Mobile Edge as Part of the Multi-Cloud Ecosystem: A Performance Study

Thomas Dreibholz*†, Somnath Mazumdar†, Feroz Zahid†, Amir Taherkordi‡ and Ernst Gunnar Gran†§

*Simula Metropolitan Centre for Digital Engineering
Pilestredet 52, N-0167 Oslo, Norway
dreibh@simula.no

†Simula Research Laboratory
Martin Linges vei 17, N-1364 Fornebu, Akershus, Norway
{mazumdar,feroz}@simula.no

‡University of Oslo
Gaustadalléen 23 B, N-0373 Oslo, Norway
amirhost@ifi.uio.no

§Norwegian University of Science and Technology
Teknologivegen 22, N-2815 Gjøvik, Oppland, Norway
ernst.g.gran@ntnu.no

***Abstract*—Cloud computing has revolutionised the development and deployment of applications by running them cost-effectively in remote data centres. With the increasing need for mobility and micro-services, particularly with the emerging 5G mobile broadband networks, there is also a strong demand for mobile edge computing (MEC). It enables applications to run in small cloud systems in close proximity to the user in order to minimise latencies. Both cloud computing and MEC have their own advantages and disadvantages. Combining these two computing paradigms in a unified *multi-cloud* platform has the potential of obtaining the best of both worlds. However, a comprehensive study is needed to evaluate the performance gains and the overheads imposed by this combination to real-world cloud applications. In this paper, we introduce a baseline performance evaluation in order to identify the fallacies and pitfalls of combining multiple cloud systems and MEC into a unified *MEC-multi-cloud* platform. For this purpose, we analyze the basic, application-independent performance metrics of average round-trip time (RTT) and average application payload throughput in a setup consisting of two private and one public cloud systems. This baseline performance analysis confirms the feasibility of MEC-multi-cloud and provides guidelines for designing an autonomic resource provisioning solution in terms of an extension proposed to our existing MELODIC middleware platform for multi-cloud applications.**

I. INTRODUCTION

Over the last few years, cloud computing has principally made a paradigm shift in computing, and the industry has witnessed an accelerated transition from small-scale, closed computing and data storage architectures, to large, open and service-oriented cloud infrastructures [1]. Today, a large number of enterprises and individuals are relying on services offered by clouds to meet their computational and storage demands. Cloud architectures offer significant advantages over traditional cluster computing systems, including flexibility, ease of setup and deployment, high-availability, and on-demand resource allocation—all packed up in an economically attractive pay-as-you-go [2] business model for its users. In general, cloud computing has compelling benefits for applications which are latency-tolerant and do not need to deliver real-time responses to the end-users. However, with the growing need of real-time data analytics and critical event handling by many modern applications, such as in the Internet of Things (IoT), it is evident that the centralised compute and storage model offered by cloud computing is not suitable for such applications, due to high end-to-end latencies [3].

Mobile Edge Computing (MEC) [4] enables a computing and storage infrastructure provisioned closely to the end-users at the edge of the cellular network. MEC enables an environment suitable for latency-sensitive applications, but is often constrained with the limited resource availability at the network edge. Combining MEC with traditional cloud infrastructures can help combat latency challenges imposed by the cloud-centric architecture, while at the same time enables applications to take advantage of the *virtually* unlimited resource capacity of clouds. In this connection, smart cloud offloading is necessary to transfer non-time-critical compute jobs to the clouds for efficient overall application execution, compensating for the limited resource availability on the edge devices.

Cloud offloading is a topic of great research interest in the context of the integration of cloud services and edge devices as well as in mobile cloud computing [5]. Previous research has proposed algorithms for designing cloud offloading systems to improve performance and energy-saving in resource-constrainted mobile systems and network edge devices [6]–[10]. Most of the existing solutions, however, are tightly bounded to a particular cloud platform and lack sufficient support of federation or inter-platform portability. In this way, cloud users are often forced into *vendor lock-in*, due to the use of incompatible protocols and standards enforced by the cloud providers. The lack of flexibility also limits the usability of the offered services. For instance, local legislation could prevent storage of confidential data outside the country; bearing in mind that even market giants have limited geographical presence. In general, as simultaneous aggregation of resources from multiple providers is not available, cloud users are prevented from achieving an optimal cost-performance ratio for their applications.

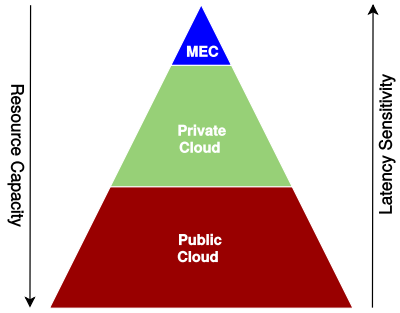


Figure 1: General relationship between resource capacity and latency sensitivity in MEC, a private, and a public cloud.

In the MELODIC project, we are developing a middleware platform that enables cloud applications to run within defined security, cost, and performance boundaries seamlessly on geographically distributed and federated cloud infrastructures. MELODIC thereby realises the potential of heterogeneous cloud environments by transparently taking advantage of distinct characteristics of available private and public clouds. The MELODIC middleware dynamically optimises resource utilization in multi-clouds, considers data locality, and provisions applications conforming to the users' privacy needs and service requirements. MELODIC, however, does not support integration with MEC environments. In general, as shown by a three-level pyramid in Figure 1, MEC, private, and public cloud architectures complement each other and address distinct application demands. By combining the approach taken by MELODIC with MEC, it could be potentially possible to seamlessly move tasks across MEC and multi-cloud infrastructures, based on the resource and latency requirements of the applications.

In this paper, we conduct a comprehensive performance evaluation of a combined multi-cloud and the MEC platform. The study is conducted on three cloud systems: a state-of-the-art MEC-based private

cloud implemented by NORNET CORE, a private OpenStack cloud, and a public cloud. We analyze application-independent performance metrics of average round-trip time (RTT) and average application payload throughput between the three infrastructures using various Internet service providers (ISP) available at the given locations. Based on the presented baseline performance analysis, we confirm the feasibility of a combined *MEC-multi-cloud* platform. Moreover, the evaluation provides guidelines for designing an autonomic resource provisioning solution, in terms of a proposed extension to the MELODIC middleware platform.

II. BACKGROUND

In the following, we provide a brief technical background of cloud computing, multi-clouds, and MEC.

A. Cloud Computing and Multi-Clouds

Broadly speaking, clouds come in three flavors: *public*, *private*, and *hybrid* clouds. The public clouds, as the name suggests, offer infrastructure and services to their customers over the Internet. The basic advantage of using a public cloud is that the organizations need not to invest a large capital expenditure to setup the hardware needed to run their applications and services. The private clouds, on the other hand, are owned and operated by a single organization, which can be thought of as acting as both the cloud provider and the cloud user. Private clouds are used to efficiently utilise available resources shared among different applications and services owned by the same organization or a small group of organizations. A hybrid cloud is a combination of private and public clouds in which resources acquired from public clouds are used to complement the available hardware in the private infrastructure. For example, hybrid cloud setups can dynamically utilise public clouds for application cloud *bursting* in high-load situations.

Cloud federation [11] enables end-users to integrate segregated resources from different cloud providers. The federated clouds offer more freedom to the cloud users, and increase the granularity of choices in the application deployment. We use the term *multi-cloud* to refer to application deployments where multiple cloud platforms are simultaneously used to deploy application components. The term *cross-cloud* is also popular. Some authors differentiate multi-cloud scenarios from cross-clouds and refer to multi-clouds when applications are capable of being deployed on different cloud platforms, but one at a time, whereas cross-cloud deployments involve application components deployed on segregated cloud platforms at the same time. In this paper, we use both terms interchangeably, and always refer to the deployment scenario where application components are deployed across multiple clouds simultaneously.

B. Mobile Edge Computing

Edge computing has begun to be of paramount significance, especially Mobile Edge Computing (MEC) in the mobile cellular networks. The main purpose of mobile edge computing is to address the challenges that are originated from Mobile Cloud Computing (MCC) systems. MEC empowers MCC by deploying cloud resources, such as storage and processing capacity, to the edge within the radio access network [12]. This provides the end-user not only with fast and powerful computing, energy efficiency, and storage capacity, but also with mobility, location and context awareness support. Previously, the technology at the edge of the Internet known as cloudlet technology has been introduced to deploy mobile cloud services. Alternatively, MEC is equipped with better offloading techniques that characterise a network with low latency and high bandwidth. According to ETSI, MEC is defined as [13]: "Mobile Edge Computing provides an IT service environment and cloud computing capabilities at the edge of the mobile network, within the Radio Access Network (RAN) and in close proximity to mobile subscribers." The general architecture of MEC is presented in Figure 2. As shown, different types of mobile devices and sensors (generated by, e.g., IoT, big data and social platforms) are connected to the core network (i.e., mobile Internet) through the edge network (i.e., radio access network) and MEC, and the core network is connected to the private cloud. With the evolution of LTE-based RAN, it is more feasible to deploy MEC, bringing cloud services near to the mobile subscribers.

III. RELATED WORK

Since the main motive for proposing MEC is to provide cloud computing capabilities at the edge of the network, a large body of work has been reported on different distributed computing aspects of integrating the cloud and edge devices. However, the idea of integrating mobile edge devices with multiple cloud platforms is quite novel, and the research community has not paid much attention to it. Nevertheless, in this section, we discuss relevant existing approaches in the area of integration and orchestration of MEC and multi-cloud platforms.

The early attempts in this area have addressed the integration of cloud and core network devices. SHINE [7] focuses on dynamic orchestration of the distributed data center and access to core network segments. The proposed architecture can scale to offer potentially unlimited bandwidth, based on an active remote node (ARN) to interface end-users and the core network. This distributed data center architecture can accelerate content delivery. It also maximises the overall performance in terms of throughput and latency, while minimising total costs and reducing core network traffic. However, its efficiency is still limited to a single-cloud platform.

Another category of existing solutions has focused on enabling a hybrid edge computing model to improve the efficiency of mobile applications. Hybrid Mobile Edge Computing (HMEC) [14] makes use of edge-level computing units, in order to fulfill the needs of interactive mobile applications. It enables an interactive and flexible usage of proximate and distant computing resources through the HMEC framework. It supports application offloading, interoperability between different operation environments, discovery of available computing units and maintaining the user's privacy and data security. ECHO [6] is an orchestration platform for dataflows across distributed edge resources. It features a hybrid dataflow composition that operates on diverse data models and streams, micro-batches and files, and interfaces with native runtime engines like TensorFlow and Storm to execute them. ECHO has the capability to schedule the dataflow on different edge and cloud resources, and also perform dynamic task migration between resources. It manages the application's lifecycle, including container-based deployment and a registry for state management. Both above frameworks are limited to the edge-level hybrid computation model, which is different from the goal of this paper.

FocusStack [15] is built on the idea of location-based situational awareness, implemented over a multi-tier geographic network. It provides an intelligent geo- and context-aware messaging bus that allows the cloud control plane to be scoped based on context that includes the device location, edge device health and capabilities, and user authorization preferences. This solves the problems of inefficient messaging and mixed control that Internet of Things (IoT) device clouds raise for traditional cloud management tools. Although FocusStack reduces management awareness traffic through location-based situational awareness, it is mainly focused on efficient messaging for single-cloud platforms, which is different from our work.

Among recent multi-cloud approaches, Multi-Cloud Application Delivery (MCAD) [8] is a platform to allow application and 5G service providers to specify multi-cloud virtual resource deployment policies, create virtual resources, deploy services in the appropriate cloud(s) and manage them while in operation. It is an extended version of AppFabric [16], which is the platform that finds the optimal locations for virtual resources based on the required cost and performance criteria of an application. The CDN as a Service (CDNaaS) [17] platform supports creating a content distribution network (CDN) slice defined as a set of isolated distributed networks of edge servers over Multi-cloud domains. An edge server in CDNaaS, hosts a single virtual network function (VNF) such as virtual cache, virtual streamer and a CDN-slice-specific coordinator for managing the lifecycle of the slice resources, uploaded videos

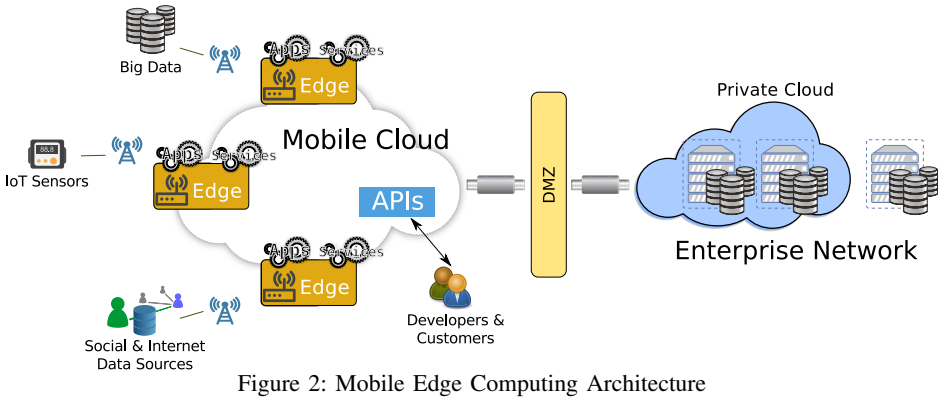


Figure 2: Mobile Edge Computing Architecture

and subscribers. CDNaaS creates a cost-efficient and QoE-aware virtual CDN slice through the optimal placement of VNFs, along with decision on the amount of virtual resources to allocate for each of them. Therefore, the above frameworks are based on the requirement of resource management and placement in multi-cloud and edge integration, while our aim in MELODIC and MEC integration is to identify performance issues and impoverishment opportunities in such integrations.

IV. Software and Infrastructure

In this section, we describe the key software components and infrastructure that together offer seamless integration of MEC with the multi-cloud paradigm. We first describe MELODIC, which is our generic provider-agnostic middleware platform for deployment, configuration, and adaptation of cloud applications on multi-clouds. Then, we present NORNET, a cloudlet infrastructure based on virtual machines (VM). Finally, we show a NORNET-based MEC architecture for our MEC-multi-cloud setup.

A. MELODIC: A Cross-Cloud Middleware Platform

The key objective of the MELODIC project is to provide a middleware platform that enables data-aware application deployments on geographically distributed and federated cloud infrastructures. The MELODIC middleware platform acts as an automatic DevOps solution for cloud applications, covering modeling, deployment, configuration, and autonomic adaptation of the applications in distributed, heterogeneous, and dynamic cross-cloud environments. The platform enables cloud users to take the advantage of distinct characteristics of available private and public clouds by dynamically optimising resource usage, considering data locality, and conforming to the user's privacy needs and Quality-of-Service (QoS) requirements.

Cross-cloud application deployments comprise of resources acquired from multiple administrative domains, ranging from locally deployed private cloud infrastructures to externally managed public cloud offerings [18]. Besides, cloud applications correspond to specific component deployment topologies, and have certain application- and user-specific deployment requirements, such as hardware/OS requirements, security and QoS constraints, allocated cloud budget, as well as scalability policies and rules. The same applies to the data sources. The user data, for instance, may need to adhere to specific location constraints and confidentiality policies in place. MELODIC follows a model-driven engineering (MDE) approach [19], and the cloud applications and corresponding data sources are first modeled so that the aforementioned requirements and constraints can be formally specified, and hence utilised by the deployment reasoning process. The MELODIC modeling interfaces, through the CAMEL modeling language [20], provide a rich set of domain-specific languages which cover different modeling aspects, spanning both the design and the runtime of a cloud application as well as data modeling traits. After the applications have been modeled, the reasoning part of the MELODIC middleware finds the most effective placement of the applications onto cross-cloud resources. Furthermore, to cater for performance unpredictability and dynamicity challenges in the cloud, applications deployed through MELODIC are continuously monitored and adapted, to make sure that the current deployment corresponds to the best possible configuration according to the current cloud resource availability, reliability, performance, user requirements, constraints, and the execution context. In addition to the applications deployment, data management is also performed in an holistic way to cater for unique cross-cloud needs, such as data access latency and storage constraints.

An overview of the MELODIC architecture is given in Figure 3. As shown in the figure, the MELODIC platform is conceptually divided into three main component groups, the MELODIC interfaces to the end-users, the *Upperware*, and the *Executionware*. The MELODIC interfaces to the end-users include tools and interfaces used by the MELODIC users to model their applications and datasets and interact with the MELODIC platform. Applications and data models created through the modeling interfaces, in the form of CAMEL, are given as input to the MELODIC Upperware. The job of the Upperware is to calculate the optimal data placements and application deployments on dynamically acquired cross-cloud resources in accordance with the specified application and data models in CAMEL, as well as in consideration of the current cloud performance, workload situation, and costs. The actual cloud deployments are carried out through the Executionware. The Executionware is capable of managing and orchestrating diverse cloud resources, and enables support of cross-cloud monitoring of the deployed applications. Besides the three main component groups, two auxiliary services are implemented to enable unified and integrated event notification mechanism and to warrant secure operations with the MELODIC platform, respectively.

B. NORNET: A VM-based Cloudlet Infrastructure

The initial motivation of NORNET [21]–[23] was to provide a platform for *realistic* research on the network resilience for critical Internet services. Primarily, it is a large-scale Internet testbed for multi-homed systems (i.e., systems which are simultaneously connected to multiple ISPs). In particular, NORNET utilises virtualisation to allow users to instantiate VMs (containers as well as full VMs) at different sites for running experiments with their own software. That is, it can be seen as a highly distributed cloud.

The NORNET infrastructure consists of two parts: NORNET EDGE and NORNET CORE. NORNET EDGE [23] is the wireless part of NORNET. It consists of single-board computers that run a standard Linux operating system. These nodes are distributed all over Norway. Each node is connected to usually at least two mobile broadband networks, i.e. Long-Term Evolution (LTE) and Universal Mobile Telecommunications System (UMTS). These nodes are powerful enough to run a wide range of distributed applications, such as audio and video streaming. On the other hand, NORNET CORE [22] is the wired-network part of NORNET. It consists of powerful servers, being located at universities and research institutions. Most of these servers are also connected to multiple ISPs, with IPv6 in addition to IPv4

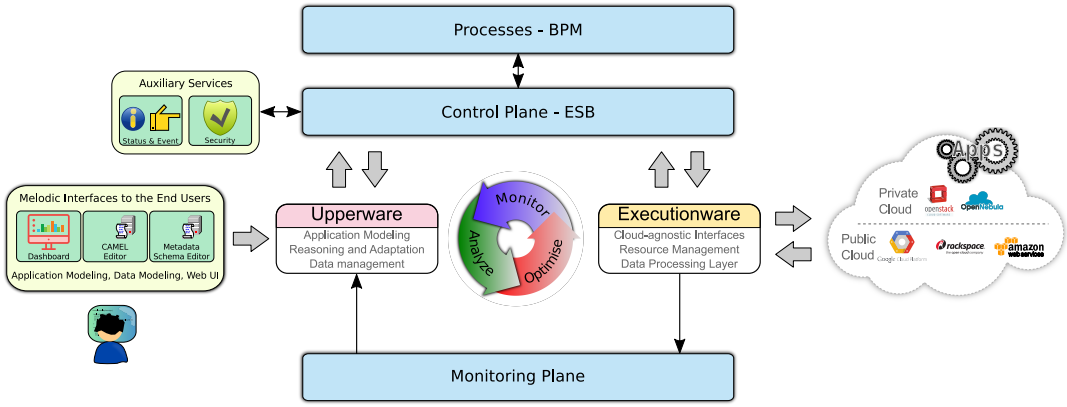


Figure 3: Overview of the Architecture of MELODIC

Table I: NORNET CORE Testbed Sites used for Our Evaluation

Index	Site	ISP 1	ISP 2	ISP 3	ISP 4
1	Simula Research Laboratory	Uninett[6]	Kvantel[6]	Telenor	PowerTech[6]
6	Universitetet i Bergen	Uninett[6]	BKK[6]	–	–
10	Høgskolen i Narvik	Uninett	Broadnet	PowerTech[6]	–
88	Hainan University	CERNET[6]	CnUnicom	–	–

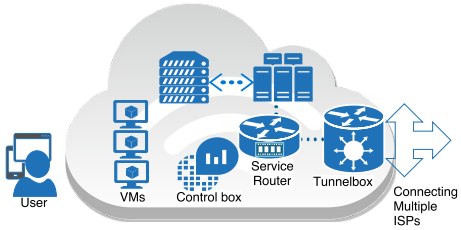


Figure 4: The NorNet-based MEC Architecture

wherever available. In this paper, we focus on NORNET CORE. We therefore introduce it in some more detail here.

Currently, NORNET CORE consists of 22 active sites which are located in seven countries. The testbed connects to sixteen different ISPs supporting both, IPv4 (total of 40 interfaces) and IPv6 (total of 23 interfaces). Table I provides an overview of the sites and ISP connections used in this paper. Entries marked with "[6]" denote IPv6 support in addition to IPv4. A particularly unique feature of NORNET CORE is that the ISPs not only consists of research networks (like UNINETT in Norway or CERNET in China), but there are also consumer-grade connections, like e.g. PowerTech and Telenor Asymmetric Digital Subscriber Lines (ADSL). This allows for experiments where systems experience a "normal" user's quality of service.

In NORNET CORE, the user application can run inside of a full Kernel-based Virtual Machine (KVM). These VMs also support Linux Containers (LXC) inside. The B-Tree File System (BTRFS) is used to avoid file duplication inside the LXCs. This approach helps to build highly resource-efficient containers in a large number. Using the KVM and LXC, the NORNET CORE can already be seen as a larger-scale, widely-distributed Cloudlet setup. At present, there are thousands of containers running in over 110 VMs on more than 60 physical servers.

The architecture of NORNET CORE [21], [22], [24] is illustrated in Figure 4. Each site has a router, which is denoted as tunnelbox. The tunnelbox connects the site to the ISPs. The tunnelbox only has one public IP address per ISP for each IP protocol, i.e, IPv4 and IPv6 (if available), since public IP addresses are a scarce resource. The tunnelbox establishes static IP tunnels (IPv4/IPv6-in-GRE over IPv4, IPv6-in-IPv6 over IPv6) between the sites (i.e., nodes at different sites can directly communicate with each other). For communication with external peers over the Internet (i.e. non-NORNET CORE addresses), network address translation (NAT) and port address translation (PAT) is used for IPv4, while the IPv6 address space is global and routed. For security reasons, all external communication is routed via the Simula site.

The tunnelbox configures [21] one routing table for each locally-connected ISP. Next, IP rules select the right routing table (and therefore the outgoing ISP) for each packet, based on a packet's source address as filter. For instance, if a source address is within the internal address range of ISP 2, the packet will be routed over ISP 2. On the other hand, if it is in the address range of ISP 1, it will go over ISP 1. For communication with other NORNET CORE sites, the destination address specifies the incoming ISP of the remote site. These features in NORNET CORE can be utilised by advanced transport protocols, such as Multi-Path TCP (MPTCP) [25], [26] or Concurrent Multipath Transfer for SCTP [27], [28].

V. EVALUATION

A. Experiment Setup

For evaluating the performance of multi-cloud, we are considering two benchmarks:

- Application Payload Throughput: the payload throughput of a TCP connection, (i.e., the throughput of higher-level application data).
- Round-Trip Time (RTT): the ICMP Echo Request/Echo Reply round-trip time (time from sending the request until reception of the reply) during the TCP measurement.

We have chosen these two basic benchmarks because: i) they are relevant for (almost) all cloud-based applications; and ii) they are independent of the actual kind of higher-level applications. For our measurements, we utilised the NETPERFMETER [29], [30] transport performance metering tool. We use NETPERFMETER to send a saturated TCP flow between two given endpoints. During the TCP measurement, we run `ping` between the same endpoints to record the ICMP Echo Request/Echo Reply RTT [31], [32]. Each measurement has been repeated at least 16 times over a 24-hour business day interval (i.e., covering usual business hours from China to the United States). The results show the average value, as well as the corresponding 95% confidence interval.

The experiments run using the instances of the private NORNET CORE (see Subsection IV-B), a private OpenStack cloud [33] at the Simula Research Laboratory (denoted as SIMULA cloud) with direct connection to the Internet and the NORNET CORE network, as well as the public cloud AMAZON AWS. Both, NORNET CORE as well as the SIMULA cloud, provide IPv4 and IPv6 support. AMAZON AWS, here using Amazon's data centre in Ohio/U.S.A. only offers IPv4. Whenever possible, we performed IPv4 as well as IPv6 experiments for comparison.

B. Evaluation

For our evaluation, we consider three multi-cloud scenarios:

1) NorNet Core and Simula private clouds,
2) Simula private cloud and Amazon AWS public cloud, and
3) NorNet Core private cloud and Amazon AWS public cloud.

C. NorNet Core *and* Simula *Private Clouds*

First, we examined the performance of the NorNet Core and Simula private clouds. For NorNet Core, we have selected 3 interesting sites, due to their locations and ISP connections (see also Table I):

- Hainan University: A site in Haikou, Hainan/China, being connected to two different ISPs. The research network ISP CERNET in China also provides IPv6 support, allowing for intercontinental IPv6 communication. China Unicom (CnUnicom) provides a business-grade fibre connection (IPv4 only).
- Universitetet i Bergen: A site in Bergen, Hordaland/Norway, being connected to the research network provider UNINETT, as well as to the business-grade ISP BKK. Both ISPs offer IPv6 in addition to IPv4.
- Høgskolen i Narvik: A site in Narvik, Nordland/Norway, in the far north of Norway. In addition to connecting to UNINETT (here: IPv4 only), it is also connected to two consumer-grade ADSL links from PowerTech (IPv4 and IPv6) and Broadnet (IPv4 only).

The Simula private OpenStack setup is hosted at the Simula Research Laboratory in Fornebu, Akershus/Norway. It is connected to the research network ISP UNINETT, business-grade ISP Kvantel, as well as to two consumer-grade ADSL connections from PowerTech and Telenor. Except for the Telenor ADSL connection, all ISPs offer IPv6 in addition to IPv4. Instances in the Simula cloud can directly communicate with endpoints in the NorNet Core infrastructure [22], [33], i.e. no NAT/PAT for IPv4 is necessary here.

1) NorNet Core *to* Simula*:* Figures 5, 6 and 7 present the results for sending data from NorNet Core instances at Hainan University, Universitetet i Bergen and Høgskolen i Narvik to the Simula cloud at the Simula Research Laboratory. Here, Subfigure (a) presents the average payload throughput, while Subfigure (b) shows the average RTT. Obviously, there are significant differences with the choice of ISP relations and IP protocol versions.

Particularly, for Hainan University to Simula (Figure 5):

- IPv6 (via CERNET only) has a better throughput than IPv4, regardless of the chosen destination ISP at Simula (Kvantel, PowerTech, UNINETT). Note, that Telenor does not offer IPv6. For example CERNET to UNINETT achieves 20.4 Mbit/s via IPv6, instead of only 1.8 Mbit/s via IPv4. The reason here is most likely applied bandwidth limitation at Hainan University, which is applied for IPv4 but not for IPv6.
- As expected, the ADSL providers generally have a low throughput (16/1 Mbit/s subscriptions). However, seeing 1.8 Mbit/s for data transmission from CERNET/IPv4 to all 4 ISPs confirms the assumption of a bandwidth limitation. For CERNET/IPv6, the values differ (e.g. 6.5 Mbit/s to Kvantel vs. 20.4 Mbit/s to UNINETT vs. 4.2 Mbit/s to PowerTech).
- The RTT also differs significantly, as expected from the throughput results. Note particularly the difference for CnUnicom/IPv4 to PowerTech and Telenor: 621.5 ms (PowerTech) vs. 331.4 ms (Telenor), although the access technology is ADSL in both cases. We explain this in more detail below.

As as result, it is quite obvious – and expected – that the performance highly differs between the choice of ISPs and IP protocol versions in an inter-continental cloud setup.

The bee-line connection between Universitetet i Bergen and Simula is just around 500 km. As Figure 6 shows, the results are more stable compared to the inter-continental setup. However:

- From Bergen, both ISPs reach ca. 85 Mbit/s to UNINETT at Simula. The reason here is a 100 Mbit/s Fast Ethernet router at Simula, which is the bottleneck. BKK in Bergen is furthermore just a 100/100 Mbit/s subscription, i.e. just around 90 Mbit/s are achieved from BKK to Kvantel, while it is 236.8 Mbit/s from UNINETT to Kvantel.
- However, the performance to Kvantel significantly differs between IPv4 and IPv6: 236.8 Mbit/s (IPv4) vs. just 14.1 Mbit/s (IPv6). Note also the significant RTT difference in this case: 9.5 ms (IPv4) vs. 38.9 ms (IPv6). This strongly indicates a significant detour for the IPv6 packets. Although IPv6 [34] is almost 20 years old, its deployment still significantly differs from IPv4!
- Again, the RTTs of the two ADSL ISPs significantly differ again: ca. 390 ms (PowerTech) vs. ca. 145 ms (Telenor).

That is, the performance via different ISPs and IP protocol versions can also significantly vary when distances are short.

Finally, having a look at the performance between Høgskolen i Narvik and Simula examines the differences for ADSL connections in some more detail (Figure 7):

- UNINETT is the only high-speed ISP at Høgskolen i Narvik. Consequently, the throughput performance is very good (204.9 Mbit/s to Kvantel, 87 Mbit/s to UNINETT – due to the 100 Mbit/s router at Simula) at low RTT (22.7 ms vs. 37.3 ms).
- The ADSL throughput is similar for all ADSL combinations (up to 0.8 Mbit/s of payload throughput), due to the subscription of 16/1 Mbit/s, leading to 1 Mbit/s in the upstream.
- However, the ADSL RTTs very significantly: ca. 77 ms for PowerTech-PowerTech or 82.2 ms for PowerTech-Telenor vs. values exceeding 600 ms for all combinations from Broadnet. Here, load in combination with buffer bloat [35] is the reason. Buffer bloat is particularly a result of the ISP's ADSL modem configuration.

That is, consumers (i.e. the "normal" Internet users) at similar locations may have very different experiences of the performance of cloud systems, depending on their ISPs.

2) Simula *to* NorNet Core*:* The reverse direction, i.e. Simula cloud to the 3 NorNet Core sites, is shown in Figures 8 (Hainan University), 9 (Universitetet i Bergen) and 10 (Høgskolen i Narvik). As expected, the observations from the other direction (NorNet Core to Simula, see Subsubsection V-C1) also apply here. But particularly notable are also:

- Telenor (IPv4 only) at Simula is now at the egress (i.e. in upstream direction). With a 16/1 Mbit/s subscription (i.e. just 1 Mbit/s in the upstream), there is a significant buffer bloat observable: almost 1500 ms (i.e. 1.5 seconds!) in the short-distance case between Simula and Bergen, and even more than 1600 ms in the others. This is even worse than the ca. 600 ms for Broadnet in the other direction (see Subsubsection V-C1).
- PowerTech, the other ADSL ISP, has significantly lower RTTs. That is, although the subscriptions are similar, there is a significant performance difference.

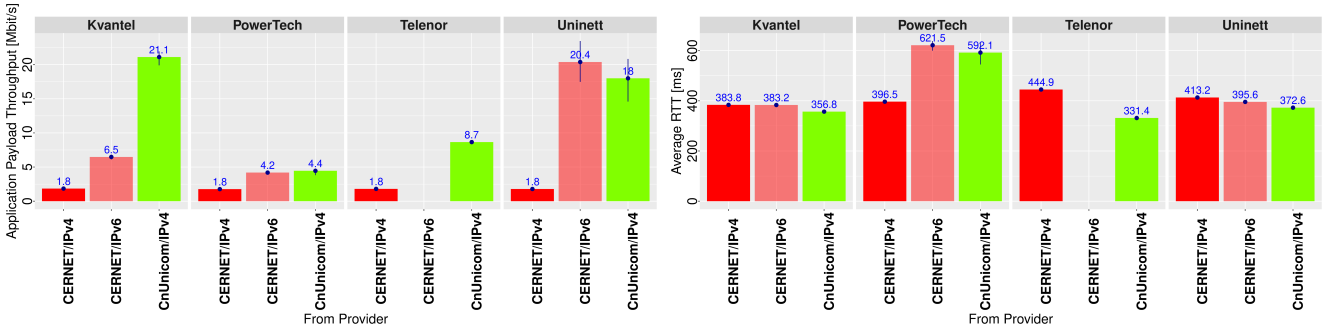
3) Summary: With a diversity of different ISP connections, and IPv4 as well as IPv6, the performances of communications between two endpoints may vary significantly. When deploying and provisioning MEC systems, this has to be taken into account, in order to achieve the best-possible user experience.

Table II: Simula and Amazon AWS

Source to Destination (location)	Source to Destination (ISP/IPv4)	Payload throughput (Mbit/s)	Avg. RTT (ms)
Simula→Amazon AWS	UNINETT→Amazon AWS	85.52±15.44	120.09±0.72
Amazon AWS→Simula	Amazon→UNINETT	36.2±8.29	118.49±0.54

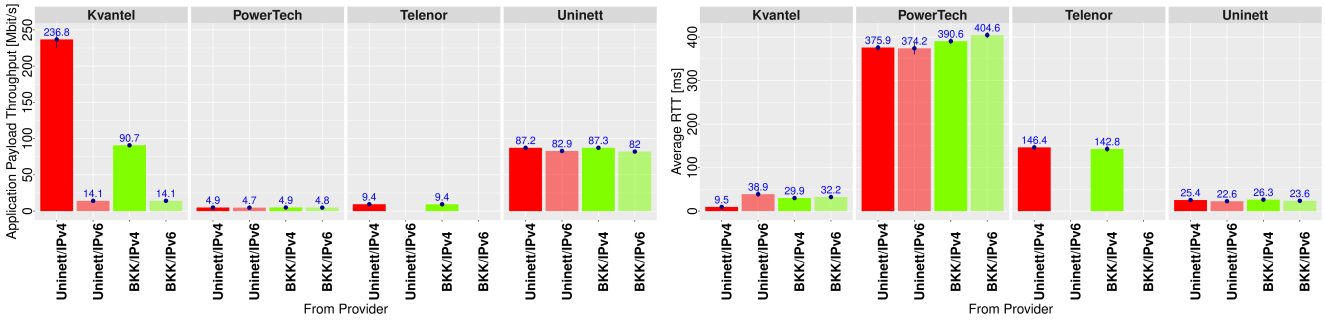
D. Simula *Private and* Amazon AWS *Public Cloud*

In the next scenario, we examine the performance between the private Simula cloud and the public Amazon AWS cloud, i.e. between the private OpenStack setup at the Simula Research Laboratory in Fornebu, Akershus/Norway and the Amazon AWS data centre in Ohio/U.S.A.. Table II presents the results with average value and 95% confidence interval for both directions.



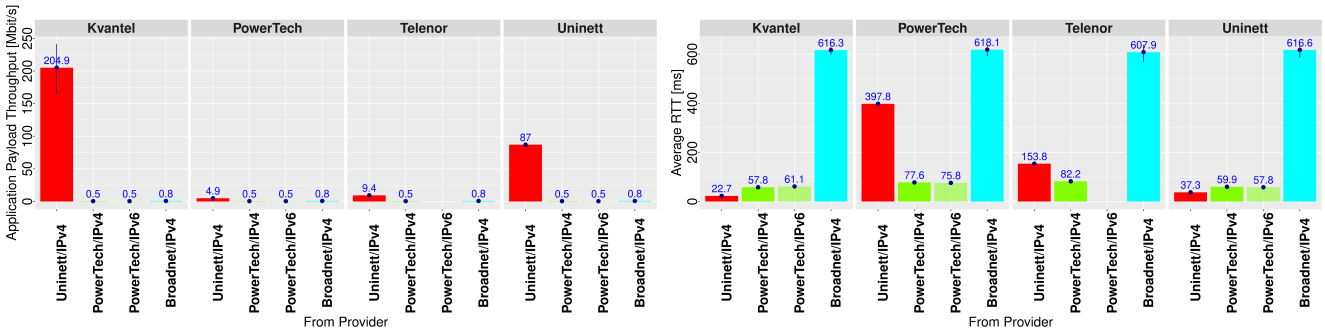
(a) Payload Throughput (b) Round-Trip Time

Figure 5: NorNet Core to Simula: Hainan University to Simula Research Laboratory



(a) Payload Throughput (b) Round-Trip Time

Figure 6: NorNet Core to Simula: Universitetet i Bergen to Simula Research Laboratory



(a) Payload Throughput (b) Round-Trip Time

Figure 7: NorNet Core to Simula: Høgskolen i Narvik to Simula Research Laboratory

In this setup, there is no room for variation: Amazon AWS in Ohio only provides IPv4. The Simula private cloud is externally (i.e. outside of the NorNet Core infrastructure) only communicating via UNINETT. This leaves only IPv4 connectivity between UNINETT and Amazon. Two things are notable:

- Simula to Amazon AWS achieves an average payload throughput of 85.52 Mbit/s, while the reverse direction achieves a throughput of 36.2 Mbit/s, both with some variance.
- On the other hand, the average RTT in both directions is quite stable at around 120 ms. The bandwidth limits are probably the result of some bandwidth limitation at Amazon, since congestion in the Internet would likely had caused a higher RTT variance.

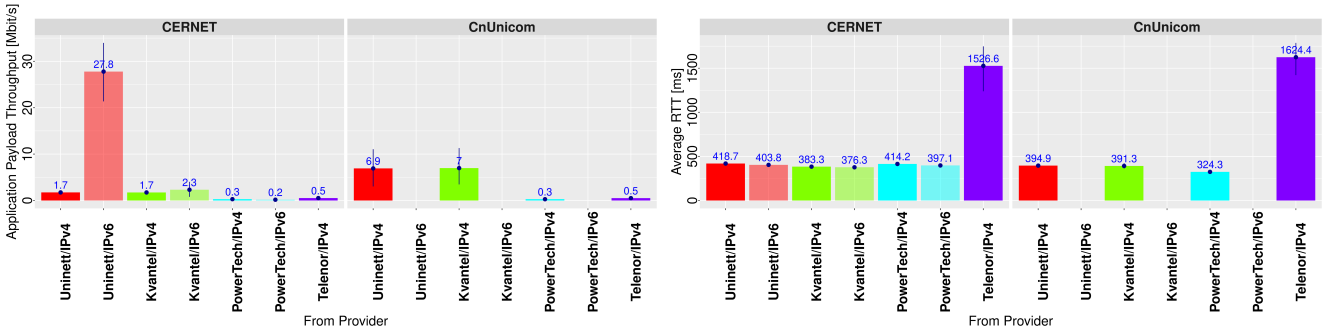
In summary, when using a public cloud, an application gets the performance provided by the given resources (location, ISP, IP protocol), i.e. "you get what you pay for" (here: using Amazon AWS Free Tier for free). There is not much possibility to otherwise influence the performance. On the other hand, the costs of the resources are minimised.

Table III: To Amazon AWS

NorNet Core to Amazon AWS (Ohio)		
Source to Destination	Payload throughput (Mbit/s)	Avg. RTT (ms)
Simula → Amazon Ohio	71.3	153.2
Universitetet i Bergen → Amazon Ohio	56.6	160.9
Høgskolen i Narvik → Amazon Ohio	55.4	174.7
HainanUniversity → Amazon Ohio	1.7	557.1

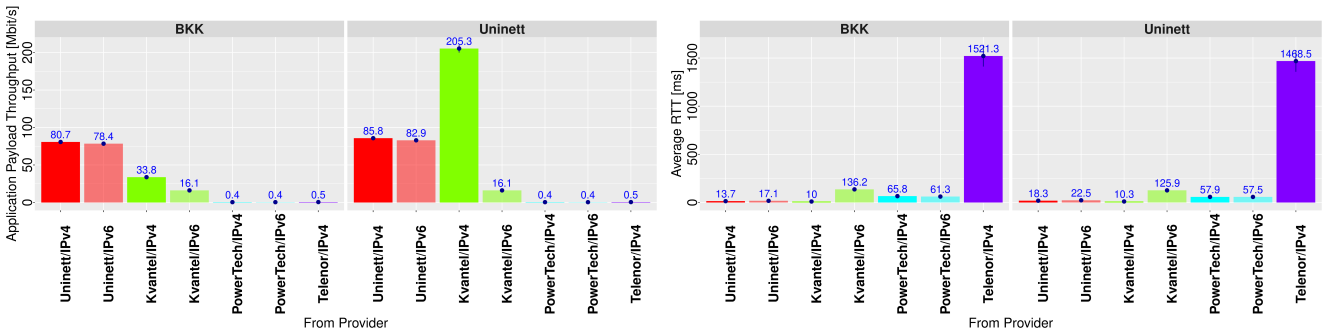
E. NorNet Core *Private and* Amazon AWS *Public Cloud*

In the last scenario, we analyse the performance between the private NorNet Core cloud and the public Amazon AWS cloud at the Amazon data centre in Ohio/U.S.A.. For NorNet Core, we again selected the 3 sites from Subsection V-C (i.e. Hainan University,



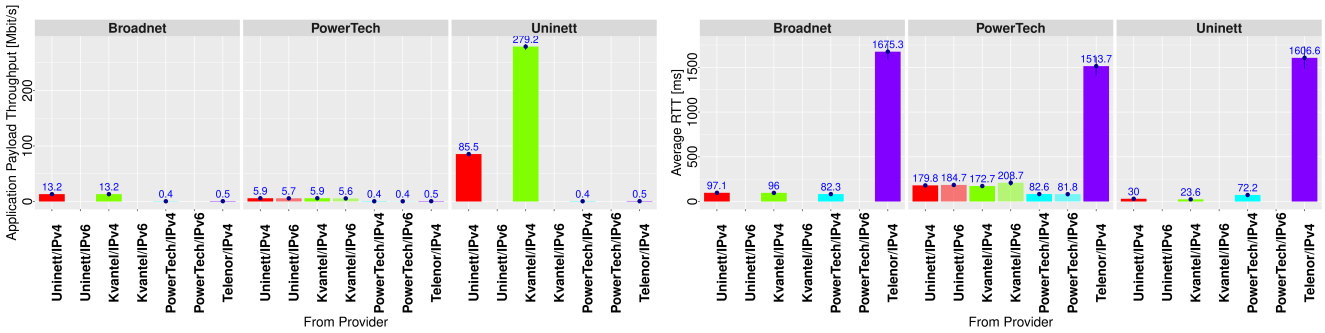
(a) Payload Throughput (b) Round-Trip Time

Figure 8: SIMULA to NORNET: Simula Research Laboratory to Hainan University



(a) Payload Throughput (b) Round-Trip Time

Figure 9: SIMULA to NORNET: Simula Research Laboratory to Universitetet i Bergen



(a) Payload Throughput (b) Round-Trip Time

Figure 10: SIMULA to NORNET: Simula Research Laboratory to Høgskolen i Narvik

Table IV: AMAZON AWS to NORNET CORE

AMAZON AWS (Ohio) to NORNET CORE		
Source to Destination	**Payload throughput (Mbit/s)**	**Avg. RTT (ms)**
Amazon Ohio → Simula	2.5	146.4
Amazon Ohio → Universitetet i Bergen	2.3	160.7
Amazon Ohio → Høgskolen i Narvik	2.1	168.3
Amazon Ohio → Hainan University	0.7	542.9

Universitetet i Bergen, Høgskolen i Narvik). Furthermore, we add the Simula Research Laboratory as fourth site. The Simula site connects the NORNET CORE infrastructure to the outside world, i.e. external connections between NORNET CORE and the public Internet go via Simula's UNINETT network, via a 100 Mbit/s Fast Ethernet router. Other sites use their primary ISP (CERNET at Hainan University, UNINETT at Universitetet i Bergen and Høgskolen i Narvik) to route external traffic to Simula's UNINETT connection first.

As expected, the performance results provided in Table III (NORNET CORE to AMAZON AWS) and Table IV) (AMAZON AWS to NORNET CORE) correspond to the results from Subsection V-C and Subsection V-D: all traffic needs to be routed via Simula. This leads to additional delay (between AMAZON AWS and Simula, and between Simula and the actual site). Therefore, the delay to/from Simula is the lowest; it is slightly higher for the other Norwegian sites at Universitetet i Bergen and Høgskolen i Narvik, and significantly higher when using another inter-continental connection to Hainan University in China. Also, the throughput results reflect this setup. However, it is notable to see a significant throughput difference between sending from NORNET CORE sites to AMAZON AWS and the reverse direction: e.g. 71.3 Mbit/s vs. 2.5 Mbit/s. The reason is a bottleneck at Simula, caused by the necessary NAT/PAT from external addresses/ports to internal NORNET CORE addresses/ports. Since only IPv4 is available for Amazon's VM, there is no possibility

to avoid NAT/PAT by using IPv6, i.e. this bottleneck is unavoidable.

In summary, when connecting different clouds to a multi-cloud, it is also necessary to take the inter-connectivity details of the clouds into consideration.

VI. Conclusion and Future Work

In this paper, we advocate for an ecosystem consisting of MEC and Cloud, which can be used seamlessly by the user. MEC platforms are smaller in capacity, while offering better latency support. On the other side, Cloud offers a huge amount of scalable computing resources at low costs. By combining MEC and Cloud, the advantages of both can be combined. The goal of the new MELODIC middleware is to provide this combination, seamlessly to the user and easy to use for the application developers and providers. As a groundwork for ongoing work towards the MELODIC middleware, this paper presents baseline performance results for the combination of three different cloud systems: NORNET CORE and a private OpenStack setup at Simula, as well as the public cloud AMAZON AWS, for the basic metrics of network bandwidth and latency. We particularly identified the fallacies and pitfalls of "just combining two cloud setups", in order to properly design and handle combined systems.

As part of our ongoing and future work, we are extending the MELODIC middleware platform to include integration with the MEC environments. By using the baseline insights obtained from the results presented in this paper, we are currently designing and testing algorithms to realise seamless resource provisioning, application deployment, and dynamic adaptation on MEC-multi-cloud infrastructures. Particularly, we believe that MELODIC will be a highly useful middleware system to deploy MEC/Cloud applications in future 5G mobile network setups, where cost-effectiveness, performance requirements, user mobility and highly dynamic setups are major challenges.

Acknowledgement

This work is partially supported by the European Union H2020 program through the MELODIC project (grant agreement number 731664) and by the Research Council of Norway (project number 208798/F50).

References

[1] K. E. Kushida, J. Murray, and J. Zysman, "Cloud Computing: From Scarcity to Abundance," *Journal of Industry, Competition and Trade*, vol. 15, no. 1, pp. 5–19, Feb. 2015.

[2] M. A. Rappa, "The Utility Business Model and the Future of Computing Services," *IBM Systems Journal*, vol. 43, no. 1, pp. 32–42, Jan. 2004.

[3] A. Botta, W. D. Donato, V. Persico, and A. Pescapé, "On the Integration of Cloud Computing and Internet of Things," in *Proceedings of the International Conference on Future Internet of Things and Cloud (FiCloud)*, Aug. 2014, pp. 23–30.

[4] M. T. Beck, M. Werner, S. Feld, and S. Schimper, "Mobile Edge Computing: A Taxonomy," in *Proceedings of the 6th International Conference on Advances in Future Internet (AFIN)*, 2014, pp. 48–55.

[5] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, Oct. 2013.

[6] P. Ravindra, A. Khochare, S. P. Reddy, S. Sharma, P. Varshney, and Y. R. Simmhan, "ECHO: An Adaptive Orchestration Platform for Hybrid Dataflows across Cloud and Edge," in *Proceedings of the 15th International Conference on Service-Oriented Computing (ICSOC)*, M. Maximilien, A. Vallecillo, J. Wang, and M. Oriol, Eds. Málaga/Spain: Springer, 2017, pp. 395–410.

[7] L. Velasco, L. M. Contreras, G. Ferraris, A. Stavdas, F. Cugini, M. Wiegand, and J. P. Fernández-Palacios, "A Service-Oriented Hybrid Access Network and Clouds Architecture," *IEEE Communications Magazine*, vol. 53, no. 4, pp. 159–165, Apr. 2015.

[8] D. Bhamare, A. Erbad, R. Jain, and M. Samaka, "Automated Service Delivery Platform for C-RANs," in *Proceedings of the 2nd International Conference on Fog and Mobile Edge Computing (FMEC)*, May 2017, pp. 219–224.

[9] H. hua Chu, H. Song, C. Wong, S. Kurakake, and M. Katagiri, "Roam, A Seamless Application Framework," *Journal of Systems and Software*, vol. 69, no. 3, pp. 209–226, Jan. 2004.

[10] C. Wang and Z. Li, "Parametric Analysis for Adaptive Computation Offloading," in *Proceedings of the ACM Conference on Programming Language Design and Implementation (SIGPLAN)*, vol. 39, no. 6, Jun. 2004, pp. 119–130.

[11] T. Kurze, M. Klems, D. Bermbach, A. Lenk, S. Tai, and M. Kunze, "Cloud Federation," *Cloud Computing*, vol. 2011, pp. 32–38, 2011.

[12] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, Sep. 2018.

[13] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile Edge Computing – A Key Technology Towards 5G," *ETSI White Paper*, vol. 11, Sep. 2015.

[14] A. Reiter, B. Prünster, and T. Zefferer, "Hybrid Mobile Edge Computing: Unleashing the Full Potential of Edge Computing in Mobile Device Use Cases," in *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, ser. CCGrid '17, Madrid/Spain, 2017, pp. 935–944.

[15] B. Amento, B. Balasubramanian, R. J. Hall, K. Joshi, G. Jung, and K. H. Purdy, "FocusStack: Orchestrating Edge Clouds Using Location-Based Focus of Attention," in *Proceedings of the IEEE/ACM Symposium on Edge Computing (SEC)*, Oct. 2016, pp. 179–191.

[16] S. Paul, R. Jain, M. Samaka, and J. Pan, "Application Delivery in Multi-Cloud Environments using Software-Defined Networking," *Computer Networks, Special Issue on Communications and Networking in the Cloud*, vol. 68, pp. 166–186, Aug. 2014.

[17] I. Benkacem, T. Taleb, M. Bagaa, and H. Flinck, "Optimal VNFs placement in CDN Slicing over Multi-Cloud Environment," *Journal on Selected Areas in Communications*, 2018.

[18] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "How to Enhance Cloud Architectures to Enable Cross-Federation," in *Proceedings of the 3rd IEEE International Conference on Cloud Computing (CLOUD)*. IEEE, 2010, pp. 337–345.

[19] D. C. Schmidt, "Model-Driven Engineering," *IEEE Computer Journal*, vol. 39, no. 2, p. 25, Feb. 2006.

[20] A. Rossini, "Cloud Application Modelling and Execution Language (CAMEL) and the PaaSage Workflow," in *Advances in Service-Oriented and Cloud Computing – Workshops of ESOCC*, vol. 567, Taormina/Italy, Sep. 2015, pp. 437–439.

[21] T. Dreibholz, "NorNet – Building an Inter-Continental Internet Testbed based on Open Source Software," in *Proceedings of the LinuxCon Europe*, Berlin/Germany, Oct. 2016.

[22] E. G. Gran, T. Dreibholz, and A. Kvalbein, "NorNet Core – A Multi-Homed Research Testbed," *Computer Networks, Special Issue on Future Internet Testbeds*, vol. 61, pp. 75–87, Mar. 2014.

[23] A. Kvalbein, D. Baltrūnas, K. R. Evensen, J. Xiang, A. M. Elmokashfi, and S. Ferlin, "The NorNet Edge Platform for Mobile Broadband Measurements," *Computer Networks, Special Issue on Future Internet Testbeds*, vol. 61, pp. 88–101, Mar. 2014.

[24] T. Dreibholz, "NorNet – The Internet Testbed for Multi-Homed Systems," in *Proceedings of the Multi-Service Networks Conference (MSN, Coseners)*, Abingdon, Oxfordshire/United Kingdom, Jul. 2016.

[25] F. Fu, X. Zhou, T. Dreibholz, K. Wang, F. Zhou, and Q. Gan, "Performance Comparison of Congestion Control Strategies for Multi-Path TCP in the NorNet Testbed," in *Proceedings of the 4th IEEE/CIC International Conference on Communications in China (ICCC)*, Shenzhen, Guangdong/People's Republic of China, Nov. 2015, pp. 607–612.

[26] T. Dreibholz, X. Zhou, and F. Fu, "Multi-Path TCP in Real-World Setups – An Evaluation in the NorNet Core Testbed," in *5th International Workshop on Protocols and Applications with Multi-Homing Support (PAMS)*, Gwangju/South Korea, Mar. 2015, pp. 617–622.

[27] K. V. Yedugundla, S. Ferlin, T. Dreibholz, Özgü Alay, N. Kuhn, P. Hurtig, and A. Brunström, "Is Multi-Path Transport Suitable for Latency Sensitive Traffic?" *Computer Networks*, vol. 105, pp. 1–21, Aug. 2016.

[28] T. Dreibholz, "Evaluation and Optimisation of Multi-Path Transport using the Stream Control Transmission Protocol," Habilitation Treatise, University of Duisburg-Essen, Faculty of Economics, Institute for Computer Science and Business Information Systems, Mar. 2012.

[29] T. Dreibholz, "NetPerfMeter: A Network Performance Metering Tool," *Multipath TCP Blog*, Sep. 2015.

[30] T. Dreibholz, M. Becke, H. Adhari, and E. P. Rathgeb, "Evaluation of A New Multipath Congestion Control Scheme using the NetPerfMeter Tool-Chain," in *Proceedings of the 19th IEEE International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Hvar, Dalmacija/Croatia, Sep. 2011, pp. 1–6.

[31] A. Conta, S. E. Deering, and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," IETF, Standards Track RFC 4443, Mar. 2006.

[32] J. B. Postel, "Internet Control Message Protocol," IETF, Standards Track RFC 792, Sep. 1981.

[33] T. Dreibholz, "Testing Applications with the NorNet Infrastructure," in *Proceedings of the MELODIC Plenary Meeting*, Warszawa, Masovia/Poland, Sep. 2017.

[34] S. E. Deering and R. M. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," IETF, Standards Track RFC 2460, Dec. 1998.

[35] V. Cerf, V. Jacobson, N. Weaver, and J. Gettys, "BufferBloat: What's Wrong with the Internet?" *ACM Queue*, vol. 9, no. 12, pp. 10–20, Dec. 2011.