This paper was accepted for publication in IEEE Internet Computing. The copyright was transferred to IEEE and the final version is available on IEEE Xplore® via https://doi.org/10.1109/MIC.2019.2918079.

A disaggregated cloud architecture for edge computing

Rafael Moreno-Vozmediano
Universidad Complutense de Madrid
Eduardo Huedo
Universidad Complutense de Madrid
Rubén S. Montero
Universidad Complutense de Madrid
Ignacio M. Llorente
Universidad Complutense de Madrid and Harvard University

The use of a disaggregated cloud architecture would allow companies and institutions to deploy their own edge computing infrastructure and services by combining their on-premises cloud infrastructure with multiple highly-dispersed edge nodes leased from third-party providers.

There is growing interest, both from researchers and industry, in edge and fog computing paradigms. These have emerged as a solution for overcoming the limitations of traditional cloud computing platforms to support low-latency environments. According to recent estimates, over 20 billion devices will be connected to the network by 2020[1] which will result in exponential growth in the amount of data with low-latency processing needs.

There have been multiple reports of use-cases in which services, or at least parts of them, were run much more efficiently next to where the data was created and/or consumed. A good example is that of a web platform offering low-latency services to users distributed in various geographic locations (e.g., e-healthcare services, remote driving assistance, online immersive games, etc.). This platform can be implemented as a central web server and a variable number of application servers distributed over several edge locations. Another example is the emerging 'internet of things' (IoT) which requires data processing near the devices in question.

These use-cases usually require decoupling the application into two parts: one residing in a centralized cloud and another running on the network edge. For example, cloud providers including AWS and Microsoft Azure are starting to offer software solutions to extend their cloud services to the edge. They are not offering edge infrastructure, but rather, service software that extends cloud data processing functionality to IoT devices. AWS IoT Greengrass extends AWS to edge devices so they can act locally upon the data they generate. This service requires the installation of the AWS IoT Greengrass Core software on a Linux system to allow the local execution of AWS Lambda code, messaging, data caching, and security for a group of nearby IoT devices. Greengrass works as a cache and a hub for local AWS-enabled devices. Azure follows a similar approach with Azure IoT Edge.

Many companies need a way to elastically serve the dynamic demand of the edge portion of these emerging applications, for example, in application servers or IoT hubs. Moreover, in most cases, the end users and IoT devices are distributed over various geographic locations. Therefore, these companies require a framework capable of efficiently deploying and managing such distributed edge computing

infrastructure which can dynamically and opportunistically support the execution of these edge hub services. Thus, here we define a disaggregated cloud model that can manage and combine both the local physical resources of on-premises data centers and the resources of multiple highly-dispersed edge nodes to build dynamic, agile, and decentralized edge computing environments.

One current question is, if the main cloud providers are not offering infrastructure at the edge, then who is? To meet the needs of edge computing, the main telecoms companies are building new 5G infrastructures to support faster mobile data speeds, thereby adding server power to their cell towers and central offices. These companies plan to offer these thousands of computing distribution points as a service to the providers of applications that require data to be transmitted and received with no latency. Meanwhile, smaller hosting providers including Packet and others in the Kinetic Edge Alliance are introducing edge infrastructures with scattered micro data centers that offer latencies of less than 10 ms into the main international locations.

The disaggregated cloud architecture we propose here can manage small to medium edge computing infrastructures comprising several tens to hundreds of geographically distributed edge resources. These could be leased to the aforementioned third-party providers which offer on-demand resources located at the edge of the network in the form of bare-metal hosts or virtual servers. Furthermore, our model could easily be extended to larger platforms by using a peer-to-peer decentralized federation of clouds.

DISAGGREGATED CLOUD ARCHITECTURE

Most edge computing environments (e.g., OpenStack Trio2o, or ENORM[2]) are based on a distributed management model, where each edge node is independent and is managed by its own management platform—also known as a Cloud operating system (OS). These distributed models include an upper-level edge gateway responsible for interacting with clients which routes client requests to the appropriate bottom edge nodes. This solution is highly scalable, however, it delegates most complex management tasks to the edge nodes, which can consume a lot of their resources. Furthermore, installing, configuring, and maintaining many Cloud OS instances can be an extremely complex and time-consuming task.

Here we propose a disaggregated management model based on a centralized Cloud OS that manages the local resources of an on-premises data center as well as those of different edge nodes, as shown in Figure 1. This model delegates most of the previously mentioned complexity to the Cloud OS, thus releasing the edge nodes from multifarious management tasks.

The main stakeholders on the disaggregated cloud architecture are:

- **Edge Application Providers**. These are the companies offering edge applications to their customers. They run their own Cloud OSs to build a disaggregated private cloud that dynamically allocates physical resources at the edge offered by **Edge Infrastructure Providers** and may also use resources from an on-premises cloud. They can deploy different types of services, such as web services using edge application servers to provide location-aware services and reduce latency[3], e-Health applications using edge e-Health gateways for rapid emergency responses[4], or IoT applications using edge IoT hubs for edge analytics[5].

- **Edge Infrastructure Providers.** These are third-party providers that lease out physical resources at the edge in the form of bare-metal servers.
- **Public Cloud Providers**. The Cloud OS can also interact with one or more external cloud providers, including AWS or Azure, to offer cloud bursting capabilities.

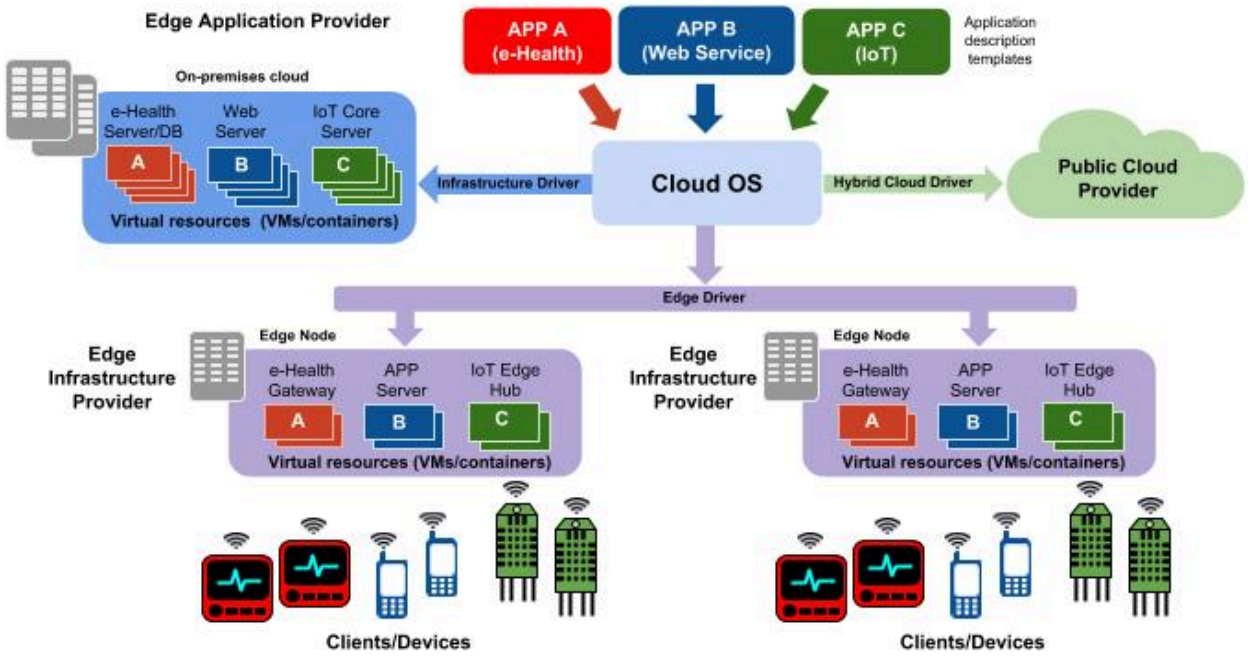

Figure 1. Disaggregated cloud architecture.

The main functions of the Cloud OS are:

- **Management.** As a cloud management platform[6], it manages physical resources located both in the on-premises cloud and the various dispersed edge locations, and provides users with a uniform access to this disaggregated resource pool. It is also responsible for the deployment and life-cycle management of virtual resources overlaid onto this physical infrastructure. It must be able to provide different virtualization technologies, such as virtual machines (VMs) or system containers, according to the service needs.
- **Auto-scaling.** The Cloud OS must also provide auto-scaling capabilities to accommodate the anticipated number of virtual resources demanded by the service. In this work we only consider reactive threshold-based mechanisms[7] which allow service providers to define scaling rules based on different performance metrics. For each service component and location (edge node or cloud site), these rules are defined in terms of specific upper and lower thresholds for the selected metric. Thus, if a metric falls above or below these cut-offs at a given moment and specific location, a scaling action is triggered which adds or removes virtual resources at this position.
- **Monitoring.** In terms of scalability, the monitoring system is one of the most critical components of the Cloud OS and one of its key parameters is the monitoring interval, i.e. the time that elapses between two consecutive readings of the monitoring metrics. This interval depends on several

factors[8], including the number of resources to be monitored, their load, and the network latency between them and the monitoring system.

The key challenges of this disaggregated model are scalability, orchestration, availability and security. A disaggregated architecture would not scale properly for very large edge computing environments, but it is eminently suitable for small to medium private edge infrastructures consisting of tens of thousands of resources. The model can be extended to larger, worldwide platforms by using a peer-to-peer decentralized federation of cloud instances.

Regarding the orchestration and availability of geo-distributed virtual resources over multiple edge locations, the Cloud OS should allow the specification of placement constraints through the definition of VM groups and affinity rules over multiple availability zones, clouds and edge locations[9]. The application must be aware of the location of its users (managing also their mobility), and it is responsible for defining where the service components must be deployed.

The use of a single Cloud OS instance greatly simplifies security management, as it provides centralized tools for user authentication, access control (ACLs), definition of security groups, and secure communication with edge nodes. Moreover, Service Level Agreements (SLA) with the edge infrastructure providers guarantee the integrity, availability, and authenticity of the edge nodes. In addition, secure VPN tunnels can be implemented to provide integrity and authentication at application level[10].

PROOF OF CONCEPT AND RESULTS

In this section, we show the viability of the disaggregated cloud architecture. Our experimental setup comprises a Cloud OS based on OpenNebula, an on-premises cloud, and a single edge node, located one hop from the clients/devices. OpenNebula is an ideal candidate for managing a disaggregated cloud because it is functional, simple, flexible, and light.

For the workload, we consider a generic IoT scenario such as the one shown in Figure 1. We assume that the application exhibits a typical performance behavior, with each device, on average, generating a request every 10 seconds and each request consuming 10 ms of CPU and 8 ms of I/O. Figure 2a shows the response time (T_R) of a single application server as a function of the server load, represented by the number of devices accessing to the server. Furthermore, we also assume that the number of devices increases linearly from 100 and 2,500, at a rate of 20 new devices per minute. We used VMs as the virtualization technology, however, the same mechanisms can also be used for system containers.

We analyzed and compared the following scenarios:

- **Classic cloud:** No edge resources were considered and all the service components were deployed within the on-premises cloud infrastructure.
- **Distributed edge:** Each edge node was managed by its own Cloud OS and service components were deployed within the edge node closest to the devices.
- **Disaggregated cloud:** The edge nodes were managed by a central Cloud OS and the service components were deployed in the edge node or cloud site closest to the devices.

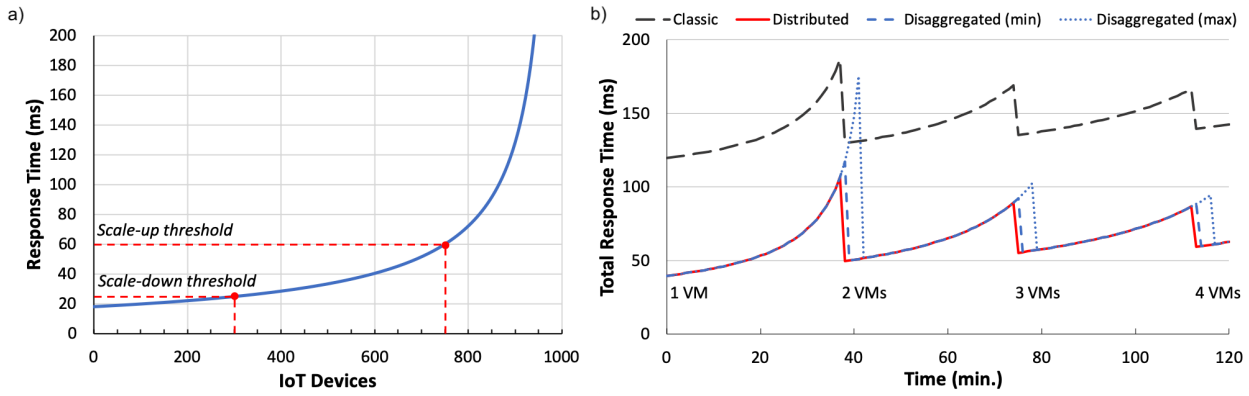

Figure 2. a) Performance profile of a single application server. b) Total response time.

The following parameters were used in these scenarios:

- **Network latency (T_{NL}).** We assumed that there is a 50 ms network latency between the devices and the computing resources for the on-premises cloud (classic cloud) and 10 ms for the edge node (distributed edge and disaggregated cloud). These values were experimentally obtained from different cloud and edge providers (AWS, Azure, and Packet).
- **Auto-scaling metric and thresholds.** To implement the threshold-based auto-scaling mechanism, we used the T_R as the service-performance metric. The scale-up and scale-down thresholds were set to 60 ms and 25 ms respectively, as shown in Figure 2a.
- **Monitoring interval (T_{MI}).** The monitoring interval is higher in the disaggregated cloud, where the Cloud OS must monitor both the local resources and a variable number of remote edge resources. We assumed a 1-minute monitoring interval for the classic cloud and distributed edge setup, and a 2 to 5-minute interval for the disaggregated cloud setup, depending on its size.
- **Auto-scaling sustained period (T_{SP}).** To avoid unnecessary fluctuations in the number of resources assigned to the service, the condition that triggers an auto-scaling action (i.e., when the monitoring metric exceeds the established threshold) must be met for a sustained period; in this case, we set this period to 3 minutes.
- **VM start-up time (T_{VS}).** This parameter depends on multiple factors[11]. In most popular cloud platforms, it takes between 1 and 6 minutes. However, in real tests on Packet edge servers, we measured start-up times of 4 s to 24 s; here, we considered a start-up time of 1 minute.
- **Total auto-scaling delay (T_{AD}).** This represents the total time elapsed between the occurrence of a condition that triggers an auto-scaling action and the provision (or retraction) of a VM. It can be computed as:

  $$T_{AD} = T_{MI} + T_{SP} + T_{VS}$$

  Thus, using the aforementioned values, the total auto-scaling delay was calculated as 5 minutes for the classic cloud and the distributed edge scenarios and between 6 and 9 minutes for the disaggregated cloud scenario.

Considering all these parameters, we ran our experiment for 2 hours; Figure 2b shows the total response time (T_{total}), taking into account both the application response time (T_R) and the network latency (T_{NL}). The peaks in these graphs represent the instants before the provision of a new VM and so, the T_{total} dropped

after these were made available. Because it supports the lowest network latency and auto-scaling delay, the distributed edge had the lowest average T_{total} (63 ms) whereas the disaggregated cloud had a slightly higher average (64.1 ms to 68.5 ms) resulting from its higher auto-scaling delay. Finally, the classic cloud had the highest average T_{total} (143 ms) because of its higher network latency.

CONCLUSION

The disaggregated cloud model proposed in this work represents a good trade-off between performance and design complexity. Compared to the more complex distributed edge architecture based on multiple Cloud OS instances (one per edge node) the disaggregated cloud can manage both on-premises and edge resources with a single Cloud OS instance and with minimal application performance degradation.

ACKNOWLEDGMENTS

This research was supported by Ministerio de Ciencia, Innovación y Universidades through the research grant RTI2018-096465-B-I00, and by Comunidad de Madrid through the Research Program EDGEDATA-CM (P2018/TCS4499).

REFERENCES

1. M. Hung, “Leading the IoT: Gartner Insights on How to Lead in a Connected World,” Gartner Inc., 2017.
2. N. Wang et al., “ENORM: A Framework For Edge NOde Resource Management,” *IEEE Transactions on Services Computing*, In press, 2017.
3. J. Zhu et al., “Improving Web Sites Performance Using Edge Servers in Fog Computing Architecture,” *2013 IEEE Seventh International Symposium on Service-Oriented System Engineering*, pp. 320-323, 2013.
4. A.M. Rahmani et al., “Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things,” *Future Generation Computer Systems*, vol. 78, pp. 641-658, 2018.
5. P. Patel, M. Intizar Ali and A. Sheth, “On Using the Intelligent Edge for IoT Analytics,” *IEEE Intelligent Systems*, vol. 32, no. 5, pp. 64-69, 2017.
6. R. Moreno-Vozmediano, R.S. Montero and I.M. Llorente, “IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures,” *Computer*, vol. 45, no. 12, pp. 65-72, 2012.
7. T. Lorido-Botran, J. Miguel-Alonso and J.A. Lozano, “A review of auto-scaling techniques for elastic applications in cloud environments,” *Journal of Grid Computing*, vol. 12, no. 4, pp. 559–592, 2014.
8. J.S. Ward and A. Barker, “Cloud cover: monitoring large-scale clouds with Varanus,” *Journal of Cloud Computing*, vol. 4, no. 1, 2015.
9. R. Moreno-Vozmediano, R.S. Montero, E. Huedo, I.M. Llorente, *“Orchestrating the Deployment of High Availability Services on Multi-zone and Multi-cloud Scenarios,”* Journal of Grid Computing, vol. 16, no. 1, pp. 39-53, 2018.
10. R. Moreno-Vozmediano, R.S. Montero, E. Huedo, I.M. Llorente, “Cross-Site Virtual Network in Cloud and Fog Computing,” *IEEE Cloud Computing*, vol. 4, no. 2, pp. 46-53, 2017.
11. M. Mao and M. Humphrey, “A Performance Study on the VM Startup Time in the Cloud,” *2012 IEEE Fifth International Conference on Cloud Computing*, pp. 423-430, 2012.

ABOUT THE AUTHORS

Rafael Moreno-Vozmediano is an associate professor at Universidad Complutense de Madrid (UCM). His research interests include distributed computing, edge/cloud computing, and networking. Contact him at rmoreno@ucm.es.

Eduardo Huedo is an associate professor at UCM. His research interests include distributed computing, cloud computing and performance management. Contact him at ehuedo@ucm.es.

Ruben S. Montero is co-founder and chief architect of OpenNebula, and an associate professor at UCM. His research interests include resource provisioning models for distributed systems and cloud computing. Contact him at rubensm@ucm.es.

Ignacio M. Llorente is co-founder and director of OpenNebula, full professor at UCM and visiting professor at Harvard University. His research interests include distributed, parallel, and data-intensive computing technologies, and innovative applications of those technologies to business and scientific problems. He is a senior member of IEEE. Contact him at imllorente@ucm.es.