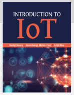
IoT Processing Topologies and Types

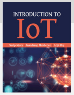
Introduction to IoT

S. Misra, A. Mukherjee, A. Roy

Overview

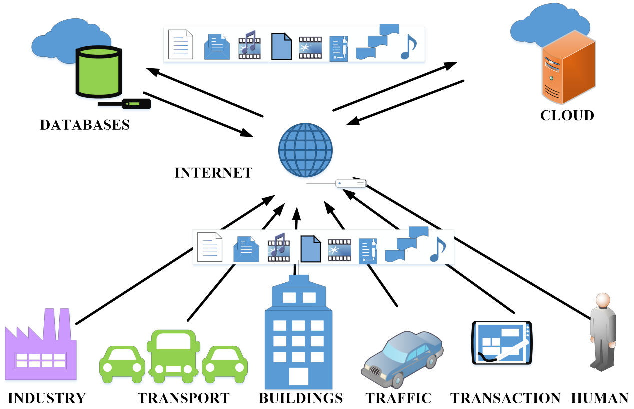


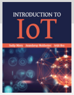
Data Formats



- The Internet is a vast space where huge quantities and varieties of data are generated regularly and flow freely.
- As of January 2018, there are a reported 4.021 billion Internet users worldwide.
- The massive volume of data generated by this huge number of users is further enhanced by the use of multiple devices by most of the users. In addition to these data-generating sources, non-human data generation sources such as sensor nodes and automated monitoring systems further add to the data load on the Internet.
- This huge data volume contained on the Internet is composed of a variety of data such as emails, text documents (Word docs, PDFs, and others), social media posts, videos, audio files, and images.
- However, these data can be broadly grouped into two types based on how they can be accessed and stored – 1) Structured data, and 2) Unstructured data.

IoT Data Sources and Formats

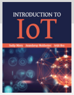




Structured Data

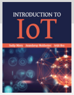
- These are typically text data, which have a pre-defined structure.
- Structured data are associated with relational database management systems (RDBMS).
- These are primarily created by using length-limited data fields such as phone numbers, social security numbers, and other such information.
- Even if the data is human or machine-generated, these data are easily searchable by querying algorithms as well as human-generated queries.
- Common usage of this type of data is associated with flight or train reservation systems, banking systems, inventory controls, and other such systems.
- Established languages such as Structured Query Language (SQL) are used for accessing these data in RDBMS. However, in the context of IoT, structured data holds a minority share of the total generated data over the Internet.

Unstructured Data

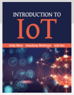


- Simply stating, all the data on the Internet, which is not structured, is categorized as unstructured.
- These data types have no pre-defined structure and can vary according to applications and data-generating sources.
- Some of the common examples of human-generated unstructured data include text, emails, videos, images, phone recordings, chats, and others.
- Whereas, some common examples of machine-generated unstructured data include sensor data from traffic, buildings, industries, satellite imagery, surveillance videos, and others.
- As already evident from its examples, this data type does not have fixed formats associated with it, which makes it very difficult for querying algorithms to perform a look-up.
- Querying languages such as NoSQL are generally used for this data type.

Importance of Processing



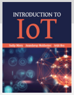
- The vast amount and types of data flowing through the Internet necessitate the need for intelligent and resourceful processing techniques.
- This necessity has become even more crucial with the rapid advancements in IoT, which is laying down enormous pressure on the existing network infrastructure globally.
- Given these urgencies, it is important to decide – *when to process and what to process*?
- Before deciding upon the processing to pursue, we first divide the data to be processed into three types based on the urgency of processing – 1) very time-critical, 2) time-critical, and 3) normal.



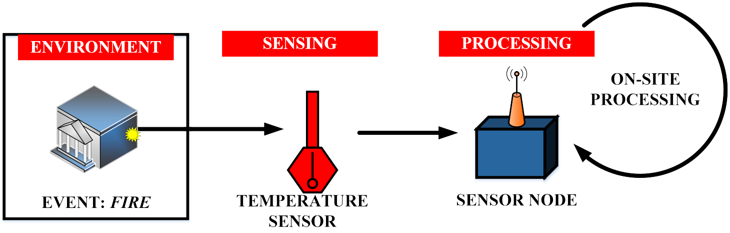
Processing Topologies

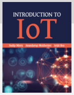
- The identification and intelligent selection of processing requirement of an IoT application are one of the crucial steps in deciding the architecture of the deployment.
- A properly designed IoT architecture would result in massive savings in network bandwidth and conserve significant amounts of overall energy in the architecture while providing the proper and allowable processing latencies for the solutions associated with the architecture.
- Regarding the importance of processing in IoT, we can divide the various processing solutions into two large topologies – 1) on-site, and 2) off-site.
- The off-site processing topology can be further divided into – 1) remote processing, and 2) collaborative processing.

On-site Processing



- As evident from the name, the on-site processing topology signifies that the data is processed at the source itself.
- This is crucial in applications that have a very low tolerance for latencies. These latencies may result from the processing hardware or the network (during transmission of data for processing away from the processor).
- Applications such as those associated with healthcare and flight control systems (real-time systems) have a breakneck data generation rate.
- These additionally show rapid temporal changes, which can be missed (leading to catastrophic damages) unless the processing infrastructure is fast and robust enough to handle such data.

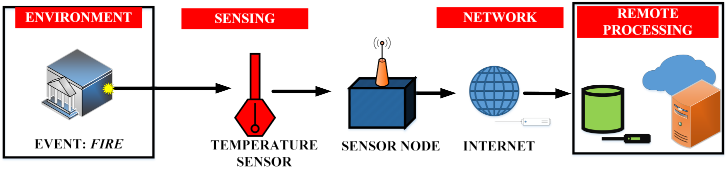

ENVIRONMENT
EVENT: FIRE
SENSING
TEMPERATURE SENSOR
PROCESSING
SENSOR NODE
ON-SITE PROCESSING



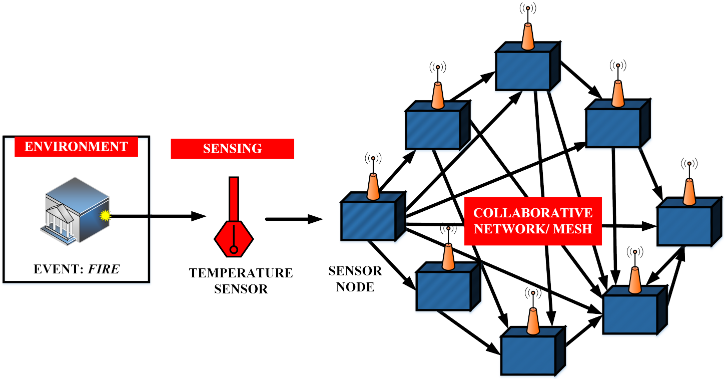
Off-site Processing

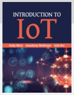
- The off-site processing paradigm, as opposed to the on-site processing, although allows for latencies (due to processing or network latencies) but is significantly cheaper.
- This difference in cost is mainly due to the low demands and requirements of processing at the source itself.
- Often, the sensor nodes are not required to process data on an urgent basis, so having a dedicated and expensive on-site processing infrastructure is not sustainable for large-scale deployments typical of IoT deployments.
- In the off-site processing topology, the sensor node is responsible for the collection and framing of data, which is eventually to be transmitted to another location for processing.
- Unlike the on-site processing topology, the off-site topology has a few dedicated high-processing enabled devices, which can be borrowed by multiple simpler sensor nodes to accomplish their tasks. At the same

Off-site Remote Processing Topology



Off-site Collaborative Processing Topology

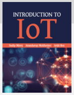




IoT Device Considerations

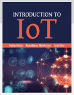
- The main consideration of minutely defining an IoT solution is the selection of the processor for developing the sensing solution (i.e., the sensor node).
- This selection is governed by many parameters, which affect the usability, design, and affordability of the designed IoT sensing and processing solution.
- In this chapter, we mainly focus on the deciding factors for selecting a processor for the design of a sensor node.

Consideration-1: Size

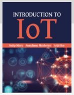


- This is one of the crucial factors for deciding the form-factor and the energy consumption of a sensor node.
- It has been observed that larger the form factor, larger is the energy consumption of hardware.
- Additionally, large form factors are not suitable for a significant bulk of IoT applications, which rely on minimal form factor solutions (e.g., wearables).

Consideration-2: Energy



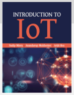
- The energy requirements of a processor is the most important deciding factor in designing IoT-based sensing solutions.
- Higher the energy requirements, higher is the energy source (battery) replacement frequency.
- This principle automatically lowers the long-term sustainability of sensing hardware, especially for IoT-based applications.



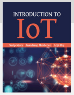
Consideration-3: Cost

- The cost of a processor, besides the cost of sensors, is the driving force in deciding the density of deployment of sensor nodes for IoT-based solutions.
- Cheaper cost of the hardware enables a much higher density of hardware deployment by users of an IoT solution.
- For example, cheaper gas and fire detection solutions would enable users to include much more sensing hardware for a lesser cost.

Consideration-4: Memory



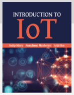
- The memory requirements (both volatile and non-volatile memory) of IoT devices determines the capabilities the device can be armed with.
- Features such as local data processing, data storage, data filtering, data formatting, and a host of other features rely heavily on the memory capabilities of devices.
- However, devices with higher memory tend to be costlier because of the obvious reasons.


INTRODUCTION TO
IoT

Consideration-4: Processing Power

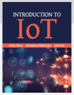
- As covered in earlier sections, processing power is vital (comparable to memory) in deciding what type of sensors can be accommodated with the IoT device/node, and what processing features can integrate on-site with the IoT device.
- The processing power also decides the type of applications the device can be associated with.
- Typically, applications requiring handling of video and image data require IoT devices with higher processing power as compared to applications requiring simple sensing of the environment.

Consideration-5: I/O Rating



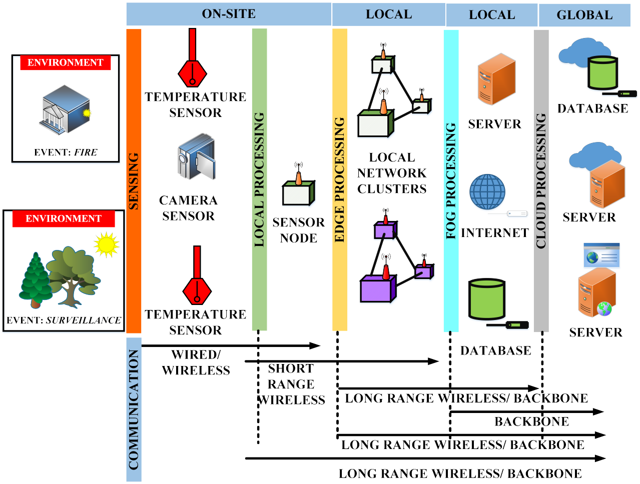
- The input-output (I/O) rating of IoT device, primarily the processor, is the deciding factor in determining the circuit complexity, energy usage, and requirements for support of various sensing solutions and sensor types.
- Newer processors have a meager I/O voltage rating of 3.3 V, as compared to 5V for the somewhat older processors.
- This translates to requiring additional voltage and logic conversion circuitry to interface legacy technologies and sensors with the newer processors.
- Despite low power consumption due to reduced I/O voltage levels, this not only affects the complexity of the circuits but also affects the costs.

Consideration-6: Add-ons

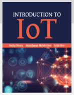


- The support of various add-ons a processor or for that matter, an IoT device provides, such as analog to digital conversion (ADC) units, inbuilt clock circuits, connections to USB and ethernet, inbuilt wireless access capabilities, and others helps in defining the robustness and usability of a processor or IoT device in various application scenarios.
- Additionally, the provision for these add-ons also decides how fast a solution can be developed, especially the hardware part of the whole IoT application.
- As interfacing and integration of systems at the circuit level can be daunting to the uninitiated, the prior presence of these options with the processor makes the processor or device highly lucrative to the users/ developers.

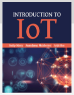
Processing Offloading



Offload Location Types



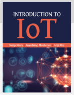
- **Edge**: Offloading processing to the edge implies that the data processing is facilitated to a location at or near the source of data generation itself. Offloading to the edge is done to achieve the aggregation, manipulation, bandwidth reduction, and other data operations directly on an IoT device.
- **Fog**: Fog computing is a decentralized computing infrastructure, which is utilized to conserve network bandwidth, reduce latencies, restrict the amount of data unnecessarily flowing through the Internet, and enable rapid mobility support for IoT devices. The data, compute, storage and applications are shifted to a place between the data source and the cloud resulting in significantly reduced latencies and network bandwidth usage.



Offload Location Types

- **Remote Server**: A simple remote server with good processing power may be used with IoT-based applications to offload the processing from resource-constrained IoT devices. rapid scalability may be an issue with remote servers, and they may be costlier and hard to maintain in comparison to solutions such as the cloud.
- **Cloud**: Cloud computing is a configurable computer system, which can get access to configurable resources, platforms, and high-level services through a shared pool hosted remotely. A cloud is provisioned for processing offloading so that processing resources can be rapidly provisioned with minimal effort over the Internet, which can be accessed globally. Cloud enables massive scalability of solutions as they can enable resource enhancement allocated to a user or solution in an on-demand manner, without the user having to go through the pains of acquiring and configuring new and costly hardware.

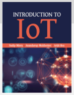
Deciding Offload Requirements



- The choice of where to offload and how much to offload is one of the major deciding factors in the deployment of an offsite-processing topology-based IoT deployment architecture.
- The decision making is generally addressed from considerations of:
    1. Data generation rate
    2. Network bandwidth
    3. Criticality of applications
    4. Processing resource available at the offload site
    5. Other factors.

Offload Decision Making: Naive approach

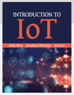
- This approach is typically a hard approach, without too much decision making.
- This can be considered as a rule-based approach in which the data from IoT devices are offloaded to the nearest location based on the achievement of certain offload criteria.
- Although easy to implement, this approach is never recommended, especially for dense deployments, or deployments where the data generation rate is high or the data being offloaded in complex to handle (multimedia or hybrid data types).
- Generally, statistical measures are consulted for generating the rules for offload decision-making.



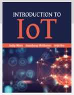
Offload Decision Making: Bargaining-based approach

- This approach, although a bit processing-intensive during the decision making stages, enables the alleviation of network traffic congestion, enhances service QoS parameters such as bandwidth, latencies, and others.
- At times, while trying to maximize multiple parameters for the whole IoT implementation, in order to provide the most optimal solution or QoS, not all parameters can be treated with equal importance.
- Bargaining-based solutions try to maximize the QoS by trying to reach a point where the qualities of certain parameters are reduced, while the others are enhanced.
- This measure is undertaken so that the achieved QoS is collaboratively better for the full implementation rather than a select few devices enjoying very high QoS.
- Game-theory is a common example of bargaining based approach.

Offload Decision Making: Learning-based approach



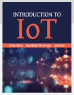
- Unlike the bargaining-based approaches, the learning-based approaches generally rely on past behavior and trends of data flow through the IoT architecture.
- The optimization of QoS parameters is pursued by learning from historical trends and trying to optimize further and enhance the collective behavior of the IoT implementation.
- The memory requirements and processing requirements are high during the decision making stages.
- The most common example of a learning-based approach is machine learning.



Offload Considerations: Bandwidth

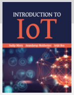
- The maximum amount of data that can be simultaneously transmitted over the network between two points is the bandwidth of that network.
- The bandwidth of a wired or wireless network is also considered to be its data-carrying capacity and often used to describe the data-rate of that network.

Offload Considerations: Latency



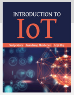
- It is the time delay incurred between the start and completion of an operation.
- In the present context, latency can be due to the network (network latency) or the processor (processing latency).
- In either case, latency arises due to the physical limitations of the infrastructure, which is associated with an operation.
- The operation can be data transfer over a network or processing of a data at a processor.

Offload Considerations: Criticality



- It defines the importance of a task being pursued by an IoT application.
- The more critical a task is, the lesser latency is expected from the IoT solution.
- For example, detection of fires using an IoT solution has higher criticality than detection of agricultural field parameters.
- The former requires a response time in the tune of milliseconds, whereas the latter can be addressed within hours or even days.

Offload Considerations: Resources



- It signifies the actual capabilities of an offload location.
- These capabilities may be the processing power, the suite of analytical algorithms, and others.
- For example, it is futile and wasteful to allocate processing resources reserved for real-time multimedia processing (which are highly energy-intensive and can process and analyze huge volumes of data in a short duration) to scalar data (which can be addressed using nominal resources without wasting much energy).

Offload Considerations: Data volume

- The amount of data generated by a source or sources that can be simultaneously handled by the offload location is referred to as its data volume handling capacity.
- Typically for large and dense IoT deployments, the offload location should be robust enough to address the processing issues related to massive data volumes.

The End