A Survey of IoT Frameworks for Low-Powered Devices

Andrei-Robert CAZACU
Bucharest University of Economic Studies, Romania
andrei.cazacu@csie.ase.ro

Thanks to technological advancements, our lives are getting more intertwined with the connected world as more smart devices are coming to market. As such, the clear separation of devices as things (end-devices in IoT systems) and human operated devices is getting increasingly buried. This led to the creation of the term Internet of Everything (IoE) which is defined by Cisco as the "the networked connection of people, process, data, and things" [1]. The main difference between IoT and IoE is inclusion of people in the ecosystem, which greatly increases the number of connected parties. This increase in connected parties creates a strain on our existing infrastructure which is relying on cloud computing for performing most operations. Even though this resource provides heaps of computational power, the weak link in this scenario is the network, where all the connected devices can easily overload the available bandwidth, leading to slow response speeds and low general availability. The answer to this problem lies with technology that already exists and is not yet fully exploited as a distributed computing powerhouse, IoT. This paper aims to summarise the concept of computing at the edge, common architectural patterns, existing solutions, while also discussing real-world applications.

Keywords: *Edge Computing, Cloud Computing, EdgeX, Internet of Things, Internet of Everything, IoT, IoE*
DOI: 10.24818/issn14531305/28.2.2024.03

1 Introduction

In our ever-evolving society, the Internet has become a foundational pillar in how we live our lives, starting with us browsing the news and ordering food to be delivered at our doorstep. All these services rely heavily on cheap and abundant computing power, which with today's technology we have plenty of, being able to rent very beefy machines for less than 0.5$ per hour [2].

According to Cisco's Annual Internet Report [3], nearly two-thirds of the global population should have Internet access as of writing this article, 5.3 billion human users. Additionally, the number of devices connected to IP networks should reach 29.3 billion, an almost 2-fold increase over the baseline of 18.4 billion in 2018. This massive increase is expected to be sustained, creating a massive strain on our existing infrastructure. The stress can be alleviated by rethinking our usage of IoT devices, which up until now were only though of simple sensing devices, acquiring data and sending it to be processed in the cloud following a simple Node-Hub-Cloud architecture.

This type of architecture is easy to implement and efficient in the real-world for simple applications like smart home, being proposed in research papers dating as far back as ten years ago, such as [5], and collectively endorsed even in recent papers such as [6]. It must be remembered that IoT is not a novel concept, the first IoT device being *invented* back in the early 1980s [7], an Internet connected Coca-Cola beverage machine. As such, what we are seeing is a strive to make this more user-friendly with the advent of cheap and low-powered Internet connected devices such as the ESP32.

The presented paper aims to broaden the public perception on IoT, outlining that the Node-Hub-Cloud model is not leveraging the full computing power of the edge devices and is not suitable for applications which require real-time characteristics.

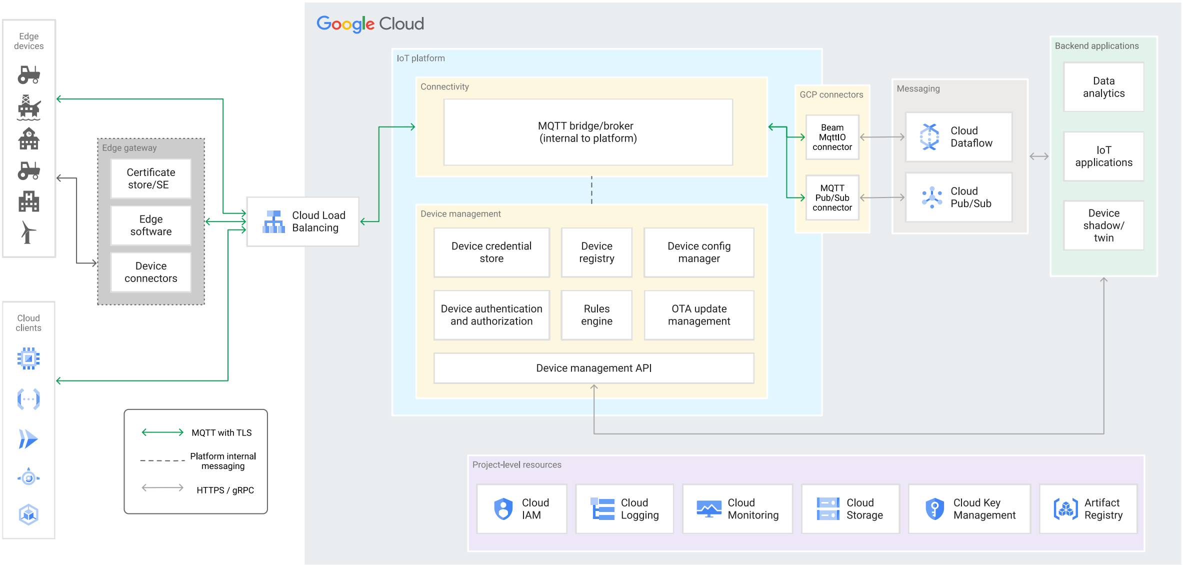

Fig. 1. Google IoT Core architecture [4]

Is it important to note the high degree of heterogeneity present in the IoT ecosystem, having devices as small as BLE (Bluetooth Low Energy) temperature sensor, smart-home devices such as Zigbee light bulbs, Wi-Fi connected washing machines, 5G connected cars to fully fledged Linux running devices such as Raspberry Pi 4. Researchers outlined standardisation [5] as a major issue preventing market-wide usage of M2M communication among IoT devices, with recent developments such as the 2022 Matter [8] protocol, a joint venture between big companies such as Amazon, Apple, Google and Zigbee Alliance, helping to bridge the gap [9].

Having a *lingua franca* among heterogenous devices working over multiple transport layer protocols, it now makes sense to think of all the possible ways we can use all the resources at our disposal to solve the growing pains of our ever-increasing base of Internet users.

Expanding IoT to include the human actors leads to the Internet of Everything, with papers such as [10] acknowledging the capability of these devices to be used as distributed computing powerhouses. This will mark a new boom in the evolution of Web, following disruptive trends such as Semantic Web and Widespread availability of Mobile Internet Connection.

Currently, IoE is an emerging technology which can be defined as the intelligent intertwining of people's daily lives with smart cities, homes, medicine, agriculture, cars and surrounding objects. All these connections shall be managed by the end-user via the mobile phone, desktop or automatically managed through AI-powered algorithms. All of these require besides processing power, a fine coordination and real-time processing of data, in which service locality becomes a major factor. Today's computing usually happens in the data centres of a Public Cloud provider such as AWS (Amazon Web Services), Azure, GCP (Google Cloud Platform) which are located only in few cities among the globe (e.g. AWS serves European customers with data centres in Frankfurt, Ireland, London, Milan, Paris, Stockholm, Zurich and Spain). This constant flux of data between local devices and far away data centres pose issues like:

- Increased latency: with the exponential increase in the number of devices, a large amount of data will sink to the cloud, creating a strain on our networking infrastructure, thus increasing latency, posing a threat to functionalities which require real-time responses (e.g. traffic light coordination)
- Environmental impact: in a world marked by resource scarcity, computing power shouldn't be wasted; as such, computing

as close to the source of the data will reduce the in-flight time, thus reducing the power consumption

- Data security and privacy: Since all the computing happens in the cloud, sensitive data must be transmitted, leaving the end-user susceptible to MiTM (Man in The Middle) attacks; performing security sensitive operations on device would negate some of the risks currently present.

2 Computing at the edge

Large amounts of data are being constantly produced at the edge of the network, ranging from megabytes sized Instagram uploads to constant streams of data from CCTV or sensors. As of now, this data is constantly sent to the cloud to be processed and analysed by one of the many distributed processing systems available such as MapReduce [11], Apache Hadoop [12] or Apache Spark [13], but work has been put towards making this process more efficient. Developments such as fog computing [14] and container-based frameworks for performing computations at the edge [15] are striving to improve the current *status quo*, demonstrating the need for this computing paradigm.

As mentioned earlier, currently all computing happens in the cloud, which even though is cheap and constantly improving in processing speed, it must cope with the ever-increasing network load since all the communication happening between the end-user and the service is happening as a pull request initiated by the client. With the growing quantity of data generated by edge devices, being already at predicted 1.6ZB per month as of 2021 [16], our capability of moving that data effectively is strained. As such, it would make sense to leverage the increase in computation power to shift some of the work to edge devices and alleviate the network pressure, using the cloud solely for resource-intensive applications. This change would mark the transition from the current Pull model to Push model, where only excess workload is being offloaded. To better visualise the data amount, a Siemens report [17] from 2021 reports an estimated of 3Gbit/s to 40 Gbit/s of data will be generated each second for any autonomous driving vehicle, which multiplied with an estimated 278 million cars in the USA alone [18] leads to an amount of data which can’t achieve real-time processing using conventional cloud systems.

Nonetheless, this paradigm shift doesn’t imply getting rid of the existing cloud infrastructure, instead introducing a symbiosis relationship which should be mutually beneficial to all parties, having edge devices compute privacy sensitive operations, along with operations which require real-time response and offloading the rest of the workload via the network to the cloud.

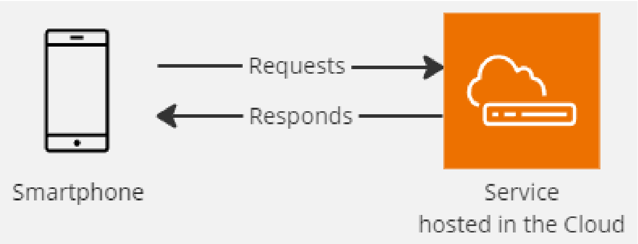

Fig. 2. Cloud computing Architecture

In Figure 2, we can observe a typical IoT architecture showing data producers pushing data to the cloud where it would be processed and stored. While this approach is not fundamentally incorrect, it leads to inefficiencies in the process due to the unidirectional communication, having the end device be solely responsible of producing data; also, most end-nodes in IoT are energy constrained devices with limited network bandwidth; hogging the entire bandwidth with a stream of data will lead to poor battery life. Additionally, this solution is bound to encounter scaling issues, having in mind the above-mentioned growth estimates [16]. All of these would indicate that doing lightweight processing of the data on the edge of the network would be more energy efficient, thus leading us to Figure 3.

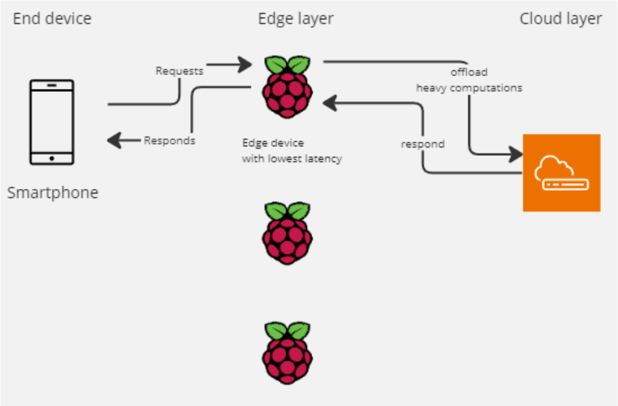

Fig. 3. Edge computing architecture

The third figure outlines the proposed edge computing solution, where all the communication is bidirectional, thus the end device could either send complete results, effectively being the edge processing node, or pass the request further to the thin edge layer, ultimately to be offloaded to the cloud if needed.

3 Common architectural patterns and existing solutions

Seeing as IoT is becoming a mainstream technology and having an ever-larger device count, edge computing has the potential to become the next disruptive technology following affordable cloud computing and artificial intelligence.

As such, it's important to learn from past mistakes [5] and have a reference architecture, with efforts already being made by the Linux foundation through their EdgeX Foundry project; commercial solutions like AWS Greengrass are also available, while research community driven solutions like FogBus2 are also popping up [15].

All these are marked by the distinguishing devices into separate layers based on their purpose:

End-Node

Any device connected to the edge network is considered an end-node, which can any smartphone, car, temperature sensor, medical device etc. These devices can be either raw data producers (temperature sensor) or consumers (end-user streaming YouTube video).

Edge

The edge layer is the pillar of this architecture, often being able to overlap with the end-node when the used devices have spare computing power. It consists of a plethora of devices widely distributed in many geographically locations close to the users, having a direct bidirectional connection to the end-nodes and to the cloud.

It has as ingress raw data can be either processed locally fully, partially, or offloaded to the cloud, and can also serve requests of end-nodes with a lower delay, thus being suitable for real-time applications.

Cloud

Among all these layers, the cloud still possesses the biggest amount of processing power available, albeit with a bigger delay induced by the increased distance between the data centre and the end-node. As such, the cloud is still the most suitable place for long-term storage of data and compute-heavy analysis.

Now, we can observe the complementary nature of these layers, where all requests surpassing the available resources of the current machine, be them compute or storage, will ripple to the next layer to be processed. To better summarize, we can state the edge layer is better suited to applications requiring real-time processing, having a high degree of locality relative to the user and having little to none network strain, while the cloud layer is suited for large scale processing, being located far away from the end device, thus posing a strain on the network.

3 1 EdgeX Foundry Project

The EdgeX Foundry project aims to bridge the interoperability gap between the *"Wild West"* of IoT devices with the civilised world of enterprise IT [19], with a stated intent of building a common framework for Industrial IoT edge computing [20]. It does so by creating a vendor-neutral open-source platform for enabling data collection at the edge from a multitude of *things*(sensors) working over a plethora of different communication protocols and performing operations at the edge of the network with the possibility to offload some of the workload to the cloud if necessary.

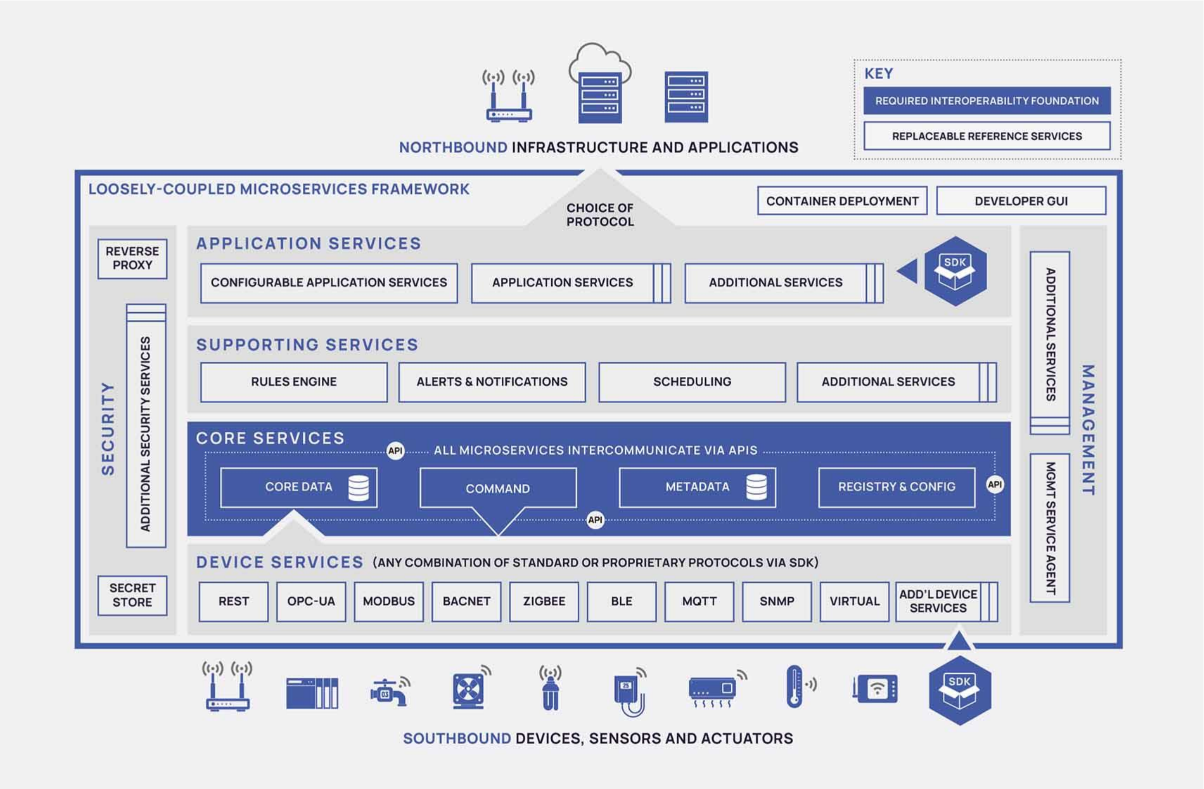

Fig. 4. EdgeX Foundry architecture [21]

This flexible architecture is created around the concepts of southbound and northbound, with EdgeX powered devices sitting between them. The south side starts with all IoT resource-constrained devices such as CCTV cameras, temperature sensors and stops at the edge of the network that communicates directly with those devices. The north side is represented by the cloud, where the computation heavy workloads are offloaded, and the data is stored for the long term.

This architecture is clearly delimited in four service layers:

Device services layer

This layer translates the raw data received from southbound devices into platform agnostic information which can be processed later.

It ships with multiple replaceable reference implementations for different communication protocols out of the box, such as MQTT, a widely used protocol among IoT devices, working in a publish-subscribe paradigm, REST, CoAP, BLE (Bluetooth Low-Energy), Zigbee and many others. Even though the out of the box offering should suffice for most of the commercially available IoT products, the EdgeX project is designed to be future-proofed, allowing the user to create bespoke device services using the available SDK (Software Development Kit).

Core services layer

The core services layer separates the south side of the network from the northbound part. It consists of four mandatory components:

- Core data: persistency repository and management service for data collected from the end devices
- Command: service which permits and controls actuation requests from the north side to the south side
- Metadata: management service and repository of metadata about objects connected to the EdgeX Foundry; allows provisioning of new devices and pairs them with the associated owning service (from the Application layer)
- Registry and configuration: responsible for orchestration of information exchange among services within EdgeX foundry and microservices

Supporting services layer

This layer is composed of multiple optional services aimed to support the functionality of the EdgeX Foundry, ranging from edge analytics to logging, scheduling and data clean up.

Application services

The application layer is meant to host user-defined microservices, allowing businesses to customise the system to their needs without having to start from scratch. It allows querying for sensor data via already defined APIs and performing local computation of it or offloading it to the cloud.

All the applications running inside the EdgeX Foundry are running in a containerised fashion, thus achieving platform agnosticism and extreme flexibility; there is also the added benefit of increased security by forbidding direct communication between end-devices and the cloud, every bit of data going through the edge layer and being orchestrated accordingly.

3 2 FogBus2 Framework

Another solution that is relying on containerisation for performing computations at the edge is the scientific research originated FogBus2 Framework [15].

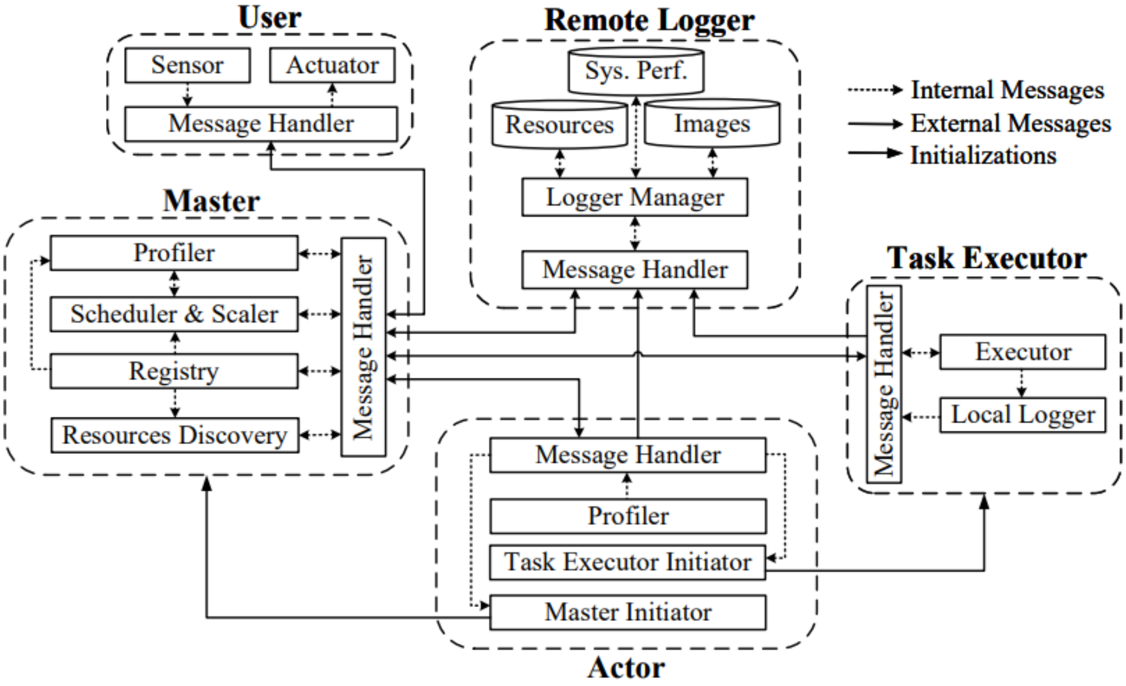

Fig. 5. FogBus2 architecture [15]

This framework is IoT oriented, and consists of five main components which containerised and deployed with Docker, thus achieving platform agnosticism:

User component

This component to be executed by the end-device, which is the resource constrained IoT device, and can either be a sensor, actuator, or both. It also sends computing requests to the master, either by submitting a dependent or independent task.

Master component

The master component schedules computing tasks received from the user component, while also managing the execution process and message passing. It can be run either in the edge layer, or in the cloud, and supports dynamic profiling of the host machine and resource discovery of actors and task executors.

Actor component

This component is used for spawning containers on the machine it's running, more specifically the master container if there is the current master is running out of resources, or for a task executor if the current actor is the one most suitable. It is to be run on any node either in the edge, or in the cloud.

Task Executor component

The task executor component is represented by any spawned container containing a user defined dependent or independent task. As such, IoT applications can be split into multiple tasks and infinitely scaled on multiple machines.

Moreover, task executors are idempotent irrespective of their input, as such they can be reused if any new task of the same type is to be scheduled during the *cooling-off* period. In this period, the container can be reused to serve another request.

When the task finishes executing, the results are sent either to dependent tasks if there are any, or to the master.

Remote Logger component

This component is used to store logs from across all components, collecting them through an event-driven message passing interface. The remote logger then stores all the data to any persistent storage, which could be either the filesystem or a database.

The main contribution of the above framework is the introduction of smart scheduling through OHNSGA (Optimised History-based Non-dominated Sorting Genetic Algorithm), using input points such as available resources and past decisions to determine the most suitable actor for executing any given task.

3 3 AWS Greengrass

A commercial solution that offers similar computing possibilities, albeit with no smart scheduling of containerised task built-in is AWS's IoT Greengrass.

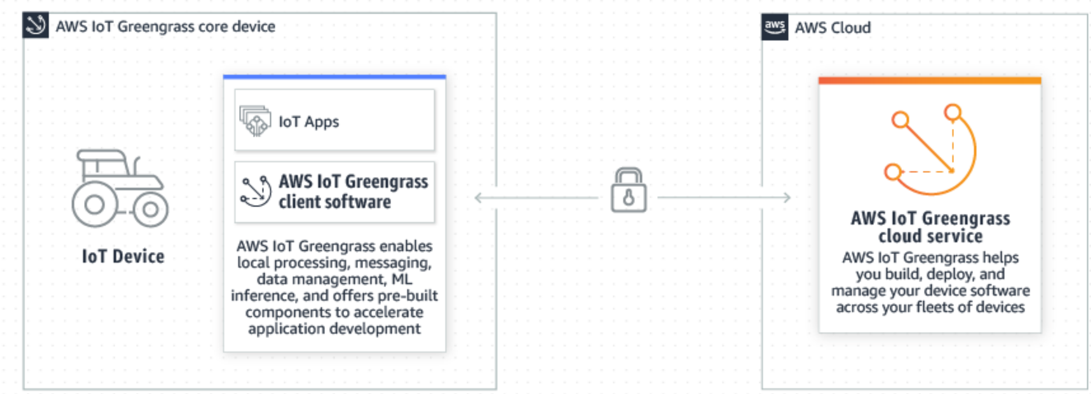

Fig. 6. AWS IoT Greengrass [22]

It has a stated intent of helping any user build, deploy and manage IoT applications easily, being designed with scalability in mind and having compatibility with all the offerings in AWS Cloud. The framework is open source, with a billing model depending on the number of devices connected and number of messages being passed between them.

The main features of this framework are the support for local processing of AWS Lambda, being able to reuse existing applications to perform computations at the edge of the network, support for running containers and feature-rich messaging library.

The above architecture is fully modular, allowing administrators to deploy built-in components at will, such as ML inference engines, sensor data processing workloads and custom-built applications.

We can observe that all the frameworks described above have a few things in common, such as the usage of containerisation to achieve platform agnosticism, leverage of message passing interfaces and segregation of the network into two main areas: edge and cloud.

4 Real-world applications

This technology has real-world applications in today’s world, both in IoT oriented solutions as well as general compute, both directions being a hot research topic. To further state the need for development in the direction of edge computing, we can exemplify a few real-word scenarios which are testing today’s cloud computing capabilities.

4.1 Video streaming applications with large number of users

Traditionally, video streaming is handled using CDNs (Content Delivery Networks) which store the static video files and serves them to the end user. Even though this solution proves effective for the moment, the rapid increase in the amount of data consumed will increase an immense strain on the network, thus significantly slowing down the overall transfer speed and availability in general.

A solution to this problem would be the caching of highly requested data in a dense layer of edge devices, which even though storage bound should greatly reduce the network load, if the traffic follows the Pareto principle. This principle, also known as the 80/20 rule, states that 80% of the effects in any large system is caused by 20% of the variables in the system; applied to this example, it would mean that 80% of the traffic is generated by 20% of the content, thus reducing our storage requirements 5-fold.

4.2 Social networks relying on user generated content

Since the advent of social media, the end-user has transformed from a net consumer of services (e.g. reading the news, streaming the latest sports game) to consuming and producing at the same time.

In fact, this important transition marked a big stepping stone in the history of the Web, marking the start of the Social Web (Web 2.0), and gave birth to multiple websites that we are well accustomed to today like Facebook, WordPress, Wikipedia, YouTube and many others.

For the user generated content to be properly displayed on the plethora of devices available, it must undergo several processing stages currently happening in the resource-rich cloud environment, which from a computing power perspective makes sense, but puts an immense strain of the network. To gain some perspective, 1 minute of 4K video recording in Blackmagic RAW format with a 3:1 Constant Bitrate uses up to 8GB [23], and with an average YouTube video size of 11.7 minutes [24] that means 93.6GB of data which needs to be streamed to YouTube’s servers to be processed. This will hog the available bandwidth, and in the long term will prove not sustainable. What this paper proposes is a dense network of edge devices close to the user which shall handle the distributed workload of encoding the file and then upload the intermediate results to the cloud server to be pieced together and saved for long-term storage. Even though the edge devices will have significantly less processing power than the cloud, this will be compensated by an increased number of devices, and the proximity of these devices to the user will lower the overall network load.

5 Conclusions

To summarize, this paper aims to prove that IoT is the answer to our present and future computing problems by providing an overview of the most currently used application architecture, which is relying on the Cloud for carrying out any sort of computation, and demonstrating that this is bound to hit a wall in terms of scalability due to the networking constraints; this scalability problem arose from the constant growth in the number of Internet connected devices, bound to reach 29.3 billion.

To solve this problem, the presented paper suggests a hybrid edge-cloud computing approach, where the end-devices are to connect to edge devices in proximity for performing computations as close to the data source, with the ability to smartly offload excess workload to the cloud. Even though marked by heterogeneity, the IoT market is starting to solidify, with actions being taken to standardise communication among devices, through the Matter protocol, and with already existing protocols such as MQTT becoming mature, frameworks designed with edge computing in mind have started to appear. Among those, this paper analyses the stated intent and architecture of three solutions from different markets, such as the Linux Foundation originated EdgeX Foundry, scientific research born framework FogBus2 and the commercial solution AWS Greengrass.

Among these three solutions, we can observe some common patterns, such as the use of containerisation to achieve platform agnosticism and ease of scaling, heavy use of message passing interface and the hard split of the network into two domains: edge and cloud.

Even though it’s only starting to get traction, edge computing already has real-world applications such as video streaming applications with large number of users, and even social media networks, since they rely on user generated content.

In conclusion, I consider that edge computing is a hot topic for researchers and will continue to see further development considering the issue that it solves.

References

[1] Cisco, “The Internet of Everything,” 2013. [Online]. Available: https://www.cisco.com/c/dam/en_us/about/business-insights/docs/ioe-value-at-stake-public-sector-analysis-faq.pdf.

[2] Amazon, “Amazon EC2 On-Demand Pricing,” [Online]. Available: https://aws.amazon.com/ec2/pricing/on-demand/.

[3] Cisco, “Cisco Annual Internet Report (2018–2023) White Paper,” 2018-2023. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

[4] Google, “IoT platform product architecture on Google Cloud,” [Online]. Available: https://cloud.google.com/architecture/connected-devices/iot-platform-product-architecture.

[5] S. Chen, H. Xu, D. Liu, B. Hu and H. Wang, “A Vision of IoT: Applications, Challenges, and Opportunities With China Perspective,” *IEEE Internet of Things Journal,* vol. 1, no. 4, pp. 349-359, 2014.

[6] P. Gokhale, O. Bhat and S. Bhat, “Introduction to IOT,” *International Advanced Research Journal in Science, Engineering and Technology,* vol. 5, no. 1, 2018.

[7] J. Teicher, “The little-known story of the first IoT device,” *IBM,* 2018.

[8] Connectivty Standards Alliance, “Matter Specification Version 1.0,” 2022. [Online]. Available: https://csa-iot.org/wp-content/uploads/2022/11/22-27349-001_Matter-1.0-Core-Specification.pdf.

[9] B. Curtis, “CES 2022: Matter And Thread Win The IoT Connectivity Wars,” *Forbes,* 2022.

[10] S. P. Singh, G. Dhiman, W. Viriyasitavat and S. Kautish, “A Novel Multi-

Objective Optimization Based Evolutionary Algorithm for Optimize the Services of Internet of Everything," *IEEE Access*, vol. 10, 2022.

[11] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *OSDI*, 2004.

[12] Apache, "HDFS Architecture Guide," 2008. [Online]. Available: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.pdf.

[13] M. Zaharia, R. S. Xin, P. Wendell, T. Das and M. Armbrust, "Apache Spark: A Unified Engine for Big Data Processing," 2016. [Online].

[14] S. Yi, C. Li and Q. Li, "A Survey of Fog Computing: Concepts, Applications and Issues," *Mobidata*, 2015.

[15] Q. Deng, "FogBus2: A Lightweight and Distributed Container-based Framework for Integration of IoT-enabled Systems with Edge and Cloud Computing," 2021.

[16] K. Compton, "Cisco's Global Cloud Index Study: Acceleration of the Multicloud Era," 13 February 2018. [Online]. Available: https://blogs.cisco.com/news/acceleration-of-multicloud-era.

[17] Siemens, "The Data Deluge: What do we do with the data generated by AVs?," 2021. [Online].

[18] A. Tilford and M. Megna, "Car Ownership Statistics 2023," October 2023. [Online]. Available: https://www.forbes.com/advisor/car-insurance/car-ownership-statistics/.

[19] B. Curtis, "The Future Of Industrial IoT Gateways: Civilizing The Wild West Of IoT," [Online]. Available: https://www.forbes.com/sites/moorinsights/2020/11/19/the-future-of-industrial-iot-gateways-civilizing-the-wild-west-of-iot/.

[20] EdgeX Foundry, "Introduction," [Online]. Available: https://nexus.edgexfoundry.org/content/sites/docs/staging/master/docs/_build/html/Ch-Intro.html.

[21] EdgeX Foundry, "Why EdgeX?," [Online]. Available: https://www.edgexfoundry.org/why-edgex/.

[22] Amazon, "What is AWS IoT Greengrass?," [Online]. Available: https://docs.aws.amazon.com/greengrass/v2/developerguide/what-is-iot-greengrass.html.

[23] Blackmagic , "Blackmagic RAW," [Online]. Available: https://www.blackmagicdesign.com/products/blackmagiccinemacamera/blackmagicraw.

[24] L. Ceci, "Average YouTube video length as of December 2018, by category (in minutes)," [Online]. Available: https://www.statista.com/statistics/1026923/youtube-video-category-average-length/.

Andrei CAZACU has graduated the Cybersecurity Master Program, Bucharest University of Economic Studies in 2022 and is currently pursuing a PhD in the field of Fog-Edge Computing. Currently, he is activating as a software developer in the IAM field.