IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 11, 2022, accepted April 20, 2022, date of publication May 2, 2022, date of current version May 12, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3171573

MESON: A Platform for Optimized Cross-Slice Communication on Edge Computing Infrastructures

DIMITRIOS LASKARATOS[1], IOANNIS DIMOLITSAS[2], GEORGE PAPATHANAIL[3], (Student Member, IEEE), MARIA-EVGENIA XEZONAKI[1], ANGELOS PENTELAS[3], VASILEIOS THEODOROU[1], DIMITRIOS DECHOUNIOTIS[2], THEODOROS BOZIOS[1], PANAGIOTIS PAPADIMITRIOU[3], (Senior Member, IEEE), AND SYMEON PAPAVASSILIOU[2], (Senior Member, IEEE)

[1]Intracom Telecom, 19002 Paiania, Greece
[2]School of Electrical and Computer Engineering, National Technical University of Athens, 157 80 Athens, Greece
[3]Department of Applied Informatics, The University of Macedonia, 546 36 Thessaloniki, Greece

Corresponding author: Vasileios Theodorou (theovas@intracom-telecom.com)

This work was supported in part by the MESON Project Co-Financed by the European Union and Greek National Funds through the Operational Program Competitiveness, Entrepreneurship and Innovation through the Call RESEARCH CREATE—INNOVATE under Project T1EDK-02947; and in part by the TANDEM (Function and Device as a Service Support for Edge Computing) Project Co-Financed by the European Union and Greek National Funds through the Operational Program Competitiveness, Entrepreneurship and Innovation by the Call RESEARCH CREATE—INNOVATE under Project T2EDK-02825.

ABSTRACT In the dawn of the 6G era, evolving service ecosystems raise the need for cross-service interactions. Existing resource/service orchestration frameworks are oblivious to such cross-service interactions, since they tend to orchestrate services independent to each other. In this respect, we stress on the need for optimized cross-service communication, and more specifically, for optimized cross-slice communication (CSC). To this end, we present a CSC orchestration platform, namely MESON, which fulfills all main CSC requirements, such as the discovery of CSC-enabled services, the selection of the most suitable (edge) computing infrastructure, and also the establishment and configuration of CSC in a secure and policy-compliant manner. MESON has been designed and built as an extension of the ETSI NFV MANO framework. Relying on MESON as a proof-of-concept for CSC, we present evaluation results from various performance and scalability tests, and further uncover significant gains from optimized CSC in relation to video streaming and Industry 4.0 use cases.

INDEX TERMS Network slicing, edge computing, service orchestration, NFV, 5G.

I. INTRODUCTION

Network slicing comprises a key enabler for the deployment of various 5G (and beyond) network services with stringent and potentially diverse network requirements [1]–[4]. In essence, a network slice is considered as a logical partition of the physical infrastructure, encompassing computing, storage, and network resources, all managed in a unified manner by the slice tenant. Network slicing greatly benefits from the evolution of underlying technologies, such as Network Function Virtualization (NFV) [5]–[7], and Software-Defined Networks [1], exploiting the flexibility of general-purpose processing and switching hardware. In this respect, network slicing reflects the current trend for convergence between computing and networking ecosystems, putting software into an unprecedented and dominant role in operational communication environments.

While slicing emerged from telecom networks, gradually evolving as a key ingredient in the realization of the "as a Service" paradigm, it has further found traction across edge computing infrastructures in order to satisfy the pressing demands of latency-sensitive applications, such as location-based services [8]. The concurrent deployment of a wide range of services within edge computing facilities opens up opportunities for cross-service interactions, *i.e.,* a network service consuming (part of) another service. For

The associate editor coordinating the review of this manuscript and approving it for publication was Rodrigo S. Couto.

instance, a touristic guide application can retrieve information from a social network in order to provide personalized recommendations and eventually enhance the experience offered to the end-users. Such services could be deployed on separate slices, which may be co-located in the same (edge) cloud infrastructure.

Such interactions among co-located services are hindered by the prevailing way of network slice instantiation. More specifically, the strong isolation between slices within the same datacenter (DC) inevitably stretches the communication path between two co-located slices, enforcing the traffic to exit the boundaries of the DC. This implication is more severe in edge computing infrastructures, due to the distance between the edge cloud and the (mobile) network core. Such unoptimized cross-slice communication (CSC) will potentially lead to latency inflation, with an adverse impact on the applications deployed on top of edge computing facilities, commonly being latency-sensitive. Furthermore, more bandwidth will be consumed from the backhaul and transport networks, which, besides the undesired implications for network operators, may also increase the expenditure of slice tenants.

Along these lines, we stress on the need for optimized CSC, such that will not introduce penalties in terms of latency or bandwidth consumption, incentivizing slice tenants to consume other services present in the same physical infrastructure [9]. In particular, we perceive optimized CSC as a form of *peering* between co-located slices, ensuring that CSC traffic does not exit the boundaries of the DC and also that any latency inflation is minimal. Note that by no means does optimized CSC impair resource and traffic isolation. On the contrary, CSC should be attained in a secure and controlled manner, *i.e.,* providing access to co-located services based on established agreements between the slice tenants.

Optimized CSC raises various requirements in terms of service and resource orchestration. So far, slice resource orchestration operations, such as the instantiation and scaling, solely pertain to resource and communication demands based on the service specification. Now, optimized CSC introduces inter-slice dependencies that should be carefully taken into account before the placement or the scaling of a slice that has requested or established CSC. A first step towards this direction has been taken in [10], where slice placement is being optimized based on both resource and CSC requirements. Furthermore, CSC establishment requires functionality not available at existing slicing orchestrators (*e.g.,* [3]), such as CSC service advertisement and discovery, as well as Point-of-Presence (PoP) selection. Although PoP selection is supported by certain NFV and slice orchestration frameworks [3], [7], [11], [12], it is oblivious to CSC.

To fulfill these requirements for CSC establishment, we present a platform for optimized CSC orchestration, namely MESON. Our main goal is to foster cross-service interactions and business-to-business (B2B) synergies, exploiting any CSC opportunities stemming from multi-tenancy and service co-location. To this end, we refrain from introducing CSC functionalities in a disruptive manner; instead, we introduce a CSC orchestration layer on top of the NFV MANO reference architecture, maintaining compatibility with state-of-the-art NFV orchestrators, such as OpenSourceMANO (OSM) [5]. The MESON platform comprises a fully-fledged orchestration system that has evolved from the CSC orchestration architecture in [9].

In particular, the main contributions of this paper are the following:

- We articulate the notion of optimized CSC in the context of edge computing and elaborate on two use cases that highlight the potential benefits from cross-service interactions.
- We present the design and implementation of a CSC orchestration platform that encompasses all required features for CSC-enabled operations, such as CSC service advertisement and discovery, CSC-optimized PoP selection, as well as CSC establishment. In this respect, we discuss in detail the interactions among the MESON components for the aforementioned operations, as well as the cross-layer interactions for CSC establishment coordinated by the CSC Optimizer MESON component.
- Coupling the MESON platform with OSM and OpenStack as a proof-of-concept, we conduct a feasibility study on CSC instantiation, assessing the timescales for CSC-enabled service discovery, PoP selection and CSC establishment. We further perform a range of micro-benchmarks on MESON-layer interactions in order to gain insights on the various CSC operations and identify potential performance overheads.
- Based on two use cases, we quantify the gains of optimized CSC in terms of service-level performance indicators based on two use cases related to video streaming and Industry 4.0.

The remainder of the paper is organized as follows. In Section II, we elaborate on the notion of optimized CSC and introduce the main primitives related to CSC. In Section III, we provide an overview of the MESON orchestration architecture and the envisaged roles within an evolved service ecosystem that fosters CSC. Section IV provides a detailed description of the MESON CSC orchestration platform, with emphasis on the functionality of the platform components and their interactions. Section V describes two use cases for CSC. In Section VI, we discuss our experimental results and assess the performance of the MESON platform, as well as the gains stemming from optimized CSC. Section VII discusses related work. Finally, in Section VIII we highlight our conclusions and outline directions for future work.

II. CROSS-SLICE COMMUNICATION

We hereby discuss the notion of CSC and stress on the need for optimized CSC in the context of edge computing. In this respect, we distinguish between two types of services: (i) *consuming* services, and (ii) *providing* services.

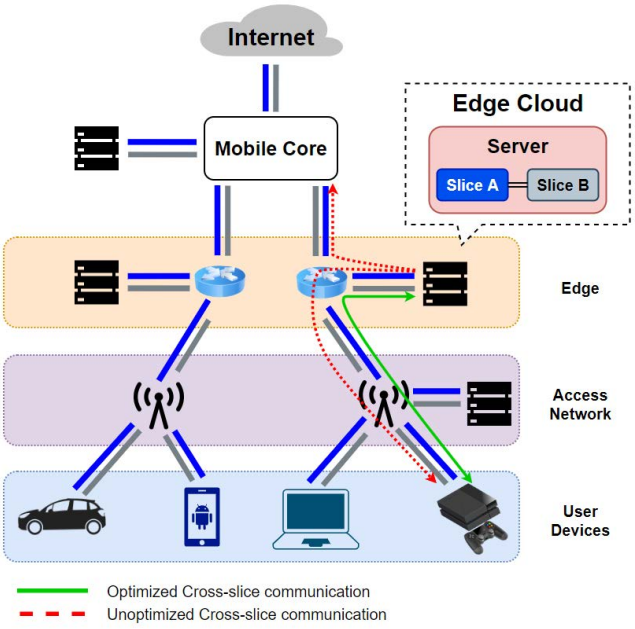

FIGURE 1. Optimized vs. unoptimized cross-slice communication.

CSC enables a *consuming* service to enhance its functionality by gaining access to service elements or functions offered by a *providing* service. Such services may be co-located in the same (edge) computing infrastructure, opening up an opportunity for peering between the two services [9]. Thereby, a *consuming* service can be empowered to consume another (*i.e., providing*) service with very low latency, enhancing its functionality and the experience offered to the end-users.

Various cross-service interactions, benefiting from CSC, can be envisaged. For instance, consider an augmented reality (AR) service co-located with a social media (SM) service. The AR service retrieves user preferences from the SM service in order to offer an enhanced personalized AR experience to users. In this example, the AR is the *consuming* service, whereas the SM fulfills the role of the *providing* service. In Section V, we elaborate on two use cases for CSC: (i) video streaming, where CSC is utilized for peering between video caches deployed in co-located slices, and (ii) robotics in the context of Industry 4.0, where establishing synergy between robotics applications can accelerate mission-critical operations, such as inventory loading/unloading.

Such cross-service interactions cannot be established efficiently, primarily because of the strong resource isolation enforced among co-located network slices (which inhibits unauthorized access to resources granted to other slice tenants) [13]. The impact of network slice isolation is more severe for edge computing, hindering opportunities for synergies among slice tenants. We explain this limitation using Fig 1, which illustrates two slices co-located within an edge computing facility (*e.g.,* micro-DC). Routing policies mandated by the need for resource and traffic isolation will enforce a detour of the CSC traffic through the mobile core, pro-longing the communication path, which is illustrated by the red dotted line in Fig. 1 [14]. Note that as service deployment is pushed towards the network edge (*i.e.,* in order to better satisfy the needs of latency-sensitive services), the stretching of the CSC path will be substantial.

To alleviate this limitation, we seek the establishment of direct *peering* between co-located slices, which we term as *optimized CSC*. Our main goal is to confine the CSC traffic within the boundaries of the DC that hosts the two communicating slices. As such, there will be no need for the CSC traffic to traverse the entire mobile network infrastructure. The optimized CSC is illustrated by the green line in Fig. 1. We emphasize that the establishment of optimized CSC should not be at the expense of lower isolation (*i.e.,* we do not wish to compromise isolation for more efficient CSC). In Section IV, we will exemplify how the proposed CSC orchestration platform enables optimized CSC in a secure and controlled manner, and based on the agreement established between the slice tenants.

Optimized CSC is expected to bring significant benefits both for slice tenants and network operators. First of all, the experience from cross-service interactions will be enhanced, since CSC traffic will traverse a short communication path with potentially minimal latency. As such, latency-sensitive applications will not have to tolerate additional delays stemming from the need to access other services in co-located slices. Furthermore, slice tenants acting as Service Providers will be in position to expand their customer base, due to the enhanced service offerings compared to similar services from other providers. At the same time, the Service Providers will benefit from lower expenditure, since optimized CSC will obviate the need to purchase additional bandwidth from the backhaul and transport network. This will be very critical for bandwidth-intensive services, which become more prevalent in the 5G (and beyond) era. Alleviating the traffic load from the network will be also beneficial for network operators, who could use this spare capacity for other services, which, in turn, can generate additional revenues.

III. MESON ARCHITECTURE OVERVIEW

In this section, we provide an overview of the MESON CSC orchestration architecture, onto which the MESON orchestration platform is based. More specifically, we outline the main roles involved in CSC, present the MESON architecture components, and also exemplify their main interactions. We will also take the opportunity to introduce the notion of *CSC slice*, which is instrumental in the establishment of CSC.

Since we aim at fostering cross-service interactions, we introduce additional business roles, as beneficiaries of CSC. More specifically, we consider the following roles:

Infrastructure Provider, who owns the (edge) cloud infrastructure and offers resources under lease basis for slice deployment.

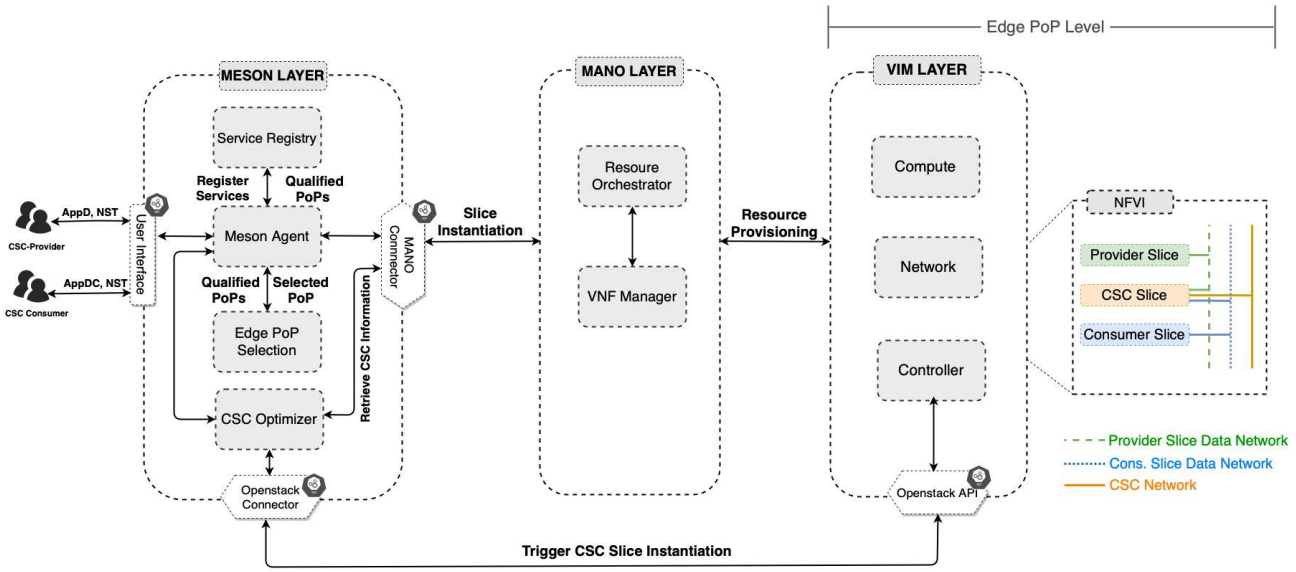

FIGURE 2. MESON architecture.

CSC-enabled Service Provider (CSC-SP), who is the tenant of a slice leased by the Infrastructure Provider. The CSC-SP deploys services onto certain PoPs, which are available for consumption via means of CSC by other slice tenants.

CSC-enabled Service Consumer (CSC-SC), who is a slice tenant that consumes services from co-located slices (*i.e.,* operated by a CSC-SP) via CSC.

The MESON architecture fulfills all the main requirements for the discovery of CSC-enabled services and the establishment of CSC between two slice tenants (*i.e.,* CSC-SP and CSC-SC). More precisely, MESON supports the following functionalities:

- The advertisement of services (by a CSC-SP), which can be consumed by another slice tenant (*i.e.,* a CSC-SC) via CSC.
- The expression of interest (by a slice tenant and potential CSC-SC) for the consumption of a CSC-enabled service.
- The matching of CSC interests with advertised CSC-enabled services for the identification of the most suitable CSC pair, based on service requirements and peering policies.
- The selection of the most appropriate PoP for the deployment of a CSC-enabled service, given that a CSC-SP may have deployed a CSC-enabled service on multiple PoPs.
- The establishment of a communication path for a cross-service interaction between two slice tenants.

These functionalities stem from the following two CSC establishment scenarios, both of which are supported by MESON:

- **Uncoordinated CSC**. This scenario involves the establishment of CSC after both slices have been in place. As such, CSC establishment is decoupled from the placement and deployment of the slice of CSC-SC. In this scenario, opportunities for CSC establishment are investigated within the particular edge PoP that the slice of CSC-SC has been deployed. Consequently, uncoordinated CSC entails less complexity, since there is no need to look up services on other PoPs, and, subsequently, select the PoP that best meets the needs of the service that will be consumed.
- **Coordinated CSC**. In this scenario, CSC is established alongside the placement of the slice of CSC-SC. As such, both operations need to be coordinated which requires more care in comparison to the previous CSC scenario. Coordinated CSC raises additional requirements, such as the ability to select the most suitable edge PoP for the placement of the slice, and, subsequently, the establishment of communication with a CSC-SP. In this respect, a service offered by the MESON architecture, namely *Edge PoP Selection*, is particularly tailored to the needs of this scenario.

The MESON architecture extends the ETSI NFV MANO reference architecture [5] in order to facilitate the discovery and consumption of CSC-enabled services in a controlled manner, exploiting service co-location. MANO provides NFV management and orchestration functionality, spanning the following two layers: (i) the Virtualized Infrastructure Manager (VIM), which is responsible for the management of the virtualized infrastructure across all resource types (*i.e.,* compute, storage, network), and (ii) the NFV orchestration (NFVO) layer, which provides support for VNF lifecycle management, as well as VNF state management.

MESON introduces a CSC orchestration layer on top of the two-layer MANO architecture, as illustrated in Fig. 2. More specifically, the MESON layer encompasses the following components:

- **Service Registry**, which is responsible for CSC-enabled service discovery. The Service Registry allows a CSC-SP to advertise services which can be consumed by other slice tenants via CSC. A CSC-enabled service offering encompasses specifications of the service and the CSC policy. The latter contains the highest computing and network resources that can be allocated for the service offering, as well as any scale-out scheme supported by the CSC-SP in order to cope with increased service demands. This information can be conveyed to the Service Registry using MEC application descriptors (`AppD`) [9]. Likewise, a CSC-SC can submit requests to the Service Registry for the consumption of services via CSC. To this end, the Registry performs a lookup across the registered service instances in order to identify suitable CSC-enabled services. In the case of *Coordinated CSC*, the search space may encompass a wide range of edge PoPs. Eventually (in this scenario), the Service Registry will return a list of all edge PoPs with CSC-enabled services that match the CSC-SC request and also comply with the CSC policy expressed by the CSC-SP.
- **Edge PoP Selection**, which identifies the most suitable edge PoP for the deployment of a CSC-enabled service on behalf of a CSC-SC. PoP Selection is only relevant to the *Coordinated CSC* scenario, where the slice of the CSC-SC has not been yet instantiated, and, thereby, its placement needs to be optimized jointly with CSC establishment. Edge PoP selection comes into play after the Service Registry has identified the service instances that match the service requested by the CSC-SC. In particular, edge PoP selection receives the corresponding list of PoPs (*i.e.,* of the matching service instances), and is entitled with the task of PoP ranking, based on *hard* and *soft* requirements from CSC-SP. The *hard* requirements are associated with attributes, such as the location (availability zone) of the PoP, whereas the *soft* requirements are related to computing and network performance indicators (*e.g.,* service response time), cost parameters, and technical support. In order to generate the PoP ranking, the edge PoP selection module employs a multi-criteria decision making (MCDM) approach, which is exemplified in [15].
- **MESON Agent**, which performs the following main tasks: (i) exposes an interface to slice tenants for the expression of CSC-enabled service advertisements and interests (by CSC-SPs and CSC-SCs, respectively) and (ii) coordinates the CSC service discovery and instantiation workflows within the MESON layer. The MESON Agent is also responsible for interoperability with the underlying MANO layers via API calls directed to the NFVO. After the edge PoP Selection has generated the edge PoP ranking (in the case of *Coordinated CSC*), the MESON Agent configures a network slice description with all required instantiation parameters, which is, in turn, conveyed to MANO for the deployment of the CSC-SC's slice. After both slices (*i.e.,* of CSC-SP and CSC-SC) are in place, the MESON Agent triggers CSC setup. To this end, MESON handles CSC establishment through the instantiation of an intermediate slice, dedicated to CSC. The benefits of this approach are two-fold: (i) CSC traffic remains isolated from the rest of the traffic within the edge PoP and (ii) the CSC slice facilitates the deployment of network processing functionality (*e.g.,* in the form of VNF chains), such as packet inspection and monitoring for the purpose of CSC policy control and enforcement, especially from the side of CSC-SP.
- **CSC Optimizer**, which is responsible for the lifecycle management of the CSC slice. More precisely, this module collects all required information (*e.g.,* network IDs, IP addresses, etc.) in order to instantiate the CSC slice and, subsequently, generates the required packet forwarding entries for secure communication between the CSC-SP and CSC-SC slice. Upon the preparation of CSC slice specification with all required instantiation parameters, the CSC optimizer conveys this specification to the VIM for the CSC slice deployment. In contrast to the CSC-SC slice, the instantiation of the intermediate CSC slice is handled directly by the VIM, without any intervention from the NFVO layer. This raises the need for interoperability between the MESON and the VIM layers, which we further explain in the MESON platform description (Section IV).

For a more elaborate description of the CSC discovery and instantiation workflows, we direct the interested reader to [9]. The same work also provides further details about the MESON descriptors (*e.g.,* `AppD`).

IV. MESON ORCHESTRATION PLATFORM

In this section, we describe in detail the features and implementation of the MESON platform, which realizes all the CSC orchestration operations of the MESON architecture specification (Section III) [9]. To this end, we delve into the system implementation and internal architecture of each component of the MESON layer (*i.e.,* Agent, Optimizer, Edge PoP Selection, and Service Registry). Furthermore, we shed light into the interaction of modules that comprise each MESON component. We also discuss the various cross-layer interactions invoked in order to deploy and configure network slices towards the establishment of CSC.

A. MESON AGENT

The MESON Agent coordinates the communication within the MESON layer for operations, such as the discovery of CSC-enabled services and CSC establishment. In essence, it functions as a communication hub between the other components residing in the MESON layer (*i.e.,* Service Registry and Edge PoP Selection).

The MESON Agent comprises several distinct modules and its internal architecture is depicted in Fig. 3. The core component of the MESON Agent is the *Request Controller*,

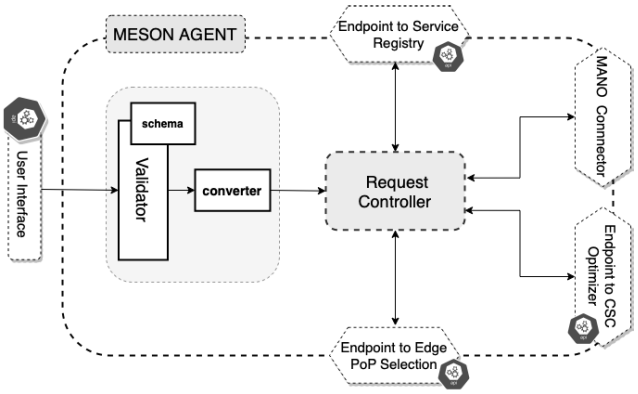

FIGURE 3. Internal architecture of MESON agent.

TABLE 1. CSC optimizer processes.

CSC Optimizer Process	Specification
Tenant Information Retrieval	Essential information about CSC-SC/CSC-SP slices and the corresponding VNFs are retrieved from the Service Registry.
NFVO Information Retrieval	Instantiation-related data, such as slice hosts, VNF connection points, and virtual networks are retrieved from the NFVO.
CSC Network Creation	A virtual network for the CSC-slice VNFs is created using the VIM's Rest API.
Cloud-Init Configuration	CSC-slice VNFs configuration and forwarding entries for enabling network connectivity between CSC-SC and CSC-SP slices.
CSC Slice Instantiation	CSC slice and CSC VNFs deployment on the Edge PoP directly through VIM's REST interface, taking into account slice co-location constraints.

which handles all the communication among MESON-layer components. The Request Controller acts as an internal orchestrator. The interaction among the components is achieved through API calls using Python's request library. The Request Controller receives the offered services along with the candidate edge PoPs from the Service Registry, and subsequently relays this information to the PoP Selection component. The Request Controller also triggers the process of CSC-SC slice instantiation through a REST API call to the NFVO layer.

Another key component of MESON Agent is the *Validator*. Exploiting the Cerberus Python library,[1] the main task of the Validator module is to determine whether the uploaded descriptors are in the correct YAML format. This validation is carried out in comparison with a well-defined JSON schema. Furthermore, the *Converter* module is responsible for the conversion of descriptors from YAML to JSON format.

The MESON Agent has been implemented as REST API in Python using the Flask.[2] The communication between the rest of the MESON modules is achieved through REST API calls, where all the appropriate information is exchanged in JSON format.

B. CSC OPTIMIZER

All the required procedures for CSC establishment are undertaken by the CSC optimizer. This component utilizes the scheduler filters of the underlying VIM (in our case, OpenStack), and, based on the NFVO data regarding the involved CSC-SC and CSC-SP slices, performs the optimized placement for the CSC slice. More specifically, the CSC Optimizer seeks to co-locate the CSC slice on the same compute host (*i.e,* server) where the CSC-SC slice is deployed, towards the optimization of CSC in terms of latency and throughput, as we show in our evaluations (Section VI).

The functionalities of the CSC Optimizer are briefly explained in Table 1. These processes require interaction among all three layers of the MESON architecture. More specifically, slice tenant information regarding CSC establishment is retrieved from the corresponding `AppDs`, which are stored in the Service Registry. In addition, the CSC optimizer queries the NFVO layer in order to collect all the required information (*i.e.,* underlying networks, computing hosts, etc.), which is crucial for CSC slice instantiation.

Subsequently, the required network configuration takes place within the VIM in order to enable slice connectivity (*e.g.,* service chaining). More precisely, leveraging the *cloud-init* industry standard, the CSC optimizer conveys all the appropriate forwarding entries to the CSC VNFs during the instantiation, which then undertake the orchestration of the communication between the slices of CSC-SP and CSC-SC. The CSC VNFs route the traffic between the networks of the two communicating slices, as illustrated in the NFVI layer of Fig. 2.

The MESON-layer interactions, as well as the cross-layer interactions with the CSC Optimizer, are realized via REST API calls. The CSC Optimizer is implemented in Python.

C. SERVICE REGISTRY

Service Registry is the component of the MESON layer responsible for storing and retrieving information concerning services offered for CSC and the PoPs where these services are available for consumption. The information for every registered service includes its technical aspects and also the aspects of the infrastructure that the service is offered on.

More specifically, the Service Registry consists of two components: (i) a Web service implemented in Java and based on the Spring Boot framework,[3] and (ii) a MongoDB database instance. The main task of the Web service is to expose a REST API, allowing CSC-SPs to register their respective service offerings. The offered services are submitted in JSON format and contain the same information as their YAML counterparts mentioned previously. Furthermore, the Web service provides the functionality to retrieve and forward

[1] https://docs.python-cerberus.org/en/stable/
[2] https://flask.palletsprojects.com/en/2.1.x/
[3] https://spring.io/projects/spring-boot

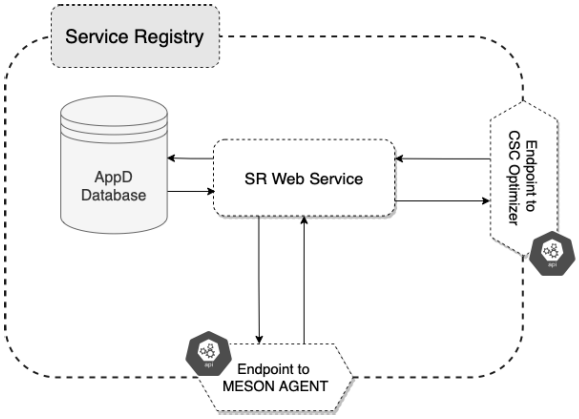

FIGURE 4. Internal architecture of service registry.

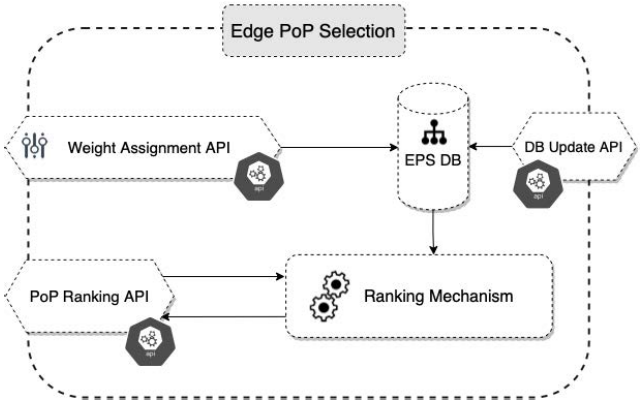

FIGURE 5. Internal architecture of edge PoP selection.

service offerings that match certain requirements, such as maximum cost and/or performance given by a CSC-SC. This effectively means that, if a service is offered at several PoPs, each one with its own specification and cost, the Registry will filter the list in order to identify the entries that match the particular requirements of the CSC-SC.

The matching procedure in the Service Registry is utilized for service discovery, which is necessary for the CSC establishment process. Service discovery binds the CSC interests of a CSC-SC with the corresponding offerings from CSC-SPs that are already registered in the database. Both service offerings and interests comprise part of the descriptors sent to the Service Registry, as explained in Section III.

In the MongoDB instance, the information is organized into two collections, *i.e.,* one that contains the available services and one that contains the available PoPs. The latter can be further associated with Key Performance Indicators (KPIs). During the process of registering offered services in the database, this information is enriched with the KPIs of the PoP that each service is offered on. Note that all requests to the Web service are supplied by the MESON Agent, as the Service Registry is not directly accessible by components external to the platform (*e.g.* a browser). This interaction is illustrated in Fig. 4.

D. EDGE POP SELECTION

In the context of CSC optimization, the selection of the most suitable edge infrastructure for the deployment of CSC-enabled services is very critical. Edge PoP Selection (EPS), as part of the MESON architecture, provides this functionality. In particular, the candidate edge infrastructures, as extracted from the matching process of the Service Registry, are evaluated based on specific KPIs advertised by each infrastructure provider, as well as based on the preferences of each slice owner, expressed as weight values for the corresponding evaluation criteria. For the ranking process, we adopt a fuzzy Multi-Criteria Decision Making (MCDM) approach, which are widely used for solving the provider selection problem [16]. The EPS framework has been developed in Python as REST API using Flask. In addition, communication with the rest of MESON layer components is achieved through REST API calls, where the essential information about the EPS internal mechanisms is exchanged.

The core component of the EPS framework is the PoP Ranking Mechanism (PRM), which is based on the Fuzzy Analytic Hierarchy Process (FAHP) methodology [15]. This mechanism is able to process different types of data simultaneously, such as boolean, numerical or even linguistic (fuzzy) values. As such, the evaluation of the candidate PoPs can take into account data that is related to both technical and non-technical characteristics. Specifications about the individual methods of the ranking process, alongside with an analytical description can be found in [15]. The PRM functionalities have been implemented in Python. Specifically, the operations of triangular fuzzy numbers with the corresponding classes, the calculations of the comparison vectors and finally the calculations of the FAHP method are defined. The required data for conducting the evaluation and afterwards the selection of the Edge PoP is obtained from specific REST interfaces. Fig. 5 shows the internal architecture of the EPS framework.

The most critical functionalities of EPS are the following:

1) HIERARCHICAL STRUCTURE DEFINITION

As mentioned earlier, the PRM is based on FAHP. This method utilizes a hierarchical structure that consists of (i) KPIs of several types and (ii) attributes, which summarize a group of KPIs. Our EPS implementation provides a SQL database for storing the hierarchical structure with the corresponding interface for initializing or updating it.

2) WEIGHT ASSIGNMENT

A CSC-SC, requesting the consumption of a service via CSC, is empowered to assign weight values to some of the top level criteria of the hierarchical structure. These values are conveyed through MESON descriptors (*i.e.,* `AppD` [9]) to the MESON Agent, which transfer these values to the EPS via an

API call to the corresponding interface (Weight Assignment API) in JSON format. The EPS receives the message, checks the consistency of the weights and, subsequently, assigns them to the corresponding criteria of the hierarchical structure stored in the database.

3) EDGE POP CANDIDATES RETRIEVAL

Every candidate Edge PoP is associated with an `AppD`, which contains the data for the corresponding KPIs. Thus, in the corresponding EPS interface (PoP Ranking API), a list of `AppDs` in JSON format is posted by the MESON Agent.

4) EDGE POP RANKING AND SELECTION

The KPI values for each candidate Edge PoP are used as input for the multi-criteria decision algorithm. The Edge PoP with the highest evaluation score is computed by the Ranking Mechanism. The output of the Ranking Mechanism is a vector that contains the normalized ranking score of every candidate Edge PoP. Then, the corresponding `AppD` is sent back to the MESON Agent in response to the previous REST API call.

Further details of the KPI inputs, the step-by-step ranking process, the weight assignment, and the adjustments on the FAHP in order to meet the functional requirements of the EPS, are available in [15].

V. USE CASES

In this section, we describe two use cases for optimized CSC in relation to the proposed MESON platform.

A. VIDEO STREAMING

Content Delivery Networks (CDNs) and caching have been used extensively in network infrastructures, as means to handle the increasing load caused by the continuous video streams [17], [18]. Concurrently, by exploiting 5G capabilities, network operators can deploy storage services in virtualized caches (vCaches) close to the end user's location in micro-datacenters (μDC) [14], [19]. This creates an opportunity for multiple virtual network operators (VNOs), co-existing in the same μDC, to establish synergies for exchanging content located in their vCaches, by bypassing the unnecessary traffic routing through the network core.

To showcase such a synergy, we apply a commercial CDN application for IPTV live streaming and Video on Demand (VoD) services, namely *fs|cdn Anywhere*, in this setting. The main components of this application encompass the Edge Cache, which streams the requested content, stored locally, directly to the user's device, and the origin server. The latter is contacted by the Edge Cache to fetch content that does not exist in its local cache. An alternative approach is to let the Edge Cache receive the missing video chunk from another co-located Edge Cache, which is the scenario currently under consideration.

In Fig. 6, each of VNOs A and B are running an instance of the Edge Cache of the CDN application in their slices, which are located on a cloud provider's μDC. Furthermore,

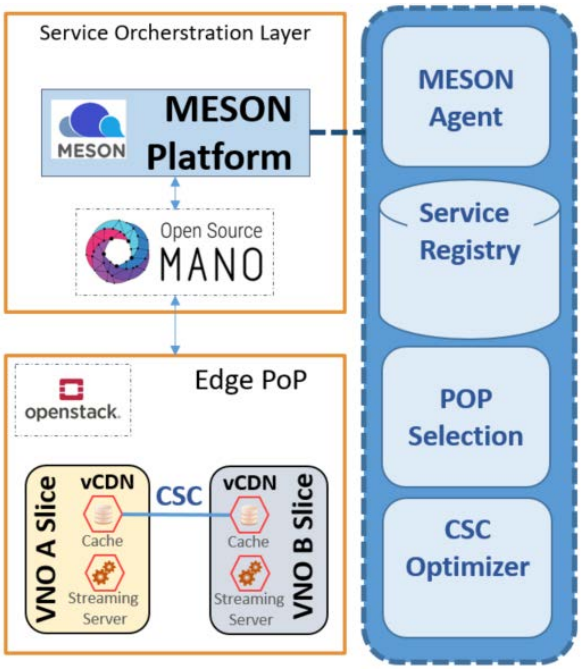

FIGURE 6. Video streaming use case.

each Edge Cache maintains its own virtual cache memory for content storage. Initially, a request for a continuous video stream is dispatched to the cache of VNO A. If not successful, the request is forwarded to the cache of VNO B. Under normal operations, the request of cache A to cache B for content would have been routed through the network core, outside the boundaries of the DC, resulting in latency noticeable by the end user. This can be avoided by establishing CSC between the slices of the VNOs through the MESON platform.

Besides a reduction in bandwidth consumption and more efficient use of the physical infrastructure, there are additional advantages using this approach. From the user viewpoint, the average perceptible latency is reduced, since there is a high chance that the requested content can be found in another Edge Cache. Furthermore, there is a reduction in the load of the origin server, when the content is provided by the Edge Caches. Lastly, network operators stand to gain from the more efficient utilization of underlying infrastructure by being able to minimize the resources allocated in temporary storage services, as well as the cost due to the reduction of traffic.

B. INDUSTRY 4.0

The adoption of modern network technologies in smart manufacturing is an open challenge for 5G community. Edge computing and network slicing aim at enabling the dynamic programmability of the industrial processes and guaranteeing any time and mission requirement. To this end, two main impediments must be overcome [20]. At first, the production workflows rely on heterogeneous and multi-vendor

equipment that raise many trust management and inter-operability issues. Second, the industrial operators are commonly not expert at networks, service deployment and their life-cycle management. As such, they usually outsource the network management of the factory floor to micro-operators.

Under this complex setting, we demonstrate the capabilities of CSC for Industry 4.0 in the case of industrial robots. In particular, we focus on time/mission critical robotic applications that must be deployed at the network edge, due to the limited computing resource and battery life of the robots [21]. Furthermore, many open-source platforms operating systems (*i.e.,* Robotic Operating System - ROS) facilitate the orchestration of robotic applications on edge PoPs.

In particular, this use case demonstrates how two warehouse robotic services, deployed by different vendors, communicate and benefit from CSC [22]. Fig. 7 shows the deployment of two slices and the CSC establishment leveraging the capabilities of MESON platform. The first robotic slice is responsible for loading commodities in various locations and consists of three individual services in form of VNFs. The Localization (LOC) service is responsible for tracking the position of the robot and informs the other services. The Mission Planner (MP) service computes the route that the robot must follow to deliver the objects in specific locations that the Asset Loading (AL) service dictates. The AL service continuously updates the status of the warehouse and informs the Asset Unloading (AU) service for the availability of specific items through the CSC mechanism. The second slice is responsible for unloading products. The LOC and MP services of this slice have similar functionality with the AL slice. The AU service is invoked by external services and triggers the MP planner for collecting the required items. Furthermore, the AU service communicates with the AL service to keep track of the availability of assets in the warehouse.

The benefits of the CSC are twofold [22]. First, from the vendor perspective, the mission completion time is significantly reduced, due to the immediate update of the warehouse status. Second, the infrastructure provider and the network operator benefit from CSC, since fewer computing resources are allocated for the deployment of VNFs and the generated traffic is confined within the edge PoP. In Section VI-E, we quantify these benefits of CSC in terms of computing and network resources.

VI. EVALUATION

In this section, we perform an extensive evaluation of the MESON platform, which takes place in three distinct phases. Initially, we carry out a performance evaluation of the MESON platform (Section VI-B). Subsequently, we conduct performance tests in an actual edge infrastructures environment, as described in Section VI-A. These performance tests aim at quantifying the performance overhead of the various operations and interactions within the MESON

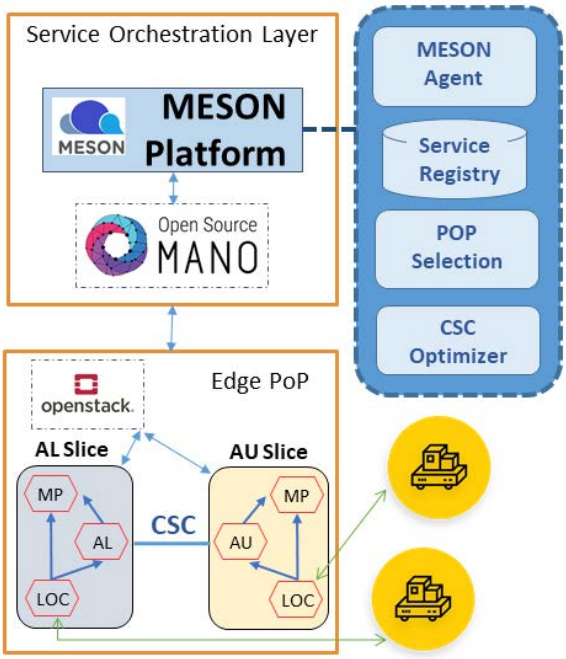

FIGURE 7. Industry 4.0 use case.

layer, with respect to CSC establishment and orchestration. In addition, to assess potential issues of scale, we resort to simulations. In this respect, Section VI-C presents simulation results regarding the most critical operations at the MESON layer, *i.e.,* Service Discovery and Edge PoP Selection. Beyond these performance and scalability tests, we further seek to quantify the gains from CSC in terms of service-level metrics. To this end, we present experimental results from CSC use-case evaluations, *i.e.,* video streaming (Section VI-D) and robotics in the context of Industry 4.0 (Section VI-E).

A. EXPERIMENTAL SETUP

Our setup for the MESON platform evaluation encompasses three experimental facilities in different locations. More specifically, all MESON-layer components and the NFVO (realized by OSM) are deployed at the testbed of University of Macedonia (UOM). Both MESON and OSM are deployed as virtual machines (VMs), each one assigned with 4 CPU cores, 4 GB of RAM, and 50 GB of storage. All VMs are co-located on the same server. In addition, the experimental facilities of Intracom Telecom (ICOM) and National Technical Univerity of Athens (NTUA) are utilized as edge cloud PoPs for the deployment of the video streaming and Industry 4.0 use cases, respectively. Each of the two edge PoPs are managed by a separate OpenStack instance [23]. NTUA's facility relies on OpenStack v18.3.2 for the management of two servers with 16 Xeon CPU cores, 64 GB RAM, and 1 TB of storage in total. ICOM's infrastructure also utilizes OpenStack (Victoria version), managing 4 CPU cores with 8 GB RAM and 50 GB of storage in one server.

TABLE 2. Network slice request specifications.

Request Type	Type 1	Type 2	Type 3
Number of NSes	1	2	3
VNFs (per NS)	1	1	1
vCPU Cores (Total)	2	4	6
RAM (Total)	2GB	4GB	6GB
Disk (Total)	10GB	80GB	120GB

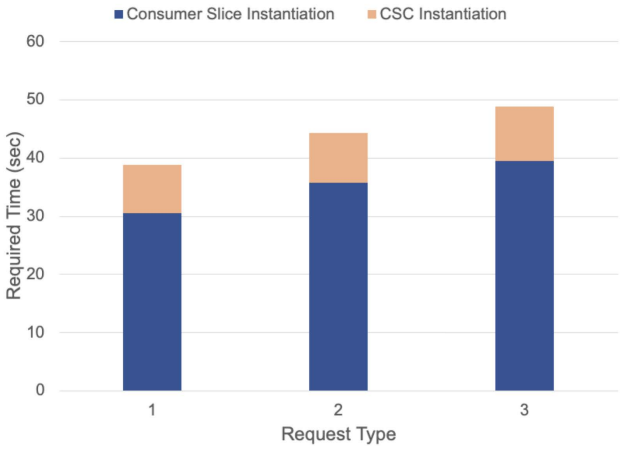

FIGURE 8. CSC instantiation overhead.

B. PERFORMANCE EVALUATION

Our performance tests are centered around the performance overhead of the MESON layer, with respect to CSC establishment. We further account for the interactions of the MESON layer with the VIM and the NFVO, which entail information retrieval, network configuration, and instantiation process execution. We note that the following performance tests pertain to the coordinated CSC case (see Section III), where the CSC-SC slice is deployed alongside with the establishment of CSC (which requires the instantiation of the intermediate CSC slice).

To quantify the performance overhead of CSC establishment, we measure the delay incurred for the instantiation of a slice with and without CSC. The results are based on requests of three different types, associated with different network services (NSes), as well as with different types of VNF instances, as shown in Table 2. This table further depicts the total resource demands for each slice request. According to Fig. 8, CSC establishment incurs less than 9 secs across all request types. This delay constitutes a small fraction of the overall time required for the instantiation of the CSC-SC slice.

To shed more light into the overhead introduced by MESON, we measure the execution time of each individual step for CSC instantiation. The steps and their corresponding delays are shown in Table 3. According to these results, CSC instantiation is dominated (time-wise) by CSC network creation and CSC VNF instantiation. More precisely, the completion of both steps requires around 8 secs on average, which corresponds to 95% of the total CSC instantiation

TABLE 3. CSC instantiation breakdown.

CSC instantiation steps	Execution Time (s)			
	Average	Min	Max	SD
Tenant data retrieval	0.0044	0.0043	0.0046	0.0001
VNF OSM data retrieval	0.1765	0.1723	0.1793	0.0025
Cloud-init configuration	0.1783	0.1726	0.1856	0.0041
Cross-layer API calls	0.188	0.1851	0.1919	0.0028
CSC network creation	4.179	3.91	4.57	0.2095
CSC VNF instantiation	3.8958	3.5608	4.3405	0.2213
Total	**8.434**	**7.82**	**9.28**	**0.4383**

time. Note that both CSC network creation and CSC VNF instantiation comprise processes executed by OpenStack (in other words, these delays are incurred by OpenStack rather than by MESON per se). On the contrary, MESON, and more specifically, CSC Optimizer, handles the information retrieval for the slices of CSC-SP and CSC-SC, and also executes calls to the lower layers (VIM/NVFO) in order to trigger CSC instantiation. As shown in Table 3, all these MESON-related processes incur negligible delays for all types of CSC requests, and, in extension, a minimal overhead on slice instantiation.

C. SCALABILITY TESTS

We perform scalability tests on CSC service discovery (handled by the Service Registry component) and edge PoP selection (carried out by the respective component at the MESON layer). Both processes are triggered upon a request for the consumption of a service via CSC. Similar to our performance tests (*i.e.,* Section VI-B), we focus on the case of coordinated CSC, which entails a higher degree of coordination, and, thereby, more complexity.

In particular, we assess the scalability of these MESON-layer operations with a diverse range of edge PoPs (10 to 100). For these tests, the Service Registry's database is populated with a sufficient number of service instances, stored in the form of `AppDs`. Furthermore, 100 `AppDs`, which correspond to CSC-enabled services, are assigned to each registered edge PoP.

Fig. 9 illustrates the time spent for service discovery and edge PoP selection across the range of PoPs under consideration. The delay incurred for both operations combined lies in the range of 10 to 40 ms. According to Fig. 9, service discovery and PoP selection exhibit different scaling properties. More precisely, the execution time of the former increases linearly, whereas the time spent for PoP selection yields a polynomial-like increase with a larger number of PoPs. This stems from the MCDM algorithms employed for PoP ranking. Given that slice instantiation is completed in the range of tens of seconds (Section VI-B), the delays incurred for service discovery and PoP selection are deemed insignificant in comparison.

D. VIDEO STREAMING USE CASE EVALUATION

We proceed with the evaluation of optimized CSC applied to the video streaming use case (Section V-A) in order to

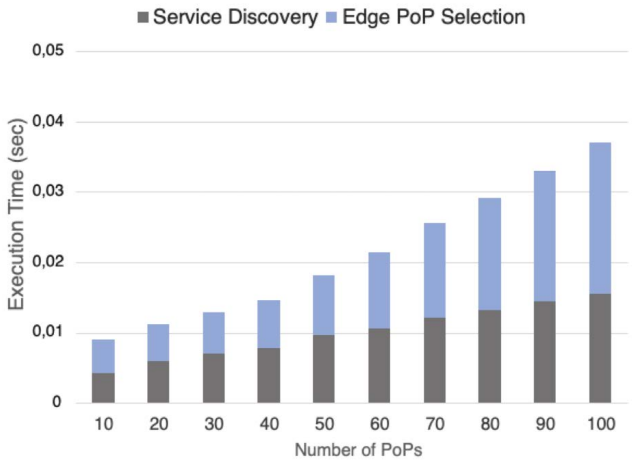

FIGURE 9. Execution time of service discovery and edge PoP selection.

show the benefits of CSC for CDN video streaming scenarios, in terms of reliably delivering high-quality video streams to a large, geographically distributed and potentially mobile customer base. The main challenge in such video streaming scenarios is that the dimensioning and coverage footprint of a particular CDN provider appears insufficient to deal with cases of spatio-temporal fluctuations in service demand[4] In this direction, we showcase improvement in video files' fetching time and throughput via the employment of CSC and the consumption of a co-located CDN edge caching service that is provided on the same Edge PoP. The throughput increase and delay decrease are automatically "sensed" by the multicast emulation protocols used by the CDN service (HLS[5] or MPEG-DASH) activating mechanisms for maintaining or upgrading the quality of the video chunks being served and, thus, the end-user experience.

Our experimental setup encompasses two interoperating remote testbeds: (i) the MESON-layer components and OSM that are deployed at the premises of UOM and (ii) an OpenStack installation representing an Edge PoP, which is deployed at ICOM cloud facilities in Peania. In particular, a request for the establishment of CSC is conveyed to the MESON Agent, which, in turn, instructs the OpenStack Edge PoP to instantiate an edge cache that will provide video streaming content to end users. This experiment is conducted twice, *i.e.,* once with a CSC link present to route the traffic between this edge cache instance and a colocated edge cache service on the same PoP, and once without CSC. In the first case, a secondary edge cache slice is already present in the infrastructure with the required content for streaming, essentially playing the role of the service provider. In both cases, we measure the throughput and the time required to store content in the edge cache from the adjacent edge cache (*i.e.,* via CSC and without CSC). We conduct a total of ten tests for each case and present the mean values for each metric.

[4]The Case for a Virtualized CDN (vCDN) for Delivering Operator OTT Video, AKAMAI White Paper.

[5]https://developer.apple.com/streaming/

TABLE 4. Throughput (Mbps) when the content is requested from (i) an adjacent edge cache (via CSC) and (ii) the origin server (without CSC).

Content Origin	Minimum	Maximum	Average
Edge cache	65.4	83.5	75.05
Origin server	40.2	52.3	44.72

TABLE 5. Latency (sec) when fetching the content from (i) an adjacent edge cache (via CSC) and (ii) the origin server (without CSC).

Content Origin	Minimum	Maximum	Average
Origin server	1.80	2.02	1.934
Edge cache	0.72	1.23	0.97

The corresponding results appear in Tables 4 and 5. When the content is fetched towards the edge cache of the CDN with support of a service provider adjacent edge cache via CSC, throughput is 75 Mbps on average (Table 4). In contrast, without CSC, the content is fetched to the edge cache solely through a remote origin server, which leads to diminished throughput (44.72 Mbps on average). Significant gains are also observed in terms of content storage, when CSC is being utilized. In particular, we observe a nearly 100% increase in the time required to store the content, when switching from an edge cache (via CSC) to the origin server (without CSC). We have validated the effect of these technical KPI improvements on the video quality delivered to end-users through a demo [24].

E. INDUSTRY 4.0 USE CASE EVALUATION

We hereby discuss the evaluation of MESON in the context of the Industry 4.0 use case. In analogy to Section VI-D, our objective is to quantify the benefits of CSC in terms of AL/AU service-level performance indicators. Furthermore, we highlight the importance of the placement functionalities of MESON through different deployment scenarios (involving CSC-SC and CSC slice placements).

Our experimental setup is as follows: (i) the MESON-layer components and OSM reside at the testbed of UOM and (ii) an OpenStack instance is deployed at the premises of NTUA, alongside the required slices (*i.e.,* AL slice, AU slice, and CSC slice) for the realization of the corresponding use case. Note that the AL slice corresponds to the role of CSC-SP slice. The `AppDs` of the AL/AU slices are forwarded to the MESON Agent including CSC specifications, whereas Network Service Descriptors (NSDs) and Slice Templates (NSTs) are uploaded to OSM. The SW images required for the deployment of AL and AU services are onboarded into the NTUA's OpenStack instance. Interrelated services are implemented using Python 3.6, while the communication between them is carried out through REST interfaces. The AL and AU slices correspond into two AlphaBot[6] robotic platforms. As mentioned in Section V,

[6]https://www.waveshare.com/wiki/AlphaBot2/

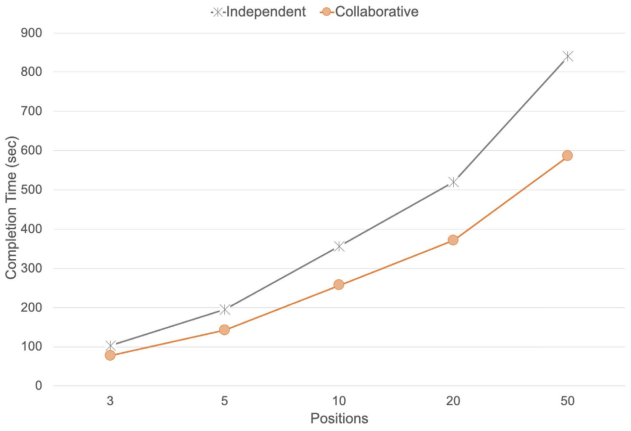

FIGURE 10. Mission completion time.

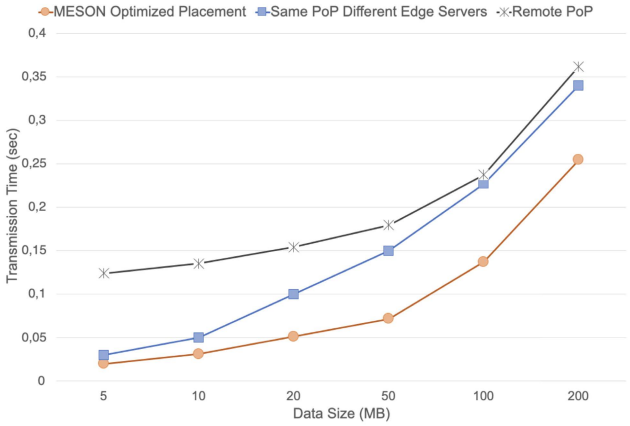

FIGURE 11. Transmission time between AL and AU with different CSC placements.

each of the two robots follow a circular path of locations in the operating ground, which simulates an industrial environment, in order to perform the automated load and unload system.

In the following, we discuss our results from two experiments aiming at different service-level indicators in relation to the Industry 4.0 use case.

1) MISSION COMPLETION TIME

In this experiment, we assess the impact of CSC on a AL/AU mission. To this end, a specific mission is performed from the robot fleet. Concerning the mission, unique type of assets are stored in each one of the k different storage points of the operating ground and they are available for retrieval by the AU robot. The scope of the AU robot is to collect a set of k assets (one from each storage point) following a circular path, and then return that set to the starting point. As assets could become out of stock at some storage points, the aim of the AL robot is to restore the load if needed. This mission initially is performed with the robots operating in an individual manner, where the AU unloads assets on its path and, at the same time, the AL robot checks every storage point for potential low stock. Afterwards, the robots operate in a collaborative manner via CSC. In particular, the AU slice triggers the AL slice only when a shortage is detected, which obviates the need of the AL robot to perform periodical scans for shortages. For these two scenarios, the mission completion time is measured for 25 loops per different number of storage points. Over the course of the experiments, the time required for the robots to move between two locations is considered fixed.

The average mission completion times as a function of the number of storage points are depicted in Fig. 10. The results demonstrate that the collaboration enabled by CSC leads to a significant reduction of the mission completion time. Specifically, it becomes apparent that, even for relatively few storage points, the gains are close to 25%, while this performance gap expands further (approximately 30%) when the number of storage points is in the order of tens.

2) TRANSMISSION TIME

We hereby quantify the gains of optimized CSC (as promoted by MESON) in comparison with alternative CSC placements. In particular, we assume the collaboration of the AU and the AL slice, and consider the following three CSC placement scenarios: (i) MESON optimized placement, where the AU, the AL and the CSC slice are deployed within the same edge PoP and server, (ii) the slices are placed on the same edge PoP, but on different servers and (iii) the AU and AL are deployed in different (remote) PoPs. During a mission, data exchange between the slices is required via the CSC slice. The data size varies from 5 to 200 MB. For each data size, the transmission time is measured for each CSC scenario repeatedly for 25 times. The average transmission time for the respective CSC scenarios is illustrated in Fig. 11. The MESON optimized placement leads to a significant reduction in the data transmission time compared to the remote PoP placement, especially for small data sizes (80% reduction). A noticeable margin (in terms of transmission time) is also observed between the two CSC placement scenarios that utilize the same edge PoP. This margin tends to increase (50%) for data sizes beyond 50 MB.

VII. RELATED WORK

This section provides an overview of studies related to cross-service/slice communication and some of the functionalities of the MESON platform.

Vaquero *et al.* present a review on next-generation service orchestration focusing on edge/fog and serverless computing [25]. One of the current challenges is the multi-organisation/-tenant orchestration, which includes service discovery and customizable service composition that are objectives of the MESON platform. Authors in [20] propose a multi-domain network slicing orchestration architecture that facilitates service management across federated domains. Similar to MESON, the Service Broker of this architecture handles slice requests from various stakeholders and is responsible for the management of slice/owner relationships and the scheduling of network slices. A centralized service

support repository maintains abstracted service capability information regarding different administrative domains.

Regarding CSC, authors in [10] tackle the challenging problem of CSC-aware network service embedding (NSE), which fits within the scope of the MESON platform. In this case, NSE optimization is not only based on resource and communication demands related to a particular network service specification; instead, it is also dependant on another service that needs to be consumed. To address the intricacies of NSE with other service dependencies, this study proposes a heuristic that utilizes a VNF embedding tree, *i.e.,* a data structure used to generate the most appropriate embedding sequence. The proposed heuristic achieves network service co-location without any perceptible penalty in terms of embedding efficiency.

Dimolitsas *et al.* present an extension of MESON platform for multi-domain edge infrastructure that provides a distributed service discovery [26]. Based on local service registries and caches, a distributed service discovery, based on time-to-live (TTL) values, efficiently discovers the requested CSC-enabled services and evaluates the candidate edge POPs for slice placement.

Blockchain technology promises to automate several orchestration processes and enable trust between untrusted entities. Backman *et al.* introduce the Blockchain Slice Leasing Ledger Concept for Industry 4.0 verticals [27]. The 5G network slice brokering concept relies on the ability of the mobile network operator/service provider to easily and automatically negotiate with external tenants network slice requests based on the current resource availability from the infrastructure provider. It supports on-demand slice placement. The essential information of every slice is stored in the Slice Ledger. After the negotiation and agreement on service terms, smart contracts are used for monitoring the SLA terms and billing. A potential application of smart factory applications is described as well. BUNKER is a blockchain-based VNF package repository that provides individual VNF packages to the end-users [28]. BUNKER is based on Ethereum BC and provides various features, such as service registration and upgrade, licensing, VNF verification and VNF rating. BUNKER is deployed on the Ethereum BC and leverages smart contracts to automate the transactions between the end-users and the repository. Authors in [29] propose a blockchain-based solution for building tailored-made service chains and establishing cross-service communication. Instead of a centralized repository, a distributed ledger is used for service registration. Smart contracts are used for the negotiation and the establishment of the cross-service communication.

Finally, based on ETSI NFV architecture, authors in [30] present the 5GZORRO architecture that focuses on automated resource orchestration mechanisms. Coupling Blockchain with AI-driven operations, a service provider can select 3rd-party resources and services to automatically re-instantiate its services in a trusted and secure manner across multiple domains.

VIII. CONCLUSION

In this work, we presented the architecture, implementation, and experimental evaluation of the MESON platform that facilitates CSC and service interactions among slice tenants. CSC is established in an optimized manner, through service co-location not merely on the same edge PoP but also on the same server, subject to resource availability. A salient feature of the MESON platform is the instantiation of dedicated slices for CSC, a process which is orchestrated jointly with the deployment of the CSC-SC slice.

Our experimental evaluation corroborates that CSC establishment incurs an insignificant performance overhead in comparison with the time spent for slice instantiation within the VIM. Tests at larger scale indicate that service discovery and edge PoP selection combined incur delays in the range of tens of msec for a number of edge PoPs as high as 100. With respect to video streaming, we quantified an average of 49% reduction in the time required by an edge cache to receive the video content from the edge cache of a neighbouring network slice, compared to the time needed for the same content to be fetched from the central CDN server. Significant gains are also identified in the context of Industry 4.0. More specifically, the collaboration among robots, empowered by CSC, reduces mission completion time by up to 30%. At the same time, a significant reduction in the data transmission is also observed with CSC.

Overall, we deem CSC as a key enabler for next-generation Service Marketplaces, where network services will not be restricted to the own functionalities; instead, they will be empowered to consume other potentially co-located service elements, offering an unprecedented quality of experience to users. The ability to discover and consume CSC-enabled services across a wide range of (edge) PoPs can be further enhanced by leveraging on AI/ML methods. This is indeed a direction that we plan to pursue in future work.

REFERENCES

[1] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 80–87, May 2017.

[2] F. S. D. Silva, M. O. O. Lemos, A. Medeiros, A. V. Neto, R. Pasquini, D. Moura, C. Rothenberg, L. Mamatas, S. L. Correa, K. V. Cardoso, C. Marcondes, A. ABelem, M. Nascimento, A. Galis, L. Contreras, J. Serrat, and P. Papadimitriou, "NECOS project: Towards lightweight slicing of cloud federated infrastructures," in *Proc. 4th IEEE Conf. Netw. Softwarization Workshops (NetSoft)*, Jun. 2018, pp. 406–414.

[3] P. Valsamas, P. Papadimitriou, I. Sakellariou, S. Petridou, L. Mamatas, S. Clayman, F. Tusa, and A. Galis, "Multi-PoP network slice deployment: A feasibility study," in *Proc. IEEE 8th Int. Conf. Cloud Netw. (CloudNet)*, Nov. 2019, pp. 1–6.

[4] C. Papagianni, P. Papadimitriou, and J. S. Baras, "Rethinking service chain embedding for cellular network slicing," in *Proc. IFIP Netw. Conf. (IFIP Networking) Workshops*, May 2018, pp. 1–9.

[5] *ETSI Network Function Virtualization*. Accessed: Mar. 31, 2022. [Online]. Available: http://www.etsi.org/technologies-clusters/technologies/nfv

[6] *OPNFV*. Accessed: Mar. 31, 31. [Online]. Available: https://www.opnfv.org/

[7] M.-A. Kourtis, M. J. McGrath, G. Gardikis, G. Xilouris, V. Riccobene, P. Papadimitriou, and E. Trouva, "T-NOVA: An open-source MANO stack for NFV infrastructures," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 3, pp. 586–602, Sep. 2017.

[8] D. Spatharakis, I. Dimolitsas, D. Dechouniotis, G. Papathanail, I. Fotoglou, P. Papadimitriou, and S. Papavassiliou, "A scalable edge computing architecture enabling smart offloading for location based services," *Pervas. Mobile Comput.*, vol. 67, Sep. 2020, Art. no. 101217.
[9] G. Papathanail, A. Pentelas, I. Fotoglou, P. Papadimitriou, K. V. Katsaros, V. Theodorou, S. Soursos, D. Spatharakis, I. Dimolitsas, M. Avgeris, D. Dechouniotis, and S. Papavassiliou, "MESON: Optimized cross-slice communication for edge computing," *IEEE Commun. Mag.*, vol. 58, no. 10, pp. 23–28, Oct. 2020.
[10] A. Pentelas and P. Papadimitriou, "Network service embedding for cross-service communication," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2021, pp. 424–430.
[11] D. Dietrich, A. Abujoda, A. Rizk, and P. Papadimitriou, "Multi-provider service chain embedding with nestor," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 1, pp. 91–105, Mar. 2017.
[12] A. Abujoda and P. Papadimitriou, "MIDAS: Middlebox discovery and selection for on-path flow processing," in *Proc. 7th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2015, pp. 1–8.
[13] J. A. Gonzalez, J. Ordonez-Lucena, E. B. Helvik, G. Nencioni, M. Xie, R. D. Lopez, and P. Grønsund, "The isolation concept in the 5G network slicing," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2020, pp. 12–16.
[14] K. V. Katsaros, V. Glykantzis, and G. Petropoulos, "Cache peering in multi-tenant 5G networks," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, May 2017, pp. 1131–1134.
[15] I. Dimolitsas, D. Dechouniotis, V. Theodorou, P. Papadimitriou, and S. Papavassiliou, "A multi-criteria decision making method for network slice edge infrastructure selection," in *Proc. 6th IEEE Conf. Netw. Softwarization (NetSoft)*, Jun. 2020, pp. 1–7.
[16] D. Dechouniotis, I. Dimolitsas, K. Papadakis-Vlachopapadopoulos, and S. Papavassiliou, "Fuzzy multi-criteria based trust management in heterogeneous federated future internet testbeds," *Future Internet*, vol. 10, no. 7, p. 58, Jun. 2018.
[17] N. Herbaut, D. Negru, D. Dietrich, and P. Papadimitriou, "Dynamic deployment and optimization of virtual content delivery networks," *IEEE MultimediaMag.*, vol. 24, no. 3, pp. 28–37, Aug. 2017.
[18] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, and R. Weber, "Pushing CDN-ISP collaboration to the limit," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 2, pp. 34–44, Jul. 2013.
[19] Z. Cao and P. Papadimitriou, "Collaborative content caching in wireless edge with SDN," in *Proc. 1st Workshop Content Caching Del. Wireless Netw. (CCDWN)*, New York, NY, USA, 2016, pp. 1–7.
[20] T. Taleb, I. Afolabi, K. Samdanis, and F. Z. Yousaf, "On multi-domain network slicing orchestration architecture and federated resource control," *IEEE Netw.*, vol. 33, no. 5, pp. 242–252, Sep. 2019.
[21] D. Spatharakis, M. Avgeris, N. Athanasopoulos, D. Dechouniotis, and S. Papavassiliou, "Resource-aware estimation and control for edge robotics: A set-based approach," *IEEE Internet Things J.*, early access, Jan. 7, 2022, doi: 10.1109/JIOT.2022.3141266.
[22] I. Dimolitsas, M. Avgeris, D. Spatharakis, D. Dechouniotis, and S. Papavassiliou, "Enabling industrial network slicing orchestration: A collaborative edge robotics use case," in *Proc. IEEE Int. Medit. Conf. Commun. Netw. (MeditCom)*, Sep. 2021, pp. 215–220.
[23] *Openstack*. Accessed: Mar. 31, 2022. [Online]. Available: https://www.openstack.org/
[24] M. Bitzi, M.-E. Xezonaki, V. Theodorou, I. Dimolitsas, D. Dechouniotis, S. Papavassiliou, G. Papathanail, I. Fotoglou, A. Pentelas, and P. Papadimitriou, "MESON: Optimized cross-slice communication for pervasive virtual CDN services," in *Proc. IEEE Int. Medit. Conf. Commun. Netw. (MeditCom)*, Sep. 2021, pp. 1–2.
[25] L. M. Vaquero, F. Cuadrado, Y. Elkhatib, J. Bernal-Bernabe, S. N. Srirama, and M. F. Zhani, "Research challenges in nextgen service orchestration," *Future Gener. Comput. Syst.*, vol. 90, pp. 20–38, Jan. 2019.
[26] I. Dimolitsas, D. Dechouniotis, S. Papavassiliou, P. Papadimitriou, and V. Theodorou, "Edge cloud selection: The essential step for network service marketplaces," *IEEE Commun. Mag.*, vol. 59, no. 10, pp. 28–33, Oct. 2021.
[27] J. Backman, S. Yrjola, K. Valtanen, and O. Mammela, "Blockchain network slice broker in 5G: Slice leasing in factory of the future use case," in *Proc. Internet Things Bus. Models, Users, Netw.*, Nov. 2017, pp. 1–8.
[28] E. J. Scheid, M. Keller, M. F. Franco, and B. Stiller, "BUNKER: A blockchain-based trusted VNF package repository," in *Proc. Int. Conf. Econ. Grids, Clouds, Syst., Services*. Cham, Switzerland: Springer, 2019, pp. 188–196.
[29] K. Papadakis-Vlachopapadopoulos, I. Dimolitsas, D. Dechouniotis, E. E. Tsiropoulou, I. Roussaki, and S. Papavassiliou, "On blockchain-based cross-service communication and resource orchestration on edge clouds," *Informatics*, vol. 8, no. 1, p. 13, 2021.
[30] G. Carrozzo, M. S. Siddiqui, A. Betzler, J. Bonnet, G. M. Perez, A. Ramos, and T. Subramanya, "AI-driven zero-touch operations, security and trust in multi-operator 5G networks: A conceptual architecture," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2020, pp. 254–258.

DIMITRIOS LASKARATOS received the B.Sc. degree in computer science from the Athens University of Economics and Business, in 2017, and the M.Sc. degree in computing science with specialization in software engineering and distributed systems from the University of Groningen, The Netherlands, in 2019. He is currently a Research Engineer at Intracom SA Telecom Solutions, Greece, in the field of software development and machine learning operations in 5G applications. Previous, he was a Support Engineer in telecommunications at Commsquare Hellas and Software Engineer at Intrasoft International SA. His research interests include software engineering at operating system level, analysis and implementation of large-scale information systems, and databases.

IOANNIS DIMOLITSAS received the Diploma degree from the School of Electrical and Computer Engineering (ECE), National Technical University of Athens (NTUA), Greece, in 2018, where he is currently pursuing the Ph.D. degree with the NETMODE Laboratory, School of ECE. His research interests include the area of multi-criteria optimization, edge and cloud computing, network optimization, and NFV/SDN.

GEORGE PAPATHANAIL (Student Member, IEEE) received the B.Sc. degree in computer engineering from the Technological Institute of Central Macedonia, Serres, in 2011, and the M.Sc. degree in management and information systems from The University of Macedonia,, Greece, in 2017, where he is currently pursuing the Ph.D. degree with the Department of Applied Informatics. His research interests include next-generation networking, NFV, SDN, and network slicing.

MARIA-EVGENIA XEZONAKI received the bachelor's degree in informatics and telecommunications from the National and Kapodistrian University of Athens, in 2015, and the master's degree in computer systems networking, in 2017. She is currently a Staff Research Engineer at Intracom S.A. Telecom Solutions, Greece, working on the fields of 5G networks, network functions virtualization (NFV), and management and orchestration (MANO), with an emphasis on use cases implementation over network infrastructures. She has also worked on the fields of the Internet of Things (IoT), network slicing, and software defined networking (SDN). She has published and presented scientific articles in international academic conferences and journals.

ANGELOS PENTELAS received the B.Sc. degree in mathematics from the Aristotle University of Thessaloniki, Greece, in 2015, and the M.Sc. degree in applied informatics from The University of Macedonia, Greece, in 2019, where he is currently pursuing the Ph.D. degree. His research interests include, among others, the application of optimization and machine-learning methods on NFV orchestration.

VASILEIOS THEODOROU received the Diploma degree from the School of Electrical and Computer Engineering (ECE), National Technical University of Athens (NTUA), Greece, in 2010, the M.A. degree in information technology from the York Universtiy of Toronto, Canada, in 2013, and the joint Dr.-Ing. and Ph.D. degree from Technische Universität Dresden (TUD) and Universitat Politècnica de Catalunya (UPC), in 2017, respectively. He is currently a Senior Research and Innovation Engineer at Intracom Telecom, Greece, active in the areas of multi-access edge computing (MEC), data-driven systems, and network function virtualization (NFV) for 5G networks. He has been a Research Associate with the Database Technologies and Information Management Group, Polytechnic University of Catalonia (UPC), and a Research Assistant with the Adaptive Systems Research Laboratory, York University of Toronto.

DIMITRIOS DECHOUNIOTIS received the Diploma degree in ECE from the University of Patras, in 2004, the M.Sc. degree in control systems and robotics from National Technical University of Athens (NTUA), in 2009, and the Ph.D. degree in ECE from the University of Patras, in 2014. He is currently a Research Associate with the NETMODE Laboratory, NTUA. From 2007 to 2016, he was a non-tenured Lecturer with the EE Department, Technical Educational Institute of Western Greece. His research interests include the area of cloud computing, the Internet of Things, edge computing, trust management, and control theory.

THEODOROS BOZIOS received the degree in computer science from the University of Crete, the Ph.D. degree from the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, and the M.B.A. (Executive M.B.A.) degree from the Athens University of Economics and Business. He is currently a Master Research Engineer at Intracom SA Telecom Solutions. He is responsible for the design and implementation of an innovative platform supporting online marketing and targeted advertising services (Smart Campaign). He is working on data analysis and personalization technologies. He is involved in various internal, national or EU research projects. Before joining Intracom SA, he was a Software Engineer for IBM AS/400 systems. He has been with Intracom S.A. and Intracom SA Telecom Solutions, first as a Telecommunication Software Engineer with the AXE Software Center, and then at Research and Development, where he participated or coordinated Intracom's participation in EU research projects. He was a technical manager, an assistant project manager or a project manager in a number of EU research projects. He was the Coordinator of the research area services delivery systems (SDS) with important projects. His research interests include telecommunication architectures and protocols supporting multimedia, distributed systems, content delivery networks and internet technologies, data analysis, and personalization.

PANAGIOTIS PAPADIMITRIOU (Senior Member, IEEE) received the B.Sc. degree in computer science from the University of Crete, Greece, in 2000, the M.Sc. degree in information technology from the University of Nottingham, U.K., in 2001, and the Ph.D. degree in electrical and computer engineering from the Democritus University of Thrace, Greece, in 2008. He is currently an Assistant Professor with the Department of Applied Informatics, The University of Macedonia, Greece. Before that, he was an Assistant Professor with the Communications Technology Institute, Leibniz Universität Hannover, Germany, and a member of the L3S Research Center, Hanover. He has been a co-PI in several EU-funded (e.g., T-NOVA, CONFINE, NECOS) and nationally-funded projects (e.g., G-Lab VirtuRAMA, MESON). He was a recipient of best paper awards at IFIP WWIC 2012, IFIP WWIC 2016, and the runner-up Poster Award at ACM SIGCOMM 2009. He has co-chaired several international conferences and workshops, such as INFOCOM SWFAN 2016, IFIP WWIC 2016, IEEE CNSM SR+SFC (2018–2019), and IEEE NetSoft S4SI 2020. His research interests include (next-generation) internet architectures, network processing, programmable dataplanes, datacenter networking, and edge computing.

SYMEON PAPAVASSILIOU (Senior Member, IEEE) is currently a Professor with the School of Electrical and Computer Engineering, National Technical University of Athens (NTUA). From 1995 to 1999, he was a Senior Technical Staff Member at AT&T Laboratories, NJ, USA. In August 1999, he joined the ECE Department, New Jersey Institute of Technology, USA, where he was an Associate Professor, until 2004. He has an established record of publications in his field of expertise, with more than 350 technical journals and conference published papers, while he has received several scientific awards and distinctions. His research interests include the areas of optimization and performance evaluation of mobile and distributed systems, wireless networks, and complex systems.