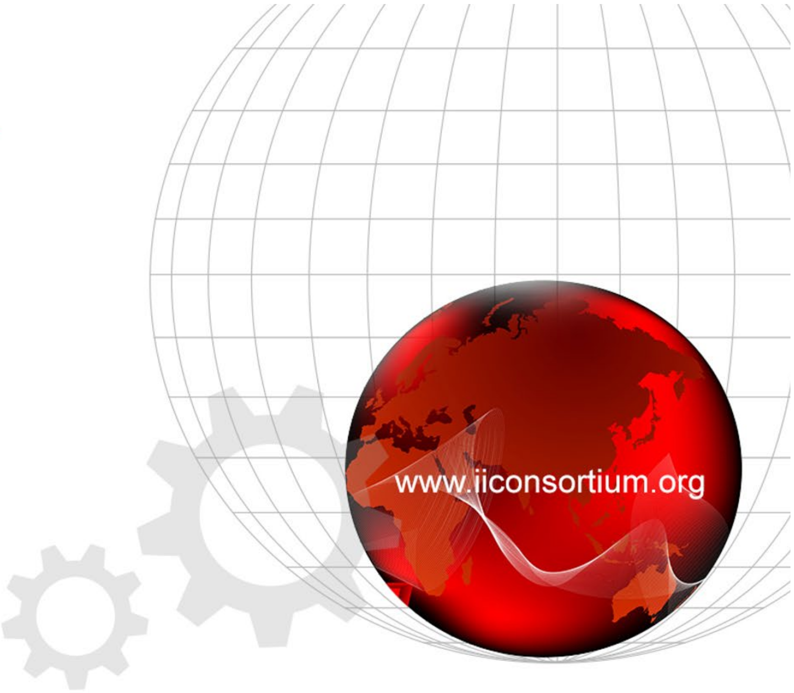


Key Criteria to Move Cloud Workloads to the Edge

Authors:

Charles Byers
Associate Chief Technical Officer
Industrial Internet Consortium
byerschuck1@gmail.com

INTRODUCTION

In today's networks, most compute workloads run in cloud data centers. This is changing, as many critical requirements are not met in the cloud, and a many of these workloads are moving completely or in part to edge computing.

The Industrial Internet Consortium (IIC) defines[1] edge as: "boundary between the pertinent digital and physical entities, delineated by IoT devices". It further defines edge computing as: "distributed computing that is performed near the edge, where the nearness is determined by the system requirements". Basically, edge computing is about taking a carefully selected subset of the computational workloads, storage capabilities, and networking infrastructures typically found in cloud data centers and moving them physically and logically closer to the sensors, actuators, and other IoT devices that generate and use the data.

There are many edge architecture philosophies. The Cloudlet work at Carnegie-Mellon university is one of the earliest examples[2]. Fog computing is another example, as exemplified by the work of the OpenFog Consortium (now part of IIC) and the IEEE 1934 – IEEE Standard for Adoption of OpenFog Reference Architecture for Fog Computing[3]. The European Telecommunications Standards Institute Multi-access Edge Computing (ETSI MEC)[4] is growing in influence. The edge computing architecture variant we will focus on in this paper is described in detail in IIC's "The Industrial Internet of Things Distributed Computing in the Edge" whitepaper [5].

[1] Industrial Internet Consortium, "Industrial Internet of Things Vocabulary Technical Report", OCT. 2020, https://www.iiconsortium.org/stay-informed/vocab.htm

[2] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges", September 2001IEEE Personal Communications 8(4):10 – 17. (PDF) Pervasive Computing: Vision and Challenges (researchgate.net)

[3] IEEE Standards Association, "IEEE 1934-2018 - IEEE Standard for Adoption of OpenFog Reference Architecture for Fog Computing", IEEE 1934-2018 - IEEE Standard for Adoption of OpenFog Reference Architecture for Fog Computing

[4] European Telecommunications Standards Institute, "Multi-access Edge Computing (MEC) standard", ETSI - Multi-access Edge Computing - Standards for MEC

[5] Industrial Internet Consortium, "The Industrial Internet of Things Distributed Computing in the Edge", OCT 2020, IIoT Distributed Computing in the Edge (iiconsortium.org)

EDGE NETWORK ARCHITECTURE

Figure 1 is an overview of edge network architecture.

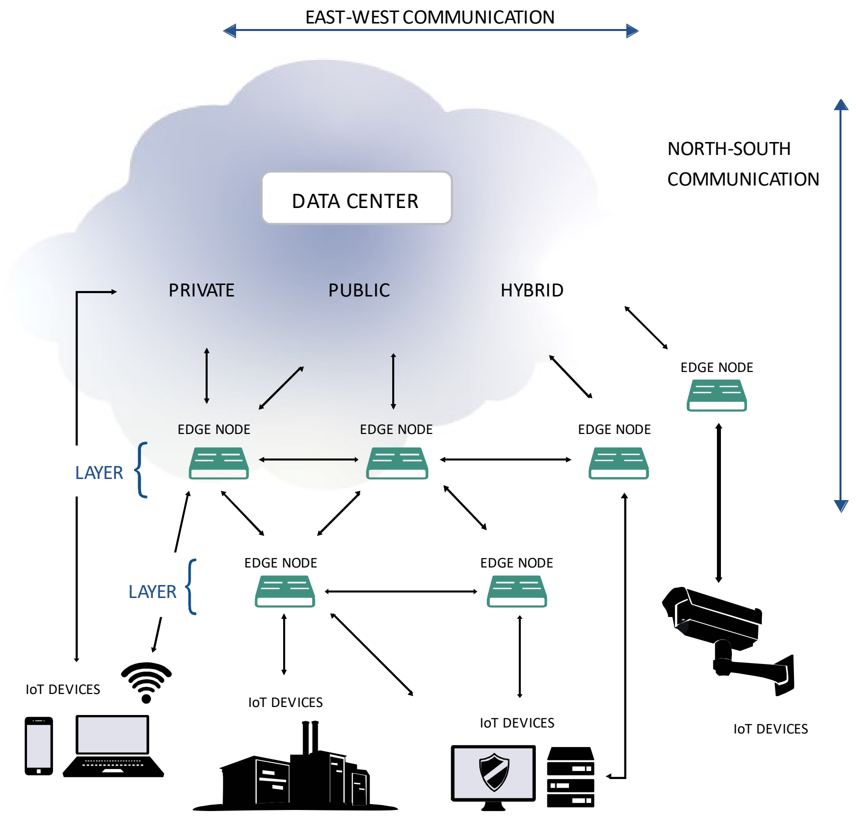
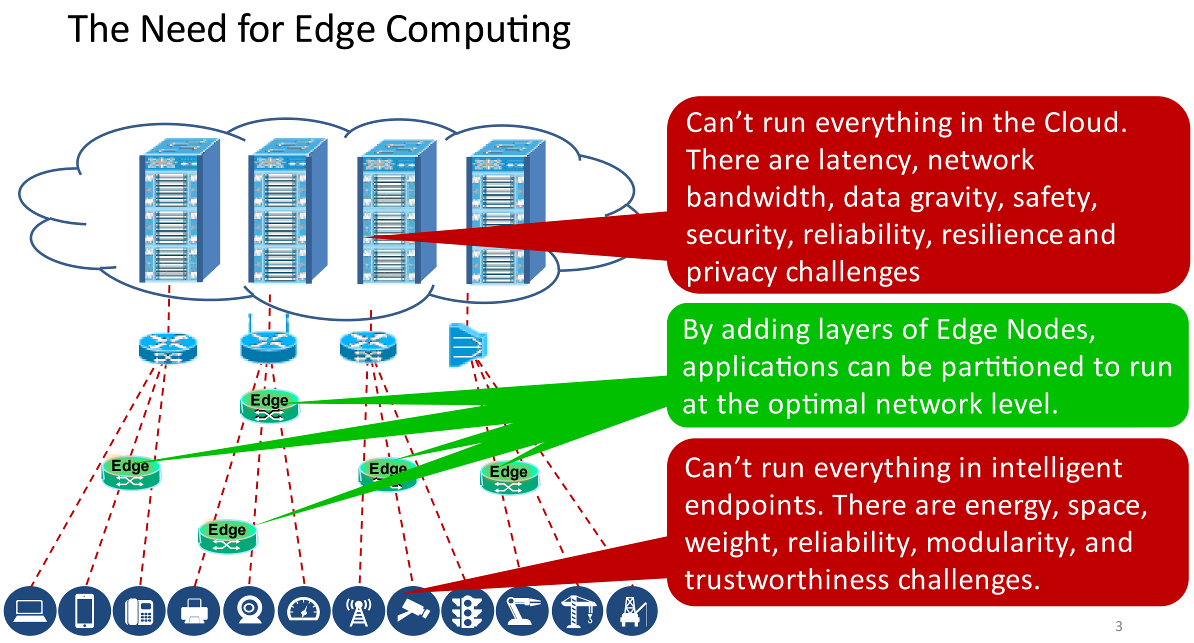


Fig. 1 - Industrial Internet Consortium Distributed Computing in the Edge Framework.

There is a hierarchy of computation, storage, and networking resources in this architecture. The data center implements the cloud. Below that are multiple layers of edge computing. Finally, the IoT devices shown can themselves implement some computation and storage capabilities (for example, an intelligent security camera with built-in video analytics capability). North-south communications links provide interconnect between the layers. East-west communications links interconnect peer devices on the same layer.

The focus of this article is to understand why certain workloads cannot be optimally performed on the cloud or the intelligent IoT devices and must (at least partially) be moved to one or more layers of edge nodes. It discusses the most important selection criteria when deciding to move computational workloads from cloud data centers to edge computing nodes.

Figure 2 is a summary of the key selection criteria to consider when deciding where to locate the processing and storage for specific workloads. The criteria will be discussed in detail in subsequent sections.



The Need for Edge Computing

Can’t run everything in the Cloud. There are latency, network bandwidth, data gravity, safety, security, reliability, resilience and privacy challenges

By adding layers of Edge Nodes, applications can be partitioned to run at the optimal network level.

Can’t run everything in intelligent endpoints. There are energy, space, weight, reliability, modularity, and trustworthiness challenges.

Fig. 2 - Key Selection Criteria for Cloud - Edge Partitioning.

Why can’t we run everything in cloud data centers?

Conventional wisdom holds that cloud data centers are the optimal location to run computational workloads and host large-scale storage. This is indeed true for many applications. However, for an increasing percentage of critical IoT workloads, hosting in cloud data centers can have significant drawbacks. In general, if you can meet all system requirements by hosting an application fully within the cloud, you should, because that architecture will optimize scalability, flexibility, and deployment cost. In practical systems, it is often not a question of “can” or “can’t, but a question of optimization of the implementation along multiple dimensions. This section will focus on some of the reasons why hosting computational workloads and storage exclusively on the cloud may not work for many classes of critical applications.

Latency

Latency is the round-trip time it takes a network to accept some input, perform computation, and produce a valid response based upon it. For IoT applications hosted in the cloud, this latency is the sum of many components. Consider a smart transportation safety use case, where a camera monitors an intersection, detects pedestrians in the traffic lanes, and attempts to apply the anti-lock brakes of approaching connected vehicles to avoid vehicle-pedestrian collisions. The first step is to photograph the crosswalk with a connected camera. That camera has certain frame rate (or in general, a sensor has a sample interval or integration time) that adds latency to the system. A 30 frame per second camera could introduce a latency of 33ms. Next, a frame of the sensor data must be packaged for transmission into the network, which could involve compression. This could add another frame (or possibly more) of latency, for an additional 33ms.

Next, the data is transmitted into a local access network, and sent to an internet point of presence. This is relatively instantaneous if the connection is completed with metro optical fiber, but in the worst case, 4G / LTE cellular network connections can add up to 150ms round trip[6]. When the data leaves the local access / wireless network and enters a long-haul fiber, it is routed on an intra-city network to the selected cloud data center (which can be thousands of km away). Light in optical fibers travels approximately 68% of the speed of light in a vacuum, so for each 1000km of distance between the IoT device and the cloud data center processing its data, a round trip delay of about 10ms[7].

Then, there are queuing and software scheduling and execution delays in the cloud data center's routers and servers which are highly variable but could easily contribute an additional 50ms. After the cloud acts upon the data and decides which action must be taken with an oncoming vehicle, a message is constructed, sent back to the approaching vehicle, and its control computers apply the brakes as commanded (with minimal latency).

All told, this architecture could have a round-trip sense-compute-actuate latency of almost 300ms. A vehicle approaching a pedestrian at 100km/hour travels about 8m during this 300ms interval getting that much closer to a collision, so one can appreciate the safety reduction for those pedestrians the 300ms latency introduces.

Let us explore how edge computing could improve this situation. If instead of sending compressed video to the cloud for analysis, incurring the 4G, fiber and cloud queueing delays, we can locate an edge computer right at the intersection capable of performing the same analytics operations and pedestrian safety application. There is no need for compression, because the camera can be directly connected by a cable to the edge node.

We can increase the frame rate (perhaps to 240FPS) because the bandwidth on this direct cable is basically free. There is no need for the high latency 4G or long-haul fiber connections. Cloud routing and queueing delays are transformed to edge node queueing delays, which we can exercise much tighter control over. A dedicated DSRC radio (which can have sub millisecond latency) connects the edge node with the oncoming vehicle. Under this scenario the latency picture is much improved: about 4ms for video frame latency, basically zero for all the

[6] 3G/4G wireless network latency: Comparing Verizon, AT&T, Sprint and T-Mobile in February 2014 | FierceWireless

[7] Calculating Optical Fiber Latency (m2optics.com)

communication links, and on the order of 10ms for edge node queueing and processing delays, for a grand total of about 15ms round-trip sense-compute-actuate latency (a twenty-fold improvement). So, instead of our 100Km/hour vehicle getting 8M closer to a dangerous collision due to system latency, this scenario allows it to only approach about 40cm before the breaks are applied. The increased safety of the lower latency edge computing-based architecture should be obvious.

Other use cases are equally latency critical. Augmented reality / virtual reality applications often compute eye views in the network to reduce the amount of computation hardware needed on the goggles. High network latency can create time lags between head motion sensing and video rendering that induce nausea in some individuals. In haptics (e.g., tactical feedback joysticks) networks where the force feedback is calculated on networked computing, latency exceeding about 15ms can negatively impact the illusion of touch[8]. By moving the computation associated with these critical applications from cloud data centers to edge nodes near the endpoints they serve, the user experience can be greatly improved.

Workloads in closed-loop industrial control systems (applications like robotics, welding, printing, etc.) can be even more latency critical, some requiring less than 1ms round-trip delay.

Workloads being considered for cloud data center execution should be carefully evaluated for their latency requirements. If the latency requirements are beyond what cloud data centers can reasonably deliver, those workloads (or at least their latency-sensitive subcomponents) should move to edge computing nodes.

Network Bandwidth

Network bandwidth refers to the peak or average data rates or data set sizes on the various wireless and wireline interconnect links in the IoT network architecture. A functional partitioning of IoT applications where the processing is all done in cloud data centers results in large datasets being transferred, or high streaming bandwidth between the IoT devices and cloud. This bandwidth can be quite costly, in terms of the network charges for the service use, and also in terms of its impact on other applications that share the same interconnect networks.

Consider a use case where a high bandwidth sensor needs to transport its entire data stream continuously to a cloud data center for analysis, for example, a 4K-resolution surveillance camera

[8] Rank M., Shi Z., Müller H.J., Hirche S. (2010) The Influence of Different Haptic Environments on Time Delay Discrimination in Force Feedback. In: Kappers A.M.L., van Erp J.B.F., Bergmann Tiest W.M., van der Helm F.C.T. (eds) Haptics: Generating and Perceiving Tangible Sensations. EuroHaptics 2010. Lecture Notes in Computer Science, vol. 6191. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-14064-8_30

in a remote location. A UHD camera (3840 x 2160 resolution) running at 30FPs and using the H.265 HVEC codec with high quality settings consumes a network bandwidth of 13.4Mb/s[9]. At this rate, this flow would transmit a gigabyte of data to the cloud in about 10 minutes, or about 144GB/day. Some cellular networks charge $10 per 50GB of data overage. So, at that incremental rate, the cost of hauling this one camera feed to the cloud for processing would be over $800 per month. Obviously, running the video analytics in the cloud is cost prohibitive because of the cost of the network bandwidth consumed. Some other access modes (satellite internet, for example) can have even higher bandwidth cost.

Excessive network bandwidth use has more problems associated with it beyond the monthly cost of the bandwidth. It also overloads networks and radio spectrum to the point where other users of the network experience delays or service reliability problems. If many high bandwidth endpoints are located close to each other, local network congestion may drastically slow down the networks, or prevent additional users from successfully connecting.

Moving the analytics algorithms from cloud data centers to edge computing nodes largely eliminates the need to send this high bandwidth traffic across wide area networks, saving those large bandwidth-related charges and preventing capacity issues. A local edge node is directly connected to the camera in the above surveillance example with a cable or short-range unlicensed wireless link that does not create monthly bills and performs the image analysis very near the camera. Then, only the results of that analytics ("This camera didn't detect any intruders during the last minute"), which are orders of magnitude smaller in bandwidth can be sent to the cloud for action.

Data Gravity

Data gravity is the property of networks where certain datasets are optimally stored or processed on specific network nodes. The preference for data location can be due to many factors, including performance, geographic considerations, user policies, and government regulation. If all data must be processed and stored in the cloud, challenges can arise.

As a concrete example, consider wearable devices that record the medical vital signs from warfighters, and transfer that data to a command, control, communications, computer, and intelligence (C4I) networks. There are certain data gravity considerations associated with this system. The data is most useful to the local chain of command near its source and becomes diminishingly less valuable as the geographic distance between the wearable device and the data processing or storage location increases. It would make very little sense to move this sort of data to a cloud server thousands of kilometers away for processing and storage. There may be policies against transporting this data across certain boundaries (outside a base, outside a theater of

[9] Bandwidth calculator | CCTV Calculator

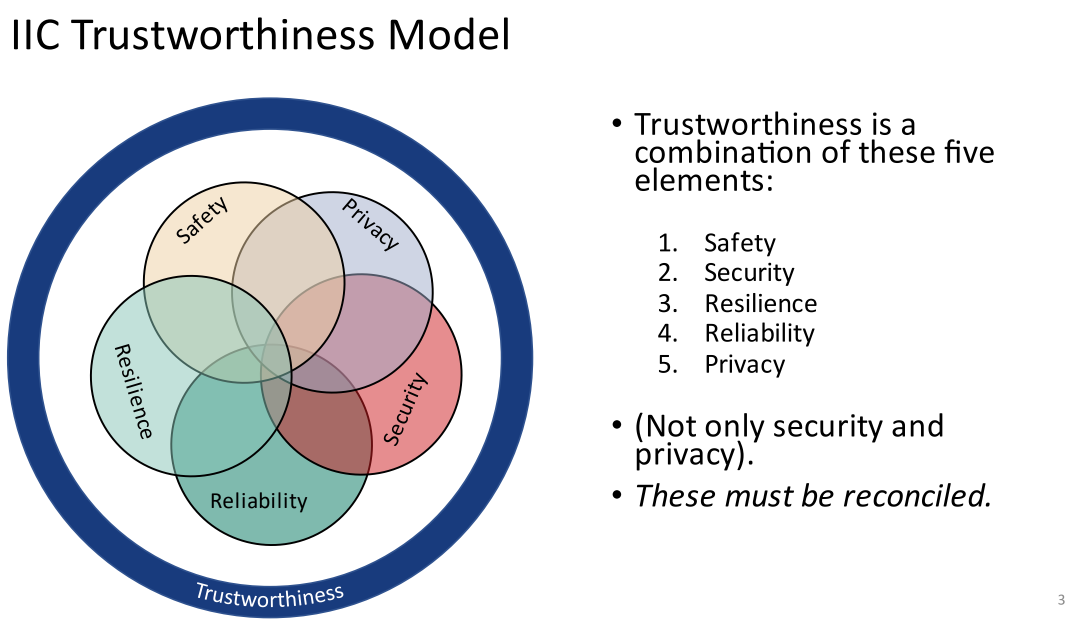
operations, for example). Finally, adversaries would find this data very valuable, and care must be taken to prevent its unintended interception by them, either as data in motion or data at rest.

Edge computing can help match the processing and storage location used by a piece of data to its data gravity preferences. In our wearable for the warfighter example, an edge node in a backpack, or on a Humvee, or in an aircraft cockpit, or at a forward fire base can accept data from all the wearables used by all the warfighters in the area, process it to filter and apply AI algorithms, and store it locally. Then, interesting subsets of the data that was stored and processed in the edge node can be responded to instantly, and appropriately condensed, aggregated, anonymized, and encrypted for safe, efficient transmission to higher layers of edge nodes or the cloud as needed.

Trustworthiness

The Industrial Internet Consortium defines trustworthiness of IoT networks as the conjunction of five properties: safety, security, reliability, resilience, and privacy (this topic was treated extensively in the September 2018 edition of this publication[10]). As we will see, relying exclusively on cloud data centers to store and process critical data can present challenges in all five of these aspects of trustworthiness, and moving at least a portion of the workloads to edge computing can improve the situation. Figure 3 is a graphical depiction of the five overlapping aspects of trustworthiness.

[10] Journal of Innovation - September 2018 | Industrial Internet Consortium (iiconsortium.org)



IIC Trustworthiness Model

- Trustworthiness is a combination of these five elements:

1. Safety
2. Security
3. Resilience
4. Reliability
5. Privacy

- (Not only security and privacy).
- *These must be reconciled.*

3

Fig. 3 - Industrial Internet Consortium Trustworthiness Model

Safety critical systems often experience challenges with cloud-based processing. Cloud data centers are occasionally down, unreachable, or unacceptably slow due to outages, network problems, hacker attacks, overloads, or disasters. If an application is being depended upon to keep people safe (for example, anti-collision systems in autonomous vehicles, elevator controls, or worker safety monitors in smart manufacturing), relying exclusively on cloud-based processing will be unacceptably risky. By moving the most safety critical aspects of these applications to edge computing nodes located physically close to where the data is being generated and used and maintaining careful control over the configuration and load of those edge nodes, the network can be relied upon to perform adequately to keep people safe.

Security is an overarching concern in IoT systems. There are many aspects to network security that have been discussed extensively in many publications, including IIC's Industrial Internet Security Framework Technical Document[11]. Critical IoT applications run exclusively in cloud data centers have certain security vulnerabilities that are often difficult to eliminate. These include unauthorized interception of data (either in motion or at rest), authorization / authentication failures, compromises to cryptographic systems, unauthorized changes to system configurations or policies, overwriting data, hacking software, etc. Because of its public, shared, and remote nature, there are many vectors for these security threats to enter the cloud and compromise the

[11] Industrial Internet Security Framework | Industrial Internet Consortium (iiconsortium.org)

IoT services. Moving critical portions of the computational workloads and storage to edge computing nodes can eliminate some of these vulnerabilities (perhaps at the expense of introducing a few new ones). Incidents of security compromise can often be easier to detect, locate, isolate, repair and restore on edge nodes than cloud data centers.

Reliability is a key aspect of trustworthiness, especially for mission-critical or life-critical workloads. The cloud is often difficult to monitor, manage, and recover after faults. Because of the huge scale of efficient cloud data centers, a site-wide outage can impact millions of application instances. Duplication and redundancy are often difficult in these high-scale cloud networks, because of the huge amount of inter-site traffic necessary to ensure that the contexts of redundant elements are geographically distributed across diverse data centers and keep the cloud-distributed databases consistently updated.

Cloud data centers are also susceptible to outages in the data links that interconnect them with the IoT devices and with their peer data centers. Edge techniques can add another dimension to reliability by distributing computation workloads and storage instances across multiple edge nodes that are still relatively physically close to each other and to the IoT devices. Edge nodes themselves can be designed to be fault tolerant, with duplicated processing, storage and I/O modules insulating system availability from single point failures. That highly reliable hardware-based fault tolerance architecture is impossible to achieve in the cloud using commodity servers.

Resilience is the property of systems to continue operation within spec even in the presence of abnormal conditions. Some cloud-based architectures are pretty brittle, that is a single, relatively small failure, overload or overflow can have a large impact on the operation of the system. A single power outage, fiber cable cut, primary router failure, or natural disaster can destroy a data center's ability to process computational loads.

Modern data center architectures do add some redundancy to the power and data networking infrastructure that supports their servers, but multiple data centers spread out by considerable distances as multiple availability zones[12] are required to provide adequate resiliency for many critical IoT applications. Edge techniques enhance resilience by providing multiple edge nodes, any one of which is capable of providing full service. Edge nodes are often arranged in multiple layers, and computational loads can be moved to an adjacent layer via the north-south links if a node on one layer fails or becomes overloaded. Further, the east-west links that interconnect edge nodes on a given layer can move data between peer edge nodes, providing resilience in the face of single node failures or localized overloads.

Privacy is the final aspect of trustworthiness. There are certain concerns with privacy unique to the cloud, especially if the cloud service is hosted by a web-scale company with significant financial stake in understanding the patterns of your data. On a public data center that is shared

[12] Regions, Availability Zones, and Local Zones - Amazon Relational Database Service

with potentially thousands of other application clients, there is always the risk that private data may be disclosed, either unintentionally or through deliberate hacking attacks, to someone not entitled to receive it. Privacy is especially critical for personally identifiable data, or healthcare data covered by HIPPA and similar laws. Since edge computers are more distributed, more local, used by fewer tenants, and generally smaller than major cloud data centers, these privacy concerns can be reduced using edge techniques.

Taken as a whole, these five aspects of trustworthiness require careful attention to system architectural considerations, and deliberate partitioning of workloads between cloud data centers and edge nodes to optimize the system.

Why can't we run everything in intelligent IoT devices / endpoints?

Let us investigate the converse of running everything in cloud data centers, namely running the computation and hosting the storage functions directly on intelligent IoT devices, without significant contributions from the cloud or edge. This approach would seem to have promise in terms of attributes such as latency, network bandwidth, and scalability. But, as shown in the following discussion, there are some significant drawbacks to the intelligent IoT device approach.

Energy

Many IoT devices are energy constrained. Edge computing applications, especially those making heavy use of video analytics or AI, can require significant power dissipation in their processors and related hardware. Many classes of IoT devices are expected to run for years on reasonably sized batteries. Certain IoT devices would cause problems if they dissipated excessive heat to the environment, especially if they required noisy, failure prone cooling fans. So, many classes of IoT devices simply cannot dissipate more than a few watts of electrical power, and that severely limits the sophistication of the processing they can perform.

By moving the energy-intensive portions of the processor workload from the IoT devices to the layer of edge computing immediately adjacent to them, we can offload the high-power dissipation from a highly energy and heat constrained IoT device to an edge computing node without those constraints. These edge computing nodes can support multiple instances of different types of high-capacity computational resources, such as CPUs, GPUs, TPUs, specialized accelerators or FPGAs, some with the equivalent of tens of thousands of processor cores[13].

Edge nodes can dissipate thousands of watts of power and be cooled by advanced forced air or liquid cooling systems. Since these edge computers are connected to the electrical grid, and often have backup power sources, they are not constrained by batteries as many IoT devices are.

[13] See the "Heterogeneous Computing in the Edge" article in this issue.

Space / Weight

Space / Weight is another key constraint preventing us from hosting advanced computation workloads in many classes of IoT devices. A 1U X86 server-class computer (a typical processor infrastructure for many IoT applications) has a volume of about 12 liters, while many IoT devices are 1 liter or less. A 1U server may weigh about 12kg, where many IoT devices weigh hundreds of grams. You can see, if the computational workloads require processing power similar to that standard 1U server, the weight and size supported by a typical IoT device is totally inadequate. By moving a subset of the high-performance computation from the IoT device to an edge computer, we can remove many of the size and weight constraints, and provide much higher computational performance for the service.

Environmental Considerations

IoT devices often must survive harsh environmental conditions. Extended industrial temperature ranges of -40°C to +85°C are often encountered, especially in outdoor applications, and many types of hardware grow very expensive or are impossible to implement over this wide temperature range. This is especially true of high-power CPUs, rotating disk drives, and optical interfaces. IoT devices often must survive other environmental extremes, such as the humidity, pressure, shock, contamination, vibration, and other environmental factors, as specified in standards such as MIL-STD-810. Designing high performance computation and storage hardware to survive these extremes is very expensive. Edge computing can be located in a somewhat more protected environment, so systems can be optimized if the environmentally sensitive electronics are moved from the IoT devices to the protected edge computers.

Modularity

Modularity is another concern for many types of IoT devices. They are often not designed for reconfiguration, expansion, update, or repair. For example, to double the memory size in a commodity IoT device like a webcam, it is more cost effective to replace the entire device than to upgrade it. Edge computers can be much more modular, expandable, and configurable, allowing different computational, storage and I/O interface modules to be included as required by the specific workloads they support, and also facilitating their expansion as system requirements evolve over time. This improves the total cost of ownership of the system, allows relatively simple edge devices to continue in service much longer, and provides an evolution path for new services.

Trustworthiness

Some of the aspects of trustworthiness we discussed in conjunction with cloud-based workloads also apply to workloads run in IoT devices. For example, IoT devices may not have the energy or processing power to run strong cryptography, compromising privacy and security. IoT devices are often exposed to physical attacks, including stealing, or destroying the device. IoT devices are usually implemented with lower cost hardware and no redundancy, compromising reliability and resilience.

Considered together, certain simple portions of the IoT workload can be successfully run on constrained IoT endpoint devices. However, as the requirements on these networks become more advanced, it is clear that many IoT devices will be extremely underpowered due to the above constraints, and we need to consider moving a significant portion of their workload to layers of edge computing.

Using These Criteria to Partition Workloads to the Edge

The above discussions should demonstrate that many IoT workloads will not perform adequately if run in cloud data centers or on intelligent IoT devices. An intelligent partitioning must be performed to decide which workloads, or sub-functions of workloads are optimally served on edge nodes. This does not mean that all workloads or computational sub-functions must move to the edge – it means that each workload or subfunction should be located on the level of the cloud-edge-IoT device hierarchy where it is most optimally executed. Figure 4 is a process flow that can assist in partitioning workloads between the cloud and edge computing.

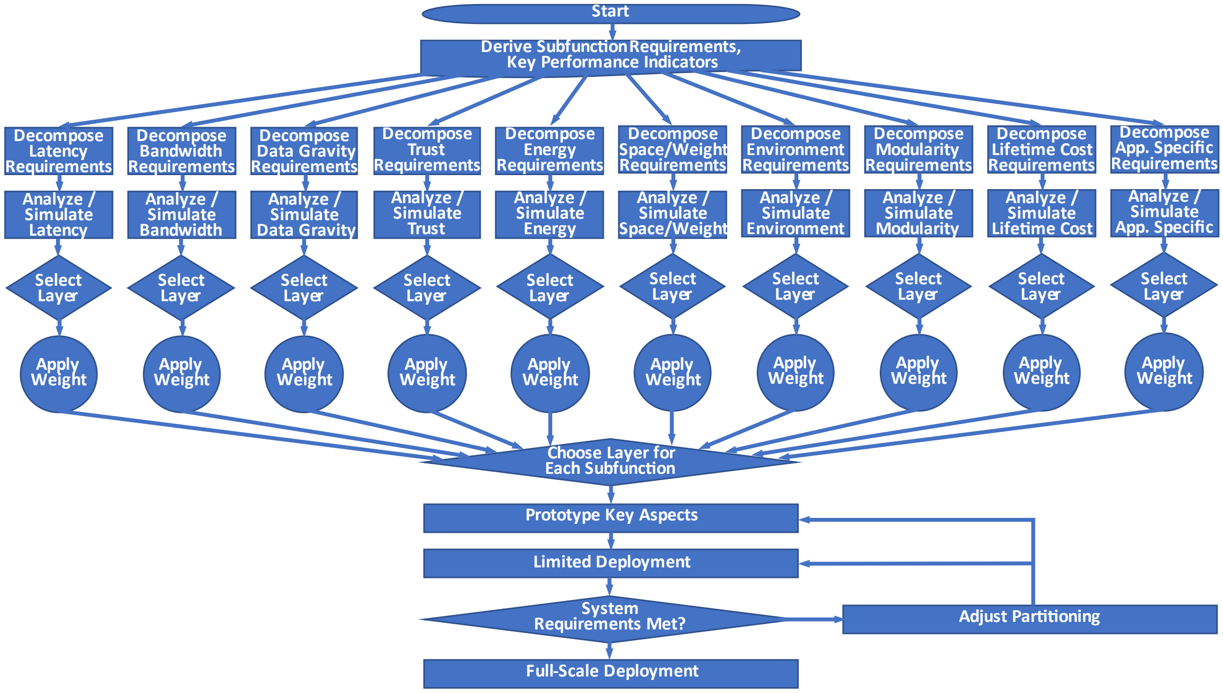

Start
Derive Subfunction Requirements, Key Performance Indicators
Decompose Latency Requirements
Decompose Bandwidth Requirements
Decompose Data Gravity Requirements
Decompose Trust Requirements
Decompose Energy Requirements
Decompose Space/Weight Requirements
Decompose Environment Requirements
Decompose Modularity Requirements
Decompose Lifetime Cost Requirements
Decompose App. Specific Requirements
Analyze / Simulate Latency
Analyze / Simulate Bandwidth
Analyze / Simulate Data Gravity
Analyze / Simulate Trust
Analyze / Simulate Energy
Analyze / Simulate Space/Weight
Analyze / Simulate Environment
Analyze / Simulate Modularity
Analyze / Simulate Lifetime Cost
Analyze / Simulate App. Specific
Select Layer
Apply Weight
Choose Layer for Each Subfunction
Prototype Key Aspects
Limited Deployment
System Requirements Met?
Adjust Partitioning
Full-Scale Deployment

Fig. 4 - Process for Partitioning Workloads Between Cloud and Edge Computing.

The process for partitioning workloads begins with system requirements. Careful attention must be paid to performance-related requirements such as latency, bandwidth, throughput, capacity, etc. Trustworthiness-related requirements are important too, as the safety, security, reliability, resilience, and privacy needs of the applications will have a strong influence on these partitioning decisions. Finally, various measures of system lifetime cost (including purchase price, programming / configuration / evolution costs, energy, and ongoing operational costs) must be factored in to determine how edge solutions can optimize the overall deployment.

The flow on Figure 4 includes a set of decomposition steps, where specific requirements criteria (latency, bandwidth, data gravity, trustworthiness, energy, space / weight, environment, modularity, and lifetime cost) are split out for individual consideration in parallel. There is also an extra category for application specific requirements that may not be included in the aforementioned criteria, but are nonetheless important to the success of a specific system.

Analysis and simulation tools are applied to each of the criteria individually, for example, to determine the performance or efficiency if that subfunction is implemented in the cloud, at some layer of the edge or in the intelligent IoT device. Based upon this analysis and simulation, an optimal implementation layer is selected for the subfunction in each of the named criteria.

Often, a specific subfunction of a network will be optimized in some layer (cloud, edge, intelligent IoT device) based upon the analysis / simulation for one of the criteria, but it is optimized in a different layer for some different criteria. This is where the weighting shown in Figure 4 comes in. Weights (derived from the system requirements) point to which of the criteria should receive higher emphasis, and in places where the criteria indicate different cloud-edge-device partitioning for the same subfunction, the weighting helps referee the discrepancy.

Prototyping is a valuable way to understand application behaviors and adjust preliminary partitioning decisions accordingly. By prototyping key aspects of an application (inner processing loops, for example), one can determine if they will operate adequately in the resources of edge nodes, if the analysis and simulation steps yielded accurate results, and what sort of tradeoffs may be involved moving sub-functions from cloud data centers to edge nodes.

Limited deployment of the final application is the best indicator of the validity of preliminary partitioning decisions. Several different partitioning models of various elements of a complex application between the cloud and edge nodes will allow you to experiment and analyze the performance differences and make an intelligent decision on optimal partitioning before full-scale roll-out. This is also where the initial deployment and ongoing operational cost structures will be adequately understood.

A final check is made of the limited deployment to determine if all system requirements are met. If not, adjustments to the cloud – edge – IoT device algorithm partitioning can be made, and a subset of the previous steps in this process can be repeated. Once all requirements are satisfied, the architecture is ready for full-scale deployment.

Let’s look at a concrete example applying the techniques in Figure 4 : a video surveillance system for a medium-sized airport. Each gate has a number of intelligent cameras, interconnected to edge nodes and a set of cloud servers. The algorithm can be partitioned into a set of sequential sub-functions, including the steps of: contrast enhancement, feature extraction, object recognition, multi-camera correlation, threat detection, automated response, and supervision. The challenge is to partition each of those sub-functions optimally into the cloud, edge or intelligent IoT device hosted computational resources. The requirements are decomposed into the ten criteria shown, and analysis or simulation is performed to measure the performance of each if implemented in the cloud, edge, or intelligent device.

We may discover, for example, that the feature extraction subfunction has the best latency if performed at the edge, but it may have a lower cost if implemented in the cloud. Weights will be applied, and the decision of which layer of computational resources represents the best compromise for that subfunction is made. The entire process is repeated for the remaining sub-functions. This generates a straw proposal for the full system partitioning, defining which sub-functions will reside in the cloud, in one or more layers of edge nodes, or in the intelligent IoT devices. At that point, the entire system is verified with a process of prototyping and limited deployment, iterating and adjusting the subfunction partitioning as required until all system requirements are met, and full-scale deployment can begin.

Finally, the partitioning between cloud, edge and IoT device execution for an initial deployment can be modified as more system experience is gained. Some edge orchestration systems use containers like Kubernetes or Docker to support their workloads, and these systems can dynamically move parts of the algorithms between levels of the network in response to changing load profiles or fault events. AI techniques are being applied to these edge orchestrators[14], so repartitioning in response to changing workloads can be at least partially automated and could potentially react on sub-one second timescales as system loads change.

CONCLUSIONS

By taking a fresh, focused look at the key performance indicators and system-level requirements of networks, it is possible to optimize the performance, trustworthiness, and lifecycle cost of applications by segmenting workloads between cloud data centers and execution on edge nodes. If the partitioning of computational workloads and storage operations between cloud data centers, edge computing nodes and intelligent devices is carefully considered, IoT networks will be better able to service their critical applications.

[14] Y. Wu, "Cloud-Edge Orchestration for the Internet-of-Things: Architecture and AI-Powered Data Processing," in *IEEE Internet of Things Journal*, doi: 10.1109/JIOT.2020.3014845. Cloud-Edge Orchestration for the Internet-of-Things: Architecture and AI-Powered Data Processing | IEEE Journals & Magazine | IEEE Xplore

ACKNOWLEDGEMENTS

The views expressed in the *IIC Journal of Innovation* are the contributing authors' views and do not necessarily represent the views of their respective employers nor those of the Industrial Internet Consortium.

➢ Return to IIC Journal of Innovation landing page for more articles and past editions