TRANSPARENT APPLICATION DEPLOYMENT IN A SECURE, ACCELERATED AND COGNITIVE CLOUD CONTINUUM

Grant Agreement no. 101017168

Deliverable D2.1
State of the Art Analysis Report

Programme:	H2020-ICT-2020-2
Project number:	101017168
Project acronym:	SERRANO
Start/End date:	01/01/2021 – 31/12/2023

Deliverable type:	Report
Related WP:	WP2
Responsible Editor:	UVT
Due date:	30/06/2021
Actual submission date:	30/06/2021

Dissemination level:	Public
Revision:	FINAL

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017168

Revision History

Date	Editor	Status	Version	Changes
22.02.01	Gabriel Iuhasz	Draft	0.1	ToC and initial structure
22.02.01	Vassiliki Andronikou	Draft	0.1	Initial structure updates
29.03.01	Ferad Zyulkyarov	Draft	0.1	High-performance Fintech Analysis
09.04.01	Marton Sipos	Draft	0.1	Secure storage first iteration
11.04.01	Aitor Fernandez Gomez	Draft	0.1	Added Machine Anomaly Detection in Manufacturing Environments contribution
11.04.01	Gabriel Iuhasz	Draft	0.2	Consolidated version
12.04.01	Aristotelis Kretsis	Draft	0.2	Intelligent telemetry initial contribution
13.04.01	Argyris Kokkinis, Kostas Siozios	Draft	0.2	Hardware and software acceleration first contribution
26.04.01	Marton Sipos	Draft	0.2	Secure storage updated contribution
27.04.01	Vassiliki Andronikou	Draft	0.2	Service Orchestrator and Abstraction Models updated contribution
28.04.01	Paraskevas Bourgos	Draft	0.2	Service Orchestrator and Abstraction Models updated contribution
04.05.01	Aristotelis Kretsis	Draft	0.2	Automated optimized resource orchestration updated contribution
07.05.01	Juan Jose Vegas Almos	Draft	0.3	Self-encrypted NVME description
09.05.01	Dmitry Khabi	Draft	0.4	Energy and resource aware flow mapping contribution
12.05.01	Argyris Kokkinis, Kostas Siozios	Draft	0.4	Updated contribution
12.05.01	Aitor Fernandez Gomez	Draft	0.4	Use-case updated contribution
18.05.01	Silviu Panica	Draft	0.5	Consolidated version
20.05.01	Annastassios Nanos	Draft	0.5	Trusted and isolation execution updates
20.05.01	Ioan Dragan	Draft	0.6	Contributions added
24.05.01	Iuhasz Gabriel	Draft	0.7	Updated version
27.05.01	Silviu Panica	Draft	0.8	Consolidated version
29.05.01	Silviu Panica	Draft	0.9	Added missing input for remaining sections
29.05.01	Silviu Panica	Draft	1.0, 1.1, 1.2	Revised versions before internal review

01.06.01	Efthymios Chondrogiannis	Draft	1.3	Section 6.4 updates
02.06.01	Paraskevas Bourgos, Makis Karadimas	Draft	1.3	Updated 6.2 related tools (StreamHandler)
02.06.01	Ferad Zyulkyarov	Draft	1.3	Updated section 7.3 – Fintech use-case
03.06.01	Juan Jose Vegas Almos	Draft	1.4	Updated section 4.2 – Self-encrypting NVMe
14.06.01	Silviu Panica	Draft	1.5	Review comments added
16.06.01	Dmitry Khabi	Draft	1.5	Review comments addressed
22.06.01	Silviu Panica	Draft	1.6	Integrated post review contributions.
24.06.01	Silviu Panica	Draft	1.7	Minor changes and fixes
28.06.01	Ioan Dragan	Draft	1.8	Minor changes
30.06.01	Silviu Panica	Final	1.9	Final version.

Author List

Organization	Author
UVT	Gabriel Iuhasz, Silviu Panica, Ioan Dragan
CC	Marton Sipos
IDEKO	Aitor Fernandez Gomez
INB	Ferad Zyulkyarov
INNOV	Vassiliki Andronikou, Efthymios Chondrogiannis, Efstathios Karanastasis, Filia Filippou
ICCS	Aristotelis Kretsis, Kostas Kontodimas, Ippokratis Sartzetakis, Polyzois Soumplis
AUTH	Argyris Kokkinis, Kostas Siozios
USTUTT/HLRS	Dmitry Khabi
MLNX	Juan Jose Vegas Almos
NBFC	Annastassios Nanos
INTRASOFT	Paraskevas Bourgos, Makis Karadimas

Internal Reviewers

Ferad Zyulkyarov, INB

Vassiliki Andronikou, INNOV

Panagiotis Kokkinos, ICCS

Abstract: The deliverable presents SERRANO project's the state-of-the-art analysis regarding the different technologies being advanced with the projects' activities.

Keywords: state-of-the-art, cloud and edge performance, machine learning, interoperability, edge acceleration, resource and service orchestration

Disclaimer: *The information, documentation and figures available in this deliverable are written by the SERRANO Consortium partners under EC co-financing (project H2020-ICT-101017168) and do not necessarily reflect the view of the European Commission. The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The reader uses the information at his/her sole risk and liability.*

Table of Contents

List of Figures

List of Tables

Abbreviations

AES-XTS	Advanced Encryption Standard-XEX Tweakable Block Ciphertext Stealing
API	Application Programming Interface
ASIC	Application Specific Integrated Circuits
CNN	Convolutional Neural Networks
CSV	Comma Separated Values
D	Deliverable
DIAs	Data Intensive Applications
DNN	Deep Neural Networks
DevSecOps	Development, Security and Operations
DSP	Digital Signal Processor
DoW	Description of Work
EC	European Commission
ETL	Extract, Transform, Load
FaaS	Function as a Service
FPGAs	Field Programmable Gate Arrays
HDF5	Hierarchical Data Format v5
HW	Hardware
JSON	JavaScript Object Notation
MPSoC	Multiprocessor System on Chip
NIC	Network Interface Card
NVMe	Non-Volatile Memory Express
ONNX	Open Neural Network Exchange
PM	Project Manager
PMML	Predictive Model Markup Language
PO	Project Officer
QEMU	Quick Emulator
SDN	Software Defined Network
SDSoC	Software Defined System on Chip
SOTA	State-of-the-Art
SW	Software
TLS	Transport Layer Security
TPM	Trusted Platform Module

1 Executive Summary

This document aims to describe a comprehensive state-of-the-art analysis regarding the different technologies being advanced with the project activities focusing on hardware and software acceleration using Cloud and Edge resources. Resource and service orchestration enhanced with Machine Learning and Artificial Intelligence, large scale intelligent telemetry for dynamic environments and cloud security (from trusted resources execution to secure storage). Moreover, each use case related topic is analysed in comparison with similar findings from relevant scientific literature. The outcome of this deliverable will be used as a guideline for the research activities regarding the SERRANO platform (WP3-5) and SERRANO Use Cases integration, development and evaluation (WP6), to focus on overcoming the current identified technical and scientific boundaries.

This document is structured based on the objectives of the project, resulting in four main topics. These topics are further analysed from different perspectives. Research directions and related work outline the current interest and findings from a particular research topic using various resources like past and ongoing research projects or relevant scientific and technical publications. Finally, last perspective presents the steps envisioned to go beyond the state-of-the-art.

2 Introduction

2.1 Purpose of this document

This deliverable presents the outcomes of the *Task 2.1 – State-of-the-Art Analysis* of *Work package 2 - Requirements and System Design* of the SERRANO project. This document presents the state-of-the-art analysis (SOTA) regarding the different technologies being advanced with the activities of the project.

The objective of this deliverable is to build a fine-grained picture of existent technologies, standards and outcomes from past and ongoing research projects, scientific publications or other important scientific and technical resources, in the field of hardware and software acceleration using Cloud and Edge resources, resource and service orchestration enhanced with Machine Learning and Artificial Intelligence, large scale intelligent telemetry for dynamic environments and cloud security (from trusted resources execution to secure storage), relevant or related to the SERRANO project objectives.

SERRANO general objectives are:

O1. Define an intent-driven paradigm of federated infrastructures consisting of edge, cloud and HPC resources;

O2. Develop security and privacy mechanisms for accelerated encrypted storage over heterogeneous and federated infrastructures;

O3. Provide workload isolation and execution trust on untrusted physical tenders;

O4. Provide acceleration and energy efficiency at the edge and cloud;

O5. Cognitive resource orchestration and transparent application deployment over edge/fog-cloud/HPC infrastructures;

O6. Demonstrate the capabilities of the secure, disaggregated and accelerated SERRANO platform in supporting highly demanding, dynamic and safety-critical applications.

Based on the objectives of the project, the SOTA is structured in four main topics:

- Security in disaggregated Cloud and Edge infrastructures
- Hardware and software acceleration in cloud/edge
- Resource and Service orchestration in disaggregated computing infrastructures
- SERRANO Use-Cases

where for each topic we discuss about the current research direction and related work (including relevant tools comparison) and a short conclusion in the form of the beyond the state-of-the-art to outline the steps that each topic will follow to extend or improve the current SOTA.

The outcome of this deliverable will serve as a guideline for each research or development activity conducted in the technical work packages (WP3-5) and the UCs development, evaluation and integration activities under WP6. Based on the analysis presented in this document each aforementioned work package will have to ensure that the results of its specific activities will go beyond the SOTA and offer relevant improvements over the analysed scientific and technological aspects.

2.2 Document structure

The present deliverable is split into nine chapters:

- Executive summary
- Introduction
- Security in disaggregated Cloud and Edge infrastructures
- Hardware and software acceleration in Cloud/Edge
- Resource and Service orchestration in disaggregated computing infrastructures
- SERRANO Use-Cases
- Overview of technologies and projects
- Conclusions
- References

2.3 Audience

This document is publicly available and should be of use to anyone interested in the current state-of-the-art of on hardware and software acceleration using Cloud and Edge resources, resource and service orchestration enhanced with Machine Learning and Artificial Intelligence, large scale intelligent telemetry for dynamic environments and cloud security (from trusted resources execution to secure storage) with respect to the SERRANO project objectives. Moreover, use cases (UCs) state-of-the-art analysis could offer interesting technological and scientific insights related with the specific business topics overviewed by the UCs partners.

3 Security in Disaggregated Cloud and Edge Infrastructures

3.1 Secure storage across heterogeneous networks

Research directions and related work

Secure storage across a heterogeneous network of storage locations brings several benefits compared to a purely cloud-based or an on-premises approach. Solutions centred on a single cloud provider's storage service are simple but may be lacking in terms of reliability, availability, cost and performance. Multi-cloud solutions address many of these issues but cannot match the performance (especially the low latency) offered by on-premises storage. While on-premises solutions were the golden standard for many decades prior to the advent and proliferation of cloud computing, they have many disadvantages in terms of up-front costs, reliability and availability [1]. Also, they are not designed with sharing outside the organization in mind. A sensible middle-ground is desirable, a heterogeneous, in some sense two-tiered approach that supports storage on both cloud and edge resources.

By exploiting the individual characteristics of the locations, each can be used according to its own strengths. Therefore, one key line of investigation concerns finding the best data distribution strategy for any storage task. In more practical terms, we can define the problem as a set of decisions that must be made when storing data. Armed with knowledge about the requirements applications have towards the data, the task is to select a set of storage locations, an erasure code and its parameters as well as an encryption schema. We refer to the configuration item that describes these choices as the storage policy. Should the requirements change, an open question is whether the policy can be adapted, or the data moved to another policy efficiently.

Overall, the most important consideration to make when providing a given Quality of Service (QoS) is the careful definition and selection of storage policies. This is in a sense an expression of the desire to move the data close to where it is created and consumed. However, if an erasure code that has the Maximum Distance Separable (MDS) property or replication is used, a second, real-time choice can be made about which storage locations to access from a given policy. For example, the closest (from a network architecture or geographic perspective) or the cheapest could be used depending on the requirements of the application.

Finally, it is wise to treat storage locations, especially those hosted by 3rd party cloud providers as benevolent, but curious. In this sense, to provide a privacy guarantee, no data should be stored or transferred from its source without first encrypting it. The task is to provide trusted storage over untrusted storage providers. A good way to accomplish this is to employ end to end encryption, complemented by other techniques such as network coding.

The use of multiple cloud storage providers and its improvements in security and performance was first demonstrated by Sipos et al [2]. SERRANO partner Chocolate Cloud developed a cloud

storage service (skyflok.com) and made it available commercially in Q1 2018, enabling its customers to choose the cloud storage locations around the world that best suit their needs, while preventing each individual provider from knowing about the stored content. This followed the principle of providing secure storage over untrusted and heterogeneous networks and storage providers advocated by Oliveira et.al [3]. Section 7.2 presents an overview of SkyFlok as well as competing solutions and describes how some of the ideas presented here are applied in practice.

The storage policy as a concept from SkyFlok can be interpreted as a practical representation of a solution to the data distribution problem. Given storage locations with various characteristics such as cost, capacity, reliability, performance and so on, what is the best way to distribute erasure coded fragments across them? Given an erasure code, what parameters should it have in terms of how much redundancy it should create and how should the fragments be spread across the nodes? This problem has been explored previously [4] in a multi-cloud scenario, though with some limiting assumptions on the nature of the cloud providers. It also appears in many other applications, for example in the context of computational grids [5]. Let us assume that a user application can present its requirements in a formal, abstract manner and it can also express its willingness to trade off achieving certain requirements over others (for example it is willing to pay more for better availability or

```
default_storage_policy:
 locations:
   bucket_ids: [14, 83, 132, 35, 82, 26]
   # Bucket IDs correspond to storage buckets
   # at cloud-based providers (e.g. AWS S3)
   # or edge storage locations

  encryption:
   enabled: True
   type: "AES-GCM-256"
  erasure_coding:
   type: "RLNC"
```

Figure 1: Example of storage policy configuration file

latency). In such cases, an optimization problem may be formulated. The solution is the policy that best matches the requirements to the optimal data distribution. Figure 1 presents a practical realization of a storage policy as defined in a YAML file.

Erasure coding is first and foremost a technique to ensure that the availability and longevity of data is maintained in distributed storage systems. Whenever a storage node fails, the rest still contain sufficient encoded fragments to recover the original data. The distribution of fragments to separate nodes forces an attacker to break into several storage nodes to be able to recover the full data. Of special interest to secure storage is Random Linear Network Coding (RLNC) [6], an erasure code that has special properties which help improve the confidentiality

of data. Coded fragments are created using randomly selected linear coefficients. While not an encryption technique, if the coefficients are not reused and the PRNG meets certain criteria, then RLNC can work similar to a one-time pad. In reality, these assumptions are usually not met due to practical considerations. Despite this, RLNC does add a second level of security compared to a conventional erasure code like the widely used Reed-Solomon. We use the terms RLNC and network coding interchangeably. In truth, the former is one implementation of the later that is well-suited for protecting data that is being transferred or at rest.

A closely related technique is that of regenerating codes, first introduced by Dimakis et al in their seminal paper entitled: Network coding for distributed storage [7]. Regenerating codes have a big advantage compared to Reed-Solomon [8] in terms of the efficiency with which lost data can be repaired. When using replication to provide redundancy in a system, repairing data is simply a case of creating a new copy whenever a replica is lost or becomes unavailable. Compared to this, conventional erasure codes like Reed-Solomon must first gather sufficient fragments to decode the data before a new fragment can be created. This leads to an explosion in the amount of repair traffic [9]. Regenerating codes like network coding avoid this by creating new linear combinations to replace lost fragments through a technique called recoding. Importantly, the new combination is made up of fewer fragments than would be necessary to decode. Interestingly, the replacement fragment is not an exact copy of the lost fragment, but rather a functional equivalent.

The question of how to repair data is an important aspect for any distributed storage system. Even more so when talking about storage across heterogeneous devices with varying levels of availability and reliability. Employing the aforementioned techniques in such a way that the system is not bogged down by repair traffic is an open research area. To this end, simply selecting storage locations is not sufficient to achieving good QoS; how redundancy is achieved and maintained is another important factor.

A closely related question to the repair problem is how a distributed storage system can adapt its data distribution when faced with changing requirements. In some cases, simply moving data from one location to another is a sensible choice. This allows the data to follow the user or to avoid a storage location that has become unreliable or is set to be decommissioned. In other cases, this is not sufficient, and the way redundancy is spread across the locations should change. One possible cause could be the requirements for the data, or the characteristics of the storage locations change. Previous work [4] has explored how network coding can be used to achieve this goal. It presents the problem in the context of speeding up downloads when faced with heterogeneous clouds that in term change their characteristics over time. Figure 2 gives an overview of how this is achieved through a data distribution that stores more of the data on the faster clouds and less on the slower ones. The blue erasure coded packets can be seen as a minimum amount necessary to ensure reliable operation in the event that one or more storage clouds become unavailable. These are created when the data is first uploaded. The green erasure coded packets are added dynamically with the goal of improving retrieval performance. If a cloud slows down compared to the others, green packets are ‘moved’ through recoding to the faster ones.

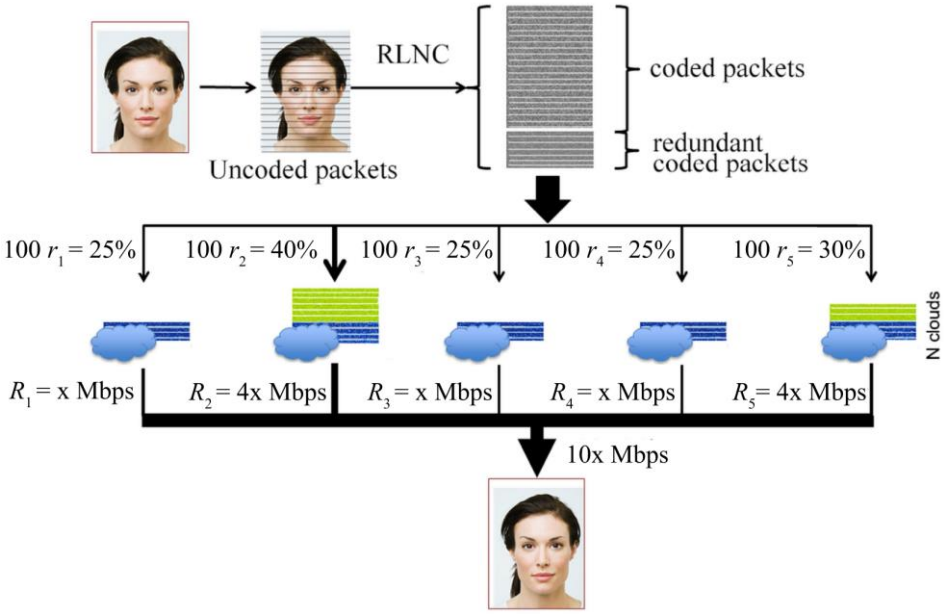


Figure 2: Example of data distribution created to enhance performance in a storage system with heterogeneous clouds through the use of network coding.

Besides the theoretical problems that have been mentioned, SERRANO will look at more practical matters as well when it comes to designing a secure distributed storage system that works across heterogeneous devices. A key enabler of this technology is containerization [10]. It underpins much of the public cloud infrastructure and is also a good choice when selecting a method to deploy software to edge resources. Reliably running and monitoring edge storage locations is key to ensuring good availability, durability and performance. Fortunately, well-established technologies such as Docker Swarm[1] and Kubernetes[2] solve this issue. Still, it is not obvious how such systems should be employed when serving a distributed storage system that relies on heterogeneous storage nodes.

Another technical question is the selection of interface that a distributed storage system should provide. The first large-scale public cloud offering was Amazon Web Services[3]. One of its first services was the Amazon Simple Storage Service (Amazon S3 for short), a versatile object store. The service's popularity means many applications have been designed with its interface in mind. Furthermore, many storage systems offer S3-compatibility either natively through a gateway like Ceph[4] or through middleware like OpenStack Swift[5]. These offer 'compatibility' by supporting the most used S3 API calls. Given these circumstances, a distributed storage system that adopts or at least supports parts of the S3 interface will be easier to integrate with, a key market advantage. The question of which subset of features to

[1] Docker Swarm - https://docs.docker.com/engine/swarm/
[2] Kubernetes, production-grade container orchestration – https://kubernetes.io/
[3] Amazon Web Services – https://aws.amazon.com/
[4] Ceph Object Gateway S3 API – https://docs.ceph.com/en/latest/radosgw/s3/
[5] OpenStack Swift – Object Storage with the S3 API - https://docs.openstack.org/newton/config-reference/object-storage/configure-s3.html

support is a more difficult question and is tied both to the requirements of the applications using the service and its internal operation.

Beyond state of the art

We foresee SERRANO advancing the state of the art in the area of secure storage over heterogeneous networks in several ways.

It will provide a theoretical solution to the problem of how to optimally spread erasure coded data across cloud and edge locations. Such a solution will take into consideration the heterogeneity of the storage resources at hand (reliability, cost, storage capabilities, latency by user location, bandwidth capabilities, etc.).

It will improve knowledge on how to design a practical system. In particular, we will evaluate whether network coding, well known to work in challenging high-churn environments, is suitable for heterogeneous storage, in particular the two-tiered system we are proposing. Other considerations including how to deploy the software on edge devices, what types of encryption to offer and so on, will also advance the state of the art for this hybrid, heterogeneous scenario.

We expect results for several practical question as well. For example, how should edge nodes be integrated into the system, is it possible to containerize their software and have it orchestrated by the platform? When offering an S3-compatible API, what minimal subset should be included such that it is possible to integrate with real-world applications?

The Secure Storage use case will showcase the ideas that have been presented in this subsection and will seek to answer these questions.

3.2 Self-encrypting NVMe-over Fabric JBOF

Research directions and related work

Non-volatile memory express (NVMe) over Fabrics (NVMe-oF) is an emerging technology used to communicate between a host and a storage system over a network (the fabric). It employs remote direct memory access (RDMA), which means, there is a drastic reduction on CPU utilization by creating direct pipeline between the network switch and the memory element. As any network protocol, it can be used to access a simple feature-less storage box (JBOF).

To enhanced storage security, self-encrypting drive (SED) or hardware-based full disk encryption (FDE) is a widely available method to maintain the symmetric encryption key independently from CPU, thus removing computer memory as a potential attack vector.

In future cloud, storing will be implemented through “just a bunch of flush” (JBOF); a box holding several flash devices that connect to a network and acts as remote storage system. In SERRANO, MLNX targets to take the SED concept to JBOF networks using NVMe-oF as the network protocol and NVMe flash as the physical drives.

To achieve this challenging task, SERRANO uses Mellanox's system on chip BlueField-2. This will enable to implement in-line AES-XTS encryption of data as it comes from the network during a WRITE operation to the JBOF, saving the encrypted data on disk. In a later READ operation from the JBOF, the encrypted data is fetched from the disk, and it is sent over the network to the client, where decryption occurs.

This approach enables to reduce operational costs in terms of capital and energy consumption, and reduce the overall latency, as data is encrypted by a single network adapter. One encryption key can be used to store data on all disks, while many independent encryption keys may be maintained regardless of the disks. This opens the door to new digital services, such as massive storage deletion by deleting the crypto key (rather than the data itself), with the subsequent savings in CPU usage and simplicity of the algorithms.

NVMe is extremely relevant for AI driven applications, as inherently it reduces the I/O time and hence improves training and analytic algorithms processing time [11]. NVMe is also being implemented with persistent memory blocks, which is a new type of memory as fast as RAM but yielding non-volatile behaviour [12]. Architecturally, the logic for NVMe is physically stored within and executed by the NVMe controller chip that is physically co-located with the storage media, commonly these days an SSD. By its design, NVM Express allows host hardware and software to fully exploit the levels of parallelism possible in modern SSDs. As a result, NVM Express reduces I/O overhead and brings various performance improvements relative to previous logical-device interfaces, including multiple long command queues, and reduced latency. The previous interface protocols were developed for use with far slower hard disk drives (HDD) where a very lengthy delay (relative to CPU operations) exists between a request and data transfer, where data speeds are much slower than RAM speeds, and where disk rotation and seek time give rise to further optimization requirements.
For the purpose of reducing the overall memory utilization, hardware offloading of the memory operations is desirable. The following figure highlights the high-level topology architecture of such attempt on a smart network interface card.

In this context, NVMe SNAP (Software-defined Network Accelerated Processing) enables hardware virtualization of NVMe storage (Figure 3). NVMe SNAP brings virtualized storage to bare-metal clouds and makes composable storage simple. It enables the efficient disaggregation of compute and storage to allow fully optimized resource utilization. NVMe SNAP logically presents networked storage, such as NVMe over Fabrics (NVMe-oF), as a local NVMe drive. This allows the host OS/Hypervisor to use a standard NVMe-driver instead of a remote networking storage protocol. The host benefits from performance and simplicity of local NVMe storage, unaware that remote Ethernet or InfiniBand connected storage is being utilized and virtualized by NVMe SNAP. Furthermore, SNAP may apply sophisticated logic and data protection mechanisms (mirroring, compression, data-de-duplication, thin-provisioning, encryption, etc.) to the network storage that it virtualizes as local NVMe.

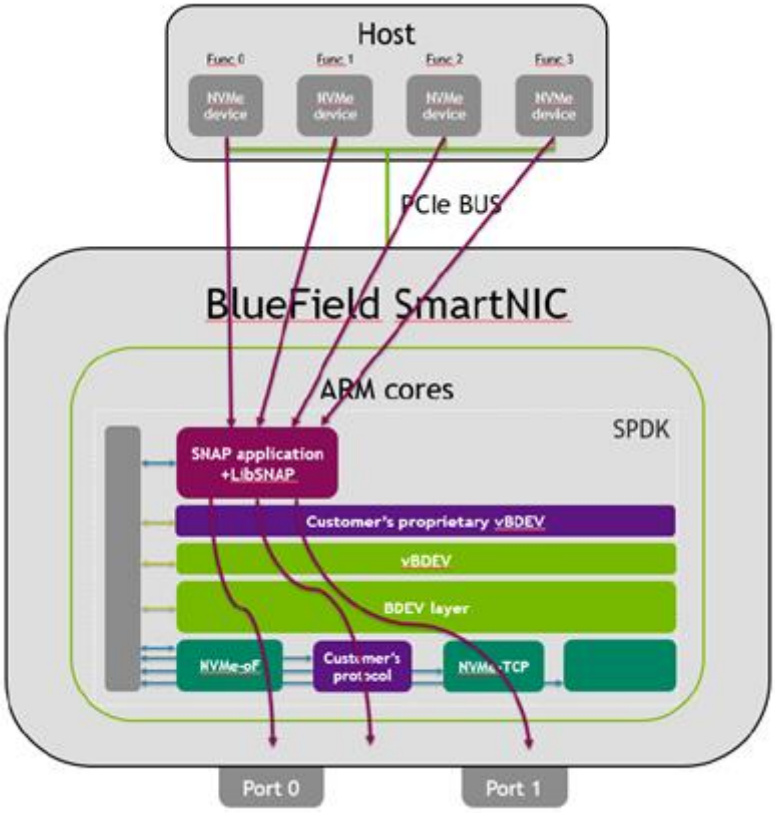


Figure 3: SNAP application in a DPU smartNIC handling memory utilization

Beyond state of the art

NVMe-oF target offload (Figure 4) is a hardware feature that terminates the target-side NVMe-oF protocol and apply the received IO requests on a set of NVMe drives accessible from the PCIe bus using peer-to-peer PCI communication. This hardware feature replaces the software flow implemented in many NVMe-oF targets: receiving the NVMe-oF I/O request from the network controller, parsing it, generating a block request and posting it on the relevant NVMe device.

Using this feature, the host software is completely not involved in the data path of serving those NVMe-oF I/O requests, and CPU cycles can be used for other activities. The system will allow to perform three distinct operations:

- **Integrity offload**: In high-end storage it is required to keep integrity field with each block of data. This integrity field holds a checksum calculated from the block of data, and possibly other metadata on the block, so that when the block is later retrieved from the storage it can be verified that nothing changed it. This offload adds, verifies, removes and transforms such integrity fields, on the path of sending and receiving data to/from NVIDIA-Mellanox NICs.

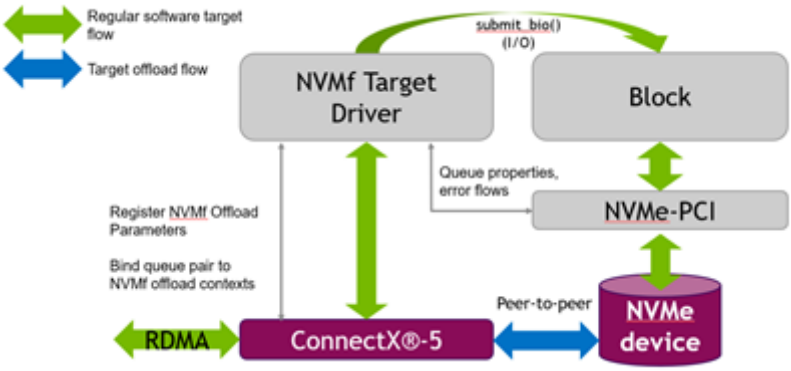


Figure 4: NVMe-oF Target Offload

- **AES-XTS storage crypto**: This crypto offload engine allows encrypting and decrypting data for storage purposes, on the path of sending and receiving data to/from NVIDIA-Mellanox NICs. It supports practically unlimited number of data encryption keys (DEKs), which makes it good fit for multi-tenancy use cases: each tenant and each volume exposed to a tenant may have a dedicated DEK for encrypting the data.
- **Compression and decompression**: These engines allow offloading some compression and decompression algorithms to the hardware. They operate in a memory-to-memory manner.

These features will be studied and expanded in SERRANO.

3.3 Trusted and isolated execution in multi-tenant edge nodes

Research directions and related work

Edge computing brings memory and computing power closer to the location where it is needed. In edge computing systems, computation is rather offloaded to nearby resources than to the cloud, due to latency reasons. However, the performance demand in the edge grows steadily, which makes nearby resources insufficient for many applications. Additionally, the number of parallel tasks in the edge increases, based on trends like machine learning, Internet of Things, and Artificial Intelligence. This introduces a trade-off between the performance of the cloud and the communication latency at the Edge. Furthermore, the need to be energy efficient in a mobile environment is high. The Edge computing paradigm, nowadays, employs single-tenant tasks, scheduled in resource-constrained devices, needing specialized Operating Systems (OSes) to host these tasks.

To be able to move to a multi-tenancy execution model at the Edge, the system must ensure non-interference and controlled data access among different and possibly concurrently running applications. To this end, virtualization plays a significant role in enabling secure multi-

tenancy execution at the Edge. However, adding another layer of abstraction may facilitate unified execution frameworks but complicates the execution stack and consumes resources for ensuring correct management, isolation and resource sharing among tenant workloads.

To alleviate the significant overhead of adding a full virtualization stack (hypervisor & Virtual Machines) at the Edge, the research community has proposed the use of container technology. However, this execution model increases the attack surface, as it exposes the full OS and runtime layer to any malicious or compromised application running in a container. Related works and critical security advisories have pointed out that containers are far too insecure for multi-tenancy. Recent works have introduced a new type of resource virtualization, bringing the benefits of isolation and secure execution, without the burden of a full virtualization stack to support generic Operating Systems. To attack the security issues that arise from multi-tenancy and remote/Edge environments, while efficiently utilizing resources, we tackle these two major issues separately:

Attack surface: SERRANO minimizes the attack surface for applications running at the Edge by leveraging the minimal exposure of trusted code to the application. Specifically, for security critical workloads, we build on unikernel technology to minimize the privileged code being accessed from applications and Virtual Machines images. To this end, we develop an efficient and secure mechanism for applications to be executed as part of a self-contained machine image, keeping only the needed dependencies, reducing the execution and exposure of trusted/kernel code to the minimum.

Trusted execution: We harden the hypervisor and Virtual Machine Monitor (VMM) framework using hardware and software techniques. Specifically, we employ a strict security attestation mechanism at the lowest level of the software stack (VMM and OS-glue to the application), to ensure that the workloads running are legitimate. Additionally, we enhance this framework with the trusted boot extensions offered by the platform (TPM or similar).

The Cloud computing paradigm appears ideal to deploy and manage application execution at scale. To support the cloud computing execution model at the Edge, devices need to support virtualization and run a full hypervisor stack.

Apart from scarce resources, Edge environments are susceptible to information leakage and exploitation. Therefore, security concerns arise, showcasing the need for controlled access to data, keeping the execution of tasks isolated and self-contained. Application execution at the edge can be categorized as follows: (i) management tasks that refer to resource provisioning (compute / network / storage) for a specific service offered by upper layers of the stack and (ii) compute tasks that refer to compute-intensive operations that provide the service needed by the end customer. These tasks may refer to data processing functions for inference in Machine Learning models, or even graphics rendering for gaming needs of mobile customers. Both categories present the need for elastic deployment: the system should be able to execute these tasks autonomously, auto-scale when needed, spawn and teardown the execution almost instantly. Therefore, the execution model for these kinds of operations resembles a lot the one that is recently introduced for lightweight function execution in the Cloud, Function-as-a-Service (FaaS), or serverless computing.

Serverless Computing is a cloud computing execution model in which the provider takes care of the hardware and software resources to execute a function / snippet of code provided by the user. Serverless computing simplifies the process of deploying code into a production system. Scaling, capacity planning and maintenance operations may be hidden from the developer or operator.

Beyond state of the art

The system software that is being developed for the SERRANO architecture encompasses the above technologies to form a unified, secure and efficient software stack that provides both strict security guarantees, while enabling the Serverless Computing paradigm at the Edge. Specifically:

- SERRANO enhances current security mechanisms to enforce trusted boot and secure access to data via NBFC's custom hypervisor stack (hybrid lightweight, fully featured VMMs such as AWS Firecracker, QEMU or NBFC's hypervisor implementation, KVMM).
- SERRANO employs an ultra-fast workload instantiation mechanism due to lightweight virtualization mechanisms, both available as open-source components (AWS Firecracker) and developed in the bosom of the project (KVMM).
- Using hardware/software co-design approaches, SERRANO enables hardware acceleration for tasks running isolated on lightweight VMMs, allowing for secure, low-latency responses for isolated multi-tenant compute-intensive tasks running at the Edge.
- Engaging in the Serverless computing paradigm, scaling is realized horizontally, while enabling task/workload migration between Edge nodes, due to the stateless characteristics of the deployable workloads.
- SERRANO completes the systems software stack by employing a unified, end-to-end orchestrator, tailored to specific workloads and resource demands, spawning unikernels, containers or Virtual Machines, with or without hardware acceleration capabilities, depending on the tasks' needs.

4 Hardware and Software Acceleration in Cloud/Edge

4.1 Acceleration for processing large data volumes of data intensive applications (DIAs)

4.1.1 Design software (SW) and hardware (HW) IP kernels

Research directions and related work

The purpose of this section is to survey the state of the art for development of hardware accelerators aiming at devices such as GPUs, FPGAs and several approximation techniques specifically in the FPGA domain. Also, it states beyond state-of-the-art methodologies towards creating a library of hardware accelerators stored as an IP repository that will interface with the SERRANO resource manager and scheduler to select different versions/approximations of kernels based on the selected device for the required task depending on the resource, performance or power requirements. For this deliverable we choose to analyse Xilinx FPGAs and Nvidia GPUs as types of devices and associated tools. The choice is motivated by the current market share, uptake in major cloud providers and impact of the novel compute elements would have on the server and edge systems.

Hardware accelerators in the cloud: Here devices such as Xilinx Alveo family or Nvidia data centre devices with often large resources and memory are usually connected via PCI-Express to the CPUs in a server [13] [14] [15]. An interface layer in the form of a shell design in the FPGA or a driver for the specific GPU along with a corresponding software component consisting of necessary libraries and executing on the server(s) are provided by the hardware or tool manufacturers. Hardware accelerators in data centres is an emerging topic that has recently gained attention by major cloud computing vendors. Applications ranging from accelerating basic signal processing functions such as FFTs [16], data encryption such as AES [17] or even Deep Learning applications [18] [19] are constantly being developed using emerging technologies from both software and hardware domains.

Hardware accelerators in the edge: Here low-power low-cost embedded accelerators in the form of a System on a Chip (SoC) with several hardware components (CPU, RAM, ports, connectors, etc.) packed in a board are used specifically for local data processing, faster decision making or filtered data transfer to the cloud. These devices (such as the Xilinx MPSoCs, or Nvidia Jetsons) do not have the raw power of the larger cloud devices but are becoming very appealing specifically for use cases like local data analytics, inferencing and machine learning where low latency and power consumption are needed. Applications can range from anomaly detection [20], inference in Deep Neural Networks [21] [22] or smart city IoTs [23].

Hardware accelerators have gained significant traction in the last years especially with the increasing complexity and variety of Machine Learning and Deep Learning applications that are used in everyday life. Devices such as GPUs or FPGAs are proven to tackle problems where reduced power consumption, lower latency or increased parallelism is needed. Their computational and communication capabilities have drastically improved in recent years owing to advances in semiconductor integration technologies. Additionally, new toolchains are being developed for seamless development of hardware accelerators that are integrated in various applications both in edge and cloud domains. Below is a list of tools developed by two of the major companies found in the FPGA and GPU industry (Xilinx, Nvidia).

Xilinx Vitis framework: The Vitis environment (Figure 5) includes a comprehensive core development kit to seamlessly build accelerated applications for both cloud and embedded Xilinx FPGAs. It's the new unified platform of Xilinx that merged the previous generations of frameworks (SDSoC for embedded FPGAs and SDAccel for cloud FPGAs) and further reduced the FPGA programming effort. The software component, or host program, is developed using C/C++ to run on x86 or embedded processors, with OpenCL API calls to manage runtime interactions with the accelerator. The hardware component, or kernel, can be developed using C/C++ and HLS, OpenCL C, or RTL. The Vitis software platform promotes concurrent development and test of the Hardware and Software elements of a heterogeneous application using several tools. Key components:

- **Vitis AI Development Environment:** The Vitis AI development environment is separate specialized development environment for accelerating AI inference on Xilinx embedded platforms or Alveo accelerator cards. It supports the industry's leading deep learning frameworks like Tensorflow and Pytorch offering comprehensive APIs to quantize, optimize, and compile trained neural networks to achieve fast inference.
- **Vitis Accelerated Libraries:** Set of open-source hardware-accelerated core libraries targeting data analytics, quantitative finance, security and many others that can be integrated in a C/C++ or Python application with minimal changes.

Xilinx Runtime library and platforms: Xilinx Runtime library (XRT) facilitates communication between the application code (running on an embedded ARM or x86 Host) and the accelerators deployed on the reconfigurable portion of PCIe based Xilinx accelerator cards (such as Alveo U200, U50) or MPSoC based embedded platforms (such as ZCU102, ZCU104).

Similarly, there are corresponding toolsets for the GPU domain. Next, we summarize the state-of-the-art tools for this purpose:

- **Nvidia Cuda Toolkit**: A development environment for building GPU-accelerated applications, including libraries, debugging and optimization tools, a C/C++ compiler, and a runtime library. CUDA introduces support for the NVIDIA architectures (Ampere, Turing, etc.), Arm processors, performance-optimized libraries, and new developer tool capabilities.

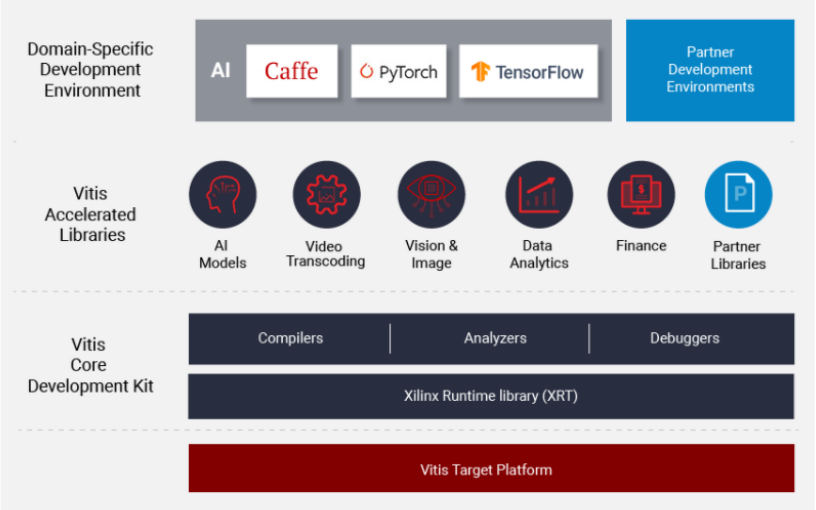
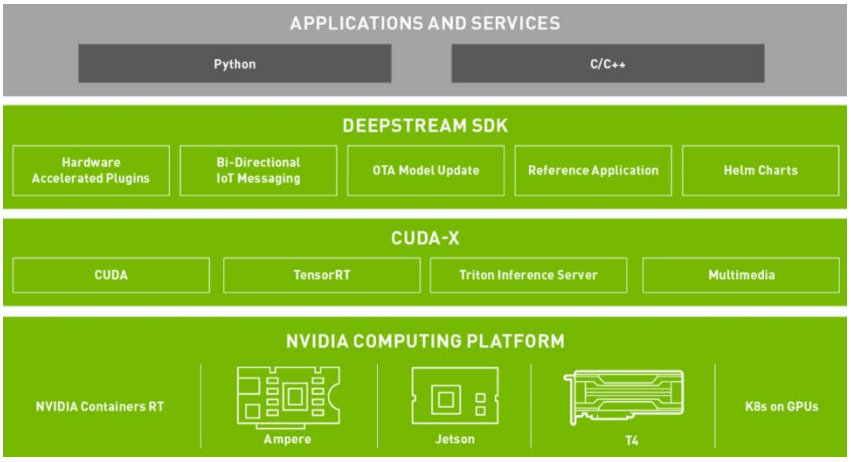


Figure 5: Xilinx Vitis Framework

- **Nvidia Cuda-X:** Built on top of NVIDIA CUDA, Cuda-X is a collection of libraries, tools, and technologies that deliver higher performance cross multiple application domains. Libraries include math functions (cuBLAS, cuFFT, etc.), Deep Learning functions (cuDNN, TensorRT, DeepStream) or even libraries from partners such as OpenCV, ArrayFire and MAGMA.

Nvidia DeepStream SDK: DeepStream (Figure 6) offers a multi-platform scalable framework to deploy on the edge and connect to any cloud the GPU-accelerated applications with a complete streaming analytics toolkit for AI-based multi-sensor processing, video, audio media supporting popular deep learning frameworks such as Tensorflow and Pytorch.



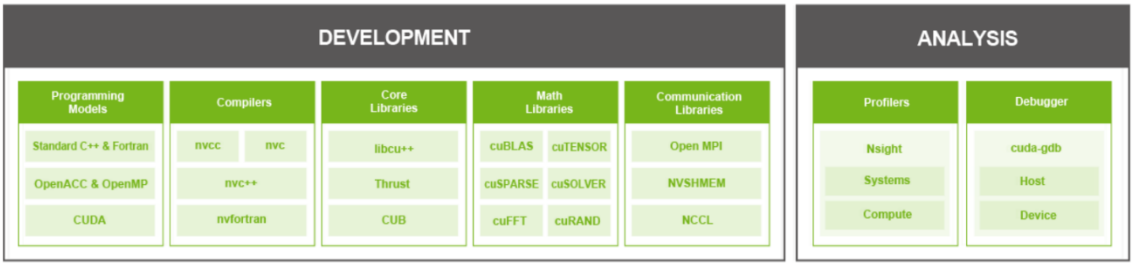


Figure 6: NVidia DeepStream SDK

Beyond state of the art

Data centres and servers must be workload-optimized to seamlessly adapt to changing throughput, latency, and power requirements from a wide range of virtualized on-demand software applications. These applications can include machine learning, video transcoding, image and speech recognition, high-performance connectivity, accelerated storage, data encryption, etc.

To support the seamless integration of heterogeneous architectures (GPUs, FPGAs) into the final platform, we will extend the Plug&Chip API to enable a very high level of abstraction on the accelerators available. We will introduce a common methodology for accelerating applications for embedded and cloud FPGAs and provide a library of IP kernels that can be selected homogeneously according to the specific task resource, performance or power requirements. The kernels will be packaged as out-of-the-box domain-specific libraries that will also support new emerging and industry standard technologies such as High Bandwidth Memory transfers (HBM), peer-to-peer (P2P) transactions between PCI devices, multiple process access, etc.

Consequently, the aim is to create a differentiated and collaborative edge-to-cloud system towards machine learning, quantitative finance and data encryption applications. A connected local orchestrator that fetches and dispatches the right IP kernels will be integrated to Plug&Chip API towards an adaptive and transparent environment that will also entail different kernel expansions for accuracy, speed, resource and power trade-offs.

4.1.2 Approximation Computing Techniques Overview

Research directions and related work

As modern applications become more and more computationally intensive, the inefficiency of traditional CPUs to provide fast, near real-time execution has led to the introduction of hardware accelerators (GPUs, FPGAs, ASICs etc.) to catch up with the performance demands. Nevertheless, these benefits do not come for free, as such devices typically require more power to operate. This limitation creates significant challenges in application design, making performance versus power trade-off a major concern.

Towards building more energy proportional computing systems, approximation techniques have been identified as a promising solution to reduce energy consumption and increase

performance while retaining the output quality of applications. Approximate computing is based on the observation that many applications feature intrinsic error-resilience properties [24] (i.e., in DSP or Machine Learning domain). In addition, certain approximate computing techniques (ACTs) can be combined with accelerators (GPUs, FPGAs etc.), leveraging the platforms' architecture for maximizing performance while satisfying the requirements for power and area footprint. In Figure 7, a classification of the ACTs, based on the feasibility of their integration on a GPU or FPGA platform, is presented.

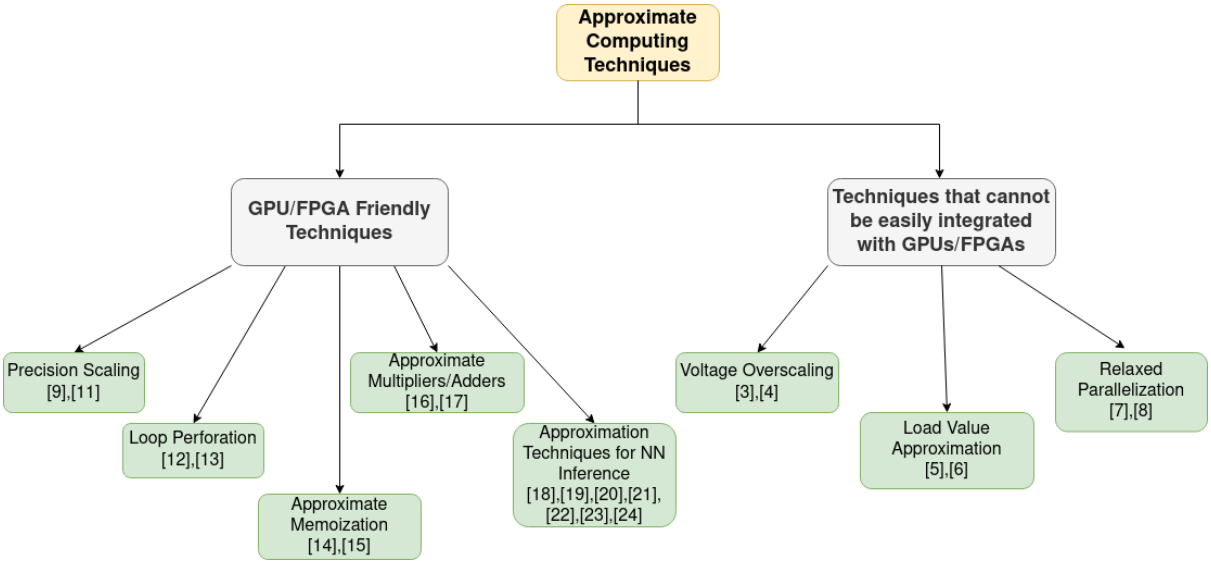


Figure 7: ARABIC 1 Classification tree of the approximate techniques based on the feasibility of their integration on GPU or FPGA platforms

Although approximate computing demonstrated its benefits in machine learning applications where the power consumption is critical in both training and inference, researchers throughout years have proposed multiple hardware/software-oriented techniques.

At a circuit level **scaling down the supply voltage** is a method that is used for designing energy-efficient systems [25]. Although reducing the reference voltage leads to a quadratic reduction in power consumption, it incurs timing errors and flipped bits. It is shown [26] [27] that when voltage scaling is applied on certain error-tolerant applications, 89% energy reduction is achieved at the cost of 20% Bit Error Rate (BER).

Similarly, techniques such as **Load-Value Approximation (LVA)** and **relaxed parallelization** are used to avoid processing stalls of the computing units. In brief, LVA method [28] [29] estimates load values allowing the processor to progress, hiding the cache miss latency [25], while relaxed parallelization is based on removing synchronization primitives of error resilient code blocks [30] [31].

GPU / FPGA Approximation Techniques Analysis. Even though the usage of the aforementioned ACTs leads to systems that utilize less resources and consume less power, their deployment in acceleration platforms is particularly challenging. On the other hand, techniques such as precision scaling, loop perforation, approximate memoization and implementing approximate multiplier/adder blocks are ACTs that can be easily applied on use

cases that leverage GPUs and FPGAs. Because these techniques are easily applicable, Design Space Exploration (DSE) methods can be exploited for determining the optimal approximate solution for each use case. In the following subsections, these ACTs are presented.

Precision Scaling. Precision scaling is a technique that changes the bit-width of input or intermediate operands to reduce storage and computing requirements. Jong Hwan Ko et al. [32] leverage the benefits of precision scaling in neural networks for audio processing achieving up to 30x processing time speedup while the performance impact degradation is less than 3.14% in the case of classification tasks. Xilinx[6] provides tools and libraries that support a wide range of fixed-point precision data types. It has been noted that FPGA design implementations using fixed point arithmetic are more efficient than their equivalent floating point due to their limited power and resource consumption. According to [33] power savings of up to 50% have been noticed for designs that have been migrated from floating to fixed point.

In Figure 8 and Figure 9, the use of precision scaling in a High-Level Synthesis (HLS) context is demonstrated through a simple example of square root vector addition. Xilinx provides *ap_int.h* and *ap_fixed.h* libraries that allow the user to define arbitrary precision integer and fixed-point data types that are tailored to the application's design. These user-specified data types are not supported in the typical C/C++ standards. In the provided example, ap_unit<10> specifies a 10 bits wide unsigned integer data type while ap_ufixed<16, 11> defines a 16 bits wide unsigned fixed precision data type with 11 bits for the integer part and 5 bits for the decimal part. Although arbitrary precision data types may lead to a power and resource efficient design, it's the developers concern to find the optimal bit width for the operands in each design.

```
float arr[1024], sum=0.0;
// Initialization
for(int i=0; i<1024; i++) {
    arr[i]=1.1234;
}
// Calculate summary of square roots of input values
for(int i=0; i<1024; i++) {
    sum+=sqrt(arr[i]);
}
```

Figure 8: Typical vector square root addition algorithm

[6] Adaptable. Intelligent – Xilinx, https://www.xilinx.com/

```
#include "ap_int.h"
#include "ap_fixed.h"

ap_ufixed<16, 11> arr[1024], sum=0.0;
// Initialization
for(ap_uint<10> i=0; i<1024; i++) {
    arr[i]=1.1234;
}

// Calculate summary of square roots of input values
for(ap_uint<10> i=0; i<1024; i++) {
    sum+=sqrt(arr[i]);
}
```

Figure 9: Vector square root addition algorithm implemented with user-specified precision data types

In Figure 10 and Figure 11, the relation between accuracy, power and resource utilization for different bit width approximations is presented for gaussian naive bayes algorithm in ZCU104 FPGA platform. An increase in decimal bits leads to an improvement at algorithm's accuracy and a deterioration at its power efficiency. For 8 decimal bits the achieved accuracy is almost equal to the floating point (79.25%) while the power consumption is 1 W 33% lower. Additionally, the resource utilization figure depicts that for up to 8 decimal bits the utilization of DSPs, FFs and LUTs [7] does not exceed 1%, 2% and 5%. Finally, for the floating-point data type, an increase in DSPs (x11), FFs (x5) and LUTs (x2.4) is observed compared to 8 decimal bits.

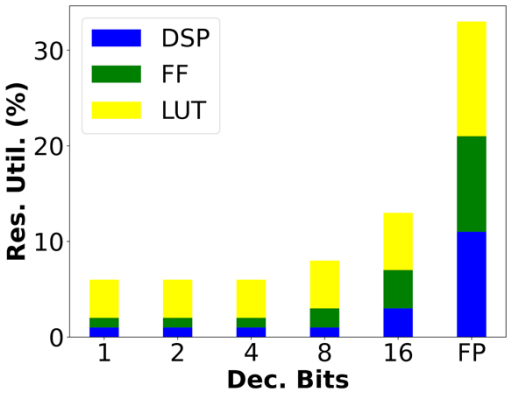


Figure 10 - Resource utilization for multiple levels of approximation

[7] Digital Signal Processing (DSP), Look-Up Table (LUT), Flip-Flop (FF) are resource types of the FPGA's programmable logic region.

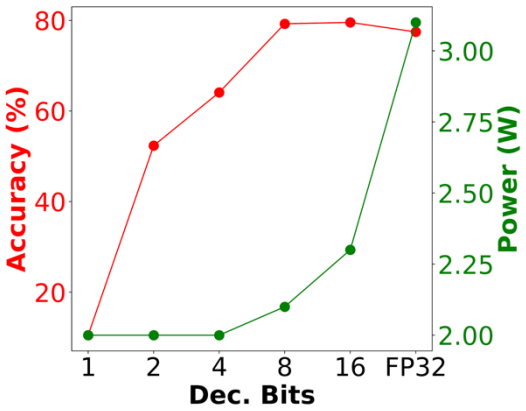


Figure 11: Accuracy vs power consumption with multiple levels of approximation

Loop Perforation. Loop perforation is another accelerator friendly method that selectively skips entire loop iterations to reduce computational overhead. This technique was introduced in software approximation, but since it is an algorithmic technique, it can be equivalently applied in a GPU/FPGA context. Koliogeorgi et al. [34] use precision scaling and loop perforation combined with HLS optimizations for a Support Vector Machine (SVM) classifier use-case and achieve x15 execution time speedup compared to a typical CPU execution while maintaining an accuracy of 96.7%. Table 1 shows the way loop perforation impacts the SVM algorithm's accuracy and speedup. For omitting up to 10% of the total loop iterations an 8.1% decrease in accuracy is observed while a speedup of 1.54 can be achieved. Furthermore, the authors in [35] propose a loop perforation space exploration algorithm that delivers performance increases of over a factor of two (and up to a factor of seven) while the drop in application's quality is less than 10%.

Table 1: Algorithm's accuracy and speedup scores at different perforation ranges

Perforation %	Accuracy %	Speedup
2 %	98.02 %	1.41
4 %	97.12 %	1.45
6 %	96.27 %	1.47
8 %	92.85 %	1.51
10 %	91.9%	1.54

In Figure 12, we provide a version of the previous algorithm, applying the loop perforation technique. In this example we skip 2 out of 1024 iterations. This technique has limitations that should be considered from the designer. For example, when the summary of a vector is calculated, the operands' range should be considered as when skipping loop iterations all additions do not have the same impact on the final result.

```
float arr[1024], sum=0.0;
// Initialization
for(int i=0; i<1024; i++) {
    arr[i]=1.1234;
}
// Calculate summary square roots of input values
for(int i=0; i<1024; i++) {
    if(i % 512==0)
        continue;
    sum+=sqrt(arr[i]);
}
```

Figure 12: Applying the loop perforation technique at the vector square root addition algorithm

Approximate Memoization. The background of this technique is to store the results of expensive function calls for later reuse and return the cached values when a similar input reoccurs. Tziantzioulis et al. [36] demonstrate that compiler-directed output-based memoization reduces energy and runtime by 50% to 75% (2-4x speedup) at the cost of a relatively small degradation in the application's output quality. Sinha et al. [37] demonstrate that memoization-based approximate computation on FPGAs achieves a significant dynamic power saving (around 20%) with very small area overhead (less than 5%) and better power-to-signal noise ratio values for a set of image-processing benchmarks.

In Figure 13, we exhibit a memoization-based example for the vector square root addition algorithm, using pre calculated values for the square root function. In brief, the *pre_calculated_values* vector stores the results of the square root function for integers between 0 and 4. Based on the *arr[i]* value and the defined set of intervals the corresponding element from the *pre_calculated_values* vector is used instead of the typical square root function.

In another example, we applied a similar memoization-based method to calculate the exponent of the sigmoid activation function in the logistic regression algorithm. Figure 14 depicts the comparison of the exact computation versus the approximate in terms of power, accuracy, resource utilization and achieved clock period. It is shown that this approximate computing technique (ACT) leads to a 4.5% decrease in accuracy and a decrease of 41.3% in power consumption for the ZCU104 platform. Using approximate memoization technique the drop in utilization of DSPs, LUTs and FFs [8] is around 66%, 63% and 37% respectively. The utilization of fewer resources impacts the maximum achievable clock frequency leading to a clock that is 20 MHz faster than the one that can be achieved without approximation. On the other hand, the drawback of this ACT relies on the higher resource utilization of block RAMs (BRAMs) due to the storage of the precalculated values.

Approximate Multipliers & Adders. Hardware oriented approximate techniques have also been proposed. Approximate adders and multipliers modify the typical addition and multiplication process for error resilient applications. Most of the approximate integer and floating-point multiplication approaches leverage logarithmic properties to minimize the computational overhead for a given error tolerance. Saadat et al. [38] proposed a custom

[8] Digital Signal Processing (DSP), Look-Up Table (LUT), Flip-Flop (FF) are resource types of the FPGA's programmable logic region.

floating-point multiplier that achieves x57 and x28 power and area improvement respectively when used as the multiplication block in AlexNet, with no significant degradation in the accuracy.

```
int arr[1024];
float pre_calculated_values = {0.0, 1.0, 1.414214, 1.732051, 2.0}
float sum=0.0;
// Initialize arr[i]
for(int i=0; i<1024; i++) {
    arr[i]=get_random_float(0, 4); // Gets random float from 0 to 3.999
}
// Calculate summary square roots of input values
for(int i=0; i<1024; i++) {
    int index=-1;

    if (arr[i] < 0.5)
        index=0
    else if (arr[i] >= 0.5 && arr[i] < 1.5)
        index=1
    else if (arr[i] >= 1.5 && arr[i] < 2.5)
        index=2
    else if (arr[i] >= 2.5 && arr[i] < 3.5)
        index=3
    else
        index=4

    sum+=pre_calculated_values[index];
}
```

Figure 13: Applying memoization to calculate the square root values at the vector square root addition algorithm

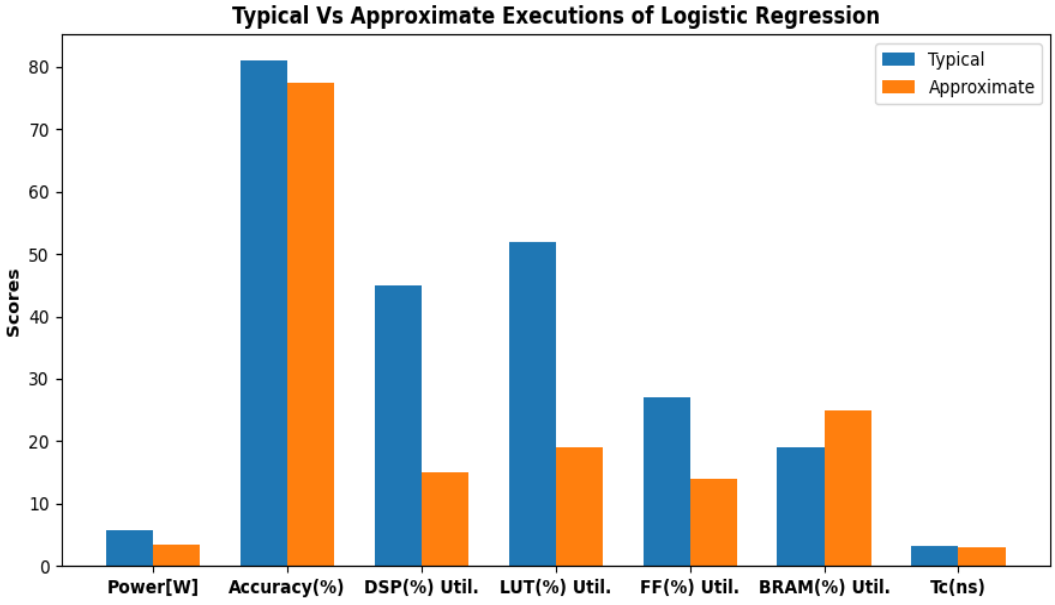


Figure 14: Overall design differences between the typical and the approximate implementations

Saadat et al. [38] proposed a custom floating-point multiplier that achieves x57 and x28 power and area improvement respectively when used as the multiplication block in AlexNet, with no significant degradation in the accuracy. Javaid et al. [39] introduce REALM, an error-configurable approximate unsigned integer multiplier, that enables significant power-efficiency (66% to 86% reduction) and area-efficiency (50% to 76% reduction) when compared

with the accurate integer multiplier. Error characterization of REALM shows that it achieves very low error bias (mostly≤0.05%), along with lower mean error (from 0.4% to 1.6%), and lower peak error (from 2.08% to 7.4%) than the classical approximate log-based multiplier and its state-of-the-art derivatives (mean errors≥2.6% and peakerrors≥7.8%).

Approximation in Neural Networks. Based on the observation that neural networks (NNs) are typically used in error-tolerant applications and are resilient to many of their constituent computations, some researchers proposed techniques to approximate their computations. Pruning is a technique to reduce the parameter-count resulting in improved speed, reduced size and reduced computation power [40]. Other techniques replace the least important neurons in a network with their approximations, trading energy for accuracy [41]. A proposal by Denton et al. [42] uses clustered filters and low rank approximation. They achieve a speedup of 2x for convolutional layers of AlexNet, compared to a non-optimized GPU implementation. Another work by Mathieu et al. [43] achieves better results by replacing convolution through FFT. Finally, the neural network compression pipeline proposed by Han et al. [44] uses pruning and weight-sharing. When this compression is applied to dense layers of AlexNet, forward propagation is 4x faster and 3x more energy efficient. In their later paper [45], they show a Titan X based GPU implementation with a throughput of 3.23 TOPs. High-end GPUs require a lot of energy. As a case in point, Nvidia's Tesla K-40 has an average power consumption of 235 W when running DGEMM [46].

Beyond State of the Art

To support different heterogeneous architectures with different levels of approximation, for each application multiple versions approximation algorithms and libraries will be implemented. These versions will target different processing elements (e.g., CPUs, GPUs, FPGAs and ASICs) and in each of these versions different levels of approximations will be performed (e.g., precision scaling, loop perforation, approximate memoization and combinations of them). The design space exploration (DSE) for optimizing both the algorithm's performance and the platform's resource and power constraints will be performed automatically, deciding the optimal design schemes and the ACTs that will be applied. This analysis will be application-specific targeting each of the SERRANO use cases (the workflow is depicted in Figure 15).

Based on the user specified performance metrics and telemetry data, a runtime mechanism will be able to deploy the specific application version (provided by our automatic DSE subsystem) to the device that optimally meets the given requirements. It will also be able to swap among different program versions to fulfil the current design goals. For example, if a deep learning (DL) inference task requires a maximum accuracy of 85% and through telemetry is identified that the achieved accuracy reaches 95%, another version of the same task can be deployed (that decreases accuracy by up to 10%) in order to use a design that utilizes fewer resources and consumes less power. Using this framework heterogeneous platforms can be effectively utilized maximizing resource usage and satisfying the applications' Quality of Service (QoS) requirements.

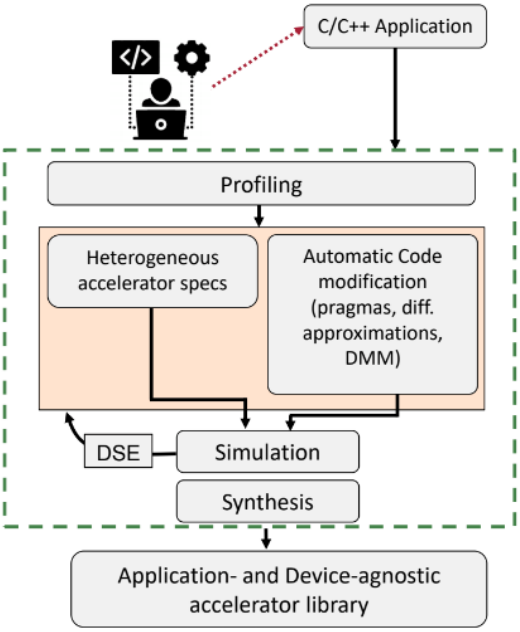


Figure 15: SERRANO use-case analysis workflow

4.2 Edge offloading

Research directions and related work

Edge-based computing aims to leverage the limited computational resources for (pre)processing data using the devices located at the network's boundary. In these cases, computing resources can range, with respect to computational power, from small factor embedded computers to complex micro datacentres [47]. As a result, we can improve applications' responsiveness by moving some of their processing elements (i.e., edge offloading) to the edge. One of the reasons this is desirable is because data processed closer to the source reduces the ingress bandwidth demands when compared to other geographically distributed solutions (i.e., cloud). Taking a decision on whether, where and what to offload from the edge is affected by several factors like task particularities, network properties or conditions or platforms' differences. Computational tasks can be executed in various places, IoT devices, edge locations or even in the cloud. The offloading destination is seen as a trade-off between the environment factors and final overall scenario objective. To minimize the costs of offloading huge amount of data, some of the computational tasks can be performed near the source, in the edge proximity where the computational power may allow data pre-processing. To tackle the challenges, Edge Clouds (also known as fog computing [48] or mobile edge computing (MEC) [49]) have been proposed. The main idea is to add resources near to the network edge; resources like computation power, bandwidth and storage resources should be moved closer to the mobile devices (or IoT devices). This approach will reduce the backbone data traffic and the round trip time latency to support resource-intensive applications hosted on the IoT. Edge Cloud are lacking the computing

capacity that a centralized cloud would have but it will excel in terms of better latency, flexible distribution and relatively better computational power than the mobile devices.

In this context, there are various efficient mechanism that allow the "offloading" of processing to some more computational powerful devices, like Edge Clouds. One way of distinguishing among various configurations is to classify based on the number of servers (single or multiple ones) that are available to perform the computation. In the single server environment, there are two classical approaches to do the offloading. MAUI [50] framework proposes to offload parts of programs remotely to preserve energy on the mobile device. Offloading is done either in the cloud or on edge devices that are near the WiFi access points or cellular networks. The proposed architecture, it is based on the client server paradigm. Both the client and the server must implement three functional components, a proxy, a solver and a profiler. In this case, a proxy is used to transmit data and control the instructions. Profiler oversees retrieving information about the programs resource consumption, while the solver oversees splitting the execution of the program over various devices/resources. The major problem identified by this approach is the cost of application development. In this case, when an application is developed parts of the code that can be offloaded to Edge devices must be marked. In a similar manner, but on a more fine-grained level, the CloneCloud [51] framework was introduced. In this framework, the work of finding what part of the program can be offloaded is performed by the framework, hence no annotations on code are required. Instead, application profiling is executed in advance. Another approach that improves the shortcomings of CloneCloud and Maui is ThinkAir [52]. ThinkAir is advancing the way applications get offloaded from cloud by introducing the concept of virtual smartphones that are capable of method level computation offloading. Yet another improvement over the existing approaches is that new devices/resources can be allocated dynamically and introduce new ways for resource allocation and distributed task handling. Another novelty was introduced with the notion of Cloudlet [53], that provided the resources on the nearest access point to the device. One of the problems that started arising was that of the ownership of the cloudlets [54], usually being owned or provided by the internet service providers (ISPs) within their local network. This approach will limit their usability because of the network constraints imposed by the ISPs and may have impact on some particular applications. Another issue identified was regarding the execution mode: the whole application is offloaded in a single VM. This could be subject of resource availability in case of multiple application offloaded on the same VM. A proposed solution would be to create elastic cloudlets. This pool of cloudlets should be create on demand based on the usage requests.

Another key aspect for this topic is the offloading balance problem. Offloading balance refers to the ability of and Edge Cloud to process the offloading requests based on the resource's availability. The amount and frequency of these requests are dynamically changing in a real time scenario and due to the resource limitation of and Edge Cloud some service level agreements (SLA) of the application cannot be fulfilled. Therefor the system needs to implement a series of scheduling algorithms to mitigate this issue. The scientific literature classifies these algorithms into online and offline based on the decision when to perform the balancing, before or after the request to offload. Online balancing is a runtime trigger that

performs the workload distribution to the appropriate resources when the request arrives at the platform. In [55] and online request admission algorithm was proposed (*Online-OBO and Online-Batch*) to maximize the throughput. A request is immediately refused if there is insufficient to fulfil the requested resources otherwise if the cost of implementing this request is below a given feasible threshold the request is accepted and implemented. Another approach would be to allocate resources geographically distributed, in small pools, close to the user or mobile devices [56]. *Primal-dual Approach* takes the request constraints, including geographic location, resource cost restrictions and latency, to achieve: (a) a balance between cost and performance; (b) to generate an optimal solution for the current request without minding the future resource requests and load; (c) flexibility on handling resource constraints. Besides the already discussed models for solving the offloading problem a *stochastic model* was proposed in [57] by studying two well-known algorithms like Best-Fit and MaxWeight, pointing out their disadvantages. Offline balancing aims to distribute the load from the over-committed resources in the Edge Clouds. When a resource is overloaded a load migration operation can be triggered to migrate part of the existent load over other less utilized available resources. One approach for offline balancing is *resource intensity aware load* (RIAL) algorithm, proposed in [58], that efficiently migrates the load against the edge cloud resources ensuring a low migration cost. To avoid overloading, RIAL periodically monitors the usage on each edge resource, and triggers a migration operation whenever a possible resource hits a threshold for resource usage. The decision on both what requests to be migrated and on what resource is done using a multi-criteria decision-making method (MCDM). To save costs in terms of energy consumption or even usage, offloading may occur also when the resources are less utilized. For example, during the night some edge resources might stay in idle or with a few requests to process. The latter case may trigger a balancing task to free up the less utilized resources to completely shut them down while not in use. But this issue will raise other challenges. The migration of processing data, memory data and so forth will eventually consume bandwidth and in most of the cases at the edge this translates into shared bandwidth. To overcome this problem, [59] proposed a *bandwidth guaranteed algorithm* aiming to solve the bandwidth competition problem by migrating the data while keeping a minimum bandwidth for the regular edge traffic.

Further, data offloading decision is influenced by several factors like network conditions, including link quality, bandwidth or network interference or data transfer costs from the source to the data warehouse. Before moving data from the edge zone, these can be pre-processed locally, using edge accelerated devices (edge devices with CPU and GPU computational power) or reduce the size of the offloaded data collected using transprecision computing. One example in this direction are anomaly detection tasks, used to detect and react to anomalous events that may occur during the life cycle of a running scenario that includes edge devices. The type of anomalies which can be detected is based on the workload of each particular instance. In SERRANO, we will tackle this point, sequential and complex/contextual anomalies from performance and quality domains. The predictive models trained to detect these anomalies once optimised using transprecise methods can be moved close to the edge nodes they monitor. It is important to note that we can and will utilise

different aggregation methods, meaning that we will use different predictive models for each detectable anomaly class (one-vs-rest strategy) and multiclass models.

Transprecision computing

Several research projects have their main research focus on transprecision computing, one of the best known and most active is OPRECOMP [60] project. This project focuses on improvements both from software and hardware point of view. This combined approach yields a significant boost in energy efficiency and is one of the pioneering research projects dealing with these issues. The OPERCOMP software stack focuses on three key objectives. The first is to offer programmers the abstractions for expressing structure and disciplined approximation of data during computation or communication. The second is to hide the complexity of a full transprecision computing hardware and system software stack from users, while addressing both architectural and intrinsic hardware heterogeneity. The third is to leverage the full range of approximate computing technologies available in the OPRECOMP hardware stack. To showcase this the project has several demonstrators that are tested using a micro benchmarking strategy inspired by [61] [62]. Key components and processing methods of real-world applications for various domains have been modified for transprecision. From these we mention:

- Deep Learning applications and their common algorithmic patterns, Convolutional Networks, Stochastic Gradient Decent, Long-Short -Term Memory (recurrent);
- Big Data and data processing, for use-cases where immediate action is required (robotics/drones, self-driving cars). Well known algorithms for unsupervised as well as supervised methods have been adapted for transprecision;
- HPC and Scientific computing methods such as Gauss-Legendre Quadrature, Fast Fourier Transform have also adapted for transprecision.

The transprecise adaptations in the aforementioned demonstrators can be largely split up into: data type adaptation, in particular low precision versions such as float (16-bit or even 8-bit); transprecise memory which lowers refresh rates thus leading to power savings; iterative transprecise concepts where exiting algorithms hyper-parameters can be tuned to give a trade-off between precision and power utilisation or even reduction in inference/training times. At this point we should clarify that transprecision is many ways tightly linked with approximate computing however, it differs in some fundamental ways. First it controls approximation in both space and time in fine grained manner utilizing several hardware and software feedback loops. Second, it does not imply the reduction of precision at the application level (although it is not excluded). Lastly, it does not strive to achieve a simple cost trade-off curve rather, a smoother and fine-grained precision curve.

Hyper-parameter optimization techniques [63] are used to improve predictive model performance by evaluating candidate solution for model optimization. The optimizations of these predictive models will enable their instantiation on hardware with limited computational resources while at the same time still providing an adequate precision and throughput. In simple terms the resulting models can be offloaded to edge nodes with minimal operational penalties. The optimization search space size is based on the target model hyper-parameter space. We can further split these methods into unguided and guided methods. The

unguided method does not evaluate each candidate solution's performance; examples of this method include the Grid and Random Search [64]. Arguably more interesting approaches are the guided ones. These generate candidate solutions using some form of evaluation (of the model performance or relationship between parameters). Guided methods include but are not limited to Tree-structured Parzen Estimator (TPE) [65], Bayesian base solutions [66] [67] and genetic algorithm-based solutions [68]. In the SERRANO context the most interesting types of optimization mechanisms are those which can utilize transprecise methods. By creating custom evaluation functions, we can more easily create tailor made predictive models with a well understood trade-off between their predictive performance and resource utilization on any given hardware and software stack. The aforementioned evaluation functions are not limited to the quantification of predictive performance but can be expanded to be energy and computational resource aware.

An area of active research regarding transprecision relates to the placement of video analytics tasks as close to the edge as possible, to minimise the data sent to the cloud. Currently, more powerful Graphical Processing Units (GPUs) such as the NVidia Jetson Nano[9] provide considerable computational capability [69]. They offer the potential to perform at the edge real-time object detection including both localisation and classification of objects by utilising deep learning-based object detectors. The "edge-only mode" [70] allows a lightweight Deep Neural Network (DNN) running in the edge. Other works for DNN study the selection of pre-trained DNNs on the fly, according to input image characteristics [71] [72].

DNNs have become one of the most utilised methods in transprecise learning as applied to edge offloading specifically in the case of inference [73]. The main reasons for this can be traced back to their versatility in solving various analytics problems and the out of core training[10] make it in many ways ideal for transprecise optimizations. Toolkits such as TensorFlow Optimization[11] have been developed specially for the task of fine tuning DNN based predictive models for edge type scenarios.

Weight clustering [74] which was first proposed by a team from ARM with hardware support for Ethos-N[12] and Ethos-U[13] ML processors improves memory footprint and inference speed. It accomplishes this by replacing similar weights in a layer by the same value. These values are calculated by running a clustering algorithm over a model's trained weights, next centroid indexes are computed, and the analysed weights are replaced with these centroid indexes. This leads to a reduction of unique float (usually float 32) values for each weight to a subset of these coupled with 2-bit indices. This methodology typically yields an inference time reduction of 4-5x and an accuracy penalty of less than 3%.

Integer quantization [75] can be applied during training or to a pre-trained model. It involves transforming the model into an equivalent representation which uses parameters and

[9] https://developer.nvidia.com/embedded/jetson-nano-developer-kit
[10] https://machinelearning.wtf/terms/out-of-core/
[11] https://www.tensorflow.org/model_optimization
[12] https://developer.arm.com/ip-products/processors/machine-learning/arm-ethos-n
[13] https://developer.arm.com/ip-products/processors/machine-learning/arm-ethos-u

computations at a lower precision. This improves execution performance and efficiency. A 8 bit integer quantization typically results in models that are up to 4x smaller and faster in computation. It should be noted that the quantization process is by its very nature a lossy process. Weight pruning [76] [77] aims to reduce the number of parameters and thus the operations involved in the computation by removing connections (parameters) between DNN layers. This method is designed to iteratively remove connections based on their magnitude during training. The pruning routine is scheduled to execute eliminating (setting to zero like dropout) the lowest magnitude values based on a "a priori" specified target sparsity (i.e., 80%).

Event and anomaly detection engine

UVT has developed an event and anomaly detection engine (EDE) designed to detect complex/contextual anomalies from large scale distributed systems [78] [79]. EDE has several sub-components which are based on a lambda type architecture[14], (see Figure 16) and can be split up into *speed*, *batch* and *serving layer*.

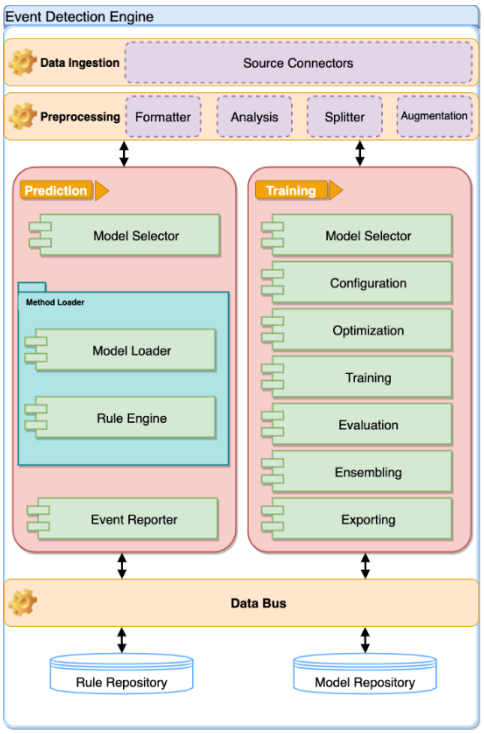


Figure 16: EDE Architecture

Because of the heterogeneous nature of most modern computing systems (including Exascale and mesh networks) and the substantial variety of solutions which could constitute a monitoring service, the data ingestion component is designed to contend with fetching data from a plethora of systems and technologies. A specialized source connector is implemented such that it serves as adapter for each supported monitoring and/or metrics storage solution. Furthermore, this component can load data directly from static file (HDF5, CSV, JSON, or even raw format). The ability to load from static sources aids in fine tuning of event and anomaly

[14] https://en.wikipedia.org/wiki/Lambda_architecture

detection methods by ensuring that model training and validation is done in a controlled and repeatable environment. Data ingestion can also be done directly via query from the monitoring solution or streamed directly from the queuing service (after ETL operations if necessary). This ensures that we have the best chance of reducing the time between an event or anomaly happening and its detection. The queueing service typically is implemented as an in memory datastore and utilizes technologies such as Redis, Kafka, Logstash etc.

The pre-processing component takes the raw data from the data ingestion component and apply several transformations to it. It handles data formatting (i.e., one-hot encoding), analysis (i.e., statistical information), splitting (i.e., splitting the data into training and validation sets) and finally augmentation (i.e., oversampling and undersampling). The training component (batch layer) is used to instantiate and train methods that can be used for event and anomaly detection. The end user can configure the hyper-parameters of the selected models as well as run automatic optimization on these (i.e., Random Search, Bayesian search, TPE etc.). Users are not only able to set the parameters to be optimized but to define the objectives of the optimization. More specifically users can define what should be optimized including but not limited to predictive performance, transprecise objectives (inference time, computational limitations, model size etc.).

Evaluation of the created predictive model on a holdout set is also handled in the training component. Current research and rankings of machine learning competitions show that creating an ensemble of different methods may yield statistically better results than single model predictions. Because of this, ensemble learning capabilities[15] are also included.

Finally, the trained and validated models must be saved in such a way that enables them to be easily instantiated and used in a production environment. Several predictive model formats must be supported, such as PMML, ONNX, HDF5, JSON. The task of event and anomaly detection can be broadly split into two main types of machine learning tasks: classification and clustering. Once a predictive model is trained and validated it is saved inside a model repository. Each saved model must have metadata attached to it denoting its performance on the holdout set, as well as other relevant information such as size, throughput etc.

The prediction component (speed layer) retrieves the predictive model form the model repository and feeds event metrics from the monitoring system. If and when an event or anomaly is detected EDE is responsible with signalling this to any reporting component as well as other tools such as resource managers and/or scheduler of the system.
Furthermore, EDE utilizes the Dask[16] framework as execution backbone. In short this means that it can be executed on a Dask cluster, each component being deployable on different Dask worker nodes. These Dask workers can run on edge nodes which in turn lets EDE prediction component to control where predictive models will be instantiated (i.e., edge offloading).

[15] https://en.wikipedia.org/wiki/Ensemble_learning
[16] https://dask.org/

Beyond state of the art

In SERRANO we will focus on creating optimized anomaly detection models utilizing transprecise methods to move the resulting models for performance and quality analysis as close to edge nodes as possible (i.e., edge offloading). The EDE system will serve as a starting point for our research, as it is easily extendable and flexible. We will focus on creating optimized DNN models as these are the most suited for transprecise methods. Furthermore, we will also investigate neuroevolutionary based methods that enable autonomous DNN topology evolution [80].

5 Resource and Service Orchestration in Disaggregated Computing Infrastructures

5.1 Intelligent telemetry for dynamic adaptive network services

Research directions and related work

Data networks are the nervous systems in any distributed infrastructure, as they provide the required interconnection between end-users, computing and storage resources. Their performance determines the efficient operation of the overall Information and Communications Technology (ICT) infrastructure and the QoS provided to the deployed applications. Any degradation in the networking operation can cause performance degradation in the applications running on top. Multiple factors can affect the network performance such as: packet loss, abnormal delay, delay variance, bad load distribution and poor behaviour of network operating systems or user applications. These factors can result in network soft (network performance degradation) or hard (network downtime) failures, which has significant costs. Therefore, it is critical to provide advanced network monitoring tools able to: reflect changes in the network state, analyse them, make predictions and dynamically take actions in response to them. As is the case with any system, a network has to be observable in order to be manageable and stable. Network Telemetry (NT) is an indispensable tool for this purpose. NT describes how data from various sources are automatically collected and transmitted to suitable equipment for analysis purposes. Indicative monitoring data set includes (but is not limited to): throughput, delay, jitter, packet loss for specific flows. NT has to fulfil four major critical requirements for the operation of a network which are:

a) Consistent high performance (e.g., in terms of throughput), which can be achieved by identifying performance bottlenecks whenever they occur.

b) Consistent high efficiency in network utilization through very dynamic and complex reconfiguration actions (e.g., perform rerouting).

c) Consistency and high availability which can be achieved by prompt failure identification and localization, or prediction of failures, if possible.

d) Robust network security able to accurately identify and address malicious attacks in a scalable and dynamic way.

NT methods can be classified into: (i) Classical approach (collection of traffic counters and logs through path probing and by using ping/traceroute mechanisms), (ii) Flow based telemetry (active or controlled detection of network flows and collection of flow performance parameters such as: drops, latency, routes, bandwidth) and (iii) Packet based telemetry (all or a sampled part of network data is analysed).

There are multiple methods to collect NT information, which are: (a) periodic downloading of the collected data to an analyser unit for subsequent analysis, (b) active streaming of collected data to the analyser unit and (c) the attachment of telemetry information to user data and the stripping of the data in the sink (data collection) points.

Traditional network architectures realize NT in a bottom-up fashion. The network operator independently configures and collects data from the network devices using tools such as sFlow[17] and NetFlow[18]. The huge amount of data is collected in a centralized entity where it is aggregated and analysed by the network operator to infer the appropriate conclusions. There are several problems with this approach in today's networks:

1) Individual network devices can capture monitoring data at a certain granularity due to inherent processing limitations. This granularity is inadequate in some cases since certain problems in the network such as intermittent errors and microbursts may be difficult or impossible to diagnose.

2) Paradoxically, even the inadequate monitoring frequency can generate a vast amount of data in case of large-scale networks. The huge amount of information that is aggregated makes the correlation of the data an intractable task. The complexity further increases due to the divergence of some data attributes that are collected from heterogeneous devices (e.g., different timestamps, sampling periods and APIs).

3) Network attacks are constantly increasing and evolving. In many cases they are automated and are programmed to be adaptable so that they are difficult to be identified. In these scenarios, a respective automated and adaptable framework is required to promptly identify and neutralize the threats.

4) With the advent of Software Defined Networking (SDN), network architectures are evolving towards an automated paradigm, where they can autonomously self-monitor and react to certain events. Under these scenarios a respective automated SDN-based NT system can accurately analyse the monitoring data without the need of a human network operator as it is usually the case in today's networks.

Given the importance of NT, and the disadvantages of the traditional approach for monitoring, a lot of effort has been directed towards developing novel NT frameworks. Recently, NT has attracted a lot of research attention and it is currently a very hot research topic due to the scientific complexity that it encompasses and its direct application to existing operating systems.

In order to overcome the inadequacy of traditional sampled flow measurements, data streaming or sketching (a sketch is a summary of a data set) frameworks have been studied [81] [82] [83]. In these cases, custom data structures and algorithms are developed to monitor specific metrics of interest. A major drawback of this approach is that each metric of interest

17 sFlow, website: https://sflow.org/
18 "Cisco Netflow," website: https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html

requires different algorithms for its monitoring. Thus, it is not a viable solution in the long term given the evolving nature of networks and the diverse requirements demanded from it. To address this, in OpenSketch [84] routers are equipped with predefined hardware functions and their controllers can be programmed to perform different tasks. While OpenSketch is a significant improvement over previous approaches, it still has some drawbacks. The switches are programmed to monitor specific metrics, and therefore cannot monitor other possibly useful ones. Furthermore, different sketches are required to monitor different tasks, which is very resource consuming. The work in [85] proposes UnivMon that employs a generic monitoring data plane with various estimation algorithms running in the control plane and uses statistics from the data plane to compute application-level metrics. When compared to OpenSketch, UnivMon achieves comparable accuracy with up to three orders of magnitude less memory consumption and communication overhead. Various other approaches for NT have also been examined: NetAssay [86] is a system that enables intentional network monitoring. It allows network operators to monitor a subset of traffic described by high level primitives (e.g., traffic based on domain names and autonomous systems). In [87], a programmable interface is described that enables end-hosts to embed probe programs into packets to query the network's internal state and perform tasks such as congestion control and load balancing. In [88], a joint framework (Sonata) is proposed for the collection and analysis of network data. Sonata provides the network operator with a high-level language that refines the operator's queries to capture only the relevant information. The results are easier query syntax for the network operator and reduced required data rates for the streaming analytics systems. Recently, [89] proposed a high-level framework for operators to specify: i) measurement queries based on their needs, ii) programmable measurement primitives at switches and iii) a runtime that translates the high-level queries into low-level API calls. In-band telemetry [90] is a relatively old (c. 2000 [91]), concept that is consistently attracting research attention until today. In-band telemetry inserts metadata into packets, and thereby collects hop-by-hop information about the state of the network. The information is then delivered to a telemetry server. It is a data-plane driven network telemetry method. Alternate marking-performance measurement (AM-PM) [92] is an in-band telemetry method to perform packet loss, delay, and jitter measurements on live traffic. This method is based on an Alternate-Marking (colouring) technique. It "marks" the packets so that the packets belonging to the same block will have the same colour, whilst consecutive blocks will have different colours. By using this technique, various metrics (such as packet loss) can be derived.

Regarding algorithms that leverage the data of a network telemetry framework, the research is significant and ongoing. In [93] and [94] streaming machine learning algorithms were applied for the real time detection of BGP anomalies. In [95] kNN and Random Forests machine learning algorithms were trained to detect and identify DDoS attacks using native cloud telemetry monitoring. In [96] the authors review different time series forecasting approaches for telemetry data collected at data centres.

Regarding the inference of certain network performance metrics, [97] focused on delay measurement and implemented the HTTP request one-way delay measurement based on INT

of the programmable data plane. EverFlow [98] and Pingmesh [99] create network probes to infer network end-to-end delay and jitter changes and detect network faults with drastic delay changes in data centres. Refs [100] and [101] use AM-PM to calculate the network hop-by-hop delay. Ref. [102] proposed CAPEST, an INT-based data plane offloading framework to estimate the capacity and available bandwidth of a network. SIMON [103] reconstructs performance metrics such as packet queuing times at switches, link utilizations, and queue and link compositions at a flow-level.

The purpose of network telemetry is to ultimately provide precious information that will be used to make efficient network control decisions. To this end, research on automatic network control architectures and mechanisms is vast and ongoing. Ref [104] reviews topics such as SDN, NFV and machine learning towards automatically adaptable networks. Ref. [105] proposes NetworkAI, an architecture for self-learning control strategies in SDN networks. NetworkAI leverages deep reinforcement learning and monitoring technologies, such as the in-band network telemetry to dynamically generate near optimal network close-loop control policies. NetworkAI applies DRL to solve real-time large-scale network control problems without relying on manual processes and assumptions of the underlying network by compressing the network state space through monitoring of certain metrics. Finally, EARS [106] is a network architecture based on SDN that leverages reinforcement learning to make automatic routing decisions based on the state of the network.

Another major topic that network telemetry plays a paramount role is network anomaly detection [107]. This topic encompasses several research areas such as intrusion detection, DDoS attack detection, BGP anomalies and failures detection. It is a relatively old research topic, and one that has consistently attracted research attention. With the advent of machine learning, new directions have recently arisen. In [108] the authors developed a framework to detect and localize gray failures (i.e., failures that are not too big to raise an alarm). The assumed in-band network telemetry and leveraged probe packets to detect the failures, and reoptimize the network. [109] focuses on the choice of the most appropriate telemetry data to detect BGP anomalies. It selects a small number of telemetry features from the vast amount of telemetry data for anomaly detection. The authors of [110] [111] leveraged streaming machine learning techniques for detecting BGP anomalies. More specifically they use DenStream, an unsupervised clustering technique aiming to detect in real-time the outliers (anomalies). In [112] the authors focus on the detection of Distributed Denial-of-Service (DDoS) attacks. More specifically they evaluate several machine learning algorithms in certain datasets to detect and identify SYN_Flood and GET_Flood DDoS attacks. [113] also used ML to detect anomalies. More specifically the authors propose a novel weighted class ML classification scheme. This scheme ensures accurate classification of anomalies under imbalanced data (i.e., when the training data for anomalies are much less than the normal data). Finally, [114] leverage machine learning to identify malicious behaviours of users in a cloud-based environment from the perspective of a cloud service provider.

The purpose of Table 2 is to compare three (3) different monitoring tools that can be used for cloud and network monitoring. Eight (8) characteristics of these tools are evaluated:

1. The OS (Operation System) the tool can be installed to.
2. Open-source tool or not.
3. The main programming languages that have been used for the creation of the tool.
4. Support of plugins.
5. Support of a built-in custom alert system.
6. Support of multiple connected machines where data can be exchanged between them.
7. Real-time monitoring without the use of a plugin.
8. Build-in dashboard that illustrate results and let the user monitor the data collected.

Table 2: Monitoring tools

Component	Netdata	Prometheus	Arkime (formerly Moloch)
OS	Linux, Debian, MacOS	Linux, Debian, Windows	Centos, Ubuntu
Open-Source	Yes	Yes	Yes
Programing Language	C, Python, JavaScript, Shell	GO, TypeScript, JavaScript	JavaScript, C, Vue, Perl
Plugins	Yes	Yes	Yes
Custom Alerts	Yes	Yes	No
Multiple Machines	Yes	Yes	No
Real Time Monitoring	Yes	No	No
Dashboard	Yes	No	Yes

Netdata: Netdata[19] is an open-source monitoring tool that collects metrics from systems and applications in real time. It can be installed on a variety of operation systems, such us Linux, Debian, MacOS and more, and it is also supported by Kubernetes and Docker. Netdata offers a plethora of configurations by letting the administrator create and install custom alerts[20] or plugins[21] to be triggered by various machines[22]. Netdata also offers the ability to stream and replicate[23] in multiple nodes. The nodes are being categorized in 3 groups: child, parent, proxies. The child node is responsible to send its metrics to the parent node via the proxy nodes. Each node can mirror its database to another Netdata and run health checks, trigger alarms or export metrics. This tool has a built-in dashboard[24] that is easy to navigate. Netdata is a combination of many programming languages, the four most used ones are C 74%, Python 8.3%, JavaScript 7.6%, Shell 3.9%.

Prometheus: Prometheus[25] is an open-source monitoring and alerting toolkit. It supports various operating systems and architectures, such as Linux, Debian, Windows and it can be

19 https://learn.netdata.cloud/docs/overview/what-is-netdata
20 https://learn.netdata.cloud/docs/monitor/configure-alarms
21 https://learn.netdata.cloud/docs/agent/collectors/plugins.d
22 https://learn.netdata.cloud/docs/monitor/enable-notifications
23 https://learn.netdata.cloud/docs/agent/streaming
24 https://learn.netdata.cloud/docs/agent/web/gui
25 https://prometheus.io/docs/introduction/overview/

installed via Docker. Prometheus supports the creation of custom alerts and plugins such as Grafana[26]. Grafana allows the user to navigate at the dashboard and create graphs or add external data sources. Prometheus is able to support multi target exporter in order to collect metrics from multiple machines, without the exporter being installed to all of them. Although Prometheus does not support real time monitoring by itself, with the use of Grafana plugin, the administrator can monitor and stream data in real time. Prometheus is a combination of many programming languages but mostly it based on Go 89.1%.

Arkime: Arkime[27] (formerly known as Moloch) is an open-source data capture software. Arkime can be installed in Ubuntu or Centos, and it has a built-in dashboard. Although Arkime does not support multiple nodes being connected, this can be achieved with Elasticsearch[28]. Elasticsearch also gives the ability to Arkime to become a near real time tool. Arkime also supports a plethora of different plugins that can be used such as Suricata[29]. Suricata plugin allows the user to create custom alerts. Arkime is a combination of many programming languages, the four most used ones are JavaScript 27.4%, C 26.1%, Vue 24.9%, Perl 10.7%.

Beyond state of the art

SERRANO aims to evolve the current and ongoing research and integrate it to its advanced architecture. SERRANO will develop a holistic network and cloud telemetry framework for heterogeneous resources (cloud, edge and HPC), coupling a hierarchical telemetry infrastructure with novel AI/ML algorithms. These algorithms will continuously monitor the various resources (network, storage, servers, and containers) and the running applications, dynamically configuring the resources according to the applications' workload. The algorithms will correlate the monitoring information to accurately identify and address evolving events of interest, in a scalable and dynamic manner.

The SERRANO telemetry mechanism will consist of local Enhanced Telemetry Agents and a Central Telemetry Handler placed at the root of the hierarchical telemetry tree. The Enhanced Telemetry Agents will collect, pre-process and correlate telemetry information at a local level. This information will then be used to feed with appropriate data the resource-hosted local service assurance mechanisms. This will enable traffic monitoring, dynamic resource re-optimization (updating the allocation of workload and the resources' configuration) and effective anomaly management (i.e., congestion discovery, BGP anomalies detection, DDoS attack detection and generalised anomaly detection) by the respective Local Orchestrator. The telemetry agents will also aggregate and forward to the Central Telemetry Handler the collected data, triggering global management actions from the Resource Orchestrator. With such a hierarchical approach, the management complexity and the intervention to the infrastructure will be kept as low as possible, avoiding to overload the central orchestration mechanisms.

26 https://grafana.com/docs/
27 https://arkime.com/arkimeetus
28 https://www.elastic.co/
29 https://suricata-ids.org/

SERRANO will research various ML algorithms for the analysis of the telemetry data and quantify their performance in terms of detection accuracy. These algorithms have not been previously applied to unified heterogeneous network and computing infrastructures. SERRANO telemetry mechanisms will feed the developed AI/ML methods with key performance metrics and post-execution validation information from the individual edge, cloud and HPC platforms. The AI and ML algorithms continuously monitor and learn the resources' behaviour, conditions, usage and workload. The algorithms: (i) correlate the monitored data, (ii) identify or predict and handle failures, degradations and shortcomings, (iii) take dynamic reconfiguration actions to optimize the infrastructure's efficiency. Also, SERRANO envisages event driven algorithms that dynamically deploy additional necessary telemetry functions whenever appropriate, e.g., in cases that the network and computing infrastructure is or will be soon unable to meet the required Quality of Service (QoS) metrics.

5.2 Automated optimized resource orchestration in disaggregated cloud and edge infrastructures

Research directions and related work

The emergence of real time applications has created the need for edge resources. Edge computing is an alternative approach to the cloud environment. It enables the service of applications as physically close as possible to the site where the data is generated in contrast to a remote centralized cloud or data centre or data storage location. With the advent of new applications with strict requirements edge resources are expected to serve 75% of the data created [115], which underlines the crucial role of edge computing over the coming years.

Although edge and cloud resources share many similarities, there are some striking differences between them. In the cloud computing model, the computational capacity is abundant, while the bandwidth and the latency between the end-users/devices and the resources are performance limiting factors. These limitations can be addressed by edge computing, which enables flexible bandwidth requirements and low latency due to the proximity of the edge resources to the end-users/devices. By applying edge computing, a valuable continuum from the device to the cloud is created, which can handle the massive amounts of data generated. Costly bandwidth updates are no longer required as there is no need to transfer gigabytes of data to the cloud. Edge resources can also analyse sensitive IoT data within a private network, thereby protecting sensitive data. Also, an infrastructure-based difference is that cloud is based on relatively few core cloud data centres versus the numerous small edge clouds/resources, which in turn are characterized by dynamicity in availability and limitations in the storage space.

Heterogeneous computing has also attracted interest in support of the diverse application requirements. This is not a new concept and different types of computing resources are already working together in a variety of applications. There are powerful nodes which are fully

virtualized [116] (i.e., Containers and Virtual Machines), less powerful edge nodes that have limited virtualization capabilities (i.e., only Containers) or even none (i.e., Bare Metal devices) [117]. Using different types of resources together provides systems with capabilities that single type systems cannot match. However, the wide variety and heterogeneity of resources operating in an edge-to-cloud infrastructure, creates service deployment and resource management challenges.

Multi-Objective Optimization

The intrinsic characteristics of a cloud-edge infrastructure introduces from an algorithmic perspective a high number of conflicting objectives that need to be addressed simultaneously. Hence, the development of **multi-objective optimization mechanisms** becomes a necessity. However, in contrast to trivial problems in which each of these target functions can be optimized independently and then the global can be calculated, in the non-trivial problem of resource orchestration in disaggregated cloud and edge infrastructures, no single solution exists that optimizes all the objective simultaneously.

To efficiently solve such problems several different approaches have been proposed in the literature. The first approaches for solving multi-objective optimization problems were based on exact methods, which search the complete space of feasible solutions to guarantee optimality [118]. They include multi-objective variations of numerical methods such as (Mixed) Integer Programming [119] [120] and Dynamic Programming [121]. However, in such case the search space is often very large which makes such approaches almost impractical for real size problem instances [122].

To decrease the search space, heuristic and meta-heuristic algorithms with polynomial complexity are used. They can produce approximate or near-optimal solutions at the cost of acceptable optimality loss [123] trading off execution time with optimality. Some notable examples of such algorithms include multi-objective particle swarm optimization (MOPSO) [124] and non-dominated sorting genetic algorithm-II (NSGA-II) [125]. Based on the concept of nondominated optimality, Clerc [126] used an external archive to store and determine which particles would be the nondominated members, and these members were used to guide other particles' flight. Kennedy [127] adopted the main mechanism of the NSGA-II algorithm to determine local optimal particle among the local optimal particles and their offspring particles and proposed a method which used the max-min strategy in the fitness function to determine the Pareto dominance. Ghodratnama et al. [128] applied the comprehensive learning particle swarm optimization (PSO) algorithm combining with Pareto dominance to solve the multi-objective optimization problems. Ozcan and Mohan [129] developed an elitist multi-objective PSO that combined the elitist mutation coefficient to improve the particles' exploitation and exploration capacity. Wang et al. [130] proposed an iterative multi-objective particle swarm optimization-based control vector parameterization to cope with the dynamic optimization of the state constrained chemical and biochemical engineering problems.

Reinforcement Learning (RL) is often considered to be a general formalization of decision-making tasks and a subfield of machine learning. In RL, agents learn not from sample data but from experiences that interact with the environment. RL algorithms combined with deep neural networks (DNN), form deep reinforcement learning (DRL) methods to solve complex problems in the real world. Deep Reinforcement Learning (DRL) models and methodologies have been proposed for dealing with multi-constraint and multi-objective optimization problems using multi-agent training methods (MADRL) [131] as a powerful data-driven model for environments with huge system states that need to be re-optimized in real time [132] [133] [134] [135].

The pioneering model is Deep Q-Network (DQN) [136], which does not require adjustment of network architecture or hyperparameters. Based on DQN, a variety of DRL algorithms was proposed mainly to accelerate the speed of learning. Double Deep Q-network (DDQN) was proposed by Hasselt et al [137]. By applying Double Q-learning to DQN, it can separate action selection and policy evaluation, reducing the risk of overestimating Q value. Duelling Deep Q-network was put forward by Wang et al. [138] [139] the model divides the abstract features extracted by convolutional neural networks (CNN) into two branches, one of which represents the state value function and the other represents the advantage function. Through this duelling network structure, agents can identify the correct behaviours faster in the process of policy evaluation and network architecture can be better integrated. Schaul et al., [140] proposed a double deep Q network with proportional prioritization based on DDQN. This method replaces uniform sampling with priority-based sampling, improves the sampling probability of some valuable samples, and thus speeds up the learning of optimal policy. Lakshminarayanan et al. [141] proposed Dynamic Frame Skip Deep Q-Network (DFDQN), which uses dynamic frameskip to replace action repeated k times per moment in DQN. Experiments show that DFDQN achieves better performance in some Atari 2600 games [141]. Vincent et al. [142] use the adaptive discount factor and learning rate in DQN, the convergence speed of the deep network is accelerated. Schaul et al. [140] developed a framework for prioritizing experience and used it in DQN to replay important transitions more frequently and learn more effectively. The success of DQN encouraged full-scale research of value-based methods by studying various demerits of DQN and developing auxiliary extensions. Hessel et al. [143] proposed Rainbow DQN, which uniting seven Q-learning-based ideas in one procedure to verify whether these merged extensions are essentially necessary for the RL algorithms.

Multi-agent Deep Reinforcement Learning (MADRL) methods are widely applied in various fields of complex real-world tasks in different domains. Such domains include autonomous driving [144], Internet advertising [145], and resource utilization ([146] [147] [148] [149]), traffic control ([150] [151] [152]) and workflow scheduling ([139]).

In the field of autonomous driving, Shalev-Shwartz et al. [144] demonstrated policy gradient iterations can be used without Markovian assumptions. In addition, they decomposed the problem into components that enabled the comfort of driving and the safety of driving respectively. Jin et al. [145] proposed an algorithm which combined clustering with MADRL to optimize the performance of real-time online bidding with a large number of advertisers. To

balance the competition and cooperation between advertisers, a distributed coordinated multi-agent algorithm was proposed. In the field of resource management, Xi et al. [146] proposed a fast and robust MADRL algorithm to solve the stochastic disturbance problem of the power grid system caused by the integration of the distributed energy and new energy with stochastic games in non-Markovian environments. Perolat et al. [147] studied the emergent behaviour of groups of independent agents in partially observable Markov games. It modelled the occupant of common-pool resources, revealed the relationship between exclusivity, sustainability and inequality, and proposed solutions to improve resource management ability. Kofinas et al. [148] proposed the fuzzy Q learning method to effectively improve the energy management ability of decentralized microgrid. Noureddine et al. [149] proposed a DRL cooperative task allocation method, which enables multiple agents to interact and effectively allocate resources and tasks. In a loosely coupled distributed multi-agent setting, agents can benefit from collaborative neighbours. In the traffic control domains, Chen et al. [150] proposed a cooperative MADRL framework to alleviate bus congestion on bus lanes in real time. They adopted coordination graphs to automatically select the coordinated holding actions when multiple buses are stationed at the stops. In addition, for sparse graphs they developed sparse collaborative Q learning algorithms for coordinated holding actions. Vidhate et al. [151] proposed a traffic control model based on cooperative MADRL for the control and optimization of the traffic systems, which can deal with unknown complex states and can realize real-time dynamic traffic control. Finally, Wang et al. [139] proposed a stochastic Markov game and a decentralized DQN-based MADRL framework, which can obtain correlated equilibrium solutions, for the problem of multi-objective work-flow scheduling in the heterogeneous IaaS cloud environment. Their case studies show that the proposed method outperforms the baseline meta-heuristic algorithms such as NSGA-II and MOPSO. In recent works, Transfer Learning [153] and Active Learning [154] are also considered for re-utilizing the gained knowledge and minimizing the amount of training data. Finally, Federated Learning (FL) [155] is indicated as an appropriate approach when data privacy and security concerns are critical.

Resource allocation in multi-clouds

Also, users tend to use multi-cloud resources to overcome the risks from relying on a single cloud and to obtain higher availability [156], [157], lower latency [157] and lower monetary cost [157]. Hosting data to multiple cloud providers with appropriate erasure coding may also increase the privacy for sensitive data, since different providers host parts of the original data that must be combined in order to be readable [158]. This enhances the level of security against either external attacks or exposure to an untrusted single provider hosting all the data. The existing studies can be divided into four categories, (i) monetary cost optimization, (ii) data availability optimization, (iii) data access latency optimization, and (iv) cost-availability trade-off.

Minimizing the monetary cost based on some QoS metrics is considered in many of the related studies. Abu-Libdeh et al. [159] proposes Redundant Array of Cloud Storage (RACS), a proxy striping user data across multiple providers to reduce the cost of switching providers. However, the authors do not propose a method to solve the data placement problem to meet

any optimization goal. Papaioannou et al. [160] propose Scalia as inspired by RACS. It is a cloud storage brokerage solution for adaptive data placement, which minimizes the storage cost. Furthermore, Mansouri et al. [161] present an algorithm to find subsets of data centres to store original data and their replicas such that the storage cost is minimized while the expected availability is guaranteed.

But the above only consider a part of the monetary cost optimization: the cost of switching providers and storage cost. The cost of data storage management in the cloud should consist of residential cost (i.e., storage and data access operations), and network cost resulting from data transfer from cloud providers [162].

In [163], Hadji proposes a commodity flow solution to minimize the cost of storing data and latency to access data centres. However, the network and operation costs are ignored when users access their data. Ma et al. [164] adopt the ensemble of replication and erasure coding leading to low bandwidth cost, low storage cost, and low latency, but ignore the proper selection of cloud providers. One function in CHARM, proposed by Zhang et al. [165], is to select the data placement configuration that contains several suitable clouds and an appropriate redundancy strategy to store the data with minimized data storage management cost and guaranteed availability. But the proposed algorithm is a simple heuristic solution and cannot obtain an optimal solution. The studies in [163] and [165] minimize only the monetary cost at a certain point in the time slot. Mansouri et al. [166] propose an optimal offline algorithm to minimize the residential and migration costs in a time slot.

There are also several studies to minimize the monetary cost based on QoS metrics for a geographical distributed cloud storage. Wu et al. [167] present a unified view of storage services in geographically distributed data centres called SPANStore. It aims to minimize the monetary cost and compute resources with the constraints of GET/PUT latencies, flexible consistency, and tolerate failures. Liu et al. [168] propose a multi-cloud service to minimize the payment cost while providing Service Level Objective (SLO) guarantee to customers. To minimize the payment cost, the authors propose a heuristic solution based on genetic algorithm to maximize the reservation benefit.

There are several recent works that study strategies for the replication of data to multiple cloud providers, in order to attain higher availability and avoid vendor lock-in, while keeping the cost low ([169] [170] [161] [163]). Mu et al. [171] introduce µLibCloud, a client-side library based on Apache libCloud to improve the data availability through erasure coding. But it cannot give any optimization model to provide an optimal data placement. In [159] and [160], the authors also use erasure coding to enhance data availability and avoid vendor lock-in but fail to provide the specific mathematical expressions. Zhang et al. [165] use erasure coding to store a data object and calculate data availability. Similarly, Wang et al. [172] optimize data availability through erasure coding.

Singh et al. [173] propose a secured cost-effective multi-cloud storage model to minimize the total cost of storing data, while maximizing QoS, but give many unreasonable assumptions and use cost minus QoS as the final optimization goal. Wang et al. [172] propose an ant colony algorithm-based approach to minimize the monetary costs and maximize data availability. Su

et al. [174] propose a systematic model to formulate data placement in multi-cloud storage scenarios by using erasure coding that solves the data placement problem under complex requirements. Wang et al. [175] proposed a scheme for minimizing total cost and maximizing data availability. Their approach is based on NSGA-II with the goal to obtain a set of non-dominated solutions (i.e., a list of cloud storage providers) and erasure coding parameters. They also use an entropy-based method to recommend the most suitable solution for users who cannot choose one from the resulted Pareto-optimal set. However, in the proposed scheme the authors do not consider latency as part of the optimization criteria.

User-perceived latency can be lowered by deploying applications across multiple cloud services rather than within a single cloud service. Furthermore, it can also be minimized by placing replicas close to users, by co-locating the data accessed by the same transactions and by determining the location and roles of replicas (master and slave) in a quorum-based configuration (Sharov et al. [176]). However, users may still experience latency variations. Nishtala et al. [177] propose a scheme for caching data in memory, in order to further reduce these latency variations. Wu et al. [178] propose a scheme where redundant reads/writes are issued to replicas for the same goal. Finally, Suresh et al. [179] propose using feedback from servers and users to prevent requests redirection to saturated servers.

In what follows, we present tools for the orchestration of containers. There are of course various tools for the orchestration of computing, storage and networking resources in general, however the following tools are more related to the scope of SERRANO.

Feature	**Docker Swarm stand-alone**	**Docker Swarm integrated**	**Kubernetes**	**Mesos**	**Mesos + Aurora**	**Mesos + Marathon**	**DC / OS**	**Nomad**
Cluster Architecture & Setup (features to setup a running container cluster on top of an OS and/or a cloud provider infrastructure)								
Configuration Management Type	declarative	declarative	declarative	n/a	declarative	declarative	delegate	
Config Language	YAML mark-up	YAML mark-up	YAML, JSON mark-up		Python	JSON mark-up	JSON mark-up	
Architectural Patterns								
Highly Available Design	yes		yes	yes	yes	yes	delegate	

HA Master Setup	fully automated & portable	fully automated & portable	not in open-source distribution	no	fully automated & portable	fully automated & portable	delegate	
Versioned HTTP API & Client API libraries	yes	yes	yes	yes	no	yes	extended	
Simple, Policy-rich, Highly Customizable Scheduling Algorithms	yes	yes	yes	n/a	yes	yes	delegate	
Container runtime architecture								
Unified Container Runtime Architecture	yes	yes	yes	yes	yes	yes	delegate	
OCI Support	yes	yes	yes	future	future	future	future	
Supported Container Runtimes	Docker Engine, runC (OCI)	Docker Engine, runC (OCI)	Docker Engine, rkt, runC & any other OCI-based container runtime	Docker & Mesos				Docker
Framework design of core orchestration engine								
External Plugin Architectures	yes	yes	yes	yes	no	yes	delegate	

Plugin Architectures for Schedulers	no	no	yes & multiple scheduler plug-in in parallel	yes	yes	yes	delegate	
Modular Interceptors	no	no	yes	yes	yes	no	no	delegate
Container Networking (features to customize how containers are interconnected, load balanced & discovered)								
Support for Docker' s libnetwork	yes	yes	no	yes	no	yes	delegate	
Separation of Data & Control Traffic	yes	yes	Multus plugin	no	no	no	no	
Application Configuration & Deployment (features to configure, compose, deploy, scale & upgrade containerized software services & apps)								
Smallest Unit of Deployment	task (always encapsulates a single container)		Pod	Task (encapsulates at most one container, but can also run non-containerized processes)				
Pods	no	no	yes	yes	no	yes	delegate	
Container-based Jobs	no	no	yes	no	yes	no	add	
Container-based Services	no	yes	yes	no	yes	yes	delegate	

Elastic Scaling of Services	yes	yes	yes	no	yes	yes	delegate	
Auto-scaling of Services	no	no	yes	no	no			
Global Containers:	no	yes (global services)	yes (daemon sets)	no	no	no	no	
Reusable Container Configuration (features to support generic but configurable container images)								
Pass Env Var to Container	yes	yes	yes	yes	yes	yes	delegate	
Self-inspection API	no	no	yes	no	no	yes	delegate	
Configuration Data & Image Separation	no	yes	yes	no	no	no	no	
Custom Entry point	yes	yes	yes	yes	yes	yes	delegate	
Custom CMD	yes	yes	yes	yes	yes	yes	delegate	
Container QoS Management (features to efficiently use the user group's resources and achieve app QoS)								

Min CPU Guarantees	yes	yes	yes	yes	yes	yes	delegate	
Min Memory Guarantees	yes	yes	yes	no	no	no	no	
Max CPU Limits	yes	yes	yes	yes	no	no	no	
Max Memory Limits	yes	yes	yes	yes	yes	yes	delegate	
Abstraction of CPU-Shares	no: rely on CPU-shares of the CFS Scheduler for minimal guarantees	yes: provide higher-level abstractions to express min guarantees hiding CPU-shares' complexity	yes: provide higher- level abstractions to express min guarantees hiding CPU-shares' complexity	no: rely on CPU-shares of the CFS Scheduler for minimal guarantees	no: rely on CPU-shares of the CFS Scheduler for minimal guarantees	no: rely on CPU-shares of the CFS Scheduler for minimal guarantees	no	
Allocation of Other Resources								
Limits for NVIDIA GPU	no	no	no GPU sharing	yes	yes	yes	delegate	
Limits for Disk Resources	no	no	local storage	yes	yes	yes	delegate	
Controlling Scheduling Behaviour by Means of Placement Constraints								
Evaluate over Node / Labels / Attributes	yes	yes	yes	n/a	yes	yes	delegate	

Define Custom Node Labels / Attributes	yes	yes	yes	yes	yes	yes:	delegate	
More Expressive Constraints	yes	yes	yes	n/a	yes	yes	delegate	
Controlling Pre-emptive Scheduling & (Re) Scheduling Behaviour (features to customize pre-emptive scheduling and rescheduling to achieve intended app QoS in exceptional conditions: resources contention at scheduler level, out-of-resource node conditions, node failures, container start failures, unbalanced services whose containers are not spread across diff nodes)								
Pre-emptive Scheduling	no	no	yes	no	yes	no	no	
Container Eviction when Node Runs Out of Resources	no	no	yes	no	yes	no	no	
Container Eviction on Node Failure	yes	yes	yes	yes	yes	yes	delegate	
Container Lifecycle Handling	no	yes	yes	yes	yes	yes	delegate	
Re-distributing Unbalanced Services	no	yes	future	no	no	no	no	
Monitoring Resource Usage & Health								
Central Monitoring of & Container Services	no	no	yes	yes .	no	no	extend	

Central Monitoring of CO Resource	Prometheus	Prometheus	yes: also monitoring GPU usage	yes: also monitoring GPU usage	yes	yes	delegate	
Framework for Container Health	yes	yes	yes	yes	yes	yes	extend	
Distributed Events Central Monitoring	no	yes	yes	yes	yes	yes	delegate	
Logging & Debugging of CO & Containers								
Logging of Containers	yes	yes	yes	yes	no	no	extend	
Internal Logging of CO Component	yes	yes	yes	yes	yes	yes	extend	
Integration with Log Aggregator Systems	not in open-source distribution		yes	no	no	no	add	
Cluster Maintenance								
Cluster State Backup & Recovery	no	yes: built-in	future	yes	yes: built-in	yes: built-in	delegate	
Official Cluster Upgrade Documentation	no	no	yes	yes	yes	yes	extend	

Upgrade Does Affect Not Active Containers	yes	yes	kubeadm (deployment tool)	yes	yes	yes	delegate	
Draining Containers from Node for Maintenance	no	yes	yes	yes	no	yes	delegate	
Garbage Collection of / Containers / Images	yes	yes	yes	yes	no	no		
Multi-cloud Support								
Cluster Across Availability Zones / Regions	yes	yes	Kubernetes limited support for multi-zone deployments	yes	yes	yes	extend	
Recovering From Network Partitions	no	no	no	yes	yes	no	no	
Multiple Clusters' Management Across	no	yes: Docker's Docker Cloud	yes: kubefed	no	no	no	multi-cluster CLI	
Federated Users' Authentication Across Clusters	no	no	yes:	no	no	no	single sign-on across clusters	
Multi-zone / Multi-region Workloads	no	yes	yes	yes	yes	yes	extend	

Another, tool for the handling of the generated data is INTRASOFT's Big data StreamHandler platform. This aims at: (i) scaling out and accommodating various data, from different domains, interoperating with all modern data storage technologies as well as other persistence approaches and (ii) supporting all important Big Data aware languages including Python, Java, R and Scala as well as other traditional programming approaches. The StreamHandler Platform consists of: (i) Data sources and Data store, (ii) Connectors and Streaming Core component, (iii) Schema Registry, (iv) Processing Infrastructure and (v)

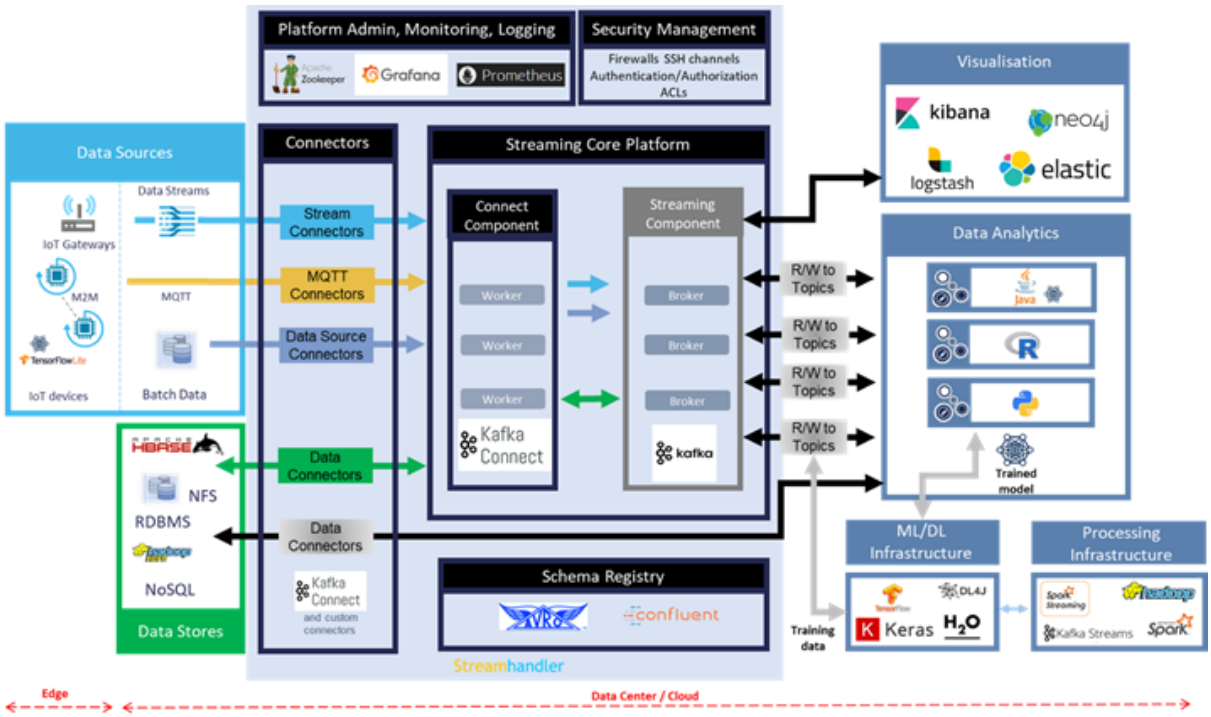


Figure 17: StreamHandler - High level Architecture

Platform Config/Admin Dashboard.

a. **Data sources and Data store**, which represent data streams and data sources, both in a structured or unstructured format that can be made available and potentially be connected to the StreamHandler platform, generated by any IoT device and/or gateway on the edge. Similarly, and according to the requirements, appropriate persistent storage can be used, as depicted in the input/output data components (Figure 17). The platform can interoperate with IoT architectures building an end-to-end IoT integration with the platform with the use of MQTT protocol. MQTT is a widely used ISO standard (ISO/IEC PRF 20922) publish-subscribe-based messaging protocol.
b. **Connectors and Streaming Core** that connect external data sources and make them available to the AI platform. External data sources are connected and made available by employing the "Stream Connectors" and "Data Source Connectors". The underlying technologies and capabilities of the Streaming Component and the multiple workers of the Connect Component allow the realisation of scalable and secure stream data pipelines. The Communication Platform represents a distributed streaming platform based on Apache Kafka. Apache Kafka allows to publish and subscribe to streams of records (Topics)

similar to the functionality provided by a message queue. The streams of records are stored in a fault-tolerant durable way and consumers (Big Data Apps or AI Apps) can process them as they occur. In general, Kafka is suitable for building real-time streaming data pipelines to reliably get data between systems or applications and for building real-time streaming applications that transform or react to the streams of data.

c. **INTRA's StreamHandler Platform** supports data models and their metadata by using the **Schema Registry** service (Kafka addon). **Schema Registry** provides a serving layer for your metadata. It provides a RESTful interface for storing and retrieving Avro schemas. It stores a versioned history of all schemas, provides multiple compatibility settings and allows evolution of schemas according to the configured compatibility settings and expanded Avro support. It also provides serializers that plug into Kafka clients to handle schema storage and retrieval for Kafka messages that are sent in the Avro format.
d. **Processing Infrastructure**. The underlying infrastructure spans multiple VMs and provides all the necessary technologies and components that enable the storage and analysis of the data involved and further allowing the usage of any technology agnostic algorithms, by providing a distributed computing environment that enables the above. Some of them include among other Apache Spark, Hadoop, Kafka Streams, Spark Streaming, H2O, etc.
e. **Platform Config/Monitoring**. Platform configuration and monitoring can be assisted by dashboards that visualise selected metrics of interest. The platform components are capable of exporting JMX metrics which in turn are made available in customised dashboards.

In **SERRANO**, components of the Big Data StreamHandler will be used by SERRANO partners, supported by INTRASOFT, to connect external data sources and make them available to the SERRANO platform, supporting the functionalities of the use cases. The StreamHandler components will provide a distributed streaming platform, based on Apache Kafka cluster that will allow to publish and subscribe to streams of records. Running Apache Kafka in a clustered manner allows for the horizontal scalability of the system, achieving thus a scalable, fault-tolerant communication-efficient framework, as well as increased throughput to support high throughput and high velocity data streams.

Beyond state of the art

SERRANO platform will be based on intent-driven approaches for the resources' orchestration. The submitted workloads will include high-level requirements that are agnostic to the characteristics of the infrastructure. The platform will then apply AI/ML methods to translate these high-level requirements to particular optimization objectives. Following that, Multi-Agent Deep Reinforcement learning techniques will be applied to optimize the multiple infrastructure-specific objectives identified, by assigning each optimization objective to a particular agent.

Regarding the distributed storage service, our proposed architecture will use on-premises edge devices along with cloud sites. The approaches presented in the literature focus on environments composed solely from cloud resources, which may lead to increased service latency and high bandwidth requirements. The on-premises edge devices, due to the

proximity to the end user can provide small latency for data transfers and minimal costs for data retrieval. The developed storage policies will be optimized to use efficiently both cloud and edge sites based on the requirements for performance, availability and security, or the dynamic characteristics of the storage resources.

Also, most related works in distributed storage, use various single or multi-objective optimization approaches to only minimize the monetary cost, maximize data availability or minimize latency or optimize the cost-availability trade off. In SERRANO, we will use multi-objective optimization with Multi-Agent Deep Reinforcement Learning to optimize all those criteria, at the same time.

Finally, we aim to exploit novel approaches for the formulation of the problems in our studies. Knowledge graphs are gaining more and more traction with machine learning so that AI processes can use the same information for multiple different scenarios. They simplify complex concepts and provide good training data for AI mechanisms.

5.3 Energy and resource aware flow mapping

Research directions and related work

High-Performance Computing (HPC) provides resources that allow scientists and engineers to solve complex real-world problems. Applications include molecular dynamics, aerodynamics, biomechanics, weather prediction, process signal analysis etc. To expand the prediction power of these computer-based simulations even larger supercomputers are being built. However, there are two major limiting factors for further rapid growth in HPC: efficiency of compute units and interconnect scalability. To meet these challenges, sophisticated resource allocation strategies must be applied depending on the requirements of the computational tasks and the system architecture.

One of SERRANO's tasks is to prepare the considered uses cases for execution on emerging hardware architectures that are not yet even fully accessible to the general or the scientific public but are expected to be available in future Exascale computers.

This means that both the HPC environment and the cloud environment need to be enhanced to take advantage of the HPC platform on the one hand, and to ensure that the efficiency of supercomputing use is not affected on the other.

There are many factors that must be considered to port an application to HPC. From a SERRANO point of view the most important of these are:

a) Vectorization. The modern processors hold the floating-point numbers in special registers, namely vector registers. In a vector register multiple floating-point numbers can be held and processed in parallel. Vectorization significantly increases the performance of the computation: Each core of a modern General-purposed server processors is equipped with more than one hundred of the vector registers. However,

not all algorithms can be vectorized. The considered use cases will be examined for vectorization and optimized for it if applicable.

b) Scalability and Work Distribution. The work must be distributed evenly between the compute units. The non-parallelized phases of the application and a poor load balance leads to poor scaling that makes the use of supercomputers disproportionately expensive.
c) Input/Output (IO). Because the compute nodes do not have any local hard discs, the shared file system must be used to perform IO operations. However, this can fast become a bottleneck for the whole supercomputer.
d) Energy efficiency. The SERRANO project will provide certain HPC services that can be used in the considered use cases. The decision whether to use these services for a given input will be made by SERRANO Orchestrator depending on the expected time to solution and energy consumption.

The section “5.2.1 HPC Resources” in [180] provides more details about the requirements on the HPC applications. Several previous activities in the field of energy aware in HPC and Cloud are used and further improved in the SERRANO project.

Project EXCESS [181] addresses the important challenge of energy efficiency on the HPC software. EXCESS provides the methods and tools to optimize important properties, such as performance, execution time and energy efficiency for a wide range of hardware and software systems.

One of the major barriers to achieve better performance and thus energy efficiency is the development progress of memory bandwidth. While semiconductor technology and processor architecture have advanced very rapidly, main memory performance has not. For this reason, processors consume a significant amount of energy while waiting for data. It becomes particularly inefficient when the memory bound algorithms are executed. The IO and communication phases of the parallel applications additionally reduce the energy efficiency.

Also, modern processors are able to adjust their operating frequency [182], however optimal frequency cannot be automatically set, while additional requirements include the synchronization between all involved processors. [183] shows that by managing the processor frequency based on the execution context, up to 20 percent of the energy can be saved without affecting the performance. For the selection of the optimal execution configuration (e.g., CPU frequency, number of cores) many factors must be considered. Therefore, simple but solid models for performance and energy efficiency are necessary.

For example, Execution-Cache-Memory performance model (ECM) [184] explains, how the data flows between various components of the processor. It shows for example very clearly, why the theoretical memory bandwidth cannot be achieved in most cases and can be used to predict the optimal frequency and expected power for many algorithms. Another approach was investigated in the Dreamcloud project. Advanced monitoring systems and sophisticated management software were used to ensure energy-aware execution of the application in the

cloud system [185]. Unfortunately, this framework cannot be used in a large HPC system due to the requirements of the monitoring systems and the limited scaling.

An important element of SERRANO workflows is task farming. In this case, a series of sequential or poorly parallelizable computational tasks must be executed. This does not match the concept of the HPC (see section 5.2.3 Task Farming on HPC Platforms in [180]). In the SERRANO project, we will define an addition to the supercomputing framework schedule for these types of applications that will perform the execution of multiple smaller tasks much more efficiently. The framework implemented by Ralf Schneider will be used as a basis, which is applied for the analysis of continuum mechanical material parameters using many direct mechanical simulations [186]. In addition, it allows the use of the established HPC optimization tools and methodology, such as those described in the Performance Optimization and Productivity Project [187].

Beyond state of the art

In SERRANO, we will examine the kernels of the considered use cases, identifying basic features, such as number of bytes per flop, dependencies of cache reusing on the size of the problem and optimize them if necessary. Using the EXCESS tools and models, such as ECM, we will determine the optimal execution configuration for the different hardware. Considering the latest processors' capabilities will bring additional details for HPC community on how the various power manager features of the processors affect performance and power consumption. The latter can be used to optimize the operation of supercomputers and the servers in the cloud and edge.

We will also aggregate the HPC services that are optimized for use on HPC platforms, so that the computationally intensive parts of the use cases can be computed on them more efficiently and faster than in the Cloud. Therefore, we will develop and combine the above referenced models and frameworks.

5.4 Service Orchestrator and Abstraction Models

5.4.1 Abstraction Models

Research directions and related work

Semantic Web technologies such as RDF, RDFS and OWL enable users to describe their knowledge in a machine processable way (aka ontologies) that can be uniquely interpreted by software agents and hence enable them to answer more complicated questions. This is done by taking into account the meaning of data rather than the sequence of characters being used (knowledge inference). Subsequently various software tools were developed that enable users or systems to further process the knowledge expressed using the technologies and support relevant tasks, such as the visual query formulation and evaluation, the correspondence specification and data transformation and the whole decision support process as well. These technologies can be useful for the SERRANO platform and especially for

service orchestration. They can be potentially used for the formal expression of Use Case (UC) application requirements and/or mapping of high-level requirements. Even to the appropriate low-level terms that facilitate the service execution of UC applications from the resource orchestrator point of view.

Visual Query Systems (VQS) enable users to graphically form the appropriate query based on the ontological representation of a particular domain of interest. Optique [188] provides a user-friendly environment consisting of three distinct areas where the users can specify the concept of particular interest, the relations among them (in the form of a diagram) as well as the condition that the attributes of a concept should satisfy. The system accordingly generates the corresponding SPARQL[30] query. Many other VQS have been published so far, especially after the 2008 when the SPARQL become a W3C standard. Regarding the correspondence specification among the terms of partially overlapping ontologies there is a plethora of algorithms and tools [189]. However, they primarily focus on 1-to-1 correspondences while the functionality provided by the GUI (if any) is limited. For this purpose, we have also developed an ontology mapping tool [190] that enable user to efficiently deal with quite complicated mismatches. Mapping rules specified can be exported to the appropriate format including EDOAL[31] XML, JSON and HTML (for presentation purposes).

Regarding the data storage, representation and exchange the use of Extensible Markup Language (XML)[32] and JavaScript Object Notation (JSON)[33] technologies is ubiquitous since they enable the storage of data in a neutral way and the information residing in the respective documents is readable from both software systems and humans. The structure of these documents can be specified in advance using XML or JSON schemas that facilitate data processing by the respective systems, since they already know the elements, they are going to meet along with their attributes and the relations among them. It should be further noted that the OWL/RDFS ontologies developed, and mapping rules specified among different ontologies can be also represented using these technologies (e.g., RDF/XML) and hence used for their transfer, storage and integration with existing systems or new ones.

The in advance specification of the format, structure and meaning of elements facilitate the development of relevant systems (e.g., ensures that all important parameters were included) while it also protects them from potentials errors (e.g., misunderstanding of data) and prompted international non-profit organizations to develop standards regarding interoperable representation and exchange of data. Especially about cloud applications the OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) [191] helps system administrators or software agents such as resource orchestrator to configure applications and their underlying infrastructure and manage them as well as. TOSCA-based orchestration is based on the YAML[34]. The latter is a human-readable data-serialization

[30] SPARQL Query Language for RDF, https://www.w3.org/TR/rdf-sparql-query/
[31] EDOAL: Expressive and Declarative Ontology Alignment Language, https://moex.gitlabpages.inria.fr/alignapi/edoal.html
[32] Extensible Markup Language (XML), https://www.w3.org/XML/
[33] JSON (JavaScript Object Notation), https://www.json.org/json-en.html
[34] YAML, https://yaml.org/

language that it is commonly used for configuration files. It uses line and whitespace delimiters instead of explicitly marked blocks (like Python) and hence it enables the representation of storage of data in a concise form that is still readable by both machines and humans.

Beyond state of the art

The design of Abstraction Models is often driven by the purpose they serve. Consequently, the parameters specified about a particular domain of interest depend to a great extent on decisions made during the design (e.g., design paradigm, axes of classification). The application of the models developed so far for supporting a different scenario is a quite complicated process, since it presumes the precise and accurate mapping of elements specified with the ones explicitly or implicitly used by the other applications. Independent design of the applications and the respective models make this mapping process much more difficult if not impossible since the system administrators should often deal with partially overlapping domains of knowledge and hence a significant amount of the elements may be practically ignored or mapped with elements with not the same meaning when moving from one world of knowledge to the other one. Knowing in advance the ultimate goal of the user and hence the parameters being necessary for the proper function of the respective systems would be important during the models design and the definition of the respective entities since it would enable the system administrators to foresee potential issues that may arise during the application of these models for covering user needs. For this purpose, the relation among the elements used for the formal expression of user needs and their impact in the execution of the respective services should be investigated. The interoperable representation of the critical elements and the relations among them would facilitate the application of these models for covering SERRANO needs (e.g., the proper translation of users' intent / goal by the AI-enhanced service orchestrator) and especially the interaction among the components of the SERRANO platform.

5.4.2 Service Orchestrator

Research directions and related work

Service orchestration serves as a mediator between the business need as expressed through the application and service intent, and the resource orchestration process. Its goal is to translate the high-level requirements of an application, service and/or task into infrastructure requirements to be processed by a resource orchestrator operating on top of multi-vendor Cloud, fog and edge infrastructures.

Intent-based Service Orchestration

The authors of paper [192] presented an interesting approach regarding the system and software components specification that facilitates a problem-solving activity. Incorporating the user's intent during the specification and design of the software systems could facilitate the problem-solving activities though the selection and use of the appropriate problem-solving strategy. For this purpose, the specification adapted should support all possible

strategies enabling users to focus on the important parameters of each task and effectively use the relevant information for accomplishing their purpose.

Particular focus is being given in the system engineering process and the *content* of the specification. The latter should also capture the purpose or goal of a system so that we can accordingly use it for solving practical problems during the different phases of a software system (e.g., check if the system behaves as intended). This is further supported by Harman [193] that highlighted that the practical reasoning is concerned with «what to intend» while formal reasoning with «what to believe». Also, the *structure* used in the system and software specification formalism is rather important and it should facilitate users and/or system to locate the subset of data that is often necessary for solving a particular problem. Top-down approaches enable users to deal with quite complicated problems. Also, models of complex systems can be often expressed in terms of a hierarchy of levels of organization. In the part-whole abstractions, each level of a hierarchy represents an aggregation of the components of lower level whereas in the means-end abstraction, each level of hierarchy serves as a goal and the lower level specify the means. Hence, they are very useful for capturing user's intents since they can be used for expressing not only «what» and «how» (also supported in the part-whole abstractions) but also «why» something should happen.

The authors of this work have also proposed a structure for intent specification of software systems. According to this structure the system and software specifications are organized along a vertical dimension about intent abstraction and two horizontal dimensions about refinement and decomposition. The vertical dimension specifies the level of intent whereas the two horizontal dimensions (types of part-whole abstraction) provide a detailed view for the particular intent. More precisely, decomposition separates units into components of the same type (top elements are environment characteristics, operators, system components and verification/validation) whereas refinement breaks down a function into more detailed steps. The intent has five hierarchical levels (i.e., system purpose, design principles, black box behaviour, design and physical representation) with the top levels focusing on the overall goals, constraints, priorities and trade-offs while the bottom layers on design and implementation issues.

Intent-based services have been used in various fields including but not limited to computer networking and system security. Intent-based applications enable users, such as the administrators of a system, to formally express in the highest level of abstraction what should be done rather than the technical details of how it should be done. Hence it facilitates the systems to automatically configure and use different platforms so that the user intentions are satisfied hiding the technical details of this process.

The authors of work [194] have presented an Intent management System for Network Functions Virtualization and end-to-end Network Service orchestration. The system consists of three layers. The Application Layer is dominated by the Intent-based Management System that captures the user's intentions and automatically translates it to the appropriate configurations for each platform. The Management Layer includes the Open Network Foundation's Mobile-Central Office Re-architected as Datacenter (M-CORD) [195] and the ETSI

Open-Source Management and Orchestration (OSM) [196] platforms that are responsible for the service orchestration for the 5G network. The Physical Layer encompasses multiple Virtualized Networking Functions (VNF) that were developed using the OpenAirInterface (OAI) [197] EvolvePacketCore (EPC) and Simulator (SIM).

The Intent Management System (IMS) enables operators to define the Quality of Service (QoS) in the form of contacts that it accordingly translates to the appropriate configurations for the bottom layers (i.e., the TOSCA File [191] for M-CORD and the appropriate JSON messages for OSM). The Intent Management System consists of seven modules (including the Platform Selector, Intent Management, Contracts Design, Resource Manager, Vocabulary Store and Policy Configuration) along with a GUI that enable users to manage their intents. The platform selected is being used by the system for driving the user's intent to the appropriate policy configurator. The user's intent is formally expressed using contracts and consists of single network slice assistance information, network architecture design and download and upload rates. The Intent manager interacts with the other modules of the application layer (including resource manager and the policy configurator) as well as the management layer.

Intent-based Cloud Services for Security Applications (IBCS) [198] is a system that combines the Interface to network security functions (I2-NSF) and the Network Function Virtualization (NFV) architectural framework with goal being the provision of a virtualized security system to secure service consumers. It allows the security service providers and the security service consumers to setup the appropriate security policy without any security expert being involved in this process. For this purpose, the following four steps are taking place. The security service consumer initially expresses their intent to use the desirable security policy in an abstract way (i.e., high level security policy). The IBCS then translates the high-level security policy to the concrete one (i.e., low-level security policy) so that it can accordingly be used for detecting the appropriate NSF from those registered based on the capabilities of each one of them. Finally, the IBCS creates an instance of the selected NSF by also delivering the translated security policy to NSF that will be responsible for the particular policy enforcement.

The above brief description of intent-based service execution highlighted the necessity for a Language, Syntax or Model regarding the formal expression of user's intent in high-level terms. Also, it is necessary a Language, Syntax, or Model for the expression of the capabilities of each Network Security Function (NFS) using low-level terms that are quite close to the functionality provided by the underlying platforms (i.e., parameters configuration). Finally, we need to specify the relation (i.e., correspondence aka mapping) among these two languages/syntaxes/models so that we can automatically translate the user's intent (i.e., their security requirement) to low level requirements so that we can accordingly provide to the appropriate NFS instance (i.e., low level security requirements injection) that could offer this functionality (i.e., enforce the security policy). The mapping of high-level service requirements to low-level service requirements could be specified by cloud-experts and/or assisted by expert systems and AI techniques. The latter could be of particular importance for inferring not only the direct needs of each application but also their future needs in computer resources and hence facilitate the execution of the services.

AI-enhanced Service Orchestration

The Machine Learning (ML) and Data Mining (DM) techniques are useful for learning from data and modelling complex systems behaviour. These techniques are often used in the context of a Knowledge Discovery System that applies different algorithms that belong to the ML or DM field for extracting data patterns that could be potentially useful for the execution of services considering the requirements they should satisfy (quality of service). The unsupervised ML/DM techniques intend to find patterns, structure or generally knowledge in unlabelled data, whereas the supervised ML/DM techniques intend to develop a function or model that understands the collected data (i.e., telemetry data) and hence can make the appropriate choices regarding the services execution and the resources being necessary. The most difficult and time-consuming process of any such system, is the training process that often require a considerable amount of time. Once the system is developed it can accordingly be used for service orchestration purposes.

There are several ML/DM techniques including but not limited to Linear/Logistic Regression [199] (i.e., special cases of Generalized Linear Models) and Gaussian Discrimination Analysis (GDA) Models (i.e., a particular type of Generative Learning Algorithms). GDA makes stronger modelling assumptions than that of Linear Regression and hence it is more efficient when the modelling assumptions are satisfied. Bayesian Networks [200] are probabilistic graph network that representing variables and the relations among them; with the simplest one of such networks being the Naïve Bayes Network. The latter assumes that the parameters/features are independent one of each other and depend only on the class of data. Hidden Markov Models [201] are statistical models that assume that the system is a Markov process (i.e., a collection of system states and potential transition from one to the others) and their goal is to find the parameters of this model. Decision Trees [202] are tree-like structures that can be used for detecting the category data belongs to, after several decision made at each node based on the features/parameters values are recorded. Random Forest is an ensemble learning method that is being composed of many decision trees and it can produce better results, but their outcome is not easily interpretable.

Support Vector Machine (SVM) [203] is a ML technique that intends to develop a hyperplane that separates the data (i.e., maximize their distance from the hyperplane) based on the minimal classification risk. Hence, it is ideal for binary classification problems and in combination with kernel methods can efficiently deal with those cases where the data is not linearly separated. Artificial Neural Networks [199] consists of several interconnected artificial neurons that utilize the information gained from the nodes of the previous layer (perceptions) for the classification of input data. They may contain one or more hidden layers (aka deep neural networks and will be discussion in the following paragraphs) and can generate nonlinear models. On the other hand, they require quite a large amount of data and computer resources for the training process.

Clustering methods [204] intend to find patterns in highly dimensional data with the most widely known methods being hierarchical clustering (i.e., a particular case of connectivity models) and K-means clustering (i.e., a particular case of centroid models). Density models

such as the DBSCAN [205] groups the data points in such way so that their outcome being dense and connected regions. Frequently Pattern Mining and Association Rules intend to discover previously unknown patterns and association rules from the data. The FP-Growth algorithm [206] creates a compact representation of the transaction data recorded in a relational database in the form of a tree that facilitates detection of the frequent patterns (FP), i.e., all the sets of items that appear above a predefined threshold (aka support). The latter can be accordingly used for generating appropriate association rules based on these item sets on condition that the element(s) existing in the left side of the rule, appear above another predefined threshold (aka confidence).

Traditional Machine Learning (ML) techniques such as Linear/Logistic Regression, Decision Trees, Bayesian Networks and Support Vector Machines are based on a predefined list of features. They are parameters of particular interest for the design of the underline model (supervised ML techniques) or the clustering of data in broader categories (unsupervised ML techniques). Consequently, the performance of the AI systems developed is highly connected with the accuracy of the features extracted and used. This is a serious limitation since in many cases we do not already know the parameter affecting the behaviour of the systems such as the parameters with the greatest impact in the service execution. Deep Neural Networks (aka Deep Learning) are multiple level representation methods directly extracted from raw data using general purpose learning procedures [207] and they can efficiently deal with high-dimensional data and discover complex patterns or hidden correlations among them.

The Deep Learning (DL) methods can be classified in three broad categories, that is, supervised, unsupervised and other hybrid methods. Autoencoders (AE) [208] are a particular type of unsupervised DL methods that intend to develop a more concise representation of their input. Deep Belief Networks (DBN) is another type of unsupervised DL method. They are probabilistic generative models that provide an alternative approach to the discriminative nature of traditional artificial neural networks. They consist of several “layers” of neural networks (aka Restricted Boltzmann Machines) that intend to facilitate the classification of data into different categories/patterns. Visually, they look like with a Multilayer NN but their training process is completely different (DBN layers are trained sequentially).

Deep Feed-Forward Neural Networks is a particular type of supervised DL method. They consist of several hidden layers that enable users to capture complex models. Convolutional NN (CNN) [209] is a particular type of Deep Feed-Forward Neural Networks. They typically have sparse interactions and hence the units existing in the deeper layers indirectly interact with most of the units residing in the previous layers. Recurrent neural networks (RNN) are a special type of NN that they have been designed so that they have memory (i.e., their behaviour depends on their past experience) and they also belong to the aforementioned category. Particular types of RNN are the Long short-term memory (LSTM) and the Gated recurrent units (GRU). Long Short-term Memory (LSTM) model exploits the idea of introducing self-loops to produce paths. It facilitates the information to flow through the units without being altered while a special component called forget gate decides whether the information should be added in the cell state or not. Gated RNNs are based on the idea of creating paths

through time that have derivatives that remain stable. They enable networks to accumulate information through the time and hence efficiently use for accomplishing a particular task.

Beyond state of the art

Intent based service orchestration is often driven by decisions made during the system design. The decision made and the mapping rules specified that properly map the high-level requirements to the appropriate resources is often driven the software-based analysis of relevant data collected from the systems as well as literature and previous knowledge. Traditional supervised machine learning techniques enable users to develop models that can be accordingly used for selecting the appropriate resources at the time of the service execution as well as forecast potential requirements that can probably arise in the near future. Deep Neural Networks can automate this process to a great extent by automatically extracting the appropriate features that can be then used for the adequate representation of complex systems' behaviour with goal being to capture the user intent-based needs.

Still the approach has a few limitations. First, it highly depends on the existence of labelled data, which often are not available. Especially deep neural networks require a large amount of data for accomplishing their purpose that it is often difficult to develop and maintain. Datasets already available online may not adequately capture user needs since they have been developed for a different purpose and hence their combination and use for training purposes might not be suitable. Another issue arises from the dynamics of the systems and the user needs. Many machine learning techniques cannot quickly adapt to new changes, especially when the latter significantly affect the optimal solutions of a problem that could not be determined through a closed form expression. Last but not least, intent based applications, which primarily focus on the end users, do not base their execution on the relevant data. More precisely, the contribution of users in this process is often limited through the provision of labelled data (if any) – often developed by different users and goals – during the system design rather than the decisions users make during the system execution and their impact in the selection of the appropriate resources. As a result, the corresponding translated requirements that the resources should satisfy based on the particular intent of each user may deviate from the true users' needs which in turn poses additional difficulties to the infrastructure providers since they should also handle inaccurate, complicated and often conflicting resource needs.

The above critical analysis of relevant technologies in the intent-based service orchestration field highlights the fact that there is a lot of space for improvement. The formal representation of user's intent and the translation of user's intent to the appropriate resources are active research fields and the existing AI technologies cannot adequately cover the topics. Ideally, the system should be capable of understanding what they users want and identify all the technical details based on the capabilities of the system infrastructure so that their needs can be fulfilled. This is a rather complicated process taking into account the various issues that the system should deal with. Given that the user and the infrastructure providers have different optimization goals. The layered architecture being followed by the SERRANO platform simplifies this process to some extent and makes the intent-based service orchestration feasible. More precisely, the upper levels of this platform and especially the intent-based

service orchestrator should focus on the translation of users' intent to the appropriate resources taking into account a large body of available knowledge that may either directly or indirectly come from the end users, leaving the technical details regarding the proper handling of the available resources, to the lower levels of the platform. For this purpose, innovative algorithms and techniques should be developed and integrated with the SERRANO platform so that they can satisfy the user needs through the maximization of the benefits gained from the use of the available resources.

6 SERRANO Use-Cases

6.1 Seamless integration of heterogeneous architectures

A crucial part of complex software systems development and delivery lifecycle is Continuous Integration (CI) and Continuous Delivery/Deployment (CD) – jointly mentioned as CI/CD. CI, in software development, is a practice of building/integrating and testing frequently, all developer working code in a shared code repository. A common practice in CI is to integrate the changed code at least daily. The frequent integration helps the contributors to notice any arising errors and correct them instantly.

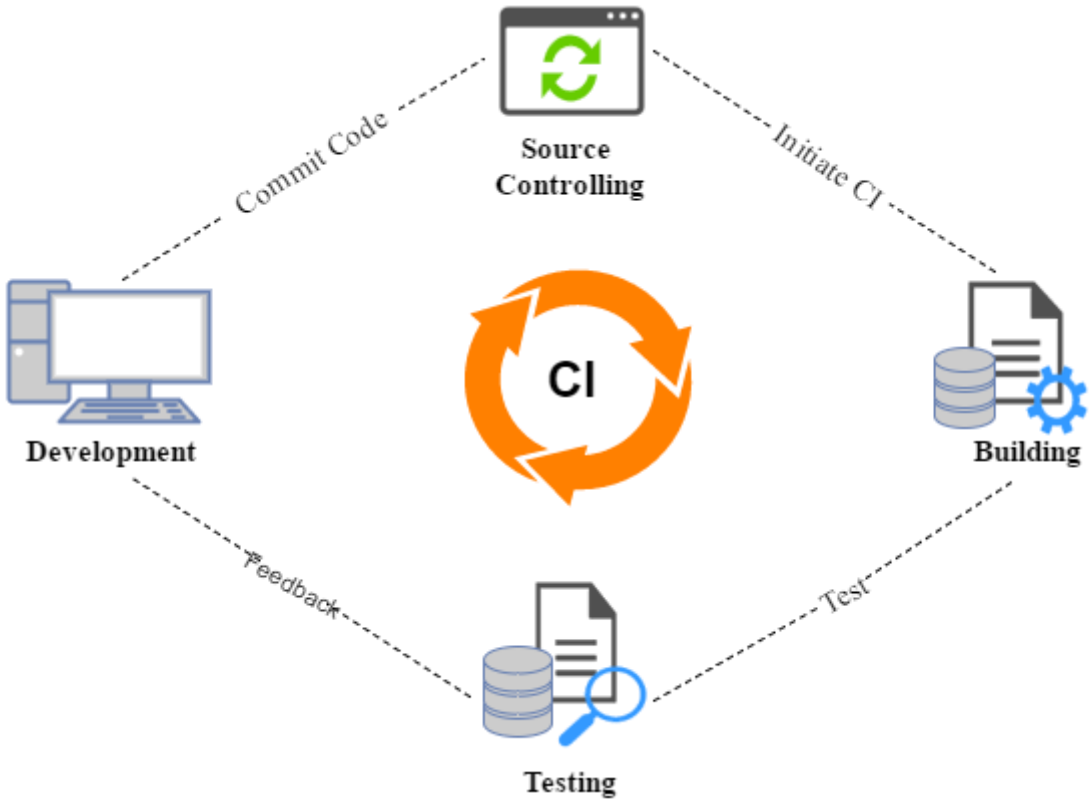


Figure 18: The Continuous Integration Lifecycle

Continuous Integration offers significant advantages such as:

- reduction of risk in the development, since it reveals any possible incompatibility between software components early during the development phase
- facility of instant bug fixing
- availability of current product version at any moment, as the code is frequently integrated

To enable the CI process and benefit from its advantages, we have to perform extensive testing of each software component/module but also of the integrated platform as a whole. Automated tests per component and combined integration tests that are performed on each new build (version) of a component shall be executed and in case of success the integrated

platform will be updated with the new version of the component. Otherwise, the developers will be notified to take proper action to fix the problem that caused the failure. A Test-Driven Development (TDD) approach may be followed, starting from unit tests per software component, upon specified test cases for each component, proceeding to integration tests that validate the correct functionality of the integrated platform that involves two or more components in an automated manner with the use of a CI server, such as Jenkins[35]. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.

In case that the resulting software system is composed of several software components/modules, developed by diverse development teams, a collaborative software development approach can be followed, during which it is important to ensure data integrity and availability. To support this, a distributed version control system (VCS) is necessary for the efficient system software delivery. For this purpose, GitLab[36] could be utilized, a source code management and VCS that aims mostly at data integrity and full version tracking. GitLab is an easy yet powerful and intuitive git VCS. Multiple developers can concurrently create, merge and delete parts of the code they are working on independently, at their local system before applying the changes to the shared GitLab repository. A DevOps framework could be adopted.

To ensure quality of software development, build automation is additionally pursued. Build automation is considered to be the act of automating processes that are associated with software building. Such processes might include various parts like source code compiling into binary code, packaging binary code and automated test running but also the delivery/deployment and documentation parts. Maven[37] is a useful tool for build automation and project management for projects that are written in numerous programming languages (Java, Ruby, C# and other). This process can be applied for components software shared and residing in the software source code repository.

It is also essential in collaborative software development to distribute the software efficiently. Among others, Nexus[38] is a popular repository manager that manages the required software artifacts. It allows developers to distribute their software easily but also to proxy, publish and collect the necessary project dependencies. The Nexus Repository offers a standardized way for cataloguing and storing developers' artifacts. Once a new library is developed, it is handed out to the repository manager. After that, other developers can efficiently access these software components by using a standardized procedure. Clearly, it is possible to control centrally the development of all artifacts and the access to them. Nexus can be used for pre-built software components, the source code of which is not available or shared – however, updated versions of software components need to be frequently built and released in new versions at the Nexus repository, to support continuous integration and delivery. The CI server

[35] Jenkins - Build great things at any scale," Creative Commons, 2019. [Online]. Available: https://jenkins.io/
[36] GitLab - https://about.gitlab.com/
[37] Apache Maven - https://maven.apache.org/
[38] Nexus Repository Pro - https://www.sonatype.com/product-nexus-repository/

will monitor the repository and will initiate a new platform test and deployment cycle after each new version is uploaded.

To further support automated delivery **(Continuous Delivery - CD)** in software development, Docker[39] has been chosen, an open-source software containerization platform, as a straightforward way to provide isolated running environments with pre-set configuration. Docker works with software containers that allow the software to run always the same, independently of the deployment environment. It wraps up the software in a complete file system, along with any necessary tools or software resources, such as libraries, code and runtime. This way, multiple docker containers can run on a single Linux instance, without any overhead for managing several virtual machines. Moreover, by using Docker, the software system deployment is simplified at a great extent, set up is minimal and uniform across all component projects. Each component is contained in a different Docker image ready to be executed in a Docker hosting machine as a separate container. These images can be published in a shared repository, such as the Docker registry, and through the Docker Compose functionality these images can be retrieved from the Docker registry and deployed together via a single configuration file. Containerization thus provides OS level virtualization. This means that multiple applications running in containers on a single host, access the same OS kernel. Hence, it is faster and more lightweight than isolating applications using VMs. Containers have an initial configuration which does not affect the configuration of other containers, even though they share the same host OS. This eliminates errors due to unexpected conflicts or missing dependencies, which are common when applications are installed on a single host without isolation. In more demanding installations due to increased load of the system, Docker is perfectly suitable to be configured with load balancing mechanisms that can scale up the performance of the system.

The following figure depicts the CI/CD workflow with all the tools mentioned above, and summarized below:

- GitLab for source control, acting as code repository and allowing code versioning
- Jenkins for automated build and testing
- Docker for containerization of services and components
- Docker Registry for easy deployment at different infrastructures

[39] Docker: Enterprise Container Platform - https://www.docker.com/

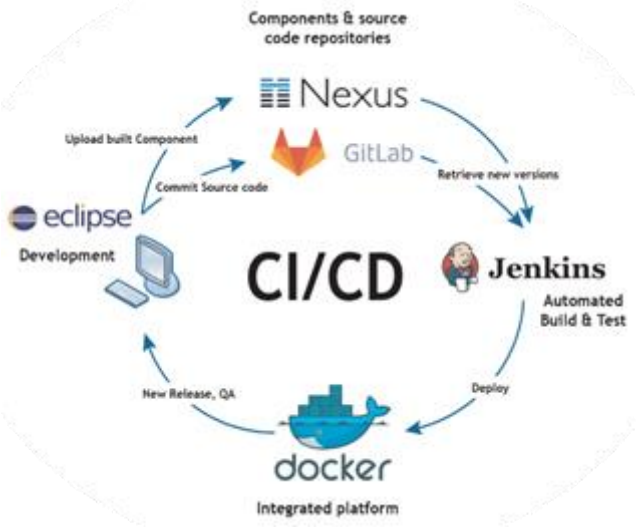


Figure 19: Continuous Integration & Continuous Delivery process

Using the CI/CD environment and tools, when developers implement new component features or integration endpoints, they *push* their code to GitLab, the central source code repository in this environment, which is then compiled, built and tested using Jenkins, while Docker is finally creating a Docker image, that is pushed to the Docker Registry.

Once components have been built and their images have been pushed to the docker registry, they are available to be pulled from *any server* that has access to the docker registry. The Deployment-to-Deployment servers can be carried out via a script which automates the entire process, as shown in Figure 20.

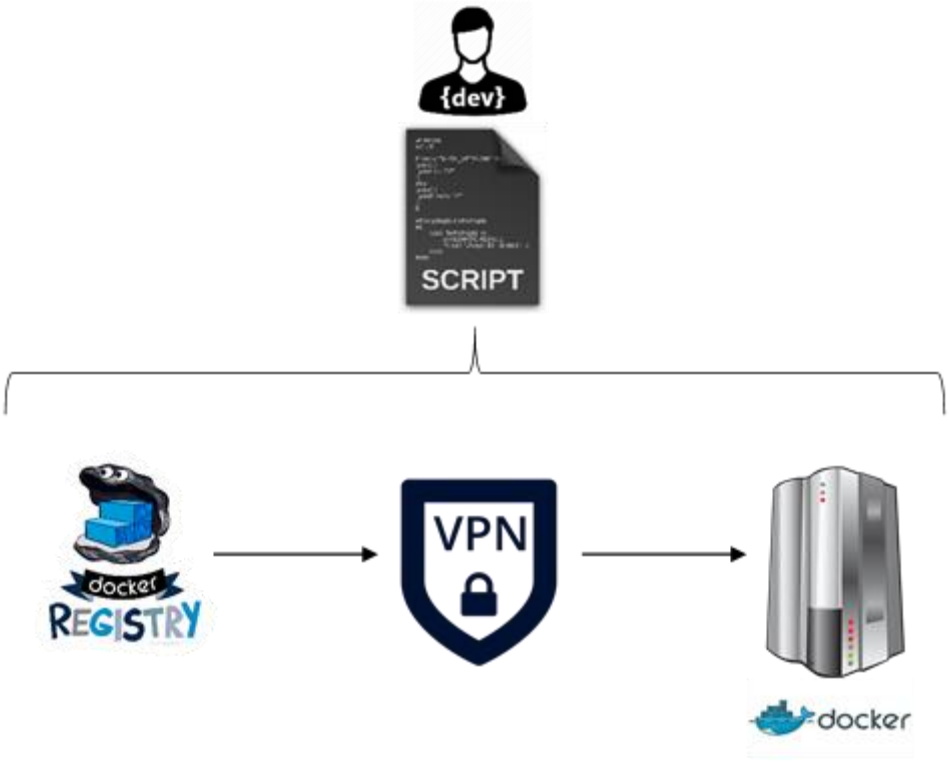


Figure 20: Flow from the docker registry to the Deployment server

Whenever code changes are required, component developers push their source code to GitLab. Afterwards, Jenkins takes over from that point automatically performing the steps described above. Eventually, the component is built into an image and *pushed* into the Docker registry. Then, a new version of the system is specified, and the deployment script is rerun to install the updated software system at the Deployment server. Figure 21 illustrates the flow from code changes (e.g., bug fix, implementation of new features) to re-deployment on any server.

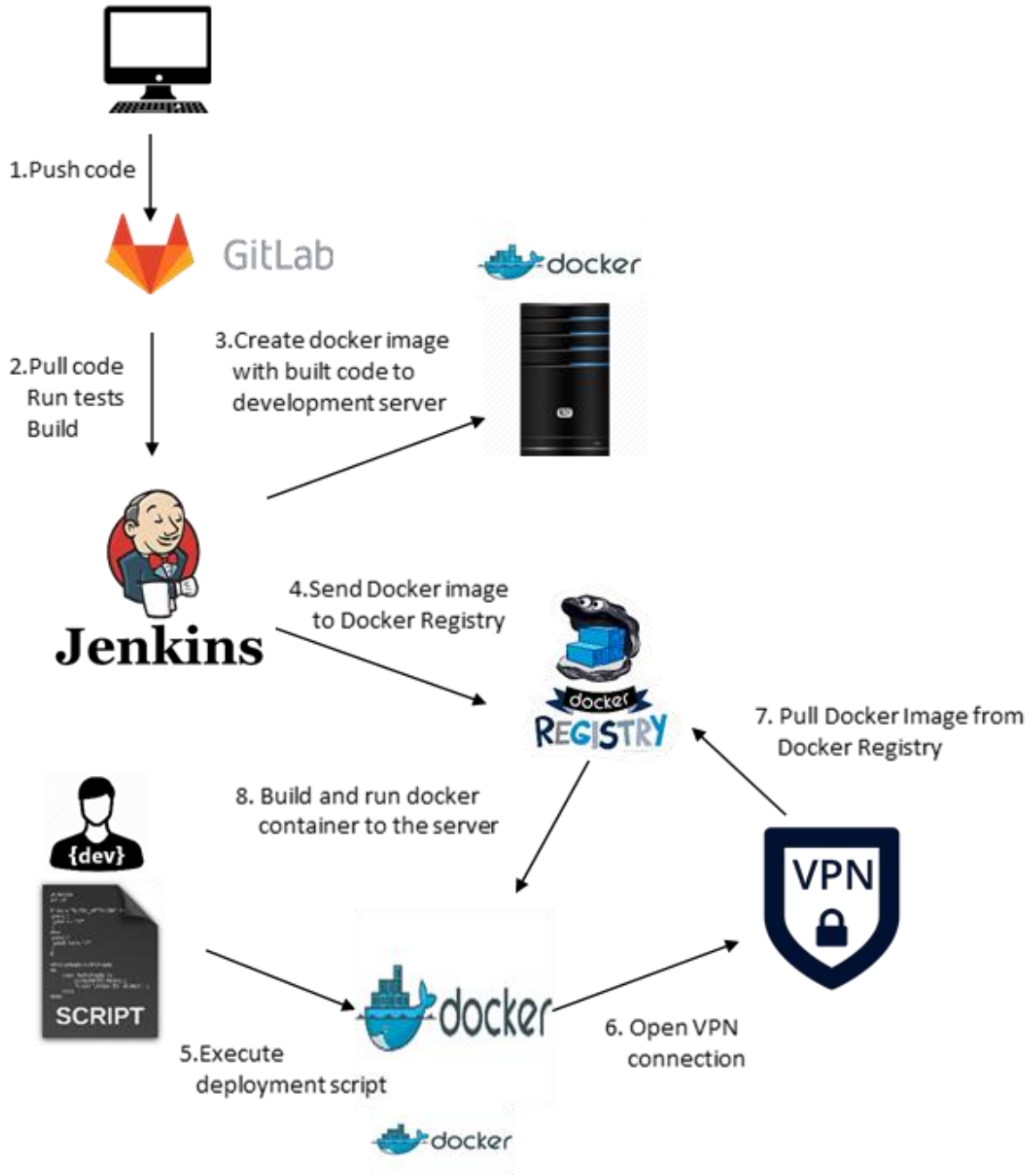


Figure 21: Code changes workflow

Docker containers are ephemeral. This means that when a container is removed, its state (and data) is also lost. To remedy this, docker offers named volumes on which containers mount. These volumes exist independently from the containers that use them.

DevSecOps Approach

DevOps is about developers and operations working closely together to make rapid, iterative changes lo systems quickly and inexpensively. This includes changes to applications and rapid provisioning and configuration of infrastructure - in public or private clouds. DevOps extends Agile and Lean ideas and ways of working into operations: small teams collaborating to solve problems in an open environment; working iteratively and incrementally; minimizing waste and delay; and leveraging automation to solve operations problems such as provisioning, configuring, hardening, updating, and patching infrastructure. Traditional security and compliance culture are at odds with DevOps and Agile values and principles. Security is about understanding and containing risk, controlling change, and protecting information. DevOps is about continuous change, continuous learning and sharing information. In order to enable the flexible collaboration between development, operations and security teams towards releasing frequently integrated but secure versions of systems/software/platform, INTRASOFT International has developed and practices a DevSecOps methodology which is incorporated in its operational Continuous Integration/Continuous Development (CI/CD) stack.

6.2 Secure Storage

The secure storage use case builds heavily on several SERRANO platform services. This is particularly true for the research and development work related to Secure storage across heterogeneous networks. As such, much of the material discussed in Section 3.1 also applies directly or indirectly to the use case. Among the other key features showcased by the use case is the acceleration of TLS connections between the storage system's components using Nvidia Bluefield devices, as described in Section 3.2. There is further potential for accelerating algorithms related to encryption and erasure coding. These aspects are explored in Section 4.1. Finally, the use case will also take full advantage of the SERRANO orchestration tools, described in Section 5 to ensuring good QoS for storage tasks.

Research directions and related work

The key goal of the Secure storage use case is to demonstrate the capabilities of the SERRANO platform in the context of secure file sharing and storage. It achieves this by extending Chocolate Cloud’s commercially available SkyFlok storage service. SkyFlok is a multi-cloud distributed storage solution that uses public cloud providers to distribute user data globally. By the very nature of the cloud resources it utilizes, it is a great choice for individuals as well as small and medium-sized companies that need a cost-effective, reliable and secure service for storing and sharing data. By providing the possibility to select cloud locations that are close to the end users, SkyFlok offers good performance. It is however unable to match the low latency of on-premises storage. This makes it a hard sell when dealing with large enterprises

who have investments in on-premises infrastructure and stringent QoS requirements, at least for some of their applications.

The use case explores adding functionality related to edge data storage and accelerated edge processing. This is done with the hope of overcoming the previously mentioned shortcomings. The overall direction can be defined as expanding the conventional multi-cloud SkyFlok system with edge storage devices and investigating potential improvements to performance, privacy and integration into an enterprise setting such an approach entails. The idea of optimizing the choice for storage locations has always been one of the key selling points of SkyFlok. This is naturally expanded in SERRANO by extending support to edge storage devices. Complementary to this, the use case introduces the idea of moving encryption and erasure coding on premises by introducing a storage gateway. This is done with the hope of improving privacy as well as opening the possibility of accelerating these tasks. Figure 22 shows an overview of the proposed extended architecture of the use case.

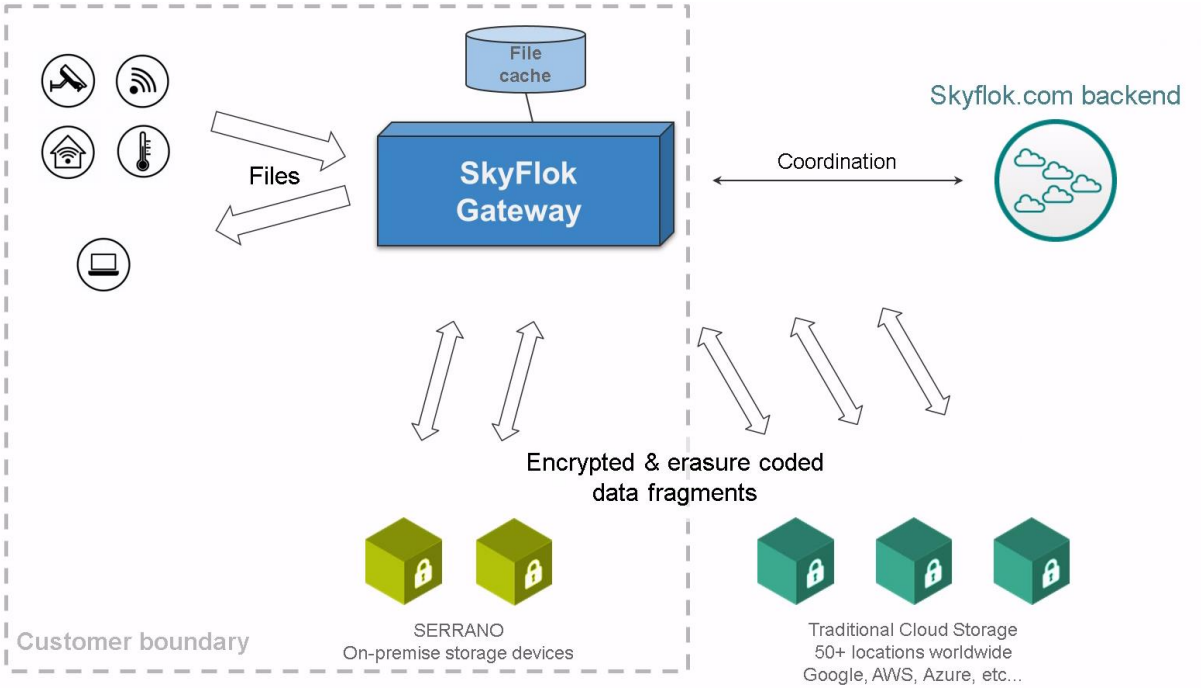


Figure 22: Overview of Secure Storage use case architecture

Finally, the use case explores extending the concept of storage policy, already present in SkyFlok.com. This is a configuration entity that dictates how and where data is distributed to, a crucial component in providing an appropriate QoS level, privacy, reliability and cost-effectiveness. The use case seeks to answer how to:

- extend storage policies to include edge locations,
- either generate a storage policy or choose from an existing list when a new storage task arrives
- adapt/change the policy of a storage task to account for changes in its requirements or the locations of the users or applications that access the data.

The most recent theoretical results that are most closely related to this use case have been discussed in Section 3.1 and Section 5.2. In this part, we focus mostly on the practical implications of these ideas and what commercially available services make use of them.

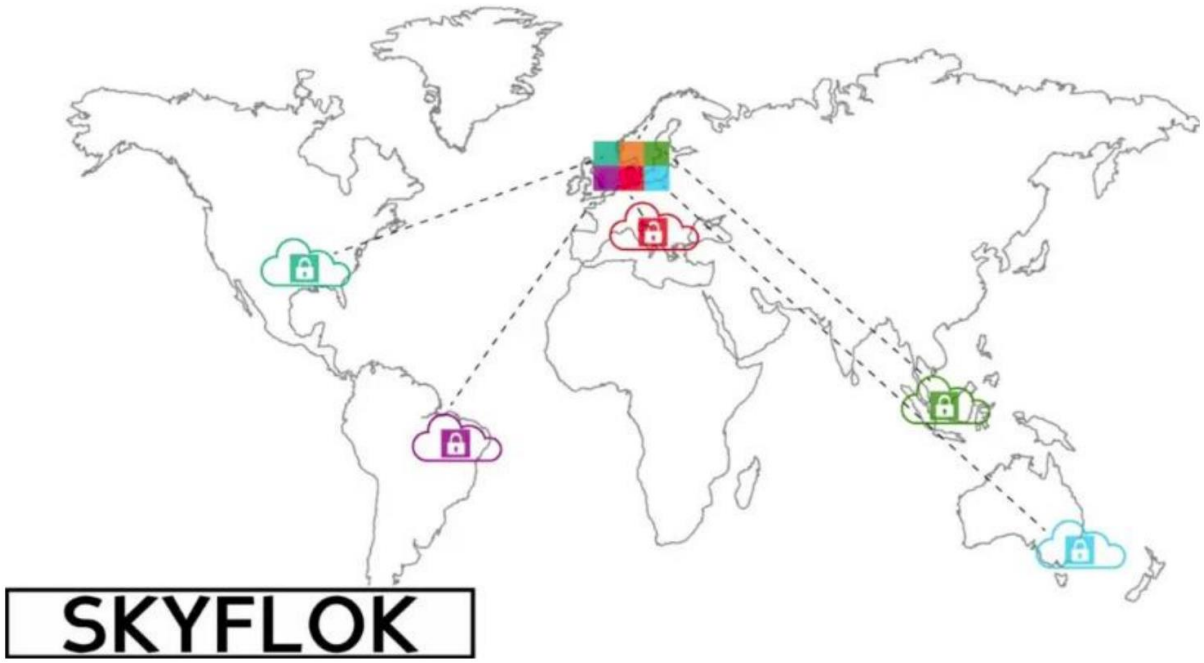


Figure 23: SkyFlok.com secure storage and sharing

Let us first introduce SkyFlok, a secure file storage and sharing service developed by Chocolate Cloud. SkyFlok users can select where their data is stored from a list of reputable cloud provider locations, spread across the world. By relying on proven cloud solutions, the system can offer excellent reliability and performance at a reasonable cost.

Through the use of network coding [6], a relatively novel erasure coding technique, data remains available even if some of the locations face technical difficulties. Using browser-based end-to-end encryption, data is never accessible unencrypted. A key consideration for many customers is GDPR-compliance. This can be easily achieved by a judicious selection of cloud locations, as defined in a storage policy.

SkyFlok is not alone in offering a global-scale storage system to end users. Several companies have emerged within the last decade to challenge the large cloud providers. Most of them offer a product that caters to one or more individual storage needs. Out of these, three share either some of the principles or their key focus with SkyFlok.

Tresorit[40] is probably the closest competitor in terms of the feature set. It offers a file storage and sharing product to both individuals and companies. Unlike the market leaders in this segment (e.g., Dropbox, Google Drive, Onedrive, etc.), Tresorit, like SkyFlok, offers end-to-end encryption. This protects users' data from being accessed by the cloud providers. Internally, Trezorit is hosted in the Azure cloud environment, with data located in data centres in the EU,

[40] Tresorit – https://tresorit.com/

North America and Switzerland. Whereas Trezorit does not allow users to tailor the choice of storage locations to the same degree SkyFlok does, it does feature another important privacy-enhancing capability: Zero Knowledge Encryption (ZKE). ZKE is a combination of techniques that give the user total control over the encryption of their data. Crucially, by not exposing the encryption keys to the service provider, it makes it impossible for Trezorit themselves to read the user data. As of the writing of this document, Chocolate Cloud is also in the process of implementing ZKE for SkyFlok, with an open public beta version available to try.

Storj.io[41], recently rebranded from Tardigrade, is another multi-cloud, technically very interesting and promising storage solution. Unlike most products in this segment, it is aimed at a specific subset of end-users, software developers. Instead of relying on conventional cloud providers to store data, it instead creates a sort of marketplace where storage locations can advertise their services. Given that these locations are inherently more unreliable compared to an object store hosted by one of the large cloud providers, it constantly monitors and repairs the data it stores. This high-churn environment is unsuitable for replication-based redundancy as it is prohibitively expensive [9]. To provide an acceptable level of reliability, the number of replicas becomes so large that the storage overhead of storing them would make such a system impractical. Instead, Storj.io uses Reed-Solomon [8], a well-known and proven erasure code in a 27+53 configuration (also referred to as n=80, k=27). This means that each piece of data is distributed to 80 storage locations, with at least 27 needing to be available to recover the data. This extreme configuration in terms of the large number of redundant fragments, is needed to be able to provide good reliability in a high-churn environment of inherently unreliable nodes. It also addresses the long tail effect, one of the key obstacles to achieving good latency. For example, if a read can be served from more than 27 locations, the system might choose to start reading from slightly more, say 35 locations and cancel the final 8. This means stragglers do not have nearly as big of an impact on retrieval performance as would normally. This idea is also implemented by SkyFlok. The choice of erasure code is quite interesting. Reed-Solomon has been used in large-scale data centre storage previously, for example by Facebook with varying success. It greatly reduced storage costs compared to replication but was found to have prohibitively high bandwidth requirements [210]. Because the storage nodes are likely to fail at some point, data must be repaired. With a conventional erasure code such as Reed-Solomon, this requires a k-times amount of data transferred compared to the lost data. Repair transfers quickly overwhelm the network, occupying bandwidth vital to the normal operation of the storage system [210]. A well-known solution to this problem is to use a regenerating code [7], an erasure coding technique that reduces repair traffic to manageable levels. Network coding, used by SkyFlok, is one of the practical realizations of this idea. Storj.io instead uses the concept of lazy repairs, introduced in [211]. The key idea is to delay repairing missing pieces until enough have accumulated, then perform a more efficient repair. This is likely also one of the reasons for the unusual 27+53 configuration used for Reed-Solomon.

[41] Storj: A descentralized Cloud Storage Network Framework – https://www.storj.io/storjv3.pdf

Not so much a direct competitor to SkyFlok, Trezorit or Storj.io, but a very interesting alternative to the cloud-based solutions is the Interplanetary File System (IPFS) [212]. If there was a step between SkyFlok and Storj.io in the direction of decentralization and democratization of storage, there is another leap when moving towards IPFS. IPFS is a community-based P2P platform that offers a distributed setting for a wide range of services to be built on top of it. It is a natural environment for offering distributed data storage and sharing. Several file sharing applications have been built on top of IPFS, some like Filetribe [213] backed by a blockchain. While blockchain technology offers a good way of maintaining a global ledger in a decentralized environment, relying on it for storing frequently accessed metadata greatly limits the scalability of such a system. Therefore, some form of centralization is needed to offer good QoS, a somewhat undesirable characteristic when dealing with untrusted nodes.

While all these systems offer reliable, secure storage, none, including SkyFlok is able to achieve latencies comparable to an on-premises solution. Through this use case, the SERRANO platform will find a good balance in the typical trade-offs between security, low latency and cost. This is achieved by on-premises components in the form of a storage gateway and edge storage locations. In this sense, the approach further decentralizes the original SkyFlok architecture in terms of where data is stored, while centralizing some operations in the gateway.

Beyond state of the art

Building on several of the ideas discussed in Section 3.1, this use case will showcase a hybrid file storage and sharing solution that gives its users the choice to store data spread across commercially available cloud providers and local, on-premises edge devices.

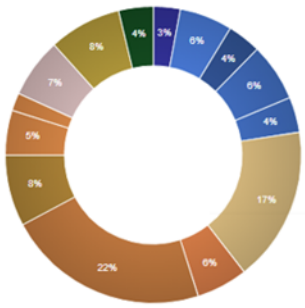
This solution will see improved latency and QoS thanks to its ability to move data closer to where it is being used. With the use of network coding, a seamless integration of heterogeneous storage locations can be achieved. Network coding also augments other privacy-ensuring techniques thanks to the random selection of coefficients that are used to create the coded fragments.

The system will operate at its full potential if each of its heterogeneous storage locations is used in according to its strengths. To accomplish this, the system will offer tailored storage policies to the specific requirements of each storage task. It might also be possible to adapt the way data is stored, should these requirements change over time.

Finally, the on-premises storage gateway will offer a subset of the S3 API. This will allow for easy integration with existing enterprise systems. By moving encryption and erasure coding to this trusted component, these operations can be accelerated. Another argument for this move is the enhanced control over data privacy it provides, as enterprises may want to customize or augment the encryption techniques used. For example, it should make it easier to offer zero-knowledge encryption through a 3rd party application.

6.3 High-performance Fintech Analysis

InbestMe (INB) is a roboadvisor that provides automatized services for investment management. INB will contribute and lead the research and development of a use case for investment management. The investment management is a continuous process of constructing investment portfolios composed of investment instruments such as shares, ETFs, funds, options etc. Typically, an investment portfolio may be composed of tens to thousands of instruments. Figure 24 shows an example INB portfolio.



Asset type	Value	%	Ret.
Preffered Shares	431,86 €	2,94%	-3,3%
Aggregate Bonds EU	849,21 €	5,78%	1,6%
Long term Government Bonds USA	538,38 €	3,67%	-3,7%
Emerging Market $ Bonds (H€)	925,70 €	6,30%	10,2%
China Bonds $	564,85 €	3,85%	1,5%
Equity Europe	2.485,00 €	16,92%	23,6%
European Equities Dividend	824,07 €	5,61%	13,7%
US Equities (H€)	3.247,92 €	22,12%	29,4%
Equity Emerging Markets	1.141,24 €	7,77%	34,5%
Equity Japan	717,96 €	4,89%	17,5%
Asia Pacific Equity ex Japan	286,16 €	1,95%	9,7%
Real Estate Equity EU	955,15 €	6,51%	11,9%
Gold and Other Precious Materials	1.162,79 €	7,92%	4,7%
Cash	553,33 €	3,77%	0,0%
Total	*14.683,62 €*	*100 %*	

Figure 24: Example investment portfolio composed of different classes of investment instruments. Each class contains several other investment instruments.

Research directions and related work

The main research direction that INB will focus and contribute to is increasing the automatization in investment management [214]. Higher automatization means developing and extending its own platform to perform more process automatically including self-monitoring and requiring minimal or no human involvement. INB identifies tremendous opportunity in automatization within the finance sector. The finance sector utilizes well understood business processes that unfortunately are performed manually or poorly integrated [215]. Automatization would respectively decrease the human errors as well as increase the accuracy due the capability of processing and analysing large volumes of data obtained from different sources [216].

In order to increase the level of automatization, INB will research and develop secure deployment and launching of large number of fintech processes related to continuous investment management [217]. It will rely on the SERRANO system monitoring capabilities to implement self-monitoring as well as unattended operation.

The finance sector is a vivid example that successfully benefits through digitalization and automation from the innovation in the computing and cloud computing technologies [218]. The strong links of finance with the statistics and numerical analysis, built the ecosystem for financial computing. During the past two decades, the application of computing was further extended to market predictions using the innovations in Machine Learning and Artificial Intelligence. As a result, computers find ideal application in automated trading and investment management. The emergence of cloud-computing further facilitated the innovation in finance and became one of the key-enablers for the FinTech industry.

In the current state-of-the-art in investment management, cloud computing is used primarily for digitalizing and automatizing business processes such as market analysis, portfolio construction, and trading [219]. For example, all fund managers, use computing power to process and analyse market data. Others, like InbestMe, also automatically create and place trade orders. Large funds like Medallion [220], Volean Group, PDT Partners apply ML and AI for decision support and market predictions [221] [222]. To speedup calculations and increase accuracy, large financial institutions like banks create their custom solutions based on accelerated devices such as Maxeller Technologies [223]. Many of existing solutions can be applied for a single fund or multiple portfolios.

While cloud computing is a key enabler for FinTech, it still has a lot of potential for innovation beyond state-state-of-the-art in application deployment and management, security, as well as scalability.

Beyond state of the art

SERRANO will advance the state-of-the-art by research and development in cloud computing that will:

- Facilitate application deployment and management (update, migration, monitoring, replication) between private and multiple third-party cloud infrastructures. Thanks to this, INB will be able to deploy and scale its application between private and third-party machines. Additionally, such functionality will enable INB to offer its applications to other financial institutions that want to host it on premise [224];
- Research and develop lightweight serverless cloud functions (FaaS). Thanks to this, INB will be able to scale its functionality and offer Roboadvisor-as-a-Services (RAaaS) to other financial institutions [225].

Also, SERRANO's research and develop activities will enable secure storage and execution of business applications and services to third party resources in a secure manner.

6.4 Machine Anomaly Detection in Manufacturing Environments

Research directions and related work

A failure in industrial equipment results in loss of productivity, increased maintenance costs, delays in supply to customers, and may even lead to safety and environmental impact. To avoid unexpected problems, maintenance strategies have evolved from corrective to preventive, then to condition-based maintenance (CBM), and lately towards intelligent predictive maintenance systems. Condition-based maintenance, also known as predictive maintenance, recommends maintenance actions (decisions) based on the information collected through condition monitoring process [226]. Three key components of CBM are data acquisition, data processing and decision making [227].

The issue of automatic diagnosis of industrial systems has attracted considerable attention for a long time [228]. Technological advances are being made in machine monitoring, production quality control or predictive maintenance. One important source of information are noise and vibration signals. An important amount of research work is being produced by the communities of control and artificial intelligence, in the areas of design of algorithms for effective detection and diagnosis of anomalies in real time.

Monitoring of machines to check their state of degradation by the use of key parameters (e.g., temperature and vibration) is done either using an additional network of sensors [229] or signals generated by the control system of the machine (e.g., position, speed and drive current consumption) [230]. A wide range of failure prediction methods have been explored to date, which can be broadly categorised into model based and data-driven techniques [227] [231]. A premise condition of the model-based methods is the prior knowledge of both the physical system and the mathematical foundation. The model-based techniques have demonstrated its suitability in multiple manufacturing processes and systems. Recently, the reduced cost of machine sensors, the enormous growth in computational power and the tremendous surge of the cloud systems have driven more attention to the data-driven alternative, underpinning the generalisation of statistical methods and machine learning algorithms at scale [231].

In relation to fault prediction service, intelligent systems based on IoT are proposed [232]. The IoT acts as an enabler for more efficient continuous maintenance, but the need for fundamental understanding of the degradation mechanisms and root causes for the failure modes remains unchanged. The three major challenges in using IoT for the fault prediction of a machine group are: (1) communication of data from the IoT sensor network, (2) non-stationary and nonlinear fault prediction and (3) massive data processing [233].

By applying a structured engineering thinking and techniques to the industrial use of AI or machine learning, a transformation from collected raw data to enriched information or smart data will be generated. The smart data are responsible for detecting relevant signals and patterns through intelligent digital signal-processing algorithms. Smart data makes sense out of big data, providing actionable information and improving decision making [234]. Another

major challenge is to develop fast computing models that are accurate and plausible enough to help improving the production processes [235]. By generating context-related persistent datasets, every manufacturing process in real production becomes an experiment. The vision of Internet of Production (IoP) is to enable real-time diagnosis and prediction in smart productions by acquiring datasets seamlessly from different data silos. This requires interdisciplinary collaboration and domain-specific expertise [235]. The main components of the machine become cyber physical production systems, mechatronic systems monitored and controlled by software brains and digital Information [236].

In specific CPS-based maintenance, the information of machine status and condition monitoring information is sent continuously to a big data storage system (e.g., Cloud). Big data analysis algorithms watch and analyse all incoming data. Maintenance plans and schedules are derived based on the results of the big data analyses. Maintenance activities are scheduled depending on the machine condition. The maintenance plan is constantly adapted according to the machine status and work schedule [237]. Use of IoT as an enabler for continuous maintenance is still at its infancy. Figure 25 shows a scheme for condition monitoring of engineering systems using IoT and cloud computing [238]. With the IoT and the cloud, condition data from various modules of an engineering system distributed across multiple locations can be collected and analysed together using cloud-based data fusion and data analytics. The knowledge about the system health can then be fed back to the design team to achieve a closed loop design process.

The reality of IoT platforms is complex because applications and solutions come with different architectures, ways of connecting and managing devices, possibilities for managing and analysing data, capabilities to build applications, and options to leverage IoT in a meaningful way, for any given use of IoT. in any given context. At the edge, the data come in from the physical world via sensors, and actions are taken to change physical state via various forms of output and actuators. The data processing, analysis, and evaluation are performed at the edge. The communications have real-time and deterministic behaviour, while the quantity of the analysed data is reduced. The cloud is a huge, interconnected network of powerful servers that performs services for business. In concrete, Savvy Data Systems' cloud, used massively by IDEKO and DANOBATGROU (Figure 26), can be divided in three major parts: machine to cloud, cloud processing, storage and modelling, and cloud data analysis and presentation [234].

Acceleration sensors have high sensitivity and can be used for chatter detection, condition monitoring or collision detection. They are used for gathering data at high-speed sampling frequency. Transferring all this high-frequency data to the cloud platform is seen as burdensome and costly, so local processing of the acceleration signals is performed to extract the most meaningful information.

The Fast Fourier Transform is used to obtain the frequency spectrum of the vibration, and only the highest peaks are transferred to the cloud platform, the vibration severity is also computed for different frequency ranges. Hence, the vibration level can be reflected with a limited amount of data thus increasing the ratio between data and information [239].

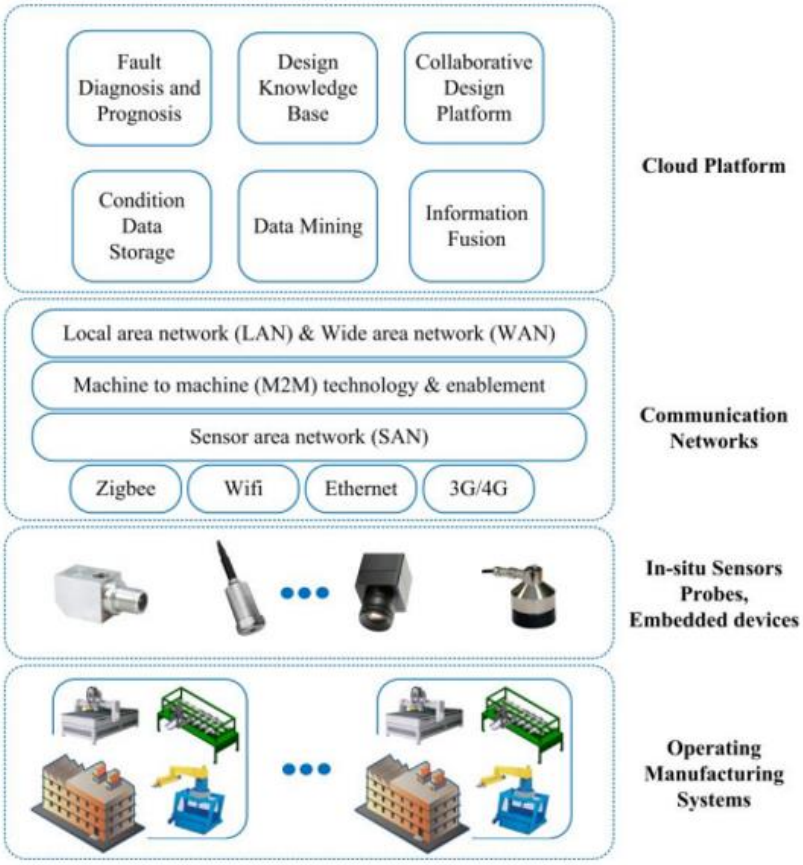


Figure 25: Condition Monitoring common architecture

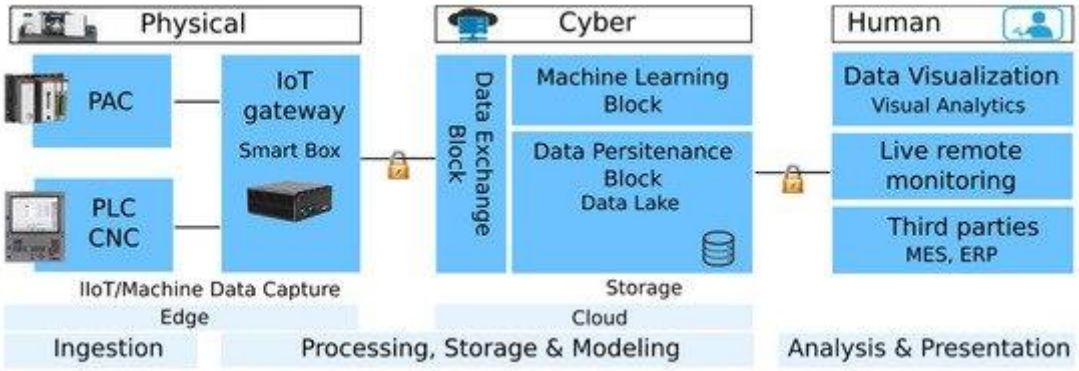


Figure 26: Architecture of the Cyber physical connected machine

Beyond state of the art

High-frequency and high-accuracy sensors used for data acquisition, generate large volumes of data that could be gathered in a continuous manner, opening the way to innovative data driven diagnosis techniques. This scenario is challenging with the current state of technology, since the volume of data generated is difficult to process in real-time at the edge, considering the capacity limitations of the resources. Introducing mechanisms that orchestrate optimally data and computationally demanding tasks in the edge, cloud and any other available computing resources will overcome this obstacle.

This SERRANO use case proposes an approach where these analyses are performed continuously, while the hardware equipment keeps running most of the time and the state of the various independent components, along with the overall status is continuously reported.

7 Overview of Technologies and Projects

This chapter focus on detailing pertinent technologies that each partner foresees to bring into SERRANO portfolio as an existent asset but also to extend and integrate it as a SERRANO component. Further, a list of research projects relevant to SERRANO objectives are presented, outlining the potential aspects that can be of interest for the project.

Table 3 depicts the components that each partner plans to use as an asset in the SERRANO technological portfolio along with the estimated current TRL level and post project targeted TRL.

Table 4 presents relevant past and current national and international projects, as well as where SERRANO intends to capitalize on their results that are relevant to SERRANO activities and to project's objectives.

Table 3: SERRANO components

Component	Present TRL Post-project TRL	Justification
Power measuring analysing system for HPC and Cloud Computing *Partner: USTUTT*	4 - 7	EXCESS cluster environment has been employed by users in several projects to identify the critical paths in the considered applications and its hardware usage (incl. CPUs, GPUs, Network). The results will be demonstrated in operational environment among others on HPC system Hawk. **WP the target TRL is achieved in:** 5
Vampir live *Partner: USTUTT*	7 - 9	The tool is integrated in HPC system "Hawk". It disposes the important metrics during the applications' execution (live) and can be used for optimal (performance- and energy-wise) distribution of the resources (compute node, IO-buffers, network). The tool will also be used for the validation of the optimizations, performed among others with the help of the EXCESS cluster. **WP the target TRL is achieved in:** 5
Dakota framework *Partner: USTUTT*	4 – 5	VVUQ will be investigated using the Dakota software; We will integrate the Dakota framework of the USTUTT HPC execution environment into the SERRANO environment to validate the considered application use cases with input data of various levels of precision. **WP the target TRL is achieved in:** 5
skyflok.com with edge storage *Partner: CC*	3 - 7	CC will take TRL skyflok.com solution for distributed storage and extend it to enable edge storage. There is significant expertise and tested software for network coding and encryption that have been deployed and tested on both Intel x86 and ARM architectures. The TRL targeted at the end of the project is realistic given the existing expertise, the availability of a solid backend from skyflok.com already deployed with network coding primitives, and the presence of edge device partners that will allow us to implement solutions in real hardware and test it under realistic conditions as one of SERRANO prototypes.

		WP the target TRL is achieved in: 3
Distributed storage with crypto-cipher acceleration *Partner: CC, MLNX*	2 - 5	CC will develop extension APIs to include crypto-cipher acceleration through hardware off-loading available in MLNX smartNICs. **WP the target TRL is achieved in:** 3
Automated investment decision support system *Partner: INB*	3 - 6	The investment decisions are currently taken by experts in the investment committee. At the end of this project we target to build an automated investment decision support system, which will utilize SERRANO platform and will advance the work of the investment committee by preparing and analysing several investment strategies. **WP the target TRL is achieved in: 5**
Acceleration for TLS symmetric crypto *PARTNER: MLNX*	2 - 5	Acceleration of NVMEoTCP and TLS through zero-copy, CRC32 and TLS overhead reduction, with a performance target of 6x reduction in CPU utilization. **WP the target TRL is achieved in:** 3
In-line AES-XTS encryption for distributed storage *PARTNER: MLNX*	2 - 5	MLNX will develop a novel in-line AES-XTS encryption of data using NVMEoF as the network protocol and NVMe flash as the physical drives to enable secure data storage through heterogeneous cloud fabrics. **WP the target TRL is achieved in:** 3
Plug&Chip API *PARTNER: AUTH*	2 - 4	AUTH has developed a Cadence-awarded framework for rapid prototyping that favours hardware/software kernels interoperability (Plug&Chip [32]). This framework will be extended to also handle many-accelerator approximate kernels by considering computations, security and communication aspects. **WP the target TRL is achieved in:** 4
REMAP Framework *PARTNER: AUTH*	2 - 4	REMAP framework enables the modelling and the design of variable-accuracy optimizations based on approximate computing. It will be used to support workload-aware run-time adaption among approximate kernels to significantly improve efficiency (hardware utilization and performance per watt metric) without affecting the application's/service's QoS. **WP the target TRL is achieved in:** 4
DMM4FPGA: Mitigation Dark Memory in Many-accelerators platform *PARTNER: AUTH*	2 - 4	Accelerators' scalability problem due to resource under-utilization in FPGA-based heterogeneous many-accelerator platforms is a problem similar in nature with the well-known "Dark Silicon" in future many-core systems. The proposed solution was applied to Xilinx Vivado HLS framework to address the static memory allocation problem, which is the de-facto memory management mechanism supported by modern design techniques and synthesis tools. **WP the target TRL is achieved in:** 4
DMon Monitoring Service *PARTNER: UVT*	4 - 5	Scalable distributed monitoring service with anomaly detection system tailored for data processing frameworks (tested in H2020 DICE and ASPIDE) will be adapted for integration in the designed platform and tested in conjunction with the use cases. **WP the target TRL is achieved in:** 5
EDE Service PARTNER: UVT	4 - 5	A scalable distributed event and anomaly detection tool which uses ML techniques. It also includes optimization mechanisms for

		transprecise computing (tested H2020 DICE and ASPIDE, Chist-Era DiPET). **WP the target TRL is achieved in:** 5
Big Data Stream Handler (Edge/Cloud) *PARTNER: INTRA*	6 - 7	The Big Data StreamHandler is currently operational at the cloud and will be extended to handle data streams at the edge. **WP the target TRL is achieved in: 7**
Hardware Acceleration for Serverless *PARTNER: NBFC*	2 - 5	Serverless frameworks are an ideal candidate to enjoy hardware acceleration functionality. However, current approaches require complex software stacks and prevent hardware resource sharing. SERRANO introduces an efficient and secure hardware acceleration framework for Serverless, using virtualization techniques for paravirtual devices while keeping the backend generic, using common acceleration frameworks such as Tensorflow, PyTorch etc. **WP the target TRL is achieved in:** 3

Table 4: National and international projects SERRANO will capitalize on

Projects	Brief Description	Potential strategies
EXCESS	EXCESS focuses on the Exascale software challenge and addresses this challenge through the process of software co-design.	EXCESS provides the methods and tools to optimize important properties, such as performance, execution time and energy efficiency, of a wide range of hardware and software systems. **Related Objective(s):** 4,5
DreamCloud	DreamCloud provides framework allowing users to easily specify mappings for cloud applications and assess the mappings quality (execution time and energy consumption).	We intend to use parts of the framework to examine the possible resource distributions for the energy efficiency. **Related Objective(s):** 5
ExaSolvers & ExaSolvers 2	These projects, funded by the German Research Foundation, aimed at development of highly scalable solvers for numerical applications. USTUTT contributed to this by investigating the energy efficiency of developed solvers.	We intent to provide the software and hardware environment for use the highly scalable solvers for practical engineering problems as part of the digital services. **Related Objective(s):** 5
ExaFLOW	Its target is to address algorithmic challenges to enable the use of accurate simulation models in exascale environments. Driven by practical engineering problems, it focuses on important simulation aspects, including heterogeneous modelling and evaluation of energy efficiency in solver design.	We intent to provide the software and hardware environment for use the highly scalable solvers for practical engineering problems as part of the digital services. **Related Objective(s):** 4,5

HPCWE	One of the objectives of this project is to analyse the uncertainty of wind power generation. Particularly, the influence of coupling the multi- scale simulations is investigated with help of VVUQ framework based on Dakota software integrated in USTUTT HPC environment.	We intent to provide the software and hardware environment for use the VVUQ framework. **Related Objective(s):** 4,5
AII – AI Investments	AII is a Eurostars project that has the objective to build AI and ML solutions for investment management.	INB is a partner in AII and will use its results as a benchmark for the project platform. **Related Objective(s): 6**
DITAS	DITAS offers a framework, including an SDK and an execution environment, which aims to overcome the barriers that hamper the adoption of Cloud computing and boost adoption of Fog computing by exploiting the full potential of these two paradigms in a synergic way.	The framework will be used to support the development and execution of data-intensive applications. The provided tools will be used to manage application data in an efficient, reliable, scalable and secure manner. **Related Objective(s):** 5
4CaaST	4CaaST main goal was to accelerate the creation of highly demanded tailored services in a timely manner.	We expect to use the service resolution engine, component that is crucial for the service matching as also the blueprint schema that will simplify the interoperability of the hardware and software description and usage. **Related Objective(s):** 5
CloudPerfect	CLOUDPERFECT aims at delivering a set of tools and processes that will enable Cloud providers to enhance the stability and performance effectiveness of their infrastructures, through modelling/understanding of the overheads.	We will use the tool that evaluates the independent validators of Cloud QoS features, through a constant monitoring, benchmarking and evaluation process. It is designed to help in the current brokerage/consultancy domain for performance evaluation and SLA auditing. **Related Objective(s):** 5
Data-centric Approximate Computing (Greek project)	Framework for designing data-centric approximate accelerators for energy efficient computations on application-agnostic workloads.	Extend existing framework to support workload-aware run-time adaptation among approximate kernels. This will significantly improve efficiency (hardware utilization and performance per watt metric) without affecting the

		application's/service's QoS. **Related Objective(s):** 4
5G-PHOS & 5G-COMPLETE	The 5G-PHOS project focuses on 5G integrated Fiber-Wireless networks that leverage existing photonic technologies towards implementing a high-density SDN-programmable network architecture. Similarly, 5G-COMPLETE aims to revolutionize the 5G architecture, by efficiently combining compute and storage resource functionality over a unified ultra-high capacity converged digital/analog Fiber-Wireless Radio Access Network.	The hardware implementation of DSP (Digital Signal Processing) processing at FPGAs and "Bluefield 2" will be used for enabling the physical implementation of low-latency data movement from edge devices to the fog and Cloud infrastructure. Additionally, the use of virtualization technologies developed in 5G- COMPLETE facilitate the integration of security and hardware acceleration features. **Related Objective(s):** 2,3,4
OPRECOMP	OPRECOMP proposed recently software techniques that enable tunable precision through arithmetic and data storage and communication for high-end supercomputers and low-end embedded systems.	The techniques related to fog devices will be investigated for their appropriateness for the dynamic allocation mechanisms. **Related to Objective(s):** 4
ASPIDE	The action "Exascale programming models for extreme data processing" provides mechanisms for processing massive amount of data at high speed and/or real-time	Use of the scalable distributed monitoring service with anomaly detection system tailored for data processing frameworks. **Related to Objective (s): 2**
DIPET	The DIPET project (CHIST-ERA programme) investigates models and techniques that enable distributed stream processing applications to seamlessly span and redistribute across fog and edge computing systems using an approach based on transprecision computing.	UVT is partner in DIPET and will use the open- source tools under development that are using machine learning to establish the proper (trans)precision for distributed streaming applications. **Related to Objective(s):** 4
SecureCloud	It aims to ensure the dependability of critical applications executed in distributed, potentially untrusted cloud infrastructures. Its novel approach to cloud dependability leverages the emergence of a new secure commodity CPU, promising to enable a new generation of dependable applications by basing trust in hardware mechanisms offered by commodity CPUs, in particular, IntelTM's Secure Guard eXtensions (SGX). It enables applications' isolation from other applications, the underlying operating system and the	Although this project has finished in December 2018, CC has developed several micro- service architectures and security components that can be used in SERRANO. **Related Objective(s)**: 5, 6

	hypervisor. Users can run their sensitive applications in public clouds with no need to unconditionally trust the cloud provider.	
Danish Market Development Fund Project	This project focused in allowing CC to test and deploy its commercial product (skyflok.com) for secure multi-cloud file storage and sharing using network coding to combine public cloud providers.	CC will bring the backend and the expertise gain in this project and extend it to integrate edge devices into its offering as well as new potential applications derived from these new capabilities. **Related Objective(s):** 5, 6
ATMOSPHERE	ATMOSPHERE aims to design and implement a framework and platform relying on lightweight virtualization, hybrid resources and Europe and Brazil federated infrastructures to develop, build, deploy, measure and evolve trustworthy, cloud-enabled applications.	Although the project is focused on Cloud-only solutions, we will maintain interactions with the consortium and, when applicable, learn and use in the context of edge security and data offloading. **Related Objective(s):** 2, 3, 5
CYCLONE	CYCLONE integrates and extends open-source software to create a unified cloud application management solution for application service providers, DevOps, and researchers.	Its framework for distributed logging for federated cloud applications will be considered as an option for SERRANO cloud and network telemetry. **Related Objective(s):** 5

8 Conclusions

This deliverable presents a comprehensive analysis over the current scientific and technological advances in different topics of interest for the SERRANO project. The topics of interest were selected based on the project objectives (as described in Section 2.1) to ensure the relevance of this analysis with the SERRANO goals. These topics were finally grouped in four main sections from which three sections covered security, acceleration and orchestration issues for building a platform able to satisfy the conditions of cloud-to-edge continuum bridge. The last section was dedicated to the use cases (UCs) with emphasis offering interesting technological and scientific insights related with the specific business topics overviewed by the UCs partners.

As already mentioned, SERRANO project intends to reuse relevant assets that project partners want to add in to form a platform of components and to leverage potential outcome from different research projects that are relevant to the project goals. In Section 7 the two resources were described by outlining several important aspects relevant for each of them.

Finally, based on this analysis, a comprehensive bibliography resulted, consisting of existent technologies, standards, scientific publications or other scientific and technical resources relevant for the SERRANO project.

This document will be further used as a guideline, for the technical activities in WP3-5 regarding the SERRANO platform and for WP6 in what regards the development and integration of the UCs. This guideline will ensure that the outcomes from WP3-6 are relevant, beyond the current state-of-the-art, from the technological and scientific point of view.

9 References

[1] C. Fisher, "Cloud versus On-Premise Computing," *American Journal of Industrial and Business Management,* vol. 8, pp. 1991-2006, 2018.

[2] M. Sipos and e. al, "Distributed cloud storage using network coding," *IEEE 11th Consumer Communicationsand Networking Conference (CCNC),* pp. 127-132, 2014.

[3] P. F. Oliveira and a. et, "Coding for Trusted Storage in Untrusted Networks," *IEEE Transactions on Information Forensics and Security,* vol. 7, no. 6, pp. 1890-1899, 2012.

[4] M. Sipos and e. al, "Adaptive Network Coded Clouds: High Speed Downloads and Cost-Effective Version Control," *IEEE Transactions on Cloud Computing,* vol. 7, no. 1, pp. 19-33, 2019.

[5] P. Kokkinos and e. al, "Efficient data consolidation in grid networks and performance analysis," *Future Generation Computer Systems,* vol. 27, pp. 182-194, 2011.

[6] Ho and e. al, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory,* vol. 52, no. 10, pp. 4413-4430, 2006.

[7] A. Dimakis and e. al, "Network coding for distributed storage systems," *IEEE Int. Conf. on Computer Comm.(INFOCOM),* pp. 2000-2008, 200.

[8] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics,* vol. 8, no. 2, pp. 300-304, 1960.

[9] F. Fitzek and e. al, "Implementation and performance evaluation of distributed cloud storage solutions using random linear network coding," *IEEE International Conference on Communications Workshops (ICC),* pp. 249-254, 2014.

[10] IBM, "Containerization," 2019. [Online]. Available: https://www.ibm.com/cloud/learn/containerization.

[11] Y. Kim and M.-H. Oh, "Performance Evaluation of Fabric-Attached Memory Pool for AI Applications," *International Conference on Electronics, Information, and Communication (ICEIC),* 2021.

[12] G. Zhu, J. Han, S. Lee and Y. Son, "An Empirical Evaluation of NVM-aware File Systems on Intel Optane DC Persistent Memory Modules," *International Conference on Information Networking (ICOIN),* 2021.

[13] M. Asiatici, N. George, K. Vipin, S. A. Fahmy and P. Ienne, "Virtualized Execution Runtime for FPGA Accelerators in the Cloud," *IEEE Access,* vol. 5, pp. 1900-1910, 2017.

[14] A. Li and e. al., "Evaluating Modern GPU Interconnect: PCIe, NVLink, NV-SLI, NVSwitch and GPUDirect," *IEEE Transactions on Parallel and Distributed Systems,* vol. 31, no. 1, pp. 94-110, 2020.

[15] C. Kachris, B. Falsafi and D. Soudris, Hardware Accelerators in Data Centers (1st. ed.), Springer Publishing Company, Incorporated, 2018.

[16] J. Sheng, C. Yang, A. Sanaullah, M. Papamichael, A. Caulfield and M. C. Herbordt, "HPC on FPGA clouds: 3D FFTs and implications for molecular dynamics," *27th International Conference on Field Programmable Logic and Applications (FPL),* pp. 1-4, 2107.

[17] C. Luo, Y. Fei, L. Zhang, A. A. Ding, P. Luo, S. Mukherjee and D. Kaeli, "Power Analysis Attack of an AES GPU Implementation," *Journal of Hardware and Systems Security,* vol. 2, 2018.

[18] T. Geng and e. al., "FPDeep: Acceleration and Load Balancing of CNN Training on FPGA Clusters," *IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM),* pp. 81-84, 2018.

[19] H. Sharma and e. al., "Bit Fusion: Bit-Level Dynamically Composable Architecture for Accelerating Deep Neural Network," *ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA),* pp. 764-775, 2018.

[20] M. Wess, P. D. S. Manoj and A. Jantsch, "Neural network based ECG anomaly detection on FPGA and trade-off analysis," *IEEE International Symposium on Circuits and Systems (ISCAS),* pp. 1-4, 2017.

[21] X. Xu, Y. Ding, S. Hu, M. Niemier, J. Cong, Y. Hu and Y. Shi, "Scaling for edge inference of deep neural networks," *Nature Electronics,* vol. 1.

[22] K. Guo and e. al., "Angel-Eye: A Complete Design Flow for Mapping CNN Onto Embedded FPGA," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 37, no. 1, pp. 35-47, 2018.

[23] M. Naphade and e. al., "The NVIDIA AI City Challenge," *IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation,* pp. 1-6, 2017.

[24] V. K. Chippa and e. al., "Analysis and characterization of inherent application resilience for approximate computing," *Proceedings of the 50th Annual Design Automation Conference,* 2013.

[25] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys (CSUR),* vol. 48, no. 4, pp. 1-33, 2016.

[26] A. Rahimi and e. al., "A variability-aware openmp environment for efficient execution of accuracy-configurable computation on shared-fpu processor clusters," *2013 International Conference on Hardware/Software Codesign and System Synthesis (CODES + ISSS),* 2013.

[27] R. Ragavan and e. al., "Pushing the limits of voltage over-scaling for error-resilient applications," *Design, Automation & Test in Europe Conference & Exhibition (DATE),* 2017.

[28] J. San Miguel, M. Badr and N. E. Jerger, "Load value approximation," *47th Annual IEEE/ACM International Symposium on Microarchitecture,* 2014.

[29] M. Sutherland, J. S. Miguel and N. E. Jerger, "Texture cache approximation on GPUs," *Workshop on Approximate Computing Across the Stack,* 2015.

[30] S. Misailovic, D. Kim and M. Rinard, "Parallelizing sequential programs with statistical accuracy tests," *ACM Transactions on Embedded Computing Systems (TECS),* vol. 12, no. 2, pp. 1-26, 2013.

[31] E. A. Deiana and e. al., "Unconventional parallelization of nondeterministic applications," *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems,* 2018.

[32] J. H. Ko and e. al., "Precision scaling of neural networks for efficient audio processing," *arXiv preprint,* 2017.

[33] A. Finnerty and H. Ratigner, "Reduce power and cost by converting from floating point to fixed point," *WP491 (v1.0),* 2017.

[34] K. Koliogeorgi and e. al, "Optimizing SVM Classifier Through Approximate and High Level Synthesis Techniques," *8th International Conference on Modern Circuits and Systems Technologies (MOCAST),* 2019.

[35] S. Sidiroglou-Douskos and e. al., "Managing performance vs. accuracy trade-offs with loop perforation," *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering,* 2011.

[36] G. Tziantzioulis, N. Hardavellas and S. Campanoni, "Temporal approximate function memoization," *IEEE Micro,* vol. 38, no. 4, pp. 60-70, 2018.

[37] S. Sinha and W. Zhang, "Low-power FPGA design using memoization-based approximate computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 24, no. 8, pp. 2665-2678, 2016.

[38] H. Saadat, H. Bokhari and S. Parameswaran, "Minimally biased multipliers for approximate integer and floating-point multiplication," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 37, no. 11, pp. 2623-2635, 2018.

[39] H. Saadat and e. al., "REALM: Reduced-error approximate log-based integer multiplier," *Design, Automation & Test in Europe Conference & Exhibition (DATE),* 2020.

[40] S. Sarkar and e. al., "An Incremental Pruning Strategy for Fast Training of CNN Models," *International Conference on Computational Performance Evaluation (ComPE),* 2020.

[41] S. Venkataramani and e. al., "AxNN: Energy-efficient neuromorphic systems using approximate computing," *EEE/ACM International Symposium on Low Power Electronics and Design (ISLPED),* 2014.

[42] E. Denton and e. al., "Exploiting linear structure within convolutional networks for efficient evaluation," *arXiv preprint,* 2014.

[43] M. Mathieu, M. Henaff and Y. LeCun, "Fast training of convolutional networks through ffts.," *arXiv preprint,* 2013.

[44] S. Han, H. Mao and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint,* 2015.

[45] S. Han and e. al., "EIE: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Computer Architecture News,* vol. 44, no. 3, pp. 243-254, 2016.

[46] P. Gysel, "Ristretto: Hardware-oriented approximation of convolutional neural networks," *arXiv preprint,* 2016.

[47] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick and D. Nikolopoulos, "Challenges and opportunities in edge computing," *2016 IEEE International Conference on Smart Cloud,* pp. 20-26, 2016.

[48] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, "Fog computing and its role in the internet of things," *Proceedings of the 1st Edition of the MCC Workshop on Mobile Cloud Computing,* pp. 13-16, 2012.

[49] E. T. S. I. (ETSI), "Mobile-Edge Computing Initiative," 2016. [Online]. Available: http://www.etsi.org/technologies-clusters/technologies/mobile-edge-computing..

[50] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra and P. Bahl, "MAUI: Making smartphones last longer with code offload," *In Proceedings of the 8th International Conference on Mobile Systems Applications, and Services (MobiSys'10),* pp. 49-62, 2010.

[51] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik and A. Patti, "CloneCloud: Elastic execution between mobile device and cloud," *In Proceedings of the 6th Conference on Computer Systems (EuroSys'11),* pp. 301-314, 2011.

[52] S. Kosta, A. Aucinas, P. Hui, R. Mortier and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," *In Proceedings of the 2012 IEEE International Conference on Computer Communications (INFOCOM'12),* pp. 945-953, 2011.

[53] M. Satyanarayanan, P. Bahl, R. Caceres and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Perv. Comput.,* pp. 14-23, 2009.

[54] T. Verbelen, P. Simoens, F. D. Turck and B. Dhoedt, "Cloudlets: Bringing the cloud to the mobile user," *Proceedings of the 3rd ACM Workshop on Mobile Cloud Computing and Services,* pp. 29-36, 2012.

[55] Q. Xia, W. Liang and W. Xu, "Throughput maximization for online request admissions in mobile cloudlets," *Proceedings of the 38th Annual IEEE Conference on Local Computer Networks.,* pp. 589-596, 2013.

[56] F. Hao, M. Kodialam, T. V. Lakshman and S. Mukherjee, "Online allocation of virtual machines in a dis- tributed cloud," *IEEE/ACM Transactions on Networking,* vol. 25, pp. 238-249, 2017.

[57] S. T. Maguluri, R. Srikant and L. Ying, "Stochastic models of load balancing and scheduling in cloud computing clusters," *2012 Proceedings IEEE International Conference on Computer Communications ,* pp. 702-710, 2012.

[58] L. Chen, H. Shen and K. Sapra, "RIAL: Resource intensity aware load balancing in clouds," *Proceedings of the IEEE Conference on Computer Communications,* pp. 1294-1302, 2014.

[59] K. Mochizuki, H. Yamazaki and A. Misawa, "Bandwidth guaranteed method to relocate virtual machines for edge cloud architecture," *Proceedings of the 2013 15th Asia-Pacific Network Operations and Management Symposium,* pp. 1-6, 2013.

[60] A. C. I. Malossi and . al., "The transprecision computing paradigm: Concept, design, and applications," *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE),* pp. 1105-1110, 2018.

[61] K. Asanovic and e. al., "The landscape of parallel computing research: A view from Berkeley," *EECS Department, University of California, Berkeley,* 2006.

[62] E. L. Kaltofen, "The "Seven Dwarfs" of Symbolic Computation," *Numerical and Symbolic Scientific Computing, ser. Texts & Monographs in Symbolic Computation,* pp. 95-104, 2012.

[63] T. Yu and H. Zhu, "Hyper-Parameter Optimization: A Review of Algorithms and Applications," *arXiv, https://arxiv.org/abs/2003.05689,* 2020.

[64] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res,* pp. 281-305, 2013.

[65] J. Bergstra, R. Bardenet, Y. Bengio and B. Kégl., "Algorithms for hyper-parameter optimization," *In Proceedings of the 24th International Conference on Neural Information Processing Systems (NIPS'11),* pp. 2546-2554, 2011.

[66] A. Quitadadmo, J. Johnson and X. Shi., "Bayesian Hyperparameter Optimization for Machine Learning Based eQTL Analysis," *In Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology,and Health Informatics (ACM-BCB '17),* pp. 98-106, 2017.

[67] J. Snoek, H. Larochelle and R. Adams, "Practical Bayesian optimization of machine learning algorithms," *n Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'12),* pp. 2951-2959, 2012.

[68] L. Tani, D. Rand, C. Veelken and e. al., "Evolutionary algorithms for hyperparameter optimization in machine learning for application in high energy physics," *Eur. Phys. J.,* 2021.

[69] J. Lee and et.al, "TOD: Transprecise Object Detection to Maximise Real-Time Accuracy on the Edge," *5th IEEE International Conference on Fog and Edge Computing (ICFEC),* 2021.

[70] G. Ananthanarayanan, V. Bahl, L. Cox, A. Crown, S. Nogbahi and Y. Shu, "Video analytics - killer app for edge computing," *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services,* 2019.

[71] J. Jiang, G. Ananthanarayanan, P. Bodik, S. Sen and I. Stoica, "Chameleon: Scalable adaptation of video analytics," *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication,* pp. 253-266, 2018.

[72] V. S. Marco, B. Taylor, Z. Wang and Y. Elkhatib, "Optimizing deep learning inference on embedded systems through adaptive model selection," *ACM Trans. Embed. Comput. Syst.,* 2020.

[73] B. Lu, J. Yang and S. Ren, "Poster: Scaling Up Deep Neural Network optimization for Edge Inference," *IEEE/ACM Symposium on Edge Computing (SEC),* 2020.

[74] S. Han, H. Mao and W. J. Dally, "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding," 2015.

[75] R. Zhao, Y. Hu, J. Dotzel, C. D. Sa and Z. Zhang, "Improving Neural Network Quantization without Retraining using Outlier Channel Splitting," *ArXiv abs/1901.09504,* 2019.

[76] Z. T. and e. al., "A Systematic DNN Weight Pruning Framework Using Alternating Direction Method of Multipliers. Computer Vision," *Lecture Notes in Computer Science - ECCV 2018,* 2018.

[77] M. Zhu and S. Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," *CoRR abs/1710.01878,* 2017.

[78] G. Iuhasz, I. Drăgan and D. Petcu, "A Scalable Platform for Monitoring Data Intensive Applications," *J Grid Computing,* vol. 17, pp. 503-528, 2019.

[79] G. Iuhasz and D. Petcu, "Perspectives on Anomaly and Event Detection in Exascale Systems," *2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity),* 2019.

[80] K. Stanley, J. Clune, J. Lehman and e. al., "Designing neural networks through neuroevolution," *Nat Mach Intell 1,* 2019.

[81] G. Cormode and S. Muthukrishnan, "An improved data stream summary: The count-min sketch and its applications," *J. Algorithms,* 2005.

[82] C. Estan and G. Varghese, "New directions in traffic measurement and accounting," *ACM SIGCOMM,* 2002.

[83] Y. Zhang, S. Singh, S. Sen, N. Dufield and C. Lund, "Online identification of hierarchical heavy hitters: Algorithms, evaluation, and applications," *IMC,* 2004.

[84] M. Yu, L. Jose and R. Miao, "Software defined traffic measurement with OpenSketch," *NSDI,* 2013.

[85] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar and V. Braverman, "One Sketch to Rule Them All: Rethinking Network Flow Monitoring with UnivMon," *SIGCOMM,* 2016.

[86] S. Donovan and N. Feamster, "Intentional Network Monitoring: Finding the Needle without Capturing the Haystack," *HotNets-XIII,* 2014.

[87] V. Jeyakumar, M. Alizadeh, Y. Geng, C. Kim and D. Mazières, "Millions of little minions: Using packetsfor low latency network programming and visibility," *SIGCOMM,* 2014.

[88] A. Gupta, R. Harrison, M. Canini, N. Feamster, J. Rexford and W. Willinger, "Sonata: query-driven streaming network telemetry," *SIGCOMM,* 2018.

[89] M. Yu, "Network telemetry: towards a top-down approach," *SIGCOMM Comput. Commun. Rev. 49,* 2019.

[90] L. Tan, W. Su, W. Zhang, J. Lv, Z. Zhang, J. Miao, X. Liu and N. Li, "In-band Network Telemetry: A Survey," *Computer Networks,* vol. 186, no. 107763, 2021.

[91] A. Yaar, A. Perrig and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," *Proceedings of the 2003 Symposium on Security and Privacy,* 2003.

[92] G. Fioccola, A. Capello, M. Cociglio, L. Castaldelli, M. Chen, L. Zheng, G. Mirsky and T. Mizrahi, "Alternate-marking method for passive and hybrid performance monitoring," *RFC 8321,* 2018.

[93] A. Putina, S. Barth, A. Bifet, D. Pletcher, C. Precup, P. Nivaggioli and D. Rossi, "Unsupervised real-time detection of BGP anomalies leveraging high-rate and fine-grained telemetry data," *INFOCOM,* 2018.

[94] J. Cordova-Garcia, "Sparse Control and Data plane Telemetry features for BGP anomaly detection," *INFOCOM,* 2019.

[95] J. H. Corrêa, P. M. Ciarelli, M. R. Ribeiro and R. S. Villaça, "ML-Based DDoS Detection and Identification Using Native Cloud Telemetry Macroscopic Monitoring," *Journal of Network and Systems Management,* vol. 29, no. 2, 2019.

[96] S. Jadon, J. K. Milczek and A. Patnakar, "Challenges and approaches to time-series forecasting in data center telemetry: A Survey," *arXiv preprint ,* vol. arXiv:2101.04224.

[97] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit and L. Wobker, "In-band network telemetry via programmable dataplanes," *Proceedings SIGCOMM,* 2015.

[98] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, D. Maltz, L. Yuan, M. Zhang, B. Zhao and H. Zheng, "Packet-level telemetry in large datacenter networks," *SIGCOMM Comput. Commun. Rev. 45,* no. 4, 2015.

[99] C. Guo, L. Yuan, D. Xiang, Y. Dang, R. Huang, D. Maltz, Z. Liu, V. Wang, B. Pang, H. Chen and e. al., "Pingmesh: A large-scale system for data center network latency measurement and analysis," *Proceedings ACM Conference on Special Interest Group on Data Communication,* 2015.

[100] G. Fioccola, A. Capello, M. Cociglio, L. Castaldelli, M. Chen, L. Zheng, G. Mirsky and T. Mizrahi, "RFC 8321: Alternate-marking method for passive and hybrid performance monitoring," *2018.*

[101] A. Riesenberg, Y. Kirzon, M. Bunin, E. Galili, G. Navon and T. Mizrahi, "Time-multiplexed parsing in marking-based network telemetry," *Proceedings of the 12th ACM International Conference on Systems and Storage,* 2019.

[102] N. Kagami, R. da Costa Filho and L. Gaspary, "CAPEST: Offloading network capacity and available bandwidth estimation to programmable data planes," *IEEE Trans. Netw. Serv. Manag.,* vol. 17, no. 2, 2020.

[103] Y. Geng, S. Liu, Z. Yin, A. Naik, B. Prabhakar, M. Rosenblum and A. Vahdat, "SIMON: A simple and scalable method for sensing, inference and measurement in data center networks," *Proceedings of the 16th USENIX Symposium ,* 2019.

[104] W. Kellerer, P. Kalmbach, A. Blenk, A. Basta, M. Reisslein and S. Schmid, "Adaptable and Data-Driven Softwarized Networks: Review, Opportunities, and Challenges," *Proceedings of the IEEE,* vol. 107, no. 4, pp. 711-731, 2019.

[105] H. Yao, T. Mai, X. Xu, P. Zhang, M. Li and Y. Liu, "NetworkAI: An intelligent network architecture for self-learning control strategies in software defined networks," *IEEE Internet of Things Journal,* vol. 5, no. 6, pp. 4319-4327, 2018.

[106] Y. Hu, Z. Li, J. Lan, J. Wu and L. Yao, "EARS: Intelligence-driven experiential network architecture for automatic routing in software-defined networking," *China Communications,* vol. 17, no. 2, pp. 149-162, 2020.

[107] T. Hagemann and K. Katsarou, "A Systematic Review on Anomaly Detection for Cloud Computing Environments," *AICCC,* 2020.

[108] C. Jia and e. al., "Rapid Detection and Localization of Gray Failures in Data Centers via In-band Network Telemetry," *IEEE/IFIP Network Operations and Management Symposium,* 2020.

[109] J. Cordova-Garcia, "Sparse Control and Data plane Telemetry features for BGP anomaly detection," *IEEE INFOCOM,* 2019.

[110] A. Putina and e. al., "Unsupervised real-time detection of BGP anomalies leveraging high-rate and fine-grained telemetry data," *IEEE INFOCOM ,* 2018.

[111] A. Putina and e. al., "Telemetry-based stream-learning of BGP anomalies," *Big-DAMA,* 2018.

[112] J. Corrêa and e. al., "ML-Based DDoS Detection and Identification Using Native Cloud Telemetry Macroscopic Monitoring," *J Netw Syst Manage,* vol. 29, no. 13, 2021.

[113] Z. Chkirbene, A. Erbad, R. Hamila, A. Gouissem, A. Mohamed and M. Hamdi, "Machine Learning Based Cloud Computing Anomalies Detection," *IEEE Network,* vol. 34, no. 6, pp. 178-183, 2020.

[114] M. Rabbani and e. al, "A hybrid machine learning approach for malicious behaviour detection and recognition in cloud computing," *Journal of Network and Computer Applications,* vol. 152, 2020.

[115] B. Gill and D. Smith, "The Edge Completes the Cloud," *A Gartner Trend Insight Report,* 2018.

[116] D. Puthal and e. al., "Secure authentication and load balancing of distributed edge datacenters," *Journal of Parallel and Distributed Computing,* vol. 124, pp. 60-69, 2019.

[117] M. Aazam and E. Huh, "Dynamic resource provisioning through Fog micro datacenter," *IEEE International Conference on Pervasive Computing and Communication Workshops,* pp. 105-110, 2015.

[118] M. Abdolshah, "A review of resource-constrained project scheduling problems (RCPSP) approaches and solutions," *International Transaction Journal of Engineering, Management, & Applied Sciences & Technologies,* vol. 5, no. 4, pp. 253-286, 2014.

[119] R. Klein, "Scheduling of resource-constrained projects," *Springer Science & Business Media,* vol. 10, 2012.

[120] A. A. B. Pritsker, L. J. Waiters and P. M. Wolfe, "Multiproject scheduling with limited resources: A zero-one programming approach," *Management science,* vol. 16, no. 1, 1969.

[121] S. E. Elmaghraby, "Resource allocation via dynamic programming in activity networks," *European Journal of Operational Research,* vol. 64, no. 2, pp. 199-215, 1993.

[122] P. Brucker, S. Knust, A. Schoo and O. Thiele, "A branch and bound algorithm for the resource-constrained project scheduling problem," *European journal of operational research,* vol. 107, no. 2, pp. 272-288, 1998.

[123] D. Nasonov, N. Butakov, M. Balakhontseva, K. Knyazkov and A. V. Boukhanovsky, "Hybrid evolutionary workflow scheduling algorithm for dynamic heterogeneous distributed computational environment," *International Joint Conference SOCO'14-CISIS'14-ICEUTE'14,* pp. 83-92, 2014.

[124] C. A. C. Coello, G. T. Pulido and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on evolutionary computation,* vol. 8, no. 3, pp. 256-279, 2004.

[125] K. Deb, A. Pratap, S. Agarwal and T. A. M. T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation,* vol. 6, no. 2, pp. 182-197, 2002.

[126] M. Clerc, "Discrete particle swarm optimization, illustrated by the traveling salesman problem," *New optimization techniques in engineering,* pp. 219-239, 2004.

[127] J. Kennedy, "Bare bones particle swarms," *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03,* pp. 80-87, 2003.

[128] A. Ghodratnama, F. Jolai and R. Tavakkoli-Moghaddam, "Solving a new multi-objective multi-route flexible flow line problem by multi-objective particle swarm optimization and NSGA-II," *Journal of Manufacturing Systems,* vol. 36, pp. 189-202, 2015.

[129] E. Ozcan and C. K. Mohan, "Analysis of a simple particle swarm optimization system," *Intelligent engineering systems through artificial neural networks,* vol. 8, pp. 253-258, 1998.

[130] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu and M. Ventresca, "Enhancing particle swarm optimization using generalized opposition-based learning," *Information Sciences,* vol. 181, no. 20, pp. 4699-4714, 2011.

[131] W. Du and S. Ding, "A survey on multi-agent deep reinforcement learning: from the perspective of challenges and applications," *Artificial Intelligence Review,* pp. 1-24, 2020.

[132] H. Mao, W. Liu, J. Hao, J. Luo, D. Li, Z. Zhang and Z. Xiao, "Neighborhood cognition consistent multi-agent reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 34, no. 5, pp. 7219-7226, 2020.

[133] W. Wang, T. Yang, Y. Liu, J. Hao, X. Hao, Y. Hu and Y. Gao, "From few to more: Large-scale dynamic multiagent curriculum learning," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 34, no. 5, pp. 7293-7300, 2020.

[134] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," *Proceedings of the 10th international conference on machine learning,* pp. 330-337, 1993.

[135] Y. Duan, X. Chen, R. Houthooft, J. Schulman and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control.," *International conference on machine learning,* pp. 1329-1338, 2016.

[136] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint,* vol. arXiv:1312.5602, 2013.

[137] H. Van Hasselt, A. Guez and D. Silver, "Deep reinforcement learning with double q-learning," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 30, no. 1, 2016.

[138] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot and N. Freitas, "Dueling network architectures for deep reinforcement learning," *International conference on machine learning,* pp. 1995-2003, 2016.

[139] Y. Wang, H. Liu, W. Zheng, Y. Xia, Y. Li, P. Chen and H. Xie, "Multi-objective workflow scheduling with deep-Q-network-based multi-agent reinforcement learning," *IEEE Access,* vol. 7, pp. 39974-39982, 2019.

[140] T. Schaul, J. Quan, I. Antonoglou and D. Silver, "Prioritized experience replay," *arXiv preprint,* vol. arXiv:1511.05952, 2015.

[141] A. S. Lakshminarayanan, S. Sharma and B. Ravindran, "Dynamic frame skip deep q network," *arXiv preprint,* vol. arXiv:1605.05365, 2016.

[142] V. François-Lavet, R. Fonteneau and D. Ernst, "How to discount deep reinforcement learning: Towards new dynamic strategies," *arXiv preprint,* vol. arXiv:1512.02011, 2015.

[143] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 32, no. 1, 2018.

[144] S. Shalev-Shwartz, S. Shammah and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint ,* vol. arXiv:1610.03295, 2016.

[145] J. Jin, C. Song, H. Li, K. Gai, J. Wang and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," *Proceedings of the 27th ACM International Conference on Information and Knowledge Management,* pp. 2193-2201, 2018.

[146] L. Xi, J. Chen, Y. Huang, Y. Xu, L. Liu, Y. Zhou and Y. Li, "Smart generation control based on multi-agent reinforcement learning with the idea of the time tunnel," *Energy,* vol. 153, pp. 977-987, 2018.

[147] J. Perolat, J. Z. Leibo, V. Zambaldi, C. Beattie, K. Tuyls and T. Graepel, "A multi-agent reinforcement learning model of common-pool resource appropriation," *arXiv preprint ,* vol. arXiv:1707.06600, 2017.

[148] P. Kofinas, A. I. Dounis and G. A. Vouros, "Fuzzy Q-Learning for multi-agent decentralized energy management in microgrids," *Applied Energy,* vol. 219, pp. 53-67, 2018.

[149] D. B. Noureddine, A. Gharbi and S. B. Ahmed, "Multi-agent Deep Reinforcement Learning for Task Allocation in Dynamic Environment," *ICSOFT,* pp. 17-26, 2017.

[150] W. Chen, K. Zhou and C. Chen, "Real-time bus holding control on a transit corridor based on multi-agent reinforcement learning," *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC),* pp. 100-106, 2016.

[151] D. A. Vidhate and P. Kulkarni, "Cooperative multi-agent reinforcement learning models (CMRLM) for intelligent traffic control," *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM) ,* pp. 325-331, 2017.

[152] J. A. Calvo and I. Dusparic, "Heterogeneous Multi-Agent Deep Reinforcement Learning for Traffic Lights Control," *AICS,* pp. 2-13, 2018.

[153] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering,* vol. 22, no. 10, pp. 1345-1359, 2009.

[154] B. Settles, "Active learning literature survey," 2009.

[155] Q. Yang, Y. Liu, T. Chen and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST),* vol. 10, no. 2, pp. 1-19, 2019.

[156] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski and M. Zaharia, "A View of Cloud Computing," *ACM Communication,* vol. 53, no. 4, 2010.

[157] Y. Mansouri, A. N. Toosi and R. Buyya, "Data storage management in cloud environments: Taxonomy, survey, and future directions. ," *ACM Computing Surveys (CSUR),* vol. 50, no. 6, pp. 1-51, 2017.

[158] S. Aldossary and W. Allen, "Data security, privacy, availability and integrity in cloud computing: issues and current solutions," *International Journal of Advanced Computer Science and Applications,* vol. 7, no. 4, pp. 485-498, 2016.

[159] H. Abu-Libdeh, L. Princehouse and H. Weatherspoon, "RACS: a case for cloud storage diversity," *Proceedings of the 1st ACM symposium on Cloud computing,* pp. 229-240, 2010.

[160] T. G. Papaioannou, N. Bonvin and K. Aberer, "Scalia: An adaptive scheme for efficient multi-cloud storage," *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis,* Vols. 1-10, 2012.

[161] Y. Mansouri, A. N. Toosi and R. Buyya, "Brokering algorithms for optimizing the availability and cost of cloud storage services," *IEEE 5th International Conference on Cloud Computing Technology and Science,* vol. 1, pp. 581-589, 2013.

[162] Y. Mansouri and R. Buyya, "To move or not to move: Cost optimization in a dual cloud-based storage architecture," *Journal of Network and Computer Applications,* vol. 75, pp. 223-235, 2016.

[163] M. Hadji, "Scalable and cost-efficient algorithms for reliable and distributed cloud storage," *International Conference on Cloud Computing and Services Science,* pp. 15-37, 2015.

[164] Y. Ma, T. Nandagopal, K. P. Puttaswamy and S. Banerjee, "An ensemble of replication and erasure codes for cloud file systems," *Proceedings IEEE INFOCOM,* pp. 1276-1284, 2013.

[165] Q. Zhang, S. Li, Z. Li, Y. Xing, Z. Yang and Y. Dai, "CHARM: A cost-efficient multi-cloud data hosting scheme with high availability," *IEEE Transactions on Cloud computing,* vol. 3, no. 3, pp. 372-386, 2015.

[166] Y. Mansouri, A. N. Toosi and R. Buyya, "Cost optimization for dynamic replication and migration of data in cloud data centers," *IEEE Transactions on Cloud Computing,* vol. 7, no. 3, pp. 705-718, 2017.

[167] Z. Wu, M. Butkiewicz, D. Perkins, E. Katz-Bassett and H. V. Madhyastha, "Spanstore: Cost-effective geo-replicated storage spanning multiple cloud services," *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles,* pp. 292-308, 2013.

[168] G. Liu, H. Shen and H. Wang, "An economical and SLO-guaranteed cloud storage service across multiple cloud service providers," *IEEE Transactions on Parallel and Distributed Systems,* vol. 28, no. 9, pp. 2440-2453, 2017.

[169] Q. Wei, B. Veeravalli, B. Gong, L. Zeng and D. Feng, "CDRM: A cost-effective dynamic replication management scheme for cloud storage cluster," *IEEE international conference on cluster computing,* pp. 188-196, 2010.

[170] A. Bessani, M. Correia, B. Quaresma, F. André and P. Sousa, "DepSky: dependable and secure storage in a cloud-of-clouds," *Acm transactions on storage (tos),* vol. 9, no. 4, pp. 1-33, 2013.

[171] S. Mu, K. Chen, P. Gao, F. Ye, Y. Wu and W. Zheng, "µlibcloud: Providing high available and uniform accessing to multiple cloud storages," *ACM/IEEE 13th International Conference on Grid Computing,* pp. 201-208, 2012.

[172] P. Wang, C. Zhao and Z. Zhang, "An ant colony algorithm-based approach for cost-effective data hosting with high availability in multi-cloud environments," *IEEE 15th International Conference on Networking, Sensing and Control (ICNSC),* pp. 1-6, 2018.

[173] Y. Singh, F. Kandah and W. Zhang, "A secured cost-effective multi-cloud storage in cloud computing," *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS),* pp. 619-624, 2011.

[174] M. Su, L. Zhang, Y. Wu, K. Chen and K. Li, "Systematic data placement optimization in multi-cloud storage for complex requirements," *IEEE Transactions on Computers,* vol. 65, no. 6, pp. 1964-1977, 2015.

[175] P. Wang, C. Zhao, W. Liu, Z. Chen and Z. Zhang, "Optimizing Data Placement for Cost Effective and High Available Multi-Cloud Storage," *Computing & Informatics,* vol. 39, no. 1, 2020.

[176] A. Sharov, A. Shraer, A. Merchant and M. Stokely, "Take me to your leader! online optimization of distributed storage configurations," 2015.

[177] R. Nishtala, H. Fugal, S. Grimm, M. Kwiatkowski, H. Lee, H. C. Li and V. Venkataramani, "Scaling memcache at facebook," *In 10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13),* pp. 385-398, 2013.

[178] Y. Wu, C. Wu, B. Li, L. Zhang, Z. Li and F. C. Lau, "Scaling social media applications into geo-distributed clouds," *IEEE/ACM Transactions On Networking,* vol. 23, no. 3, pp. 689-702, 2014.

[179] L. Suresh, M. Canini, S. Schmid and A. Feldmann, "C3: Cutting tail latency in cloud data stores via adaptive replica selection," *In 12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15),* pp. 513-527, 2015.

[180] D2.2, "SERRANO use cases, platform requirements and KPIs analysis," 2021.

[181] EXCESS, *The EU FP7 project EXCESS (Execution Models for Energy-Efficient Computing Sytems),* 2013-2016.

[182] R. S. T. I. D. M. J. S. R. G. Daniel Hackenberg, "An Energy Efficiency Feature Survey of the Intel Haswell Processor," *IEEE International Parallel and Distributed Processing Symposium Workshop,* 2015.

[183] D. Khabi, "Energy efficiency of processors in high performance computing applications of engineering science," HLRS, Stuttgart, 2018.

[184] J. E. D. F. Johannes Hofmann, "Execution-Cache-Memory Performance Model: Introduction and Validation," 2015.

[185] D. H. A. T. M. G. Pavel Skvortsov, "Monitoring in the Clouds: Comparison ofECO2Clouds and EXCESS Monitoring Approach," 2016.

[186] R. Schneider, "Analyse kontinuumsmechanischer, anisotroper Materialparameter mikrostrukturierter Volumina mit Hilfe direkter mechanischer Simulation," *HLRS,* 2016.

[187] A. C. o. E. i. HPC, "Performance Optimisation and Productivity," 5 2021. [Online]. Available: https://pop-coe.eu/.

[188] A. Soylu, E. Kharlamov, D. Zheleznyakov, E. Jimenez-Ruiz, M. Giese, M. G. Skjæveland and I. Horrocks, "OptiqueVQS: a visual query system over ontologies for industry," *Semantic Web,* vol. 9, no. 5, pp. 627-660, 2018.

[189] L. Otero-Cerdeira, F. J. Rodríguez-Martínez and A. Gómez-Rodríguez, "Ontology matching: A literature review," *Expert Systems with Applications,* vol. 42, no. 2, pp. 949-971, 2015.

[190] E. Chondrogiannis, V. Andronikou, E. Karanastasis and T. A. Varvarigou, "An Intelligent Ontology Alignment Tool Dealing with Complicated Mismatches," *SWAT4LS,* 2014.

[191] "Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0 OASIS Standard," [Online]. Available: http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.html.

[192] N. G. Leveson, "Intent specifications: An approach to building human-centered specifications," *IEEE Transactions on software engineering,* vol. 26, no. 1, pp. 15-35, 2000.

[193] G. Harman, "Logic, reasoning, and logical form," *Language, Mind, and Brain,* pp. 13-19, 1982.

[194] A. Rafiq and e. al., "Intent-based end-to-end network service orchestration system for multi-platforms," *Sustainability,* vol. 12, no. 7, 2020.

[195] O. N. Foundation, "CORD (Central Office Re-architected as a Datacenter) platform," [Online]. Available: https://opennetworking.org/cord/ .

[196] ETSI-hosted, "Open Source Management and Orchestration (MANO) software stack," [Online]. Available: https://osm.etsi.org/.

[197] "OpenAirInterface (OAI) Software Alliance," [Online]. Available: https://openairinterface.org/ .

[198] J. T. Kim and e. al., "IBCS: Intent-Based Cloud Services for Security Applications.," *IEEE Communications Magazine,* vol. 58, no. 4, pp. 45-51, 2020.

[199] C. M. Bishop, "Pattern recognition and machine learning," *Springer,* 2006.

[200] I. Ben-Gal, "Bayesian networks," *Encyclopedia of statistics in quality and reliability,* vol. 1, 2008.

[201] Z. Ghahramani, "An introduction to hidden Markov models and Bayesian networks," *Hidden Markov models: applications in computer vision,* pp. 9-41, 2001.

[202] J. Ali and e. al., "Random forests and decision trees," *International Journal of Computer Science Issues (IJCSI) ,* vol. 9, no. 5, 2012.

[203] W. S. Noble, "What is a support vector machine?," *Nature biotechnology ,* vol. 24, no. 12, pp. 1565-1567, 2006.

[204] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern recognition letters,* vol. 31, no. 8, pp. 651-666, 2010.

[205] E. Schubert and e. al., "DBSCAN revisited, revisited: why and how you should (still) use DBSCAN," *ACM Transactions on Database Systems (TODS),* vol. 42, no. 3, pp. 1-21, 2017.

[206] J. Han, J. Pei and Y. Yin, "Mining frequent patterns without candidate generation," *ACM sigmod record,* vol. 29, no. 2, 2000.

[207] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," *Nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[208] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," *Proceedings of ICML workshop on unsupervised and transfer learning. JMLR Workshop and Conference Proceedings,* 2012.

[209] J. Gu and e. al., "Recent advances in convolutional neural networks," *Pattern Recognition,* vol. 77, pp. 354-377, 2018.

[210] K. Rashmi and e. al., "A solution to the network challenges of data recovery in erasure-coded distributed storage systems: A study on the Facebook warehouse cluster," *Proceedings of the 5th USENIX Conference on Hot Topics in Storage and File Systems, Berkeley,* 2013.

[211] M. Silberstein and e. al., "Lazy Means Smart: Reducing Repair Bandwidth Costs in Erasure-coded Distributed Storage," *Proceedings of International Conference on Systems and Storage (SYSTOR),* vol. 15, pp. 1-7, 2014.

[212] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," https://arxiv.org/abs/1407.3561 2014, 2014.

[213] L. Sari and e. al., "FileTribe: Blockchain-based Secure File Sharing on IPFS," *25th European Wireless Conference,* 2019.

[214] D. Jung, V. Dorner, F. Glaser and S. Morana, "Robo-Advisory Digitalization and Automation of Financial Advisory," *Business & Information Systems Engineering ,* 2018.

[215] F. Fabozzi and H. Markowitz, The Theory and Practice of Investment Management: Asset Allocation, Valuation, Portfolio Construction, and Strategies, Business & Economics, 2011.

[216] A. Khodadadi, R. Tütüncü and P. Zangari, "Optimisation and quantitative investment management," *Journal of Asset Management ,* 2006.

[217] J. H. Kim, W. C. Kim and F. Fabozzi, "Recent advancements in robust optimization for investment management," *Annals of Operations Research ,* 2018.

[218] I. Leea and Y. J. Shinb, "Fintech: Ecosystem, business models, investment decisions, and challenges," *Business Horizons,* 2018.

[219] P. Maresova and B. Klimova, "Investment evaluation of cloud computing in the European business sector," *Applied Economics,* 2015.

[220] R. Ziemba and W. Ziemba, "The Renaissance Medallion Fund," *World Scientific Handbook in Financial Economics Series,* 2011.

[221] M.-Y. Day, T.-K. Cheng and J.-G. Li, "AI Robo-Advisor with Big Data Analytics for Financial Services," *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM),* 2018.

[222] D. Tokic, "BlackRock Robo-Advisor 4.0: When artificial intelligence replaces human discretion," *Artificial Intelligence, FinTech, and other Globally Strategic Issues,* 2018.

[223] T. Becker, O. Mencer, S. Weston and G. Gaydadjiev, "Maxeler Data-Flow in Computational Finance," *FPGA Based Accelerators for Financial Applications,* 2015.

[224] M. Faloon and B. Scherer, "Individualization of Robo-Advice," *The Journal of Wealth Management Summer ,* 2017.

[225] P. Oliveira and E. Hippel, "Users as service innovators: The case of banking services," 2011.

[226] R. Ahmad and S. Kamaruddin, "An overview of time-based and condition-based maintenance in industrial application," *Computeres and industrial engineering. 63(1),* pp. 135-149, 2012.

[227] T. V. Tung and B.-S. Yang, "Machine Fault Diagnosis and Prognosis: The State of The Art," *International Journal of Fluid Machinery and Systems,* p. vol2, 2009.

[228] R. Randall, "State of the art in monitoring rotating machinery - part 1," *Sound and Vibration, 38(3),* pp. 14-21, 2004.

[229] M. Esperon-Miguez, P. John and I. Jennions, "A review of intergrated vehicle health management tools for legacy platforms: challenges and oportunities," *Progress in Aerospace Sciences 56,* pp. 19-34, 2015.

[230] L. v. D. J.I.A. de Vos, "Performan cecentered Maintenance as a core policy in strategic maintenance control," *Procedia CIRP 38,* pp. 255-258, 2015.

[231] X. B. J. A. S. R. J. M. Y.Zhang, "A CPPS based on GBDT for predicting failure events in milling," *The International Journal of Advanced Manufacturing Technology,* 2019.

[232] T. C. M. M. X. Xu, “Intelligent Fault Prediction System BAsed on Internet of Things,” *Computers & Mathematics with Applications 64,* pp. 833-839, 2012.

[233] L. W. M. H. R. T. Robert X. Gao, “Big data analytics for smart factories of the future,” *CIRP Annals- Maufacturing Technology,* pp. 1-25, 2020.

[234] R. Merino, I. Bediaga, A. Iglesias and J. Munoa, “Hybrid Edge–Cloud-Based Smart System for Chatter Suppression in Train Wheel Repair,” *Applied Sciences 9,* p. 4283, 2019.

[235] T. Xi, I. M. Beninca, S. Kehne, M. Fey and C. Brecher, “Tool wear monitoring in roughing and finishing processes based,” *The International Journal of Advanced Manufacturing Technology, 113,* pp. 3543-3554, 2021.

[236] R. L. V.-H. B. Cruz Salazar LA, “Cyber-physsicla production systems architecture based o multi-agent's design patter- comparison of selected approaches mapping four agents,” *Int J Adv Manuf technology 105,* p. 4035, 2019.

[237] H. A. S. Y. B. B. J. Lee, “Industrial Big Data Analytics and Cyber Physical Systems for Future Maintanance & Service Innovation,” *Procedia CIRP, 38,* pp. 3-7, 2015.

[238] T. L. Y. Z. C. d. S. M. Xia, “Closed Loop Design Evolution of Engineering System Using Condition Monitoring Through Internet of Thins and Cloud Computing,” *Computer Networks,* 2016.

[239] X. Beudaert, J. Argandoña, J. Loc’H, I. Bediaga and J. Munoa, “Monitoring and analytics platform for machine tools,” *14th International Conference on High Speed Machining,* 2018.