ACCORDION

Adaptive edge/cloud compute and network continuum over a heterogeneous sparse edge infrastructure to support nextgen applications

Deliverable D2.2

State of the art report (I)

ACCORDION receives funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 871793

DOCUMENT INFORMATION

PROJECT	
PROJECT ACRONYM	ACCORDION
PROJECT FULL NAME	Adaptive edge/cloud compute and network continuum over a heterogeneous sparse edge infrastructure to support nextgen applications
STARTING DATE	01/01/2020 (36 months)
ENDING DATE	31/12/2022
PROJECT WEBSITE	http://www.accordion-project.eu/
TOPIC	ICT-15-2019-2020 Cloud Computing
GRANT AGREEMENT N.	871793
COORDINATOR	CNR
DELIVERABLE INFORMATION	
WORKPACKAGE N. | TITLE	WP2: Requirements & System Design
WORKPACKAGE LEADER	HUA
WORKPACKAGE PARTICIPANTS	All
DELIVERABLE N. | TITLE	D2.2: State of the art report (I)
CONTRIBUTOR(S)	Ioannis Korontanis (HUA), Emanuele Carlini (CNR), Hanna Kavalionak (CNR), Patrizio Dazzi (CNR), Vangelis Psomakelis (ICCS), Lorenzo Blasi (HPE), Zinelaabidine Nadir (AALTO), John Violos (ICCS), Andrea Toro (HPE), Marco Russo (HPE), Eduard Marin Fabregas (TID), George Vasios (HUA), Konstantinos Tserpes (HUA), Saman Zadtootaghaj (TUB), Bartlomiej Lipa (BSOFT), Maria Pateraki (OVR), Marco Di Girolamo (HPE)
EDITOR(S)	Lorenzo Blasi (HPE)
REVIEWER(S)	Maria Pateraki (OVR)
CONTRACTUAL DELIVERY DATE	10/2020
ACTUAL DELIVERY DATE	30/10/2020
VERSION	V1.0
TYPE	Report
DISSEMINATION LEVEL	Public
TOTAL N. PAGES	82
KEYWORDS	Edge, resource pooling, orchestration, machine learning, models

EXECUTIVE SUMMARY

The present document is the result of the collaborative effort of all ACCORDION partners participating to Task 2.2. The document offers a review of the state of the art for a series of topics strictly related to the work performed in the ACCORDION project. There is actually a strict correlation between the topics analyzed in this document and the Tasks that are part of the three research Work Packages of ACCORDION (WP3, WP4, and WP5).

The main part of this document is section 2, in which all the state of the art analysis results have been reported. Section 2 has a subsection for each of the topics researched in the project, which includes: a description of the objectives, a list of outcomes expected from the research work, and an analysis of the state of the art.

Section 2.1 (Resource monitoring & characterization) reports on monitoring, characterization and classification of Edge resources, identifying Prometheus, TOSCA and the automatic creation of taxonomies, respectively, as the best solutions for each of the three fields.

Section 2.2 (Resource indexing & discovery) focuses on discussing solutions and data structures for organizing data in Resource Discovery Services.

Section 2.3 (Edge storage, availability, reliability and performance) presents the advantages and disadvantages of both block and object storage, and then discusses some solutions, identifying OpenStack and MinIO as the most promising ones, even if not completely suitable. Some open research issues are also summarized.

Section 2.4 (Pooling Edge resources), after listing some orchestration challenges typical of Edge computing and the techniques to cope with them, reports on several solutions to be considered as possible baselines for the ACCORDION Minicloud.

Section 2.5 (AI-based network orchestration) first lists the main machine learning techniques, then explores both Federated Learning techniques and further evolutions such as Meta-Learning Framework and Multi-Agent Reinforcement Learning.

Section 2.6 (Resilience policies & mechanisms over heterogeneous edge resources) starts by discriminating between reactive and proactive protection strategies and describing some of them. Then other Fault Tolerance approaches are explored both reported in the literature and adopted in common distributed computing frameworks (Openstack, Cloudstack, Kubernetes, Openshift, and Mesos). Finally techniques for movement behaviour and resource utilization prediction are analysed, with a particular focus on the the LSTM model for Neural Networks. The conclusion is that the most promising solution to efficiently adapt the deep learning topologies for the fault tolerance needs is the hyper parameter optimization approach.

Section 2.7 (Techniques for secure Edge application development & deployment) offers an analysis of the most common types of security attacks (Distributed Denial-of-Service, Malware Injection, and Authentication-based attacks) and their related countermeasures, along with some threat modelling methods, while DevSecOps methods and tools are also described.

Section 2.8 (Privacy preserving mechanisms) starts by analysing Machine Learning techniques with a focus on privacy preserving ones, and then lists a number of works analysing how cookie synchronization techniques adopted for web advertising can expose users to privacy leaks.

Section 2.9 (Application model for automatic deployment / migration of components) looks for application description models suitable for ACCORDION, i.e. with a machine-processable syntax, able to represent resource capacity requirements, containerization, and recovery policies. Three available solutions, TOSCA, Juju charms and CAMP, are compared along with the projects that are using them. Furthermore, tools supporting the three above solutions are described, and works researching the interoperability among the solutions are also analysed.

Section 2.10 (Modelling and assessing QoE for NextGen applications) reports on different types of objective models that can be used to estimate the Quality of Experience (QoE) perceived by users of multimedia applications, and about the latest ITU-T Recommendations on QoE models and methodologies that can be applied to Next Generation Applications. For the ACCORDION project it has been decided to follow the standardized approach to build models for QoE assessment of ACCORDION applications.

Section 2.11 (DevOps tools to automate Edge applications' deployment) sets the context and reports the starting points for the evaluation of Continuous Integration and Continuous Deployment tools. The identified state-of-the art solutions are Jenkins for the CI/CD pipeline and Kubernetes as the runtime deployment environment.

Section 2.12 (Collaborative VR), starting from the general requirements for Virtual Reality applications, reports considerations about the still limited power of the available HMDs and discusses the trade-offs conditioning the possibility to offload computation from the end devices to the Edge.

Finally Section 2.13 (Resource federation models) describes the main features of the federation model proposed by the H2020 5GeX project and lists the additional constraints and issues raised by an Edge providers' federation, which have to be further investigated.

Not all project’s research Tasks have a related section in this document about their main topics, yet. Monitoring the State of the Art is an ongoing activity in ACCORDION, and the next version of this document foreseen in M22 will improve the State of the Art analysis by adding further details, covering more topics and reporting on possible new approaches that appeared in the meantime.

DISCLAIMER

ACCORDION (871793) is a H2020 ICT project funded by the European Commission.
ACCORDION establishes an opportunistic approach in bringing together edge resource/infrastructures (public clouds, on-premise infrastructures, telco resources, even end-devices) in pools defined in terms of latency, that can support NextGen application requirements. To mitigate the expectation that these pools will be "sparse", providing low availability guarantees, ACCORDION will intelligently orchestrate the compute & network continuum formed between edge and public clouds, using the latter as a capacitor. Deployment decisions will be taken also based on privacy, security, cost, time and resource type criteria.
This document contains information on ACCORDION core activities. Any reference to content in this document should clearly indicate the authors, source, organisation and publication date.
The document has been produced with the funding of the European Commission. The content of this publication is the sole responsibility of the ACCORDION Consortium and its experts, and it cannot be considered to reflect the views of the European Commission. The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content.
The European Union (EU) was established in accordance with the Treaty on the European Union (Maastricht). There are currently 27 members states of the European Union. It is based on the European Communities and the member states' cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice, and the Court of Auditors (http://europa.eu.int/).

REVISION HISTORY LOG

VERSION No.	DATE	AUTHOR(S)	SUMMARY OF CHANGES
V0.1	31/3/2020	Konstantinos Tserpes	Proposed ToC
V0.2	21/4/2020	Lorenzo Blasi	Updated ToC
V0.3	3/6/2020	Lorenzo Blasi	First draft integrated
V0.4	16/9/2020	Lorenzo Blasi	Second draft integrated, references integrated
V0.5	5/10/2020	Lorenzo Blasi	Intro, Exec Summary, further contributions' integration, editing, ready for review
V0.6	9/10/2020	Lorenzo Blasi	Integrated references for section 2.2
V0.7	12/10/2020	Maria Pateraki, Lorenzo Blasi	Internal review, structure improved, removed empty & poor sections
V0.8	23/10/2020	Lorenzo Blasi (HPE), Ioannis Korontanis (HUA), Emanuele Carlini (CNR), Hanna Kavalionak (CNR), Patrizio Dazzi (CNR), Vangelis Psomakelis (ICCS), Zinelaabidine Nadir (AALTO), John Violos (ICCS), Andrea Toro (HPE), Marco Russo (HPE), Eduard Marin Fabregas (TID), Saman Zadtootaghaj (TUB), Bartlomiej Lipa (BSOFT), Maria Pateraki (OVR), Marco Di Girolamo (HPE)	Updates after internal review
V0.9	27/10/2020	Maria Pateraki, Lorenzo Blasi	Adjustments after second internal review
V1.0	28/10/2020	Lorenzo Blasi	Final editing

GLOSSARY

Acronym	Description
6DoF	Six Degrees of Freedom
AI	Artificial Intelligence
API	Application Programming Interface
ARM	Advanced RISC (Reduced Instruction Set Computing) Machine
CCUs	Concurrent Users
CI/CD	Continuous Integration / Continuous Deployment
CPU	Central Processing Unit
CRD	Custom Resource Definition
DAI	Distributed Artificial Intelligence
DHT	Distributed Hash Table
DL	Deep Learning
DNS	Distributed Naming Service
DoA	Description of Action
DP	Differential Privacy
DRL	Deep Reinforcement Learning
ETSI	European Telecommunications Standards Institute
EC	European Commission
EU	European Union
FaaS	Function as a Service
FL	Federated Learning
Gb	Gigabit
GB	Gigabyte
GPU	Graphical Processing Unit
H2020	Horizon 2020 EU Framework Programme for Research and Innovation
HDD	Hard Disk Drive
HDFS	Hadoop Distributed File System
HMD	Head Mounted Display
HTTP	HyperText Transfer Protocol
IaaS	Infrastructure as a Service
ID	IDentifier
IP	Internet Protocol
JSON	JavaScript Object Notation
K8s	Kubernetes
KB	Kilobyte
KPI	Key Performance Indicator
LF	Linux Foundation
LSTM	Long Short Term Memory
MAC	Media Access Control
MAML	Model-Agnostic Meta-Learning

MARL	Multi-Agent Reinforcement Learning
MAS	Multi-Agent System
MB	Megabyte
MDP	Markov Decision Process
MEC	Multi-access Edge Computing
ML	Machine Learning
MLF	Meta-Learning Framework
MPC	MultiParty Computation
OASIS	Organization for the Advancement of Structured Information Standards
OCCI	Open Cloud Computing Interface
P2P	Peer-to-peer
PaaS	Platform as a Service
PC	Personal Computer
PP	Privacy Preserving
QoE	Quality of Experience
QoS	Quality of Service
RAM	Random Access Memory
RDS	Resource Discovery System
REST	REpresentational State Transfer
RL	Reinforcement Learning
RNN	Recurrent Neural Network
RPC	Remote Procedure Call
RT	Research Topic
SaaS	Software as a Service
SLA	Service Level Agreement
SoA	State of the Art
SoC	System On a Chip
TCP	Transmission Control Protocol
ToC	Table of Contents
TOSCA	Topology and Orchestration Specification for Cloud Applications
UDP	User Datagram Protocol
VIM	Virtual Infrastructure Manager
VLAN	Virtual Local Area Network
VM	Virtual Machine
VR	Virtual Reality
XML	eXtensible Markup Language
YAML	YAML Ain't Markup Language

s

TABLE OF CONTENTS

EXECUTIVE SUMMARY 3

GLOSSARY 7

TABLE OF CONTENTS 9

1 Relevance to ACCORDION 11
1.1 Purpose of this document 11
1.2 Relevance to project objectives 11
1.3 Structure of the document 12
1.4 Relation to other workpackages 12

2 Analysis of the status of the art 13
2.1 Resource monitoring & characterization 13
2.1.1 Objectives 13
2.1.2 Outcome 13
2.1.3 State-of-the-art 13
2.2 Resource indexing & discovery 20
2.2.1 Objectives 20
2.2.2 Outcome 21
2.2.3 State-of-the-art 21
2.3 Edge storage, availability, reliability and performance 24
2.3.1 Objectives 24
2.3.2 Outcome 24
2.3.3 State-of-the-art 24
2.4 Pooling Edge resources 27
2.4.1 Objectives 28
2.4.2 Outcome 28
2.4.3 State-of-the-art 28
2.5 AI-based network orchestration 32
2.5.1 Objectives 32
2.5.2 Outcome 32
2.5.3 State-of-the-art 32
2.6 Resilience policies & mechanisms over heterogeneous edge resources 35
2.6.1 Objectives 36
2.6.2 Outcome 36
2.6.3 State-of-the-art 36
2.7 Techniques for secure Edge application development & deployment 40
2.7.1 Objectives 40
2.7.2 Outcome 41
2.7.3 State-of-the-art 41
2.8 Privacy preserving mechanisms 44
2.8.1 Objectives 44

2.8.2 Outcome *44*
2.8.3 State-of-the-art *44*
2.9 Application model for automatic deployment / migration of components 47
2.9.1 Objectives *47*
2.9.2 Outcome *47*
2.9.3 State-of-the-art *47*
2.10 Modelling and assessing QoE for NextGen applications 50
2.10.1 Objectives *50*
2.10.2 Outcome *50*
2.10.3 State-of-the-art *51*
2.11 DevOps tools to automate Edge applications' deployment 52
2.11.1 Objectives *53*
2.11.2 Outcome *53*
2.11.3 State-of-the-art *53*
2.12 Collaborative VR 55
2.12.1 Objectives *55*
2.12.2 Outcome *56*
2.12.3 State-of-the-art *56*
2.13 Resource federation models 60
2.13.1 Objectives *60*
2.13.2 Outcome *60*
2.13.3 State-of-the-art *60*
3 Conclusions 64
4 References 65

1 Relevance to ACCORDION

1.1 Purpose of this document

The present document is the result of the collaborative effort of all ACCORDION partners participating to Task 2.2. ACCORDION Description of Action (DoA) says: *The objective of this task is to review the state of the art of tools and technologies used in domains such as the one investigated by ACCORDION, namely in MEC, cloud computing, VR/AR and […] cloud gaming*.
The document offers a review of the State of the Art for a series of topics strictly related to the work performed in the ACCORDION project. There is actually a strict correlation between the topics analyzed in this document and the Tasks included in the three research Work Packages of ACCORDION (WP3, WP4, WP5).
Monitoring the State of the Art is an ongoing activity in ACCORDION, aimed at ensuring that project results are in line with the expected innovation level, and at avoiding some identified technical risks (Tec4 and Tec5 as indicated in p. 25 of deliverable D1.2). For this, three versions of this document will be provided during the life cycle of the project. The present document is the first version, due at M10, and two other versions will follow at M22 and M35.

1.2 Relevance to project objectives

The research topics analyzed in this document are extremely relevant to the ACCORDION project objectives. As described in the DoA, each project Objective (Ox) is supported by several Enablers (Ex) and each topic analyzed in this document is related to at least one of the stated Enablers. The following table indicates the relationship between each topic and the related Enablers. Please note that not all the planned Research Topics will be included in this first version of the deliverable: if the section number is not indicated in this table, the analysis of the related Research Topic will be included in one of the next versions of the deliverable.

Table 1 - Relationship between research topic and their related Enablers

Section	Research Topic	Enabler
2.1	Resource monitoring & characterization	O1-E1
2.2	Resource indexing & discovery	O1-E2
2.3	Edge storage, availability, reliability and performance	O1-E3
	Automatize Unikernel creation	O1-E4
2.4	Pooling Edge resources	O1-E*
	Edge resource allocation approaches	O2-E1
2.5	AI-based network orchestration	O2-E2
2.6	Resilience policies & mechanisms over heterogeneous edge resources	O2-E3
2.7	Techniques for secure Edge application development & deployment	O2-E4
2.8	Privacy preserving mechanisms	O2-E5
2.9	Application model for automatic deployment / migration of components	O3-E1
2.10	Modelling and assessing QoE for NextGen applications	O3-E2
	Orchestration for Edge network performance assessment and recovery	O3-E3
2.11	DevOps tools to automate Edge applications' deployment	O3-E4

Section	Research Topic	Enabler
2.12	Collaborative VR	O4-E1
2.13	Resource federation models	O5-E*
	Efficient exploitation of GPUs, accelerators and FPGA	O4-E*

1.3 Structure of the document

The main part of this document is section 2, in which all the State of the Art analysis results have been reported. Section 2 has a subsection for each of the topics that have been deemed relevant in the preparatory work. Each subsection includes: a description of the Objectives of the related Research Topic, a list of the Outcomes expected from the ACCORDION research work on this topic, and an analysis of the State of the Art for the topic. In some cases more than one subtopic, related to the main one, have been analyzed and reported.

1.4 Relation to other workpackages

As indicated at the beginning of the related subsection, each Research Topic analyzed in this document is connected to one project Task from Work Packages WP3, WP4 and WP5. The Objectives and Outcomes parts of each subsection explain the relationship between the analyzed topic and the related Task. The State of the Art analysis work reported in this document has been a useful starting point for each of those Tasks and allowed some of them to find several baseline tools and technologies to be reused, after proper evaluation.

2 Analysis of the status of the art

This section includes a subsection for each Research Topic. The analysis of each topic include statements on its objective(s), outcome and the current status of the art in the relevant field. Given that it might not be entirely clear what the objectives and outcomes are at this stage, the authors provided those that they judged having the highest scientific interest and the best impact on the related task, given the resources at hand.

2.1 Resource monitoring & characterization

The objectives and outcomes for this Research Topic are related to the work performed in Task 3.1.

2.1.1 Objectives

2.1.1.1 CHARACTERIZING EDGE RESOURCES

Regarding the Characterization of Edge resources we need to develop a taxonomy model with the ability to classify them based on their a) computing power and b) energy capacity, but also on as much information as possible that can characterize the target platform. A model like that is of great importance as it has to be able to characterize heterogeneous resources and their capability to host a specific application task based on the monitoring.

2.1.1.2 MONITORING EDGE RESOURCES AT RUN TIME

The goal of Monitoring Edge resources is to be able to have information about the performance, behaviour, workload, etc. This information will be stored in a database so it will be available via querying. Operations like orchestration, scaling and migration can benefit from the monitoring as in each possible scenario database will have metrics about the resources and contribute towards ensuring a high QoS and QoE level to its user.

2.1.2 Outcome

The ACCORDION research work on this topic will:

- Decide the ontology model to describe the resources
- Find suitable classification algorithms
- Decide a monitoring architecture (bare metal Vs virtualization layer)

2.1.3 State-of-the-art

The next subsection 2.1.3.1 (Monitoring) corresponds to the objective described in sect. 2.1.1.2 (Monitoring Edge Resources at Runtime), whereas the subsections 2.1.3.2 (Characterization) and 2.1.3.3 (Classification) correspond to the objective in 2.1.1.1 (Characterizing Edge Resources).

2.1.3.1 MONITORING

The expectations that we have from a monitoring tool are not static as they are continuously evolving. As it is understandable there is no tool that can satisfy all expectations but we have some factors to define and choose a tool that is the most suitable for our monitoring or even develop a custom one. To guarantee a decently performing Cloud the monitor needs to be done in Iaas, Paas and in SaaS [1], monitoring across layers can provide significant information for the application performance and system performance to both the consumer and cloud provider.

Monitoring tools have to be able to connect the information that is provided from the physical and virtual layer, so it would be able to describe in a comprehensive way how a problem in the physical layer can impact the virtual layer [20]. Performance to comply with the SLA agreements [2] mappings have to be done between low resource metrics and high SLA parameters in order to avoid violations. There is also a rising need for cloud agnostic monitoring tools [3], to cover the monitoring of different cloud solutions with a single monitoring tool. Open source tools (Prometheus [4], PyMon [5]) tend to have this capability; on the other hand commercially available tools usually provide monitoring solutions for their own particular resources and services (CloudWatch [6]). Cloud agnostic monitoring tools are commonly used in hybrid cloud to monitor both public and private cloud services and in multi-cloud solutions.

A big factor is the heterogeneity, there are plenty of devices with different computational power that can be used on Cloud and Edge environments but in each case different things have to be considered. RaspberryPIs are commonly used as Edge resources and upon selection one has [7][8] to consider technologies and monitoring tools that support ARM architectures like Docker Swarm and Kubernetes. Kubernetes has also K3s[1] for ARM devices which is a lightweight solution to avoid the additional workload. Even the model of the device can play a big role. For example cAdvisor [9] which is a container that exposes metrics of other containers of the same host to Prometheus and is a perfect solution for a cluster of PCs has no Docker image available for arm7. There are several projects on Github that try to make Docker monitoring an easy job. Monit-docker [11] is a command line tool that monitors Docker containers and executes Docker commands. The resource usage metrics that are available with monit-docker are the status, memory usage, memory percent, network transmit, network receive, CPU usage, CPU percent, io_read, io_write and memory limit of containers.

Wrappers for Docker are also available, for example DoMonit [10] which uses the Docker API. The goal of DoMonit is to write python scripts easily for monitoring all Docker containers. Both of monit-docker [11] and DoMonit [10] can be used to create a custom solution for monitoring docker containers on RaspberryPIs. Docker-alertd [12] monitors the Docker stats and sends alerts via email when usage limits have been exceeded. The alerts can be triggered based on the container existence, running state, memory usage, CPU usage and minimum process of the running container. For physical monitoring the combination of Node Exporter [41] with Prometheus [4] is the most popular in case of PCs and RaspberryPIs as there are a lot of tutorials available.

Prometheus can be used for cluster monitoring either on Kubernetes or Docker Swarm. A project that uses a combination of container/pod and node monitoring is Cluster Monitoring stack for ARM / X86-64 platforms in Github[2]. This project uses Prometheus Operator[3] to manage and configure Prometheus instances on Kubernetes / K3s. It uses Node Exporter [41] to monitor the nodes of a

[1] https://rancher.com/docs/k3s/latest/en/

[2] https://github.com/carlosedp/cluster-monitoring

[3] https://github.com/prometheus-operator/prometheus-operator

Kubernetes / K3s cluster and expose their hardware and OS metrics. Also in this monitoring stack there is kube-state-metrics[4], which is as agent that listens to the Kubernetes API server and generates metrics (health of nodes, pods, deployments, etc.).

2.1.3.2 CHARACTERIZATION

Monitoring tools provide metrics, but also a model or a protocol is required to represent and characterize resources as part of this Task. OCCI [13] is a RESTful protocol and API that has models to describe IaaS, PaaS and SaaS. One of these models is the infrastructure model. The OCCI infrastructure model has five Infrastructure types:

- Compute – Information about processing resources, example: VM/container (CPU architecture, number of virtual CPU cores assigned, Fully Qualified DNS hostname, relative number of CPU shares, maximum RAM in gigabytes allocated, current state, human-readable explanation of the current instance state)
- Network – Interconnection resource that represents an L2 networking resource (802.1q VLAN identifier, tag based VLANs, current state, human-readable explanation of the current instance state)
- Storage – Information about storage resources (storage size of the instance in gigabytes, current status, human-readable explanation of the current instance state)
- NetworkInterface – which is a connection of Compute and Network instance (identifier that relates the link to the link's device interface, MAC address associated with the link's device interface, current status, human-readable explanation of the current instance state)
- IPNetworkMixin In order to support L3/L4 capabilities (e.g., IP, TCP, etc.) an OCCI mixin is herewith defined (Internet Protocol (IP) network address (e.g., 192.168.0.1/24, fc00::/7), Internet Protocol (IP) network address (e.g., 192.168.0.1, fc00::), address allocation mechanism: dynamic e.g., uses the dynamic host configuration protocol, static e.g., uses user-supplied static network configurations)
- StorageLink - connects a Compute instance to a Storage instance (device identifier as defined by the OCCI service provider, point to where the storage is mounted in the guest OS, current status, human-readable explanation of the current instance state.)

The OCCI Infrastructure model is an extension of the OCCI Core model designed to describe IaaS APIs. The OCCI Core model can describe resources in JSON through OCCI JSON Rendering [14] or in text with Text Rendering [15]. In case of JSON we have a JSON Object that presents information of OCCI Kind, OCCI Mixin, OCCI Action, OCCI Link and OCCI Resource. OCCI Actions are also presented with a JSON Object. OCCI Resource Instance Rendering consists of a JSON object. Compute namespace of OCCI Infrastructure model is being used along with the *network* interface and *ipnetwork* interface to describe the compute resource. *Network* interface is a subtype of *Link*, it is being used to add network information to the compute type. The most important thing is the key com.example.occi.templates.myosmixin which indicates that custom namespaces could be created to have additional information. In the document of OCCI JSON Rendering [14] there is a JSON schema for validation purposes. In case of Text Rendering [15] a different syntax is presented to describe a

[4] https://github.com/kubernetes/kube-state-metrics

resource, a resource must have a *Category*, a *Link* and an *Attribute type*. There are two options: either to use plain text or OCCI Head Rendering to be placed in the HTTP Header, and header fields must follow the specification in RFC 7230. As it is possible that this information could be available through a Web Service, it is preferable to use JSON Rendering instead of Text Rendering. In our case a model that describes physical resources is preferable, but the documentation of the OCCI Infrastructure model is not clear if it has this capability or not.
There are researches that present ontologies to describe infrastructures, as the CoCoOn [19] that is an ontology on OWL for describing IaaS. In CoCoOn IaaS can be classified in three categories: cocoon:ComputeService, cocoon:StorageService and cocoon:NetworkService. In case of Google Cloud, CoCoOn is able to describe the maximum number of disks that can be assigned to a VM and the maximum total disk size that a VM has. Price description is also available for cloud services virtual machine images, storage, network services and network transactions.

Table 2 - CoCoOnc lasses that describe IaaS characteristics

CoCoOn	Description
cocoon:ComputeService	Similar to the Compute type of OCCI [13] as it describes the physical resources assigned to a VM.
cocoon:StorageService	Local Storage and Network Storage to describe snapshot options, the maximum number of input and output operations and the max storage throughput
cocoon:NetworkService	cocoon:StaticIPService to represent the reservation of an external static IP of a VM
cocoon:CloudServicePriceSpecification	1. gr:hasCurrencyValue to define the currency. 2. cocoon:chargedPerCore to define the price of VM based on the number of CPUs 3. cocoon:forCoresMoreThan to describe the price of vms with more cores than the specified 4. cocoon:forCoresLessEqual to define the price of vms less number of CPUs than or equal to the specified 5. cocoon:forUsageLessEqual and cocoon:forUsageMoreThan to define the price of network service based on monthly usage
cocoon:Location	To describe locations like cities. If we know the address of a location we could use cocoon:inPhysicalLocation, otherwise we could use cocoon:inJurisdiction.
cocoon:Region	subclass of cocoon:Location to represent known cloud services regions
cocoon:UnitOfMeasure	uses unit vocabulary [17] to have units like unit:Hour or unit:MegabitsPerSecond and to define other units like cocoon:GB, cocoon:GBPerHour, cocoon:GBPerMonth
gr:BusinessEntity	to describe providers like cocoon:Azure

An alternative is Cloud Modeling Language or CloudML [22], which is an XML-based language to describe computational resources, network resources, services profiles, developers' requests and the geographical locations in Distributed Clouds. CloudML is able to represent physical resources and virtual resources in a Distributed Cloud.

Cloud Modeling Language	**Description**
XML element NodeStatusType	CPU, RAM and Storage description for a node. CPU and RAM attributes have percentage values while Storage has a value that defines the used space, the unit of the space (KB, MB, GB, etc.) and an ID.
XML element VirNodeStatusType	to describe the status of a VM.It has three attributes i) an ID that is a unique value to identify a VM, ii) the Owner to describe the developer who owns the VM and iii) VMstate to present the current state (running, stopped, suspended) of the VM. Also has a NodeStatusType to describe the resources used by VMs.
XML element PhysicalNodeType	To describe physical resources, it consists of an ID attribute and NodeParams, PhyIface, VirNodeID, VirEnvironment elements. 1. NodeParams describes memory, processor and storage of a node along with its geographical location, its role in the network (switch, server, etc.), its current status via NodeStatusType element and OtherParams to provide general information and extension. 2. PhyIface describes physical links. 3. VirNodeID a list of VM IDs hosted on a node 4. In VirEnvironment we can have information like the architecture (32 or 64 bits), the virtualization method (full or paravirtualized) and the hypervisor.
XML element VirtualNodeType	To describe VMs with the attributes ID, Owner and VMState along with NodeParams, VirtInterface (virtual links) and VirEnvironment elements. VirEnvironment has two attributes, one for the hypervisor and one to describe the virtualization mode of the VM.
InfrastructureType	Phyinfra is PhysicalInfrastuctureType has an ID attribute and the elements PhyNode and PhyLink to describe the physical nodes and their connections. PhyInfra element and none or more VirInfra element. PhyInfra describes a physical infrastructure and VirInfra (VirtualInfrastructureType) describes the virtual infrastructures that are being currently hosted to it.
PhysicalLinkType	PhyLink is a PhysicalLinkType. contains an ID attribute, a LinkParams element and none or multiple VirLinkID elements
VirtualInfrastructureType	Describes the collection of virtual nodes and links. It has an ID attribute, an Owner attribute which describes the client who owns the virtual resources along with none or multiple VirNode elements of VirtualNodeType and VirLink of VirtualLinkType which is like

	the PhysicalLinkType.
PhysicalLinkType	Contains an ID attribute, a LinkParams element and none or multiple VirLinkID elements. LinkParametersType describes link technology (Ethernet cable, Wi-Fi), capacity, current status (current delay, current, allocated rate and bit error rate) and OtherParams to extend the description. VirLinkID presents the virtual links currently allocated on this physical link.

Another way to describe physical resources is with TOSCA Simple Profile in YAML [16], TOSCA is an open standard of OASIS that defines the interoperable description of services and applications hosted on the cloud. TOSCA Simple Profile in YAML has a node template to describe components of an application and a topology template to describe nodes and their relationships. TOSCA YAML has the capability to group nodes of a node template [16] by grouping physical resources and produce cluster description files which could be provided to other components of ACCORDION via a web service. TOSCA seems to have easier representation than OCCI [13], if we compare TOSCA with CloudML [22] and CoCoOn [19]. As it is an open standard we can easily find documentations tutorials and papers that could help in adding extensions to make it suitable for our case, CloudML and CoCoOn have not reached yet the popularity of TOSCA.

Table 3 - Comparing Characterization Technologies

	Advantages	**Disadvantages**	**Format**
OCCI	Describes the resources of physical infrastructure that are assigned to a VM/container. A JSON schema is available [14] for validation purposes	It is not able to describe physical resources. It seems to have a very complex description for resources JSON Rendering [14].	JSON / TEXT
CoCoOn	Describes the resources of physical infrastructure that are assigned to a VM/container. On github there is a set of SPARQL generate scripts [21] for mapping data from to semantic data along with examples of CoCoOn [18].	It is not able to describe physical resources. OWL can be expressive and describe objects and their connections but to extend an ontology like CoCoOn to fit our case it may be time consuming.	OWL
CloudML	Describes both resources of physical infrastructure and the resources of physical infrastructure that are assigned to a VM/container.	Its description can be too verbose, due to the XML format, so it may be difficult to create cluster descriptions. There are no available tools that can help.	XML
TOSCA YAML	Describes the resources of physical infrastructure. It can also describe a group of servers. There are tools that can parse and validate TOSCA YAML. It seems to have an easier way of extension.	It is designed to represent containers, web applications and the infrastructure (Compute node), extensions need to be made to add more information..	YAML

2.1.3.3 CLASSIFICATION

We may have to create taxonomies of physical resources based on their computational power to assist the orchestration of our Use Cases scenarios. Basically to clarify which group of the physical

resources can support high computational tasks and which are less capable. One solution is to apply benchmarks on the compute nodes to assess their capabilities. To apply benchmarks to the Cloud [29] we have to use Cloud services to put compute nodes under test. Benchmarking in the Cloud usually has to answer which IaaS does most effectively host a number of parallel mid-size application instances but in our case we need to answer which physical resources are capable to host power consuming services with multiple instances or containers as we will have heterogeneous devices and we have to create taxonomies.

Based on [30] the performance of the physical properties of the Cloud can be evaluated on communication, computation, memory cache and storage through transaction speed, availability, latency, reliability (Failure Rate or Mean Time Between Failure), data throughput, scalability and variability (the state of spread of a set of data) to find the performance of the related Cloud service. In [37] benchmark tools were combined to perform the evaluation of resources. For the evaluation of CPU performance minimum GFLOPS, maximum GFLOPS, average GFLOPS, Integer (single core or multicore) and Floating Point (single core or multicore) calculation were used. For the evaluation of memory performance an Integer average and a Floating Point average were used. Integer and Floating Point are computing stressing calculations. Integer calculations uses complete numbers, text and other similar items but Floating Point is used in more compute consuming situations like worksheets, graphical theory applications and video games. FLOPS are being used to evaluate the performance of a processor, they can calculate the floating point unit of a processor. For disk performance the research performed read, random read, write, random write operations and finally for network performance bandwidth, jitter and throughput operations. All average results were calculated by repeating the benchmarks frequently. The target of this research was to compare the performance of Azure with OpenStack. In the experiments LINPACK [32] was used to evaluated the CPU performance based on performing numerical linear algebra calculations. Geekbench [31] was used to evaluate the single core and multicore CPU performance the of Windows Azure and OpenStack instances by doing floating point calculations. For the evaluation of memory RAMspeed [33] was used to do integer and floating point calculations on Azure and OpenStack instances. STREAM [34] benchmark evaluated the VMs of Azure and OpenStack based on raw memory performance. IOzone [35] did sequential and random read and write in the storages of the OpenStack and Azure instances. In case of iPerf [36] it gathered the information about UDP packet loss and in case of TCP it measured bandwidth and jitter.

Another solution for creating groups of computing nodes is to use traditional classification or clustering algorithms or even use other algorithms to create the taxonomies. For instance in [38] two evolutionary algorithms for classification, Imperialist Competitive Algorithm (ICA) and Particle Swarm Optimization (PSO) are compared in the training of a neural network. Imperialist Competitive Algorithm [39] adds some random cost to nodes, then creates two groups based on the cost, the ones with low cost are named imperialists and the other ones colonies. There are three steps in this algorithm Assimilation, Revolution and Competition. In the step of Assimilation imperialists try to take on their side colonies by improving their position. In the second step, colonies as they are moving closer to their imperialists they may reach a position with a better cost than imperialist, in this case the colony and the imperialist will change roles. In the step of Competition in this step imperialists try to keep their colonies and steal colonies of other imperialists. Each imperialist who cannot keep or increase their colonies will be eliminated, the weaker imperialists will lose their colonies until all colonies will belong to one imperialist. The colonies will have the same cost with the imperialist but they will be under control of the imperialist. Similar to this logic is the Particle Swarm Optimization [40], a number of particles is initialized with random positions and velocities. The best position of the best particle will be stored Global best, the position of each particle will stored to Local best. The

new position of each particle will be calculated by the summation of current values of the position of particle, velocity vector, distance of Local best from position of particle and distance of Global best from position of particle. In each step we will compare the new position of each particle with the Local best to find if it better than its Local best or not, if it is better we have to change the Local best and then it will be compared with the Global best too.

Table 4 - How we can use benchmarks or evolutionary algorithms

Evaluation Methods	Conclusions from bibliography
Benchmarks	Combine Geekbench [31] and LINPACK [32] for CPU performance, RAMspeed [33] and STREAM [34] for RAM performance, IOzone for disk assessment [35] and iPerf [36] for network assessment like in research [37] to know which physical resources can handle high computational containers / services.
ICA & PSO	Imperialist Competitive Algorithm (ICA) [39] and Particle Swarm Optimization (PSO) [40] had accuracy 97.60% and 97.50% respectively in generation 20, which is a decent speed and accuracy to be considered for our case. We can consider ICA as a solution candidate for creating groups of powerful and weak computing nodes in the Edge, based on clustering results in [38].

2.1.3.4 CONCLUSIONS AND HIGHLIGHTS

To summarize, a mechanism that will monitor the physical resources and virtual resources (containers/VMs) despite the device heterogeneity is needed. Container monitoring needs to have metrics on how the containers impact the physical resources of the host. The best candidate seems to be the "Cluster Monitoring stack for ARM / X86-64 platforms" project (reviewed in 2.1.3.1) as it monitors both nodes and pods on K3s clusters by using Prometheus. In addition, characterization will be able to describe the physical resources of a cluster, to provide to other components of ACCORDION as much information as possible about the devices. As we will have descriptions for physical resources the next step would be to create taxonomies based on their performance and provide the created groups to the orchestrator component of ACCORDION.

2.2 Resource indexing & discovery

The objectives and outcomes for this Research Topic are related to the work performed in Task 3.2.

2.2.1 Objectives

2.2.1.1 EDGE RESOURCES' DISCOVERY

Resource Discovery Systems (RDSs) are the backbones of orchestration services, as they implement the querying and indexing capability necessary for computing optimal placement of services on top of computational resources. RDSs need to be carefully designed when dealing with edge applications having the end-users in mind, as they have to manage the load coming from multiple and heterogeneous devices while dealing with possible sensible information.

2.2.1.2 EDGE RESOURCES' INDEXING

Once resources at the edge have been discovered, it is of paramount importance to define proper solutions aimed at their indexing. Indexing approaches need to be properly tailored to the nature of edge resources that could be ephemeral in presence and highly variable in terms of availability.

2.2.2 Outcome

The ACCORDION research work on this topic will:

- Design a distributed RDS for the indexing of potential sensible information, such that to keep the locality and privacy
- Scale-up the RDS with the amount and dynamicity (e.g. rate of changes) of resources and environment
- Support resource discovery query for a class of specific applications (i.e. latency-aware, interactive)
- Define a flexible indexing approach
- Let the indexing approach deal with uncertainty of availability and presence
- Ensure the appropriateness of the solution taking into account the specific needs of the use cases

2.2.3 State-of-the-art

This SoA section corresponds to the two objectives indicated above, which are the same addressed in Task 3.2 (Resource indexing & discovery). In this section we provide an overview of the existing most effective resource information dissemination and information advertising strategies. Therefore, this section will focus on those approaches that are or could be (i) able to store and retrieve information about resources having a different level of dynamicity in a scalable fashion, (ii) support a user-provided level of performance and precision, and (iii) support different kind of queries.
A resource management system [42] evaluates resources in terms of four perspectives: (i) resource type, (ii) objective, (iii) resource location, and (iv) resource use. Within ACCORDION, exploiting edge resources has a specific impact on those four aspects, multiplying available options and allowing most parameters (including resource availability and membership) to dynamically change. Composite, heterogeneous platforms need a (composed) Resource Discovery Service (RDS) to keep track of the whole infrastructure, enhancing and mediating resource localization as well as retrieval and advertising of resource information. To address such infrastructures a static, centralized RDS architecture is hardly practical. The challenge within ACCORDION is instead to create a scalable, efficient and robust RDS that fits with all the platform architectural requirements. As stated in [43] an RDS design has three relevant macro aspects: (i) underlying service structure, (ii) query and protocol design, and (iii) service evaluation metrics. The *underlying* aspect sums up the computing environment in which the RDS operate, how the RDS own resources are distributed and organized. Many proposals have a take on these aspects, where a core choice is about the degree of decentralization for the resource directory, the most common distributed approach being Distributed Hash Tables (DHTs) or distributed Tree-like data structures. An important aspect in this category is whether solutions employ data delegation, in which data are not stored locally where they are generated, but its storage is delegated to other nodes. The delegation naturally offers some advantages to manage heterogeneity, as simple device can offload to more powerful one (e.g. in a hierarchical fashion). However, it might introduce extra overhead, especially when data is highly dynamic. The *query and protocol design* aspect concerns what kind of resource queries are generated and how (sub) queries can navigate the platform. Where even blind search can be appropriate for large, dynamic environments with high resource volatility, informed search improves its efficiency and effectiveness by exploiting assumptions in the nature and stability of resources. For the purposes of ACCORDION

is important to support multi attribute range queries to support resource discovery. Finally, the *service evaluation metrics* describe the target objective of the RDS. Common evaluation targets for many RDS, which are also core aspects in ACCORDION, are (1) scalability, both in terms of amount of indexed resources and bandwidth of requests to manage, and (2) efficiency, in terms of discovery latency and overhead. Generally, unlike centralized approaches, distributed approaches naturally offer an inherent degree of scalability support and efficiency that well match the needs of ACCORDION.
Starting from these considerations, here we discuss those solutions more closely related to the ACCORDION needs, comparing them with respect to the following aspects: (i) support to multi attribute queries, (ii) underlying distributed data structure, (iii) data delegation.

Table 5 - List of discussed solutions for organizing data in Resource Discovery Services

Approach name	Multi-attribute solution	Underlying distributed data structure	Data delegation
dragon	Yes, Space Filling Curves (SFC)	Binary tree over DHT	no
Baton	yes	Binary tree	yes
Q-tree	yes	Quad tree over DHT	yes
PHT	no	Trie over DHT	yes
P-Grid	no	Bonary tree, balanced trie	yes
Saturn	no	Multiple (virtual) order preserving DHT	yes
MAAN	Yes, multiple spaces	Single DHT	yes
LORM	yes	Single DHT	yes
Squid	Yes, SFC	Single DHT	yes
MatchTree	yes	Multicast tree on demand	no
IoT Discovery Service	yes	Prefix hash tree over Kademlia DHT	yes
ECHO	no	Prefix hash tree over DHT (optimized)	yes

Baton [44], BAlanced Tree Overlay Network, is based on a binary balanced tree structure in which each node of the tree is maintained by a peer. Each node, both leaf and internal, is paired with a range of values and data published is stored on the peer managing the corresponding interval of values, hence data delegation is exploited. These ranges are dynamically adjusted at each node so to guarantee load balancing, with overloaded nodes transferring part of their contents to other nodes.
Q-tree [45] provides a multi-attribute query resolution for hierarchically clustered environments like tele-immersive interactive systems. Each node is assigned a range interval that specifies which items it stores and also knows the entire range of the subtree rooted at it. Data delegation is exploited to store data in the tree.
PHT [46] builds a static tree to route the query resolution, however, it supports only single attribute range queries. Specifically, it builds a trie[5] on top of a DHT by exploiting the high-level operations lookup, put and delete of the underlying DHT. The management of a key K is delegated to a leaf node whose label is a prefix of K. In order to resolve queries and perform updates, PHT requires to find the leaf of the trie, which in turn requires a variable number of DHT lookup depending on the number of unique prefixes in the trie.
P-Grid [47] builds a static tree to route the query resolution and it does not require an underlying DHT. However, it resolves only single attribute range queries. In P-Grid each peer is associated with a leaf of the binary tree and for each level of the tree it maintains a reference to some other peer that

[55] Please note that this is not a misspelling for "tree": the data structure is actually named "trie".

does not pertain to the peer's subtree at that level. P-Grid needs a sample of data to build a balanced trie and exploits data delegation mechanism to store the data in the peers.
An instance of Saturn [48] consists of an order preserving DHT ring and a number of virtual rings where resources are distributed using a defined Multiring Hash Function. Range queries are solved by randomly selecting one of the virtual rings, exploring the nodes from the lower until the upper bound of the queries. As PHT, also Saturn does not provide support for multi-attribute range queries.
MAAN [49] maps objects on separate address spaces (one for each dimension), which are then commonly collapsed in a single DHT. Multi-attributes range queries are resolved considering the attribute that minimize the address space to explore, and by filtering out the results for the rest of attributes. MAAN can lead to high network overhead when values change rapidly, since it requires an update for the whole resource for each attribute. MAAN can also create hot-spots since bulk of information can end up managed by few nodes.
LORM [50] organizes nodes in a set of clusters distributed in a DHT ring, such that each cluster is responsible for the management of one attribute. Range queries are solved routing one sub-query for every attribute to the cluster responsible of the attribute and then aggregating the results. The downsides of this approach are the following: (i) nodes can be responsible of a large amount of information, causing large overhead in case of churn (due to data delegation); (ii) resources information must be refreshed periodically, increasing the network load.
Squid [51] solves multi attribute range queries using a locality preserving indexing scheme based on the Hilbert SFC. The SFC index space is chosen to be the same as the node identifier space, and each peer is responsible for the data in its segment. Squid's query resolution approach can be viewed as constructing a tree and visiting it top-down, increasing the prefix by one at every level of the tree and checking if the cluster of the SFC-based index space matches the range query. However, this approach can overload the root of the trees because the peer handling the shortest prefixes of the identifier space are contacted frequently to start the query processing. In addition, Squid requires a load balancing mechanism as the uniformly distribution of node identifiers leads to data unbalancing.
MatchTree [52] is a self-organizing recursive-partitioning multi-cast tree where the tree structure is built according to the query. Queries are propagated into the tree according to the goodness of the values, and results are returned aggregated and sorted by rank. MatchTree employs a set of heuristics to increase query performance and a redundant topology to support failure. MatchTree provides the resolution of multi attribute range queries on top of the tree. Even if it adds interesting functionalities like ranking results, MatchTree suffers of network bandwidth consumption because a different tree must be created for every request.
Paganelli et al. [53] proposed a Discovery Service specifically designed for Internet of Things scenarios which supports multi-attribute range-queries and adopts a peer-to-peer approach for guaranteeing scalability, robustness, and maintainability of the overall system. The Discovery Service linearises multi-attributes through space filling curves and exploits an indexing PHT structure (previously presented) built on top of the Kademlia DHT overlay network.
Echo [54] employs an extended Prefix Hash Tree (PHT) on top of a DHT underlying structure. ECHO extends the PHT by introducing an additional distributed structure called Tabu Routing Table which stores information about the index structure shape. This additional data structure aims to reduce the search space and, consequently, reduces the message overheads and improves the speed of query resolution
Dragon [55] is a distributed storage and query data structure designed to support fast changing data. Dragon builds an aggregation tree on top of a DHT address space, in which every node manages a contiguous part of the tree. Dragon also supports multi-attribute range queries by linearizing the attributes space using a space filling curve approach.

2.2.3.1 CONCLUSIONS AND HIGHLIGHTS

In recent years a lot of research has been performed in the field of data indexing and discovery. Different strategies for the data discovery impact on the definition of the distributed data structure. Important directions to investigate include the possibility to exploit a mix of structured and unstructured overlay mechanisms, in order to achieve the desired level of scalability. Finally, particular emphasis will be put on making the system configurable at run-time, allowing to select preferred levels of precision, according to the needs of the application and the resources available.

2.3 Edge storage, availability, reliability and performance

The objectives and outcomes for this Research Topic are related to the work performed in Task 3.3.

2.3.1 Objectives

2.3.1.1 DATA STORAGE AND RETRIEVAL OPTIMIZATION

Most, if not all, of the applications that will be using ACCORDION are highly dependent on big amounts of data. These data can be located either in edge devices, cloudlets or the cloud. We have to create a platform that can identify where the data are or should be located for optimal performance and response time, how we can combine the two in order to avoid moving or duplicating the data and how we get from the present state of the data storage to the optimal one, if needed. This process may have to overcome issues like unstable network connections, unreliable edge devices, data privacy and security and other related issues. We will also examine current research issues in the field such as intelligent caching and pre-fetching, pro-active and real time data bundling and edge resource optimization.

2.3.1.2 DATA CLUSTER MANAGEMENT

We expect to tackle challenges like scheduling the process of different micro-services orchestrators (K8S for cloud & K3S for edge), providing efficient ways of interconnecting different clusters and managing services among them, creating and managing persistent volumes and storage for enabling stateful applications in K3S and K8S.

2.3.2 Outcome

The ACCORDION research work on this topic will:

- Coordinate the edge, cloud and cloudlet data resources
- Handle infrastructure failures ensuring QoS and QoE requirements
- Optimize the data resource demand, balancing and selection process

2.3.3 State-of-the-art

This section's State-of-the-art (SoA) covers both of the Objectives stated above (sect. 2.3.1), but the first part on Storage systems (sect. 2.3.3.2) is more related to the first objective (Data storage and

retrieval optimization, sect. 2.3.1.1), whereas the Software part (sect. 2.3.3.3) and the Open research issues (sect. 2.3.3.4) are related to both objectives.

2.3.3.1 INTRODUCTION

Any computing task needs data in order to perform its main function. Following that fact we cannot have an edge computing platform without an edge storage management system. This system needs to overcome the inherent edge challenges, like unreliable devices and network, hardware and software incompatibilities that arise due to the plethora of different devices used, mobility of the devices and the users, limited resources of the edge devices, security and privacy concerns and other challenges. Our research revolves around the decisions we need to take in order to design an edge storage system, considering the storage type, the storage system and the system architecture, both the physical and the software architecture.
When referring to edge networks we assume the network between devices close to the users of the network. Edge computing is covering a vast area of architectures, ranging from remote servers, to personal computers, laptops, tablets or even mobiles interconnecting to form peer to peer or hierarchical clusters. European Telecommunications Standards Institute (ETSI) is distinguishing the Mobile Edge Computing, or multi-access edge computing as it was later called [56][57] (MEC) from the edge computing that uses homogenous machines, as an extension of the cloud network.

2.3.3.2 STORAGE SYSTEMS

First we need to define the main types of storage systems, which are:

- Traditional file systems,
- Block storage systems,
- Object storage systems.

The traditional file systems are usually found in powerful machines or in single machine clusters as they do not natively support distributed architectures, scaling or elasticity. They are direct representations of data blocks located in disk drives. They can be used in clusters by customizing the network and user access rights and their replication and distribution amongst the nodes, which is a tedious work.
The block storage systems are trying to evolve the file systems by managing the data in block level. As we mentioned, file systems split the data into data blocks that are stored in a disk drive. Block storage systems are taking these blocks and managing their distribution, replication and recovery, if needed, across all available disks in the cluster, providing a distributed alternative to the traditional file systems. Of course, the security options and data access is following the same rules as the traditional file systems since all the distribution, replication and recovery actions are taken in a lower level, so it is still a tedious task to manage the data themselves and the users that can access them.
Object storage systems are considered as the most easy to use but also as the slowest and most resource demanding. They manage the distribution, replication and recovery on object level, which can aid us in the storage of a great deal of metadata which the block storage cannot provide. Since objects are larger than blocks and objects are usually split between physical locations, storage systems of this category add many overheads on even the simplest of queries so they are not recommended for clusters with many nodes, especially in scenarios that these nodes are devices with poor resources.
At a later stage we have to identify possible software solutions that fit an edge storage platform and analyse their strengths and weaknesses. In our research we encountered a plethora of open source

tools that support storage over clusters. The most relevant and popular of them are presented in the next subsections.

2.3.3.3 SOFTWARE

OpenStack [58] is a framework that enables uniform communication and coordination between cluster nodes. It provides many tools that include computation and storage node setup and management. Regarding the storage it is providing two options, the Cinder and Swift frameworks. Cinder [59] is a block storage option whereas Swift [60] is an object storage one. They both provide a terminal filesystem like interface in order to write, access, modify and configure files. Another interesting component of OpenStack is Nova [61], which manages the cluster nodes, ensuring security and integrity during transactions. There have been attempts to modify the Nova component and couple it with systems other than Cinder and Swift in order to create a fully decentralized solution, optimized for fog computing and edge storage [62], [63]. The main issue when deploying OpenStack in edge clusters is the centralized AMQP bus that it uses for transaction control. So Lebre et al [62] proposed to replace that with a distributed solution such as RabbitMQ, ActiveMQ or ZeroMQ. Their second issue was to distribute the databases themselves and they tackled it by using Redis, which is a P2P solution and hence inherently decentralized. Nova was communicating with Redis using a customized ORM library that communicates with Redis through RPC, just like SQLAlchemy interacts with relational databases.

Another option is Hadoop [64], an innately distributed and very lightweight system that is used for turning sets of commodity machines into high performance computational clusters. Hadoop has its own storage system, called HDFS [65], which is also highly distributed and scalable. Moreover, its lightweight nature makes it a good choice for edge devices as described in relative literature [66]. HDFS is implementing a master-slave architecture so it needs a central node, called NameNode, that manages the rest of the nodes, called DataNodes, and acts like an access point for the clients. The DataNodes are responsible for the block storage implementation of the storage system. They handle the distribution, replication and recovery of the blocks as the NameNode coordinates them and requests the blocks needed in order to serve the clients. Nevertheless, user data never pass through the NameNode. The system is designed in such a way that data blocks are directly exchanged between the client and the DataNodes, so the resource requirements of all the nodes are balanced.

CEPH [67] is also a viable solution for cloud, fog and edge storage solutions. It is an open source system that has been used in a number of edge applications as it can support big clusters of edge devices and is greatly scalable [66]. CEPH also supports both object storage and block storage, making it pretty agile and compatible with almost any use case scenario. It also employs a hierarchical architecture which needs even more resources than other systems since the metadata it keeps add a great overhead to its processes. The downside is that it runs only on linux systems, the resources needed are becoming too high to handle as the cluster grows and it is quite complex to configure and maintain.

MinIO [68] is a newer open source framework by IBM. It is an inherently decentralized, peer to peer solution that is also greatly scalable up to 20 nodes and very lightweight. It supports a hierarchical structure in order to form federations of clusters if our use case demands more than 20 nodes. It has been used in relevant works [69] and it has been proven as a valid candidate for an edge data storage system. MinIO uses object storage over block storage so it is in fact a combination of the two systems, preserving the lightweight distributed nature of block storage while providing the plethora of metadata and easy usage of the object storage.

2.3.3.4 OPEN RESEARCH ISSUES

The current literature in the field of Edge computing reveals a number of open research issues that should be explored over the proposed software solutions. The most referenced such issue is the optimization of small data packets that tend to add considerable overheads to almost all edge storage solutions [70]–[72]. This can be achieved with simple statistical tools that reveal relations between the small data packets, allowing us to bundle them together, similar to the shopping carts logic. It can also be achieved using machine learning algorithms in order to predict which packets are more likely to be used at the next time frame in order to bundle them together. These machine learning mechanisms can take into consideration more complex parameters like user or application profiling. To our knowledge none of the explored software solutions are providing such an optimization mechanism.
Another issue worthy of exploration is an intelligent admission control that enables us to identify the correct time frame that data should be pre-fetched to the edge [73]. When considering the "correct time" we should take into account the network bandwidth and activity as well as the resource availability of the involved edge devices and possibly their mobility and reliability. It is a complex problem that should be tackled using machine learning and predictive analytics, aiming at having a more concrete prediction model, optimizing the off-loading process by preventing bottlenecks and violations on QoS and QoE expectations of the platform. This admission control is on a pretty basic level, based on ping delays and system availability monitoring, on most software solutions and not supported on some of them.
One final issue is the cache optimization. Most of the systems under examination support a caching mechanism either directly or over plugins and third party solutions. The systems supporting a master-slave architecture like OpenStack or HDFS offer this caching on master (or gateway) level storing some of the recently accessed data on the master node for faster retrieval. Other systems that are operating using decentralized architectures like MinIO are performing caching on the nodes using system cache solutions like memcached. Of course these solutions are not optimized for edge applications and they have no use case specific customization or optimization options. Machine learning could also help in this case by predicting the cache violations that an eviction would create, allowing us to evict the data packet least probable to be accessed in the near future.

2.3.3.5 CONCLUSIONS AND HIGHLIGHTS

It is apparent that the two most viable candidates are OpenStack, due to its ripe state and popularity, and MinIO, due to its decentralized and lightweight nature that makes it the state of the art in edge storage applications. These systems have flaws that need to be addressed. As mentioned OpenStack is resource demanding and its hierarchical nature makes it hard to use in edge networks that are usually unreliable and low on resources. This means that heavy modifications need to be performed on the OpenStack components in order to make it appropriate for edge deployment. On the other hand, MinIO is a more recent framework that takes into consideration the low resource availability and unreliability of edge devices, providing a lightweight and decentralized solution. Nevertheless, it is a new addition to the field and mostly untested in real world scenarios. Therefore, extensive testing needs to be performed on it in order to isolate potential weaknesses.

2.4 Pooling Edge resources

The objectives and outcomes for this Research Topic are related to the work performed in Task 3.5.

2.4.1 Objectives

2.4.1.1 EDGE SERVICES ORCHESTRATION

Edge services orchestration aims at providing functionalities for configuring, provisioning, coordinating and managing the execution of services running on Edge resources. The typical basic unit of deployment for services is containers, but Unikernels can also be used, to reduce startup delays.

2.4.1.2 USE OF BLOCKCHAIN FOR SECURING MULTI-PARTY COOPERATION

In the resource federation a secure interaction among partners should be guaranteed. Blockchain technology can be used for implementing Smart Contracts with Edge resource providers, but also to support revenue reconciliation.

2.4.2 Outcome

The ACCORDION research work on this topic will:

- Cope with limitations (e.g. of availability) that affect resources, but also the orchestrator itself
- Extend the orchestrator to handle unikernels along with containers
- Support the identified resource federation model
- Identify threats in resource federation (in collaboration with Task 4.4)
- Investigate how to use distributed ledger technologies to increase security

2.4.3 State-of-the-art

All the following content is related to the first of the two objectives stated above (Edge services orchestration), the state-of-the-art for the second objective (Use of blockchain for securing multi-party cooperation) has not been researched yet and it will be part of Year 2 work.

2.4.3.1 EDGE SERVICES ORCHESTRATION

Orchestration is about the automated management and coordination of applications and services. A common DevOps lifecycle starts with provisioning and configuration of the needed infrastructure, continues with application deployment and then orchestration keeps the application running smoothly and efficiently, scaling it and coping with failures.
The basic units of execution that support applications are continually shrinking, from VMs to containers down to unikernels and serverless functions. Orchestration could be applied to all of them. Currently the most used and extended tool for container orchestration is Kubernetes[6], an open source supported by the Cloud Native Computing Foundation[7]. For unikernel orchestration different solutions have been proposed, from kubernetes extensions [74] to the improvement of existing Virtual Infrastructure Managers [75]. For serverless functions orchestration, beside the solutions from the

[6] https://kubernetes.io/

[7] https://www.cncf.io/

three major FaaS providers [76], there are multiple open source solutions: OpenWhisk[8], OpenFaaS[9], Knative[10] and Kubeless[11].
Edge computing, as deeply analysed by Vaquero et al. [77], poses a number of additional challenges to orchestration with respect to those already impacting cloud orchestration [78]: volatility, heterogeneity, scale and others. Volatility is inherent to edge resources, it impacts services' availability, requires dynamic discovery to exploit new resources as soon as they are available, and quickly makes obsolete the status information of all resources. Heterogeneity is also common and higher in the Edge, which is populated by different types of resources, each with different access methods and protocols, potentially managed by different administrative domains. Another challenge results from the scale of the Edge: it is composed by a high number of small resources, and this, coupled with the ever-smaller execution units of the applications, makes it difficult for the orchestrator to take optimal decisions.
Some techniques to cope with the above challenges have been experimented for both edge and fog orchestration. One technique, called late calibration by Wen et al. [79], deploys the first solution found, even if suboptimal, and then improves it with further calculation and updated information to possibly migrate/redeploy part of the application. Vaquero et al. [77] reviews similar techniques, labelling them as Eventually Consistent/Probabilistic Orchestration, and notes that most current orchestration frameworks employ similar incremental techniques to progressively take the system closer to the desired state. This declarative paradigm is the same on which many current DevOps tools are based (e.g. Terraform[12] or Kubernetes[13]), and the base idea for this technique, asymptotic configuration, dates back to 1998 [80].
Another technique is P2P/Agent-based Orchestration: independent agents, each controlling a set of resources, recognize resources based on predefined ontologies, and execute deployment requests by negotiating among them [81]. Distributed orchestration is around since more than 15 years [82], but has not been so successful, as noted by Vaquero et al. [77].
Recently Machine Learning (ML) techniques have been researched with the aim of applying them to different orchestration-related tasks. The difficulty in this case is to find enough quality data to train a model complex enough to cover all orchestration activities, therefore current research is experimenting the combination of simpler models [83].

2.4.3.2 WHICH OPEN SOURCE BASELINE FOR THE EDGE MINICLOUD VIM?

The initial idea for implementing the Edge Minicloud VIM is to use Kubernetes (K8s) and extend it to handle the additional resources and to deploy also both VMs and unikernels. K8s extension is done using Custom Resource Definitions (CRD), Custom Controllers and Operators. The first step of this search is therefore to understand which lightweight K8s solution supports extensibility.

[8] https://openwhisk.apache.org/

[9] https://www.openfaas.com/

[10] https://knative.dev/

[11] https://kubeless.io/

[12] https://www.terraform.io/

[13] https://kubernetes.io/

Rancher's web site[14] says that K3s supports deploying Helm charts through a specific CRD, hence K3s supports CRD. Other lightweight Kubernetes are: KubeEdge, minikube, kind, K3os, and MicroK8s. KubeEdge should support CRDs, as it uses them to support device management [84]. MicroK8s on its home page[15] prides itself to be a lightweight K8s but not a subset: hence it should support CRDs.
The blog page [85] compares MicroK8s and Minikube. MicroK8s is relatively new but very easy to install and upgrade, its major problem is that its installer runs only on Ubuntu Linux, which may not be the best Linux distribution for an Edge environment, even if it should be noted that Ubuntu runs also on Raspberry PI [86]. There are some lightweight Ubuntu-based distributions: Zorin, Xubuntu, Linux Lite, LXLE, WattOS, Ubuntu Minimal; but all of them come with a complete GUI and are therefore still too heavy. The most lightweight Linux distributions, Tiny Core Linux [87] and Alpine [88], have VM images weighting around 100MB or even less.
Another possible approach to this search would be to look for solutions already extending K8s to run both containers and VMs. Two approaches that manage both VMs and containers are Kata Containers[16] and KubeVirt[17].
Kata Containers is an open source container runtime, building lightweight virtual machines that seamlessly plug into the containers ecosystem. As indicated by an article comparing Kata Containers and KubeVirt [89], Kata's approach is radically different from KubeVirt's one: KubeVirt uses CRD and runs a VM into a pod, Kata (currently at v1.10.5) uses the Container Runtime Interface (CRI) and runs containers as light VMs directly on (Intel's) HW. Another solution is Virtlet[18], indicated as "a Kubernetes runtime server which allows you to run VM workloads, based on QCOW2 images". Virtlet has full support for Kubernetes networking and multiple CNI implementations, such as Calico, Weave and Flannel. A further solution, from the same Rancher that developed K3s, is RancherVM[19]. RancherVM extends Kubernetes using CRDs to run KVM images as Kubernetes pods. They say: "A VM pod looks and feels like a regular pod. Inside of each VM pod, however, is a container running a virtual machine instance. You can package any QEMU/KVM image as a Docker image, distribute it using any Docker registry such as DockerHub, and run it on RancherVM". An additional open source framework that can manage both containers and Virtual Machines is EVE[20], which is a stage 2 project of LF Edge[21], part of the Linux Foundation. EVE (Edge Virtualization Engine) targets IoT applications and "aims to provide a flexible foundation for IoT edge deployments with choice of any hardware, application and cloud". Several hardware platforms are currently supported, all based on either Intel or ARM architectures, including Raspberry PI. EVE requires hardware-assisted virtualization and needs direct access to and control of the underlying hardware resources.

[14] https://rancher.com/docs/k3s/latest/en/helm/

[15] https://microk8s.io/

[16] https://katacontainers.io/

[17] https://kubevirt.io/

[18] https://github.com/Mirantis/virtlet

[19] https://github.com/rancher/vm

[20] https://www.lfedge.org/projects/eve/

[21] https://www.lfedge.org/

Another VIM requirement is that it should be compliant with cloud-based established standards such as OCCI and CDMI. One ACCORDION KPI (KPI-O1-E1-1) even puts the extension of OCCI Infrastructure model specification as a target. OCCI[22] is a set of specifications from Open Grid Forum[23]; the specification includes a core model, a protocol, a REST-based API and the support for adding extensions. It was originally initiated to create a remote management API for IaaS model-based Services, for deployment, autonomic scaling and monitoring; then it evolved into a flexible and extensible API, suitable not only for IaaS models, but for PaaS and SaaS as well. Several OCCI implementations and tools are available. Further information about OCCI can be found in section 2.1.3.2 of this document.

An OCCI extension for containers would be useful to ACCORDION, so the next question to research is: has OCCI already been extended for containers? Zalila et al [90] propose OCCIware, a framework to design, validate, generate, implement, deploy, execute, and manage resources with OCCI. Their proposal includes OCCIware Studio, a model-driven tool chain for OCCI, based on the OCCIware Metamodel that defines the OCCI static semantics. The framework also includes OCCIware Runtime, a generic OCCI runtime implementation targeting all the cloud service models (IaaS, PaaS, and SaaS). OCCIware has been successfully applied to several use cases and domains, such as virtual infrastructure (with the development of a VMware connector), cloud simulation (to simulate a configuration with CloudSim), Docker containers (to both execute a configuration and obtain a model from a running one), Google cloud and even mobile robots. Specific papers are referenced for each use case. Paraiso et al [91] report about the use case applying OCCIware to Docker containers; they propose an approach to model Docker containers and ensure their deployability and management. The proposed system provides both model-based deployment and runtime-based model generation.
Another interesting question is: what about OCCI support in the existing open source VIM implementations? OpenVIM[24] is a light NFV VIM implementation supporting Enhanced Platform Awareness (EPA) features. EPA is a methodology aiming at improving VIMs with a greater awareness of the capabilities of the platforms they control. EPA features have been proposed for both OpenStack [92] and Kubernetes [93]. OpenVIM has a REST API that allows operating on the following entities: Tenant, Image, Flavor, Server (instance of a VM), Network, Ports, Hosts. Documentation for the OpenVIM API [94] dosn't mention OCCI, but says that it's similar to the Openstack v2 API, and OCCI adapters for OpenStack do exist, so they can be extended to obtain an OpenVIM OCCI adapter. The last version of OpenVIM is part of Open Source MANO (OSM)[25]. OSM components are implemented as (14) Docker containers, but can be layered on K8s, too. OSM supports different kinds of VIM: OpenVIM, OpenStack, VMware vCloud Director, AWS, Azure, Eclipse fog05. The most interesting of these VIMs is Eclipse fog05 [95], as it's able to deploy both containers and unikernels, along with Virtual Machines. Eclipse fog05 was originally written in Python, but has been rewritten in OCaml to make its agent a (MirageOS) unikernel.

2.4.3.3 CONCLUSIONS AND HIGHLIGHTS

[22] https://occi-wg.org/

[23] https://www.ogf.org/

[24] https://github.com/nfvlabs/openvim

[25] https://osm.etsi.org/

From the state-of-the-art research on Edge services orchestration it appears that the best techniques to cope with the challenges posed by the Edge are those incremental techniques that progressively take the system closer to the desired state. Incremental techniques are typically based on a machine-readable description of the desired state of the system, usually in text-based configuration files, and lead to the so called "declarative paradigm" that is the same on which many current DevOps tools, such as Terraform and Kubernetes, are based.
About the search for an open source baseline for the ACCORDION Edge Minicloud VIM, a final decision has not been taken yet, but some of the solutions indicated above (fog05, EVE, Kata Containers, KubeVirt and RancherVM) have been evaluated and the evaluation result, along with a detailed rationale for the selection, will be reported in the next WP3 deliverable D3.1, due at M14.

2.5 AI-based network orchestration

The objectives and outcomes for this Research Topic are related to the work performed in Task 4.2.

2.5.1 Objectives

2.5.1.1 AI-BASED ORCHESTRATION

We will design and develop a multi-domain AI-based orchestration framework of the network elements by ensuring reliability and latency and managing DL models. This framework aims to optimize allocation according with current queries and Optimize dispatching of queries according with current allocation using different AI techniques that learn the best policy for dispatching the queries.

2.5.1.2 SELF-CONFIGURATION AND SELF-OPTIMIZATION OF NETWORK SLICES

Self-Optimization of network slices can take place with Deep Reinforcement Learning. A prominent difficulty in Deep Learning models is to propose the capacity of neural networks in terms of number of layers, number of neurons, etc. We plan to adapt hyperparameter optimization techniques to Deep Reinforcement Learning to find close to optimal solutions. We will investigate also about the utilization of multi-agent reinforcement learning in case the decision should be taken by multiple actors.

2.5.2 Outcome

The ACCORDION research work on this topic will study the following hyperparameter optimization approaches in order to conclude and adapt the most appropriate:

- Evolutionary optimization
- Gradient-Based optimization
- Bayesian optimization

2.5.3 State-of-the-art

The main role of task 4.2 is to design and develop a multi-domain AI-based orchestration framework of the network elements by ensuring reliability and latency, enabling slice tenants to define blueprints for their VR/AR ready slices, and facilitating the efficient life cycle management of the slices with the aim of rapid slice creation and activation. Machine learning (ML) is recently gaining more considerable attention from both the academia and industry due to its ability to provide smart and scalable solutions. This section summarizes the state-of-the-art of the ML techniques solutions that have been devised to enable automated orchestration and intelligent management operations: they are the basis for both AI-based orchestration and the self-configuration and self-optimization of network slices. These objectives will be studied in the next years and reported in the next deliverables.

2.5.3.1 MACHINE LEARNING TECHNIQUES

Machine learning techniques, such as supervised learning, unsupervised learning, and reinforcement learning, have been widely adopted by network, cloud, and telecom providers. These techniques provide the ability to learn with time from the environment and offer a self-optimized configuration that can adapt and cohabit according to the network and industrial vertical states. The latter is continuously and dynamically changing due to heterogeneity in vertical demands, mobility patterns, and dynamic workload. Moreover, these techniques are able to provide fast settings while making an abstraction on the environment, which facilitates the integration of the same solution in different scenarios. The abstraction of the complex environment helps to prevent the use of many assumptions in the modeling, and thus providing inaccurate results.

Supervised Learning: In supervised learning algorithms, the inner relations of the data may not be known, but the output of the model is. This technique requires labeled training data that is used for training the model. The latter is mainly used for either classification or regression problems. Moreover, the training of the algorithms mostly requires three sets of data. The most important part of the data is used for training, while the two remaining pieces are used to test and evaluate the derived model.

Unsupervised Learning: Unlike the supervised learning approach, the data is unlabeled in these techniques. Relevant types of models try to find a correlation between the input data and classify it into different clusters (i.e., clustering).

Reinforcement Learning: The reinforcement learning (RL) technique has been widely used in the literature for self-optimizing the continuously and dynamically changing network. Reinforcement learning belongs to the same family of the Markov decision process (MDP). In contrast to MDP, which is a model-based approach (i.e., Transition probability), RL is a model-free approach. RL adopts a unique model training method that is based on trial and error and reward functions. The agent in RL periodically makes decisions, observes the environment, and then adjusts the next action policies for achieving optimal configuration. RL summarizes the environment and actions in a Q-table. Unfortunately, the Q-table can provide an optimal strategy in a complicated situation that has a large number of states and actions. A new paradigm, dubbed Deep Reinforcement Learning (DRL), has been recently introduced to overcome the limitations of RL [96]. DRL leverages Deep Learning (DL) [97] for presenting the Q-table as a function by leveraging the strength of neuron networks. DRL is mainly classified into two classes: *i)* Value-based approach, such as Deep Q-Networks (DQN), Double DQN (DDQN), and Dueling-DDQN Algorithms; *ii)* Policy-based approach, such as Advantage Actor Critic (A2C), Asynchronous Advantage Actor Critic (A3C), Deep Deterministic Policy Gradient (DDPG), and Proximal Policy Optimization (PPO) algorithms. The policy-based method uses the actor-critic technique for providing different policies [98].

2.5.3.2 DISTRIBUTED ARTIFICIAL INTELLIGENCE AND MULTI-AGENT SYSTEM

Distributed Artificial Intelligence (DAI) is seen as a sub-field of Artificial Intelligence. It solves problems related to coordination, concurrency, and decision-making. Such systems are composed of entities called Agents, computational entities (e.g., a software program, or a robot) that can perceive their environment and can act accordingly. Traditional AI models focus on solving problems by searching through a space of potential solutions. With ML algorithms, the goal changes to learning a target function F, given a variable X that maps its output Y. To find this function, the underlying ML algorithm uses a dataset containing instances with their respective classes (in case of a supervised learning), and from it, derives the function, or looks for a correlation between attributes to regroup them into similar categories (in case of unsupervised learning). Most of the application is done locally whereby a data source is used to feed the learning component holding the model. The success of the model hinges on the learning algorithm, the attributes selection and more importantly on the data source. The latter plays a major role: according to its volume and diversity, it is possible to extract interesting behaviors that are hard to discover with a regular data analysis.
A multi-agent system (MAS) is a complex system that consists of several agents, designed to carry out specific tasks. These agents cooperate among themselves to achieve a common objective. In such a system, all participating agents and their coordination are usually defined by design to accomplish the task. An agent refers to a computational entity (e.g., a software program and a robot) that has the ability to perceive its environment and can act accordingly. For this project, agents can be divided into two categories:

- **Active Agents:** do not only learn from their surroundings but have also the ability to impact their respective environments.
- **Passive Agents:** act as passive watchers, acquiring knowledge from their respective environments without necessarily affecting them.

2.5.3.3 DISTRIBUTED FEDERATED LEARNING

Federated learning (FL) is a machine learning technique that enables the distribution of the learning process among many parties (e.g., mobile devices, edges, or clouds). These parties collaborate with a central orchestration server (e.g., service provider) while keeping the training data decentralized. In this ML technique, the learning agents can collaboratively train a global learning model without the need to share their local datasets. The central agent will be able to learn from the different agents, and then accordingly helps the other agents for enhancing their local models, and thus perceiving improved learning performance. This strategy has many folds:

I. Decentralized computation that leads to achieving the optimal learning rate in an optimal time;
II. Sharing the gained knowledge (i.e., trained models) with the new members;
III. Increasing data privacy (i.e., images and videos) by treating generated data close to their source origination.

Authors in [99] suggest MOCHA approach that aims to reduce the communication cost and ensure fault tolerance. Meanwhile, authors in [100] have proposed supervised federated learning, dubbed FedProx, that aims to optimize the local model to fit the local datasets from one side. In contrast, the global model is optimized to perform well on distributed datasets by aggregating the local learning parameters. However, these conventional federated learning is mainly focused on a single learning task (i.e., the global model) with non-i.i.d (non-independent and non-identically distributed) datasets. Moreover, the global model can be easily biased by the agents that have more massive datasets with too many updates [100]. From another side, the convergence (i.e., overfitting or underfitting) of the

global model can be hurt by the generalization and specialization at each sub-model. To overcome the heterogeneity of the underlying data distribution at different agents, authors in [101] suggest Model-Agnostic Meta-Learning (MAML) framework that leverages federated averaging technique for providing a more personalized model for each agent, and hence offers better model convergence. Unfortunately, the MAML framework lacks cohesion relations between the generalization and the personalization.

2.5.3.4 MLF: META LEARNING FRAMEWORK

FL aims to solve the prediction for the same task in a distributed multi-agent environment by leveraging multiple datasets. However, the delivered global model from FL is adequate to address the same task due to its rigid design, which limits the practicality of the model in complex scenarios or unseen data. To mitigate these issues, meta-learning framework (MLF) [102] has been suggested that enables the generalization from a broad training data of similar tasks. The provided model by MLF is designed to be easy to fine-tune by making a slight modification on the gradient descent method for enabling the generalization in the prediction.

2.5.3.5 MULTI-AGENT REINFORCEMENT LEARNING

In the literature, the model-based problem description, i.e., Markov Decision Process (MDP), has been extended from a single agent to a multi-agent system by forming a stochastic game that enables extensive and exciting use-cases. Due to the advances achieved in model-free approach RL techniques [103], there is a recent impetus towards enabling a multi-agent system RL by providing a new concept called multi-agent reinforcement learning (MARL) [104]. This approach would create more benefits for a distributed system, such as autonomous driving, network management and orchestration automation, where more than one agent should collaborate to achieve the desired objectives.
MARL algorithms can mainly be classified into three main classes: *i)* A fully cooperative approach; *ii)* A fully competitive approach; *iii)* A hybrid approach. While in the former, the agents collaborate to optimize a single objective, the agents in the second approach compete to enhance their benefits similar to a zero-sum game where mixed-strategy Nash equilibrium is sought. Meanwhile, the latter method, both previous approaches are mixed to optimize the global objective besides the objectives of each agent. This approach is similar to the general-sum game, where the dominant strategy and Nash equilibrium are sought in pure- and mixed-strategies. MARL will mainly leverage various theories varying from optimization theory, dynamic programming, and game theory to decentralized control.

2.5.3.6 CONCLUSIONS AND HIGHLIGHTS

Several ML techniques have been studied, which can be applied to multi-agent systems. Results in Distributed Artificial Intelligence and Federated Learning promise to solve problems related to coordination, concurrency, and decision-making to obtain AI-based orchestration. In ACCORDION we will adapt hyperparameter optimization techniques to Deep Reinforcement Learning to find close to optimal solutions and will investigate the application of multi-agent reinforcement learning to cases where the decision should be taken by multiple actors.

2.6 Resilience policies & mechanisms over heterogeneous edge resources

The objectives and outcomes for this Research Topic are related to the work performed in Task 4.3.

Resilience policies are a vital part of the edge/cloud continuum orchestrator. The FT mechanism will cover two main functionalities the Mobile aware FT mechanism and the Resource utilization aware FT mechanism.
The Mobile aware FT mechanism will use a next time position predictor that estimates the geo-location (lat, lon) of a mobile entity in an area of interest in the next time frames. If the mobile entity will be out of the coverage of the FOG infrastructure or there will be a high distribution of mobile entities around a specific Point of interest, then the FT mechanism will predict potential QoS deterioration and trigger the proactive measures. The proactive measure will be an intelligent replication that meets the geographically needs of the FOG environment.
The Resource utilization aware FT mechanism will monitor the resources usage that runs on each one of the hybrid edge miniclouds in order to reveal at run-time, potential QoS deterioration. In case that the deployed resources cannot satisfy the increasing amount of resources usage on the next predicted time steps, then the middleware will trigger mitigation policies such as hot- and cold-migration between neighboring hybrid edge miniclouds and workload processors.

2.6.1 Objectives

An ML prediction model for run-time QoS deterioration will be implemented in order to ensure high-availability and fault-tolerance transparent workload mobility. A Time Series approach will be studied in order to leverage the trend of resource usage sequences. An additional prediction model will be implemented in order to capture the mobility behaviours of the users that cause faults i.e. when many users are gathered and offload their task to a limited resource edge node. Anomaly detection models following an autoencoder, clustering or classification approach can also be analysed. We will put special emphasis in deep learning solutions with hyperparameter optimizations.

2.6.2 Outcome

The ACCORDION research work on this topic will propose, evaluate and develop an accurate real time Resilience policies model extending and adapting Time Series models and Deep Learning models with Hyperparameter optimization techniques.
The AI Fault Tolerance model aims to predict potential QoS deterioration and selects the VMs that need to be replicated, the number of replicas, and estimate the locations where the replicas should be placed. The Fault Tolerance mechanism will use feature engineering techniques including the interpolation with monitoring data, data fusion, feature selection and extraction.
The first see of experiments will be with traditional machine learning methods such as Bayesian models, SVM, Instant - based (K-NN), Decision Trees, Random Forests. Next Special emphasis will be given to Deep Learning (DL) models and specifically to leverage metaheuristic techniques for hyperparameter optimisation. We plan to conduct research using genetic algorithms, evolutionary strategy and swarm intelligence. In addition we will use different types of layers such as simple recurrent NN, Long Short Term Memory (LSTM) and layers with Gated Recurrent Units (GRU). In case that we will deploy the Fault tolerance model in new environments with not available training datasets we will also employ Deep Reinforcement Learning models.

2.6.3 State-of-the-art

Deep Learning (DL) models are the state of the art in machine learning and prediction mechanisms. However DL has not been studied enough as a basis for proactive fault tolerance. So we aim to tailor the DL methodology in order to address the Fault Tolerance challenges. In our research we aim to analyse and make experiments on how we can employ DL in order to decrease the mean time to failure and increase the availability of the edge infrastructures. The hyperparameter optimization methodology will help us to conclude to a close-to-optimal neural network topology for the QoS deterioration and the user mobility.

2.6.3.1 REACTIVE FT STRATEGIES:

Some typical reactive FT strategies are: Replication, Resubmission, Retry, and Checkpointing. Replication [106] means that a task is replicated to more than one executing nodes. Replication increases both the chance that the task will be completed correctly and the reliability of the system, while it overuses the resources for redundancy.
Resubmission [107] means that the failed task will be executed again from the beginning in the same node or in a different node but it has the disadvantage of increasing the response time.
Retry [108] means that the failed task is executed again only from the same processing node. This approach has the benefit that it does not bind more resources but it has the disadvantage that increase the response time in the same way as the resubmission approach.
Check pointing approaches [109] takes snapshots during the task execution and if a task failed then it is re-executed from the last checkpoint. The advantage of this approach is that the task does not start from the beginning but it needs a storage space to keep the snapshots.
A hybrid reactive mechanism that combines both replication and resubmission techniques is proposed in the study [110]. This mechanism decides the most appropriate node to offload a task using a reliability assessment method instead of replicating the task to all available nodes. The highest reliability processing node is defined by its adaptability, minimum reliability, maximum reliability and a resubmission factor. The reliability for an execution node is increased when a task status is succeed and decreased when status is failed.

2.6.3.2 PROACTIVE FT MECHANISM

Proactive FT mechanisms predict a potential fault before it occurs and avoids its influence on the task [111]. The prediction can leverage data collected during the processing operations and trigger the follow-up operational actions. The proactive models can use a preemptive migration technique [112] that after a processing node fault prediction, the task shifts smoothly and transparent to another processing node. An interesting study proposes to support the training process of the proactive FT model with a reactive FT mechanism [113]. A related but different approach uses time checks which are different from checkpoints. Time checks define the maximum normal time duration which a task need to complete its job. If this time passes then actions are taken. This approach is not efficient because of the wasting time in normal time duration execution and for calling another machine.
One prominent proactive FT mechanism that optimises the cost of the distributed infrastructures is proposed in the study from AbdElfattah et al. [114]. This study proposes an autoregressive integrated moving average (ARIMA) model to estimate the workload and afterwards to deploy in the suitable geolocation positions a suitable number of replicas. The ARIMA model leverage historical data of resource allocation and changes in workload patterns.

2.6.3.3 ADDITIONAL FAULT TOLERANCE APPROACHES

FT mechanisms can be efficiently combined with load balancing for a timely provision of services [115] and resource utilization for real-time applications [116]. The continuous monitoring of the

workflow and the resources can help us to predict resource shortage which leads to faults [117]. FT can be a main functionality in the context of scheduling problems. An efficient heuristic approach is based on the Particle Swarm Optimization [118] which is tailored to address the task assignment issue with a rescheduling mechanism to meet the deadline constraints of real-time tasks. Using optimization theory [119] we can directly estimate close to optimal provisioning solutions which are also fault tolerant and incorporate the FT as constraint in an objective function [120] of task offloading. The solutions that minimise the objective function also satisfy the resilience of the system.
Machine Learning models such as K-Means, Decision Tree and K-Nearest-Neighbors can be used to timely identify errors and faults [121] for proactive FT. In addition, FT mechanisms can be implemented using an anomaly detection mechanism [122]. In this approach a clustering algorithm creates a model of the normal behaviours by monitoring the components and when an anomaly is detected a premeditation system will revert the component into the normal state. The availability and reliability of the resources in a distributed computing environment can also be predicted by a multi-state semi-Markov process [123].
An architecture design of edge infrastructures using a fat tree [124] structure can provide a reliable and fault tolerant mean to store, transfer and process data and workload. Resilience as a Service is a paradigm that introduced by the RADIC architecture (Redundant Array of Distributed Independent Fault Tolerance Controllers) [125] and can provide distributed computing with a highly available and scalable FT service. A FT method that exploits the cloud-fog-mist-dew architecture by replicating and redirecting the application in the best possible level in the hierarchy has also been proposed [126]. This approach makes the replications based on the processing power and distance from the end IoT device.
Replication with canceling is a known expensive implementation of FT. This method initiates multiple concurrent replicas and uses the first successful result. An improved approach of the replication with canceling [127] has been proposed that estimates the possible inefficiency of remote services. It allows replications only when static routing fails, and requires one or more replicas of the same request to be completed using a majority rule decision. A different replication approach tries to minimise the workflow cost with deadline constraints using a FT scheduling algorithm [128]. An elastic resource provisioning mechanism [129] can achieve FT using the primary-backup model. Primary-backup model is widely used to realize FT by duplicating a task into two copies - a primary copy and a backup copy but this redundancy incurs extra overhead. To address this issue, it has been proposed a real-time FT scheduling algorithm with rearrangement (RFTR) [130] which dynamically rearranges the execution orders of tasks.
A Replication mechanism can leverage timing bounds [131] that captures the relation between traffic/service parameters and loss-tolerance/latency requirements in order to schedule replication actions and meet needed levels of assurance. A reliability assessment process [132] can evaluate the VMs and afterwards to replicate the workload of the VMs with low reliability. The majority of reliability assessments mechanisms rely only on the ratio of successfully completed tasks to total requested tasks. A better approach that has been proposed is SaRa [133] which use a probabilistic FT model to estimate the reliability of task-level behaviors (e.g. success or fail) and task characteristics (e.g. priority of a task).

2.6.3.4 FAULT TOLERANCE IN DISTRIBUTED COMPUTING FRAMEWORKS

FT mechanisms are part of the core architecture in many available frameworks for managing cloud, fog, storage, and other types of infrastructure resources. Openstack [58] which is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter use HAProxy, database replication such as Galera, Keepalived, Pacemaker, and Corosync

[134]. Cloudstack, also used for creating, managing, and deploying infrastructure cloud services, achieves FT by grouping resources into multiple geographic regions [135]. Kubernetes (K8s), one of the most popular open-source systems for automating deployment, scaling, and management of containerized applications, achieves FT by using multiples Control Planes [136].
OpenShift is an enterprise-ready Kubernetes container platform with full-stack automated operations to manage hybrid cloud and multicloud deployments. In OpenShift the FT is supported by the use of multiple masters using native HA [137]. Native HA provides a high availability solution and consists of two nodes in which one node assumes the role of the active node and the other node assumes the role of the standby node [138]. Docker, one of the most popular platforms for building applications based on containers, recommends to implement an odd number of nodes for FT [139].
FT in Mesos [140] involves three components: master, slave and framework. In case of a slave failure the master will inform the framework and the framework will reschedule the tasks running to other healthy slaves. If the framework fails then the tasks will continue to being executed and when it implements failover it will get information about the status of all its tasks. In case of master failure, Mesos uses a leader election to select a high available master.

2.6.3.5 MOBILE AWARE FAULT TOLERANT MOBILITY

Movement analysis for a human, mobile object or any kind of moving entity provides highly valuable information in the context of fog computing and FT. Specifically, in the FT we need to predict if a mobile entity will be out of the coverage of the Edge/Fog infrastructure or there will be a high distribution of mobile entities around a specific point of interest. The analysis, the prediction and the knowledge extraction of movement behaviors belong in the domains of spatio-temporal data mining [141] and trajectory analysis [142].
The techniques to extract knowledge from trajectories can be grouped in the following three machine learning approaches. Firstly, in the classification approach [143] where we assign new coming observations into predefined classes based on a labeled training dataset. Secondly, in the clustering approach [144], where the trajectories are grouped together based on similar features. Finally, in the pattern recognition approach [145] where regularities of movement behaviors are recognized. The framework of knowledge extraction from trajectories can exploit additional machine learning techniques to refine and prepare data [146] such as feature engineering, data normalization and noise removal.
In the FT component, we focus on the problem of predicting the next position of the movement of an object based on its previous positions. This problem has been addressed in the literature with methods such as Markov Chains [147] and well-established machine learning models [148]. Our approach is different from the previous because we use an LSTM RNN to forecast the next position and transfer learning to improve the training process.
LSTM (Long Short Term Memory) is a powerful DL predictive model used effectively in the domain of Times Series and Natural Language Processing [149]. While time series approach has been used by many spatio-temporal applications [150], to our best of our knowledge LSTM RNN have not been used in predicting the next position of a moving object.
LSTM is a special kind of Recurrent Neural Networks (RNN). RNN have been used in the domain of trajectory analysis for the needs of point-based classification [151] and have presented very high predictive outcomes. LSTM is more advanced than RNN because they have special memory units in addition to standard neurons. The memory units keep information from a sequence of data for long periods of time. DL models involve the synaptic weights and biases, which are the learnable parameters. These parameters are estimated using an optimization process which include back propagation and a gradient descent method such as RMSprop, AdaDelta, and Adagrad [152].

The capacity of the RNN, also known as the hyper-parameters, is the number of layers and the number of neurons that each layer has. The capacity is often selected based on domain-expertise or in a trial and error method. In the FT component, we will use a genetic algorithm for optimum or a close to optimum hyper-parameter selection. Genetic algorithms have been used in the past for RNN architecture that contain only sequential layers [153]. In our component, we will apply them with LSTM layers to explore the multi-dimensional space of hyper-parameters of mobile aware FT with a smart and efficient way.

2.6.3.6 RESOURCE UTILIZATION AND FAULT TOLERANCE

The main approaches of modeling the resource utilization include analytical models and machine learning models. Analytical models often use queuing theory [154] to model computing infrastructures that operate under steady request arrival rate, average service time and resource utilization with probabilistic behavior. In addition, new models have been proposed which enhance the queuing models focusing on parallelizable tasks [155].
Machine learning models predict the resource utilization using a data-driven approach. They employ a large number of methods and techniques to discover patterns of resource usage in various circumstances from historical data. An ensemble approach [156] has been proposed integrating a multi-regression model feature selection mechanism. Sequencing resource observations is commonly exploited by time series approaches like Autoregressive Integrated Moving Average models [157]. Furthermore, a research close to our proposed model predicts CPU utilization using evolutionary neural networks [158].
The FT Resource utilization aware component differentiates from the above-mentioned models by combining the time series approach with deep learning models. Then it makes a smart search using genetic algorithms to identify an optimal LSTM-RNN that can forecast the resource utilization of CPUs, Ram, Disk I/O, and Network as a Multi-Output Regression Model in a unified way. The genetic algorithm excels the evolutionary approach in the exploration of hypothesis space as the later use only mutation as reproduction strategy while the former use both crossover and mutation. The FT Resource utilization aware approach is designed to make intelligent replication timely and with high accuracy leveraging state of the art methods.

2.6.3.7 CONCLUSIONS AND HIGHLIGHTS

Many Fault tolerance mechanisms are proposed in the literature. However, edge computing with geographical characteristics brings in new challenges that have not been so far addressed efficiently. In addition, Deep Learning models provide accurate techniques that have not been tailored for the fault tolerance. We propose that the hyper parameter optimization approach will adapt the deep learning topologies efficiently for the fault tolerance needs. Last but not least, the resource usage observations and the user mobility have time series characteristics. To sum up our conclusions, we propose an innovative time series deep learning model with hyper parameter optimization for proactive fault tolerance in edge computing infrastructures.

2.7 Techniques for secure Edge application development & deployment

The objectives and outcomes for this Research Topic are related to the work performed in Task 4.4.

2.7.1 Objectives

Security by Design and DevSecOps techniques are successful approaches to improve security in the application development lifecycle. How can they be applied to Edge development and operation? Which security requirements should be applied to resources in the discovery process, and which should be applied to communication channels to connect those resources? How to manage secrets in the Edge environment?

2.7.2 Outcome

The ACCORDION research work on this topic will:

- Identify and model threats in the Edge environment
- Investigate how to apply DevSecOps techniques and tools, also considering the limits on available resources
- Identify security requirements for computing and network Edge resources
- Secret management in Edge environments

2.7.3 State-of-the-art

The following content is related to the first two objectives stated above: threat modelling in Edge environments, and DevSecOps techniques and tools. State of the art for the other two objectives (security requirements and secret management in Edge environment) will be part of Y2 work.

2.7.3.1 THREAT MODELING IN EDGE COMPUTING ENVIRONMENT

The rapid developments of the Internet of Things (IoT) and smart mobile devices in recent years have been dramatically incentivizing the advancement of edge computing. The edge computing architecture will become an important complement to the cloud, even replacing the role of the cloud in some scenarios [159][160][161]. It serves as a key enabler for many future technologies like 5G, Internet of Things (IoT), augmented reality and vehicle-to-vehicle communications by connecting cloud computing facilities and services to the end users.

There are several security threats and attacks faced by edge computing. Distributed denial-of-service (DDoS) is one of them. It refers to a cyberattack in which attackers aim to disrupt normal services provided by one or more servers based on distributed resources such as a cluster of compromised edge devices (also known as botnet) [162]. DDoS attacks may happen when malicious edge devices communicate with the edge servers. In such attacks, the attacker takes full control of a cluster of edge devices by compromising them and then commands each device to launch a denial-of-service attack targeting the edge server, causing the shutdown of its services.

In the edge computing environment, the DDOS attacks can be classified into two types of attacks. The first type of DDOS attacks are Flooding-based attacks. This kind of attacks aims to shut down the normal service of a server based on a large amount of flooded malformed/malicious network packets and are mainly taxonomized as UDP flooding [163], ICMP flooding [164], SYN flooding [165], ping of death (PoD) [166], HTTP flooding [167], and Slowloris [168], according to the respective attack technique. The protocol-level design flaws/vulnerabilities within the network communication protocols is the root cause of flooding-based attacks. According to that, current defense solutions against this type of attacks mainly adopt a detect-filter philosophy. Since a flooding-based DDoS attack is launched mainly by sending an enormous number of malicious or malformed network packets, detecting and filtering those packets can have an effective defense. In [169], this

observation was exploited. Hu et al. proposed integrating packet filtering mechanisms into congestion control frameworks to mitigate the attacks. When a suspicious packet is identified, the packet will be dropped by the network before it arrives at the destined edge server. Moreover, Luo et al. [170] apply the identifier/location separation techniques by detecting possible DDoS packets based on packet identifiers. The proposed approach hardens the security to control machines. However, a more sophisticated attacker can easily circumvent such detection mechanisms by changing the identifiers of the packets.

The second type of DDOS attacks are Zero-day DDoS attacks which is more advanced than flooding-based DDoS, but it is more difficult to implement. In such an attack, an attacker must find an unknown vulnerability in a piece of code running on the target edge server/device. This can cause memory corruption and finally result in a service shutdown.

To defend against this type of attacks, different mechanisms were developed such ECC-memory [171] and pointer taintedness detection [172]. These methods require the availability of the original source codes. Frassetto et al. [173] provided an in-process memory isolation; Spectre and Meltdown remain threats. Dietz et al. [174] proposed lightweight isolation mechanisms on access routers. These mechanisms serve as guards before an IoT botnet virus can access real edge devices.

Another type of edge computing attack is Malware Injection Attacks in which attackers aim to inject malware, i.e., malicious codes, into edge devices or edge servers. The SQL injection [175] is one of them; this kind of attacks is targeting edge servers which is a code injection technique that destroys the back-end databases. Several methods for injecting malware into IoT devices exist. An example can be found in [176] where Cui et al. discover that in some LaserJet printers, due to the lack of the signature verification check, an attacker is able to modify any pre deployed firmware. Moreover, an attack was implemented by Ronen et al. [177] where the zigbee light link protocol was used to inject malicious firmware into IoT devices.

Authentication is critical to many services, if an attacker intends to access protected edge devices or edge servers, he will seek the methods to bypass the authentication process. One of the simple authentication-based attack is a dictionary attack. In such an attack, an attacker groups the mostly used credentials and passwords in a dictionary and try to put them to the target authentication system in order to find a possible match.

Adding one more layer of authentication can be a solution to defend against this kind of attacks. This method is known as two-factor authentication. Several second authenticators were used by well-known two-factor authentication mechanisms such as face authentication [178], fingerprints [179], authentication code via SMS messages [180]. The first two-factor authentication technique was proposed by Pinkas et al. [181], in this work a challenge was added. This challenge is infeasible to be answered by automated programs from dictionary attacks.

As threat modeling methodologies evolve, security professionals are recognizing the importance of choosing the right threat modeling methodology. Several threat-modeling methods have been developed such as CVSS which was developed by NIST [182]; OCTAVE method [183] created by the CERT Division of the Software Engineering Institute in 2003 and refined in 2005 which is a risk based strategic assessment and planning method for cybersecurity; a risk-centric threat modeling framework PASTA [184] developed in 2012 and LINDDUN threat modeling method [185] that focuses on privacy concerns and can be used for data security.

One of the oldest and most widely applied techniques is the attack trees method [186] and the most mature threat-modeling method is STRIDE invented by Loren Kohnfelder and Praerit Garg in 1999 and adopted by Microsoft in 2002.

2.7.3.2 METHODS

Nowadays the main methodology to secure a DevOps toolchain, named DevSecOps, is to insert additional security tools on each intermediate pipeline step to ensure that the current phase of the CI/CD is compliant with the known threats.
To have an idea we can think to a simple chain to build a container: we will have to choose a starting container image, add some piece of software and libraries, write a source code and push the container to the registry. The container will run in a host interacting with other containers and resources.
In this process we can analyse the *source code*, the *library* we have used to build the code, the *container image*, the *host* where the container runs and finally we can do some *penetration tests* to ensure the application is strong enough to support malicious attacks. The analysis tools are usually executed as fully automated steps in the CI/CD process.

2.7.3.3 TOOLS

Examples of the security tools for a generic CI/CD pipeline follow:

1. Static source code analysis (e.g. Sonarqube)
2. Library dependency vulnerability check (e.g OWASP Dependency Check)
3. Container Image Vulnerability Check (e.g. Anchore)
4. Container Host Analysis (e.g. Docker bench security)
5. Penetration Testing (e.g. OWASP ZED ATTACK PROXY)

2.7.3.4 CONCLUSIONS AND HIGHLIGHTS

Several cyberattacks that can target Edge computing have been described, from multiple types of Distributed denial-of-service (Flooding-based and Zero-day attacks), to Malware Injection and authentication-based attacks. Several threat modeling techniques have also been presented that can help in the assessment of risks impacting Edge environments: CVSS, OCTAVE, PASTA, LINDDUN, attack trees, and STRIDE; the last two are the most widely used and mature methods.

The research on the state-of-the-art related to DevSecOps highlighted the following elements:

- Strong emphasis is on security process automation: regardless of particular methods and tools used, security controls like code review, monitoring, testing and reporting needs to be automated as much as possible to reduce human intervention and to be seamlessly integrated in the CI/CD pipelines.
- To be effective, security must be part of the development process and pipeline from the beginning of the application creation process. This is often reflected in the terminology, where *SecDevOps* term is used instead of *DevSecOps* to emphasize the design and adoption of security controls and tools, policies, guidelines and coding standards early in the application lifecycle.
- Organizational aspects are considered fundamental, in particular the strong collaboration between security, development and operation teams.

These elements suggest that a key aspect to consider in the selection of specific tools will be the automation capability.

Security-first approach must be encouraged for ACCORDION applications: security related aspects such as coding standards, threat analysis, automated security controls and tools selection for the CI/CD pipeline etc. must be considered before the development starts and integrated as a key shared process for all the individual developers and teams involved in the application design, development, deployment and maintenance.

2.8 Privacy preserving mechanisms

The objectives and outcomes for this Research Topic are related to the work performed in Task 4.5.

2.8.1 Objectives

2.8.1.1 PRIVACY-PRESERVING MACHINE LEARNING TECHNIQUES

Several Machine Learning (ML) models will be needed on this project. In order to ensure privacy-preserving ML algorithms, we will enable multiple parties to jointly learn an accurate neural network model without sharing their input data sets, by applying a differentially-private scheme and federated learning. Then we will investigate the trade-offs between federated and pure peer-to-peer solutions, with respect to convergence, accuracy and privacy, and how much information may be unnecessarily shared during the federated or peer-to-peer learning, with respect to the performance achieved from each method.

2.8.1.2 SENSITIVE DATA LEAKAGE DETECTION AND BLOCKING

The growth of aggregated data inside the project magnifies privacy challenges to limit access to certain types of data, prevent unauthorized access (confidentiality) and protect data from being modified or corrupted without detection. We will design and implement a set of methods to detect and block potential leakage of sensitive data and explore a differentially-private scheme for ML method.

2.8.2 Outcome

The ACCORDION research work on this topic will:

1) Propose, evaluate and develop federated learning model and algorithms to improve privacy-preserving mechanisms
2) Identify access points with high vulnerabilities and block potential leakage

2.8.3 State-of-the-art

This section briefly presents existing work in both privacy-preserving machine learning (objective 1) and sensitive data leakage detection/blocking (objective 2)

2.8.3.1 PRIVACY-PRESERVING MACHINE LEARNING TECHNIQUES

Several initiatives from startups and online community efforts have been recently proposed in the space of Decentralized Machine Learning (DML). Unlike traditional ML models, DML is a (now abandoned) Blockchain-based project that enables its participants to build models in a distributed fashion, while also growing the block chain network itself [187]. Other open-source community efforts propose libraries and platforms to allow users to train ML models in a decentralized, secured and privacy-preserving fashion. For example, OpenMined [188] proposes PySyft and PyGrid, two libraries that employ multiparty computation (MPC), homomorphic encryption, Differential Privacy (DP) and Federated Learning (FL) for secure privacy-preserving and decentralized ML modeling in both mobile and desktop environments. In addition, Datafleets [189] utilizes DP, secure MPC and role-based access control to build models inside or across enterprises, and on edge computing nodes. With similar technologies, FATE (Federated AI Technology Enabler) [190] focuses on desktop deployments.

Building on the popular TensorFlow (TF) framework, TensorFlow Federated (TFF) is an open-source framework for ML and other computations on decentralized data [191]. coMind [192] proposes a custom optimizer for TF to easily train neural networks via FL. There have also been benchmark frameworks proposed like LEAF [193], for learning in FL settings, with applications including FL, multi-task learning, meta-learning, and on-device learning.

Regarding DP and FL, DP-noise can be introduced at different stages of the FL system: (i) in the data source at the client side (i.e., local-DP) [194][195][196][197], (ii) in the central server side [198], or (iii) at an intermediate stage such as edge computing nodes [199] or base stations [200], or (iv) with hybrid and hierarchical methods [201][202][203]. However, the introduction of DP-noise in the ML-pipeline affects the convergence rate of the FL-trained model.

FL can also build models in a hierarchical fashion across different network layers including (i) end-user device, (ii) ISP edge nodes, or (iii) the central server. Recent works considered the hierarchical FL case, where multiple network stages are involved in the training process [199][200][202]. Such efforts showed convergence and accuracy can be improved with proper designs under such settings.

In ACCORDION, we plan to study optimal ways to build FL models in a hierarchical fashion with built-in privacy. Our goal is to optimize the ML models such that they converge well, offer high ML performance and preserve the privacy of the data owners. For the latter, it is of utmost importance to identify and protect highly-sensitive ML layers. One promising way to protect such ML layers is to place them inside a Trusted Execution Environment (TEE). In this project, we plan to explore how best to use TEEs so as to preserve the privacy of the ML model while considering the current limitations of TEEs, in order to achieve a suitable trade-off between model privacy and cost.

2.8.3.2 SENSITIVE DATA LEAKAGE DETECTION/BLOCKING

When users are browsing the Web, they are continuously exposed to online tracking via cookies, fingerprinting and other methods such as cross-device tracking, and data sharing between 3rd-parties via cookie synchronization.

The ad-industry continuously develops new mechanisms for creating more relevant and effective ads, to deliver contextual, targeted-behavioral, and retargeted ads. However, in order to serve such highly related ads, advertisers often employ questionable and privacy intrusive techniques for collecting user information. They typically apply techniques for tracking user visits across different websites, which allow them to reconstruct parts of the users' browsing history. To that end, numerous works [204][205][206][207][208][209][210][211][212] investigate the various approaches employed by trackers, and propose protection mechanisms. Also, a large body of work has investigated targeted behavioral advertising with regards to different levels of personalization, based on the type of

information used to target the user [213][214][215], and its effectiveness [216][217][218][219][220][221].
Also, cookie synchronization (Csync) has become a commonplace on the Web. One of the first works to discuss this mechanism [222][36] studies programmatic auctions from a privacy perspective and presents CSync as an integral part of communication between the participating entities. The study identified over 100 CSync events while crawling the top 100 sites. In [223], authors conducted a CSync privacy analysis by studying a small dataset of 3000 crawled sites, in conjunction with re-spawning cookies and how, together, they affect the reconstruction of user's browsing history by trackers. In [224] they measured the advertising ecosystem cost to users. Focusing on user privacy and targeted advertising, they used CSync as a metric for anonymity loss, showing that users receive 3.4 CSyncs per ad-impression. Papadopoulos et al. [225] showed how CSync can wreck a secure browsing session. They show cases where 3rd-parties may leak a user's cookie IDs and browsing history, thus increasing the identifiability of the user to a snooping ISP. By probing the top 12k Alexa sites they find 1 out of 13 websites exposing their users to these privacy leaks even when they use TLS and secure VPN services. In a recent census by Englehardt et al. [226], authors measure CSync and its adoption in a small subset of 100,000 crawled sites, before highlighting the need of further investigation given its increased privacy implications. Their results show that 157 of top 200 (i.e. 78%) 3rd parties synchronize cookies with at least one other party.
In [227], the authors study the economics and the revenue implications of CSync from the point of view of an informed seller of advertising space, uncovering a trade-off between targeting and information leakage. Similarly, in [228], authors explore the role of data providers on the price and allocation of consumer-level information and develop a simple model of data pricing that captures the key trade-offs involved in selling information encoded in 3rd-party cookies. In [229] they investigate tracking groups that share user-specific identifiers in a dataset collected after recording the browsing history of 100 users for two weeks. In this dataset, they detect 660 ID- sharing groups and found domains with sensitive content (such as health-related) that shared IDs with domains related to ad-trackers.
In [230] they aim to enhance the transparency in ad ecosystem with regards to information sharing, developing a content-agnostic methodology to detect client- and server-side flows of information between ad exchanges by leveraging retargeted ads. By using crawled data, authors collected 35448 ad impressions and identified 4 different kinds of information sharing behavior between ad exchanges. In [231], they study the diffusion of user tracking caused by RTB-based programmatic ad-auctions, considering CSync as the core component of such auctions and the primary factor of the diffusion of privacy leaks. Results of their study show that under specific assumptions, no less than 52 tracking companies can observe at least 91% of an average user's browsing history.
In ACCORDION, we will develop new techniques for identifying fingerprinting happening on user devices, as well as uniqueness of devices. Based on the obtained results, we will develop methods for blocking such identifications and removing sensitive information leaked to unauthorized 3rd-parties.

2.8.3.3 CONCLUSIONS AND HIGHLIGHTS

In recent years, we have witnessed a significant increase in the number of attacks against ML models as well as in the number of solutions proposed to defend against them. However, we noted that it still remains unclear which of the proposed solutions performs best in each context from a security, privacy and performance point of view. Similarly, recent works have shown that the techniques to fingerprint users when browsing the Web are becoming more pervasive, sophisticated and difficult to detect. Clearly, there is a need for more research to document and understand these new tracking techniques as well as to prevent them from learning the users' behaviour in the Web.

2.9 Application model for automatic deployment / migration of components

The objectives and outcomes for this Research Topic are related to the work performed in Task 5.1.

2.9.1 Objectives

A language to define the topology and orchestration specifications for an ACCORDION application, extended to also define the conditions under which VNFs will be used and where. The language should be accompanied by appropriate clients to facilitate automatic delivery.

2.9.2 Outcome

- Creation of application models & policies
- Software that transforms application models to deployment plan for Docker Swarm
- Web services to provide the deployment plan & policies

2.9.3 State-of-the-art

This SoA section covers all the indicated objectives.
The central objective of this research is to find a way to describe the ACCORDION applications and their components, i.e. to have an application model which can represent the requirements of the application components in terms of resource capacity. We also need a description with a certain syntax, so we can build a parser and validator for this description.
Another objective is to produce policies that will describe which actions have to be initialized when a failure happens.
One candidate specification that seems to be covering some of the required objectives is TOSCA (Topology and Orchestration Specification for Cloud Applications) [232] which is an OASIS open standard that provides a way to describe service components and their relationships using a service topology. TOSCA supports TOSCA Simple Profile in YAML [233] to write specifications in YAML and simplify the TOSCA service templates. With TOSCA Simple Profile in YAML we can describe services and applications hosted on the cloud including their components, relationships, dependencies, requirements, and capabilities. TOSCA Simple Profile in YAML has a node template to describe components of an application and a topology template to describe nodes and their relations.
TOSCA and Docker can be combined to provide solutions as TOSCA can be used for Docker topologies and application lifecycle management [234]. TOSCA has also been used for deployment description in OpenStack [235] instances. The first step was to read a TOSCA CSAR file, which is a zip file that contains the TOSCA-Metadata directory and the Definitions directory [232]. TOSCA-Metadata contains metadata which describe other contents of CSAR, while the Definitions directory contains definitions that are related to the cloud application, e.g. relationships and nodes. The information can then be transformed with Heat Translator [236] to a compatible Heat Orchestration Template (HOT) [237], which is an alternative of TOSCA that OpenStack uses for deployment of resources in Cloud, and finally deploy the service to a Cloud System. HOT [237] like TOSCA uses YAML to describe the orchestration. In these documents we find two main elements: components and resources. The big difference [245] between TOSCA and HOT is that TOSCA can describe

services at IaaS, PaaS and SaaS layer, while HOT can only describe IaaS layer services. Heat Translator uses the OpenStack TOSCA parser [238] to parse TOSCA Simple Profile in YAML or TOSCA CSAR.
OASIS does not provide a tool that can parse or validate TOSCA or even perform an orchestration based on the description, but in this research [239] which was the basis for OpenTOSCA [254] which is a TOSCA runtime environment, a TOSCA container is introduced for managing the properties assigned to nodes and relationship instances.
TOSCA containers have also been proposed as a solution in [240] to deploy applications by processing the CSAR files in two ways. In the first one it processes the CSAR and does the deployment according to the description of the Service Template. In the second one the nodes are being deployed without their requirements and when the deployment of nodes finishes it checks which nodes satisfy the requirements.
Another tool which uses the OpenStack TOSCA parser [238] is a TOSCA YAML validator named Sommelier [242] which is developed by the Computer Science Department of the University of Pisa. Sommelier [241] takes as input a TOSCA YAML file or a CSAR file, the input is forwarded to OpenStack TOSCA parser [238] to check the syntax of the description, and if the syntax is correct then some Python objects are produced to represent the described application. Sommelier checks those objects to validate the topology. The validation operation is described on this research [242].
Another project of the Computer Science Department of the University of Pisa is Tosker [244]. Tosker [243][244] basically uses TOSCA Simple Profile in YAML to orchestrate the Docker containers instead of the Docker Compose files, it is a very interesting approach and something that we want also to achieve. Tosker, as it is mentioned in [244], deploys the containers on the same host, in our case we want a similar mechanism that can deploy containers on Docker Swarm.
TOSCA can also work well with Kubernetes. In [246], TOSCA is used to describe the Kubernetes cluster federation in order to automate the distribution and federation of a cluster container, to automate the service status management by describing in TOSCA the horizontal pod auto-scaler. The input of the orchestrator system are YAML scripts, the TOSCA node and relationship template were defined as YAML-based Cloudify [247] plugins.
Besides Docker and Kubernetes, TOSCA YAML Simple Profile in case of INDIGO Orchestrator [248] has also been used so as to orchestrate and initialize resources on Cloud Management Frameworks (like OpenStack and OpenNebula) and Mesos clusters.
As TOSCA is being used in all the above mentioned tools and researches it seems a fitting solution for our case, but it is not the only solution and other tools have to be considered too. In [249], two declarative modeling tools are proposed for automation deployment of a container platform, Juju [250] and Apache Brooklyn [251]. Juju [250] from Canonical can deploy applications on Cloud; it uses “charms” which define requirements, configuration, installation, installation and upgrade of the service. A charm bundle uses YAML files to describe the composition of the services that would be deployed with charms, the configuration parameters and the relationships of the services.
Juju has a command line client and a web GUI client available. Juju has a controller for orchestrating automated deployments and maintenance. The controller is communicating with agents on nodes to monitor continuously the status of the deployment. Controller’s goal is to ensure that the deployment is always following the model. If the model changes while it is being deployed, this change will also happen to the deployment in a way that all components of the service will be informed about it until the whole system will reach a new stable state. When an event causes changes to the model it triggers some action scripts to deal with it, these action scripts are also defined in the charms and they can do operations like pausing or resuming a service or even establish a connection between two services.

Finally, with Apache Brooklyn [251] we can have service modelling, monitoring and policy-based automation. Apache Brooklyn uses blueprints which are YAML files that describe an application including their components, configurations, relationships and deployment scenarios. It has a web UI to deploy, monitor and manage applications. Both cases of Juju [250] and Brooklyn [251] specify the application components, their deployment and the policies for them but they have a whole environment that can parse their own specifications, then deploy services on nodes and monitor those services. In the case of TOSCA, we have to find and extend or develop a custom tool like Juju or Brooklyn.
In [249], they chose Juju over Apache Brooklyn in their model. They also used constraints for the units and node capabilities, the node roles and the connection to the storage module. The deployment plan that was produced from the orchestrator contained the step of selecting the physical server, deploying the OS to the server and installing the required software for the components assigned to those nodes. By changing the number of the nodes in the model they achieved the automatic scaling. But in another case [252] the tool of preference was TOSCA for application modeling and Apache Brooklyn for the development of a unified API for the management of IaaS and PaaS. The justification was that Apache Brooklyn, except from the provisioning and deployment of cloud applications, can also monitor the health of the application and its metrics, and in addition it manages the dependencies between the components.
As the work in [253] proposes to combine TOSCA with Apache Brooklyn we might need to have a transformation from TOSCA to CAMP [255], as CAMP is not able to have a topology like TOSCA does. In addition to TOSCA, OASIS has another standard named CAMP (Cloud Application Management for Platforms) [255]. CAMP is a standard that aims to PaaS and it describes deployment, management and monitoring of cloud applications in a platform agnostic way. As it was also proposed in [252], CAMP aims to develop a unified API for Cloud platforms and to benefit developers to create CAMP specifications for services and mechanisms that interact with the different platforms by using their own interfaces. CAMP [252] [256] can describe platform, resources, services, sensors and operations. In terms of platforms it describes the layer under execution on top of the platform, in terms of resources it refers to the functionalities of the platform, in terms of services it describes the interaction with the platform like the deployment of a Web application, in terms of sensors it refers to the management of the access on metrics via a RESTful API based on the status of applications components and in terms of operations it refers to actions that can interact with resources. There are two proposed methods of transformation by the research in [253]. The first proposes the generation of an intermediate graph that should contain all the information of the application modules and their relationships, but the representation is done neither in TOSCA or CAMP [256]. Each node of the graph describes a component and edges represent the relationships [256]. When a transformation like this one is happening we need to maintain the knowledge between the specifications to be sure that we did not skip some property in the process that is needed for the deployment. Then from this graph we can produce the deployment plan that Brooklyn expects in CAMP YAML [255][256]. Some small transformations needed to be made from the agnostic description to use CAMP YAML plan on Apache Brooklyn. For example a PostgreSQL that in the agnostic description can be named `agnostic.service.type.sql.PostgreSQL` shoud be renamed `org.apache.brooklyn.entity.database.postgresql.PostgreSqlNode`. The tool that does this type of transformation is named TOMAT (TOSCA sMArt Translator) [256] and it is available on github [257]. The only disadvantage of this tool is that it takes as input TOSCA files in XML format rather than TOSCA YAML.
The second method proposes to use meta-model transformations, so we need in this case to define the meta-model of TOSCA-extended and the Brooklyn plan. Apache Brooklyn is based on CAMP

and supports its features and concepts but the research [256] stresses the fact that since TOSCA YAML was available it could have been used for the deployment plan of Apache Brooklyn without the use of the CAMP specification.

2.9.3.1 CONCLUSIONS AND HIGHLIGHTS

In our case TOSCA seems to be the best solution to define the application models of the ACCORDION applications, their requirements and relationships. As it seems to have reached high popularity for the handling of deployment [238][241][243][248][254]. We could follow the logic of Tosker [243][244] and combine TOSCA and Docker/Kubernets to develop a software that can parse and validate TOSCA YAML application model files and then transform them to Docker Compose/Docker stack files/Kubernets so orchestrator could use them as a deployment plan, as it seems there is no tool that does this job right now. These files can be provided through a web service to the ACCORDION orchestrator component. In addition a possible extension in our TOSCA YAML application model could be some information for the container architecture of the described application. An alternative that we should investigate is if we can use TOSCA and Brooklyn to do the orchestration as the research [253] proposes it also seems to be suitable for our case.
In case of policies TOSCA is again the best candidate, as it supports policies to describe which actions have to be initialized when a failure happens and it can be extended [234] to describe the conditions under which VNFs will be used. Hence, we have to predefine the policies on TOSCA YAML files and then provide them via a web service to other ACCORDION components.

2.10 Modelling and assessing QoE for NextGen applications

The objectives and outcomes for this Research Topic are related to the work performed in Task 5.2.

2.10.1 Objectives

The main goal of this research is the development of objective models that can predict the quality of new generation applications, running on top of IP-based networks, such as online gaming, cloud gaming and Virtual Reality based streaming applications. In this task, monitoring QoE models will be developed to manage the services in the ACCORDION framework through optimal resource allocations leading to an improved QoE for the end users.

2.10.2 Outcome

Within the ACCORDION research work, it is expected to investigate the factors influencing the QoE of applications that are relevant in the project. For modelling QoE it is required to conduct subjective tests to create quality datasets. In general, four main outcomes of this task can be summarised as follows:

- Identifying the QoE influencing factors for ACCORDION applications
- Defining the methodologies for quality assessment of ACCORDION applications
- Development of training and validation quality datasets for ACCORDION applications
- Development of QoE models for QoE monitoring of ACCORDION applications

2.10.3 State-of-the-art

Recent years have witnessed a significant increase of multimedia services and applications over IP networks. While the number of services is growing and new generation of applications are stacking to the market, providing a service which can satisfy the costumers is a challenging task. Thus, quality assessment is an essential task of any service providers. Quality assessment has shifted from a service-oriented perspective to a user centric perspective by moving from Quality of Service (QoS) to Quality of Experience (QoE). The International Telecommunication Sector defines QoE as:
"*The degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility and / or enjoyment of the application or service in the light of the user's personality and current state.*"
In contrast to QoS that only concentrates on service characteristics, the QoE takes into account many different dimensions of quality which are related to user factors and even to environmental factors.
Quality assessment is a subjective task which sees the quality from the eyes of customers and studies how a service or an application is perceived by users. Therefore, several factors play a role in the final judgement of a user about a service. Subjective quality assessment is an expensive process and service providers tend to develop and use objective models that can predict the quality automatically. Recent year, we have seen several efforts of development and usage of quality models that can be used for quality prediction of different types of services. For example, Netflix developed a video quality metrics, Video Multimethod Assessment Fusion (VMAF), to measure the video quality of compressed videos [262]. VMAF was not only used for quality assessment of the service but also for better tuning the encoding under different circumstances. In addition, standardization bodies such as ITU Telecommunication Standardization Sector (ITU-T) also developed standardized QoE models. The recently published quality models, ITU-T Recommendation G.1072 [263], P.1203 [264] and P.1204 [265] series are a few examples for such models developed in ITU-T Study group 12.
The objective quality assessment models can be classified into different categories depending on the level of information that is accessible [266]. The objective quality models can be classified into four categories: planning models, bitstream models, signal-based models and hybrid models.
Planning models estimate the quality based on the assumed network characteristics, server and client parameters. Bitstream models use the packet-header or payload information as well as edge, server and client information to predict the quality. Signal-based models estimate the quality by analyzing the media signal. The signal-based models are used together with prior models (e.g. bitstream models) which form a hybrid model.

Next Generation Applications

While we have seen several efforts in development of objective models to measure the quality of traditional online services, new generation of applications and services on top of IP-based networks offer new challenges for quality assessment and prediction. Emerging technologies such Virtual Reality (VR) and Augmented Reality (AR) as well as interactive cloud-based applications, e.g. cloud gaming, have special characteristics and require critical resources. These services have different network requirements and constraints, which require different assessment of QoE. The difference is not only affecting the modelling process but also the methodologies to conduct subjective experiments.
In order to standardize such emerging services, there are ongoing standardization works concerning the subjective and objective quality assessment of gaming and VR services. For gaming applications, three recommendations are studied under Study Group 12 and published. ITU-T Recommendation G.1032 [267] identifying the factors influencing the QoE in gaming application. ITU-T Recommendation P.809 [268] describes the methodology for subjective tests. G.1032 was established

with the aim of developing a QoE- based gaming model for predicting the overall quality based on the characteristics of the network, system, as well as player and usage context factors. G.1072 is developed to predict the cloud gaming QoE based on the basic parameters of the network and encoding setting.
In addition, the influencing factors affecting the quality of VR applications are identified in G.1035 with the aim to build a QoE model for VR applications. In addition, other recommendations for adaptive video streaming up to 1080p resolution, P.1203, or higher resolutions, P.1204, are developed within ITU-T study group 12.
To develop QoE models that predict quality of multimedia services, three steps are necessary to be followed. First, all influencing factors that may impact the QoE must be identified which typically differs from one service to another. As an example of this step, ITU-T Recommendation G.1032 [267] identifies the factors influencing the QoE in gaming application and G.1035 [269] lists the factors that might be considered in QoE assessment of VR applications. This is the first objective that ACCORDION project aims for each next generation application that is considered.
The second step is to develop methodologies to conduct subjective tests for a certain service or application. It is very important to develop a subjective method that meets three criteria of validity, reliability and objectivity of measurement method. The method should be suitable with regard to its objective. In addition, it must be reliable in which it produces similar results under consistent conditions. Multiple standardized subjective methodologies are built to assess the quality of multimedia services, among them we can refer to ITU-T P.910 "Subjective video quality assessment methods for multimedia applications" [270], ITU-T P.809 "Subjective evaluation methods for gaming quality" [268], and ITU-T P.913 "Methods for the subjective assessment of video quality, audio quality and audiovisual quality of Internet video and distribution quality television in any environment" [271]. In order to meet the second objective of QoE modeling, it is required to develop the subjective methodologies for ACCORDION applications which can be developed based on the existing knowledge of state-of-the-art researches for applications which are similar to ACCORDION next generation application, e.g. ITU-T Recommendation P.809 [268].
As a final step, the QoE models are developed based on the influencing factors that are identified in the first step and subjective methodologies that is built in the second step. Within the recent years several models are developed and standardized for different multimedia applications that they all follow these three steps. For example, G.1072 is developed to predict the cloud gaming QoE based on the basic parameters of the network and encoding setting. G.1072 is developed based on the influencing factors identified in ITU-T G.1032 [267], and methodologies that is developed in ITU-T P.809 [268]. Similar approach can be seen for other recommendations that are developed for video streaming application. For adaptive video streaming up to 1080p resolution, ITU-T P.1203 series [264], or higher resolutions, ITU-T P.1204 series [265], are developed within ITU-T study group 12.

2.10.3.1 CONCLUSIONS AND HIGHLIGHTS

In the above state-of-the-art section, a short overview of the standardization activities for quality assessment of different multimedia services has been presented. While there are several other methods and models are developed for different applications in the literature, within the ACCORDION project it has been decided to follow the standardized approach to build models for QoE assessment of ACCORDION applications. Therefore, three steps of identifying influencing factors, developing subjective methodology and model development are taken into consideration for each ACCORDION application.

2.11 DevOps tools to automate Edge applications' deployment

The objectives and outcomes for this Research Topic are related to the work performed in Task 5.4.

2.11.1 Objectives

DevOps tools consist of configuration management, test and build systems, application deployment, version control, and monitoring tools. Automation of Edge Applications development and deployment is one of the key features of ACCORDION and our goal is to choose a set of DevOps tools that would best fit the project and would allow for meeting the requirements, many of which refer to application deployment times. Our key is to improve efficiency, reliability, and to reduce failure rates throughout an automated deployment lifecycle.

2.11.2 Outcome

- Deployment as automated, Edge Application centralized process
- Externalized Edge Application properties for smooth & easy management
- Failure prone process, reducing the chance for failure on production
- Improved scalability and flexibility
- Improved deployment speed

2.11.3 State-of-the-art

The technologies described in this section refer to the goals defined in Task 5.4: DevOps to support application deployment. This section describes modern and effective tools for continuous integration and delivery in order to improve the speed and quality of applications and increase the level of control over them. The final choice of DevOps technology will be based on the ACCORDION system architecture developed in Task T2.3 (Frameworks architecture & specifications).
The tools that comprise the DevOps toolchain are varied and ever-changing. In order to achieve a stable, standardized and production-ready format of Edge Applications, we're most leaning towards well-known container technologies including Docker & Kubernetes.
Docker makes the use of containerization for prototyping, creating, launching, and running an application much easier and ensures its operation in different environments. Docker works in a sense like virtualization, thanks to isolating resources without having to create a virtual operating system - a lot of containers can work on one operating system. Due to the lower containerization surcharge, applications often run faster than in a virtual environment.[26]
Docker makes DevOps much easier and eliminates environmental inconsistencies, which gives additional benefits for complex applications:

- Unification of development, testing (QA, UAT) and production environments,
- Using the same tools,
- Reduction of the time between releases,
- Reduction in the number of unsuccessful implementations.

[26] https://docs.docker.com

Kubernetes (usually abbreviated as K8s) is a portable, expandable open-source software platform for managing tasks and services running in containers, which enables declarative configuration and automation.[27]
K8s supports deployment automation, application scaling, container management, process and change monitoring. Application owners and development teams using the platform can focus more on their product development than on DevOps activities (infrastructure management and product customization).
K8s allows to manage clusters (groups of cooperating servers) so that from the user's perspective they look as one machine; K8s removes from the developer the obligation to adapt the application to the requirements of the infrastructure: a specialist orders the launch of the application, and Kubernetes can distribute services between clusters and servers on its own.
Next key point to achieve is deployment automation, which could be carried thanks to Jenkins: a tool, which enables developers to reliably automate building, testing, and deploying their software. Other Jenkins benefits are deployment speed improvements & higher fault tolerance.
Jenkins is a Continuous Integration / Continuous Deployment (CI/CD) tool, so it allows for full automation of all processes:[28]

- Project compilation,
- Execution of unit tests,
- Execution of integration tests,
- Building and publishing of applications.

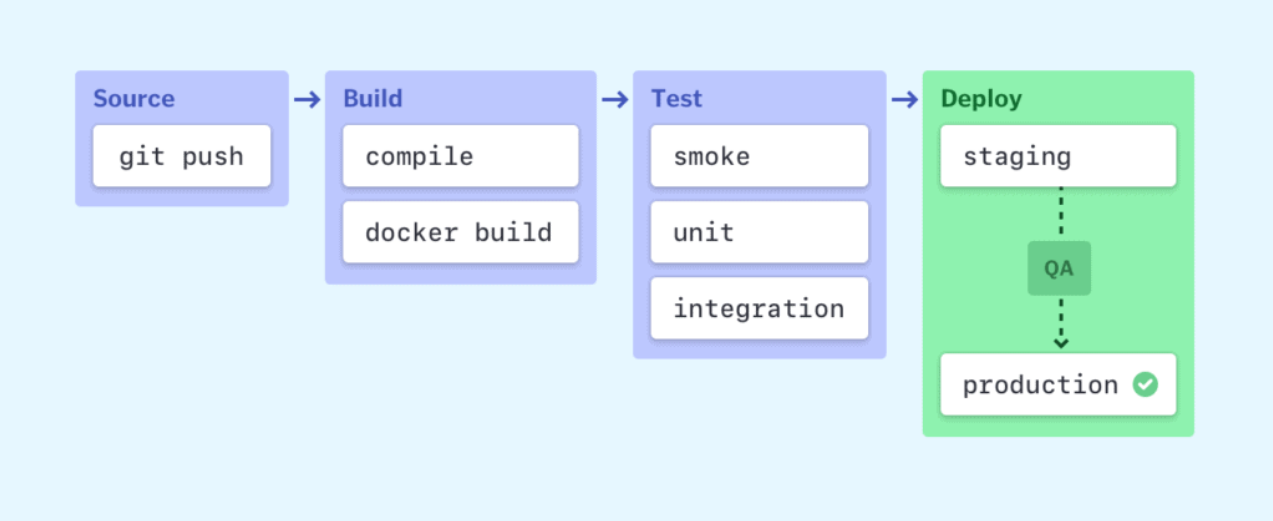

Figure 1 CI/CD Flow [29]

Figure 1 presents a standard CI/CD process. It consists of 4 basic steps:

[27] https://kubernetes.io/docs/home/

[28] https://www.jenkins.io/doc/

[29] https://semaphoreci.com/blog/cicd-pipeline

1. Pushing the changes into the repository. The developer is responsible for this task. He puts his version of code in the repository.
2. The next step is to download the uploaded code and build it. Jenkins is responsible for this and the next step. It must have the necessary libraries and dependencies.
3. Having already built the application, it is necessary to perform tests. They can be on many levels: smoke[30], unit and integration. This is an extremely important activity, because it saves money on evaluating the quality of the software.
4. The last stage is the deployment of a working application. It can be performed for test environments as well as production environments.

Final, but not least important part is to ensure applications monitoring throughout their lifecycle. Any emergency situation, i.e. increased memory usage, should be immediately controlled and in order to achieve that, a reliable monitoring tool should be implemented.

2.11.3.1 CONCLUSIONS AND HIGHLIGHTS

Both Docker, Kubernetes and Jenkins are solutions that support many research and commercial projects. They are constantly tested by many user groups who use them for their purposes. In addition, these components are constantly updated, so in addition to fixing bugs, security packages are also prepared.

2.12 Collaborative VR

The objectives and outcomes for this Research Topic are related to the work performed in Task 6.2.

2.12.1 Objectives

Although in the last years there have been significant advances in VR hardware aiming to unlock the next level of professional VR with human eye resolution and improved processing (such as the Varjo HMDs[31]), still there are a number of factors which limit VR mass adoption on untethered HMDs, such as the need for a large amount of processing power, storage, and GPU processing. Beyond the single user, collaborative VR applications are further exposed to network traffic issues that affect latency, QoE, the immersion and consequently suppressing the embodiment factor and the authenticity of the virtual experience. The latter is of outmost importance in professional VR and high precision medical training (OVR) as the 3D representation of objects has to have a high degree of resemblance to real-life objects and interaction in the virtual environment has to be simulated as in real life. Therefore, the objective is to enable immersive, untethered/ mobile collaborative VR experiences, without the necessity to attach a separate processing unit and support of a large number of remote users in a collaborative, shared VR environment with low-latency. To support this paradigm, more emphasis must be placed on VR software development and the networking solution

[30] Smoke Testing, also known as "Build Verification Testing", is a type of software testing that comprises of a non-exhaustive set of tests that aim at ensuring that the most important functions work. [From https://softwaretestingfundamentals.com/]

[31] https://varjo.com

adopted in parallel to an architecture that leverages edge-cloud resources to supplement the processing, storage and application execution away from the device and allows streaming only necessary content to the client headsets. Hence, there will be less constraints in resource limits, supporting longer lifespan, battery and mobility on HMDs.

2.12.2 Outcome

The ACCORDION research work on this topic will:

- Develop a device agnostic framework supporting existing and upcoming HMDs;
- Exploit computation offloading migrating part of the computing for the OVR application in edge-cloud resources considering the benefit of remote execution, the cost of data transmission and the complexity of application partitioning;
- Adopt a networking solution in multi-user VR environments as part of the master-client paradigm, but with less dependency on a master server for game synchronization and continuity which can be optimally offloaded to the cloud-edge continuum;
- Extend the Geometric algebra interpolation engine to dynamically adapt to network characteristics.

2.12.3 State-of-the-art

VR business applications are very diverse and can be divided according to the application scenario and whether these are targeted for business or industry use or for individual consumers. In the OVR case a strong interactive VR service is assumed supporting a 6DoF perceived experience. 6DoF permits rotational and translational movements within a volumetric space, thus allowing the user to freely traverse a VR scene, which also accounts for a VR experience that is highly non-linear and interactive. Moreover, for acceptable user experience the QoE requirements are (1) Low input-to-display latency, i.e., under 10-25ms [272] motion-to-photon latency, (2) High-quality visual effects, i.e. supporting 4K or higher resolution frames, and (3) Mobility, i.e., the headset or the VR system should be untethered so as not to constrain user interactions [273], both 6DoF and user interactions via controllers. Furthermore, the networking solution as a focal aspect in multi-user VR environments needs to exploit advanced controls for game-state synchronization and message communication supporting the QoE requirements.
The following sections aim to cover the all the indicated objectives in this report. Developments closely linked to the aforementioned objectives will be continuously monitored and will be reported in Y2 work.

2.12.3.1 VR HARDWARE

Existing VR systems can be divided into two categories: High- quality VR and standalone VR systems. Due to the requirements of high quality and low latency, most high-quality VR systems,

such as Varjo[32], Vive Cosmos[33] and Oculus Rift S[34], leverage a powerful desktop PC to render rich graphics contents at high frame rates and visual quality. Therefore, most of these solutions are tethered as they still require extreme processing power on GPU (i.e NVIDIA RTX cards of 2080 and above) while considering that these systems operate at 2160x1200 resolution and 90Hz the generated data rate is much higher than those supported by existing wireless communication products such as normal and 60Ghz Wi-Fi. Standalone untethered HMDs (e.g. Oculus Quest[35], Vive Focus[36]) still lag behind users' expectations for 4K video. Recently announced SoC solutions[37] aim to reduce the performance gap to high-end desktop GPUs in power consumption with a decreasing cost, though the uptake of untethered VR in high quality immersive applications is pending merely on the software end. A higher pixel density implies that computation needs to be performed for a larger number of pixels and in a finer detail, and a higher frame rate implies that the computation for each frame needs to be performed faster. The key bottleneck is the computation involved in rendering frames as mobile GPUs suitable for standalone HMDs are typically 6 − 10× less capable than state-of-the-art GPUs found in desktop/gaming PCs. For example mobile GPUs for HMDs today operate at approximately 1200GFlops (Adreno) with Qualcomm Snapdragon to lead the way while the best desktop GPUs operate at 20000GFlops. As a result due to the constrained CPU/GPU capabilities, either a separate unit for computation is maintained or the processing on the 3D content and interactions of the user is limited or the resolution of the model is reduced. The Table below lists the characteristics and requirements of different HMDs currently available. One further consideration is to enable support of current and upcoming HMDs without residing on devices's proprietary APIs. The OpenXR 1.0 specification[38] released in July 2019 aims towards a unifying open standard that provides cross-platform access to VR and AR platforms and devices.

Table 6 - Characteristics and requirements of different HMDs currently available

Model	Resolution per eye	Refresh rate (Hz)	Field of view (deg)	Requirements*	Type
HTC Vive Pro	1440 x 1600	90	110	NVIDIA® GeForce® GTX 970 or AMD Radeon ™ R9 290 (NVIDIA® GeForce® GTX 1070/Quadro P5000 or above, or AMD Radeon™ Vega 56 or above)	Tethered *untethered with Vive wireless adapter
HTC Vive Cosmos	1440 x 1700	90	110	Same as above	Tethered

[32] https://varjo.com/products/vr-2/

[33] https://www.vive.com/us/product/vive-cosmos/features/

[34] https://www.oculus.com/rift-s/

[35] https://www.oculus.com/quest/

[36] https://enterprise.vive.com/us/product/vive-focus/

[37] https://www.roadtovr.com/qualcomm-snapdragon-xr2-5g-announcement/

[38] https://www.khronos.org/openxr/

HP Reverb pro	2160 x 2160	90	114	DX12 capable graphics. NVIDIA® GTX 1080, NVIDIA® Quadro® P5200, AMD RadeonTM Pro WX 8200	Tethered
Oculus Rift S	1280 x 1440	80	110	NVIDIA GTX 1050 Ti/AMD Radeon RX 470 or greater	Tethered
Pixmap 5 Plus	1440 x 2560	90	170 (max)	NVIDIA GTX 1070 or above	Tethered
Varjo VR-1	1920x1080 (central focal area), 1440x1600 (peripheral display)	60 Hz (central display) 90 Hz (peripheral display)	87	NVIDIA® GTX 1080, NVIDIA® Quadro® P5000 (NVIDIA® GTX 2080, NVIDIA® Quadro® RTX6000)	Tethered
Varjo VR-2	1920x1080 (central focal area), 1440x1600 (peripheral display)	60 Hz (central display) 90 Hz (peripheral display)	87	NVIDIA® GTX 1080, NVIDIA® Quadro® P5000 (NVIDIA® GTX 2080, NVIDIA® Quadro® RTX6000)	Tethered
Oculus Quest	1440 x 1600	72	100 (estimate)	-	Untethered
HTC Vive Focus	1440 x 1600	75	110	-	Untethered
Pico Neo 2	1920 x 2160	90	100	-	Untethered

2.12.3.2 COMPUTATION OFFLOADING AND DATA TRANSMISSION

Currently the OVR platform exploits a multi-player basic networking on Unity where the storage, rendering, compression takes place at the end-device (untethered HMDs) or in a separate processing unit (tethered HMDs). To enable untethered and immersive collaborative VR experiences of multiple Concurrent Users (CCUs) the overhead of computationally intensive tasks must be offloaded from the edge device across the network continuum, while also account for efficient mechanisms for optimizing data transfer considering the network characteristics. Most existing research for supporting untethered VR experiences has independently focused on optimizing the wireless link [274][275] or the VR graphics pipeline [276][277][278] (e.g. pre-rendering and caching, collaborative rendering). When considering computation offloading the device may migrate part of the computation to the cloud or to decentralized edge nodes close to the data source. Solutions residing purely on cloud resources are less suitable due to latency (i.e. RTT~100 ms) limiting computation offloading applicability to delay-tolerant applications [279], while edge computing is a promising way to address the latency issue, as computing happens close to the data source, and it builds on decentralized data processing, reducing the overhead of data transmission. Still one has to consider a) the trade-off between the benefit of remote task execution and the cost of data transmission; b) whether an application component can be offloaded or not, according to its dependencies on the system and /or hardware components (i.e. modules involving UIs; modules interacting with device sensors; and modules depending on local APIs); c) the heterogeneity of the edge nodes considering hardware and software abstractions to encapsulate the difference in platforms (i.e. ARM-based chips, x86 CPUs). In the case of the rendering modules which represent the majority of the CPU consumption and most of the GPU consumption these are rarely independent and are all

mixed by series of calls. This is a challenge for the implementation of code offloading since it reduces the gains in terms of performances, and it imposes to generate calls between distant machines [280]. Closely related to the above, a further challenge in collaborative multi-player environments is to adopt a networking solution that effectively manages communication among clients and handles the game state with less dependency on the master server. Offloading this functionality from the client-host can be realized by a relay server that is able to handle beyond the broadcasting of messages, also host migration and solve for game-state continuity. In most available solutions, the host is the authoritative point for synchronizing the game state among all clients as in the case of UNET by Unity3D[39]. Solutions like Photon[40] aim to support extended features in relay servers, like storing custom properties that can be shared between client instances, automatically detecting when clients disconnect and handling host migration for the most part, as well as allowing any client to send a message to multiple clients without communicating with the master-client. Open source solutions, like Mirror[41] and MLAPI [42], still lack the necessary documentation and have not been fully tested to support the aforementioned characteristics.
For optimizing data transfer across multiple participants the Geometric Algebra Interpolation Engine [281] by OVR, may support fast and efficient compression, simplifying and compacting the broadcasted information on the network. The elements of CGA that handle transformations (CGA motors) can support translation, rotation and dilation (uniform scaling) of joints under a single, GPU-supported mathematical framework and avoid conversion between different mathematical representations in contrast to quaternions and dual-quaternions that support only rotation and rotation-translation, respectively. Extensibility of the method would consider an adaptive transmission rate based on the network characteristics and the QoE model to be developed.

2.12.3.3 CONCLUSIONS AND HIGHLIGHTS

Although the VR hardware landscape evolves rapidly, standalone (untethered) VR solutions are still less capable than tethered devices due to their reduced GPU capabilities and battery life. This favors the exploitation of VR software solutions based on the cloud-edge paradigm to support processing, storage and application execution. In parallel, the dependency on device proprietary API's could be lifted by adopting open standards for cross-platform access to VR devices. In this respect, the OpenXR 1.0 specification can be adopted. For application offloading, due to the main bottleneck of latency, cloud resources are less suitable compared to resources that reside closer to the data source, namely edge resources. In tandem to the above, optimization of the VR graphics pipeline should consider interaction of modules with system and hardware components along with the cost of data transmission. The full offloading option where the HMD device is responsible for UI, input/output, and data sensing seems as a viable option as the rendering modules are rarely independent and are mixed by series of call imposing data transmission challenges in generating calls between distant machines. Furthermore, the networking solution for optimized game synchronization and reduced dependency on the master server should enable the exploitation of a relay server that would solve for game-state continuity beyond broadcasting of messages. Existing open source solutions that were put to test seem not to fully support this functionality yet as opposed to the Photon solution.

[39] https://www.unity3d.com

[40] https://www.photonengine.com

[41] https://mirror-networking.com

[42] https://mlapi.network

2.13 Resource federation models

The objectives and outcomes for this Research Topic are related to the work performed in Task 7.5.

2.13.1 Objectives

ACCORDION aims at relying on pools of federated resources, but how this federation should work? Which federation model to use? How to incentivize providers to join the federation? How to admit new members? How to coordinate providers supporting the same service, and handling a joint Service Level Agreement? How to share revenues among them?

2.13.2 Outcome

The ACCORDION research work on this topic will:

- Identify federation requirements
- Compare federation models and analyze their characteristics
- Select one model based on requirements
- Identify functionalities needed to manage the selected model

2.13.3 State-of-the-art

Please, note that this SoA section covers most of the indicated objectives. A couple of them (incentives and revenue sharing, the one less tied to the technical implementation) will be elaborated in the second version of the present SoA deliverable (D7.9), since they touch upon financial and business topics that will be more in our focus during Y2, when we will start to dive deeper into the project exploitation. Even the deliverable D7.5, containing the results of the task 7.5, might likely provide some anticipation about such aspects.
Business models to enable resource sharing and provider federation along the ACCORDION objectives and guidelines have not been implemented in the market yet. The known references come from research, in particular from the telecommunication domain, where co-opetitive models of this kind (collaboration among entities that are normally competitors) have been investigated.
An important baseline point are still the results of the investigation performed by the H2020 5GEx project[43]. 5GEx developed a multi-domain orchestration platform, enabling telecommunication operators and service providers to share selected portions of their resources, to deliver end-user services combining computational and connectivity resources running in different administrative domains. The services addressed by 5GEx are meant for the portfolio of telecommunication operators (NFV network services, or added value connectivity services). Nevertheless, the mechanisms and

[43] https://cordis.europa.eu/project/id/671636

tenets underlying the 5GEx federation can be taken as reference even for a different scenario like the ACCORDION one.
Hereafter, we outline the main choices performed in the 5GEx business model, trying to make a first assessment of their possible application to ACCORDION.

- 5GEx federation is **not open**. The investigation showed that a fully open admission could bring up risks, due to the scale of involved infrastructures, the complexity of contractual agreements and the challenge of defining an automated admission procedure. In 5GEX, the admission is controlled through offline procedures, not specified in full detail. ACCORDION could re-evaluate this point, given its edge coverage, with a different expectable granularity of providers, and also the availability of new options like Blockchain to manage on-the-fly admissions
- 5GEx federation has not a unique entry point for users, nor a centralized business management represented by a brokering entity. Telecommunication operators had a clear preference for a distributed entry-point model, where each provider acts as entry-point to the system for its own users (or customers). This can probably be a reasonable option for ACCORDION as well, since the ACCORDION could be even more distributed than the 5GEx one.
- Another key feature of a federation is the *coordination model*, i.e. the operational scheme followed when a set of resources must be collected and allocated in order to compose an end-to-end service or application. 5GEx performed an accurate investigation on this aspect [258]. Six different models were analysed and assessed especially in terms of their scalability. Eventually, the choice fell on a **per-provider centralized** model. Centralized means that one of the federation peers plays, for each given service request, the role of *service aggregator*, owning the function of business service composition and end-to-end SLA setup. This was slated more efficient that a cascading model, based on one or more bilateral negotiations between the peers. *Per-provider* means that the aggregator function role can be taken by each of the federation peers, whereas in a fully centralized flavour the role is appointed to the same federation entity for all the service requests. The matching of this coordination model with ACCORDION requirements will have to be better investigated in the specification and design phases.
- The *governance model* of the 5GEx federation is based on differentiation of trust levels, and a set of basic rules characterizing a **close** community. The rules enforce fairness of participation to the federation, e.g. imposing non-discriminatory pricing policies as well as any practice aimed at giving undue advantage to any of the peers. Rules enforce transparency and openness of information sharing about resource status and availability. Penalty and reward sharing are duly ruled, as well as tighter revenue sharing mechanisms in case the federation is shifted up to conduct joint business. Finally, admittance and withdrawal procedures have been defined, and minimal technical requirements (e.g., obligation to install a 5GEx compliant multi-domain orchestrator) prescribed.
- 5GEx performed a theoretical investigation on service provisioning policies for a federation under the aforementioned rules. Two possible approaches were analysed: **Task Forwarding** (TF), where each provider can forward part of the requests coming from his customers to be

served remotely by other providers; **Capacity Sharing** (CS), where each provider can grant part of its resources to the other federation participants in the form of "resource slice". Then, the provider that has the control of the granted infrastructure can deploy service components (e.g.Virtual Network Functions) over it. Both the options enabled viable policies, depending upon the preferences of the federation peers.

- The 5GEx investigation didn't come up with one preferred charging and pricing scheme. Different schemes were found suitable, with a suggested tailoring on the maturity stage of the federation. A possible solution was hinted based on e-negotiation agents, for automating the execution of business agreements. After the end of 5GEx, a good lot of research has taken place in the telecommunication domain looking at smart contracts and blockchain as a mean to negotiate multi-party business conditions. Permissioned blockchains are best candidates for this purpose, in a scheme where each of the cooperating party owns a node in the blockchain, and a shared smart contract could act as distributed authority [259].
- Any federation where parties share resources to jointly provision a service or an application needs a mechanism to properly handle and enforce an end-to-end Service Level Agreement. This implies that the party directly interacting with the end user must be able to keep under monitoring all the resources composed to deliver the service, including the ones deployed and running in other parties' domains. His requirement can be met by two different approaches: allowing the customer endpoint party to directly run probes into other parties' infrastructures or trusting the federated peers and accessing the needed metrics in a shared monitoring database fed by each of the resource owners. 5GEx went with the latter option, which poses far less security and isolation problems. This requirement was mapped into 5GEx architecture, by designing an east-west API between different parties' multi-domain orchestrators where the monitored metrics could be accessed, then elaborated by the overarching multi-domain SLA assurance component [260]. The application of this choice to ACCORDION will depend upon the design choices, and the architectural decisions taken for orchestrating the different mini-clouds. Ultimately, this requirement is calling for an architecture model blending local (intra-party) and distributed control mechanisms.
- Connected to the previous point, there is the need for the SLA enforcement subsystem to trigger proper corrective actions when needed, which can be resource migrations to different nodes as well as resource scaling (horizontally or vertically). 5GEx implemented a mechanism for the SLA assurance subsystem to pass such inputs to the multi-party orchestration via a dedicated API.

The above described features of 5GEx federation model should be taken as starting point and matched against the specific requirements of ACCORDION. Many of these specifics come from the edge foundation of the ACCORDION federation, different from a NFV telco infrastructure layer. An edge federation raises several peculiar constraints and issues, analysed in good detail in [261]. Among the items to be investigated we have in particular:

- Network dynamic adaptation. This requires (or at least takes significant advantage from) both network programmability inside each party's domain, and ability to coordinate SDN control planes through a proper east-west API.
- Resource description and modelling (actually in full scope of ACCORDION WP3)
- Resource exposure and discovery mechanisms (also in scope of WP3)
- Offloading mechanisms, paramount in an edge environment
- Distribution of federated resources
- Profiling of applications using the federated resources, less easily to serve by a service catalog-style repository
- Modelling of mobility, and management of resource handover where needed
- Modelling of the network resources, and timely update of shared resource exposure, in case of status changes in available topologies; more in general, assurance that resource status changes are quickly propagated through the federation, to ensure seamless handover or migrations even with applications extremely sensitive to latencies and delays
- Ability to effectively simulate the interaction mechanisms among federated resource domains

2.13.3.1 CONCLUSIONS AND HIGHLIGHTS

The main available references of business models for resource federations alike the ACCORDION case come to date from the telecommunications sector, apart from some more academic proposals and studies which, at a first glance, do not appear equally complete.
In ACCORDION, we focused our attention on the model developed and proposed not much time ago by the 5GEx project, which outlined a business framework for resource providers to cooperate for delivering advanced services including network and computational resources. We went through the single elements of the model that are relevant to the ACCORDION envisioned scenarios, and evaluated the fit of 5GEx guidelines with respects to our project's requirements. As a conclusion, it looks quite reasonable to take the 5GEx work as a starting point to develop the ACCORDION federation proposition. The gaps left to be filled will be analysed and sorted out by the task 7.5, the one devoted in ACCORDION to deliver these results.

3 Conclusions

In the State of the Art analysis reported in this document, the contributors identified, for most ACCORDION research Tasks, methods and tools that perform / support the task in question or contribute towards that direction. This has also been complemented with other novel approaches from the literature that attack the problem or propose a fitting solution.
The next version of this document, D2.6 due at M22, will improve the State of the Art analysis by adding more detail, by covering more topics and reporting possible new approaches appeared in the meantime.

4 References

[1] K. Alhamazani, R. Ranjan, K. Mitra, F. A. Rabhi, P. P. Jayaraman, S. U. Khan, A. Guabtni and V. Bhatnagar, "An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art," Computing, vol. 97, no. 4, pp. 357-377, 2014.

[2] V. C. Emeakaroha, I. Brandic, M. Maurer and S. Dustdar, "Low level Metrics to High level SLAs - LoM2HiS framework: Bridging the gap between monitored metrics and SLA parameters in cloud environments," in 2010 International Conference on High Performance Computing & Simulation, Caen, France, 2010.

[3] K. Fatema, V. C. Emeakaroha, P. Healy, J. Morrison and T. Lynn, "A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives," Journal of Parallel and Distributed Computing, vol. 74, no. 10, pp. 2918-2933, 2014.

[4] "Overview | Prometheus," 2020. [Online]. Available: https://prometheus.io/docs/introduction/overview/. [Accessed 22 April 2020].

[5] M. Großmann, "PyMon," 2017. [Online]. Available: https://github.com/whatever4711/PyMon. [Accessed 22 April 2020].

[6] "Amazon CloudWatch - Application and Infrastructure Monitoring," Amazon Web Services, Inc, 2020. [Online]. Available: https://aws.amazon.com/cloudwatch/. [Accessed 22 April 2020].

[7] M. Großmann and C. Klug, "Monitoring Container Services at the Network Edge," in 2017 29th International Teletraffic Congress (ITC 29), Genoa, Italy, 2017.

[8] A. Souza, N. Cacho, A. Noor, P. P. Jayaraman, A. Romanovsky and R. Ranjan, "Osmotic Monitoring of Microservices between the Edge and Cloud," in 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Exeter, United Kingdom, United Kingdom, 2018.

[9] "Monitoring Docker container metrics using cAdvisor | Prometheus," 2020. [Online]. Available: https://prometheus.io/docs/guides/cadvisor/. [Accessed 4 April 2020].

[10] A. E. Amri, "DoMonit," 2017. [Online]. Available: https://github.com/eon01/DoMonit. [Accessed 22 April 2020].

[11] decryptus, "monit-docker," 2019. [Online]. Available: https://github.com/decryptus/monit-docker. [Accessed 22 April 2020].

[12] J. Willette, "docker-alertd," 2017. [Online]. Available: https://github.com/deltaskelta/docker-alertd. [Accessed 22 April 2020].

[13] T. Metsch, A. Edmonds and B. Par´ak, "Open Cloud Computing Interface – Infrastructure," 2016. [Online]. Available: https://redmine.ogf.org/attachments/220/infrastructure.pdf. [Accessed 23 April 2020].

[14] R. Nyr´en, F. Feldhaus, B. Par´ak and Z. Sustr, "Open Cloud Computing Interface – JSON Rendering," 2016. [Online]. Available: https://www.ogf.org/documents/GFD.226.pdf. [Accessed 25 April 2020].

[15] A. Edmonds and T. Metsch, "Open Cloud Computing Interface – Text Rendering," 2016. [Online]. Available: https://www.ogf.org/documents/GFD.229.pdf. [Accessed 25 April 2020].

[16] P. Lipton, C. Lauwers, M. Rutkowski, C. Noshpitz and C. Curescu, "TOSCA Simple Profile in YAML Version 1.3," 2020. [Online]. Available:https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.3/os/TOSCA-Simple-Profile-YAML-v1.3-os.html#_Toc26969433. [Accessed 27 April 2020].

[17] "qudt:Unit," 2011. [Online]. Available: http://qudt.org/1.1/vocab/OVG_units-qudt-(v1.1).ttl. [Accessed 29 April 2020].

[18] M. Zhang, "CoCoOn v1.0.1 Examples," 2019. [Online]. Available: https://github.com/miranda-zhang/cloud-computing-schema/blob/master/example/quickstart.md. [Accessed 29 April 2020].

[19] Q. Zhang, A. Haller and Q. Wang, "CoCoOn: Cloud Computing Ontology for IaaS Price and Performance Comparison," in The Semantic Web – ISWC, 2019, pp. 325-341.
[20] J. Montes, A. Sánchez, B. Memishi, M. S. Pérez and G. Antoniu, "GMonE: A complete approach to cloud monitoring," Future Generation Computer Systems, vol. 29, no. 8, pp. 2026-2040, 2013.
[21] M. Zhang, "SPARQL-GENERATE Scripts," 2019. [Online]. Available: https://github.com/miranda-zhang/cloud-computing-schema/tree/master/example/sparql-generate. [Accessed 29 April 2020].
[22] G. Gonçalves, P. Endo, M. Santos, D. Sadok, J. Kelner, B. Melander and J.-E. Mångs, "CloudML: An Integrated Language for Resource, Service and Request Description for D-Clouds," in Third IEEE International Conference on Coud Computing Technology and Science, Athens, Greece, 2011.
[23] S. Parikh, N. M. Patel and H. B. Prajapati, "Resource Management in Cloud Computing: Classification and Taxonomy," 2017.
[24] C. Wang, V. Talwar, K. Schwan and P. Ranganathan, "Online detection of utility cloud anomalies using metric distributions," in 2010 IEEE Network Operations and Management Symposium - NOMS 2010, Osaka, Japan, 2010.
[25] R. N. Calheiros, K. Ramamohanarao, R. Buyya, C. Leckie and S. Versteeg, "On the effectiveness of isolation-based anomaly detection in cloud data centers," Concurrency and Computation Practice and Experience, vol. 29, 2017.
[26] "VMware ESXi," VMware, [Online]. Available: https://www.vmware.com/products/esxi-and-esx.html. [Accessed 2020].
[27] D. Smith, Q. Guan and S. Fu, "An Anomaly Detection Framework for Autonomic Management of Compute Cloud Systems," in 2010 IEEE 34th Annual Computer Software and Applications Conference Workshops, Seoul, South Korea, 2010.
[28] "sysstat," [Online]. Available: https://github.com/sysstat/sysstat. [Accessed 2020].
[29] E. Folkerts, A. Alexandrov, K. Sachs, A. Iosup, V. Markl and C. Tosun, "Benchmarking in the Cloud: What It Should, Can, and Cannot Be," in TPCTC 2012:Selected Topics in Performance Evaluation and Benchmarking, Istanbul, Turkey, 2012.
[30] Z. Li, L. OBrien, R. Cai and H. Zhang, "Towards a Taxonomy of Performance Evaluation of Commercial Cloud Services," in 2012 IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, USA, 2012.
[31] "Geekbench," PRIMATE LABS, 2019. [Online]. Available: https://www.geekbench.com/. [Accessed 11 May 2020].
[32] "Intel® LINPACK Benchmark Download – License Agreement," Intel, 2012. [Online]. Available: https://software.intel.com/content/www/us/en/develop/articles/intel-linpack-benchmark-download-license-agreement.html. [Accessed 11 May 2020].
[33] "RAMspeed," Alasir, 2002. [Online]. Available: http://alasir.com/software/ramspeed/. [Accessed 11 May 2020].
[34] J. Hammond, "STREAM benchmark," 2016. [Online]. Available: https://github.com/jeffhammond/STREAM. [Accessed 11 May 2020].
[35] "IOzone," 2016. [Online]. Available: http://www.iozone.org/. [Accessed 11 May 2020].
[36] J. Dugan, S. Elliot, B. A. Mah, J. Poskanzer and K. Prabhu, "iPerf," [Online]. Available: https://iperf.fr/. [Accessed 11 May 2020].
[37] M. Sajjad, A. Ali and A. S. Khan, "Performance Evaluation of Cloud Computing Resources," International Journal of Advanced Computer Science and Applications, vol. 9, 2018.
[38] H. Malmir, F. Farokhi and R. S. Nadooshan, "Optimization of Data Mining with Evolutionary," in Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 2013.
[39] E. A. Gargari and C. Lucas, "Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition," in 2007 IEEE Congress on Evolutionary Computation, Singapore, Singapore, 2007.

[40] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," IEEE Transactions on Evolutionary Computation, vol. 6, no. 1, pp. 58 - 73, 2002.
[41] "Node Exporter," Prometheus, [Online]. Available: https://prometheus.io/docs/guides/node-exporter/. [Accessed 24 May 2020].
[42] K. Tocze´ and S. Nadjm-Tehrani. A taxonomy for management and optimization of multiple resources in edge computing. Wireless Communications and Mobile Computing, 2018, 2018.
[43] J. Zarrin, R. L. Aguiar, and J. P. Barraca. Resource discovery for distributed computing systems: A com- prehensive survey. Journal of Parallel and Distributed Computing, 113:127–166, Mar. 2018.
[44] H. V. Jagadish, B. C. Ooi, Q. H. Vu, BATON: A balanced tree structure for peer-to-peer networks, in: Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, August 30 - September 2, 2005, 2005, pp. 661-672.
[45] M. A. Are n, M. Y. S. Uddin, I. Gupta, K. Nahrstedt, Q-tree: A multi-attribute based range query solution for tele-immersive framework, in: 29th IEEE International Conference on Distributed Computing Systems (ICDCS 2009), 22-26 June 2009, Montreal, Quebec, Canada, 2009, pp. 299-307.
[46] S. Ramabhadran, S. Ratnasamy, J. M. Hellerstein, S. Shenker, Prefix hash tree: An indexing data structure over distributed hash tables, in: Proc. of the 23rd ACM Symposium on Principles of Distributed Computing, 2004.
[47] A. Datta, M. Hauswirth, R. John, R. Schmidt, K. Aberer, Range queries in trie-structured overlays, in: Peer-to-Peer Computing, 2005. P2P 2005. Fifth IEEE International Conference on, IEEE, 2005, pp. 57-66.
[48] T. Pitoura, N. Ntarmos, P. Trianta llou, Saturn: range queries, load balancing and fault tolerance in DHT data systems. IEEE Transactions on Knowledge and Data Engineering, 24 (7) (2012) 1313-1327.
[49] M. Cai, M. Frank, J. Chen, P. Szekely, Maan: A multi-attribute addressable network for grid information services, Journal of Grid Computing 2 (1) (2004) 3-14.
[50] H. Shen, C.-Z. Xu, Leveraging a compound graph-based DHT for multi-attribute range queries with performance analysis, IEEE Transactions on Computers 61 (4) (2012) 433-447.
[51] C. Schmidt, M. Parashar, Squid: Enabling search in DHT-based systems, Journal of Parallel and Distributed Computing 68 (7) (2008) 962-975.
[52] K. Lee, T. Choi, P. O. Boykin, R. J. Figueiredo, Matchtree: Flexible, scalable, and fault-tolerant wide-area resource discovery with distributed matchmaking and aggregation, Future Generation Computer Systems 29 (6) (2013) 1596-1610.
[53] F. Paganelli, D. Parlanti, A DHT-based discovery service for the Internet of Things, Journal of Computer Networks and Communications 2012.
[54] Hidalgo, N., Arantes, L., Sens, P., & Bonnaire, X. (2016). ECHO: Efficient Complex Query over DHT Overlays. J. Parallel Distributed Comput., 88, 31-45.
[55] Carlini, E., Lulli, A., Ricci, L., Dragon: Multidimensional range queries on distributed aggregation trees, Future Generation Computer Systems(55), 2016, pp. 101-115.
[56] S. Dahmen-Lhuissier, “ETSI - Multi-access Edge Computing - Standards for MEC,” ETSI. https://www.etsi.org/technologies/multi-access-edge-computing (accessed Apr. 26, 2020).

[57] L. Frost, “Smart Cities Deserve an Easier Task! Standards Will Help.” ETSI, 2017, Accessed: Apr. 26, 2020. [Online]. Available: https://www.etsi.org/images/files/ETSInewsletter/etsinewsletter-issue2-2017.pdf.
[58] OpenStack, “Build the future of Open Infrastructure.,” OpenStack. https://www.openstack.org/ (accessed Apr. 26, 2020).
[59] OpenStack, “OpenStack Docs: OpenStack Block Storage (Cinder) documentation.” https://docs.openstack.org/cinder/latest/?_ga=2.207234068.668521204.1587912028-1107826335.1583944620 (accessed Apr. 26, 2020).
[60] OpenStack, “OpenStack Docs: Welcome to Swift’s documentation!” https://docs.openstack.org/swift/latest/?_ga=2.207234068.668521204.1587912028-1107826335.1583944620 (accessed Apr. 26, 2020).
[61] OpenStack, “OpenStack Docs: OpenStack Compute (nova).” https://docs.openstack.org/nova/latest/?_ga=2.35253286.668521204.1587912028-1107826335.1583944620 (accessed Apr. 26, 2020).
[62] A. Lebre, J. Pastor, A. Simonet, and F. Desprez, “Revising openstack to operate fog/edge computing infrastructures,” in 2017 IEEE international conference on cloud engineering (IC2E), 2017, pp. 138–148.
[63] D. von Leon, L. Miori, J. Sanin, N. El Ioini, S. Helmer, and C. Pahl, “A lightweight container middleware for edge cloud architectures,” Fog and edge computing: principles and paradigms, pp. 145–170, 2019.
[64] Apache Software, “Apache Hadoop.” https://hadoop.apache.org/ (accessed Apr. 26, 2020).
[65] Apache Software, “Apache Hadoop 3.2.1 – HDFS Architecture.” https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html (accessed Apr. 26, 2020).
[66] H. Ning, Y. Li, F. Shi, and L. T. Yang, “Heterogeneous edge computing open platforms and tools for internet of things,” Future Generation Computer Systems, 2020.
[67] CEPH Foundation, “ceph.io,” Ceph, 2020. https://ceph.io/ (accessed Apr. 26, 2020).
[68] MinIO Inc, “MinIO | Enterprise Grade, High Performance Object Storage,” MinIO, 2020. https://min.io (accessed Apr. 26, 2020).
[69] L. Baresi and D. F. Mendonca, “Towards a Serverless Platform for Edge Computing,” in 2019 IEEE International Conference on Fog Computing (ICFC), Jun. 2019, doi: 10.1109/icfc.2019.00008.
[70] M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi, “Edge computing perspectives: architectures, technologies, and open security issues,” in 2019 IEEE International Conference on Edge Computing (EDGE), 2019, pp. 116–123.
[71] S. Shahzadi, M. Iqbal, T. Dagiuklas, and Z. U. Qayyum, “Multi-access edge computing: open issues, challenges and future perspectives,” J. Cloud Comput., vol. 6, no. 1, p. 30, 2017.
[72] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, and D. S. Nikolopoulos, “Challenges and opportunities in edge computing,” in 2016 IEEE International Conference on Smart Cloud (SmartCloud), 2016, pp. 20–26.
[73] S. Sondur, K. Kant, S. Vucetic, and B. Byers, “Storage on the Edge: Evaluating Cloud Backed Edge Storage in Cyberphysical Systems,” in 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2019, pp. 362–370.

[74] Yenel, A. (2020). Distributed Execution of Unikernel Applications on Container Orchestration Platform Kubernetes for IoT Scenarios. Master thesis. Technische Universitat Munchen.
[75] Ventre, P. L., Lungaroni, P., Siracusano, G., Pisa, C., Schmidt, F., Lombardo, F., & Salsano, S. (2018). On the fly orchestration of unikernels: Tuning and performance evaluation of virtual infrastructure managers. IEEE Transactions on Cloud Computing.
[76] López, P. G., Sánchez-Artigas, M., París, G., Pons, D. B., Ollobarren, Á. R., & Pinto, D. A. (2018, December). Comparison of faas orchestration systems. In 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion) (pp. 148-153). IEEE.
[77] Vaquero, L. M., Cuadrado, F., Elkhatib, Y., Bernal-Bernabe, J., Srirama, S. N., & Zhani, M. F. (2019). Research challenges in nextgen service orchestration. Future Generation Computer Systems, 90, 20-38.
[78] Casalicchio, E. (2017, May). Autonomic Orchestration of Containers: Problem Definition and Research Challenges. In proceedings of the 10th EAI International Conference on Performance Evaluation Methodologies and Tools on 10th EAI International Conference on Performance Evaluation Methodologies and Tools (pp. 287-290).
[79] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos. Fog orchestration for internet of things services. IEEE Internet Computing, 21(2):16–24, Mar 2017. ISSN 1089-7801. doi: 10.1109/MIC.2017.36.
[80] G. Pollock, D. Thompson, J. Sventek, and P. Goldsack. The asymptotic configuration of application components in a distributed system. Technical report, University of Glasgow. 1998
[81] K. M. Sim. Agent-based cloud computing. IEEE Transactions on Services Computing, 5(4): 564–577, Fourth 2012. ISSN 1939-1374. doi: 10.1109/TSC.2011.52.
[82] Girish B. Chafle, Sunil Chandra, Vijay Mann, and Mangala Gowri Nanda. Decentralized orchestration of composite web services. In Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers &Amp; Posters, WWW Alt. '04, pages 134–143, New York, NY, USA, 2004. ACM. ISBN 1-58113-912-8. doi: 10.1145/1013367.1013390. URL http://doi.acm.org/10.1145/1013367.1013390.
[83] Talagala, N., Sundararaman, S., Sridhar, V., Arteaga, D., Luo, Q., Subramanian, S., ... & Roselli, D. (2018). ECO: Harmonizing Edge and Cloud with ML/DL Orchestration. In USENIX Workshop on Hot Topics in Edge Computing (HotEdge 18).
[84] KubeEdge Device Management User Guide, online at https://docs.kubeedge.io/en/latest/contributing/device_crd_guide.html (accessed Sep 2020)
[85] Local Kubernetes for Linux – MiniKube vs MicroK8s, blog page online at https://codefresh.io/kubernetes-tutorial/local-kubernetes-linux-minikube-vs-microk8s/ (accessed Sep 2020)
[86] Ubuntu Server installation images for Raspberry Pi 2, 3, or 4; online at https://ubuntu.com/download/raspberry-pi (accessed Sep 2020)
[87] Tiny Core Linux, Virtualboxes image, available online at https://virtualboxes.org/images/tiny-core-linux/ (accessed Sep 2020)
[88] Alpine Linux virtual image, available online at https://sourceforge.net/projects/virtualimages/files/AlpineLinux/ (accessed Sep 2020)
[89] Steve Gordon, You got your VM in my container, published online at https://opensource.com/article/18/3/you-got-your-vm-my-container (accessed Sep 2020)

[90] Zalila, F., Challita, S., & Merle, P. (2019). Model-driven cloud resource management with OCCIware. Future Generation Computer Systems, 99, 260-277.
[91] Paraiso, F., Challita, S., Al-Dhuraibi, Y., & Merle, P. (2016, June). Model-driven management of docker containers. In 2016 IEEE 9th International Conference on cloud Computing (CLOUD) (pp. 718-725). IEEE.
[92] OpenStack Wiki, "Enhanced Platform Awareness – For PCIe Devices", availale online at https://wiki.openstack.org/wiki/Enhanced-platform-awareness-pcie (accessed September 2020)
[93] Intel, "Enhanced Platform Awareness in Kubernetes", Intel Feature Brief, available online at https://builders.intel.com/docs/networkbuilders/enhanced-platform-awareness-feature-brief.pdf (accessed September 2020)
[94] Telefonica, openmano, "openvim API - Northbound API documentation", available online at https://github.com/nfvlabs/openmano/blob/master/docs/openvim-api-0.6.pdf (accessed September 2020)
[95] A. Corsaro et al., Eclipse fog05 - Eclipse incubation proposal, availableonline at https://projects.eclipse.org/proposals/eclipse-fog05 (accessed September 2020)
[96] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. G Belle-mare, A. Graves, M. Riedmiller, A. K Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” Nature, vol. 518, pp. 529–33, 02 2015.
[97] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, “Playing atari with deep reinforcement learning,” ArXiv, vol. abs/1312.5602, 2013.
[98] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, “Deep reinforcement learning: A brief survey,” IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26–38, 2017.
[99] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, “Federated multi-task learning,” in Proceedings of the 31st International Conference of Neural Information Processing Systems, ser. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 4427–4437.
[100] A. K. Sahu, T. Li, M. Sanjabi, M. Zaheer, A. Talwalkar, and V. Smith, “On the convergence of federated optimization in heterogeneous networks,” CoRR, vol. abs/1812.06127, 2018. [Online]. Available: http://arxiv.org/abs/1812.06127
[101] A. Fallah, A. Mokhtari, and A. Ozdaglar, “Personalized federated learning: A meta-learning approach,” 2020.
[102] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in Proceedings of the 34th International Conference on Machine Learning - Volume 70, ser. ICML’17. JMLR.org, 2017, p. 1126–1135.
[103] V. Mnih, K. Kavukcuoglu, D. Silver, A. Rusu, J. Veness, M. G Belle-mare, A. Graves, M. Riedmiller, A. K Fidjeland, G. Ostrovski, S. Pe-tersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” Nature, vol. 518, pp. 529–33, 02 2015.
[104] K. Zhang, Z. Yang, and T. Bas¸ar, “Multi-agent reinforcement learning: A selective overview of theories and algorithms,” ArXiv, vol. ssabs/1911.10635, 2019.
[105] V. B. Souza, X. Masip-Bruin, E. Marín-Tordera, W. Ramírez, and S. Sánchez-López, “Proactive vs reactive failure recovery assessment in combined Fog-to-Cloud (F2C) systems,” in 2017 IEEE 22nd International Workshop on Computer Aided Modeling and Design of

Communication Links and Networks (CAMAD), Jun. 2017, pp. 1–5, doi: 10.1109/CAMAD.2017.8031528.
[106] R. Chen et al., “Replication-Based Fault-Tolerance for Large-Scale Graph Processing,” IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 7, pp. 1621–1635, Jul. 2018, doi: 10.1109/TPDS.2017.2703904.
[107] K. Vinay and S. M. Dilip Kumar, “Fault-Tolerant Scheduling for Scientific Workflows in Cloud Environments,” in 2017 IEEE 7th International Advance Computing Conference (IACC), Jan. 2017, pp. 150–155, doi: 10.1109/IACC.2017.0043.
[108] A. Lakhan and X. Li, “Transient fault aware application partitioning computational offloading algorithm in microservices based mobile cloudlet networks,” Computing, vol. 102, no. 1, pp. 105–139, Jan. 2020, doi: 10.1007/s00607-019-00733-4.
[109] P. Karhula, J. Janak, and H. Schulzrinne, “Checkpointing and Migration of IoT Edge Functions,” in Proceedings of the 2nd International Workshop on Edge Systems, Analytics and Networking, Dresden, Germany, Mar. 2019, pp. 60–65, doi: 10.1145/3301418.3313947.
[110] E. AbdElfattah, M. Elkawkagy, and A. El-Sisi, “A reactive fault tolerance approach for cloud computing,” in 2017 13th International Computer Engineering Conference (ICENCO), Dec. 2017, pp. 190–194, doi: 10.1109/ICENCO.2017.8289786.
[111] Y. Tian, J. Tian, and N. Li, “Cloud reliability and efficiency improvement via failure risk based proactive actions,” Journal of Systems and Software, vol. 163, p. 110524, May 2020, doi: 10.1016/j.jss.2020.110524.
[112] A. Power and G. Kotonya, “A Microservices Architecture for Reactive and Proactive Fault Tolerance in IoT Systems,” in 2018 IEEE 19th International Symposium on “A World of Wireless, Mobile and Multimedia Networks” (WoWMoM), Jun. 2018, pp. 588–599, doi: 10.1109/WoWMoM.2018.8449789.
[113] A. Power and G. Kotonya, “Providing Fault Tolerance via Complex Event Processing and Machine Learning for IoT Systems,” in Proceedings of the 9th International Conference on the Internet of Things, Bilbao, Spain, Oct. 2019, pp. 1–7, doi: 10.1145/3365871.3365872.
[114] T.-Y. Hsu and A. D. Kshemkalyani, “A Proactive, Cost-aware, Optimized Data Replication Strategy in Geo-distributed Cloud Datastores,” in Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing, Auckland New Zealand, Dec. 2019, pp. 143–153, doi: 10.1145/3344341.3368799.
[115] R. K. Devi, G. Murugaboopathi, and P. Vijayakumar, “A Graph-Based Mathematical Model for an Efficient Load Balancing and Fault Tolerance in Cloud Computing,” in 2017 Second International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM), Feb. 2017, pp. 136–140, doi: 10.1109/ICRTCCM.2017.25.
[116] H. Han, W. Bao, X. Zhu, and X. Feng, “An learning-based fault-tolerant model for real-time applications on clouds,” in 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), Nov. 2017, pp. 130–133, doi: 10.1109/ICSESS.2017.8342880.
[117] H. El-Kassabi, M. A. Serhani, R. Dssouli, N. Al-Qirim, and I. Taleb, “Cloud Workflow Resource Shortage Prediction and Fulfillment Using Multiple Adaptation Strategies,” in 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), Jul. 2018, pp. 974–977, doi: 10.1109/CLOUD.2018.00149.

[118] P. Guo and Z. Xue, “Cost-effective fault-tolerant scheduling algorithm for real-time tasks in cloud systems,” in 2017 IEEE 17th International Conference on Communication Technology (ICCT), Oct. 2017, pp. 1942–1946, doi: 10.1109/ICCT.2017.8359968.
[119] J. Liu and N. Yang, “Optimal fault tolerant service provisioning for cloud application,” in 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), Jul. 2017, pp. 189–194, doi: 10.1109/ICEIEC.2017.8076541.
[120] M. Abdulazeez, P. Garncarek, D. R. Kowalski, and P. W. H. Wong, “Lightweight Robust Framework for Workload Scheduling in Clouds,” in 2017 IEEE International Conference on Edge Computing (EDGE), Jun. 2017, pp. 206–209, doi: 10.1109/IEEE.EDGE.2017.36.
[121] S. Prathibha, “Investigating the Performance of Machine Learning Algorithms for Improving Fault Tolerance for Large Scale Workflow Applications in Cloud Computing,” in 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dec. 2019, pp. 187–190, doi: 10.1109/ICCIKE47802.2019.9004379.
[122] A. Gulenko, F. Schmidt, A. Acker, M. Wallschläger, O. Kao, and F. Liu, “Detecting Anomalous Behavior of Black-Box Services Modeled with Distance-Based Online Clustering,” in 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), Jul. 2018, pp. 912–915, doi: 10.1109/CLOUD.2018.00134.
[123] T. M. Mengistu, D. Che, A. Alahmadi, and S. Lu, “Semi-Markov Process Based Reliability and Availability Prediction for Volunteer Cloud Systems,” in 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), Jul. 2018, pp. 359–366, doi: 10.1109/CLOUD.2018.00052.
[124] H. Emesowum, A. Paraskelidis, and M. Adda, “Achieving a Fault Tolerant and Reliable Cloud Data Center Network,” in 2018 IEEE International Conference on Services Computing (SCC), Jul. 2018, pp. 201–208, doi: 10.1109/SCC.2018.00033.
[125] J. Villamayor, D. Rexachs, E. Luque, and D. Lugones, “RaaS: Resilience as a Service,” in 2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), May 2018, pp. 356–359, doi: 10.1109/CCGRID.2018.00055.
[126] J. Grover and R. M. Garimella, “Reliable and Fault-Tolerant IoT-Edge Architecture,” in 2018 IEEE SENSORS, Oct. 2018, pp. 1–4, doi: 10.1109/ICSENS.2018.8589624.
[127] V. Marbukh, “Dynamic Job Replication for Balancing Fault Tolerance, Latency, and Economic Efficiency: Work in Progress,” in 2018 IEEE International Conference on Services Computing (SCC), Jul. 2018, pp. 257–260, doi: 10.1109/SCC.2018.00043.
[128] Z. Li et al., “Fault-Tolerant Scheduling for Scientific Workflow with Task Replication Method in Cloud:,” in Proceedings of the 3rd International Conference on Internet of Things, Big Data and Security, Funchal, Madeira, Portugal, 2018, pp. 95–104, doi: 10.5220/0006687300950104.
[129] J. Wang, W. Bao, X. Zhu, L. T. Yang, and Y. Xiang, “FESTAL: Fault-Tolerant Elastic Scheduling Algorithm for Real-Time Tasks in Virtualized Clouds,” IEEE Transactions on Computers, vol. 64, no. 9, pp. 2545–2558, Sep. 2015, doi: 10.1109/TC.2014.2366751.
[130] P. Guo and Z. Xue, “Real-time fault-tolerant scheduling algorithm with rearrangement in cloud systems,” in 2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Dec. 2017, pp. 399–402, doi: 10.1109/ITNEC.2017.8284760.

[131] C. Wang, C. Gill, and C. Lu, “FRAME: Fault Tolerant and Real-Time Messaging for Edge Computing,” in 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Jul. 2019, pp. 976–985, doi: 10.1109/ICDCS.2019.00101.
[132] K. Devi and D. Paulraj, “Multi level fault tolerance in cloud environment,” in 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), Jun. 2017, pp. 824–828, doi: 10.1109/ICCONS.2017.8250578.
[133] Y. Alsenani, G. Crosby, and T. Velasco, “SaRa: A Stochastic Model to Estimate Reliability of Edge Resources in Volunteer Cloud,” in 2018 IEEE International Conference on Edge Computing (EDGE), Jul. 2018, pp. 121–124, doi: 10.1109/EDGE.2018.00024.
[134] O. Khedher, Mastering OpenStack. Packt Publishing Ltd, 2015.
[135] “Concepts and Terminology — Apache CloudStack 4.11.0 documentation.” http://docs.cloudstack.apache.org/projects/archived-cloudstack-getting-started/en/latest/concepts.html
[136] “Creating Highly Available clusters with kubeadm,” Kubernetes. https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/high-availability/ (accessed Jun. 27, 2020).
[137] “Planning - Installing a Cluster | Installation and Configuration | OpenShift Container Platform 3.3.” https://docs.openshift.com/container-platform/3.3/install_config/install/planning.html
[138] “Cisco DCNM Fundamentals Guide, Release 10.4(2) - Configuring DCNM Native High Availability [Cisco Data Center Network Manager 10],” Cisco. https://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/10_4_2/fundamentals/DCNM_Fundamentals_10_4_2/nativeha.html
[139] “How nodes work,” Docker Documentation, Jun. 25, 2020. https://docs.docker.com/engine/swarm/how-swarm-mode-works/nodes/ (accessed Jun. 27, 2020).
[140] “Apache Mesos Essentials [Book].” https://www.oreilly.com/library/view/apache-mesos-essentials/9781783288762/ (accessed Jun. 27, 2020).
[141] Caluwe, R. de, Tré, G.D., Bordogna, G., 2013. Spatio-Temporal Databases: Flexible Querying and Reasoning. Springer Science & Business Media.
[142] Parent, C., Spaccapietra, S., Renso, C., Andrienko, G., Andrienko, N., Bogorny, V., Damiani, M.L., Gkoulalas-Divanis, A., Macedo, J., Pelekis, N., Theodoridis, Y., Yan, Z., 2013. Semantic Trajectories Modeling and Analysis. ACM Comput. Surv. 45, 42:1–42:32. https://doi.org/10.1145/2501654.2501656
[143] Ferrero, C.A., Petry, L.M., Alvares, L.O., Zalewski, W., Bogorny, V., 2019. Discovering Heterogeneous Subsequences for Trajectory Classification. arXiv:1903.07722
[144] Liu, B., Souza, E.N. de, Matwin, S., Sydow, M., 2014. Knowledge-based clustering of ship trajectories using density-based approach, in: 2014 IEEE International Conference on Big Data (Big Data). Presented at the 2014 IEEE International Conference on Big Data (Big Data), pp. 603–608. https://doi.org/10.1109/BigData.2014.7004281
[145] Souza, E.N. de, Boerder, K., Matwin, S., Worm, B., 2016. Improving Fishing Pattern Detection from Satellite AIS Using Data Mining and Machine Learning. PLOS ONE 11, e0158248. https://doi.org/10.1371/journal.pone.0158248

[146] Etemad, M., Soares Júnior, A., Matwin, S., 2018. Predicting Transportation Modes of GPS Trajectories Using Feature Engineering and Noise Removal, in: Bagheri, E., Cheung, J.C.K. (Eds.), Advances in Artificial Intelligence, Lecture Notes in Computer Science. Springer International Publishing, pp. 259–264.

[147] Gambs, S., Killijian, M.-O., del Prado Cortez, M.N., 2012. Next Place Prediction Using Mobility Markov Chains, in: Proceedings of the First Workshop on Measurement, Privacy, and Mobility, MPM '12. ACM, New York, NY, USA, pp. 3:1–3:6. https://doi.org/10.1145/2181196.2181199

[148] Valsamis, A., Tserpes, K., Zissis, D., Anagnostopoulos, D., Varvarigou, T., 2017. Employing traditional machine learning algorithms for big data streams analysis: The case of object trajectory prediction. Journal of Systems and Software 127, 249–257. https://doi.org/10.1016/j.jss.2016.06.016

[149] Pesaranghader, Ahmad, Pesaranghader, Ali, Matwin, S., Sokolova, M., 2018. One Single Deep Bidirectional LSTM Network for Word Sense Disambiguation of Text Data, in: Bagheri, E., Cheung, J.C.K. (Eds.), Advances in Artificial Intelligence, Lecture Notes in Computer Science. Springer International Publishing, pp. 96–107.

[150] Wu, X., Zurita-Milla, R., Verdiguier, E.I., Kraak, M.-J., 2018. Triclustering Georeferenced Time Series for Analyzing Patterns of Intra-Annual Variability in Temperature. Annals of the American Association of Geographers 108, 71–87. https://doi.org/10.1080/24694452.2017.1325725

[151] Jiang, X., de Souza, E.N., Pesaranghader, A., Hu, B., Silver, D.L., Matwin, S., 2017. TrajectoryNet: An Embedded GPS Trajectory Representation for Point-based Classification Using Recurrent Neural Networks, in: Proceedings of the 27th Annual International Conference on Computer Science and Software Engineering, CASCON '17. IBM Corp., Riverton, NJ, USA, pp. 192–200.

[152] Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

[153] Bouras, I., Aisopos, F., Violos, J., Kousiouris, G., Psychas, A., Varvarigou, T., 2019. Mapping of Quality of Service Requirements to Resource Demands for IaaS, in: ResearchGate. Presented at the 9th International Conference on Cloud Computing and Services Science, Crete, Greece. https://doi.org/10.5220/0007676902630270

[154] T. M. Truong, A. Harwood, R. O. Sinnott, and S. Chen, "Performance Analysis of Large-Scale Distributed Stream Processing Systems on the Cloud," in 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), Jul. 2018, pp. 754–761, doi: 10.1109 CLOUD.2018.00103.

[155] X. Li, S. Liu, L. Pan, Y. Shi, and X. Meng, "Performance Analysis of Service Clouds Serving Composite Service Application Jobs," in 2018 IEEE International Conference on Web Services (ICWS), Jul. 2018, pp. 227–234, doi: 10.1109/ICWS.2018.00036.

[156] G. Kaur, A. Bala, and I. Chana, "An intelligent regressive ensemble approach for predicting resource usage in cloud computing," Journal of Parallel and Distributed Computing, vol. 123, pp. 1–12, Jan. 2019, doi: 10.1016/j.jpdc.2018.08.008.

[157] Q. Zia Ullah, S. Hassan, and G. M. Khan, "Adaptive Resource Utilization Prediction System for Infrastructure as a Service Cloud," Computational Intelligence and Neuroscience, Jul. 25, 2017. https://www.hindawi.com/journals/cin/2017/4873459/

[158] K. Mason, M. Duggan, E. Barrett, J. Duggan, and E. Howley, “Predicting host CPU utilization in the cloud using evolutionary neural networks” Future Generation Computer Systems, vol. 86, pp. 162–173, Sep. 2018, doi:10.1016/j.future.2018.03.040.
[159] Yousefpour, A., Fung, C., Nguyen, T., Kadiyala, K., Jalali, F., Niakanlahiji, A., . . . Jue, J. P. (2018). All one needs to know about fog computing and related edge computing paradigms. Journal of Systems Architecture.
[160] Mouradian, C., Naboulsi, D., Yangui, S., Glitho, R. H., Morrow, M. J., & Polakos, P. A. (2017). C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, “A comprehensive survey on fog computing: State-ofthe-art and research challenges,. IEEE Communications Surveys \& Tutorials, 416--464.
[161] Bilal, K., Khalid, O., Erbad, A., & Khan, S. U. (2018). Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers. Computer Networks, 94--120.
[162] Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). DDoS in the IoT: Mirai and other botnets. Computer, 80--84.
[163] Xiaoming, L., Sejdini, V., & Chowdhury, H. (2010). Denial of service (dos) attack with udp flood. School of Computer Science, University of Windsor, Canada.
[164] Udhayan, J., & Anitha, R. (2009). Demystifying and rate limiting ICMP hosted DoS/DDoS flooding attacks with attack productivity analysis. In 2009 IEEE International Advance Computing Conference (pp. 558--564). IEEE.
[165] Bogdanoski, M., Suminoski, T., & Risteski, A. (2013). Analysis of the SYN flood DoS attack. International Journal of Computer Network and Information Security (IJCNIS), 1--11.
[166] Elleithy, K. M., Blagovic, D., Cheng, W. K., & Sideleau, P. (2005). Denial of service attack techniques: analysis, implementation and comparison. SCHOOL OF COMPUTER SCIENCE & ENGINEERING FACULTY PUBLICATIONS.
[167] Dhanapal, A., & Nithyanandam, P. (2017). An effective mechanism to regenerate HTTP flooding DDoS attack using real time data set. In 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT) (pp. 570--575). IEEE.
[168] Wiens, F., & Zitzmann, A. (1999). Predation on a wild slow loris (Nycticebus coucang) by a reticulated python (Python reticulatus). Folia Primatologica, 362.
[169] Hu, Y.-H., Choi, H., & Choi, H.-A. (2004). Packet filtering to defend flooding-based DDoS attacks [Internet denial-of-service attacks]. In 2004 IEEE/Sarnoff Symposium on Advances in Wired and Wireless Communications (pp. 39--42). IEEE.
[170] Luo, H., Lin, Y., Zhang, H., & Zukerman, M. (2013). Preventing DDoS attacks by identifier/locator separation. IEEE network, 60--65.
[171] Qin, F., Lu, S., & Zhou, Y. (2005). SafeMem: Exploiting ECC-memory for detecting memory leaks and memory corruption during production runs. In 11th International Symposium on High-Performance Computer Architecture (pp. 291--302). IEEE.
[172] Chen, S., Xu, J., Nakka, N., Kalbarczyk, Z., & Iyer, R. K. (2005). Defeating memory corruption attacks via pointer taintedness detection. In 2005 International Conference on Dependable Systems and Networks (DSN'05) (pp. 378--387). IEEE.
[173] Frassetto, T., Jauernig, P., Liebchen, C., & Sadeghi, A.-R. (2018). {IMIX}: In-Process Memory Isolation EXtension. In 27th {USENIX} Security Symposium ({USENIX} Security 18) (pp. 83--97).

[174] Dietz, C., Castro, R. L., Steinberger, J., Wilczak, C., Antzek, M., Sperotto, A., & Pras, A. (2018). IoT-botnet detection and isolation by access routers. In 2018 9th International Conference on the Network of the Future (NOF) (pp. 88--95). IEEE.
[175] Anley, C. (2002). Advanced SQL injection in SQL server applications.
[176] Cui, A., Costello, M., & Stolfo, S. (2013). When firmware modifications attack: A case study of embedded exploitation.
[177] Ronen, E., Shamir, A., Weingarten, A.-O., & O'Flynn, C. (2017). IoT goes nuclear: Creating a ZigBee chain reaction. 2017 IEEE Symposium on Security and Privacy (SP), 195--212.
[178] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815--823).
[179] Jin, A. T., Ling, D. N., & Goh, A. (2004). Biohashing: two factor authentication featuring fingerprint data and tokenised random number. Pattern recognition, 2245--2255.
[180] Aloul, F., Zahidi, S., & El-Hajj, W. (2009). Two factor authentication using mobile phones. In 2009 IEEE/ACS International Conference on Computer Systems and Applications (pp. 641--644). IEEE.
[181] Pinkas, B., & Sander, T. (2002). Securing passwords against dictionary attacks. In Proceedings of the 9th ACM conference on Computer and communications security (pp. 161--170).
[182] Technology, N. I. (n.d.). National Vulnerability Database. Retrieved from https://nvd.nist.gov/vuln-metrics/cvss
[183] Alberts, C., Dorofee, A., Stevens, J., & Woody, C. (n.d.). Introduction to the OCTAVE Approach. Retrieved from https://resources.sei.cmu.edu/library/Asset-view.cfm?assetid=51546
[184] UcedaVelez, T., & Morana, M. M. (2015). Risk centric threat modeling. Wiley Online Library.
[185] Wuyts, K., Van Landuyt, D., Hovsepyan, A., & Joosen, W. (2018). Effective and efficient privacy threat modeling through domain refinements. In Proceedings of the 33rd Annual ACM Symposium on Applied Computing (pp. 1175--1178).
[186] Potteiger, B., Martins, G., & Koutsoukos, X. (2016). Software and attack centric integrated threat modeling for quantitative risk assessment. In Proceedings of the Symposium and Bootcamp on the Science of Security (pp. 99--108).
[187] DML. Decentralized machine learning. https://decentralizedml.com, 2018
[188] OpenMined. https://www.openmined.org, 2020.
[189] Datafleets: The federated intelligence platform. https://www.datafleets.com, 2020.
[190] FATE. Federated ai technology enabler. https://github.com/FederatedAI/FATE, 2020.
[191] Tensorflow Federated: Machine learning on decentralized data. https://www.tensorflow.org/federated, 2020.
[192] CoMind: Collaborative machine learning. https://comind.org, 2019.
[193] S. Caldas, S. Meher Karthik Duddu, P. Wu, T. Li, J. Konecny, H. B. McMahan, V. Smith, and A. Talwalkar. Leaf: A benchmark for federated settings. https://leaf.cmu.edu, 2019.
[194] Y.Zhao, J.Zhao, M.Yang, T.Wang, N.Wang, L.Lyu, D.Niyato and K.-Y. Lam. Local differential privacy based federated learning for internet of things.https://arxiv.org/pdf/2004.08856.pdf, 2020.

[195] S.Truex, L.Liu, K.-H.Chow, M.E.Gursoy and W.Wei. Ldp-fed: Federated learning with local differential privacy. In 3rd International Workshop on Edge Systems, Analytics and Networking (EdgeSys), April 2020.
[196] M. Seif, R. Tandon and M. Li. Wireless federated learning with local differential privacy. https://arxiv.org/pdf/2002.05151.pdf, 2020.
[197] K.Wei, J.Li, M.Ding, C.Ma, H.H.Yang, F.Farokhi, S.Jin, T.Q.Quek and H. V. Poor. Federated learning with differential privacy: Algorithms and performance analysis. IEEE Transactions on Information Forensics and Security, 2020.
[198] R.C.Geyer, T.Klein and M.Nabi. Differentially private federated learning: A client level perspective. In NIPS Workshop: Machine Learning on the Phone and other Consumer Devices, 2017.
[199] L.Liu, J.Zhang, S.Song and K.B.Letaief. Client-edge-cloud hierarchical federated learning. https://arxiv.org/pdf/1905.06641.pdf, 2019.
[200] M.SalehiHeydarAbad, E.Ozfatura, D.Gunduz and O.Ercetin. Hierarchical federated learning across heterogeneous cellular networks. https://arxiv.org/pdf/1909.02362.pdf, 2019.
[201] J.Zhang, J.Wang, Y.Zhao and B.Chen. An efficient federated learning scheme with differential privacy in mobile edge computing. In International Conference on Machine Learning and Intelligent Communications, pages 538–550. Springer, 2019.
[202] S.Truex, N.Baracaldo, A.Anwar, T.Steinke, H.Ludwig, R.Zhang and Y. Zhou. A hybrid approach to privacy-preserving federated learning. In Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, 2019.
[203] C.Briggs, Z.Fan and P.Andras. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. https://arxiv.org/pdf/2004.11791.pdf, 2020.
[204] PAPADOPOULOS, P., KOURTELLIS, N., RODRIGUEZ, P. R., AND LAOUTARIS, N. If you are not paying for it, you are the product: How much do advertisers pay to reach you? In Proceedings of the ACM Internet Measurement Conference (2017), pp. 142–156.
[205] LE´ CUYER, M., DUCOFFE, G., LAN, F., PAPANCEA, A., PETSIOS, T., SPAHN, R., CHAINTREAU, A., AND GEAMBASU, R. Xray: Enhancing the webs transparency with differential correlation. In 23rd USENIX Security Symposium (2014), pp. 49–64.
[206] MATHUR, A., VITAK, J., NARAYANAN, A., AND CHETTY, M. Characterizing the use of browser-based blocking extensions to prevent online tracking. In Fourteenth Symposium on Usable Privacy and Security (SOUPS) (Baltimore, MD, 2018), USENIX Association, pp. 103–116.
[207] MELICHER, W., SHARIF, M., TAN, J., BAUER, L., CHRISTODORESCU, M., AND LEON, P. G. (do not) track me sometimes: Users contextual preferences for web tracking. Proceedings on Privacy Enhancing Technologies, 2 (2016).
[208] DAS, A., BORISOV, N., AND CAESAR, M. Tracking mobile web users through motion sensors: Attacks and defenses. In Proceedings of the Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA (2016).
[209] PAPADOPOULOS, E. P., DIAMANTARIS, M., PA- PADOPOULOS, P., PETSAS, T., IOANNIDIS, S., AND MARKATOS, E. P. The long-standing privacy debate: Mobile websites vs mobile apps. In Proceedings of the 26th ACM International Conference on World Wide Web (2017), pp. 153–162.

[210] PAN, X., CAO, Y., AND CHEN, Y. I do not know what you visited last summer - protecting users from third- party web tracking with tracking free browser. In Proceedings of the Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA (2015).
[211] PAPADOPOULOS, P., KOURTELLIS, N., AND MARKATOS, E. Cookie synchronization: Everything you always wanted to know but were afraid to ask. In The World Wide Web Conference (2019), ACM, pp. 1432–1442.
[212] PACHILAKIS, M., PAPADOPOULOS, P., MARKATOS, E. P., AND KOURTELLIS, N. No more chasing water- falls: A measurement study of the header bidding ad-ecosystem. In 19th ACM Internet Measurement Conference (2019).
[213] BLEIER, A., AND EISENBEISS, M. Personalized on- line advertising effectiveness: The interplay of what, when, and where. Marketing Science 34, 5 (2015), 669–688.
[214] AGUIRRE, E., MAHR, D., GREWAL, D., DE RUYTER, K., AND WETZELS, M. Unraveling the personalization paradox: The effect of information collection and trust-building strategies on online advertisement effectiveness. Journal of Retailing 91, 1 (2015), 34–49.
[215] TUCKER, C. E. Social networks, personalized advertising, and privacy controls. Journal of Marketing Re- search 51, 5 (2014), 546–562.
[216] FARAHAT, A., AND BAILEY, M. C. How effective is targeted advertising? In Proceedings of the 21st ACM International Conference on World Wide Web (2012), pp. 111–120.
[217] LEWIS, R. A., RAO, J. M., AND REILEY, D. H. Here, there, and everywhere: correlated online behaviors can lead to overestimates of the effects of advertising. In Proceedings of the 20th ACM International Conference on World Wide Web (2011), pp. 157–166.
[218] GIRONDA, J. T., AND KORGAONKAR, P. K. iSpy? tailored versus invasive ads and consumers perceptions of personalized advertising. Electronic Commerce Re- search and Applications 29 (2018), 64–77.
[219] CARRASCOSA, J. M., MIKIANS, J., CUEVAS, R., ERRAMILLI, V., AND LAOUTARIS, N. I always feel like somebody's watching me: measuring online behavioural advertising. In Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies (CONEXT) (2015).
[220] BASHIR, M. A., ARSHAD, S., ROBERTSON, W., AND WILSON, C. Tracing information flows between ad ex- changes using retargeted ads. In 25th USENIX Security Symposium (2016), pp. 481–496.
[221] IORDANOU, C., KOURTELLIS, N., CARRAS- COSA, J. M., SORIENTE, C., CUEVAS, R., AND LAOUTARIS, N. Beyond content analysis: Detecting targeted ads via distributed counting. arXiv preprint arXiv:1907.01862 (2019).
[222] Lukasz Olejnik, Minh-Dung Tran, and Claude Castelluccia. 2014. Selling off User Privacy at Auction. In 21st Annual Symposium Network and Distributed System Security (NDSS'14).
[223] Gunes Acar, Christian Eubank, Steven Englehardt, Marc Juarez, Arvind Narayanan, and Claudia Diaz. 2014. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. New York, NY, USA.
[224] Panagiotis Papadopoulos, Nicolas Kourtellis and Evangelos P. Markatos. 2018. The Cost of Digital Advertisement: Comparing User and Advertiser Views. In Proceedings of the 27th International Conference on World Wide Web (WWW'18).

[225] Panagiotis Papadopoulos, Nicolas Kourtellis and Evangelos P. Markatos. 2018. Exclusive: How the (Synced) Cookie Monster Breached My Encrypted VPN Session. In Proceedings of the 11th European Workshop on Systems Security (EuroSec'18).
[226] Steven Englehardt and Arvind Narayanan. Online Tracking: A 1-million-site Measurement and Analysis. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16).
[227] Arpita Ghosh, Mohammad Mahdian, R.Preston McAfee and Sergei Vassilvitskii. To Match or Not to Match: Economics of Cookie Matching in Online Advertising. ACM Trans. Econ. Comput. 2015.
[228] Dirk Bergemann and Alessandro Bonatti. 2015. Selling cookies. American Eco- nomic Journal: Microeconomics 7, 3 (2015), 259–294.
[229] Marjan Falahrastegar, Hamed Haddadi, Steve Uhlig and Richard Mortier.2016. Tracking Personal Identifiers Across the Web.
[230] Muhammad Ahmad Bashir, Sajjad Arshad, William Robertson and Christo Wilson. 2016. Tracing information flows between ad exchanges using retargeted ads. In Proceedings of the 25th USENIX Security Symposium.
[231] Muhammad Ahmad Bashir and Christo Wilson. 2018. Diffusion of User Tracking Data in the Online Advertising Ecosystem. Proceedings on Privacy Enhancing Technologies 4 (2018), 85–103.
[232] P. Lipton, S. Moser, D. Palma and T. Spatzier, "OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) TC," 2011. [Online]. Available: http://docs.oasis-open.org/tosca/TOSCA/v1.0/os/TOSCA-v1.0-os.html#_Toc356403634. [Accessed 17 May 2020].
[233] P. Lipton, C. Lauwers, M. Rutkowski, C. Noshpitz and C. Curescu, "TOSCA Simple Profile in YAML Version 1.3," 2020. [Online]. Available: https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.3/os/TOSCA-Simple-Profile-YAML-v1.3-os.html#_Toc26969433. [Accessed 27 April 2020].
[234] "OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) TC," Oasis Open Standards. Open Source, 2017. [Online]. Available: https://www.oasis-open.org/committees/tosca/faq.php. [Accessed 17 May 2020].
[235] G. Tricomi, A. Panarello, G. Merlino, F. Longo, D. Bruneo and A. Pualiafito, "Orchestrated Multi-Cloud Application Deployment in OpenStack with TOSCA," in 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, China, 2017.
[236] "Heat-Translator," OpenStack, 2016. [Online]. Available: https://wiki.openstack.org/wiki/Heat-Translator#Overview. [Accessed 17 May 2020].
[237] "Heat Orchestration Template (HOT) Guide," OpenStack, 2020. [Online]. Available: https://docs.openstack.org/heat/rocky/template_guide/hot_guide.html#hot-guide. [Accessed 17 May 2020].
[238] "OpenStack/tosca-parser," OpenStack, 2020. [Online]. Available: https://github.com/openstack/tosca-parser. [Accessed 17 May 2020].
[239] T. Binz, U. Breitenbücher, O. Kopp and F. Leymann, "TOSCA: Portable Automated Deployment and Management of Cloud Applications," in Advanced Web Services, New York, NY, Springer, 2013, pp. 527-549.

[240] A. Brogi, J. Soldani and P. Wang, "TOSCA in a Nutshell: Promises and Perspectives," in Service-Oriented and Cloud Computing: Third European Conference, ESOCC 2014, Manchester, UK, 2014.
[241] A. Brogi, A. Di Tommaso and J. Soldani, "di-unipi-socc/Sommelier," 2018. [Online]. Available: https://github.com/di-unipi-socc/Sommelier. [Accessed 17 May 2020].
[242] A. Brogi, A. Di Tommaso and J. Soldani, "Sommelier: A Tool for Validating TOSCA Application Topologies," in Model-Driven Engineering and Software Development, Springer International Publishing, 2018, pp. 1-22.
[243] A. Brogi, L. Rinaldi and J. Soldani, "di-unipi-socc/Tosker," Computer Science Department of the University of Pisa, 2018. [Online]. Available: https://github.com/di-unipi-socc/TosKer. [Accessed 2020 May 18].
[244] A. Brogi, L. Rinaldi and J. Soldani, "TosKer: A synergy between TOSCA and Docker for orchestrating multicomponent applications: TosKer: A synergy between TOSCA and Docker," Software Practice and Experience, vol. 48, no. 11, pp. 2061-2079, 2018.
[245] A. Esposito, B. Di Martino and G. Cretella, "Defining Cloud Services Workflow: a Comparison between TOSCA and OpenStack Hot," in 9th International Conference on Complex, Intelligent, and Software Intensive Systems, Blumenau, Brazil, 2015.
[246] D. Kim, H. Muhammad, E. Kim, S. Helal and C. Lee, "TOSCA-Based and Federation-Aware Cloud Orchestration for Kubernetes Container Platform," Applied Sciences, vol. 9, no. 1, 2019.
[247] "Cloudify," [Online]. Available: https://cloudify.co/. [Accessed 19 May 2020].
[248] "indigo-dc/ orchestrator," [Online]. Available: https://github.com/indigo-dc/orchestrator. [Accessed 19 May 2020].
[249] G. Attardi, A. Barchiesi, A. Colla, R. di Lallo and F. Galeazzi, "Declarative Modeling for Deploying a Container Platform," in 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA), 2018.
[250] "Juju," Canonical, [Online]. Available: https://jaas.ai/. [Accessed 2020 May 19].
[251] "Apache Brooklyn," Apache License v2.0., [Online]. Available: https://brooklyn.apache.org/. [Accessed 19 May 2020].
[252] J. Carrasco, F. Duran and E. Pimentel, "Towards a Unified Management of Applications on Heterogeneous Clouds," in Advances in Service-Oriented and Cloud Computing, Springer, Cham, 2018, pp. 233-246.
[253] J. Carrasco, J. Cubo and E. Pimentel, "Towards a flexible deployment of multi-cloud applications based on TOSCA and CAMP," Communications in Computer and Information Science, vol. 508, pp. 278-286, 2015.
[254] "OpenTOSCA Research Prototype," [Online]. Available: https://www.opentosca.org/. [Accessed 20 May 2020].
[255] A. Karmarkar and G. Pilz, "Cloud Application Management for Platforms," OASIS , 2012. [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=camp. [Accessed 20 May 2020].
[256] J. Carrasco, J. Cubo, E. Pimentel and F. Duran, "Deployment over Heterogeneous Clouds with TOSCA and CAMP," in 6th International Conference on Cloud Computing and Services Science, 2016.
[257] "TOMAT," [Online]. Available: https://github.com/kiuby88/tomat. [Accessed 20 May 2020].

[258] 5GEx Deliverable D2.3: 5GEx Business and Economic Layer
[259] K. Antevski, C. Bernardos: "Federation of 5G services using Distributed Ledger Technologies". Wiley, 2020: https://onlinelibrary.wiley.com/doi/pdf/10.1002/itl2.193
[260] 5GEx project Deliverable D2.2: Final System Requirements and Architecture
[261] Baktir et al:, 2018 Addressing the Challenges in Federating Edge Resources; book chapter accepted to the Fog and Edge Computing: Principles and Paradigms; Editors Buyya, Srirama (https://arxiv.org/pdf/1803.05255)
[262] Netflix. 2020. Perceptual video quality assessment based on multi-method fusion. Retrieved June 18, 2020 from https://github.com/Netflix/vmaf
[263] ITU-T Recommendation G.1072. 2020. Opinion Model Predicting Gaming QoE for Cloud Gaming Services. Geneva Switz. Int. Telecommun. Union.
[264] ITU-T Recommendation P.1203. 2017. Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport. Geneva Switz. Int. Telecommun. Union.
[265] ITU-T. Recommendation P.1204.3 - Video quality assessment of streaming services over reliable transport for resolutions up to 4K with access to full bitstream information. Tech. rep. International Telecommunication Union, 2019.
[266] Alexander Raake, M.-N. Garcia, Sebastian Moller, Jens Berger, Fredrik Kling, Peter List, Jens Johann, and Cornelius Heidemann. 2008. TV-model: Parameter-based prediction of IPTV quality. In 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 1149–1152.
[267] Recommendation ITU-T G.1032 (2017), Influence factors on gaming quality of experience, Geneva, Switzerland: International Telecommunication Union.
[268] ITU-T Recommendation P.809. 2018. Subjective evaluation methods for gaming quality. Geneva Switz. Int. Telecommun. Union.
[269] ITU-T Recommendation G.1035. 2020. Influencing factors on quality of experience for virtual reality services. Geneva Switz. Int. Telecommun. Union.
[270] ITU-T Recommendation P.910. 2008. Subjective video quality assessment methods for multimedia applications. Geneva Switz. Int. Telecommun. Union.
[271] ITU-T Recommendation P.913. 2016. Methods for the subjective assessment of video quality, audio quality and audiovisual quality of Internet video and distribution quality television in any environment. Geneva Switz. Int. Telecommun. Union.
[272] 2014. What VR Could, Should, and almost certainly Will be within two years. Online at: http://media.steampowered.com/apps/abrashblog/Abrash%20Dev%20Days%202014.pdf.
[273] Liu L., Zhong R. Zhang W., Liu Y., Zhang J., Zhang L., Grutese M. Cutting the Cord: Designing a High-quality Untethered VR System with Low Latency Remote Rendering. MobiSys '18: Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services June 2018, pp. 68–80 https://doi.org/10.1145/3210240.3210313
[274] O. Abari, D. Bharadia, A. Duffield, and D. Katabi. Enabling high-quality untethered virtual reality. In NSDI, pages 531–544, 2017.
[275] T. Wei and X. Zhang.Pose information assisted 60ghz networks:Towards seamless coverage and mobility support. In Proc. of the 23rd nAnual International Conference on Mobile Computing and Networking, MobiCom '17, pages 42–55, New York, NY, USA, 2017. ACM.

[276] Z. Lai, Y.C. Hu, Y. Cui, L. Sun, and N. Dai. Furion: Engineering high-quality immersive virtual reality on today's mobile devices. In Proc.of the 23rd International Conference on Mobile Computing and Net- working (MobiCom'17). ACM, Snowbird, Utah, USA, 2017.

[277] K. Boos, D. Chu, and E. Cuervo. Flashback: Immersive virtual reality on mobile devices via rendering memoization. In Proc. of the 14th Annual International Conference on Mobile Systems, Applications, and Services, pages 291–304. ACM, 2016.

[278] L. Liu, R. Zhong, W. Zhang, Y. Liu, J. Zhang, L. Zhang, and M. Gruteser. 2018. Cutting the Cord: Designing a High-quality Untethered VR System with Low Latency Remote Rendering. In Proc. of the 16th Annual International Conference on Mobile Systems, Applications, and Services(MobiSys '18)., 68–80. DOI:https://doi.org/10.1145/3210240.3210313

[279] F. Messaoudi, A. Ksentini and P. Bertin, "On Using Edge Computing for Computation Offloading in Mobile Network," GLOBECOM 2017 - 2017 IEEE Global Communications Conference, Singapore, 2017, pp. 1-7, doi: 10.1109/GLOCOM.2017.8254635.

[280] F. Messaoudi, G. Simon and A. Ksentini, "Dissecting games engines: The case of Unity3D," 2015 International Workshop on Network and Systems Support for Games (NetGames), Zagreb, 2015, pp. 1-6, doi: 10.1109/NetGames.2015.7382990.

[281] Papaefthymiou, M., Hildenbrand, D. & Papagiannakis, G. A Conformal Geometric Algebra Code Generator Comparison for Virtual Character Simulation in Mixed Reality. Adv. Appl. Clifford Algebras 27, 2051–2066 (2017). https://doi.org/10.1007/s00006-016-0689-3