source IP 10.0.0.3 to destination IP 10.0.0.4 (Line 1-4). The application then defines a flow set containing those two flows (Line 5). In addition to the basic reachability, the application specifies its requirements for those two flows such as the bandwidth of the FTP flow is to be at least 100 Mbps (Line 6), and the HTTP flow passes a firewall middlebox before reaching the destination (Line 7). Finally, the application specifies its resource discovery intent for those two flows. In the example is to find the bandwidth the network can provide for flow1 and flow2 subject to the application's requirements (Line 8-9).

  In addition to the waypoint (*{"FW"} subset flow_1.route}*) and QoS requirements (*flow_1.bandwidth* $\geq$ *100 Mbps*), the SQL-style language can also express other common resource requirement predicates such as bi-direction symmetry (*flow.links == inv(flow_1.links)*), node disjointness (*size(flow_1.nodes intersect flow_2.nodes)==0*),

```
flow_1: {src_ip = 10.0.0.1 and dst_ip = 10.0.0.2
        and dst_port = 20};
flow_2: {src_ip = 10.0.0.3 and dst_ip = 10.0.0.4
        and dst_port = 80};
flow_set: {flow_1, flow_2};
req_1: flow_1.bandwidth >= 100 Mbps;
req_2: {"FW"} subset flow_2.route;
select bandwidth from flow_set
        where req_1 and req_2;
```

Figure 9 – An example resource discovery query.

link disjointness (*size(flow_1.links intersect flow_2.links)==0*), and blacklist of device (*!({"FW"} subset flow_1.nodes)*).

2.4.2 Resource Information Exposure

In MUDED, when the network needs to expose the information for a set of flows to applications, it uses mathematical programming constraints to capture the resource availability and sharing information of these flows, providing a unified resource representation.

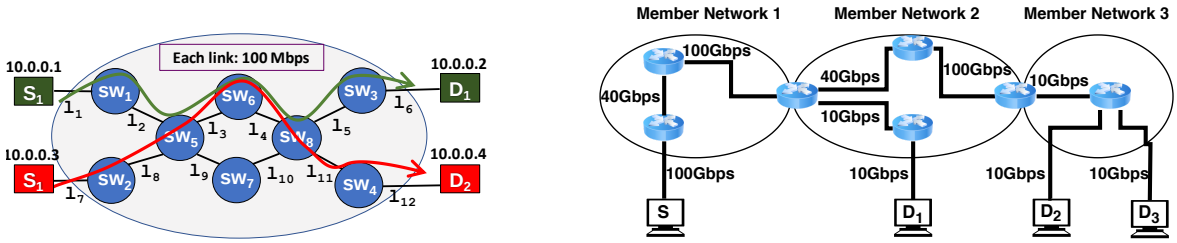
- **Basic Idea.** Suppose MUDED receives the resource discovery request of a set of flow F. For each flow $f_j \in F$, we use x_j to denote an available resource (*e.g.*, bandwidth) the application can reserve for this flow. Upon receiving this request, MUDED first checks the routes – computed by the underlying routing protocol – for each flow f_j. Then all the links are enumerated. For each link l_u, it generates a linear inequality: $\sum x_j \leq l_u.resource, \forall f_j$ that uses link l_u in its route.

  To illustrate this formulation, consider the network topology in Figure 10a, where an application wants to reserve bandwidth for two flows $f_1 : (S_1, D_1)$ and $f_2 : (S_2, D_2)$. The routes for the two flows share common links, *i.e.*, l_3 and l_4, hence it is infeasible for both circuits to each reserve a 100 Mbps bandwidth. Therefore, the MUDED framework will generate the following set of algebraic expressions (*i.e.*, linear inequalities) (HEORHIADI *et al.*, 2016):

Linear Inequalities	**ID**
$x_1 \leq 100, \forall l_u \in \{l_1, l_2, l_5, l_6\}$	Π_a
$x_2 \leq 100, \forall l_u \in \{l_7, l_8, l_{11}, l_{12}\}$	Π_b
$x_1 + x_2 \leq 100, \forall l_u \in \{l_3, l_4\}$	Π_c

Table 3 – Resource abstraction for the reservation request from Figure 10a

  Where x_1 and x_2 represent the available bandwidth that can be reserved for (S_1, D_1), and (S_2, D_2), respectively. Each linear inequality represents a constraint on the



(a) (b)

Figure 10 – An example where an application tries to discover information of two and three flows (a) from one single network, and (b) from a collaboration network composed of three member networks, respectively.

reservable bandwidths over different shared resources by the two flows. For example, Π_c indicates that both flows share a common resource and that the sum of their bandwidths can not exceed 100 Mbps.

- **Removing Redundant Linear Inequalities.** Taking a deeper look at the set of previous linear inequalities, one can conclude that inequalities of Π_a and Π_b can be implicitly derived from that of Π_c. Thus, these inequalities are considered redundant. The problem of finding redundant linear constraints has been widely studied (PAULRAJ *et al.*, 2010). Specifically, redundant linear inequalities are removed via a polynomial-time, optimal algorithm (KARMARKAR, 1984). In our example, the compressed set will only contain one inequality: $\Pi_c : x_1 + x_2 \leq 100, \forall l_u \in \{l_3, l_4\}$.

- **From Single Domain to Multiple Domains.** To illustrate the basic aggregation abstraction from a single network to multiple networks, consider a collaboration network composed of three member networks, as shown in Figure 10b. A user wants to reserve bandwidth for three circuits, from source host S to destination hosts D_1 , D_2, and D_3.

  The resource abstraction captures the constraints from all networks using the set of linear inequalities, as depicted in Table 4. Specifically, the variables f_1, f_2, f_3

Table 4 – Resource abstraction for the reservation request from Figure 10b

Network	Linear Inequalities	ID
Member Network 1	$f_1 + f_2 + f_3 \leq 100$	Π_{11}
	$f_1 + f_2 + f_3 \leq 40$	Π_{12}
	$f_1 + f_2 + f_3 \leq 100$	Π_{13}
Member Network 2	$f_2 + f_3 \leq 40, f_1 \leq 10$	Π_{21}
	$f_2 + f_3 \leq 100, f_1 \leq 10$	Π_{22}
Member Network 3	$f_2 + f_3 \leq 10$	Π_{31}
	$f_2 \leq 10$	Π_{32}
	$f_3 \leq 10$	Π_{33}

represent the available bandwidth that can be reserved for (S, D_1), (S, D_2), and (S, D_3), respectively. Each linear inequality represents a constraint on the reservable bandwidths over different shared resources by the three circuits. For example, the inequality Π_{11} indicates that all three circuits share a common resource and that the sum of their bandwidths can not exceed 100 Gbps.

After removing the redundant inequalities of each member network, the resource abstraction of each member network is:

Network	**Linear Inequalities**	**ID**
Member Network 1	$f_1 + f_2 + f_3 \leq 40$	Π_{12}
Member Network 2	$f_2 + f_3 \leq 40, f_1 \leq 10$	Π_{21}
Member Network 3	$f_2 + f_3 \leq 10$	Π_{31}

Table 5 – Resource abstraction for the reservation request from Figure 10b after removing the redundant inequalities.

Although each domain may already conduct redundancy optimization, there can be cross-domain redundancy. For example, the constraint Π_{31} at member network 3 ($f_2 + f_3 \leq 10$) can eliminate those at member network 1 ($f_1 + f_2 + f_3 \leq 40$) and member network 2 ($f_2 + f_3 \leq 40$). Using a classic compression algorithm (TELGEN, 1983), we can remove this cross domain redundancy. Therefore, the compressed multi-domain set of linear inequalities will contain:

$$f_1 \leq 10, f_2 + f_3 \leq 10. \tag{2.1}$$

2.4.3 Abstracted Network Inventory (ANI)

MUDED also implements a novel mechanism to proactively construct service-optimized network views over the same network infrastructure. This novel mechanisms is incorporated into the Abstracted Network Inventory (ANI) component. The ANI receives two inputs (See Fig. 11): (i) services requirements from a catalog, and (ii) a network inventory representation. Both inputs guide the right level of abstraction to generate a Logical Network Inventory (LNI). Each LNI is optimized to a service in terms of its requirements such, as CPU, memory, latency, etc. Every new service in a catalog triggers the creation of another LNI that will be part of the optimized network inventory.

We illustrate the LNI generation with an example given in Figure 12. The left side of this figure shows two network services, where the numbers in rectangles represent requested CPU capacity, and the numbers near the links represent required bandwidth capacity. The network service 1 (*ns1*) connects two VNFs (*nf1* and *nf2*) with 45 units of bandwidth on the edge between them. The network service 2 (*ns2*) requires the bandwidth 5 over the edge (*nf2*, *nf4*) and 15 units over the edge (*nf3*, *nf5*), and the CPU resources

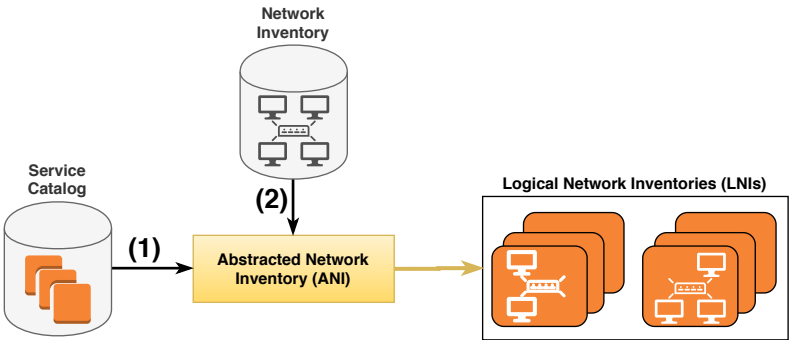


Figure 11 – ANI: Logical Network Inventory (LNI) generation

60, 70, 65 at VNFs nodes, *nf3*, *nf4*, and *nf5*, respectively. Figure 12 (right side) also depicts a network inventory. The number near the links is the available bandwidth, and the numbers in rectangles represent the available CPU resources at the vertices. Once the process of determining an LNI is performed, the network inventory contains two logical graphs $LNI_i = \{L_{ns1}, L_{ns2}\}$, where each graph has a dedicated subset of vertices and edges, which represent an optimized network view to the requirements of each network service. Note that also a vertex (v_c) is overlapped since $V(L_{ns1}) \cap V(L_{ns2}) = \{v_c\}$.

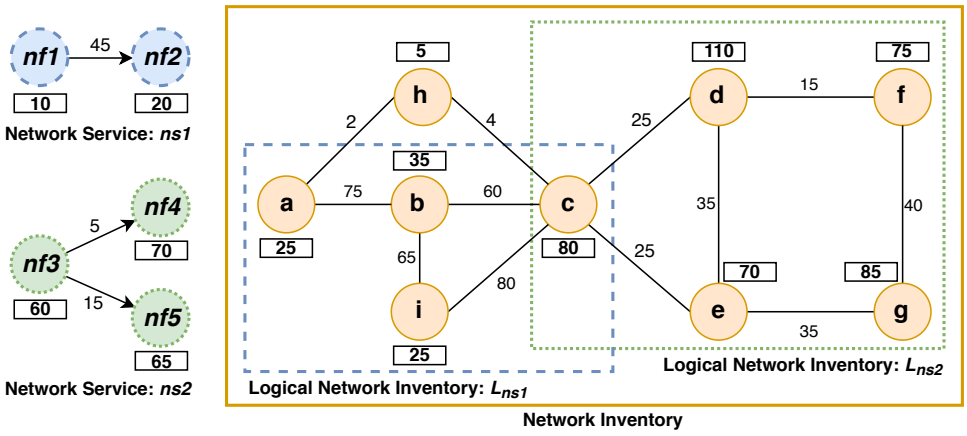


Figure 12 – An example of Network Service & Network Inventory & Logical Network Inventory (LNI).

2.5 Concluding Remarks

The collaboration between networks and applications brings benefits to both parties, yet realizing it is non-trivial. In this chapter, we review huge possibilities in designing and implementing NAI by application-aware networking and network-aware applications. We design MUDED, an NAI possibilities discovery and exposure framework to address the key barriers of systematically realizing NAI.

Future research to extend the studies in this chapter should understand the shortcomings contained herein. Among the most important improvements that can be made

on this study, we highlight:

- Although Chapter 3 will provide more information about the ALTO, the ALTO-based specification of a resource discovery language and resource information exposure is not explained sufficiently. However, while not explained in this chapter, this specification is under discussion with other members into the IETF ALTO WG for an eventual adoption in the next ALTO re-chartering.

- There is no discussion on where MUDED is to be used (*i.e.*, datacenter, Cooperate, WAN) and for what (*e.g.*, resource trading, fast connection setup, CDN optimization). Future activities will detail information about potential embodiment scenarios, such as network resource reservation systems. Such applications are looking for optimal configurations in data center network topologies (*e.g.*, fat-tree) where a large number of paths are designed between any end-host pairs to achieve full bandwidth.

- MUDED may rise to privacy and security issues. Therefore, it is necessary a systematic review of the system design to ensure that queries to the network can provide enough information without compromising the privacy of clients/applications. As emphasized in the next chapter, mechanisms to ensure that information is transformed and aggregated will be explored to deal with the network information exposure issues.

3 Multi-domain Information Exposure & Discovery using ALTO

3.1 Introduction

Many multi-domain use cases are emerging with the development of new technologies, such as software-defined networking (SDN), network function virtualization (NFV), and 5G. Examples of such use cases include multi-domain, collaborative data sciences (CMS, 2008; LCLS, 2020; LHC, 2020; SKA, 2020), multi-domain service function chaining (SFC) (ALLIANCE, 2015; HALPERN; PIGNATARO, 2015; KATSALIS *et al.*, 2016; ETSI, 2020), and multi-domain SDN (XIANG *et al.*, 2018; GUPTA *et al.*, 2015; MARQUES *et al.*, 2009). Such use cases can benefit substantially from the exposure of network information, with which users can perform application-layer resource optimization to improve the performance.

The Application-Layer Traffic Optimization (ALTO) protocol (ALIMI *et al.*, 2014) already introduces basic mechanisms (*e.g.*, modularity, dependency) and abstractions (*e.g.*, map services) for applications to take optimized actions based on network information. However, exposing network information to support multi-domain use cases places additional requirements that existing solutions such as the current ALTO design do not satisfy. First, abstractions that aggregate multiple networks into a single, virtual network are required to simplify the application-layer optimization conducted by end-users. Second, such abstractions need to provide a unified representation of multiple resources (*e.g.*, networking, computation, and storage) in multiple networks.

This chapter reviews standardization efforts and research project solutions in the context of multi-domain scenarios (Section 3.2). Then, we introduce the IETF ALTO protocol, including the key components, architecture, its evolution and the current rechartering discussion towards the support of new use cases (Section 3.3). Section 3.4 presents several important multi-domain use cases that can benefit substantially from network information discovery and exposure using ALTO. Next, as a main contribution, this chapter elaborates the key design requirements of network information exposure to support those use cases (Section 3.5). Finally, before concluding (Section 3.7), another main contribution of this chapter is to summarize novel mechanisms and abstractions based on recent research to improve the ALTO framework in the multi-domain settings (Section 3.6).

3.2 Multi-domain Approach: Context

Different standardization efforts (*e.g.*, IETF, MEF, ETSI, NGMN) and research projects activities (*e.g.*, 5GEx (BERNARDOS *et al.*, 2016), T-NOVA (T-NOVA, 2014), 5G-Transformer (5G-TRANSFORMER, 2017), MATILDA (MATILDA, 2017), VITAL (VITAL, 2015)) have been focused on multi-domain network service chaining. Standardization is essential to provide recommendations to create interoperable architectures with standardized protocols, and solutions (being developed by different projects) are addressing a diverse range of requirements to provide network services provided using multiple domains.

3.2.1 Standardization Activities

- **IETF:** Service chaining that span domains owned by single or multiple administrative entities are being discussed in the IETF. The Hierarchical Service Function Chaining (hSFC) (DOLSON *et al.*, 2018), for example, defines an architecture to deploy SFC in large networks. This RFC proposes to decompose the network into smaller domains (domains under the control of a single organization). Another proposed initiative is (LI *et al.*, 2018) that describes SFC crossing different domains owned by various organizations (*e.g.*, ISPs) or by a single organization with administration partitions. The proposed architecture uses an SFC eXchange Platform (SXP) to collect and exchange information (topology, service states, policies, etc.) between different organizations and it works both in centralized (Multiple SFC domains connected by a logical SXP) and distributed (SXP server as a broker) environments.

  Another initiative is the Network Function Virtualization Research Group (NFVRG). The draft "Multi-domain Network Virtualization" (BERNARDOS *et al.*, 2018) envisions a complete end-to-end logical network as stitching services offered by multiple domains from multiple providers. It also points to the need for creating solutions that enable the exchange of relevant information (resources and topologies) across different providers.

- **ETSI:** The European Telecommunication Standards Institute (ETSI) for Network Functions Virtualization (ISG NFV) is paving the way toward viable architectural options supporting the efficient placement of functions in different administrative domains. More specifically, the document (European Telecommunication Standards Institute, 2018) reports different NFV MANO architectural approaches with use cases related to network services provided using multiple administrative domains. Besides, it gives a non-exhaustive list of key information to be exchanged between administrative domains (monitoring parameters, topology view, resource capabili-

ties, etc.) and recommendations related to security to permit the correct and proper operation of the final service.

- **MEF:** With its work on the Service Operations Specification MEF 55 (Metro Ethernet Forum, 2019), MEF has defined a reference architecture and framework for describing functional management entities (and interfaces between them) needed to support Lifecycle Service Orchestration (LSO). This LSO architecture enables automated management and control of E2E connectivity services across multiple operator networks. Automated service management includes fulfillment, control, performance, assurance, usage, security, analytics, and policy capabilities that make it possible, for example, expanding the footprint of service providers to interact with potentially several operators to manage and control the access portions of E2E services.

- **NGMN.** Next Generation Mobile Networks (NGMN) in (NGMN Alliance, 2017) provides key requirements and high-level architecture principles of Network and Service Management including Orchestration for 5G. Based on a series of user stories (*e.g.*, slice creation, real-time provisioning, 5G end-to-end service management), the document establishes a common set of requirements such as self-healing, scalability, testing and automation, analysis, and modeling.

  The document (NGMN Alliance, 2018) defines the requirements necessary that characterize an End-to-End framework. It considers three possible orchestration architecture: (i) Vertical (Hierarchical), which involves processes that ranges from the business level to lower level resource instantiations, (ii) Federated, when the services are provisioned over multiple operators' networks or over various domains, and (iii) Hybrid (Federated and Vertical), that include characteristics of both federated and vertical orchestration.

3.2.2 Research projects

Several projects include an architectural model integrating NFV management with SDN control capabilities to address the challenges towards flexible, dynamic, cost-effective, and on-demand service chaining (SGAMBELLURI *et al.*, 2017; VITAL, 2015; T-NOVA, 2014; 5G-TRANSFORMER, 2017; MATILDA, 2017):

- The 5G Exchange (5GEx) project (BERNARDOS *et al.*, 2016) aims to integrate multiple administrations and technologies through the collaboration between operators in the context of emerging 5G networking. Formed by a consortium of vendors, operators, and universities, 5GEx allows end-to-end network and service elements to mix in multi-vendor, heterogeneous technology, and resource environments. In

such a way, the project targets business relationships among administrative domains, including possible external service providers without physical infrastructure resources.

- (VITAL, 2015; T-NOVA, 2014) follow a centralized approach where each domain advertises its capabilities to a federation layer, which will act as a broker.

  The H2020 VITAL project addresses the integration of Terrestrial and Satellite networks through the applicability of two key technologies such as SDN and NFV. The main VITAL outcomes are (i) the virtualization and abstraction of satellite network functions and (ii) supporting Multi-domain service/resource orchestration capabilities for a hybrid combination of satellite and terrestrial networks (PROJECT, 2016).

  The focus of the FP7 T-NOVA project is to design and implement an integrated management architecture for the automated provision, configuration, monitoring and optimization of network connectivity and Network Functions as a Service (NFaaS). Such architecture includes: (i) a micro-service based on NFV orchestration platform– called TeNOR (RIERA *et al.*, 2016), (ii) an infrastructure visualization and management environment, and (iii) an NFV Marketplace where a set of network services and functions can be created and published by service providers and, subsequently, acquired and instantiated on-demand by customers or others providers.

- The 5G-Transformer project (5G-TRANSFORMER, 2017) is defining flexible slicing and federation of transport networking and computing resources across multiple domains. This project consists of a group of 18 companies including mobile operators, vendors, and universities. The objective of the project is to transform current's mobile transport network into a Mobile Transport and Computing Platform (MTP) based on SDN, NFV, orchestration, and analytics, which brings the Network Slicing paradigm into mobile transport networks.

  Likewise, 5G-Transformer defines three new components to the proposed architecture: (i) *vertical slicer* as a logical entry point to create network slices, (ii) *Service Orchestrator* for end-to-end service orchestration and computing resources, and (iii) *Mobile Transport and Computing Platform* for integrating fronthaul and backhaul networks. The Service Orchestrator is the main decision point of the system. It interacts with others Service Orchestrators (SOs) to the end-to-end service (de)composition of virtual resources and orchestrates the resources even across multiple administrative domains.

- The MATILDA project (MATILDA, 2017) is to design and develop a holistic framework that supports the interconnection among the development of 5G end-to-end applications, the creation of the required computational and networking infrastructure (using an application-aware network slice), and the networking mechanisms ac-

tivation for the support of the industry vertical applications. The MATILDA framework includes a multi-site virtual infrastructure manager supporting the multi-site management of the allocated resources (per network slice), along with a multi-site NFVO supporting the lifecycle management of the network functions.

3.3 ALTO Background

Applications can benefit from network information exposure to make them more flexible in terms of rate adaptation, transmission time, server/path selection, among others. The IETF Application Layer Traffic Optimization (ALTO) protocol provides network information that applications use for modifying network resource consumption patterns while improving their performance.

The ALTO protocol (ALIMI *et al.*, 2014), published in 2014, specifies an information-publishing interface to provide abstract network information between an ALTO client and an ALTO Server. The network information is conveyed in the form of abstract Map Services (Network Map and Cost Map) by an ALTO server (see Fig. 13). A Network Map divides all endpoints (*e.g.*, IPv4/IPv6 addresses or prefixes) in Provider-Defined Identifiers (PIDs) and a Cost Map allows ALTO clients (*i.e.*, applications) to determine preferences between each pair of PIDs (PEREZ *et al.*, 2016). Since then, ALTO has been considered in different use case applications such as P2P, Content Delivery Networks (CDNs), and data center applications.

3.3.1 Key Concepts

The information presented in this subsection is basically referenced by the RFCs 7285 (ALIMI *et al.*, 2014), 5693 (SEEDORF; BURGER, 2009), and 7971 (STIEMERLING *et al.*, 2016).

- **ALTO Server:** An ALTO server is a logical entity that provides Rest-based APIs to consult ALTO information services. The ALTO specification allows that at least three entities can operate as an ALTO server:
  - **Network operators:** An entity that has a detailed knowledge of its network topology information, such as Network Service Providers (NSPs). Usually, the source of the network information and the ALTO server are part of the same organization.
  - **Third parties:** This entity is separate from network operators; however, it could be able to retrieve network information from arrangements with network operators. For example, CDNs.

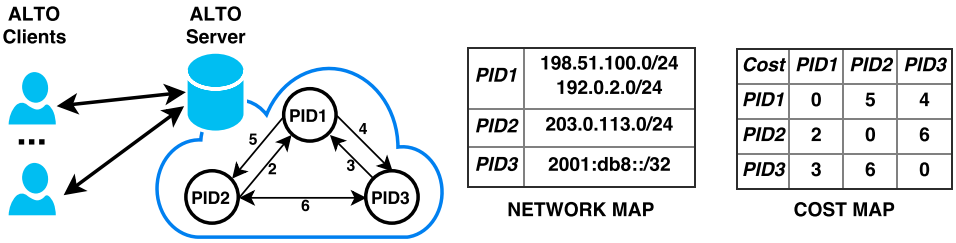


Figure 13 – Concept of ALTO and two main services: Network Map and Cost Map.

  - **User communities:** Entities not associated with network providers, who may obtain network information from public data, running distributed measurement for estimating a particular topology.

- **ALTO Client:** An ALTO client is a logical entity sending ALTO queries to gather guiding information from the ALTO server. Depending on the application architecture, an ALTO client can be situated as:

  - **Resource consumer:** When the ALTO client is located on the actual host that runs the application. For example, a P2P file sharing application trying to connect to other destination peers without using a Tracker, such as edonkey (HECKMANN; BOCK, 2002).
  - **Resource directory:** A BitTorrent Tracker[1] would be an example of this type of ALTO client. The Tracker acts as an ALTO client and resource consumers (peers) ask for a list of destination peers that can provide the desired resource.

- **ALTO Client Protocol:** It is used for sending queries from an ALTO client to an ALTO server as well as transmit the corresponding ALTO replies from the ALTO server to the ALTO client.

3.3.2 ALTO Evolution

Before the ALTO base protocol, the ALTO problem statement (SEEDORF; BURGER, 2009) and requirements (KIESEL *et al.*, 2012) were published in 2009 (RFC5693) and 2012 (RFC6708), respectively. Afterward, several extensions have been standardized, such as deployment considerations (RFC7971) (STIEMERLING *et al.*, 2016), a multi-cost map to retrieve several cost metrics in a single query/response transaction (RFC8189) (RANDRIAMASY *et al.*, 2017), and server discovery (RFC7286) (SONG *et al.*, 2014) and cross-domain server discovery (RFC8686) (KIESEL; STIEMERLING, 2020) to identify a topologically nearby ALTO server or ALTO servers outside of a network domain, respectively.

[1] <https://wiki.theory.org/BitTorrentSpecification#Tracker_HTTP.2FHTTPS_Protocol>

In addition, the current ALTO Working Group (WG) charter (2014) is very close to finalizing its milestones with new extensions:

- **ALTO Cost Calendar** (RANDRIAMASY *et al.*, 2020): It provides ALTO cost values where each value corresponds to a specific time interval (currently in the RFC editor queue).
- **ALTO Incremental Updates Using Server-Sent Events (SSE)** (ROOME; YANG, 2020): It allows an ALTO Server to expose ALTO cost values to specified time intervals (currently in the RFC editor queue).
- **CDN Interconnection (CDNI) Advertisement using ALTO:** (SEEDORF *et al.*, 2020): It defines a new ALTO service to provide CDNI footprint and capabilities advertisement interface (FCI) information (passed WG last call).
- **ALTO Path Vector Extension:** (GAO *et al.*, 2020b): It introduces a new cost type to provide more detailed routing information using Abstract Network Elements (ANEs).
- **ALTO Performance Cost Metrics:** (WU *et al.*, 2020): It presents a set of new network performance metrics, including network delay, jitter, packet loss rate, hop count, and bandwidth.
- **Unified properties for the ALTO protocol:** (ROOME *et al.*, 2020): It generalizes the concept of ALTO endpoint properties by presenting those as "property maps".

3.3.3 ALTO Architecture

ALTO already provides a generic architecture to expose network information for applications to improve their performance. Figure 14 presents a high-level overview of key ALTO mechanisms and abstractions.

In particular, ALTO introduces generic mechanisms such as:

- Modularity and flexibility through an explicit division of ALTO network information into (network) information resources.
- An information resource directory (IRD) providing a list of available information resources in an ALTO server.
- Information consistency (tag, dependency, multi-info resources [ALTO-MULTIPART]) to specify a dependency among different information resources.

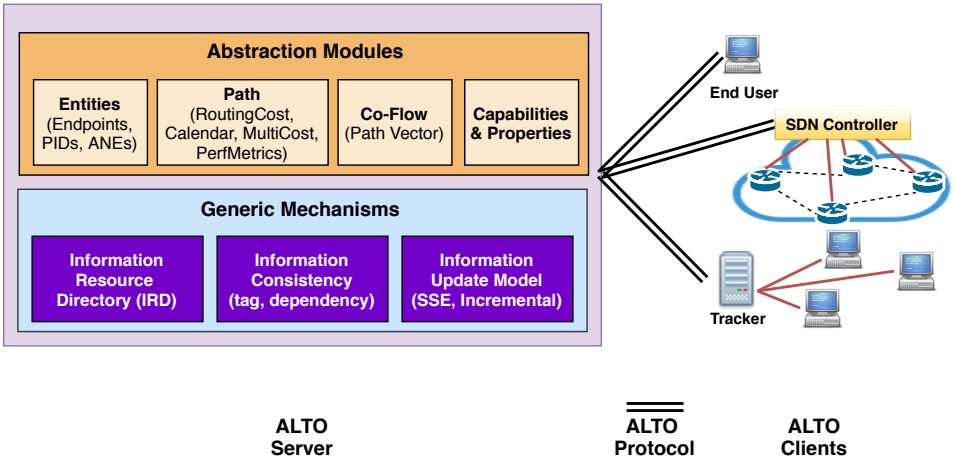


Figure 14 – High Level ALTO Architecture.

- A generic framework with Server-Sent Events (SSEs) [ALTO-SSE] to perform stream-control, push, incremental update of information resources.

ALTO also introduces generic abstractions, such as:

- Entities such as endpoints, aggregation of endpoints (PID), ANEs. A network/property map consists of a set of entities.
- Paths traversing a network/property map from (some types) of a source entity to a destination entity. Each Path has properties called cost metrics (*e.g.*, routingcost, PerfMetrics, MultiCost, CostCalendar). The ECS queries endpoint to endpoint and the cost map queries aggregation to aggregation.
- A set of paths can form a co-flow (path vector), with shared ANEs cross the co-flows.
- Each entity supports inheritance and can have capabilities and a set of properties.

Another generic concept introduced is the filter so that information resources can be filtered (*e.g.*, filtered network map, filtered cost map). Besides, each individual information resource is provided as a RESTful service with a very simple, but well-working grammar (essentially JSON grammar (BRAY, 2017)).

3.3.4 ALTO Re-Chartering

Currently, technical discussions are taking place on re-chartering the WG to support the emerging new uses of ALTO. Following a use-case driven approach, five groups of ALTO service extensions are being sought: (i) extensions for cellular networks; (ii) extensions for Multi-access Edge Computing (MEC); (iii) extensions for huge data; (iv) extensions for inter-domain; and (v) ALTO optimizations.

Each group includes a list of personal drafts[2] proposing extensions for specific use cases such as cellular information exposure, multi-domain network information exposure, determine service edge, delivering functions over edge computing, predictive throughput for TCP reactive flows, generic query language, in-bound/out-bound network information exposure, HTTP/2/3 support, multi-part message, among others.

3.4 Motivating Multi-domain Use Cases & ALTO Benefits

A common setting in many emerging applications (*e.g.*, data-intensive science applications, flexible inter-domain routing, multi-domain service function chaining) is that the traffic from a source to a destination traverses multiple network domains. Such applications can benefit substantially from network information exposure to make application-layer resource optimization and improve their performance.

Next, we review several important multi-domain use cases that can benefit substantially from network information exposure using ALTO:

3.4.1 Multi-domain, collaborative data sciences

Many of today's premier science experiments, such as the Large Hadron Collider (LHC)[3] and the Square Kilometre Array (SKA)[4], rely on finely-tuned workflows that coordinate geographically distributed resources (*e.g.*, instrument, compute, storage) to enable scientific discoveries. One example is the movement of LHC data from Tier 0 (*i.e.*, the data center at the European Organization for Nuclear Research, known as CERN) to Tier 1 (*i.e.*, national laboratories) storage sites around the world. Another example is that the Fermilab is experimenting with moving the exascale LHC workflow to Amazon EC2 for more computation power (HOLZMAN *et al.*, 2017).

The key to supporting these distributed workflows is the ability to orchestrate multiple resources across multiple network domains to facilitate predictable workflow performance (*e.g.*, available bandwidth, packet loss rate). As such, multi-domain network information exposure is a cornerstone to enable this ability.

- **How can multi-domain resource orchestration benefit from ALTO?**

  One key design challenge for multi-domain resource orchestration is its resource information model. Existing design options such as resource graph and ClassAds are inadequate because they cannot simultaneously (i) allow member networks to

[2] https://www.ietf.org/proceedings/108/slides/slides-108-alto-one-slide-alto-extensions-for-recharter-discussion-03

[3] https://home.cern/topics/large-hadron-collider

[4] https://www.skatelescope.org/

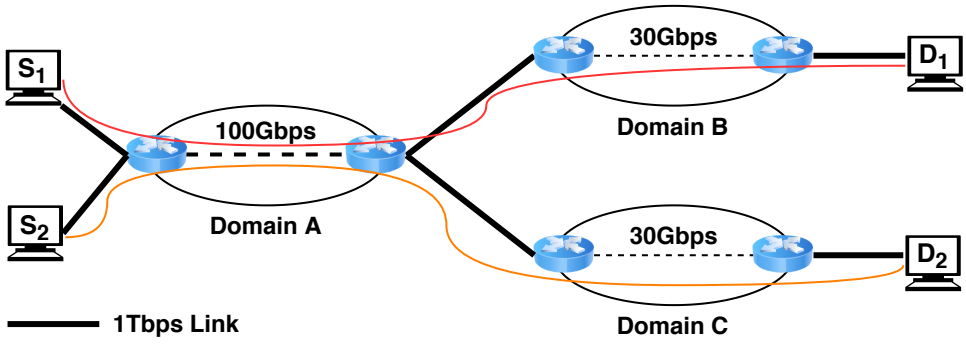


Figure 15 – Multi-domain resource orchestration.

provide accurate information on different types of resource, (ii) avoid the exposure of private information of member networks such as topology, and (iii) allow data analytics jobs to describe their requirements of different types of resources accurately.

In contrast, ALTO is well suited for providing a generic representation that (i) allows different types of data analytics jobs to accurately describe their resource requirements, and (ii) allows member networks to provide accurate information on different types of resources they own and at the same time maintain their privacies.

- **Example**

  Consider an example of three member networks in Figure 15, where S_1 and S_2 are storage endpoints, and D_1 and D_2 are computation endpoints. Assume a data analytics job is composed of two parallel flows F_1 and F_2. F_1 needs dataset X as input, and F_2 needs dataset Y as input.

  Using the ALTO endpoint property service, an ALTO client in the resource orchestrator can discover that D_1 satisfies the computing requirements of F_1, and D_2 satisfies the computing requirements of F_2. Hence there are only two candidate endpoint pairs: $F_1 : (S_1, D_1)$ and $F_2 : (S_2, D_2)$.

  Afterward, the ALTO client can retrieve the bandwidth availability/sharing information of flows F_1 and F_2 using the ALTO path vector extension. With such information, the resource orchestrator can make the optimal resource orchestration decision to reserve 30 Gbps bandwidth for task F_1, and 30 Gbps bandwidth for task F_2.

3.4.2 Multi-domain Service Function Chaining (SFC)

This use case refers to building E2E services by composing multiple service functions (SFs) in an abstract sequence across multiple network domains (SUN *et al.*, 2018). It is identified as an important value-added service in 5G (KATSALIS *et al.*, 2016; ETSI, 2020). Exposing multi-domain network and resource information (*e.g.*, link bandwidth,

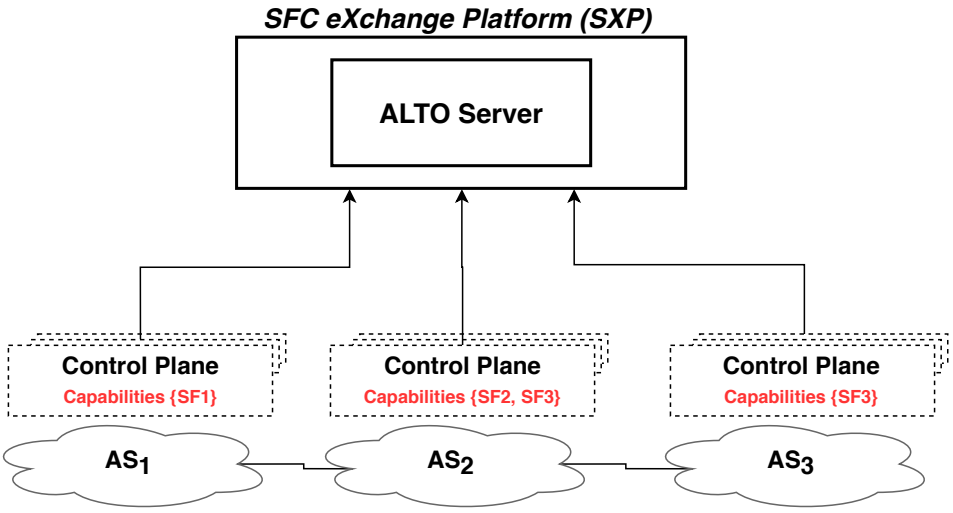


Figure 16 – ALTO as part of the SFC eXchange Platform.

CPU utilization) can substantially improve the efficiency of constructing and managing such SFCs.

- **How can multi-domain SFC benefit from ALTO?**

  A "dialogue" between potential domains that provide multi-domain SFC could be beneficial for more efficient use of resources and increasing the SFC performance. However, constrained knowledge of the network services and underlying network topology based only on localized views from the point of view of a single domain limits the potential and scope for multi-domain SFC.

  ALTO (and customized ALTO extensions) can be used to offer aggregated/abstracted views on various types of information, including domain-level topology, storage resources, computation resources, networking resources, and SF capabilities. This generic representation contributes to a more simple and scalable solution for resource and service discovery in multi-domain, multi-technology environments.

- **Example**

  Figure 16 shows a SFC eXchange Platform (SXP), connecting three different domains (AS_1, AS_2, AS_3). A SXP is a logical entity to make possible the negotiation between different domains, and it could be deployed, for example, in future Software-defined IXP (as a trusted third-party platform) (LI *et al.*, 2020a).

  In this scenario, each domain provides different SFs: $AS_1 = \{SF_1\}$; $AS_2 = \{SF_2, SF_3\}$; and $AS_3 = \{SF_3\}$. The SXP also includes an ALTO server component to provide abstract topology, resource, and service information for the high-level control plane in each domain (PEREZ *et al.*, 2018).

  The ALTO Property Map Service (ROOME *et al.*, 2020) can provide a clear global view of the resource information offered by other domains. This information allows

discovering which candidate domains may be contacted to deliver the remaining requirements of a requested end-to-end service deployment. In our example, the Property Map (see Table 6) includes a property value grouped by AS. This value contains the supported SFs. Additional properties could be considered, such as resource availability (*e.g.*, CPUs, Memory, and Storage), orchestrator entry points, etc.

Domain	**Capabilities**	**Entry Point**	**CPU**	**Memory**	**Storage**	**...**
AS_1	$\{SF_1\}$	http://...	...	...	...	...
AS_2	$\{SF_2, SF_3\}$	http://...	...	...	...	...
AS_3	$\{SF_3\}$	http://...	...	...	...	...

Table 6 – ALTO Property Map Example

Once the candidate domains are discovered, it is necessary to compute multi-domain SF paths to select the SF location from those different candidate domains. The connectivity information among discovered domains can be retrieved by an ALTO Cost Map service. In our example, the Cost Map defines a path vector as an array of ASes, representing the AS-level topological distance for a given SFC request. Table 7 shows a brief example of a service request and its multi-domain SF path response containing a list of potential domains to be traversed to deliver such service.

SFC Request	**Multi-domain Service Function Path(s)**
$\{\text{SF}_1 \rightarrow SF_2 \rightarrow SF_3\}$	1:$\{AS_1 : SF_1 \rightarrow AS_2 : SF_2 \rightarrow AS_2 : SF_3\}$ 2:$\{AS_1 : SF_1 \rightarrow AS_2 : SF_2 \rightarrow AS_3 : SF_3\}$

Table 7 – ALTO Cost Map Example

3.4.3 Multi-domain Software-Defined Networking (SDN)

Network providers are expanding the fine-grained capability of SDN from intradomain set-up to multi-domain settings to provide flexible interdomain routing as a valuable service (XIANG *et al.*, 2018; GUPTA *et al.*, 2015; MARQUES *et al.*, 2009). Users of this service can specify routing actions at the provider network based on flexible matching conditions of flow parameters such as TCP/IP 5-tuple. This service requires provider networks to expose their available routing information to users. However, handling routing information of each network individually is too complex for users. As such, a multi-domain network exposure solution that aggregates information of multiple networks into a single abstraction can simplify the use of this service.

- **How can flexible interdomain routing benefit from ALTO?**

  ALTO provides provider ASes a standardized approach to expose its routing capability to client ASes. Traditional interdomain routing protocols such as BGP are

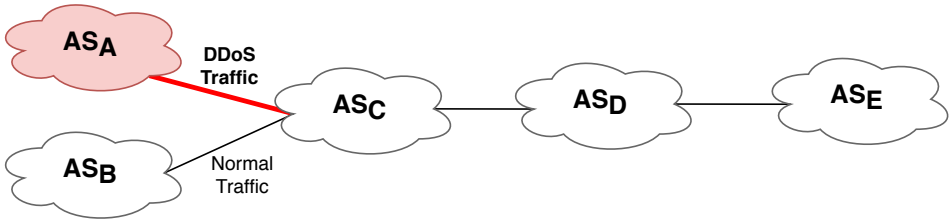


Figure 17 – Flexible interdomain routing for DDoS mitigation.

not good options because they only expose the currently used routes, limiting client ASes' choices to specify flexible routes. In contrast, ALTO and its extensions provide interfaces for provider ASes to expose not only currently used routes, but also available yet unused routes, to client ASes so that they can have the flexibility to specify different routes for different data traffic.

- **Example**

  Consider the example in Figure 4. AS_A is compromised and being used to send DDoS traffic to AS_E. Without flexible interdomain routing, AS_E can setup a firewall locally, but normal traffic from B to E will still be congested at $C-D-E$ due to the existence of malicious traffic from A to E. If AS_C provides a flexible interdomain routing service, AS E can specify such a firewall at AS_C to block DDoS traffic from A, and at the same time avoid the congestion of normal traffic from B to E.

3.5 ALTO Requirements on Multi-domain Network Information Exposure and Discovery

Supporting previous use cases with multi-domain network information exposure and discovery requires new features and extensions which are not fully satisfied by the current ALTO design. To appreciate such ALTO limitations, consider a P2P application example (the first and main use case for the development of ALTO (STIEMERLING *et al.*, 2016)). Figure 18 depicts a tracker-based P2P application with a global tracker (ALTO client) in domain A accessing ALTO servers at two ISPs (domains B and C). Using the current ALTO client protocol, the ALTO server in each domain will provide only local information to ALTO clients, *i.e.*, the tracker will receive only partial information of a single domain (domain B or domain C). On the other hand, using an ALTO server-to-server protocol, ALTO servers would be able to exchange information and the ALTO client would receive merged information from multiple domains. In the example (See Fig. 18), the tracker will receive merged information from domain A and domain B.

Next, we list several design issues of using ALTO to provide multi-domain information. Such issues can be roughly categorized in three aspects: (i) communication

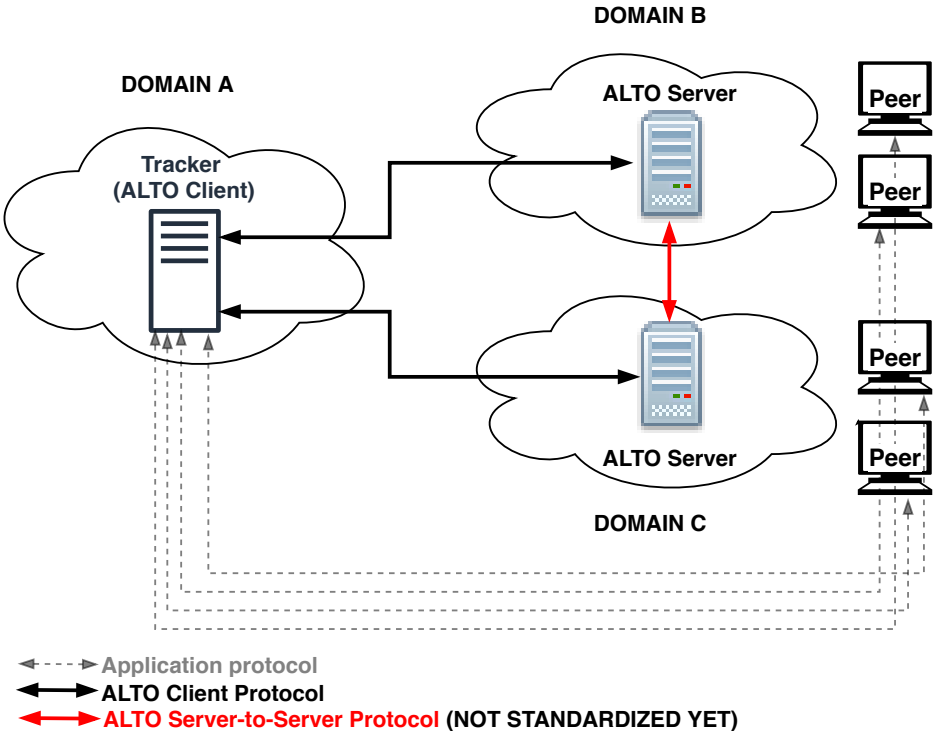


Figure 18 – From single- to multi-domain information exposure using ALTO: (i) ALTO client protocol and (ii) ALTO server-to-server protocol.

mechanisms; (ii) conceptual query interfaces and data representation; and (iii) computation model.

- **Communication Mechanisms**

3.5.1 Server-to-Client ALTO communication

In multi-domain scenarios, it is not possible to optimize the traffic with only locally available network information. For example, compute costs for source/destination pairs correctly if a source and/or a destination is outside the domain it belongs to. Therefore, communications among multiple ALTO servers are necessary to exchange detailed network information of multiple domains. The ALTO protocol specification states (See Section 3.1 of (ALIMI *et al.*, 2014)) that *"It may also be possible for an ALTO server to exchange network information with other ALTO servers (either within the same administrative domain or another administrative domain with the consent of both parties) in order to adjust exported ALTO"*. However, such a protocol is outside the scope of the specification.

3.5.2 Domain connectivity discovery

To find the resources shared by different source/destination pairs, an application needs to discover which domains are involved in the data movement of each node

pair. Besides, a set of candidate paths needs to be computed in order to know how to reach a remote destination node. The current ALTO extensions do not have this feature.

3.5.3 ALTO server discovery

Once the multi-domain connectivity discovery is performed, an application needs to be aware of the presence and the location of ALTO servers to get appropriate guidance. Those ALTO servers will be located in different domains, so that multi-domain ALTO server discovery mechanisms are also needed.

- **Conceptual Query Interfaces and Data Representation**

3.5.4 Single-domain composition

In the current ALTO framework, each domain can have its own representation of the same network information. For example, suppose that the path cost for member domain B (See Fig. 15) is utilization charge instead of available bandwidth. In this case, both values are not comparable together. Even, if all the member domains have the same utilization charge property, there may not necessarily have a uniform form of billing because each member domain is autonomous. Member domain A may charge using dollar, member domain B may charge using euros, while member domain C may use some form of local units. Therefore, it is necessary to design multi-domain composition mechanisms, so that network information in multiple domains are adapted together to a single and consistent "virtual" abstraction.

3.5.5 Simple resource query language

Applications also need to express their requirements in a query. For example, find the bandwidth the network can provide for flow $F_1(S_1, D_1)$ subject to reachability requirements (*e.g.*, from S_1 to D_1), bi-direction symmetry (*e.g.*, data traffic from S_1 to D_1 and from D_1 to S_1), waypoint traversal (*e.g.*, F_2 must traverse one middlebox m_1), blacklist of devices (*e.g.*, F_1 should not pass a certain device m_2), link/node disjointness (*e.g.*, F_1 and F_2 flows being transmitted along two link-disjoint paths), and QoS metrics (*e.g.*, the bandwidth of the flow F_1 needs to be at least 30 Gbps). The current query interface in ALTO (*e.g.*, filtered network/cost map) can not express such flexible queries.

- **Computation Model**

3.5.6 Scalability

Resource-filtering mechanisms may effectively reduce the redundancy in the network view. ALTO must have the capability to solve the optimization problems specified by the applications' requirements, which can be computationally expensive and time-consuming. For example, the number of available paths for each flow is increased exponentially with the number of domains involved, as does the number of available configurations for a set of flows with both the network size and the number of flows.

3.5.7 Security & Privacy

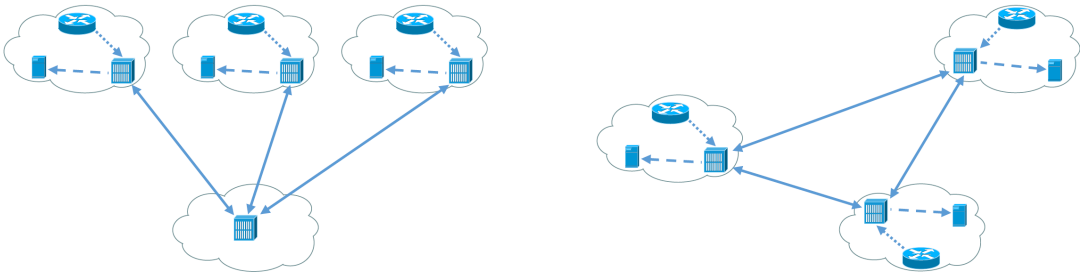
The information provided by the ALTO protocol is considered coarse-grained in several multi-domain use cases. New ALTO extensions have been designed to provide fine-grained network information to the applications. Using these ALTO extension services for multi-domain scenarios would raise new security and privacy concerns.

3.6 A multi-domain ALTO Framework: Envisioned solutions & on-going efforts

In order to address the aforementioned issues, this section summarizes envisioned solutions and on-going efforts to allow ALTO to expose network information across multiple domains. See Table 8 to identify the relationship between the key design issues and their corresponding mechanisms to consider in a multi-domain ALTO framework.

Table 8 – Issues of applying the current ALTO framework in the multi-domain setting & solutions

Current Key Issues	**Envisioned Mechanisms**	**Reference(s)**	**Sub-Section**
Server-to-Client ALTO communication	**Server-to-Server ALTO communication**	(DULINSKI *et al.*, 2015; XIANG *et al.*, 2020c; PEREZ; ROTHENBERG, 2020; PEREZ *et al.*, 2020b)	subsection 3.6.1
Domain connectivity discovery	**Multi-domain connectivity discovery**	(REKHTER *et al.*, 2006; VASSEUR *et al.*, 2009; KING; FARREL, 2012; GREDLER *et al.*, 2016)	subsection 3.6.2
ALTO server discovery	**Multi-domain ALTO server discovery**	(KIESEL; STIEMERLING, 2020; ROUX, 2006; DONG *et al.*, 2017)	subsection 3.6.3
Single-domain composition	**Unified Resource Representation**	(XIANG *et al.*, 2020b; XIANG *et al.*, 2019; XIANG *et al.*, 2018)	subsection 3.6.4
Simple resource query language	**Generic/Flexible query language**	(ETSI, ; ASSOCIATION, 2019b; SCHMIDT *et al.*, 2013; CLEMM *et al.*, 2020)	subsection 3.6.5
Scalability	**Computation complexity optimization**	(GAO *et al.*, 2018; XIANG *et al.*, 2018)	subsection 3.6.6
Security & Privacy	**Security/Privacy preserving**	(XIANG *et al.*, 2020c; XIANG *et al.*, 2018)	subsection 3.6.7



(a) Hierarchical design (b) Mesh design

Figure 19 – ALTO Server deployments[5].

3.6.1 Server-to-Server ALTO communication

ALTO servers may consider either a hierarchical or mesh architectural deployment design (DULINSKI *et al.*, 2015). When a hierarchical architecture is used (See Fig. 19a), ALTO servers in domain partitions gather locally-available network information and send it to central server, which in turn merges data and distributes ALTO services. In a mesh deployment (See Fig. 19b), ALTO servers may be set up in each domain independently, connected to each other, and gathering the network information from other domains.

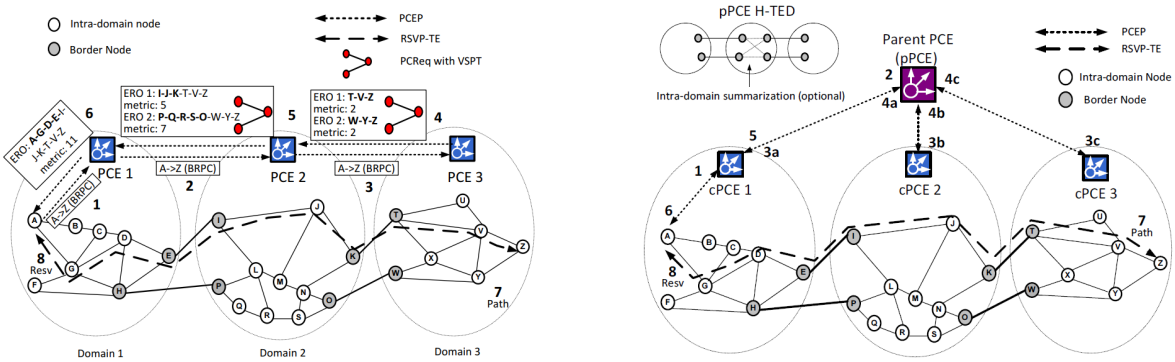
(XIANG *et al.*, 2020c) presents Unicorn, a resource orchestration framework for multi-domain, geo-distributed data analytics. This work proposes a collaborative approach with one or more ALTO servers deployed in each member domain. Unicorn also contains an ALTO client that communicates with the ALTO servers at member networks to retrieve resource information. The key information to be provided by the use of ALTO including different types of resources, *e.g.*, the computing, storage, and networking resources.

(PEREZ; ROTHENBERG, 2020; PEREZ *et al.*, 2020b) propose an ALTO-based Broker-assisted architecture where a broker plane works as a coordinator between a set of top-level control planes, *i.e.*, Multi-domain Orchestrator (MdOs). A logically centralized ALTO server provides abstract maps with a simplified information view about MdOs involved in the federation. This information includes the abstract network topology, resource availability, and capabilities.

3.6.2 Multi-domain connectivity discovery

Multi-domain mechanisms combining domains sequence computation and paths computation need to be defined, or standardized computation protocols could be leveraged for inspiring this design requirement. In the latter case, the IETF has a set of well-defined protocols, such as BGP (REKHTER *et al.*, 2006), PCE (VASSEUR *et al.*, 2009; KING; FARREL, 2012), or BGP-LS (GREDLER *et al.*, 2016).

[5] Figures source: <https://datatracker.ietf.org/meeting/93/materials/slides-93-alto-2>



(a) RFC5441 (b) RFC6805

Figure 20 – Multi-domain end-to-end paths computation: (a) A PCE entity cooperates with other PCE entities in adjacent domains, and (b) A PCE entity cooperates with a parent PCE entity[6].

The BGP protocol (REKHTER *et al.*, 2006), for instance, provides multi-domain sequence computation to know how to reach a destination by identifying the next hop for IP traffic delivery; however, it does not advertise multiple alternative routes. BGP-LS (GREDLER *et al.*, 2016) allows visibility of the network topology (real physical or abstracted) and export traffic engineering information with external domains using the BGP routing protocol.

Following the PCE-based architecture (VASSEUR *et al.*, 2006) for computing optimal multi-domain end-to-end paths, (VASSEUR *et al.*, 2009) and (KING; FARREL, 2012) define mechanisms where a PCE entity cooperates either with other PCE entities in adjacent domains (See Fig. 20a) or with a parent PCE entity (See Fig. 20b), respectively. A mix between BGP-LP and PCE may also be considered, with the first one providing topology/link-state network information, and with the second one making the necessary path computations between domains.

3.6.3 Multi-domain ALTO server discovery

The ALTO cross-domain server discovery document (KIESEL; STIEMERLING, 2020) specifies a procedure for identifying ALTO servers outside of the ALTO client's own network domain. This document specifies an ALTO cross-domain server discovery procedure for client-side usage inspired by Location Information Server (LIS) Discovery Using IP Addresses and Reverse DNS (THOMSON; BELLIS, 2014), and reuses parts of the basic ALTO Server Discovery procedure (SONG *et al.*, 2014).

Other mechanisms could also be leveraged, such as those based on PCE or BGP architectures. For example, RFC4674 (ROUX, 2006) proposes a set of functional requirements to allow a Path Computation Client (PCC) to automatically and dynamically

[6] Figures source: A Survey on the Path Computation Element (PCE) Architecture

discover the location of PCEs entities (including additional information about supported capabilities) for each controller domain. Inline with those requirements, (DONG *et al.*, 2017) is defining extensions to BGP to also carry PCE discovery information. Specifically, this document extends BGP to allow a PCE entities to advertise their location and some useful information to a PCC for the PCE selection.

3.6.4 Unified Resource Representation

Although the existing abstractions (network/cost map, unified property, and path vector) are already powerful, they cannot handle the composition across multiple domains. Therefore, multi-domain composition mechanisms are necessaries so that network information from ALTO servers in multiple domains can fit into a single and consistent "virtual" domain abstraction. Network maps, cost maps, unified entity properties, network capabilities, and routing path abstractions (path vectors) of individual domains need to follow a common semantic as well as be consistently integrated to provide the abstraction of a single, coherent network to the applications.

(LACHOS *et al.*, 2020; XIANG *et al.*, 2020b; XIANG *et al.*, 2019; XIANG *et al.*, 2018) propose the use of mathematical programming constraints for multi-domain composition and represent the capacity regions for a set of flows. In particular, (XIANG *et al.*, 2020b) specifies a new cost metric called "variable-list". This cost metric indicates that the cost value is a list of variables that will be used in mathematical programming constraints. It also introduces a new entity domain "cstr" (short for constraint), which is registered in the property map. Each entity in the "cstr" domain has an identifier of a constraint. Each constraint has one property, which represents the semantics of this constraint, for example, a "bw-cstr" property indicates that this constraint represents the bandwidth sharing among flows. This property is provided in information resources called "Property Map Resource" and "Filtered Property Map Resource".

3.6.5 Generic/Flexible query language

With a flexible and generic query language, the network can filter out a large number of unqualified domains. The language specification could be inspired by standard (*e.g.*, 5G Slice Templates or ETSI NFV Network Service Descriptor) or pre-standard (*e.g.*, Socket Intents) mechanisms (GAO *et al.*, 2020a), implemented with a user-friendly grammar (*e.g.*, SQL-style query).

Table 9 summarizes the main standardization and pre-standardization efforts for expressing features of services and applications. The application descriptors in MEC or slice templates in GSMA and 3GPP, for instance, allow specifying the application requirements in a static manner. Other methods such as the descriptors in ETSI ZSM and

SDO	Purpose	Method to express needs	Metrics			
			Edge	VNF	VIM	E2E
ETSI ZSM	A model-driven approach to perform service and resource management	Templates detailing virtual function and virtual link requirements		✓	✓	✓
ETSI NFV-IFA	Deployment template which consists of parameters used by the NFVO during lifecycle of an NS	Descriptor templates detailing virtual function requirements and virtual link needs		✓	✓	
GSMA & 3GPP	5G network slice templates with attributes that describe the service characteristics	Templates with attributes & values				✓
ETSI MEC	Application templates for MEC environments that specify resources and service lifecycle aspects.	Descriptors detailing application requirements	✓	✓	✓	
TM Forum	Service requests and lifecycle management through standard APIs	Set of APIs with very specific purposes		✓	✓	✓

Table 9 – Summary of Standardization and Pre-Standardization Efforts for Expressing Features of Services and applications. Adapted from (GAO *et al.*, 2020a).

ETSI NFV include the possibility of embedding indications to embed prompts to scale the functions composing a service.

3.6.6 Computation complexity optimization

ALTO servers need to support mechanisms such as pre-computation, projection, and/or compression to improve the scalability and performance. Such mechanisms should effectively reduce the redundancy in the network view as much as possible while still providing the same information.

For example, (GAO *et al.*, 2018) describes equivalent transformation algorithms that identify/remove redundant information to obtain a more compact view. Specifically, authors propose a supplement to the ALTO path vector extension through three algorithms, which can effectively reduce the redundancy in the network view while still providing the same information as in the original path vectors. The equivalent aggregation algorithm compresses the original path vectors by aggregating the network elements with the same set of pairs. The redundant constraints algorithm compresses the original path vectors by removing the network elements that provide only redundant information. Finally, the equivalent decomposition algorithm compresses the original path vectors by decomposing redundant network elements to obtain the same end-to-end routing metrics.

Meanwhile, (XIANG *et al.*, 2018) proactively discovers network resource information for a set of flows, and project the pre-computed result to get the information when receiving actual requests from applications. Specifically, given a set of member networks, each network periodically sends updated information to an aggregator and when the net-

work receives and successfully executes a resource request from the application, it sends a notification to the aggregator with the request details so that the aggregator can update the projected abstraction.

3.6.7 Security/Privacy preserving

ALTO needs mechanisms (with little overhead) that provide accurate sharing network information, and at the same time, protects each member domain. This privacy-preserving multi-domain information process may consider, for instance, a secure multi-party computation (RAYKOVA, 2012).

For example, for collaborative science networks where member domains share resources to conduct common tasks collaboratively (*e.g.*, analytics, data transfers, and storage), (XIANG *et al.*, 2020c; XIANG *et al.*, 2018) propose a semi-honest security model. In this model, all member domains and an aggregator will not deviate from the security protocol, but just try to collect information during the protocol execution. Specifically, authors design a novel resource abstraction obfuscating protocol leveraging the random matrix theory. In particular, each domain D_i independently computes and sends to an aggregator a set of disguised linear equations, which are derived from the private linear inequalities, a random matrix P_i known only to D_i, two random matrices C_i and D_i known only to D_i and D_{i-1}, and two random matrices C_{i+1} and D_{i+1} known only to D_i and D_{i+1}.

3.7 Concluding Remarks

In this chapter, we review important multi-domain use cases that can benefit from network information exposure/discovery using ALTO. Next, we discuss the ALTO design issues for gathering such multi-domain information. We then present a set of mechanisms and envisioned solutions based on recent research to substantially improve the ALTO framework to support important multi-domain environments.

The multi-domain aspects of ALTO raised in this chapter are currently under discussion into the IETF ALTO WG. Therefore, future research to solve several dilemmas, not included in its completeness, remain open, under which we announce its principal shortcomings below:

- By expanding the current ALTO protocol to a multi-domain approach, our discussion lacks considerations on the feasibility. That is, if there are industrial players willing to implement or deploy the ALTO extensions on a scale that requires and justifies the effort for standardization. In this context, BECOCS is considering to

implement a multi-ISP collaboration system[7] and standardize many of the multi-domain aspects involved in its federated FlowDirector.

- There are some concerns if the potential solutions or extensions are already ripe for standardization. In this way, there are currently different interactions and research discussions in the ALTO WG to identify a shorted list of extensions along with real use case implementations to motivate the adoption of those different extensions.

- Also, there is a gap regarding how the work on ALTO so far has grappled with the business implications in multi-domain environments, especially in multiple administrative domains. In this case, future activities will include a review of coordination models, Service Level Agreements (SLAs), pricing schemes, economic incentives, and Operations Support Systems (OSS)/Business Support Systems (BSS) integration.

- The ALTO capabilities to expose and discover multi-domain network information should be compared with other non-ALTO specific approaches (pros and cons). Only thus can the propositions of this chapter be confirmed as a whole.

- Finally, most of the examples and proposals for unified resource representation are related to the use of mathematical programming constraints for representing the bandwidth resource. Future discussions about this multi-domain mechanism will need consider the analysis of other universal units (*e.g.*, latency, packet loss) and non-universal units (*e.g.*, utilization charge). In both cases, it will also be necessary to consider normalized mechanisms to abstract real metric values into non-real numerical scores or ordinal ranking.

[7] <https://youtu.be/MdBwzWug06M?t=4290>

4 ANI: Abstracted Network Inventory

4.1 Introduction

Emerging use cases like virtual and augmented reality, autonomous vehicles, smart cities, and drones call for transforming the way telecommunications operators deploy new network services, shifting from a manual and long process to a more flexible and programmable way (SOUSA *et al.*, 2019; GUERZONI *et al.*, 2017). In this context, cloud computing (LE *et al.*, 2016; WEERASIRI *et al.*, 2017), SDN (KREUTZ *et al.*, 2015; JARRAYA *et al.*, 2014), and NFV (MIJUMBI *et al.*, 2015; SLIM *et al.*, 2017; Yong Li; Min Chen, 2015; BHAMARE *et al.*, 2016) arise as technological pillars to achieve the necessary flexibility and programmability during the provision of such network services. By softwarizing a network service, Virtual Network Functions (VNFs) are separated from the hardware and offered through virtualized services that can be instantiated on data centers (as any other cloud applications) with the adequate connectivity.

A distributed cloud is a cloud execution environment for VNFs or applications that is distributed across multiple cloud sites (edge, regional, and central), with the required connectivity (networking) between them (ERICSSON, 2018). Distributed cloud deployment models keep latency-sensitive applications closer to the edges of the network (close to users), and move non-real-time applications to centralized data centers (AT&T, 2017).

In such distributed cloud environments with edge and more centralized computing facilities, multiple cloud sites become candidate hosting targets for VNFs and applications. Cloud sites are typically geographically distributed and interconnected through a Wide Area Network (WAN). Figure 21 shows interconnected edge cloud sites in which a centralized orchestrator, or Central Orchestrator (CO), can establish communication with Local Orchestrators (LOs) placed at individual edge cloud sites. An LO is also part of an edge cloud site so that some orchestration components can be deployed locally in a data center without always accessing the WAN. Eventually, regional cloud sites could also be deployed between central and edge cloud sites. Examples of open source projects that consider local and central orchestrators include Akraino[1] (See Fig 22a) and ONAP[2] (See Fig 22b).

To take optimized placement decisions, COs or LOs need to maintain an inventory of the network providing a real-time representation of the available resources in the network infrastructure along with their relationships. The size of a network inventory can

[1] <https://wiki.akraino.org/display/AK/Akraino+Edge+Stack>
[2] <https://wiki.onap.org/display/DW/Edge+Automation+through+ONAP>

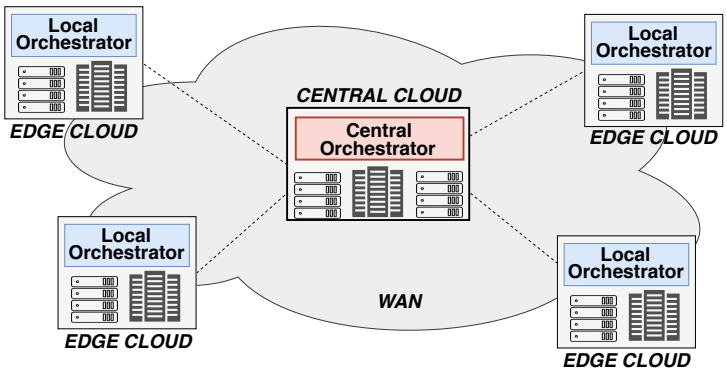


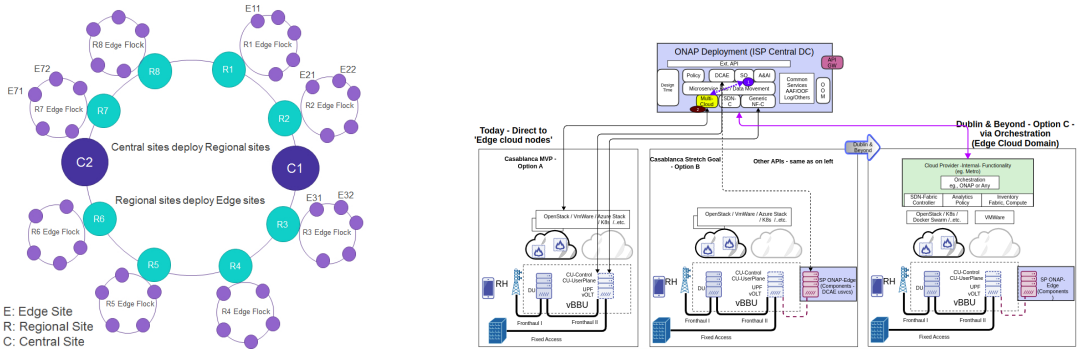
Figure 21 – A distributed cloud computing environment comprised by interconnected cloud sites and a centralized orchestrator communicating with local orchestrators in edge clouds.

become very large in distributed cloud scenarios because a typical service provider will have hundreds or thousands of edge cloud deployments. As a result, COs and LOs face scalability challenges when processing large amounts of data to decide where to instantiate a service or part of the service. A common system engineering principle to deal with scalability requirements is to introduce proper abstraction mechanisms that reduce the discovery time of resources while simplifying and optimizing their management.

Some examples of resource abstraction mechanisms are one-big-switch abstractions (SONKOLY *et al.*, 2015), virtual-link abstractions (FIORANI *et al.*, 2015; FIORANI *et al.*, 2016), and linear inequalities representation (XIANG *et al.*, 2018). However, to the best of our knowledge, none of the existing approaches take into account service requirements to generate a logical network inventory representation. The novel abstraction method[3] presented in this chapter is directed to the Abstracted Network Inventory (ANI) component (i) receiving services requirements from a catalog, (ii) receiving a network representation from a network inventory, and (iii) processing those inputs to generate an optimized abstract network representation, referred to as Logical Network Inventory (LNI). Using the LNI delivers two main advantages. First, a reduced time for placement of VNFs since resource candidates for the placement are logically reduced, in terms of the number of compute nodes and links, in comparison to the original representation in the conventional network inventory. Second, a summarized topology per service can be used to simplify and optimize the management of resources. An example of management is life-cycle operations in which service resources should be rearranged such as scaling VNFs or workloads.

A challenging task for LNI generation is to find an optimal mapping of VNFs within a network service to the components of a network inventory. To address this issue, we formalize a system model to solve the LNI generation problem based on network service

[3] PCT Patent Application has been filed at EPO on 04/02/2019 (Serial No. PCT/EP2019/058274).



(a) Akraino[4] (b) ONAP[5]

Figure 22 – Open source projects using centralized orchestrators.

requirements and infrastructure capabilities. Afterward, we also develop three algorithms to build different types of LNIs efficiently: (i) Node-oriented LNI, (ii) Edge-oriented LNI, and (iii) Node/Edge-oriented LNI. Then, we evaluate these proposed algorithms through different experiments using multiple topologies.

This chapther is organized as follows. Section 4.2 covers related work for network inventory creation and abstraction. Section 4.3 describes in detail the ANI component (key concepts, deployment, and LNI generation). Section 4.4 gives the formulation of the ANI model, including the introduction of the three different algorithms for the LNIs generation. Section 4.5 evaluates the LNI methods and their impact in the network service management. Finally, we conclude the chapter in Section 4.6.

4.2 Related Work

Several solutions have been proposed for network inventory creation and abstraction (SONKOLY *et al.*, 2015; XIANG *et al.*, 2018; FIORANI *et al.*, 2015; FIORANI *et al.*, 2016; LICCIARDELLO *et al.*, 2017; SOENEN *et al.*, 2016; ONAP, 2018). Table 10 shows different projects/related work and their target environment, inputs, and abstraction method. We observed that the solutions available are not using the network service requirements as an input to generate network inventory abstractions.

UNIFY (SONKOLY *et al.*, 2015), for instance, provides an abstraction of type big-switch and big-software that includes compute and network resources. Solution in (XIANG *et al.*, 2018) uses linear inequalities to represent network resources availability in terms of bandwidth. However, all these solutions do not consider service requirements to generate a network representation. In addition, network representations are typically either coarse- or fine-grained. The former (See Fig 23a) does not provide enough information from the infrastructure for placement decisions (SONKOLY *et al.*, 2015). Fine-grained methods

[4] Figure source: <https://wiki.akraino.org/display/AK/Akraino+Edge+Stack>
[5] Figure source: <https://wiki.onap.org/pages/viewpage.action?pageId=28381325>

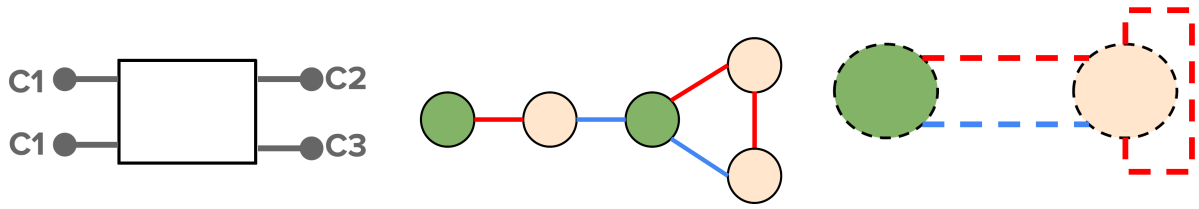
Project/ Related Work	**Target**	**Input**		**Abstraction Method**			
		Network Resources	**Network Service**	**Big Switch**	**Graph based**	**Cluster based**	**Others**
UNIFY (SONKOLY *et al.*, 2015)	Compute and network resources	✓		✓			
Mercator (XIANG *et al.*, 2018)	Bandwidth resource availability	✓					✓
(FIORANI *et al.*, 2015; FIORANI *et al.*, 2016)	Optical transport networks	✓		✓	✓		
(LICCIARDELLO *et al.*, 2017)	Distributed data center networks	✓		✓	✓		
(SOENEN *et al.*, 2016)	Service function chaining	✓				✓	
ANI [this work]	Multi-domain distributed environments	✓	✓		✓		

Table 10 – Target, abstraction method, and input of different abstraction related work and projects.

(See Fig 23b) are too costly for a distributed cloud environment, meaning that since the number of infrastructure resources such as switches and compute devices can be very large in a distributed cloud environment. Solution in (XIANG *et al.*, 2018) does not consider network services in terms of VNFs since it is a solution for workloads with available bandwidth requirements.

Likewise, (FIORANI *et al.*, 2015; FIORANI *et al.*, 2016) provide different abstraction models (big-switch, virtual link with single weights, virtual link with multiple weights, and optical transport transformation) of optical transport networks, focusing especially on centralized radio access networks (C-RANs). However, both solutions also do not consider the use of service requirements as an input to generate an abstract network representation. Besides, abstraction models for cloud environments are out of scope, as stated by the authors.

Work in (LICCIARDELLO *et al.*, 2017) uses abstraction strategies described in (FIORANI *et al.*, 2016) for distributed data center network scenarios, however, it does not provide new abstraction mechanisms to represent information related to the network infrastructure. (SOENEN *et al.*, 2016) supports grouping by joining entities within hypernodes or hyperedges (See Fig 23c). Our proposal has two main differences (i) filtering instead of aggregation as abstraction mechanism and (ii) service requirements as an additional input. Finally, ONAP AAI (ONAP, 2018) is a component that provides real-time network inventory of infrastructure resources, but there is no defined an abstraction process.



(a) (b) (c)

Figure 23 – Different approaches for network inventory abstractions: (a) One-Big-Switch approach, (b) Graph-based approach, and (c) Cluster-based Approach.

4.3 Abstracted Network Inventory (ANI)

In this section, we will provide more information about the basic definitions (network inventory and network service), basic exemplary system architectures in which the ANI component may be executed, and the LNI generation process.

4.3.1 Basic Definitions

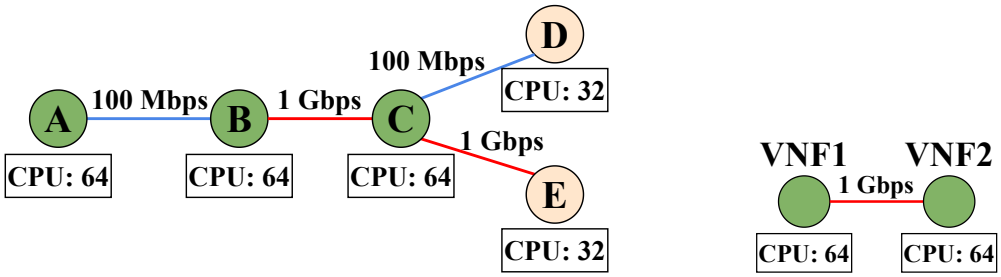
- **Network Inventory.** Network infrastructure resources are represented in the form of a network inventory. A network inventory may comprise nodes[6] and links[7] each providing a particular capacity (See Fig. 24a). Examples of node-related capacities are number of CPUs, amount of RAM (*e.g.*, 64 GB of RAM), and amount of disk space (*e.g.*, 10 TB of disk space). Examples of link-related capacities include bandwidth characteristics (*e.g.*, 100 Mbps) and latency characteristics (*e.g.*, 1 ms RTT).

- **Network Service.** A network service, or simply service, is deployed or instantiated over the infrastructure resources. As Figure 24b shows, a service specifies one or more nodes (a set of required VNFs) as well as links (how VNFs are connected) (HALPERN; PIGNATARO, 2015). Nodes include resource demands (*e.g.*, CPU, memory, storage), and links contain performance objectives (*e.g.*, latency, bandwidth).

4.3.2 ANI Component & Deployment

The ANI component proactively constructs multiple network views over the same network infrastructure, called LNIs. Each LNI is optimized to a service in terms of its requirements such as CPU, memory, latency, etc. Every new service in a catalog triggers

[6] The terms node and vertex are used interchangeably
[7] The terms link and edge are used interchangeably



(a) (b)

Figure 24 – ANI Basic Definitions: (a) Network Inventory and (b) Network Service.

the creation of another LNI that will be part of the optimized network inventory. As such, service requirements are used in the method to guide the right level of abstraction.

The ANI component may be executed as part of the CO (see Fig. 25a). As shown in the figure, the ANI receives two inputs: (i) services requirements from a catalog, and (ii) a central network inventory representation. The CO component may build a central network inventory based on local resource infrastructure information received from one or more LOs, which maintain a local network inventory. In another variant (see Fig. 25b), the ANI component may be executed in a distributed manner across a service provider (hosting the CO component), and several cloud providers (hosting the LO component).

4.3.3 Logical Network Inventory (LNI)

A two-step procedure is performed to generate a LNI. In the first step, a classification of a service is carried out to determine whether the service is node-oriented, edge-oriented, or node/edge-oriented. In the second step, the actual node and/or edge oriented mode is executed over the network inventory to generate an LNI.

1. **Service Classification.** This classification may be executed: (i) calculating resource and performance reference values in the network service. In the case of the resource reference value (RRV_s), it can be computed as the sum of CPU demands, and the performance reference value (PRV_s) can be computed as the maximum bandwidth objective; (ii) determining a node and edge reference values in the network inventory. The node reference value (NRV_i) can be computed as the sum of CPU capacities, and the edge reference value (NRV_i) as the average or worst-case link capacities in terms of bandwidth; (iii) Comparing previous references values to determine if node and/or edges are relevant for the placement of a service. Figure 26 depicts an illustrative workflow of this classification process[8]:

   - If the performance and resource reference values are greater than the node and edge reference values ($RRV_s >> NRV_i \wedge PRV_s >> ERV_i$). It means that node

[8] Other different mechanisms may be used to classify a service (out of the scope of this work).

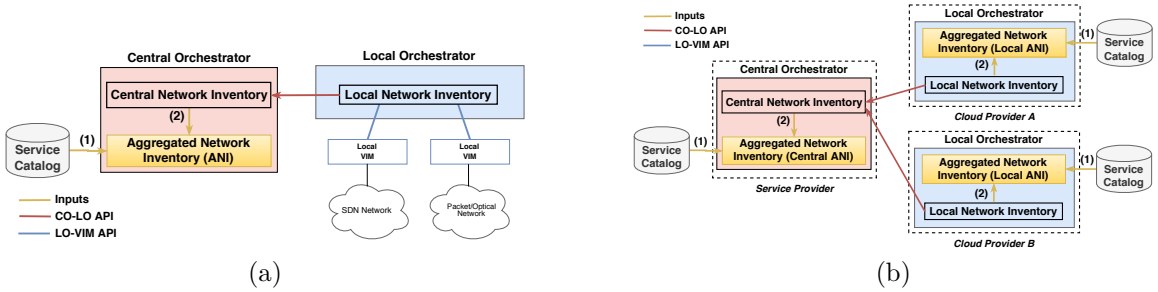


Figure 25 – Exemplary system architectures in which the ANI component is (a) executed as part of a CO component or (b) in a distributed manner across several orchestrators components, *i.e.*, CO and LOs.

and edge capacities in the network inventory are under-provisioned, therefore node and edge are relevant for the placement and the service is classified as node/edge-oriented.

- On the other hand, If only the resource reference value is greater than the node reference value ($RRV_s >> NRV_i$), it means that node capacities in the network inventory are under-provisioned the service is classified as node-oriented.
- Otherwise ($PRV_s >> ERV_i$), the service is classified as edge oriented.

2. **LNI Generation.** The process of building up an LNI is in accordance with the service classification, so that we have three types of LNIs: (i) node-oriented LNI, (ii) edge-oriented LNI, and (iii) node/edge-oriented LNI. Section 4 provides more detailed information about this generation.

   - **Node-oriented LNI.** A set of vertices in the network inventory are assigned to the LNI (links are discarded). Such selected vertices have to support the CPU capacity constraint of VNF nodes in a service. In our previous example in Figure 24, if the requested service is classified as node-oriented, pink nodes and all the links are logically discarded from the network inventory (See Fig. 27a) based on the network service demands (green nodes).
   - **Edge-oriented LNI.** A set of edges in the network inventory are assigned to the LNI. Selected edges have to support the required bandwidth capacity of edges in a service. For example, if the requested service in Figure 24 is classified as edge-oriented, blue links are discarded and only the red links are part of the LNI (See Fig. 27b).
   - **Node/Edge-oriented LNI.** A set of vertices and edges in the network inventory are assigned to the LNI. Vertices and edges are selected according to the CPU and bandwidth capacity constraints in a service, respectively. Using the example in Figure 24, pink nodes and blue links are logically discarded from

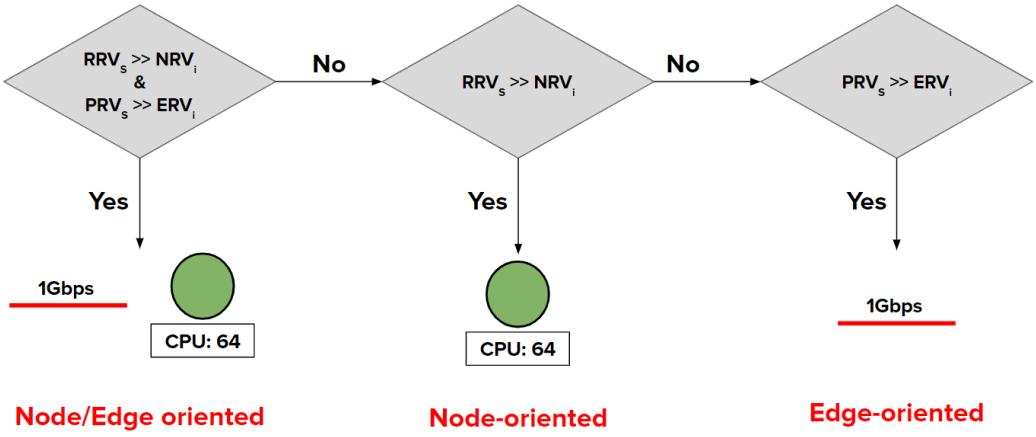
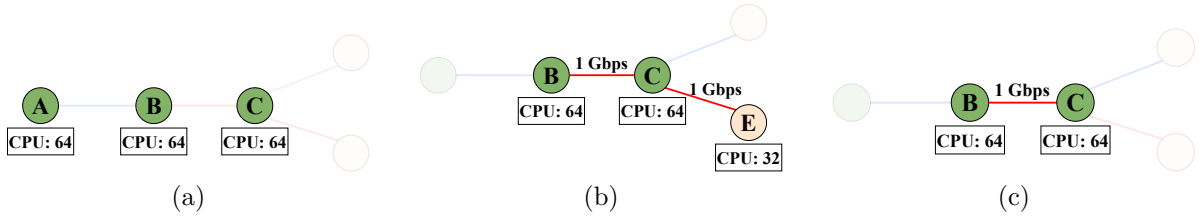


Figure 26 – Service classification workflow: (i) Node/Edge-Oriented, (ii) Node-oriented, and (iii) Edge-oriented.



(a) (b) (c)

Figure 27 – LNI Generation: (a) Node-oriented LNI, (b) Edge-oriented LNI, and (c) Node/Edge-oriented LNI.

the network inventory (See Fig. 27c) based on the network service demands (green nodes and red link) and service classification (node/edge-oriented).

4.4 Network Model & Proposed Algorithms

In this section, we will formalized the ANI network model (network service and network inventory) along with the formulation of three algorithms to create a LNI: (i) Node-oriented LNI, (ii) Edge-oriented LNI, and (iii) Node/Edge-oriented LNI.

4.4.1 Network Model

- **Network Service.** We model a network service as a directed graph denoted by $G_s = (V_s, E_s)$, where V_s is a set of VNFs connected via a set of directed edges E_s. Each VNF $v_s \in V_s$ is associated with a requested CPU capacity value cpu_{v_s}. Each edge $e_s(y, z) \in E_s$, connecting two VNFs y and z, is associated with a requested bandwidth capacity value bw_{e_s}[9].

[9] We are considered a basic scenario, with only CPU and bandwidth constraints. Additional capacity constraints of nodes (*e.g.*, memory) and edges (*e.g.*, delay) can be easily extended in our model.

- **Network Inventory.** In its basic form, a network inventory is modeled as an undirected graph $G_i = (V_i, E_i)$, where a vertex $v_i \in V_i$ has an available CPU capacity (CPU_{v_i}), and an edge $e_i(m, n) \in E_i$, between two vertices m and n , is associated with a bandwidth capacity BW_{e_i}. We also denote the set of all loop-free paths from the source vertex s to the destination vertex d by $P_i(s, d)$. Therefore, the available bandwidth capacity of a path $p_i \in P_i$ is given by:

$$BW(p_i) = \min_{e_i \in p_i} BW(e_i)$$

  In order to support graph collections, our network inventory model is extended to include a set of LNI graphs $G_i = (V_i, E_i, LNI_i)$, where LNI_i represents multiple possible views of the same network inventory.

  We model a LNI graph $L_l \in LNI_i$ as an undirected graph denoted by $L_l = (V_l, E_l)$ where V_l is a subset of vertices such that $V_l \subseteq V_i$, and E_l is a subset of edges such that $E_l \subseteq E_i$. Besides, LNI graphs may overlap such that $\forall L_y, L_z \subseteq LNI_i$: $|V(L_y) \cap V(L_z)| \geq 0 \wedge |E(L_y) \cap E(L_z)| \geq 0$.

4.4.2 Proposed Algorithms

The primary objective here is to design three algorithms that efficiently create LNIs. It is worth mentioning that the proposed algorithms can be categorized as offline. Offline algorithms optimize over a large set of service requests and search near-optimal or optimal solutions, which typically comes at the expense of long run-times (NÉMETH *et al.*, 2016).

The ANI component considers three modes of LNIs generation:

4.4.2.1 Node-oriented LNI

It takes a network service $G_s = (V_s, E_s)$ and a network inventory $G_i = (V_i, E_i, LNI_i)$ as input and returns a set of vertices $V(L_y)$ such that $L_y \subseteq LNI_i$.

For each service node v_s, the algorithm searches all nodes v_i in the network inventory and adds nodes with CPU capacity greater than or equal to the CPU requirement (Line 3) into the subset of vertices $V(L_y)$ (Line 4). The pseudo-code of this mode is provided in Algorithm 1.

4.4.2.2 Edge-oriented LNI

Algorithm 2 provides an overview of this proposed mode. It takes a network service $G_s = (V_s, E_s)$ and a network inventory $G_i = (V_i, E_i, LNI_i)$ as inputs. However, it returns a set of vertices $V(L_y)$ and a set of edges $E(L_y)$ as output.

Algorithm 1: Node-oriented LNI

Input:
$G_s = (V_s, E_s)$: Network Service;
$G_i = (V_i, E_i, LNI_i)$: Network Inventory;
Output:
$V(L_y), \forall L_y \in LNI_i$ and $V_l \subseteq V_i$;

```
1 foreach v_s in V_s do
2     foreach v_i in V_i do
3         if CPU(v_i) ≥ cpu(v_s) then
4             V(L_y) ← V(L_y) ∪ {v_i};
```

For each service edge e_s, we first iterate all the possible network inventory nodes to get a source node $v_{i_{src}}$ and a destination node $v_{i_{dst}}$ (Lines 1-6). After, the algorithm finds the paths from $v_{i_{src}}$ to $v_{i_{dst}}$ (Line 7) and only considers those paths that respect the bandwidth requirement in a service link $bw(e_s)$ (Line 8). Finally, all the nodes and edges which are in a path p_i (Line 9) will be part of the subset of vertices $V(L_y)$ and edges $E(L_y)$ (Lines 10-12).

4.4.2.3 Node/Edge-oriented LNI

This algorithm (see Algorithm 3) is quite similar to Algorithm 2. It also takes a network service $G_s = (V_s, E_s)$ and a network inventory $G_i = (V_i, E_i, LNI_i)$ as inputs, and

Algorithm 2: Edge-oriented LNI

Input:
$G_s = (V_s, E_s)$: Network Service;
$G_i = (V_i, E_i, LNI_i)$: Network Inventory;
Output:
$V(L_y), \forall L_y \in LNI_i$ and $V_l \subseteq V_i$
$E(L_y), \forall L_y \in LNI_i$ and $E_l \subseteq E_i$;

```
1  foreach e_s(src, dst) in E_s do
2      v_{i_src} ← ∅, v_{i_dst} ← ∅;
3      foreach v_i in V_i do
4          v_{i_src} ←v_i;
5          foreach v_i in V_i do
6              v_{i_dst} ←v_i;
7              foreach p_i in P_i(v_{i_src}, v_{i_dst}) do
8                  if BW(p_i) ≥ bw(e_s) then
9                      foreach e_i(s, d) in p_i do
10                         V(L_y) ← V(L_y) ∪ s;
11                         V(L_y) ← V(L_y) ∪ d;
12                         E(L_y) ← E(L_y) ∪ e_i;
```

Algorithm 3: Node/Edge-oriented LNI

Input:
$G_s = (V_s, E_s)$: Network Service;
$G_i = (V_i, E_i, LNI_i)$: Network Inventory;
Output:
$V(L_y), \forall L_y \in LNI_i$ and $V_l \subseteq V_i$
$E(L_y), \forall L_y \in LNI_i$ and $E_l \subseteq E_i$;

```
1  foreach e_s(src, dst) in E_s do
2      v_{i_src} ← ∅, v_{i_dst} ← ∅;
3      foreach v_i in V_i do
4          if CPU(v_i) ≥ cpu(src) then
5              v_{i_src} ←v_i;
6              foreach v_i in V_i do
7                  if CPU(v_i) ≥ cpu(dst) then
8                      v_{i_dst} ←v_i;
9                      foreach p_i in P_i(v_{i_src}, v_{i_dst}) do
10                         if BW(p_i) ≥ bw(e_s) then
11                             foreach e_i(s, d) in p_i do
12                                 V(L_y) ← V(L_y) ∪ s;
13                                 V(L_y) ← V(L_y) ∪ d;
14                                 E(L_y) ← E(L_y) ∪ e_i;
```

it returns a set of vertices $V(L_y)$ and a set of edges $E(L_y)$ as output.

For each service edge e_s, the node/edge-oriented LNI algorithm only considers the network inventory nodes that satisfy the CPU constraints of service nodes (Lines 4 and 7) and obtain a source node $v_{i_{src}}$ (Line 5) and a destination node $v_{i_{dst}}$ (Line 8). Then, the algorithm finds the paths from $v_{i_{src}}$ to $v_{i_{dst}}$ (Line 9) and only considers those paths that satisfy the bandwidth requirement in a service link $bw(e_s)$ (Line 10). Last, nodes and edges which are in a path p_i (Line 11) will be included in the subset of vertices $V(L_y)$ and edges $E(L_y)$ (Lines 12-14).

4.5 Experimental Evaluation

In this section, we analyze the performance of our proposed algorithms. Specifically, we evaluate the quality of LNIs, generated by the ANI component, using randomly generated network inventory topologies. Another experimental evaluation, integrating the ANI component into the MUDED platform, is discussed in Chapter 5 (Section 5.5.2). In this latter, we analyze the impact of the ANI on the network service provisioning time using a real graph dataset. For each experiment, we first describe the experimental setup, and then present and discuss the evaluation results.

The platform used in all the experiments is an Intel® Core™ I7-4790 @ 3.60GHz x 8 with 16GB RAM, running Ubuntu 14.04LTS (Linux) 64-bit. For reproducibility purposes, all supporting codes are publicly available in our research group repository.[10]

4.5.1 Simulation Setup

Different network inventory topologies are randomly created using Networkx library in Python[11]. Table 11 shows information about the different standard algorithms used to create the network topology graphs, including information about the number of nodes, edges, edge probability, and degree.

Specifically, we generate 3 network topology graphs:

- **Binomial Graph**
  Function: $binomial\_graph(n, p)$
  Returns: A $G_{n,p}$ random graph. The $G_{n,p}$ model chooses each of the possible edges with probability p.
  Parameters:

  n: The number of nodes.

  p: Probability for edge creation.

- **Random Regular Graph**
  Function: $random\_regular\_graph(d, n)$
  Returns: A random d-regular graph on n nodes. The resulting graph has no self-loops or parallel edges.
  Parameters:

  d: The degree of each node.

  n: The number of nodes. The value of $n \times d$ must be even.

- **Dense Random Graph**
  Function: $dense\_gnm\_random\_graph(n, m)$
  Returns: A $G_{n,m}$ random graph. In the $G_{n,m}$ model, a graph is chosen uniformly at random from the set of all graphs with n nodes and m edges.
  Parameters:

  n: The number of nodes.

  m: The number of edges.

[10] <https://github.com/intrig-unicamp/ani>
[11] <https://networkx.org/documentation/stable/reference/generators.html>

Network Model	**# Nodes**	**# Edges**	**Edge Prob.**	**Degree**
Binomial Graph	[50, 100, 150, 200, 250]	-	[0.25, 0.5, 0.75, 1, 0.25]	-
Random Regular Graph	[50, 100, 150, 200, 250]	-	-	[10, 20, 30, 40, 50]
Dense Random Graph	[50, 100, 150, 200, 250]	[20, 40, 60, 80, 100]	-	-

Table 11 – Network topology graph algorithms and parameters: number of nodes, number of edges, degree, and edge probability.

In addition, the number of nodes and edges varies between 50-250 and 20-100, respectively. Each pair of nodes are randomly connected with probability that varies between 0.25 and 1 and the degree is a value between 10 and 50. The CPU and bandwidth capacity is a number uniformly distributed between 1-16 and 1-100, respectively. LNIs are created from service requests. Each service has 2 VNFs (*i.e.*, 2 nodes). The CPU demand of each VNF is normally distributed between 1 and 16 and the bandwidth requirement of each link is a number between 1 and 100, uniformly distributed.

4.5.2 Performance Metrics

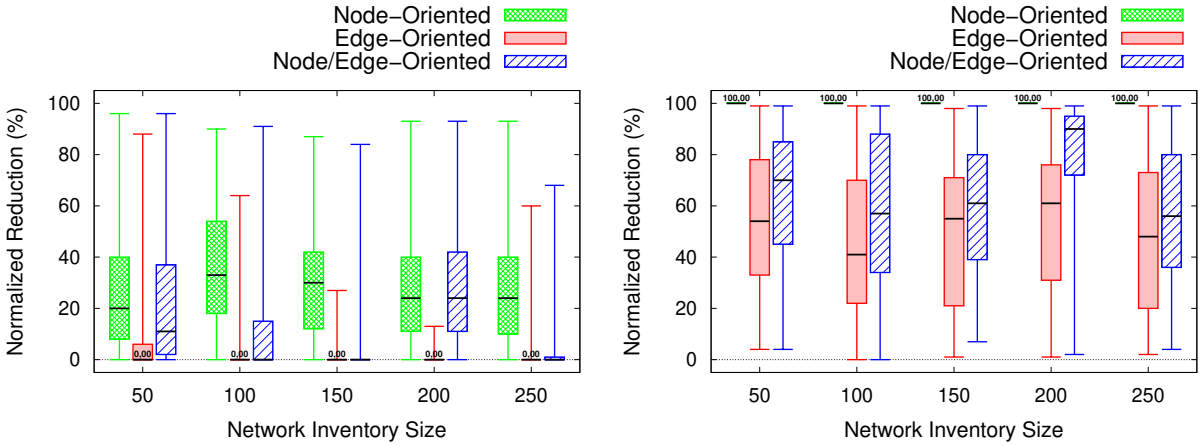
We use two measures of nodes/edges reduction and degree to evaluate the quality of LNIs generated by the three different algorithms: (i) Node-oriented LNI (N-LNI), (ii) E-oriented LNI (E-LNI), (iii) Node/Edge-oriented LNI (N/E-LNI).

For each series of experiments, we randomly generate 100 service requests with two VNFs. A new LNI is generated from a service request and then the percentual node and edge reduction and average degree that a LNI topology obtains, using the three algorithms, is compared to the same metrics in the full network inventory topology (used as a baseline). In case of the average degree evaluation, we are considering two variations: (i) different network inventory topology sizes with a service composed of two VNFs and (ii) same network inventory topology size (100 nodes) but increasing the number of VNFs from 2 to 6.

4.5.3 Simulation Results

4.5.3.1 Binomial Regular Graph

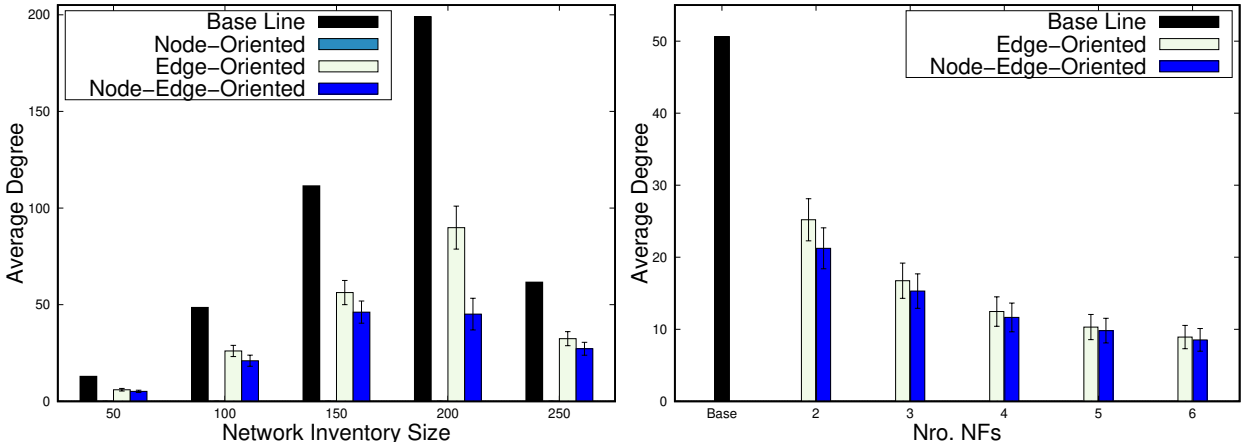
Figures 32 shows the normalized reduction of nodes and edges (as candlesticks with median, quartiles, and max/min values) with different Binomial Graph topology sizes by N-LNI, E-LNI, and N/E-LNI. The node reduction (see Fig. 28a) by N-LNI achieves the higher values, and its edge reduction is 100%. This is because N-LNI only considers nodes supporting the CPU constraints, and edges are discarded. Besides, E-LNI achieves the lowest node reduction value since, it traverses all the nodes without restrictions. On the other hand, the edge reduction measure (see Fig. 28b) by N/E-LNI is higher than E-LNI.



(a) Nodes Reduction (b) Edge Reduction

Figure 28 – **Binomial Graph:** Normalized reduction in nodes (a) and edges (b) using the three algorithms: Node-oriented LNI, Edge-oriented LNI, and Node/Edge-oriented LNI

This behavior is expected because N/E-LNI only considers nodes and edges supporting CPU and bandwidth constraints, respectively.



(a) Degree (b) Degree: 2-6 NFs

Figure 29 – **Binomial Graph:** The average degree at 95% of confidence level obtained from three algorithms (Node-oriented LNI, Edge-oriented LNI, and Node/Edge-oriented LNI) with different topology sizes (a) and with different amount of NFs (b).

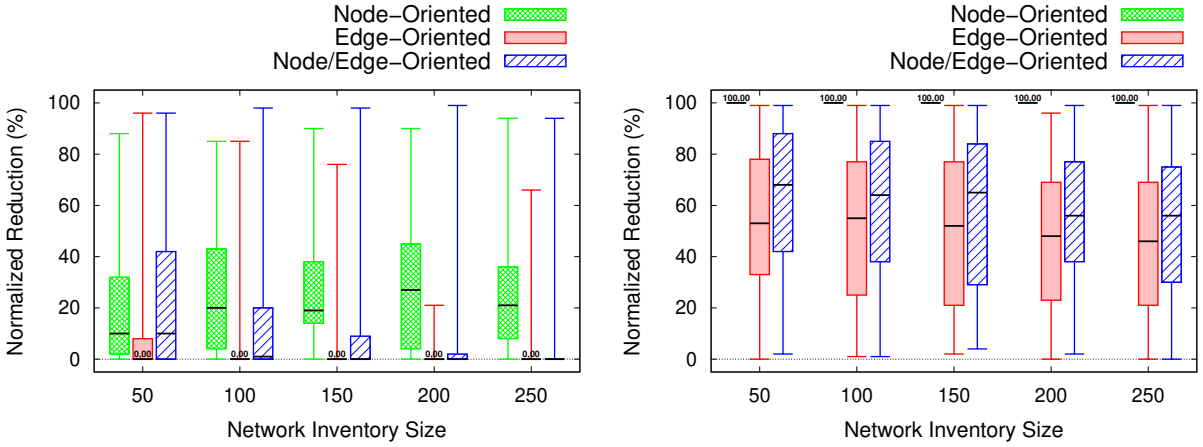
Figure 29a shows the average degree comparison between the three algorithms with different topology sizes. The degree measure is always 0 for N-LNI because edges are no longer considered, and therefore all the nodes are isolated. N/E-LNI achieves a lower degree than E-LNI. This is because, as we just mentioned, N/E-LNI combines both nodes and edge restrictions. A further observation is that, the binomial graph model builds densely connected network topologies (see baseline values), however, E-LNI and N/E-LNI always achieve much lower degree values ($\sim$2.0-2.8x) than the baseline. In addition, for the network topologies with 50, 100, 150, and 200 nodes, the configuration of the edge

probability parameter is gradually increased (0.25, 0.5, 0.75, 1). However, for the network topology with 250 nodes, the edge probability value parameter is reduced to 0.25 (See Table 11), resulting in a drop of the degree value.

To further test the effect of the number of VNFs on the LNI quality, we set services with different amounts of VNFs (2-6). Figure 29b shows the degree comparison between E-LNI and N/E-LNI on a network inventory topology with 100 nodes. As shown in the figure, the degree's value is decreasing as the number of VNFs increases. This is because a new VNF adds new CPU and bandwidth constraints, therefore we can infer that the number of VNFs decreases the nodes and/or edges supporting such requirements.

4.5.3.2 Random Regular Graph

In the Random Regular graph, the N-LNI achieves the higher values in the node reduction results, as shown in Figure 30a. Similar to the Binomial graph, E-LNI achieves the lowest node reduction value since, it traverses all the nodes without restrictions. This is basically because the E-LNI algorithm traverses all the nodes without considering the nodes (CPU) constraints. Besides, the edge reduction (see Fig. 30b) by N-LNI is 100% due to the edges are discarded (it only considers nodes supporting the CPU requirements). Another observation is that the edge reduction by N/E-LNI is higher than E-LNI. This is N/E-LNI considers nodes the CPU and bandwidth constraints to select the nodes and edges, respectively.

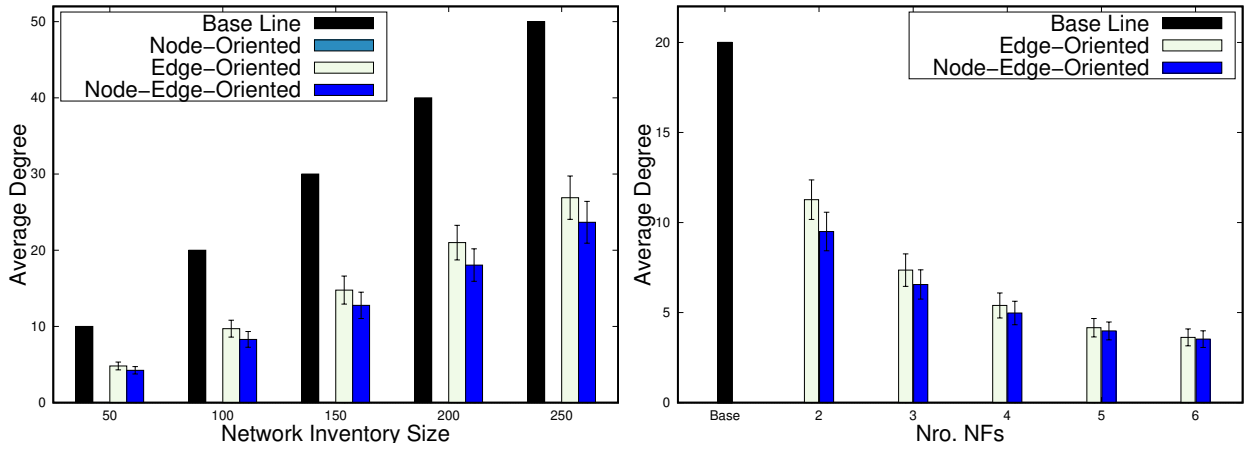


(a) Nodes Reduction

(b) Edge Reduction

Figure 30 – **Random Regular Graph:** Normalized reduction in nodes (a) and edges (b) using the three algorithms: Node-oriented LNI, Edge-oriented LNI, and Node/Edge-oriented LNI

In the case of the average degree comparison (See Fig 31a), one observation is that, as with the Binomial graph, the Random Regular Graph model also builds densely connected topologies (see baseline). In this case, N/E-LNI achieves lower degrees that E-LNI considering than N/E-LNI combines CPU and bandwidth restrictions. Degree values



(a) Degree (b) Degree: 2-6 NFs

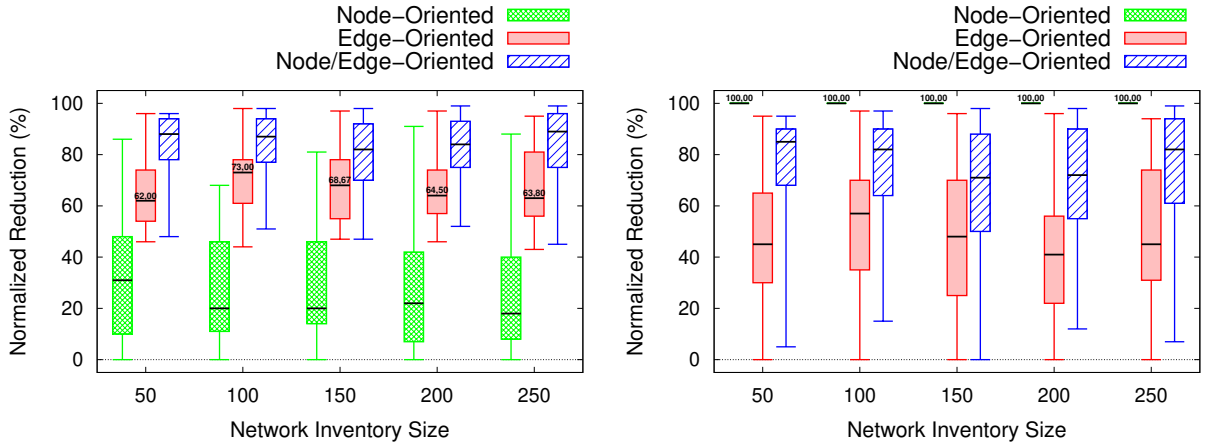
Figure 31 – **Random Regular Graph:** The average degree at 95% of confidence level obtained from three algorithms (Node-oriented LNI, Edge-oriented LNI, and Node/Edge-oriented LNI) with different topology sizes (a) and with different amount of NFs (b).

are always 0 for N-LNI because edges are not considered (*i.e.*, all the nodes are isolated). In addition, the degree value effect with a different number of VNFs in a network service (See Fig 31b) is the same as the Binomial graph model. The degree values are decreasing as the number of VNFs increases because new VNFs add new constraints in terms of CPU (node) and bandwidth (edge).

4.5.3.3 Dense Random Graph

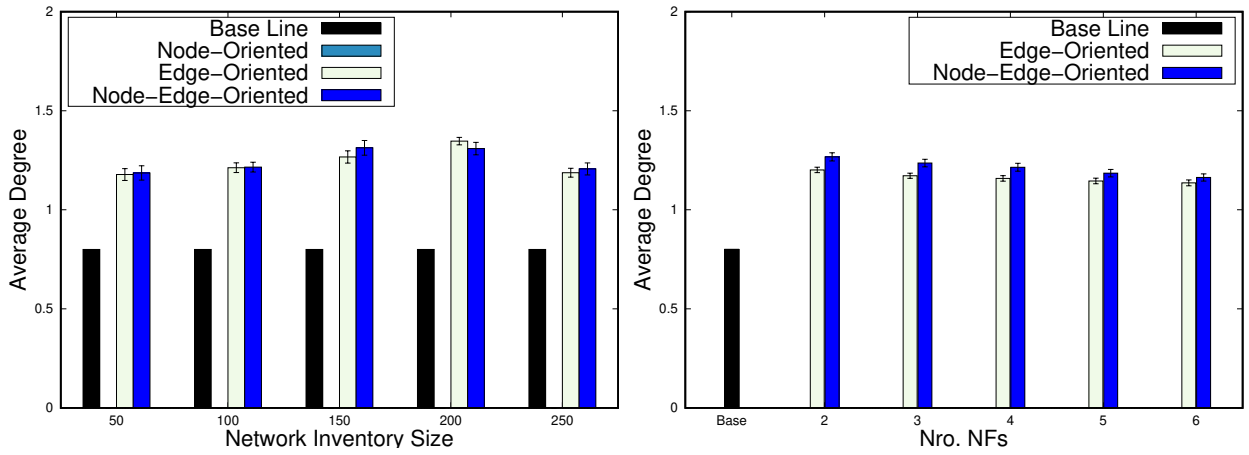
Using the Dense Random Graph model, a graph is created with a specific number of nodes and edges. Therefore, from the configuration values of the edge parameter (See column 3 of Table 11), this graph model does not generate densely connected network topologies (avg. degree = 0.8, See Fig. 33a) with the presence of isolated nodes. This explains why the node reduction by N-LNI achieves the lower values (See Fig. 32a). In other words, E-LNI and N/E-LNI achieve higher values because they discard the isolated nodes. In the case of edge reduction (See Fig. 32b), results are basically the same as in previous experiments, with edge reduction values by N/E-LNI higher than E-LNI.

On the other hand, the average degree comparison between the three algorithms with different topology sizes is shown in Figure 33a. A key observation is that the degree values are increased relative to the baseline (original network topology graph). This is because, E-LNI and N/E-LNI discard all the nodes with degree 0 (*i.e.*, there are not isolated nodes). However, the average degree for all the topologies remains about an average value of ∼1.2. Finally, there is no significant effect when the number of VNFs, in a network service, is increased (See Fig. 33b). The average degree values are between ∼1.1 and ∼1.2 for the E-LNI and N/E-LNI algorithms, respectively. The small increase



(a) Nodes Reduction (b) Edge Reduction

Figure 32 – **Dense Random Graph:** Normalized reduction in nodes (a) and edges (b) using the three algorithms: Node-oriented LNI, Edge-oriented LNI, and Node/Edge-oriented LNI



(a) Degree (b) Degree: 2-6 NFs

Figure 33 – **Dense Random Graph:** The average degree at 95% of confidence level obtained from three algorithms (Node-oriented LNI, Edge-oriented LNI, and Node/Edge-oriented LNI) with different topology sizes (a) and with different amount of NFs (b).

in the average degree value in the N/E-LNI is explained because it generates a small set of disconnected edges compared to the E-LNI algorithm that generates many sets of disconnected edges and, therefore, a small degree.

4.6 Concluding Remarks

One of the foremost challenges for management systems in multi-domain environments is how to effectively handle the scale and complexity of network service placement and management considering actual resource inventories. This chapter contributes with a novel component called Abstracted Network Inventory (ANI) that generates optimized

network views called Logical Network Inventory (LNI) based on network service requirements and network inventory capabilities. Our results reveal that, when using an LNI methodology, we can reduce the time to place network services while optimizing the management of resources by following the principle of abstraction, *i.e.*, by logically reducing the sets of candidate resources in terms of compute nodes and links.

We reckon that, as the deployment size and heterogeneity complexity of softwarized networks increase, properly applying fundamental software principles like layering (*e.g.*, SDN controller foundations), indirection (*e.g.*, overlay tunnels), or abstraction (*e.g.*, ANI/LNI), just to cite a few examples, will be more important than ever to deliver scalable and manageable systems.

On the other hand, this chapter is not also free of limitations and there is a challenging amount of future work to be undertaken. The main deficiencies of this chapter, in our opinion, follow listed below:

- The network model is shown for a basic scenario with only one CPU and bandwidth constraints, therefore, more details are needed to understand how this assumption is justified. Also, even though this chapter briefly mentioned that the network model is easy and totally extensible to consider other constraints, a future analysis also needs to justify what is needed to extend the model for more capabilities and resources.
- The experimental evaluation part only considered network services consisting of 2-6 VNFs, therefore, it is hard to make very strong statements about the resulting efficiency, as proper topology abstraction becomes much harder in more complex services involving multiple VNFs and multiple interconnections. In this context, a more extensive evaluation on more complex network services and more complex infrastructures is in our roadmap.
- Related work discussion is satisfactory, but the state-of-the-art is not well detailed to clearly understand the timeliness and significance of ANI. In order to address this limitation, a future systematic review will discuss other benefits of the ANI component compared to the state-of-the-art, such as privacy (not expose the exact network inventory structure) and incremental updates (because not all infrastructure changes will affect an ANI).
- The presented experiments have been conducted to evaluate the ANI procedures. Even though the qualitative evaluation of the LNI creation is presented and discussed, there is not an evaluation of the ANI component with respect to existing models to show the brought value. Future activities will target to address this important gap as the related work analysis is detailed in this chapter and many different works has tried to deal with network inventories and abstractions.

5 MUDED Use case: 5GEx Information Exchange

5.1 Introduction

The provision of a complete E2E service requires chaining services provided by multiple network operators with multiple technologies. This multi-provider orchestration process requires an advertising mechanism through which single domains can describe their abstract network topology, resource availability (*e.g.*, CPUs, Memory, and Storage), and supported VNFs in an interoperable manner. Moreover, a discovery mechanism is also necessary so that source domains can obtain candidate domains (with the corresponding connectivity information), which can provide a part of the service and/or slice in an E2E service requirement.

The 5G Exchange (5GEx) project (BERNARDOS *et al.*, 2016) aims to enable E2E service orchestration across multiple administrations and multiple technologies. To do so, a Multi-domain logical inter-working architecture is proposed (See Fig. 34), where Multi-domain Orchestrators (MdOs) (at the higher level) are the main entities to exchange functions, information, and control through the inter-operator orchestration APIs (2). In this chapter, we propose a new MUDED-based approach to announce resources and services in the exchange in 5GEx in order to solve some challenges in the distribution of this inter-domain network information. In our prototype, each MdO involved in the federation advertises to the federation layer (acting as a broker) the intra-domain resource and topology information. From this local information, the broker creates an aggregated inter-domain information exposed as a set of abstract and unified Map Services accessible to the MdOs through ALTO-based REST APIs. Moreover, the ANI component is also integrated into the MUDED-based prototype to obtain service-optimized network inventory views to reduce the time to place network services.

The remainder of this chapter is structured as follows. Section 5.2 provides an overview of the 5GEx reference architecture, including the main components. Section 5.3 presents our MUDED-based approach for the 5GEx information exchange with ALTO and ANI components. A prototype implementation based on the aforementioned MUDED architecture is described in section 5.4. Section 5.5 validates the proof of concept with a functional analysis and a performance evaluation in the network service provisioning time. Finally, we present our conclusions in Section 5.6.

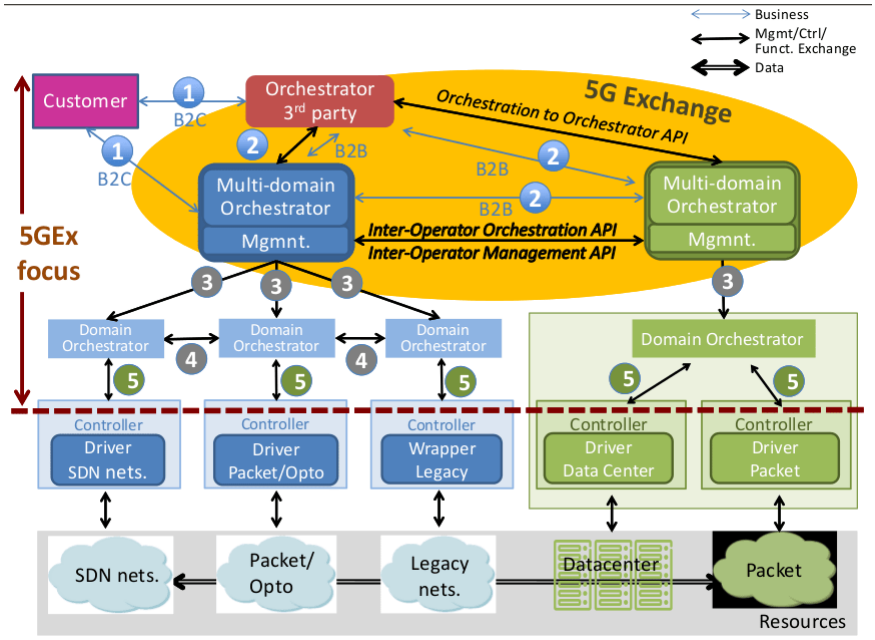


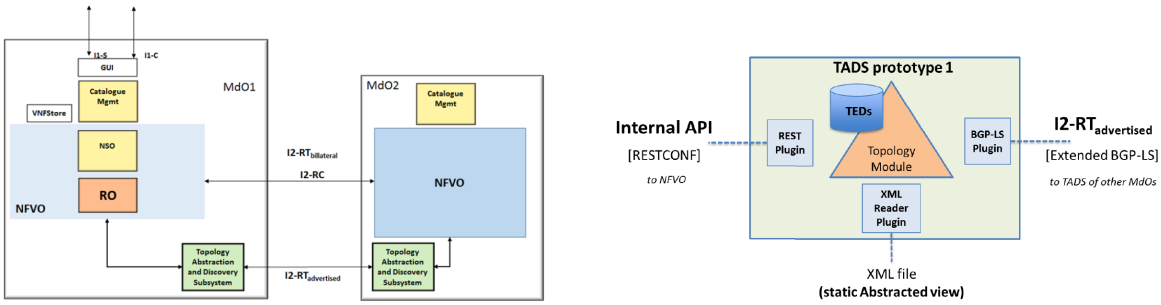
Figure 34 – 5GEx Project Architecture (BERNARDOS *et al.*, 2016)

5.2 Background

The MdO, the core 5GEx component, implements automated deployment of network services spanning across multiple providers and multiple technological domains integrating network and computing. The key MdO module utilized for resource discovery of other providers in the 5GEx community is the Topology Abstraction and Distribution System (TADS). TADS (See Fig. 35a) supports the initial exchange of basic resource availability information such as (i) abstracted topology with the traffic engineering metrics, (ii) 5GEx entry point, and (iii) overall IT information (*i.e.*, CPUs, storage, and memory). The TADS subsystem uses the BGP-LS plugin (Speaker) in the MdO discovery process (See Fig. 35b). Specifically, the BGP-LS plugin (through the I2-RTadvertised interface) is responsible for both exchanging network topology and IT resource information between different MdOs. TADS also includes an XML reading plugging to load the static abstracted view of a local provider. Then, this abstracted information is imported into the correct Traffic Engineering Database (TED).

In the current advertising and discovery process, a number of limitations can be identified:

- **Lack of Abstractions:** Multiple vendors with heterogeneous technologies need an information model to adequately represent in confidentiality-preserving fashion the resource and topology information.
- **Scalability:** Involves the distribution of topology and resource information in a peer-to-peer fashion (MdO-to-MdO). Multi-operator multi-domain environments



(a) 5GEx MdO System Design (b) TADS Architecture

Figure 35 – 5G Exchange Project ((KTH) PAOLO MONTI, 2016)

where the information distribution is advertised in a peer-to-peer model scales linearly. It means the more MdO interconnections one has, the more it "costs" to distribute.

- **Complexity.** Refers to the discovery mechanism to pre-select candidate domains, accounting for resources and capabilities, necessary for an E2E network service deployment. An intrinsic complexity exists in the process of assembling, logically organizing, and enabling abstraction views of different resources and capabilities in multi-domain scenarios.

5.3 MUDED-based Approach

We propose a MUDED-based approach where a broker-plane working on top of MdOs assists the coordinated creation of an E2E network service spanning over multi-operator multi-domain networks. This proposed 5GEx information exchange design resorts in ALTO and ANI to address the lack of abstractions to discover and adequately represent in confidentiality-preserving fashion the abstract network topology, resource availability (*e.g.*, CPUs, Memory, and Storage) and capability (*e.g.*, supported network functions) from different administrative domains.

The proposed brokered design is showed in Figure 36. In this reference architecture, the broker component is conceived to be working as coordinator of a set of MdOs. In turn, a MdO is assumed to manage a set of DOs responsible for various resource domains (featuring physical and virtual, software and hardware components).

The main architectural components are described next:

- ANI Components
  - **Inter-domain Resource (IdR):** It creates a hierarchical database that contains inter-domain resource information such as resource availability (*i.e.*, CPU,

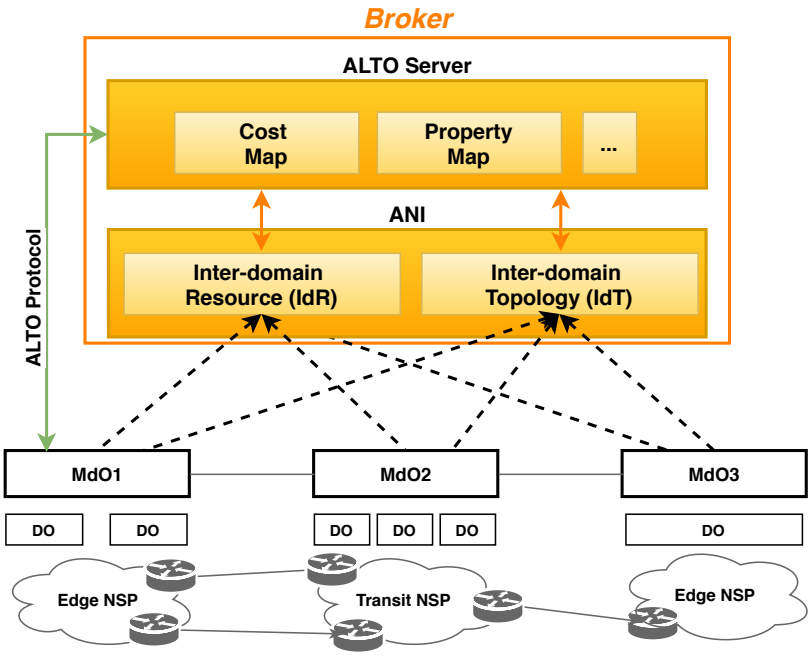


Figure 36 – Broker-assisted Multi-operator Network Architecture

memory, and storage), Virtual Network Functions (VNFs) and Physical Network Functions (PNFs) supported and Service Access Points (SAPs) to access those resources and VNFs/PNFs.

UNIFY (UNIFY D3.2a, 2015), TOSCA (OASIS, 2013), ETSI-NFV (ETSI, 2014), among other data models can be used to create the interface between IdR and MdOs.

– **Inter-domain Topology (IdT):** A hierarchical TED (Traffic Engineering Database) that contains inter-domain network topology information, including additional key parameters (*e.g.*, throughput and latency of links). From this inter-domain TED information, can be created an aggregated domain-level topology map.

The communication between IdT and MdOs components can be done using BGP-LS or REST interfaces.

- ALTO Components

– **ALTO Server:** The ALTO server component is the core of the broker layer. The information collected from the IdR and IdT modules is processed here to create and provide abstract maps with a simplified, yet enough information view about MdOs involved in the federation. This information includes domain-level topology, storage resources, computation resources, networking resources and PNF/VNF capabilities.

– **ALTO Client:** As an ALTO client, each MdO sends ALTO service queries to the ALTO server.

The ALTO server provides aggregated inter-domain information exposed as set ALTO base services defined in (ALIMI *et al.*, 2014), *e.g.*, Network Map, Cost Map and ALTO extension services, *e.g.*, Property Map (ROOME *et al.*, 2020), Multi-Cost Map (RANDRIAMASY *et al.*, 2017), Path Vector (GAO *et al.*, 2020b). In particular, the ALTO Filtered Cost Map extension is introduced in the first version of the ALTO-based Multi-domain Orchestration IETF draft (PEREZ; ROTHENBERG, 2020) with the goal to support the main functionalities in the proposed architecture.

5.4 Prototype Implementation

The strawman use case scenario refers to an E2E network service orchestration involving seven different administrative domains (3 Service Providers (SPs) and 4 Transit Providers (TPs)), as shown in Figure 37. In this section, we provide information about the implementation choices and prototype details such as the MdO components, the Neo4j[1] graph-based database used as the back-end for the ALTO information, and the OpenDaylight[2] (ODL) controller used as ALTO server.

5.4.1 MdO Components

As mentioned above, the MdO functional components and interfaces follow the 5GEx project architectural proposal. Each administrative domain has a MdO to manage resource and/or service orchestration at a multi-operator level (via interface I2 APIs). Within the same administrative domain, each MdO uses emulated DOs (*e.g.*, SDN, Mininet, Openstack, etc.) with emulated I3 interfaces, since no data-plane is present, *i.e.*, DOs use static configuration files to load local information about topology (I3-RT) and resources (I3-RC).

The different MdO components are based on existing open source software tools, for example, ESCAPE[3] and Netphony-topology[4] are used as Resource Orchestrator and Resource Topology, respectively. ESCAPE (Extensible Service ChAin Prototyping Environment) is a framework which supports the development of several parts of the service chaining architecture (*e.g.*, VNF implementation, traffic steering, virtual network embedding, etc.) and it also includes a simple service layer interacting with clients. Netphony-topology is Java-based TED working as a BGP-LS Speaker that contains a Topology

[1] <http://neo4j.com/>
[2] <https://www.opendaylight.org/>
[3] <https://github.com/5GExchange/escape>
[4] <https://github.com/telefonicaid/netphony-topology>

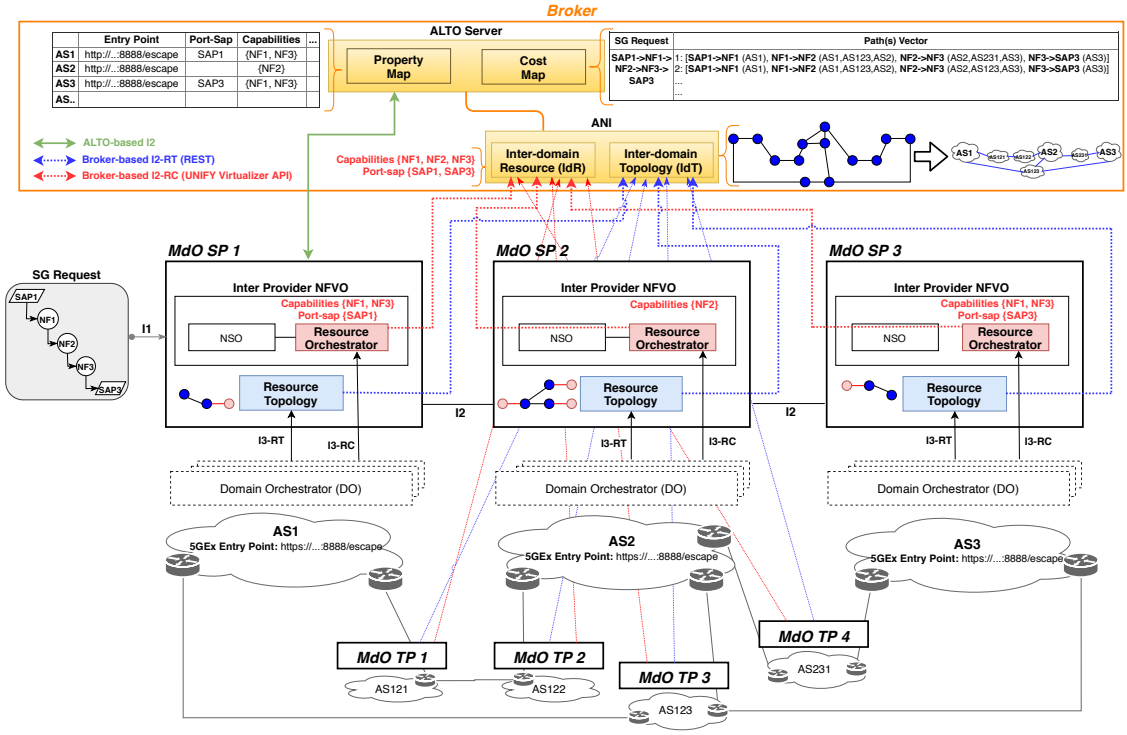


Figure 37 – 5GEx Multi-domain Orchestration Scenario

Module with a collection of TEDs and plugins to export and import the TEDs. Besides, MdOs expose I1 interfaces to the tenants who request services and/or slices, which should follow a Network Function Forwarding Graph (NFFG) (UNIFY D3.2a, 2015) format.

5.4.2 Broker Components

In the case of the broker layer, the IdR and IdT components use the UNIFY Virtualizer API (UNIFY D3.2a, 2015) (broker-based I2-RC API) and REST API (broker-based I2-RT API) respectively, to create the hierarchical databases. From the inter-domain information, two different ALTO Map Services are created: (i) *Property Map* and (ii) *Cost Map*.

- The **Property Map** includes property values grouped by Autonomous System (AS). Such values are SAPs, NFs, and the 5GEx Entry Point (*e.g.*, the URL of the ESCAPE orchestrator).
- The **Cost Map** defines a path vector as an array of ASes, representing the AS-level topological distance between entities (*i.e.*, AS→AS, SAP→SAP, NF→NF, or SAP↔NF). Moreover, as described in the Multi-Cost Map (RANDRIAMASY *et al.*, 2017), path vector constraints can be applied to restricts the response to costs that satisfy a list of simple predicates (*e.g.*, $=$, $>$, $<$, $\geq$, $\leq$). Moreover, it is possible to use a special *"shortest"* predicate provide the shortest path between entities.

When an MdO receives a Service Graph (SG) request, it uses the ALTO server (through ALTO-based I2 APIs) to determine the underlying network graph and a potential set of paths before bilateral negotiation between MdOs is started.

5.4.3 Back-end/Front-end Servers

The resulting data for each broker component (IdR, IdT, and ALTO server) is stored in Neo4j graph-based database (Back-end Server). We opt for this property graph[5] since it provides a natural modeling approach and it uses a key-value store abstraction for JSON object coding. Neo4j is an open-source non-relational graph database implemented in Java, and it supports true ACID transactions, high availability, and scales to billions of nodes and relationships (NEO4J, 2015). Moreover, its native traversal query language, such as Cypher, highly facilitates the development of applications.

The ALTO web server (Front-end Server) has been derived from the ALTO OpenDaylight (ODL) framework. The ALTO server in ODL[6] includes, among other modules, ALTO Northbound providing basic ALTO services as RESTful web services (Northbound APIs) for ALTO client/server communications. ALTO Northbound APIs generate ALTO services from data stored in the MD-SAL data store (an ODL core component). For our implementation, it was necessary to modify the Northbound APIs to generate ALTO services from the data stored in the Neo4j back-end and converts it into the ALTO format specification.

5.4.4 Basic Workflow

A procedural flow for the MUDED-based 5GEx information exchange is shown in Figure 38. The procedure details are explained below:

1. For each MdO, the XML Reader Plugin creates local resource information, such as resource availability (CPU, memory, and storage), VNFs and PNFs, and SAPs.

2. For each MdO, the XML Reader Plugin creates local TED information (including basic IT information and the URL of the local MdO entry point). BGP-LS plugins are off.

3. In the broker component, the IdR requests local resource information (3.1) and creates inter-domain resource database (3.2).

4. In the broker component, the IdT requests local topology information (4.1) and creates inter-domain topology database (4.2).

[5] A graph where (i) vertices and edges can have any number of key/value properties, (ii) there can be many types of relationships between vertices and (iii) edges have a directionality.

[6] <https://wiki-archive.opendaylight.org/view/ALTO:Main>

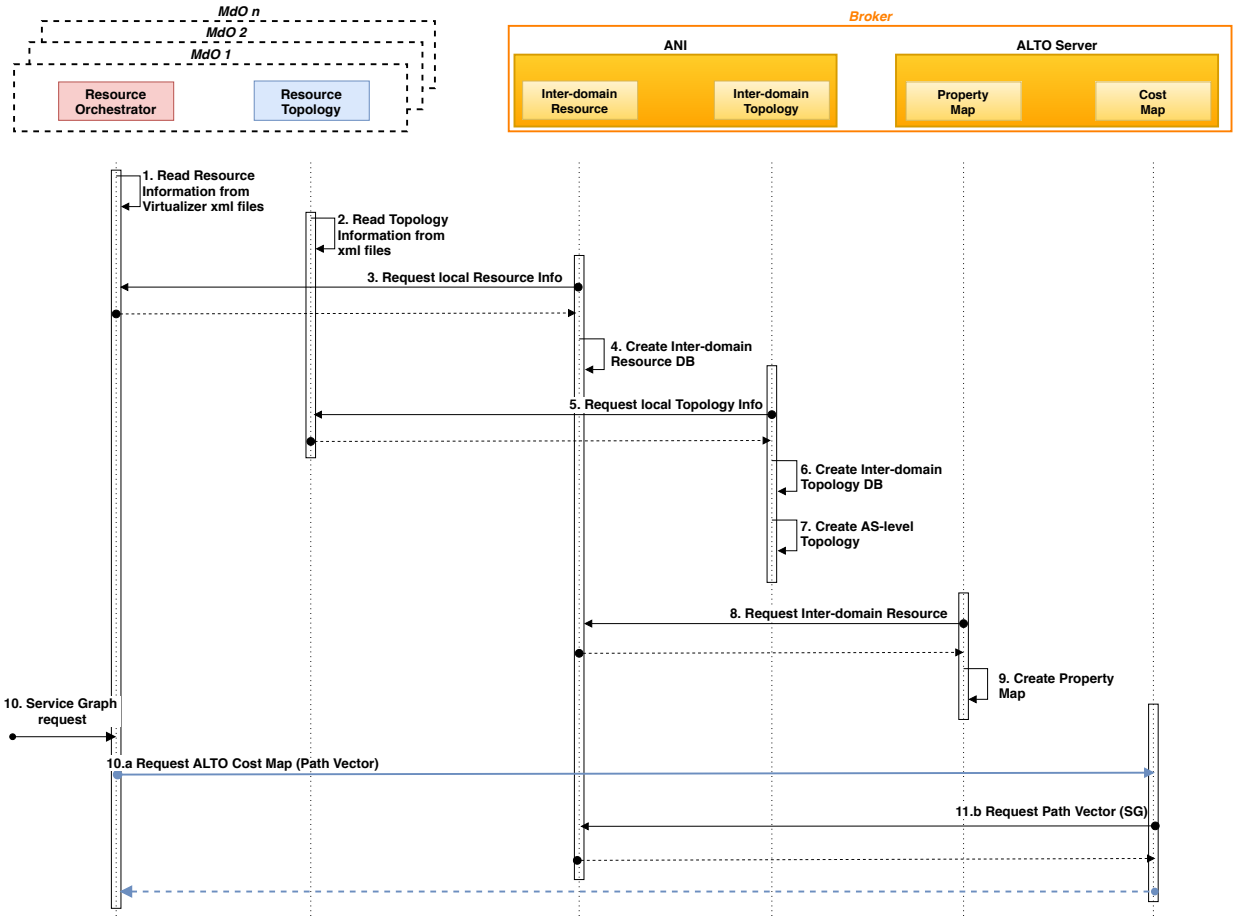


Figure 38 – 5GEx Information Exchange with MUDED: Workflow

5. The broker component requests inter-domain resource information (5.1). Based on this information, the broker component creates and stores Property Maps (with information about the CPU, storage, and memory, and MdO entry points) into the ALTO Server (5.2).

6. Subsequently, It is possible to create Cost Maps (*e.g.*, Multi-Domain Path Resolution). The Cost Map creation is performed on the fly (or reactively) based on MdO requests (ALTO client) asking for a path cost for a network service (6.1 and 6.2).

7. The broker component requests inter-domain path vector information (7.1) and creates Cost Maps into the ALTO Server (7.2).

5.5 Experimental Evaluation

We evaluate the MUDED-based 5GEx prototype by carrying out two different types of experiments[7]: (i) functional behavior, in accordance with the ALTO specifications,

[7] Single server configuration Intel® Core™ I7-4790 @ 3.60GHz x 8 with 16GB RAM, running Ubuntu 14.04LTS (Linux) 64-bit.

and (ii) service placement time using a real-world topology.

5.5.1 Functional Evaluation

For this testing environment, we evaluate whether our ALTO server delivers ALTO services in compliance with ALTO base services defined in RFC7285 (ALIMI *et al.*, 2014), *e.g.*, Network Map, Cost Map and ALTO extension services, *e.g.*, Property Map [DRAFT-PM], Multi-Cost Map [RFC8189], Path Vector [DRAFT-PV]. For that purpose, we used a REST client tool[8] to retrieve ALTO information in JSON format, communicating with the ALTO server via HTTP request.

Examples of the Filtered Property Map and Filtered Cost Map queries and the corresponding responses are featured below.

- **Filtered Property Map Service**

  In this example, the ALTO client wants to retrieve the Property Map for PID entities with the *"unifyslor"* (or MdO entry-point), *"cpu"*, *"mem"*, *"storage"*, *"port"* and *"nf"* properties. The PIDs entities are MdO SP1 (0.0.0.1), MdO TP3 (0.0.0.123), MdO SP2 (0.0.0.2) and MdO SP3 (0.0.0.3).

  – HTTP Request

```
POST /controller/nb/v2/alto/filtered/propertymap/my-default-property-map
Host: 172.28.0.10:8181
Accept: application/alto-propertymapfilter+json,application/alto-error+json

{
    "pids": [ "0.0.0.1", "0.0.0.123","0.0.0.2", "0.0.0.3" ]
}
```

  – HTTP Response

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"
        }
    },
    "property-map": {
        "0.0.0.1": {
            "unifyslor": "https://172.25.0.10:8888/escape",
            "cpu": "50.0",
            "mem": "60.0",
            "storage": "70.0",
            "port": [ "SAP1", "SAP1211", "SAP1231", "SAP2" ],
            "nf": [ "COMPRESSOR", "DECOMPRESSOR" ]
```

[8] <https://www.getpostman.com/>

```
        },
        "0.0.0.123": {
            "unifyslor": "https://172.52.0.10:8888/escape",
            "cpu": "0.0",
            "mem": "0.0",
            "storage": "0.0",
            "port": [ "SAP1231", "SAP1232", "SAP1233" ],
            "nf": []
        },
        "0.0.0.2": {
            "unifyslor": "https://172.26.0.10:8888/escape",
            "cpu": "10.0",
            "mem": "20.0",
            "storage": "30.0",
            "port": [ "SAP1221", "SAP1232", "SAP2311", "port-SAP4" ],
            "nf": [ "FORWARDER" ]
        },
        "0.0.0.3": {
            "unifyslor": "https://172.27.0.10:8888/escape",
            "cpu": "80.0",
            "mem": "90.0",
            "storage": "100.0",
            "port": [ "SAP1233", "SAP2312", "SAP3" ],
            "nf": [ "COMPRESSOR", "DECOMPRESSOR" ]
        }
    }
}
```

Appendix B.1 gives another example of a full Property Map query and the corresponding response.

- **Filtered Cost Map Service: ASes**

  In the Filtered Cost Map below, the ALTO client requests the AS-level topological distance from source AS *"0.0.0.1"* to destination ASes *"0.0.0.2"* and *"0.0.0.3"*. The request also includes a constraint (*"constraints" : [">= 3", "<= 4"]*) to indicate that the ALTO server should only considers AS-level paths for which the number of AS hops are greater than or equal to 3 and less than or equal to 4.

  – HTTP Request

```
POST /controller/nb/v2/alto/costmap/pv
Host: 172.28.0.10:8181
Accept: multipart/related, application/alto-costmap+json,
application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json

{
    "cost-type" :{
            "cost-mode": "array",
            "cost-metric": "ane-path"
    },
```

```
    "pids" : {
        "srcs" : [ "0.0.0.1" ],
        "dsts" : [ "0.0.0.2","0.0.0.3" ]
    },
    "constraints" : [ ">= 3", "<=4"]
}
```

– HTTP Response

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"
        }
    },
    "cost-map": {
        "0.0.0.1": {
            "0.0.0.2": [
                ["0.0.0.1", "0.0.0.121", "0.0.0.122", "0.0.0.2"],
                ["0.0.0.1", "0.0.0.123", "0.0.0.3", "0.0.0.231", "0.0.0.2"]
            ],
            "0.0.0.3": [
                ["0.0.0.1", "0.0.0.123", "0.0.0.2", "0.0.0.231", "0.0.0.3"]
            ]
        }
    }
}
```

Two more examples of the Cost Map service asking the AS-level topological distance without constraints (full Cost Map) and with constraints ["shortest"] are given in Appendix B.2.1 and Appendix B.2.2, respectively.

- **Filtered Cost Map Service: E2E Service Requirements**

  The following example uses the Filtered Cost Map service to request the path vector for a given E2E service requirement. The SG request information is composed of three NFs: (NF1) *"COMPRESSOR"*, (NF2) *"FORWARDER"*, (NF3) *"DECOMPRESSOR"* and, two SAPs (SAP1 and SAP3). Links connecting the NFs and SAPs (*"sg_links"* tag) are also included, followed by an E2E requirement (*"reqs"* tag) with information about the order in which NFs are traversed from SAP1 to SAP3.

  Note that the request includes a constraint (*"constraints" : ["= 9"]*) in order to return just AS-level paths for which the number of AS hops in the E2E requirement is equal to 9.

  – HTTP Request

```
POST /controller/nb/v2/alto/costmap/pv
Host: 172.28.0.10:8181
```

```
Accept: multipart/related, application/alto-costmap+json,
application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json

{
    "cost-type" :{
            "cost-mode": "array",
            "cost-metric": "ane-path"
    },
    "constraints" : [ "= 9"],
    "sg" :{
          "nfs": [ "COMPRESSOR", "FORWARDER", "DECOMPRESSOR"],
          "saps": [ "SAP1", "SAP3" ],
          "sg_links": [ {
                  "id": 1,
                  "src_node": "SAP1",
                  "dst_node": "COMPRESSOR"
                },{
                  "id": 2,
                  "src_node": "COMPRESSOR",
                  "dst_node": "FORWARDER"
                },{
                  "id": 3,
                  "src_node": "FORWARDER",
                  "dst_node": "DECOMPRESSOR",
                },{
                  "id": 4,
                  "src_node": "DECOMPRESSOR",
                  "dst_node": "SAP3"
                }],
          "reqs": [ {
                "src_node": "SAP1",
                "dst_node": "SAP3",
                "sg_path": [ 1, 2, 3, 4 ]
            }
          ]
    }
}
```

– HTTP Response

For each SG link in the E2E requirement (SAP1->COMPRESOR, COMPRESOR->FORWARDER, FORWARDER->DECOMPRESOR, DECOMPRESOR->SAP3), the ALTO server returns sub-arrays indicating potential candidate paths calculated as the AS-level topological distance corresponding to the amount of traversing domains. This AS-level distance is limited to 9 hops as defined by the HTTP request of the above example.

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"
```

```
        }
    },
    "cost-map": {
        "SAP1": {
            "SAP3": {
                "SAP1": {
                    "COMPRESSOR": [
                        [ "0.0.0.1" ]
                    ]
                },
                "COMPRESSOR": {
                    "FORWARDER": [
                        [ "0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2" ]
                      ]
                },
                "FORWARDER": {
                    "DECOMPRESSOR": [
                        [ "0.0.0.2","0.0.0.123","0.0.0.3"],
                        [ "0.0.0.2","0.0.0.231","0.0.0.3"]
                    ]
                },
                "DECOMPRESSOR": {
                    "SAP3": [
                        [ "0.0.0.3" ]
                    ]
                }
            }
        }
    }
}
```

Appendix B.3.1 and Appendix B.3.2 show two more Cost Map service examples asking the AS-level topological distance (for a given E2E requirement), without constraints (full Cost Map) and with constraints ["shortest"], respectively.

5.5.2 Network Service Provisioning

We now focus on quantifying the time it takes for a 5GEx MdO, in terms of execution time, to map a requested network service considering an optimized network inventory (*i.e.*, an LNI).

- **Simulation Setup.** We consider a real wide-area network topology obtained from the Internet Topology Zoo project (KNIGHT *et al.*, 2011). Specifically, we extend the *Interoute* topology — one of Europe's largest cloud service providers[9] — to represent a full network inventory. This network topology is composed of 110 nodes connected through 148 links[10]. As with the previous experiment, node and link capacities are uniformly distributed between 1-16 and 1-100, respectively.

[9] Interoute was acquired by GTT Communications in 2018 (COMMUNICATIONS, 2018)
[10] <http://www.topology-zoo.org/files/Interoute.gml>

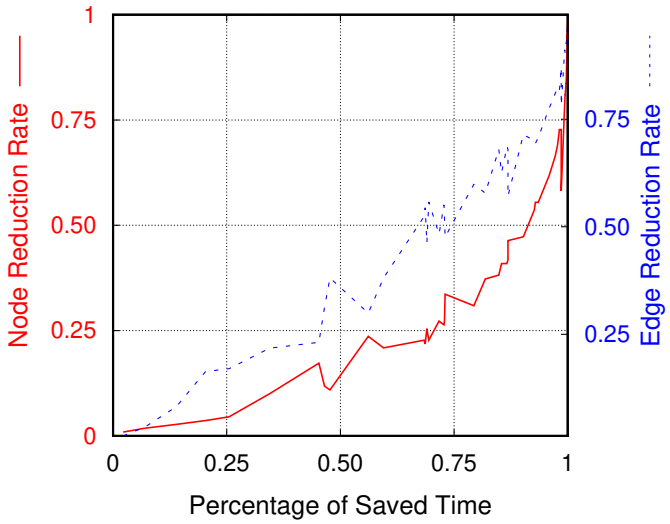


Figure 39 – Time saving rate for mapping a service according to the percentage of reduced nodes and edges using a Node/Edge-oriented LNI.

In case of the MdO, we also use ESCAPE. As aforementioned, ESCAPE is a framework that supports the development of several parts of the service chaining architecture and it can run a number of emulated LOs with emulated interfaces to load information about a local network inventory. ESCAPE also includes a simple service layer where users can request services.

- **Performance Metrics.** ESCAPE uses a heuristic-based greedy backtracking algorithm to map service requests to a network inventory topology. In this experiment, we measure the ESCAPE's time-saving rate for mapping of a service. This value is defined as a fraction of the amount of saved time using a LNI topology (generated by Node/Edge-oriented LNI algorithm) out of running time using a full network inventory topology. We generate 50 different LNIs from 50 service requests. Each service is with two NFs where the CPU and bandwidth demands are normally distributed between 1 and 16 and between 1 and 100, respectively.

- **Simulation Results.** Figure 39 presents the average saving time at 95% of confidence level according to the percentage of reduced nodes and edges when using a Node/Edge-oriented LNI and when using a full network inventory. Results indicate significant improvements in terms of saving time when the orchestrator uses LNI compared to the approach in which a full topology is used. More specifically, the key observation here is: a CO can achieve up to 50% time saving for service deployment with the reduction of less than 20% of nodes in a network inventory. Even, with a reduction of nodes and edges by less than 55%, it is possible to obtain up to 75% time-saving rate.

5.6 Concluding Remarks

Evolving networking environments require the provision of value-added services in multi-domain (multi-operator/multi-technology) scenarios. In this work, we designed and implemented a use case prototype inspired in the MUDED-based approach through which single domains can describe their resource and network capabilities in an interoperable manner. The experiments presented bring essential guidelines towards potential benefits to the challenges of multi-domain orchestration (*e.g.*, lack of abstractions, scalability, and flexibility) by leveraging the map services and generality of the ALTO protocol and the proper abstraction mechanisms of the ANI component.

Although the MUDED-assisted information exchange has several advantages, it also has some limitations. Preponderantly, this chapter should include future research to extend the studies of the following shortcomings:

- The MUDED entity should have independence from the operators, but it is not clear what kind of organization will manage and support the operation of the broker. In addition, if a broker is used to exchange information, then it should be ensured that the data delivered amongst the operators by this 3rd party is not changed or manipulated. Therefore, further analysis of the business aspects of MUDED system is necessary. In this context, the broker entity must be trusted by each operator since it stores and handles sensitive information. For example, future deployment of SDN at IXPs can be used as a trusted third-party platform to support rich business models between different operators (LI *et al.*, 2020a).

- In the case of peer-to-peer information exchange model, a MdO failure concerns only the domain where the failure occurs, other peers can perform the information exchange without any limitation. However, If any error occurs in the broker entity, the information exchange among all involved domains will be impacted. Future activities will consider a systematic review of different mechanisms to avoid this single point of failure (*e.g.*, local restoration/replication options).

- The MdO information exchange depends on the policies. Operators have a preference to share a different view about their compute and network resources towards different operators. For example, a detailed view for the operators that are belonging to same operator group and a high-level information towards the other operators. No explored in this chapter, this gap is planned for future work in order to know how the fine-grained/coarse-grained information exchange will be handled.

- This chapter can improve its studies by performing the experimental evaluation in more complex/real environments, instead of performing all the tests on a single physical machine using containers.

- Finally, in the network service provisioning experiment (Section 5.5.2), the time-saving rates only consider the time needed for placing the service, *i.e.*, it is not considered the time needed for generating the abstractions. Future experiments will include an analysis of the total time of (i) generating the LNI and (ii) placing the service in the LNI, and compare that to placing the service directly in the full network inventory topology.

6 Conclusion & Future Work

In evolving networking scenarios (*e.g.*, 5G), where applications provide softwarized services requiring resource orchestration across multiple network domains, network and application integration (NAI) is fundamental. The networking community (academy and industry) has pointed at the importance of regularly revisiting this topic to identify its possibilities and challenges (SIGCOMM'20: Workshop on Network-Application Integration/CoDesign[1]). Throughout this thesis, the entire line of reasoning fits according to solve the main barriers to systematically release NAI, all of which are addressed in Chapters 2, 3, 4, and 5. After elaborating the final thoughts of the core chapters, herein, we answer the research questions, and finally, the validation of the hypothesis presented in Chapter 1.

According to NAI analysis in Chapter 2 and use case experimental evaluation in Chapter 5, we contribute to answering the research question #1: *What are the possibilities of a much stronger network & application collaboration than the current mainstream networking?* our analysis suggests many possibilities in designing and implementing NAI by application-aware networking and network-aware applications. This assumption is validated by different standard proposals, research contributions, and real deployment examples in both approaches presented. Besides, we design and evaluate MUDED, a multi-domain NAI possibilities discovery and exposure framework to address the key barriers of systematically realizing NAI. The key components of MUDED include a unified, abstract representation of network information using mathematical programming constraints, a resource discovery language for applications to express their intents on discovering network information, and an efficient service-optimized network inventory component. Still, we understand that much more work needs to be elaborated to establish a robust methodology applied in other realistic multi-domain use cases, especially when handling run-time network service operations.

By tackling objectives **O1**-**O2**, this work demonstrates how barriers **B1** and **B2** can be removed, taking advantage of maturing NAI protocols such as ALTO (and extensions). At the same time, it allows us to suggest an answer to research question #2: *How to expose and discover multi-domain NAI possibilities using the IETF ALTO protocol?*. Our study summarises the benefits of using multi-domain information and discusses the ALTO design issues for discovering and exposing it. Besides, we also present key design requirements to be addressed in order to realize the proposal of providing multi-domain information by ALTO services. Accordingly, we suggest improvements to be performed in the ALTO base protocol together with our ongoing standardization efforts in order to

[1] <https://conferences.sigcomm.org/sigcomm/2020/workshop-nai.html>

realize the proposal of discovering and exposing multi-domain information. Regarding our proposed solutions, while they are not mature enough to have an immediate impact on industry standards, many of them are in an excellent position to be adopted in relevant bodies like IETF.

Towards addressing the scalability concerns **(O3)** and remove the barrier **(B3)**, the ANI component is designed and implemented. It suggests the ideas to answer the research question #3: *How to effectively handle the scale and complexity of multi-domain environments to create proper abstract network views?*. From the proper abstraction method of the network inventory infrastructure presented in the ANI component and its subsequent analysis and experimental evaluation, we reckon that our novel methodology, taking into account the service requirements as a second input, defines an effective manner to create logical network inventories (*i.e.*, LNIs) to deal with the deployment size and heterogeneity complexity of multi-domain softwarized networks. Experimental evaluations reveal that, when using an LNI methodology, we can optimize the management of resources while reducing the time to place network services. Also, ANI emerged as a per patent application which, when filed, it was incorporated into the MUDED system, demonstrating the two-way transfer of benefits between academy and industry.

In line with the research questions explored, this thesis investigated the hypothesis that *"**Application and network integration** is a key component for multi-domain settings, which for widely use should consider generic and standard mechanisms satisfying the ever so important features of **NAI possibilities exposure and discovery**, along with **addressing the scalability** and performance concerns"*. In a distinctive way, three argumentative lines make up the scope of this thesis, as elucidated in the core chapters (*i.e.*, 2, 3, 4, and 5). Each one of them (NAI, ALTO, ANI, Use Case) contribute to the confirmation of the hypothesis above. Chapters 2 and 5 review the possibilities of NAI through application-aware networking and network-aware applications, and design and evaluate a multi-domain generic framework for NAI possibilities exposure and discovery, called MUDED. Chapter 3 elaborates key design requirements of ALTO for exposing multi-domain information along with a set of generic mechanisms to design a multi-domain ALTO framework. Chapter 4 proposes the ANI component that implements a novel method to deal with the scalability issues in multi-domain softwarized environments. Consequently, based on the answers given to the research questions posed, we suggest our hypothesis is validated through all the content of this thesis.

Finally, as prominent future work, a set of elaborated shortcomings was identified in the core chapters of this thesis. Among them, we can highlight overall objectives, such as: (i) identify a sorted list of the proposed ALTO extensions to be considered in the next ALTO chapter; (ii) more extensive experimental evaluation of the ANI component using more complex network services and network infrastructures; and (iii) a prototype

evolution of the MUDED platform to consider different information exchange models and complex run-time service deployment operations. In addition, it will also be beneficial to extend the proposed offline algorithms (or propose new ones) to online algorithms implemented via heuristics to process single requests arriving over time one after another. Finally, MUDED defines core components and northbound interfaces; however, the system design needs further study regarding the southbound interfaces. Future research activities in MUDED will include different southbound interfaces such as intra/inter protocols (ISIS, OSPF, BGP) and flow-based protocols (NetFlow, sFlow) in order to provide flexibility and obtain up to date network information.

Bibliography

23009, I. *Dynamic Adaptive Streaming over HTTP*. 2017. Https://mpeg.chiariglione.org/standards/mpeg-dash. Available from Internet: <https://mpeg.chiariglione.org/standards/mpeg-dash>. Cited on page 37.

3GPP. *Management and orchestration; provisioning*. 2017. Technical report, TS 28.351. Cited on page 33.

3GPP. *Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)*. 2019. Technical report, TS 26.247. Cited on page 36.

3GPP. *System architecture for the 5G System (5GS)*. 2020. Technical report, TS 23.501. Cited on page 36.

5G-TRANSFORMER. *5G-Transformer – 5G Mobile Transport Platform for Vertical*. 2017. Http://5g-transformer.eu/. Available from Internet: <http://5g-transformer.eu/>. Cited 3 times on pages 50, 51, and 52.

AL., M. B. et. *HEPiX Network Functions Virtualisation Working Group Report*. Zenodo, 2020. Available from Internet: <https://doi.org/10.5281/zenodo.3741402>. Cited on page 19.

ALIMI, R.; PENNO, R.; YANG, Y.; KIESEL, S.; PREVIDI, S.; ROOME, W.; SHALUNOV, S.; WOUNDY, R. *Application-Layer Traffic Optimization (ALTO) Protocol*. [S.l.], 2014. <http://www.rfc-editor.org/rfc/rfc7285.txt>. Available from Internet: <http://www.rfc-editor.org/rfc/rfc7285.txt>. Cited 9 times on pages 21, 22, 36, 40, 49, 53, 62, 93, and 97.

ALLIANCE, N. 5g white paper. *Next generation mobile networks, white paper*, p. 1–125, 2015. Cited 2 times on pages 19 and 49.

ASSOCIATION, G. *Generic network slice template*. 2019. Https://www.gsma.com/newsroom/wp-content/uploads//NG.116-v2.0.pdf. Available from Internet: <https://www.gsma.com/newsroom/wp-content/uploads//NG.116-v2.0.pdf>. Cited on page 33.

ASSOCIATION, G. *Generic Network Slice Template*. 2019. <https://www.gsma.com/newsroom/wp-content/uploads//NG.116-v2.0.pdf>. Cited on page 64.

AT&T. *AT&T Edge Cloud (AEC) - White Paper*. 2017. Available from Internet: <https://about.att.com/ecms/dam/innovationdocs/Edge_Compute_White_Paper%20FINAL2.pdf>. Cited on page 71.

BERNARDOS, C.; CONTRERAS, L.; VAISHNAVI, I.; SZABO, R.; LI, X.; PAOLUCCI, F.; SGAMBELLURI, A.; MARTINI, B.; VALCARENGHI, L.; LANDI, G.; ANDRUSHKO, D.; MOURAD, A. *Multi-domain Network Virtualization*. [S.l.], 2018. <http://www.ietf.org/internet-drafts/draft-bernardos-nfvrg-multidomain-05.txt>. Available from Internet: <http://www.ietf.org/internet-drafts/draft-bernardos-nfvrg-multidomain-05.txt>. Cited on page 50.

BERNARDOS, C. J.; GERÖ, B. P.; GIROLAMO, M. D.; KERN, A.; MARTINI, B.; VAISHNAVI, I. 5gex: realising a europe-wide multi-domain framework for software-defined infrastructures. *Transactions on Emerging Telecommunications Technologies*, Wiley Online Library, v. 27, n. 9, p. 1271–1280, 2016. Cited 5 times on pages 10, 50, 51, 89, and 90.

BHAMARE, D.; JAIN, R.; SAMAKA, M.; ERBAD, A. A survey on service function chaining. *Journal of Network and Computer Applications*, Elsevier, v. 75, p. 138–155, 2016. Cited on page 71.

BLACK, D. L. *Relaxing Restrictions on Explicit Congestion Notification (ECN) Experimentation*. RFC Editor, 2018. RFC 8311. (Request for Comments, 8311). Available from Internet: <https://rfc-editor.org/rfc/rfc8311.txt>. Cited on page 37.

BRAY, T. *The JavaScript Object Notation (JSON) Data Interchange Format*. RFC Editor, 2017. RFC 8259. (Request for Comments, 8259). Available from Internet: <https://rfc-editor.org/rfc/rfc8259.txt>. Cited on page 56.

CAMPANELLA, M.; KRZYWANIA, R.; REIJS, V.; WILSON, D.; SEVASTI, A.; STAMOS, K.; TZIOUVARAS, C. Bandwidth on demand services for european research and education networks. In: IEEE. *2006 IEEE First International Workshop on Bandwidth on Demand*. [S.l.], 2006. p. 65–72. Cited 2 times on pages 20 and 30.

CHEN, J.; AMMAR, M.; FAYED, M.; FONSECA, R. Client-driven network-level qoe fairness for encrypted'dash-s'. In: *Proceedings of the 2016 workshop on QoE-based Analysis and Management of Data Communication Networks*. [S.l.: s.n.], 2016. p. 55–60. Cited on page 19.

CLEMM, A.; CIAVAGLIA, L.; GRANVILLE, L.; TANTSURA, J. *Intent-Based Networking - Concepts and Definitions*. [S.l.], 2020. <http://www.ietf.org/internet-drafts/draft-irtf-nmrg-ibn-concepts-definitions-01.txt>. Available from Internet: <http://www.ietf.org/internet-drafts/draft-irtf-nmrg-ibn-concepts-definitions-01.txt>. Cited on page 64.

CMS. The CMS experiment at the CERN LHC. *Journal of Instrumentation*, v. 3, n. 08, 2008. ISSN 1748-0221. Cited on page 49.

COMMUNICATIONS, G. *GTT Completes Acquisition of Interoute*. 2018. Https://www.gtt.net/gb-en/news/press-releases/gtt-completes-acquisition-of-interoute/. Available from Internet: <https://www.gtt.net/gb-en/news/press-releases/gtt-completes-acquisition-of-interoute/>. Cited on page 101.

DOLSON, D.; HOMMA, S.; LOPEZ, D.; BOUCADAIR, M. *Hierarchical Service Function Chaining (hSFC)*. [S.l.], 2018. Cited on page 50.

DONG, J.; CHEN, M.; DHODY, D.; TANTSURA, J.; KUMAKI, K.; MURAI, T. *BGP Extensions for Path Computation Element (PCE) Discovery*. [S.l.], 2017. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-dong-pce-discovery-proto-bgp-07>. Cited 2 times on pages 64 and 67.

DULINSKI, Z.; WYDRYCH, P.; STANKIEWICZ, R. *Inter-ALTO Communication Problem Statement*. [S.l.], 2015. Work in Progress. Available from Internet: <https:

//datatracker.ietf.org/doc/html/draft-dulinski-alto-inter-problem-statement-02>. Cited 2 times on pages 64 and 65.

ERICSSON. *Distributed cloud – a key enabler of automotive and industry 4.0 use cases*. 2018. Https://www.ericsson.com/en/ericsson-technology-review/archive/2018/distributed-cloud. Available from Internet: <https://www.ericsson.com/en/ericsson-technology-review/archive/2018/distributed-cloud>. Cited on page 71.

ETSI. *Network Functions Virtualisation (NFV) Management and Orchestration V1.1.1*. 2014. Http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf. Cited on page 92.

ETSI. *Network Functions Virtualisation (NFV); Management and Orchestration; Network Service Templates Specification* . 2016. Https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/014/02.01.01_60/gs_NFV-IFA014v020101p.pdf. Available from Internet: <https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/014/02.01.01_60/gs_NFV-IFA014v020101p.pdf>. Cited on page 32.

ETSI. *Mobile edge computing (MEC);deployment of mobile edge computing in an NFV environment*. 2018. Https://www.etsi.org/deliver/etsi_gr/MEC/001_099/017/01.01.01_60/gr_MEC017v010101p.pdf. Available from Internet: <https://www.etsi.org/deliver/etsi_gr/MEC/001_099/017/01.01.01_60/gr_MEC017v010101p.pdf>. Cited on page 33.

ETSI. *Multi-access edge computing (MEC); framework and reference architecture*. 2019. Https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf. Available from Internet: <https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf>. Cited on page 33.

ETSI, G. 004,"zero-touch network and service management (zsm). *Reference Architecture*, 2020. Cited 3 times on pages 19, 49, and 58.

ETSI, N. Network functions virtualisation (nfv); management and orchestration; network service templates specification. *ETSI, Group Specification ETSI GS NFV-IFA*, v. 14, p. V2. Cited on page 64.

European Telecommunication Standards Institute. *Report on architecture options to support multiple administrative domains V3.1.1*. 2018. Available from Internet: <http://www.etsi.org/deliver/etsi_gr/NFV-IFA/001_099/028/03.01.01_60/gr_NFV-IFA028v030101p.pdf>. Cited on page 50.

FERGUSON, A. D.; GUHA, A.; LIANG, C.; FONSECA, R.; KRISHNAMURTHI, S. Participatory networking: An api for application control of sdns. *ACM SIGCOMM computer communication review*, ACM New York, NY, USA, v. 43, n. 4, p. 327–338, 2013. Cited on page 19.

FERGUSON, A. D.; GUHA, A.; LIANG, C.; FONSECA, R.; KRISHNAMURTHI, S. Participatory networking: An api for application control of sdns. In: *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. New York, NY, USA: Association

for Computing Machinery, 2013. (SIGCOMM '13), p. 327–338. ISBN 9781450320566. Available from Internet: <https://doi.org/10.1145/2486001.2486003>. Cited 2 times on pages 33 and 34.

FIORANI, M.; ROSTAMI, A.; WOSINSKA, L.; MONTI, P. Transport abstraction models for an sdn-controlled centralized ran. *IEEE Communications Letters*, IEEE, v. 19, n. 8, p. 1406–1409, 2015. Cited 4 times on pages 21, 72, 73, and 74.

FIORANI, M.; ROSTAMI, A.; WOSINSKA, L.; MONTI, P. Abstraction models for optical 5g transport networks. *IEEE/OSA Journal of Optical Communications and Networking*, IEEE, v. 8, n. 9, p. 656–665, 2016. Cited 4 times on pages 21, 72, 73, and 74.

GAO, K.; CONTRERAS, L. M.; RANDRIAMASY, S. Bi-directional network and application interaction: Application intents upon abstracted network information (invited paper). In: *Proceedings of the Workshop on Network Application Integration/CoDesign*. New York, NY, USA: Association for Computing Machinery, 2020. (NAI '20), p. 43–50. ISBN 9781450380447. Available from Internet: <https://doi.org/10.1145/3405672.3409491>. Cited 5 times on pages 11, 19, 32, 67, and 68.

GAO, K.; LEE, Y.; RANDRIAMASY, S.; YANG, Y. R.; ZHANG, J. J. *ALTO Extension: Path Vector*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-ietf-alto-path-vector-11>. Cited 4 times on pages 40, 42, 55, and 93.

GAO, K.; XINWANG2014@HOTMAIL.COM; XIANG, Q.; GU, C.; YANG, Y. R.; CHEN, G. *Compressing ALTO Path Vectors*. [S.l.], 2018. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-gao-alto-routing-state-abstraction-08>. Cited 2 times on pages 64 and 68.

GREDLER, H.; MEDVED, J.; PREVIDI, S.; FARREL, A.; RAY, S. *North-Bound Distribution of Link-State and Traffic Engineering (TE) Information Using BGP*. RFC Editor, 2016. RFC 7752. (Request for Comments, 7752). Available from Internet: <https://rfc-editor.org/rfc/rfc7752.txt>. Cited 3 times on pages 64, 65, and 66.

GUERZONI, R.; VAISHNAVI, I.; CAPARROS, D. P.; GALIS, A.; TUSA, F.; MONTI, P.; SGANBELLURI, A.; BICZÓK, G.; SONKOLY, B.; TOKA, L. *et al.* Analysis of end-to-end multi-domain management and orchestration frameworks for software defined infrastructures: an architectural survey. *Transactions on Emerging Telecommunications Technologies*, Wiley Online Library, v. 28, n. 4, p. e3103, 2017. Cited on page 71.

GUPTA, A.; VANBEVER, L.; SHAHBAZ, M.; DONOVAN, S. P.; SCHLINKER, B.; FEAMSTER, N.; REXFORD, J.; SHENKER, S.; CLARK, R.; KATZ-BASSETT, E. Sdx: A software defined internet exchange. *ACM SIGCOMM Computer Communication Review*, ACM, v. 44, n. 4, p. 551–562, 2015. Cited 3 times on pages 19, 49, and 60.

HALPERN, J.; PIGNATARO, C. *Service Function Chaining (SFC) Architecture*. [S.l.], 2015. Cited 3 times on pages 19, 49, and 75.

HECKMANN, O.; BOCK, A. *The edonkey 2000 protocol*. [S.l.], 2002. Cited on page 54.

HEORHIADI, V.; REITER, M. K.; SEKAR, V. Simplifying software-defined network optimization using {SOL}. In: *13th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 16)*. [S.l.: s.n.], 2016. p. 223–237. Cited on page 44.

HINDMAN, B.; KONWINSKI, A.; ZAHARIA, M.; GHODSI, A.; JOSEPH, A. D.; KATZ, R.; SHENKER, S.; STOICA, I. Mesos: A platform for fine-grained resource sharing in the data center. In: *NSDI*. [S.l.: s.n.], 2011. Cited on page 21.

HOLZMAN, B.; BAUERDICK, L. A.; BOCKELMAN, B.; DYKSTRA, D.; FISK, I.; FUESS, S.; GARZOGLIO, G.; GIRONE, M.; GUTSCHE, O.; HUFNAGEL, D. *et al.* Hepcloud, a new paradigm for hep facilities: Cms amazon web services investigation. *Computing and Software for Big Science*, Springer, v. 1, n. 1, p. 1, 2017. Cited on page 57.

JARRAYA, Y.; MADI, T.; DEBBABI, M. A Survey and a Layered Taxonomy of Software-Defined Networking. *IEEE Communications Surveys & Tutorials*, v. 16, n. 4, p. 1955–1980, 2014. ISSN 1553-877X. Available from Internet: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6805151>. Cited on page 71.

JIANG, J.; SEKAR, V.; ZHANG, H. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In: *Proceedings of the 8th international conference on Emerging networking experiments and technologies*. [S.l.: s.n.], 2012. p. 97–108. Cited on page 19.

KARAGIANNIS, T.; RODRIGUEZ, P.; PAPAGIANNAKI, K. Should internet service providers fear peer-assisted content distribution? In: *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*. [S.l.: s.n.], 2005. p. 6–6. Cited 2 times on pages 19 and 30.

KARMARKAR, N. A new polynomial-time algorithm for linear programming. In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. [S.l.: s.n.], 1984. p. 302–311. Cited on page 45.

KATSALIS, K.; NIKAEIN, N.; EDMONDS, A. Multi-domain orchestration for nfv: Challenges and research directions. In: IEEE. *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*. [S.l.], 2016. p. 189–195. Cited 3 times on pages 19, 49, and 58.

KIESEL, S.; PREVIDI, S.; STIEMERLING, M.; WOUNDY, R.; YANG, Y. R. *Application-Layer Traffic Optimization (ALTO) Requirements*. RFC Editor, 2012. RFC 6708. (Request for Comments, 6708). Available from Internet: <https://rfc-editor.org/rfc/rfc6708.txt>. Cited on page 54.

KIESEL, S.; STIEMERLING, M. *Application-Layer Traffic Optimization (ALTO) Cross-Domain Server Discovery*. RFC Editor, 2020. RFC 8686. (Request for Comments, 8686). Available from Internet: <https://rfc-editor.org/rfc/rfc8686.txt>. Cited 3 times on pages 54, 64, and 66.

KIM, C.; SIVARAMAN, A.; KATTA, N.; BAS, A.; DIXIT, A.; WOBKER, L. J. In-band network telemetry via programmable dataplanes. In: *ACM SIGCOMM*. [S.l.: s.n.], 2015. Cited on page 36.

KING, D.; FARREL, A. *The Application of the Path Computation Element Architecture to the Determination of a Sequence of Domains in MPLS and GMPLS*. RFC Editor, 2012. RFC 6805. (Request for Comments, 6805). Available from Internet: <https://rfc-editor.org/rfc/rfc6805.txt>. Cited 3 times on pages 64, 65, and 66.

KNIGHT, S.; NGUYEN, H. X.; FALKNER, N.; BOWDEN, R.; ROUGHAN, M. The internet topology zoo. *IEEE Journal on Selected Areas in Communications*, IEEE, v. 29, n. 9, p. 1765–1775, 2011. Cited on page 101.

KREUTZ, D.; RAMOS, F. M. V.; VERISSIMO, P. E.; ROTHENBERG, C. E.; AZODOLMOLKY, S.; UHLIG, S. Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, v. 103, n. 1, p. 14–76, 1 2015. ISSN 0018-9219. Available from Internet: <http://ieeexplore.ieee.org/document/6994333/>. Cited on page 71.

(KTH) PAOLO MONTI, K. F. C. K. J. M. H. A. A. R. A. B. S. B. J. C. B. R. S. E. G. G. H. A. M. H. I. V. H. D. M. R. O. G. d. D. T. J. M. G. U. C. J. B. C. U. F. T. U. A. S. *Deliverable 3.5: Software Prototype Documentation and User Manual*. [S.l.], 2016. Cited 2 times on pages 10 and 91.

KUZMANOVIC, A.; KNIGHTLY, E. W. Tcp-lp: low-priority service via end-point congestion control. *IEEE/ACM Transactions on Networking*, IEEE, v. 14, n. 4, p. 739–752, 2006. Cited 2 times on pages 19 and 30.

LACHOS, D.; ROTHENBERG, C.; XIANG, Q.; YANG, Y. R.; OHLMAN, B.; RANDRIAMASY, S.; BOTEN, F.; CONTRERAS, L. M. Supporting multi-domain use cases with alto. In: *Proceedings of the Applied Networking Research Workshop*. New York, NY, USA: Association for Computing Machinery, 2019. (ANRW '19), p. 59–61. ISBN 9781450368483. Available from Internet: <https://doi.org/10.1145/3340301.3341126>. Cited on page 23.

LACHOS, D.; XIANG, Q.; ROTHENBERG, C.; RANDRIAMASY, S.; CONTRERAS, L. M.; OHLMAN, B. Towards deep network amp; application integration: Possibilities, challenges, and research directions. In: *Proceedings of the Workshop on Network Application Integration/CoDesign*. New York, NY, USA: Association for Computing Machinery, 2020. (NAI '20), p. 1–7. ISBN 9781450380447. Available from Internet: <https://doi.org/10.1145/3405672.3405804>. Cited 2 times on pages 42 and 67.

LCLS. *The Linac Coherent Light Source*. 2020. <https://lcls.slac.stanford.edu/>. Cited on page 49.

LE, N. T.; HOSSAIN, M. A.; ISLAM, A.; KIM, D.-y.; CHOI, Y.-J.; JANG, Y. M. Survey of promising technologies for 5g networks. *Mobile information systems*, Hindawi, v. 2016, 2016. Cited on page 71.

LEE, J.; TURNER, Y.; LEE, M.; POPA, L.; BANERJEE, S.; KANG, J.-M.; SHARMA, P. Application-driven bandwidth guarantees in datacenters. In: *Proceedings of the 2014 ACM conference on SIGCOMM*. [S.l.: s.n.], 2014. p. 467–478. Cited 3 times on pages 19, 20, and 30.

LHC. *The Large Hadron Collider (LHC) Experiment*. 2020. <https://home.cern/topics/large-hadron-collider>. Cited on page 49.

LI, C.; HAVEL, O.; LIU, W. S.; OLARIU, A.; MARTINEZ-JULIA, P.; NOBRE, J. C.; LOPEZ, D. *Intent Classification*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-irtf-nmrg-ibn-intent-classification-00>. Cited on page 33.

LI, G.; LI, G.; LI, T.; XU, Q.; ZHOU, H. chun. *Hybrid Hierarchical Multi-Domain Service Function chaining*. [S.l.], 2018. <http://www.ietf.org/internet-drafts/draft-li-sfc-hhsfc-05.txt>. Available from Internet: <http://www.ietf.org/internet-drafts/draft-li-sfc-hhsfc-05.txt>. Cited on page 50.

LI, G.; LI, G.; XU, Q.; ZHOU, H. chun; FENG, B. *Hybrid Hierarchical Multi-Domain Service Function chaining*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-li-sfc-hhsfc-08>. Cited 2 times on pages 59 and 103.

LI, Z.; PENG, S.; LI, C.; XIE, C.; VOYER, D.; LI, X.; LIU, P.; LIU, C.; EBISAWA, K. *Application-aware IPv6 Networking (APN6) Encapsulation*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-li-6man-app-aware-ipv6-network-02>. Cited 2 times on pages 33 and 34.

LICCIARDELLO, M.; FIORANI, M.; FURDEK, M.; MONTI, P.; RAFFAELLI, C.; WOSINSKA, L. Performance evaluation of abstraction models for orchestration of distributed data center networks. In: IEEE. *2017 19th International Conference on Transparent Optical Networks (ICTON)*. [S.l.], 2017. p. 1–4. Cited 2 times on pages 73 and 74.

LU, Q.; ZHANG, L.; SASIDHARAN, S.; WU, W.; DEMAR, P.; GUOK, C.; MACAULEY, J.; MONGA, I.; YU, S.-y.; CHEN, J. H. *et al.* Bigdata express: Toward schedulable, predictable, and high-performance data transfer. In: IEEE. *2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS)*. [S.l.], 2018. p. 75–84. Cited on page 34.

MADHYASTHA, H. V.; ISDAL, T.; PIATEK, M.; DIXON, C.; ANDERSON, T.; KRISHNAMURTHY, A.; VENKATARAMANI, A. iplane: An information plane for distributed services. In: *Proceedings of the 7th symposium on Operating systems design and implementation*. [S.l.: s.n.], 2006. p. 367–380. Cited 2 times on pages 19 and 30.

MAO, H.; CHEN, S.; DIMMERY, D.; SINGH, S.; BLAISDELL, D.; TIAN, Y.; ALIZADEH, M.; BAKSHY, E. Real-world video adaptation with reinforcement learning. *arXiv preprint arXiv:2008.12858*, 2020. Cited on page 37.

MAO, H.; NETRAVALI, R.; ALIZADEH, M. Neural adaptive video streaming with pensieve. In: *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. [S.l.: s.n.], 2017. p. 197–210. Cited on page 37.

MARQUES, P. R.; MAUCH, J.; SHETH, N.; GREENE, B.; RASZUK, R.; MCPHERSON, D. R. *Dissemination of Flow Specification Rules*. RFC Editor, 2009. RFC 5575. (Request for Comments, 5575). Available from Internet: <https://rfc-editor.org/rfc/rfc5575.txt>. Cited 3 times on pages 19, 49, and 60.

MATILDA. *MATILDA H2020 Project*. 2017. Http://www.ict-vital.eu/. Available from Internet: <http://www.ict-vital.eu/>. Cited 3 times on pages 50, 51, and 52.

Metro Ethernet Forum. *Lifecycle Service Orchestration (LSO): Reference Architecture and Framework*. 2019. Available from Internet: <https://www.mef.net/Assets/Technical_Specifications/PDF/MEF_55.pdf>. Cited on page 51.

MIJUMBI, R.; SERRAT, J.; GORRICHO, J.; BOUTEN, N.; TURCK, F. D.; BOUTABA, R. Network function virtualization: State-of-the-art and research challenges. *Communications Surveys Tutorials, IEEE*, PP, n. 99, p. 1–1, 2015. ISSN 1553-877X. Cited on page 71.

MONGA, I.; GUOK, C.; MACAULEY, J.; SIM, A.; NEWMAN, H.; BALCAS, J.; DEMAR, P.; WINKLER, L.; LEHMAN, T.; YANG, X. Sdn for end-to-end networked science at the exascale (sense). In: IEEE. *2018 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS)*. [S.l.], 2018. p. 33–44. Cited on page 35.

NÉMETH, B.; SONKOLY, B.; ROST, M.; SCHMID, S. Efficient service graph embedding: A practical approach. In: IEEE. *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. [S.l.], 2016. p. 19–25. Cited on page 79.

NEO4J. *The Neo4j Manual v2.3.0-M03*. 2015. Http://neo4j.com/docs/milestone/. Available from Internet: <http://neo4j.com/docs/milestone/>. Cited on page 95.

NGMN Alliance. 5G Network and Service Management including Orchestration. 2017. Available from Internet: <https://www.ngmn.org/fileadmin/user_upload/170307_5G_Network_and_Service_Management___including_Orchestration_2.12.7.pdf>. Cited on page 51.

NGMN Alliance. *5G End-to-End Architecture Framework v2.0*. [S.l.], 2018. 1–37 p. Available from Internet: <https://www.ngmn.org/fileadmin/ngmn/content/downloads/Technical/2018/180226_NGMN_E2EArchFramework_v2.0.0.pdf>. Cited on page 51.

OASIS. *TOSCA specification*. 2013. Http://docs.oasis-open.org/tosca/TOSCA/v1.0/os/TOSCA-v1.0-os.pdf. Cited on page 92.

ONAP, O. N. A. P. *Active and Available Inventory Project*. 2018. Https://wiki.onap.org/display/DW/Active+and+Available+Inventory+Project. Available from Internet: <https://wiki.onap.org/display/DW/Active+and+Available+Inventory+Project>. Cited 2 times on pages 73 and 74.

OTT, M. *Intelligent network based application recognition*. [S.l.]: Google Patents, 2005. US Patent 6,961,770. Cited 2 times on pages 19 and 30.

Pacific Wave. *R&E Networks in CA, OR, and WA Announce Collaborative Support for COVID-19 Wester States Pact*. 2020. Accessed 2020-05-03. Available from Internet: <https://tinyurl.com/ydgfa7kf>. Cited on page 19.

Pacific Wave. *R&E Networks in CO AND NV Join Networks in CA, OR, and WA to Support COVID-19 Wester States Pact*. 2020. Accessed 2020-05-03. Available from Internet: <https://tinyurl.com/y6w3cz2n>. Cited on page 19.

PAULRAJ, S.; SUMATHI, P. *et al.* A comparative study of redundant constraints identification methods in linear programming problems. *Mathematical Problems in Engineering*, Hindawi, v. 2010, 2010. Cited on page 45.

PEREZ, D.; XIANG, Q.; ROTHENBERG, C.; YANG, Y. *Multi-domain Service Function Chaining with ALTO*. [S.l.], 2018. <http://www.ietf.org/internet-drafts/draft-lachos-multi-domain-sfc-alto-00.txt>. Available from Internet: <http://www.ietf.org/internet-drafts/draft-lachos-multi-domain-sfc-alto-00.txt>. Cited on page 59.

PEREZ, D. A. L.; BRITO, S. H. B.; FONTES, R. dos R.; ROTHENBERG, C. E. Delivering application-layer traffic optimization services based on public routing data at internet exchange points. In: *XXXXIV Brazilian Symposium on Computer Networks and Distributed Systems (SBRC)*. [S.l.: s.n.], 2016. Cited on page 53.

PEREZ, D. A. L.; ROTHENBERG, C. E. Multi-domain orchestration leveraging the application-layer traffic optimization protocol. *Proceedings of the V Workshop Pre-IETF (WPIETF-CSBC)*, v. 5, 2018. ISSN 2595-6388. Available from Internet: <http://ojs.sbc.org.br/index.php/wpietf/article/view/3214>. Cited on page 42.

PEREZ, D. A. L.; ROTHENBERG, C. E. *ALTO-based Broker-assisted Multi-domain Orchestration*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-lachosrothenberg-alto-brokermdo-04>. Cited 4 times on pages 42, 64, 65, and 93.

PEREZ, D. A. L.; ROTHENBERG, C. E.; SANTOS, M.; GOMES, P. H. Ani: Abstracted network inventory for streamlined service placement in distributed clouds. In: IEEE. *2020 6th IEEE Conference on Network Softwarization (NetSoft)*. [S.l.], 2020. p. 319–325. Cited on page 22.

PEREZ, D. A. L.; XIANG, Q.; ROTHENBERG, C. E.; YANG, Y. R. *Multi-domain Service Function Chaining with ALTO*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-lachos-sfc-multi-domain-alto-01>. Cited 2 times on pages 64 and 65.

PERRY, J.; OUSTERHOUT, A.; BALAKRISHNAN, H.; SHAH, D.; FUGAL, H. Fastpass: a centralized" zero-queue" datacenter network. In: *Proceedings of the 2014 ACM conference on SIGCOMM*. [S.l.: s.n.], 2014. Cited on page 32.

PROJECT, H. V. *D2.3 System Architecture: Final Report*. 2016. Accessed 2017-12-05. Available from Internet: <https://drive.google.com/file/d/0B5yhgJbT3R8kam5DbUM1ZHNSVGs/view>. Cited on page 52.

PUJOL, E.; POESE, I.; ZERWAS, J.; SMARAGDAKIS, G.; FELDMANN, A. Steering hyper-giants' traffic at scale. In: *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*. [S.l.: s.n.], 2019. p. 82–95. Cited 2 times on pages 21 and 39.

RAMAKRISHNAN, K.; FLOYD, S.; BLACK, D. *The Addition of Explicit Congestion Notification (ECN) to IP*. [S.l.], 2001. <http://www.rfc-editor.org/rfc/rfc3168.txt>. Available from Internet: <http://www.rfc-editor.org/rfc/rfc3168.txt>. Cited 2 times on pages 36 and 37.

RANDRIAMASY, S.; ROOME, W.; SCHWAN, N. *Multi-Cost Application-Layer Traffic Optimization (ALTO)*. [S.l.], 2017. Cited 4 times on pages 40, 54, 93, and 94.

RANDRIAMASY, S.; YANG, Y. R.; WU, Q.; LINGLI, D.; SCHWAN, N. *Application-Layer Traffic Optimization (ALTO) Cost Calendar*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-ietf-alto-cost-calendar-21>. Cited on page 55.

RAYKOVA, M. P. *Secure computation in heterogeneous environments: how to bring multiparty computation closer to practice?* Tese (Doutorado) — Columbia University, 2012. Cited on page 69.

REKHTER, Y.; HARES, S.; LI, T. *A Border Gateway Protocol 4 (BGP-4)*. RFC Editor, 2006. RFC 4271. (Request for Comments, 4271). Available from Internet: <https://rfc-editor.org/rfc/rfc4271.txt>. Cited 3 times on pages 64, 65, and 66.

RIERA, J. F.; BATALLé, J.; BONNET, J.; DíAS, M.; MCGRATH, M.; PETRALIA, G.; LIBERATI, F.; GIUSEPPI, A.; PIETRABISSA, A.; CESELLI, A.; PETRINI, A.; TRUBIAN, M.; PAPADIMITROU, P.; DIETRICH, D.; RAMOS, A.; MELIáN, J.; XILOURIS, G.; KOURTIS, A.; KOURTIS, T.; MARKAKIS, E. K. Tenor: Steps towards an orchestration platform for multi-pop nfv deployment. In: *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. [S.l.: s.n.], 2016. p. 243–250. Cited on page 52.

ROOME, W.; RANDRIAMASY, S.; YANG, Y. R.; ZHANG, J. J.; GAO, K. *Unified properties for the ALTO protocol*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-ietf-alto-unified-props-new-12>. Cited 4 times on pages 40, 55, 59, and 93.

ROOME, W.; YANG, Y. R. *ALTO Incremental Updates Using Server-Sent Events (SSE)*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-ietf-alto-incr-update-sse-22>. Cited on page 55.

ROUX, J.-L. L. *Requirements for Path Computation Element (PCE) Discovery*. RFC Editor, 2006. RFC 4674. (Request for Comments, 4674). Available from Internet: <https://rfc-editor.org/rfc/rfc4674.txt>. Cited 2 times on pages 64 and 66.

SCHMIDT, P.; ENGHARDT, T.; KHALILI, R.; FELDMANN, A. Socket intents: Leveraging application awareness for multi-access connectivity. In: *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*. NY, USA: [s.n.], 2013. (ACM CoNEXT '13). ISBN 9781450321013. Cited 4 times on pages 19, 33, 34, and 64.

SEEDORF, J.; BURGER, E. *Application-Layer Traffic Optimization (ALTO) Problem Statement*. [S.l.], 2009. <http://www.rfc-editor.org/rfc/rfc5693.txt>. Available from Internet: <http://www.rfc-editor.org/rfc/rfc5693.txt>. Cited 2 times on pages 53 and 54.

SEEDORF, J.; YANG, Y. R.; MA, K. J.; PETERSON, J.; ZHANG, J. J. *Content Delivery Network Interconnection (CDNI) Request Routing: CDNI Footprint and Capabilities Advertisement using ALTO*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-ietf-alto-cdni-request-routing-alto-12>. Cited on page 55.

SGAMBELLURI, A.; TUSA, F.; GHARBAOUI, M.; MAINI, E.; TOKA, L.; PEREZ, J.; PAOLUCCI, F.; MARTINI, B.; POE, W. Y.; HERNANDES, J. M. *et al.* Orchestration of network services across multiple operators: The 5g exchange prototype. In: IEEE. *2017 European Conference on Networks and Communications (EuCNC)*. [S.l.], 2017. p. 1–5. Cited on page 51.

SKA. *The Square Kilometre Array*. 2020. <https://www.skatelescope.org/>. Cited on page 49.

SLIM, F.; GUILLEMIN, F.; GRAVEY, A.; HADJADJ-AOUL, Y. Towards a dynamic adaptive placement of virtual network functions under onap. In: IEEE. *Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017 IEEE Conference on*. [S.l.], 2017. p. 210–215. Cited on page 71.

SOENEN, T.; SAHHAF, S.; TAVERNIER, W.; SKÖLDSTRÖM, P.; COLLE, D.; PICKAVET, M. A model to select the right infrastructure abstraction for service function chaining. In: IEEE. *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. [S.l.], 2016. p. 233–239. Cited 3 times on pages 21, 73, and 74.

SONG, H.; KIESEL, S.; STIEMERLING, M.; SCHWAN, N.; SCHARF, M. *Application-Layer Traffic Optimization (ALTO) Server Discovery*. RFC Editor, 2014. RFC 7286. (Request for Comments, 7286). Available from Internet: <https://rfc-editor.org/rfc/rfc7286.txt>. Cited 2 times on pages 54 and 66.

SONKOLY, B.; CZENTYE, J.; SZABO, R.; JOCHA, D.; ELEK, J.; SAHHAF, S.; TAVERNIER, W.; RISSO, F. Multi-domain service orchestration over networks and clouds: a unified approach. In: ACM. *ACM SIGCOMM Computer Communication Review*. [S.l.], 2015. v. 45, n. 4, p. 377–378. Cited 4 times on pages 21, 72, 73, and 74.

SOULÉ, R.; BASU, S.; MARANDI, P. J.; PEDONE, F.; KLEINBERG, R.; SIRER, E. G.; FOSTER, N. Merlin: A language for provisioning network resources. In: *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. [S.l.: s.n.], 2014. p. 213–226. Cited 3 times on pages 19, 20, and 30.

SOUSA, N. F. S. de; PEREZ, D. A. L.; ROSA, R. V.; SANTOS, M. A.; ROTHENBERG, C. E. Network service orchestration: A survey. *Computer Communications*, Elsevier, 2019. Cited on page 71.

STIEMERLING, M.; KIESEL, S.; SCHARF, M.; SEIDEL, H.; PREVIDI, S. *Application-Layer Traffic Optimization (ALTO) Deployment Considerations*. RFC Editor, 2016. RFC 7971. (Request for Comments, 7971). Available from Internet: <https://rfc-editor.org/rfc/rfc7971.txt>. Cited 3 times on pages 53, 54, and 61.

SUN, G.; LI, Y.; LIAO, D.; CHANG, V. Service function chain orchestration across multiple domains: A full mesh aggregation approach. *IEEE Transactions on Network and Service Management*, IEEE, 2018. Cited 2 times on pages 20 and 58.

T-NOVA. *T-NOVA Project, Network Functions as a Service over Virtualised Infrastructures*. 2014. Http://www.t-nova.eu/. Available from Internet: <http://www.t-nova.eu/>. Cited 3 times on pages 50, 51, and 52.

TELGEN, J. Identifying redundant constraints and implicit equalities in systems of linear constraints. *Management Science*, INFORMS, v. 29, n. 10, p. 1209–1222, 1983. Cited on page 46.

THOMSON, M.; BELLIS, R. *Location Information Server (LIS) Discovery Using IP Addresses and Reverse DNS*. RFC Editor, 2014. RFC 7216. (Request for Comments, 7216). Available from Internet: <https://rfc-editor.org/rfc/rfc7216.txt>. Cited on page 66.

TM-FORUM. *TM Forum Open APIs*. 2020. Https://projects.tmforum.org/wiki/display/API/Open+API+Table. Available from Internet: <https://projects.tmforum.org/wiki/display/API/Open+API+Table>. Cited on page 33.

UNIFY D3.2a. *Network Function Forwarding Graph (NFFG) specification*. 2015. Available from Internet: <https://www.eict.de/fileadmin/redakteure/Projekte/Unify/Deliverables/UNIFY-D3.2a-Network_Function_Forwarding_Graph_specification.pdf>. Cited 4 times on pages 42, 92, 94, and 122.

VASSEUR, J.; FARREL, A.; ASH, G. *A Path Computation Element (PCE)-Based Architecture*. RFC Editor, 2006. RFC 4655. (Request for Comments, 4655). Available from Internet: <https://rfc-editor.org/rfc/rfc4655.txt>. Cited on page 66.

VASSEUR, J.; ROUX, J.-L. L.; ZHANG, R.; BITAR, D. N. N. *A Backward-Recursive PCE-Based Computation (BRPC) Procedure to Compute Shortest Constrained Inter-Domain Traffic Engineering Label Switched Paths*. RFC Editor, 2009. RFC 5441. (Request for Comments, 5441). Available from Internet: <https://rfc-editor.org/rfc/rfc5441.txt>. Cited 3 times on pages 64, 65, and 66.

VERMA, A.; PEDROSA, L.; KORUPOLU, M.; OPPENHEIMER, D.; TUNE, E.; WILKES, J. Large-scale cluster management at google with borg. In: *Proceedings of the Tenth European Conference on Computer Systems*. [S.l.: s.n.], 2015. p. 1–17. Cited on page 21.

VITAL. *VITAL – VIrtualized hybrid satellite-TerrestriAl systems for resilient and fLexible future networks*. 2015. Http://www.ict-vital.eu/. Available from Internet: <http://www.ict-vital.eu/>. Cited 3 times on pages 50, 51, and 52.

WEERASIRI, D.; BARUKH, M. C.; BENATALLAH, B.; SHENG, Q. Z.; RANJAN, R. A Taxonomy and Survey of Cloud Resource Orchestration Techniques. *ACM Computing Surveys*, v. 50, n. 2, p. 1–41, may 2017. ISSN 03600300. Available from Internet: <http://dl.acm.org/citation.cfm?doid=3071073.3054177>. Cited on page 71.

WU, Q.; YANG, Y. R.; LEE, Y.; DHODY, D.; RANDRIAMASY, S.; CONTRERAS, L. M. *ALTO Performance Cost Metrics*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-ietf-alto-performance-metrics-12>. Cited on page 55.

XIANG, Q.; GUOK, C.; LE, F.; MACAULEY, J.; NEWMAN, H.; YANG, Y. R. Sfp: Toward interdomain routing for sdn networks. In: ACM. *Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*. [S.l.], 2018. p. 87–89. Cited 3 times on pages 19, 49, and 60.

XIANG, Q.; WANG, X. T.; ZHANG, J. J.; NEWMAN, H.; YANG, Y. R.; LIU, Y. J. Unicorn: Unified resource orchestration for multi-domain, geo-distributed data analytics. *Future Generation Computer Systems*, Elsevier, v. 93, p. 188–197, 2019. Cited 2 times on pages 64 and 67.

XIANG, Q.; ZHANG, J.; GAO, K.; LIM, Y.-s.; LE, F.; LI, G.; YANG, Y. R. Toward optimal software-defined interdomain routing. In: IEEE. *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. [S.l.], 2020. p. 1529–1538. Cited 2 times on pages 37 and 38.

XIANG, Q.; ZHANG, J.; LE, F.; YANG, Y. R. *ALTO Extension: Unified Resource Representation*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-xiang-alto-unified-representation-03>. Cited 3 times on pages 42, 64, and 67.

XIANG, Q.; ZHANG, J.; LE, F.; YANG, Y. R.; NEWMAN, H. *Resource Orchestration for Multi-Domain, Exascale, Geo-Distributed Data Analytics*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-xiang-alto-multidomain-analytics-05>. Cited 3 times on pages 64, 65, and 69.

XIANG, Q.; ZHANG, J. J.; WANG, X. T.; LIU, Y. J.; GUOK, C.; LE, F.; MACAULEY, J.; NEWMAN, H.; YANG, Y. R. Fine-grained, multi-domain network resource abstraction as a fundamental primitive to enable high-performance, collaborative data sciences. In: ACM. *Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*. [S.l.], 2018. p. 27–29. Cited 8 times on pages 21, 64, 67, 68, 69, 72, 73, and 74.

XIE, H.; YANG, Y. R.; KRISHNAMURTHY, A.; LIU, Y. G.; SILBERSCHATZ, A. P4p: Provider portal for applications. *ACM SIGCOMM Computer Communication Review*, ACM New York, NY, USA, v. 38, n. 4, 2008. Cited 4 times on pages 19, 21, 30, and 37.

YANG, S.; CUI, L.; XU, M.; YANG, Y. R.; HUANG, R. *Delivering Functions over Networks: Traffic and Performance Optimization for Edge Computing using ALTO*. [S.l.], 2020. Work in Progress. Available from Internet: <https://datatracker.ietf.org/doc/html/draft-yang-alto-deliver-functions-over-networks-01>. Cited 2 times on pages 37 and 38.

YANG, Y. R.; GAO, K.; GOKARSLAN, K.; GUO, D.; LEET, C. Magellan: Toward automatic mapping from high-level sdn programs to low-level , optimized datapath pipelines. In: . [S.l.: s.n.], 2017. Cited 2 times on pages 33 and 34.

YANG, Y. R.; POPKIN, L.; WOUNDY, R.; GRIFFITHS, C.; LIVINGOOD, J. *Comcast's ISP Experiences in a Proactive Network Provider Participation for P2P (P4P) Technical Trial*. RFC Editor, 2009. RFC 5632. (Request for Comments, 5632). Available from Internet: <https://rfc-editor.org/rfc/rfc5632.txt>. Cited on page 38.

Yong Li; Min Chen. Software-Defined Network Function Virtualization: A Survey. *IEEE Access*, v. 3, p. 2542–2553, 2015. ISSN 2169-3536. Available from Internet: <http://ieeexplore.ieee.org/document/7350211/>. Cited on page 71.

ZHANG, Y.; LI, G.; XIONG, C.; LEI, Y.; HUANG, W.; HAN, Y.; WALID, A.; YANG, Y. R.; ZHANG, Z.-L. Mowie: Toward systematic, adaptive network information exposure as an enabling technique for cloud-based applications over 5g and beyond (invited paper).

In: *Proceedings of the Workshop on Network Application Integration/CoDesign*. New York, NY, USA: Association for Computing Machinery, 2020. (NAI '20), p. 20–27. ISBN 9781450380447. Available from Internet: <https://doi.org/10.1145/3405672.3409489>. Cited 2 times on pages 37 and 38.

ZHAO, S.; MUPPALA, G.; LI, Z.; MEDHI, D. Smooth streaming with mpeg-dash using sdn-based application-aware networking. In: IEEE. *2018 International Conference on Computing, Networking and Communications (ICNC)*. [S.l.], 2018. p. 77–81. Cited 2 times on pages 33 and 34.

ZHENG, X.; VEERARAGHAVAN, M.; RAO, N. S.; WU, Q.; ZHU, M. Cheetah: Circuit-switched high-speed end-to-end transport architecture testbed. *IEEE Communications Magazine*, IEEE, 2005. Cited 2 times on pages 20 and 30.

ANNEX A – YANG data model of Network Function Forwarding Graph (NFFG)

The following YANG data model of the NFFG and its tree representation was extracted from (UNIFY D3.2a, 2015):

```
+--rw nffg
   +--rw parameters
   |  +--rw id         string
   |  +--rw name?      string
   |  +--rw version    string
   +--rw node_nfs* [id]
   |  +--rw id                 string
   |  +--rw name?              string
   |  +--rw functional_type    string
   |  +--rw specification
   |  |  +--rw deployment_type?   string
   |  |  +--rw resources
   |  |     +--rw cpu          string
   |  |     +--rw mem          string
   |  |     +--rw storage      string
   |  |     +--rw delay        string
   |  |     +--rw bandwidth    string
   |  +--rw ports* [id]
   |     +--rw id          string
   |     +--rw property*   string
   +--rw node_saps* [id]
   |  +--rw id        string
   |  +--rw name?     string
   |  +--rw domain?   string
   |  +--rw ports* [id]
   |     +--rw id          string
   |     +--rw property*   string
   +--rw node_infras* [id]
   |  +--rw id           string
   |  +--rw name?        string
   |  +--rw domain?      string
   |  +--rw type         string
   |  +--rw supported* [functional_type]
   |  |  +--rw functional_type    string
   |  +--rw resources
   |  |  +--rw cpu          string
   |  |  +--rw mem          string
   |  |  +--rw storage      string
   |  |  +--rw delay        string
   |  |  +--rw bandwidth    string
   |  +--rw ports* [id]
   |     +--rw id           string
   |     +--rw property*    string
   |     +--rw flowrules* [id]
```

```
45      |        +--rw id           string
46      |        +--rw match        string
47      |        +--rw action       string
48      |        +--rw bandwidth?   string
49      +--rw edge_links* [id]
50      |  +--rw id          string
51      |  +--rw src_node    string
52      |  +--rw src_port    string
53      |  +--rw dst_node    string
54      |  +--rw dst_port    string
55      |  +--rw backward?   string
56      |  +--rw reqs
57      |     +--rw delay?       string
58      |     +--rw bandwidth?   string
59      +--rw edge_sg_nexthops* [id]
60      |  +--rw id           string
61      |  +--rw src_node     string
62      |  +--rw src_port     string
63      |  +--rw dst_node     string
64      |  +--rw dst_port     string
65      |  +--rw flowclass?   string
66      +--rw edge_reqs* [id]
67         +--rw id          string
68         +--rw src_node    string
69         +--rw src_port    string
70         +--rw dst_node    string
71         +--rw dst_port    string
72         +--rw reqs
73         |  +--rw delay?       string
74         |  +--rw bandwidth?   string
75         +--rw sg_path* [edge_sg_nexthop_id]
76            +--rw edge_sg_nexthop_id    string
```

ANNEX B – ALTO Map Services examples

B.1 Property Map Service

This HTTP request example corresponds to the full (unfiltered) Property Map. The ALTO server defines a GET-mode resource which returns the entire Property Map for all PID entities with the *"unifyslor"*, *"cpu"*, *"mem"*, *"storage"*, *"port"* and *"nf"* properties.

- HTTP Request

```
GET /controller/nb/v2/alto/propertymap/my-default-property-map HTTP/1.1
Host: 172.28.0.10:8181
Accept: application/alto-propmap+json,application/alto-error+json
```

- HTTP Response

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"
        }
    },
    "property-map": {
        "0.0.0.1": {
            "unifyslor": "https://172.25.0.10:8888/escape",
            "cpu": "50.0", "mem": "60.0", "storage": "70.0",
            "port": [ "SAP1", "SAP1211", "SAP1231", "SAP2" ],
            "nf": [ "COMPRESSOR", "DECOMPRESSOR" ]
        },
        "0.0.0.121": {
            "unifyslor": "https://172.50.0.10:8888/escape",
            "cpu": "0.0", "mem": "0.0", "storage": "0.0",
            "port": [ "SAP1211", "SAP1212" ],
            "nf": []
        },
        "0.0.0.122": {
            "unifyslor": "https://172.51.0.10:8888/escape",
            "cpu": "0.0", "mem": "0.0", "storage": "0.0",
            "port": [ "SAP1212", "SAP1221" ],
            "nf": []
        },
        "0.0.0.123": {
            "unifyslor": "https://172.52.0.10:8888/escape",
            "cpu": "0.0", "mem": "0.0", "storage": "0.0",
            "port": [ "SAP1231", "SAP1232", "SAP1233" ],
            "nf": []
        },
```

```
        "0.0.0.2": {
            "unifyslor": "https://172.26.0.10:8888/escape",
            "cpu": "10.0", "mem": "20.0", "storage": "30.0",
            "port": [ "SAP1221", "SAP1232", "SAP2311", "port-SAP4" ],
            "nf": [ "FORWARDER" ]
        },
        "0.0.0.231": {
            "unifyslor": "https://172.53.0.10:8888/escape",
            "cpu": "0.0", "mem": "0.0", "storage": "0.0",
            "port": [ "SAP2311", "SAP2312" ],
            "nf": []
        },
        "0.0.0.3": {
            "unifyslor": "https://172.27.0.10:8888/escape",
            "cpu": "80.0", "mem": "90.0", "storage": "100.0",
            "port": [ "SAP1233", "SAP2312", "SAP3" ],
            "nf": [ "COMPRESSOR", "DECOMPRESSOR" ]
        }
    }
}
```

B.2 Cost Map Services: ASes

B.2.1 Full Cost Map

- HTTP Request

```
POST /controller/nb/v2/alto/costmap/pv
Host: 172.28.0.10:8181
Accept: multipart/related, application/alto-costmap+json,
application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json

{
    "cost-type" :{
            "cost-mode": "array",
            "cost-metric": "ane-path"
    },
    "pids" : {
        "srcs" : [ "0.0.0.1" ],
        "dsts" : [ "0.0.0.2","0.0.0.3" ]
    }
}
```

- HTTP Response

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
```

```
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"
        }
    },
    "cost-map": {
       "0.0.0.1": {
            "0.0.0.2": [
                ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2"],
                ["0.0.0.1","0.0.0.123","0.0.0.3","0.0.0.231","0.0.0.2"],
                ["0.0.0.1","0.0.0.123","0.0.0.2"]
            ],
            "0.0.0.3": [
                ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.231","0.0.0.3"],
                ["0.0.0.1","0.0.0.123","0.0.0.2","0.0.0.231","0.0.0.3"],
                ["0.0.0.1","0.0.0.123","0.0.0.3"],
                ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.123","0.0.0.3"]
            ]
        }
    }
}
```

B.2.2 Filtered Cost Map

- HTTP Request

```
POST /controller/nb/v2/alto/costmap/pv
Host: 172.28.0.10:8181
Accept: multipart/related, application/alto-costmap+json,
application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json

{
    "cost-type" :{
            "cost-mode": "array",
            "cost-metric": "ane-path"
    },
    "pids" : {
        "srcs" : [ "0.0.0.1" ],
        "dsts" : [ "0.0.0.2","0.0.0.3" ]
    },
    "constraints" : [ "shortest"]
}
```

- HTTP Response

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"
        }
    },
```

```
    "cost-map": {
       "0.0.0.1": {
            "0.0.0.2": [
                ["0.0.0.1","0.0.0.123","0.0.0.2"]
            ],
            "0.0.0.3": [
                ["0.0.0.1","0.0.0.123","0.0.0.3"]
            ]
        }
    }
}
```

B.3 Cost Map Service: E2E Service Requirements

B.3.1 Full Cost Map

- HTTP Request

```
POST /controller/nb/v2/alto/costmap/pv
Host: 172.28.0.10:8181
Accept: multipart/related, application/alto-costmap+json,
application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json
{
    "cost-type" :{
            "cost-mode": "array", "cost-metric": "ane-path"
    },
    "constraints" : [ "shortest"],
    "sg" :{
          "nfs": [ "COMPRESSOR", "FORWARDER", "DECOMPRESSOR"],
          "saps": [ "SAP1", "SAP3" ],
          "sg_links": [ {"id": 1,
                  "src_node": "SAP1",
                  "dst_node": "COMPRESSOR"
                },{"id": 2,
                  "src_node": "COMPRESSOR",
                  "dst_node": "FORWARDER"
                },{"id": 3,
                  "src_node": "FORWARDER",
                  "dst_node": "DECOMPRESSOR",
                },{"id": 4,
                  "src_node": "DECOMPRESSOR",
                  "dst_node": "SAP3"
                }],
          "reqs": [ {
                "src_node": "SAP1",
                "dst_node": "SAP3",
                "sg_path": [ 1, 2, 3, 4 ]
            }
          ]
```

```
    }
}
```

- HTTP Response


```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"}
    },
    "cost-map": {
        "SAP1": {
            "SAP3": {
                "SAP1": {"COMPRESSOR": [
                        ["0.0.0.1","0.0.0.123","0.0.0.2","0.0.0.231","0.0.0.3"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.231",
                        "0.0.0.3"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.123",
                        "0.0.0.3"],
                        ["0.0.0.1","0.0.0.123","0.0.0.3"],
                        ["0.0.0.1"]
                    ]
                },
                "COMPRESSOR": {"FORWARDER": [
                        ["0.0.0.3","0.0.0.123","0.0.0.1","0.0.0.121","0.0.0.122",
                        "0.0.0.2"],
                        ["0.0.0.3","0.0.0.231","0.0.0.2","0.0.0.122","0.0.0.121",
                        "0.0.0.1","0.0.0.123","0.0.0.2"
                        ],
                        ["0.0.0.3","0.0.0.231","0.0.0.2","0.0.0.123","0.0.0.1",
                        "0.0.0.121","0.0.0.122","0.0.0.2"],
                        ["0.0.0.3","0.0.0.231","0.0.0.2"],
                        ["0.0.0.1","0.0.0.123","0.0.0.3","0.0.0.231","0.0.0.2"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.123",
                        "0.0.0.3","0.0.0.231","0.0.0.2"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2"],
                        ["0.0.0.3","0.0.0.123","0.0.0.2"],
                        ["0.0.0.1","0.0.0.123","0.0.0.2"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.231",
                        "0.0.0.3","0.0.0.123","0.0.0.2"]
                    ]
                },
                "FORWARDER": {"DECOMPRESSOR": [
                        ["0.0.0.2","0.0.0.231","0.0.0.3","0.0.0.123","0.0.0.1"],
                        ["0.0.0.2","0.0.0.123","0.0.0.3"],
                        ["0.0.0.2","0.0.0.122","0.0.0.121","0.0.0.1"],
                        ["0.0.0.2","0.0.0.122","0.0.0.121","0.0.0.1","0.0.0.123",
                        "0.0.0.2","0.0.0.231","0.0.0.3"],
                        ["0.0.0.2","0.0.0.231","0.0.0.3","0.0.0.123","0.0.0.2",
                        "0.0.0.122","0.0.0.121","0.0.0.1"],
                        ["0.0.0.2","0.0.0.123","0.0.0.3","0.0.0.231","0.0.0.2",
                        "0.0.0.122","0.0.0.121","0.0.0.1"],
                        ["0.0.0.2","0.0.0.123","0.0.0.1","0.0.0.121","0.0.0.122",
                        "0.0.0.2","0.0.0.231","0.0.0.3"],
```

```
                        ["0.0.0.2","0.0.0.123","0.0.0.1"],
                        ["0.0.0.2","0.0.0.122","0.0.0.121","0.0.0.1","0.0.0.123",
                        "0.0.0.3"],
                        ["0.0.0.2","0.0.0.231","0.0.0.3"]
                    ]
                },
                "DECOMPRESSOR": {
                    "SAP3": [
                        ["0.0.0.1","0.0.0.123","0.0.0.2","0.0.0.231","0.0.0.3"],
                        ["0.0.0.3"],
                        ["0.0.0.1","0.0.0.123","0.0.0.3"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.123",
                        "0.0.0.3"],
                        ["0.0.0.1","0.0.0.121","0.0.0.122","0.0.0.2","0.0.0.231",
                        "0.0.0.3"]
                    ]
                }
            }
        }
    }
}
```

B.3.2 Filtered Cost Map

In this Filtered Cost Map service, the ALTO server returns connectivity information for an SG request provided by the HTTP request example. This request includes a constraint predicate (*"constraints" : ["shortest"]*) so that, the ALTO server returns the shortest AS-level topological distance which meets the E2ENS requirement.

- HTTP Request

```
POST /controller/nb/v2/alto/costmap/pv
Host: 172.28.0.10:8181
Accept: multipart/related, application/alto-costmap+json,
application/alto-propmap+json, application/alto-error+json
Content-Length: [TBD]
Content-Type: application/alto-costmapfilter+json
{
    "cost-type" :{
            "cost-mode": "array", "cost-metric": "ane-path"
    },
    "constraints" : [ "shortest"],
    "sg" :{
          "nfs": [ "COMPRESSOR", "FORWARDER", "DECOMPRESSOR"],
          "saps": [ "SAP1", "SAP3" ],
          "sg_links": [ {"id": 1,
                  "src_node": "SAP1",
                  "dst_node": "COMPRESSOR"
                },{"id": 2,
                  "src_node": "COMPRESSOR",
                  "dst_node": "FORWARDER"
                },{"id": 3,
```

```
                  "src_node": "FORWARDER",
                  "dst_node": "DECOMPRESSOR",
                },{"id": 4,
                  "src_node": "DECOMPRESSOR",
                  "dst_node": "SAP3"
                }],
          "reqs": [ {
                "src_node": "SAP1",
                "dst_node": "SAP3",
                "sg_path": [ 1, 2, 3, 4 ]
            }
          ]
    }
}
```

- HTTP Response

```
{
    "meta": {
        "vtag": {
            "resource-id": "my-default-property-map",
            "tag": "4VSt4OFTRMBdc5gHIuLGhKUBL4xMXsP8"}
    },
    "cost-map": {
        "SAP1": {
            "SAP3": {
                "SAP1": {"COMPRESSOR": [
                        [ "0.0.0.1" ]]
                },
                "COMPRESSOR": {"FORWARDER": [
                        [ "0.0.0.1","0.0.0.123","0.0.0.2" ]]
                },
                "FORWARDER": {"DECOMPRESSOR": [
                        [ "0.0.0.2","0.0.0.123","0.0.0.3" ]]
                },
                "DECOMPRESSOR": {
                    "SAP3": [
                        [ "0.0.0.3" ]]
                }
            }
        }
    }
}
```