Digital Object Identifier 10.1109/ACCESS.2017.DOI

The Continuum of Computing

Giovanni Jiayi Hu
[1]Department of Mathematics, University of Padua, Italy I-35121

ABSTRACT Work in Progress.

I. INTRODUCTION

The Internet has evolved significantly since its inception. From just a simple communication layer for information sharing between researchers, it has grown into an ubiquitous platform for everyday services. This transformation is driven by many changes and infrastructure trends.

First, during the past decade, the sharing of server and networking capabilities - known as cloud computing paradigm - has become a reality, giving users and companies access to virtually unlimited amounts of storage and computing power. Nowadays, there is an immense amount of seamlessly connected mobile devices, servers, and network components which offer their virtualized capabilities to their users.

Second, as mobile computing evolved, people started bringing their devices with them and accessing the Internet anywhere and anytime. Nowadays, we are in the midst of the so-called Internet of Things (IoT), where devices (things) are connected to the Internet and each other. These "things" comprise a multitude of heterogeneous devices ranging from consumer devices, such as mobile phones and wearables, to industrial sensors and actuators [23], and smart transportation, grids [21], and cities [22].

It this thus natural that as consumers, we want our Internet-capable devices (e.g., mobile phones, thermostats, electrical vehicle) to adapt seamlessly to our changing lifes and expectations, regardless of location [12].

Besides, our capacity for collecting data is expanding dramatically, and yet, our ability to timely manage, manipulate, and analyze this data to transform it into information and act upon it has not kept the pace [11]. Data sources and their volumes of generation are growing exponentially and have outpaced the Internet ability to transport this data in a reliable and timely manner to the cloud. IoT applications might require very short response time, involve private data, and produce a large quantity of data which could be a heavy load for networks. Cloud computing and the Internet infrastructure is not capable enough to support these applications and their exploding multitude.

In the vision of this thesis, the future holds us an information-rich pervasive computational continuum that seamlessly combine this data and computing power to model, manage, control and make use of virtually any realizable subsystem of interest [11]. Use cases exist in diverse application areas from managing extreme events (e.g. environmental monitoring) to optimizing everyday processes (e.g. manifacturing) and improving life quality (e.g. healthcare and smart cities).

A natural consequence of this vision is that, in edge networks, computational capabilities are gradually being introduced during the recent years. New approaches that combine distributed services close to the data sources (i.e. edge nodes) with resources in the cloud and along the data path can constitute a computing continuum and be effective in processing this data. Depending on the use case and service level requirements (SLR), user applications may require processing and storage locally, in the cloud or somewhere in the between (fog).

This trend paves the way for novel and ubiquitous services in a wide range of application domains. To cite Haller et al. [1], this thesis believes that we are close to "a world where physical objects are seamlessly integrated into the information network, and where the physical objects can become active participants in business processes. Services are available to interact with these 'smart objects' over the Internet, query their state and any information associated with them, taking into account security and privacy issues."

This thesis proposes a vision which follows naturally from the previous premises - the **continuum of computing** - a ubiquitous system where distributed resources and services on the whole computing continuum are dynamically aggregated on-demand to support different ranging services.

In the vision of this thesis, there will be pervasive service platforms anywhere the user is and a multitude of services available over the Internet. The granularity of these services will be very different, ranging from high-level business services to low-level sensor services provided by the Internet of Things.

It is fundamental that services are treated as first-class citizens, allowing services to be dynamically placed by decoupling them from their location. The continuum must also facilitate the elastic provisioning of virtualized end-to-end service delivery infrastructures, in which virtual capabilities are leased, dynamically configured and scaled as a function of user demand and service requirements. The continuum must have as a fluid, which continuously adapts its shapes to fit the surroundings, as first described in [10].

This vision goes however beyond the traditional elasticity of clouds, since in addition to computational, network and storage resources, the managed capabilities also include end-to-end network configuration and high-level service and device functionalities.

The notion of "the continuum of computing" is not completely novel and has been anticipated before. The authors of [10] envision a Fluid Internet, which "seamlessly provisions virtualized infrastructure capabilities, adapting the delivery substrate to the dynamic requirements of services and users, much like a fluid adapting to fit its surroundings". [11] also presents the notion of computing in the continuum, that is "realizing a fluid ecosystem where distributed resources and services are programmatically aggregated on-demand to support emerging data-driven application workflows". Authors of [12] also present the continuum of computing and seek to develop approaches that include the entire computing continuum as a collective whole.

Both cloud computing and IoT topics gained popularity in the last decade, and the number of papers dealing with Cloud and IoT separately has been showing an increasing trend since 2008. More recent works like [17] focus on the integration of the Cloud and IoT, but keeping the two as distinct spaces where the latter sends data and offloads computation to the former. Only the few pubblications mentioned in the previous paragraph foresee a continuum of computing encompassing the whole Internet as substrate for ubiquitous services.

The remaining parts of this paper are organized as follows. Section II discusses the present trends which lead to the continuum and presents the challenges of this vision. Section III shows the problem space addresses by this thesis and the technology baseline. Section IV contains the evaluation of the architecture presented in the preceding section and, lastly, the thesis concludes in Section V with the final remarks.

II. THE PROBLEM STATEMENT

In the past decade and now more than ever, cloud computing has been providing users with the potential to perform computing tasks utilizing resource physically distant to them. A popular definition of cloud computing [2] is "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

The successful cloud architecture can be split into four layers: datacenter (hardware), infrastructure, platform, and application [18]. Each of them can be seen as a service for the layer above and as a consumer for the layer below. This results in mainly three types of service: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS).

Conventional single provider infrastructures hosting cloud services offer undoubtedly a lot of benefits but not without challenges. A large data center's energy consumption is high just to keep it operational and like any other centralized computing model, in case of a failure the resulting issues would be adverse.

Another issue is that required data have to be transferred and stored to separate places from the source, because data centers are often geographically distant from the application users, especially when the data is generated at edge locations. However, exchange of sensitive or personal data is considered critical and private for applications.

Besides, the evolution of the Internet of Things is having a significant impact on cloud computing generally and it's stretching the limitations of the latter. The number of connected devices is increasing exponentially with estimations of dozens of billions of "things" going live in coming years [24].

As a consequence of connecting these sensors to the Internet, large volumes of data are being generated at unprecedented volumes, variety and velocity. This data is currently transferred and stored in the cloud in a centralized manner. Data transfer, especially in these volumes, is extremely expensive and retards computational performance.

It becomes thus obvious that traditional methods of data processing where all data is collected and processed in a central instance will not suffice. A more decentralized solution is required where data processing could take place before transfer and storage. The key is to reduce the number of messages and the amount of data transmitted throughout the continuum so that data should be evaluated locally or only where it makes sense.

With these premises, in the vision of this thesis, we shall apply the aforementioned cloud computing definition to the Internet of Things and the network along the data path (fog and edge). The rational is two-fold. First, it offers on-demand virtual capabilities regarding storage, memory and processing units that augment IoT devices and components with limited computation capabilities due to build factors. Note however that fog and edge nodes cannot scale "infinitely" as the cloud data centers.

Second, scaling fog and edge infrastructure is costly and time-consuming even more as in cloud infrastructure, due to additional factors such as heterogeneity, network unreliability and difficulty to predict capacity needs in advance. Underprovisioning means potentially losing business while overprovisioning means wasting money on unused infrastructure. Moreover, user demand at the edge can be very often spiky, meaning that companies would need to provision for anomalous peaks like gatherings and events, investing in significant infrastructure that sits underutilized most of the time. This also has an environmental cost when underutilized infrastructure consumes significant amounts of power.

The cloud computing model applied to the fog and edge as services is attractive since it allows businesses to decrease the required capital expenditure (CAPEX) and frees them from the need to invest in the infrastructure. Businesses can rent resources according to their needs and only paying for the usage [17]. Moreover, it allows to decrease operating costs (OPEX), as service providers do not have to provision capacities according to maximum peak load. In fact, resources are released when service demand is low.

On the other hand, the cloud can benefit from the fog and edge by extending its scope to deal with real world things in a more distributed and dynamic manner, and for delivering new services in a large number of real life scenarios. This will impact future application development, where this new flow of information gathering, processing, and transmission will generate new challenges.

A. DECENTRALIZATION

As stated before, highly distributed network is the most effective architecture for the Continuum, particularly as services becomes more complex and more bandwidth hungry. Although often referred to as a single entity, the Internet is actually composed of thousands of different networks. This means that content generated at the edge must travel over multiple networks to reach its centrally-hosted data center.

Unfortunately, inter-network data communication is neither an efficient nor reliable operation and can be adversely affected by a number of factors.

Capacity at peering points where networks exchange traffic typically lags demand, due in large part to the economic structure of the Internet [8]. The economic incentive flows in at the first mile (cloud data centers) and at the last mile (IoT), with however very little interest to invest in the middle network composed by peering points. These points become thus the cause for packet loss and increase latency.

Across the Internet, outages are happening all the time, caused by a wide variety of reasons such as cable cuts, misconfigured routers, DDoS attacks, power outages, or even earthquakes and other natural disasters. While failures vary in scope, large-scale occurrences are not uncommon [9].

When dealing with an pervasive system such as the continuum, it is therefore no longer sufficient to simply have enough server and network bandwidth resources. One must consider the throughput of the entire path from IoT devices to data centers to end users. The bottleneck is not likely to be at just the far ends of the path. It could be at a peering point as mentioned, or it could be due to the network latency between server and device.

Autonomous vehicle are a great example of these issues. One Gigabyte data will be generated by the car every second and it requires real-time processing for the vehicle to make correct decisions [13]. A network bandwidth and reliability can be easily challenged for its capability of supporting a large number of vehicles in one area. Moreover, if all the data were to be sent to the cloud for processing, the response time would be too long. In such cases, the data needs to be processed at the edge for shorter response time, more efficient processing and smaller network pressure.

Secondly, in many other cases, most of the end nodes in IoT (e.g. sensors and actuators) are energy constrained things so offloading some computing tasks to the edge could be more energy efficient. Even end end mobile phones benefit greatly from reduced energy consumption, especially for tasks like Mobile Augmented Reality [25].

For these reasons, alternate models such as fog computing and edge computing have emerged in the recent years. A fog and edge-based platform, with servers anywhere the end device is, can achieve the scale needed with each location supporting higher orders of throughput, low response times and higher energy efficiency.

However, the nearest physical layer may not always be a good option. If the task requires long computation, as in the case of big-data analysis, it will be better to offload to the more distant but capable node, or even the cloud. The latency of a request is the sum of two components: the computing latency and the transmission latency. A high computing latency can outweight the transmission efficiency. Therefore, the wish for edge computing is to determine the ideal trade-off between computing latency and transmission latency leveraring the whole continuum.

Similar attention must be dedicated to the energy issue. Battery is the most precious resource for things at the edge of the network, but the wireless communication module is usually very energy hungry [13]. For a given workload, is it not always the case that the most energy efficient solution is to offload the whole workload to the edge rather than compute locally. The key is the tradeoff between the computation energy consumption and transmission energy consumption. Disburdening to another node is preferable only if the latte overhead is smaller than computing locally.

It shall be clear by now that the vision of a computing continuum is not that one form of computing (e.g. edge computing) will supplant another (e.g. cloud computing), but that we must try harnessing the entire computing space. Some trending computing models relevant for the aforementioned decentralization are not presented.

1) Fog Computing

Fog computing [20] is a decentralized computing infrastructure which is used particularly as a complement to cloud computing. It leverages the compute resources at the edge network and brings the computational processing closer to the data source by offloading workload to edge nodes from cloud data centers. The network nodes near the edge providing these resources are called fog nodes.

Overall, any device with computing, storage and network connectivity can constitute a fog node, such as switches and routers, industrial controllers, embedded servers and video surveillance cameras [16].

It's first usually a single hop away from the edge and is an extension of edge computing. The Cloudlet [7] architecture is one of the first approaches that offer an example of fog computing with virtual machine (VM) techniques. Its computing power is much less than a conventional cloud infrastructure as cloudlets are composed of less powerful processors and are significantly smaller in size.

2) Mobile Edge Computing

Edge computing [13] is any computing and network resource that takes place only on the edge of the network. The rationale of edge computing is that computing should happen at the proximity of data sources, eliminating the costly data transfer to a remote data center. This significantly improves user Quality of Service (QoS) as, similar to fog computing, there is considerable network latency reduction and bandwidth consumption by the end users.

When the end users are mobile devices, edge computing is also referred as mobile edge computing (MEC). In contrast to the broader term edge computing, MEC focuses on co-locating computing and storage resources at base stations of cellular networks. Being co-located at base stations, computing and storage resources of MEC servers are available in close proximity to mobile users like smartphones and smart vehicles and can support mobility between cellular cells. MEC is seen as a promising approach to increase the quality of experience in cellular networks and a natural direction for the evolution to 5G networks [26].

3) Serverless Computing

Serverless Computing [27] involves building, running and providing applications and services without taking into consideration the server side. "Serverless" does not mean that there is no server usage but rather the main focus is on the application itself rather than what happens on the physical infrastructure.

Serverless Computing is synonymous with Function-as-a-Service (FaaS) and event-based programming as the execution of an application will be executed only when necessary and not all the time, thus meaning that an event can trigger the execution of a function. In particular, FaaS triggers a server only when a function is requested, executes the expected operations and then terminates. The major advantages of this model are increased scalability and infrastructure independency of the applications and lower costs.

Recently, there is AWS Lambda@Edge, that allows using serverless functions at the AWS edge location in response to CDN event to apply moderate computations. TODO

Recent works in literature have shown the advantages of the approach for edge computing. The authors of [15] show an augmented reality use case for Mobile Edge Computing, in which computation and data intensive tasks are offloaded from the devices to serverless functions at the edge, outperforming the cloud alternative up to 80% in terms of throughput and latency.

B. CHALLENGES

Having discussed the trend towards a decentralized interconnected continuum, this section presents the challenges imposed by such system. As an example, the heterogeneity of the connected devices is immense and can be seen from different perspectives, such as computing performance, storage and network requirements, communication protocols, energy consumption to name a few. Such diversity poses questions about how to deal with the virtualisation of the underlying resources and the interoperability of the nodes.

In order to achieve the continuum, this thesis identifies the following challenges as the most significant.

1) *Service orientation*: as stated in the introduction of this thesis, services must be treated as first-class citizens, allowing them to be dynamically placed and decoupling them from their location. Consumers should only be concerned with what they want to do and accomplish and providers with how that could be done and provided to the user;
2) *Orchestration*: advanced orchestration systems are required to support the resource management of heterogeneous devices and be adapted to many applications running on the edge connected devices;
3) *Virtualisation*: the system have to provision resources and provide access to heterogeneous IoT resources and hardware such as GPUs, FPGAs, etc. Besides, today's common practice in cloud assume enumerated resources and predictable computing capabilities. However, in the continuum, the capabilities and numbers of components change dramatically over time;
4) *Interoperability*: applications on a continuum should be able to seamlessly amalgamate services and infrastructure from different locations. Every nodes on the Internet should also expose an uniform interface to its resources and services;
5) *Portability and Programmability*: developers should be provided the languages and tools to program applications and services to run in such dynamic environment without having to reinvent themselves;
6) *Mobility*: efficient migration of each application and service has to be supported from platform to platform and to follow the users' movements in the network (roaming).
7) *Reliability*: services in the continuum could fail due to various reasons, notably at the edge. The establishment of reliable communication between nodes must be supported;
8) *Security and Privacy*: personal and sensitive user data are subjected to high risk while many users access Internet services from anywhere. Security and privacy matter even more when personal data may have to be stored closer to the users/devices to facilitate computing and processing on the edge or fog layer;
9) *Context awareness*: services need to be aware of different aspects of the environment they are acting in, whereas today's services are rarely context-sensitive;
10) *Energy efficiency*: obtaining energy efficiency in both data processing and transmission is an important open

issue;

11) *Storage*: TODO

1) Service orientation

This thesis visions service-oriented paradigms as necessary for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. A Service Oriented Architecture (SOA) provides "a uniform mean to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations" [6].

According to this model, the major components of a basic SOA and their possible interactions are constituted by a service provider which publishes his service interface via a service registry where a service requester/consumer can find it and subsequently may bind to the service provider [1].

The central concept of the SOA reference model is the existence of services which provide access to capabilities by well-defined interfaces to be exercised following a service contract with constraints and policies. This enables a loose coupling of services and minimization of mutual dependencies.

Services are provided by the service providers and are to be used by others, the service consumers. Services may be composed on the basis of other, existing services, thereby adhering to the principle of reuse. The are responsible only for the logic and information they encapsulate, uniformly described and publicly retrievable via service discovery mechanisms. This is in net contrast to the practice of today, where we continue to build ad-hoc programs confined in relatively small single nodes, defining individual device computing behaviours [12].

However, to enable the seamless and agile interoperation of services in the continuum, a number of challenges still exist with respect to their organization and implementation.

First of all, the lack of neutral, trustworthy and widely accepted service intermediaries in the Internet of today still prevents the establishment of a Continuum as described above. Service intermediaries are necessary for facilitating the efficient retrieval of services which match a given user need and for providing service performance through monitoring of service quality and availability. Unfortunately, so far single cloud providers such as Amazon Web Services, Google Cloud Platform or Microsoft Azure have shown no interest for inter-cooperation.

Secondly, the lack of means for composing diverse services towards higher-order services meeting actual user demands imposes additional challenges to the emergence of the Continuum of Computing. The lack of interoperation prevents services from being set up quickly and easily. Cloud providers and their respective systems adhere to different conventions with respect to interfaces and communication protocols. For instance, Google Cloud Platform services use Protocol Buffers [28], which are a Google technology for serializing structured data. For this reason, a major effort will be required to uniformly provide the service semantics and interfaces in a comprehensible way for both humans and machines.

On this latter matter, services in the Internet of today can be considered as mute and dull. In the future, services are expected to become more open and reactive to their respective environments, particularly in two respects [1]. Existing service interfaces (e.g., REST-based) are not yet designed to be interpreted and used by machines but rather by humans. This frequently prevents the systems from easily discovering and interacting with them. Methods are needed to allow to both "put a face" on services and to thus improve both user-service interaction and machine-to-machine (M2M) communication. Fortunately, recent efforts in the REST community are trying to compensate the lack of M2M communication [29].

a: Service-oriented architectures

The idea of service-orientation can be partially traced back to the object-oriented programming (OOP) literature [3]. However, the evolution of objects into services, and the relative comparisons, has to be treated carefully since the first focus on encapsulation and information is hidden in a shared-memory scenario, while the second is built on the idea of independent deployment and asynchronous communication via well-defined interfaces. It is therefore a paradigm shift, where both the paradigms share the common idea of componentization.

The last decade has seen a further shift towards the concept of service first and the natural evolution to microservices afterwards. Service oriented architectures (SOA) have been introduced to harness the complexity of distributed systems and to integrate different software applications. In SOA, a service offers functionalities to other components, accessible via message passing. Services decouple their interfaces (how other services access their functionalities) from their implementation.

This is in striking contrast to monolithic architectures, where the application is composed of a single program, typically providing a user interface and data access through a database. However, the monolithic architecture poses real challenges and difficulties when growing exponentially because of the application need to scale up.

Service Oriented Architectures (SOA) are instead built around services, which are designed to work in an orchestrated manner to modularize the system. It is more challenging to divide the application into multiple services, but it enables greater flexibility, extensibility and reusability of existing services for multiple use cases. The benefits of this model are application modularity and service reusability [4], which are all very important to a successful continuum of computing. A major disadvantage however is the complexity in orchestrating and monitoring all the services, especially when the project is complex and the components are huge.

The first generation of SOA architectures defined daunting and nebulous requirements for services (e.g., discoverability and service contracts) [3], and this hindered the adoption of the model. Microservices are the second iteration, introduced as a solution for the gaps in the SOA approach.

The microservices approach divides applications into more granular components by distributing them into small self-contained services. Each service implements and attends to separate business functions and capabilities to maintain independence from other services. They are mainly deployed in an automated manner, through a container (more on that in §II-B3) and communicating through REST APIs [5], thus making the impact of programming language insignificant. This allows microservices to be easily deployed in the cloud, offering great reusability.

In terms of the continuum, the *as-a-service* model allows consumers to be only concerned with what they want to do and accomplish, and providers with how that could be done and provided to the user. A successful interface establishment between those two actors can lead to minimal direct consumer interaction with provider's infrastructure, thus allowing full control to the provider and no cost of ownership for the consumer.

Furthermore, this results in a system where various service implementations should already exist, maybe provided in the same fashion as service marketplace (e.g. Shopify App Store [30]), and the consumer himself does not have to be an expert and develop them. Even the physical resources provided by the infrastructure should not be consumer-aware and there may be several diverse implementations to meet specific service demands. These implementations can differ in hardware type and could be characterized by different price and performance attributes.

b: Web services

TODO

2) Orchestration

Orchestration can be defined as the use of programming technology to manage the interconnections and interactions among workloads on distributed edge-cloud infrastructure [4]. This is accomplished through three main attributes of orchestration, which are closely related: service orchestration, workload orchestration, and resource orchestration.

Orchestration in the continuum will be a crucial feature for many IT organisations and DevOps adopters as a way to speed the delivery of services, simplify optimisation, and reduce costs [8]. However, the orchestration of virtualised environments is challenging due to the scale, heterogeneity, and diversity of resource types and the uncertainties of the underlying environments. The uncertainties arise from a number of factors including resource capacity demand (e.g. bandwidth and memory), failures (e.g. failure of a network link), user access pattern (e.g. number of users and location) and lifecycle activities of applications.

A key enabler to the orchestration and management of a continuum architecture is the virtualisation, further presented in the next subsection. The evolution of virtualisation has moved away from virtual machines towards more lightweight solutions such as containers. This is specifically relevant for software packaging at application and service level. Different application packages such as containers and orchestration solutions such as Docker container and Kubernetes architectures [34] have been proposed to for the edge computing [16]. Yet, there is still a need for a topology specification and a derived orchestration plan for the continuum of computing.

TODO

3) Virtualisation

TODO

Compared to VM, the OS-level virtualisation provided by containers offers resource isolation in a much lower cost. Adoption of container technique leads to a platform where the end user can deploy services and applications on edge computing platform on heterogeneous devices with minimal efforts. Containers provide resource isolation, self-contained packaging, anywhere deployment, and easiness of orchestration.

Several works like [14] combine container technology along with the serverless computing model to enable end user the need to only register events of interest and provide corresponding handler functions to the system, which automatically handle the events behind the scene.

Virtualisation for IoT devices has also another facet rarely discussed, to the best of this thesis' knowledge. Existing edge systems fail to promptly recognize the environmental context changes under which IoT nodes are subject to, which are often critical to many applications like video analysis [16]. This is because the service controller can only validate the usefulness of the data (e.g. video frame) after they are received from the devices. The latter will continue to perform poorly (or even uselessly) until the controller makes adjustments.

Virtualisation on constrained devices would allow to dynamically adapt to environmental context changes based on the application requirements by running pre-processing on the node itself. The experimental results in [16] show that virtualisation of multiple IoT cameras to serve applications and reconfigure the cameras can significantly enhance performance in terms of object detection accuracy, computational cost, and response time.

4) Interoperability

For connecting and integrating all the things into the continuum, there are and will be many different technologies. In embedded systems standards are in place or standardisation efforts are underway. For example, ZigBee, 6LoWPAN, MQTT and CoAP [31] are gaining popularity in the wireless sensor networking area, and OPC [32] is well accepted in factory automation. However, the technologies are too different to expect any standard to be able to cover them all.

For all these reasons, it is clear that we will have to deal with heterogeneity when building the edge infrastructures of the continuum. Standards are surely helpful, but will be hardly achieved. What is asked for here is interoperability.

The key is to separate the functionality from its technical implementation. Service oriented architectures are ideally suited for this, since they encapsulate functionality in services with a common interface, abstracting from the underlying hardware and protocols. TODO

Having infrastructures that allow connecting and integrating a diverse set of technologies is not just a "necessary evil", but rather a strength, since it offers two key benefits. First, it allows applying different solutions to different applications. Depending on the application requirements, the best-fitting technology can be used.

Secondly, an infrastructure where diverse technologies can easily be integrated into will be future-proof. Especially in the area of edge computing and IoT, the technical developments are not complete and we can expect that new technologies, protocols and standards will arise. An infrastructure built with technology diversity in mind will allows these to be interoperable with existing and already deployed devices and networks.

With that said, cloud platforms heterogeneity is also a non-negligible concern. Cloud services typically come with proprietary interfaces, causing resource integration to be properly customized based on the specific providers. This issue can be exacerbated when users when services in the continuum can depend on multiple providers in order to provide the necessary resources an application may require or to improve application performance and resilience [17].

Both cloud and IoT services and applications have typically been conceived as isolated vertical solutions, in which all system components are tightly coupled to the specific application context or the cloud provider. By applying the cloud service delivery models, the IoT should instead ease service delivery without any vendor lock-in.

This challenge involves thus several aspects, where solutions need to be being investigated in terms of unifying cloud providers [33], interoperable programming interfaces, and means for copying with data diversity.

TODO: data-diversity

5) Portability and Programmability

In cloud computing, users program their code and deploy them on the cloud. The cloud provider is in charge to decide where the computing is conducted in a cloud. Users have only partial knowledge of how the application runs. This is one of the benefits of cloud computing: the infrastructure is transparent to the user. Usually, the program is written in the programming language the developer is most familiar with between the supported ones (e.g. JavaScript, Java, Python, .NET) and compiled for a certain target platform, since the program only runs in the cloud.

However, in the edge computing and thus in the continuum, the nodes have diverse platforms. Mobile devices may have Android and iOS operative systems, edge nodes are likely to have different Linux distributions or versions of the same distribution. Embedded devices lacks of any operative system feature such as file system or the notion of processes. Even the different CPU architectures (x86, ARM32, ARM64) give programmers a hard time compiling for the different platforms. For all these reasons, the runtime of the nodes differ from each other, and the programmers face huge difficulties to write a service/application that may be deployed in the continuum paradigm.

To address the programmability in the context of edge computing, [13] proposes the concept of computing stream that is defined as a serial of functions applied on the data along the data propagation path. Likewise, [12] proposes to program the continuum so that new algorithms and deep learning models can be pushed to appropriate locations (i.e., edge, fog, cloud, and/or HPC computing resources) using a simple FaaS abstraction.

Serverless architecture naturally solves two important problems for portability and programmability [14]. First, the serverless programming model greatly reduces the burden on users or developers in developing, deploying and managing applications, as there is no need to understand the complex underlying procedures and distributed system to run the applications. Second, the functions are flexible to run on either edge or cloud, which achieves the desired portability.

However, in this thesis's view, the serverless computing model is able to satisfy only a limited subset of services, notably those with event-driven nature and request-reply nature. Continuum application/services are however comprised of both context-aware control loops and servers triggered by specific actions. The FaaS model can answer only the demand for the latter case. There is thus space for serverless computing at the edge, but a more general-purpose baseline has to be laid first of all.

TODO: Dynamic System Updates and Dynamic configuration

TODO: OCI Images

6) Mobility

In the continuum, services can be reallocated to follow the movements of the end users, and the data and state along should also be reallocated. Therefore, the collaboration issues (e.g., synchronization, aforementioned data/state migration, etc.) have to be addressed across multiple layers in the infrastructure.

When mobility is required provisioning of data and services needs also to be performed with high reactivity and reliability. For instance, in the context of smart mobility, vehicles are often on the move and the vehicular networking and communication is often intermittent or unreliable [19].

Being co-located at base stations, MEC servers are available anywhere the user moves and can support mobility between cellular cells.

TODO

7) Reliability

When applications are deployed in resource-constrained environments a number of challenges related to device failure or unreachability exists. Things at the edge of the network could fail due to various reasons and they could also report fail to report data under unreliable condition such as low battery level.

Various new communication protocols for IoT data collection have been proposed in the last years (e.g. CoAP, MQTT, AMPQ) [31]. These protocols serves well for the support of low energy and highly dynamic network condition. However, their connection reliability is not as good as BlueTooth or WiFi. If both sensing data and communication are not reliable, the system must leverage multiple reference data source and historical data record to be provide a reliable service.

8) Security and Privacy

The integration of edge computing, fog computing and cloud computing will raise some new and unforeseen security issues. Unique and unstudied scenarios, such as the interplay of heterogeneous edge nodes, and the migration of services across global and local scales, create the potential for original channels of malicious behavior [35].

Similar to the health record data, end user data collected at the edge of the network should be stored at the edge and the user should be able to control if the data should be used by service providers. However, as edge computing stores and processes data at the edge, the privacy-sensitive information associated with end users could be exploited and be more vulnerable than cloud servers.

Moreover, for applications like smart grids or sensor networks, an adversary could report false data, modify the other user data, tamper with their own smart meter, or spoof IP addresses, and further disrupt the effectiveness of sensor management in IoT systems.

9) Context awareness

Today's web services are rarely context-sensitive. However, in order to support context-sensitive services at the edge, they need to be aware of different aspects of the environment they are acting in. Context services can provide such information to the application services that implement local control loops and that trigger specific actions based on context events. Such context services can be composed of lower-level services which deliver individual sensor readings.

On this matter, the MEC architecture provides the advantage of data locality, which lets a given MEC server to store the context data only regarding the devices within the region covered by its base station. Such advantage is two-fold: providing location aware services is easier and less data must be persisted on each MEC server. As an example, MEC servers in [25] can retrieve the features of the points-of-interest in an augmented reality scene, match them against a local database, and return the corresponding data (information about monuments, buildings and other points of interest) to the client application.

10) Energy efficiency

Obtaining energy efficiency in both data processing and transmission is an important open issue. For handling such issue, several directions have been proposed: more efficient data transmission and compression technologies, data caching mechanisms for reusing collected data in time-tolerant applications, middlewares to improve availability and to compress data in case of continuous and long-duration monitoring of data.

11) Storage

The edge is the source of prodigious data, and will become the most important part of big data generation, if it is not already. Thus, massive data amounts need to be uploaded to edge or cloud-based storage. The storage space of edge nodes is limited though, and there is no large-scale and long-lived storage to compare with the cloud computing data centers.

III. SYSTEM DESIGN

A. ARCHITECTURE

TODO

B. KUBERNETES

TODO

C. REST

Stateless: the feasibility of hosting dedicated virtual machines, containers, and stateful applications is limited, as edge nodes cannot scale "infinitely" to host always-running VMs/containers as the cloud itself.

Most things are connected to the cloud through web based interfaces, which are able to reduce the complexity for developing such applications. However, they are not specifically designed for efficient machine-to-machine communications and introduce overhead in terms of network load, delay, and data processing.

TODO: Web of Things

D. COAP

Inefficient communications protocols: although it was designed for reliability and congestion-avoidance, TCP carries significant overhead and can have suboptimal performance for links with high latency or packet loss, both of which are common across the wide-area Internet [8]. Middle mile congestion exacerbates the problem, as packet loss triggers TCP retransmissions, further slowing down communications.

Additionally, for interactive applications, the multiple round trips required for HTTP requests can quickly add up, affecting application performance.

IV. EVALUATION

A key component in IoT environments is represented by sensor networks [35]. For example, they can cooperate with RFID systems to better track the status of things, getting information about position, movement, temperature, etc. Sensor networks are typically composed of a potentially high number of sensing nodes, communicating in a wireless multi-hop fashion.

Wireless sensor networks (WSNs) can provide various useful data and are being utilized in several areas like health-care, government and environmental services (natural disaster relief), defense (military target tracking and surveillance), hazardous environment exploration, seismic sensing, etc.

Recent technological advances have made efficient, low-cost, and low-power miniaturized devices available for use in large-scale, remote sensing applications. In this context, the timely processing of huge and streaming sensor data, subject to energy and network constraints and uncertainties, has been identified as the main challenge.

V. CONCLUSION

TODO

.

Appendixes, if needed, appear before the acknowledgment.

ACKNOWLEDGMENT

TODO

REFERENCES

[1] Haller S., Karnouskos S., Schroth C. (2009) The Internet of Things in an Enterprise Context. In: Domingue J., Fensel D., Traverso P. (eds) Future Internet – FIS 2008. FIS 2008. Lecture Notes in Computer Science, vol 5468. Springer, Berlin, Heidelberg.

[2] Mell, P., and Grance, T. 2011. The NIST definition of cloud computing, Recommendations of the National Istitute of Standards and Technology, NIST Special Publication 800-145.

[3] Dragoni N. et al. (2017) Microservices: Yesterday, Today, and Tomorrow. In: Mazzara M., Meyer B. (eds) Present and Ulterior Software Engineering. Springer, Cham.

[4] T Lynn, JG Mooney, B Lee, PT Endo. 2020. The cloud-to-thing continuum: opportunities and challenges in cloud, fog and edge computing.

[5] Roy Fielding. 2000. Representational State Transfer (REST). https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. Accessed on: Apr. 15, 2021.

[6] MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R. and Hamilton, B.A., 2006. Reference model for service oriented architecture 1.0. OASIS standard, 12(S 18).

[7] Satyanarayanan, M., Bahl, P., Caceres, R. and Davies, N., 2009. The case for vm-based cloudlets in mobile computing. IEEE pervasive Computing, 8(4), pp.14-23.

[8] Nygren, E., Sitaraman, R.K. and Sun, J., 2010. The akamai network: a platform for high-performance internet applications. ACM SIGOPS Operating Systems Review, 44(3), pp.2-19.

[9] Prolonged AWS outage takes down a big chunk of the internet. 2020. [Online]. Available: https://www.theverge.com/2020/11/25/21719396/amazon-web-services-aws-outage-down-internet. Accessed on: Apr. 15, 2021.

[10] Latre, S., Famaey, J., De Turck, F. and Demeester, P., 2014. The fluid internet: service-centric management of a virtualized future internet. IEEE Communications Magazine, 52(1), pp.140-148.

[11] AbdelBaky, M., Zou, M., Zamani, A.R., Renart, E., Diaz-Montes, J. and Parashar, M., 2017, June. Computing in the continuum: Combining pervasive devices and services to support data-driven applications. In 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS) (pp. 1815-1824). IEEE.

[12] Beckman, P., Dongarra, J., Ferrier, N., Fox, G., Moore, T., Reed, D. and Beck, M., 2020. Harnessing the Computing Continuum for Programming Our World. Fog Computing: Theory and Practice, pp.215-230.

[13] Shi, W., Cao, J., Zhang, Q., Li, Y. and Xu, L., 2016. Edge computing: Vision and challenges. IEEE internet of things journal, 3(5), pp.637-646.

[14] Yi, S., Hao, Z., Zhang, Q., Zhang, Q., Shi, W. and Li, Q., 2017, October. Lavea: Latency-aware video analytics on edge computing platform. In Proceedings of the Second ACM/IEEE Symposium on Edge Computing (pp. 1-13).

[15] Baresi, L., Mendonça, D.F. and Garriga, M., 2017, September. Empowering low-latency applications through a serverless edge computing architecture. In European Conference on Service-Oriented and Cloud Computing (pp. 196-210). Springer, Cham.

[16] Jang, S.Y., Lee, Y., Shin, B. and Lee, D., 2018, October. Application-aware IoT camera virtualisation for video analytics edge computing. In 2018 IEEE/ACM Symposium on Edge Computing (SEC) (pp. 132-144). IEEE.

[17] Botta, A., De Donato, W., Persico, V. and Pescapé, A., 2016. Integration of cloud computing and internet of things: a survey. Future generation computer systems, 56, pp.684-700.

[18] Zhang, Q., Cheng, L. & Boutaba, R. Cloud computing: state-of-the-art and research challenges. J Internet Serv Appl 1, 7–18 (2010).

[19] He, W., Yan, G. and Da Xu, L., 2014. Developing vehicular data cloud services in the IoT environment. IEEE transactions on industrial informatics, 10(2), pp.1587-1595.

[20] Cisco fog computing solutions: Unleash the power of the Internet of Things. [Online]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-solutions.pdf. Accessed on Apr. 15, 2021.

[21] Mugarza, I., Amurrio, A., Azketa, E. and Jacob, E., 2019. Dynamic software updates to enhance security and privacy in high availability energy management applications in smart cities. IEEE Access, 7, pp.42269-42279.

[22] Mitton, N., Papavassiliou, S., Puliafito, A. and Trivedi, K.S., 2012. Combining Cloud and sensors in a smart city environment.

[23] Chen, B., Wan, J., Celesti, A., Li, D., Abbas, H. and Zhang, Q., 2018. Edge computing in IoT-based manufacturing. IEEE Communications Magazine, 56(9), pp.103-109.

[24] Gartner. Leading the IoT. 2017. [Online]. Available: https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf. Accessed on Apr. 15, 2021.

[25] Baresi, L., Mendonça, D.F. and Garriga, M., 2017, September. Empowering low-latency applications through a serverless edge computing architecture. In European Conference on Service-Oriented and Cloud Computing (pp. 196-210). Springer, Cham.

[26] Yousaf, F.Z., Bredel, M., Schaller, S. and Schneider, F., 2017. NFV and SDN—Key technology enablers for 5G networks. IEEE Journal on Selected Areas in Communications, 35(11), pp.2468-2478.

[27] Jonas, E., Schleier-Smith, J., Sreekanti, V., Tsai, C.C., Khandelwal, A., Pu, Q., Shankar, V., Carreira, J., Krauth, K., Yadwadkar, N. and Gonzalez, J.E., 2019. Cloud programming simplified: A berkeley view on serverless computing. arXiv preprint arXiv:1902.03383.

[28] Protocol Buffers. [Online]. Available: https://developers.google.com/protocol-buffers. Accessed on Apr. 15, 2021.

[29] OpenAPI Specification, Version 3.0.3. [Online]. Available: https://swagger.io/specification/. Accessed on Apr. 15, 2021.

[30] Shopify App Store. [Online]. Available: https://apps.shopify.com/. Accessed on Apr. 15, 2021.

[31] Naik, N., 2017, October. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In 2017 IEEE international systems engineering symposium (ISSE) (pp. 1-7). IEEE.

[32] Grüner, S., Pfrommer, J. and Palm, F., 2016. RESTful industrial communication with OPC UA. IEEE Transactions on Industrial Informatics, 12(5), pp.1832-1841.

[33] Grozev, N. and Buyya, R., 2014. Inter-Cloud architectures and application brokering: taxonomy and survey. Software: Practice and Experience, 44(3), pp.369-390.

[34] Bernstein, D., 2014. Containers and cloud: From lxc to docker to kubernetes. IEEE Cloud Computing, 1(3), pp.81-84.

[35] Yu, W., Liang, F., He, X., Hatcher, W.G., Lu, C., Lin, J. and Yang, X., 2017. A survey on the edge computing for the Internet of Things. IEEE access, 6, pp.6900-6919.

• • •