UNIT IV: Solution framework for IoT applications

Implementation of Device integration, Data acquisition and integration, Device data storage- Unstructured data storage on cloud/local server, Authentication, authorization of devices.

4.0 IMPLEMENTATION OF DEVICE INTEGRATION

What Is an IoT Device:-

As described earlier, a 'Thing" in Internet of Things (161) can be any object that has a unique identifier and which can send/receive data (including user data) ova a network (e.g.. smart phone, smart TV, computer, refrigerator, car, etc.). loT devices are connected to the Internet and send information about themselves or about their surroundings (e.g. information sensed by the connected sensors) ova a network (to other devices or servers/storage) or allow actuation upon the physical entities/environment around them remotely. Some examples of loT devices are listed below:

- A home automation device that allows remotely monitoring the status of appliances and controlling the appliances.
- An industrial machine which sends information allows its operation and health monitoring data to a server.
- A car which sends information about its location to a cloud-based service.
- A wireless-enabled wearable device that measures data about a person such as the number of steps walked and sends the data to a cloud-based service.

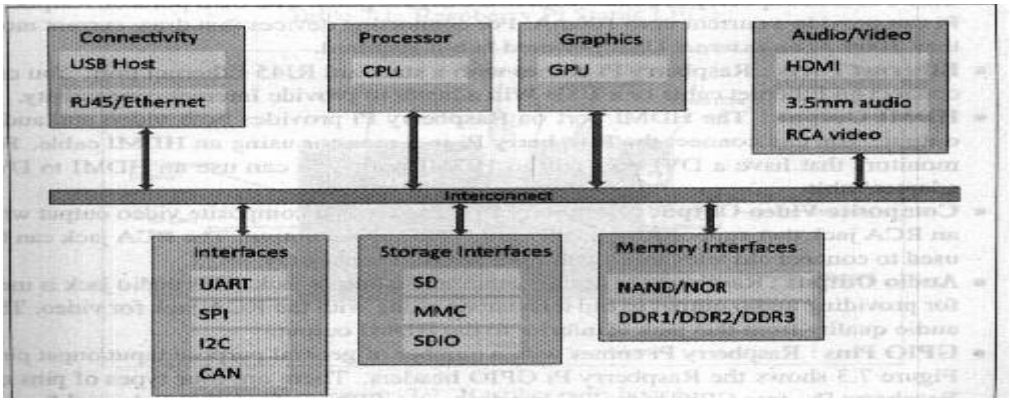
Basic building Socks of an loT Device

An loT device can consist of a number of modules based on functional attributes, such as:

Sensing: Sensors can be either on-board the IoT device or attached to the device. loT device can collect various types of information from the on-board or attached sensors such as temperature, humidity, light intensity, ctc. The sensed information can be communicated either to other *devices* or cloud-based servers/storage.

- Actuation: loT devices can have various types of actuators attached that allow taking actions upon the physical entities in the vicinity of the device. For example, a relay switch connected to an loT device can turn an appliance on/off based on the commands sent to the device.
- Communication: Communication modules are responsible for sending collected data to other devices or cloud-based servers/storage and receiving data from other devices and commands from remote applications.
- Analysis & Processing: Analysis and processing modules are responsible for making sense of the collected data.

The representative IoT device used for the examples in this book is the widely used single-board mini-computer called Raspberry Pi (explained in later sections). The use of Raspberry Pi is intentional since these devices are widely accessible, inexpensive, and available from multiple vendors. Furthermore.



Level 1: Physical Devices and Controllers (Edge; Things)

The first layer of the IoT Reference Model is the physical devices and controllers layer. This layer is home to the "things" in the Internet of Things, including the various endpoint devices and sensors that send and receive information. The size of these "things" can range from almost microscopic sensors to giant machines in a factory. Their primary function is generating data and being capable of being queried and/or controlled over a network

Level 2: Connectivity

In the second layer of the IoT Reference Model, the focus is on connectivity. The most important function of this IoT layer is the reliable and timely transmission of data. More specifically, this includes transmissions between Layer 1 devices and the network and between the network and information processing that occurs at Layer 3 (the edge computing layer).

Level 3: Edge (Fog) Computing

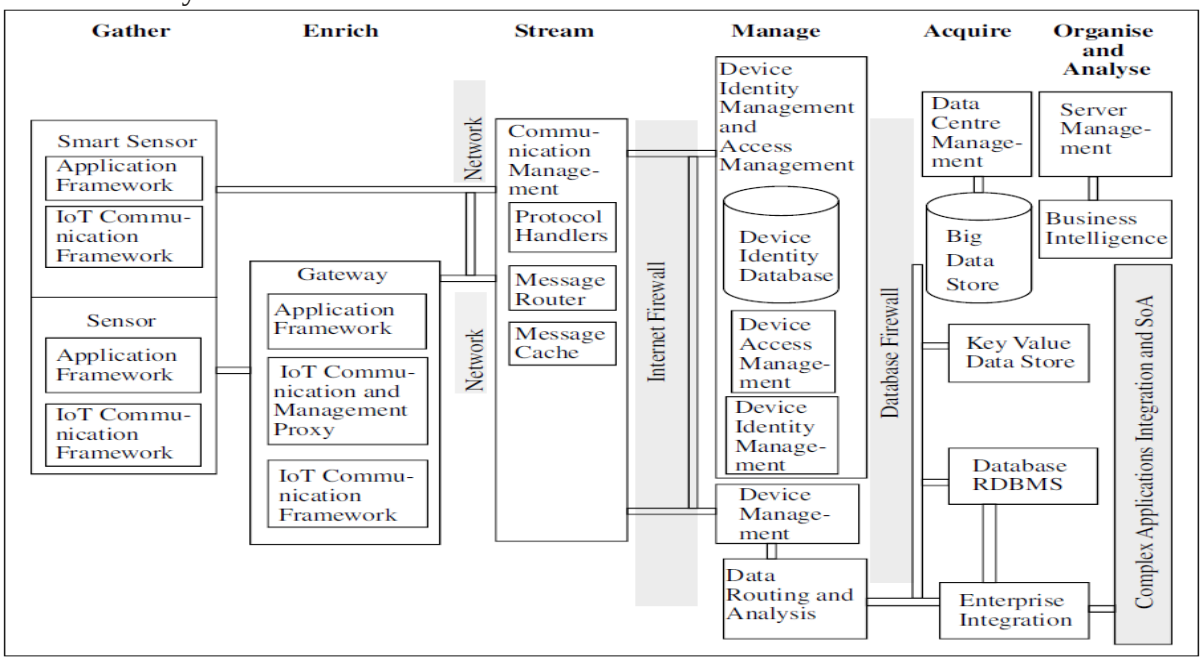
Edge computing is the role of Layer 3. Edge computing is often referred to as the "fog" layer and is discussed in the section "Fog Computing," later in this chapter. At this layer, the emphasis is on data reduction and converting network data flows into information that is ready for storage and processing by higher layers. One of the basic principles of this reference model is that information processing is initiated as early and as close to the edge of the network as possible

Level 4: Data Accumulation (Storage)

- This level converts data-in-motion to data-at-rest.
- Data format is converted from network packets to database relational tables.
- It transforms event-based computing to query based computing.
- Data is also reduced through filtering and selective storage.

Level 5: Data Abstraction (Aggregation and Access)

- This level creates schemas and views of data in the manner that applications want.
- It combines data from multiple sources.
- It simplifies, filters, selects, projects and reformats data to serve client applications.
- It reconciles the differences in data shape, format, semantics, access protocol and security.



Level 6: Application (Reporting, Analytics, Control)

- This level controls the applications and performs business intelligence and analytics.

Level 7: Collaboration and Processes

- This level involves the people and business processes.

4.1 DATA ACQUISITION AND INTEGRATION

Data acquisition and integration is the process of Acquisition of the data from data generated by different devices and validate the data and store the data.

4.1.0 Data Generation

Data generates at devices that later on, transfers to the Internet through a gateway. Data generates as follows:

- **Passive devices data:** Data generate at the device or system, following the result of interactions. A passive device does not have its own power source. An external source helps such a device to generate and send data. Examples are an RFID or an ATM debit card . The device may or may not have an associated microcontroller, memory and transceiver. A contactless card is an example of the former and a label or barcode is the example of the latter.
- **Active devices data:** Data generates at the device or system or following the result of interactions. An active device has its own power source. Examples are active RFID, streetlight sensor or wireless sensor node. An active device also has an associated microcontroller, memory and transceiver.
- **Event data:** A device can generate data on an event only once. For example, on detection of the traffic or on dark ambient conditions, which signals the event. The event on darkness communicates a need for lighting up a group of streetlights. A system consisting of security cameras can generate data on an event of security breach or on detection of an intrusion. A waste container with associate circuit can generate data in the event of getting it filled up 90% or above. The components and devices in an automobile generate data of their performance and functioning. For example, on wearing out of a brake lining, a play in steering wheel and reduced air-conditioning is felt. The data communicates to the Internet. The communication takes place as and when the automobile reaches near a Wi-Fi access point.
- **Device real-time data:** An ATM generates data and communicates it to the server instantaneously through the Internet. This initiates and enables Online Transactions Processing (OLTP) in real time.
- **Event-driven device data:** A device data can generate on an event only once. Examples are: (i) a device receives command from Controller or Monitor, and then performs action(s) using an actuator. When the action completes, then the device sends an acknowledgement; (ii) When an application seeks the status of a device, then the device communicates the status.

4.1.1Data Acquisition

Data acquisition means acquiring data from IoT or M2M devices. The data communicates after the interactions with a data acquisition system (application). The application interacts and communicates with a number of devices for acquiring the needed data. The devices send data on demand or at programmed intervals. Data of devices communicate using the network, transport and security layers in Seven-layer generalised OSI model

An application can configure the devices for the data when devices have configuration capability. For example, the system can configure devices to send data at defined periodic intervals. Each device configuration controls the frequency of data generation. For example, system can configure an umbrella device to acquire weather data from the Internet weather service, once each working day in a week. An ACVM(Automatic Chocolate Vending Machines) can be configured to communicate the sales data of machine and other information, every hour. The ACVM system can be configured to communicate instantaneously in event of fault or in case requirement of a specific chocolate flavour needs the Fill service .

Application can configure sending of data after filtering or enriching at the gateway at the data-adaptation layer. The gateway in-between application and the devices can provision for one or more of the following functions—transcoding, data management and device management. Data management may be provisioning of the privacy and security, and data integration, compaction and fusion .

Device-management software provisions for device ID or address, activation, configuring (managing device parameters and settings), registering, deregistering, attaching, and detaching. Automotive Components and Predictive Automotive Maintenance System (ACPAMS) application gives the process of acquiring data from the embedded component devices in the automobiles for Automotive Components and Predictive Automotive Maintenance System (ACPAMS) application.

4.1.2 Data Validation:

Data acquired from the devices does not mean that data are correct, meaningful or consistent. Data consistency means within expected range data or as per pattern or data not corrupted during transmission. Therefore, data needs validation checks. Data validation software do the validation checks on the acquired data. Validation software applies logic, rules and semantic annotations. The applications or services depend on valid data. Then only the analytics, predictions, prescriptions, diagnosis and decisions can be acceptable.

Large magnitude of data is acquired from a large number of devices, especially, from machines in industrial plants or embedded components data from large number of automobiles or health devices in ICUs or wireless sensor networks, and so on. Validation software, therefore, consumes significant resources. An appropriate strategy needs to be adopted. For example, the adopted strategy may be filtering out the invalid data at the gateway or at device itself or controlling the frequency of acquiring or cyclically scheduling the set of devices in industrial systems. Data enriches, aggregates, fuses or compacts at the adaptation layer.

4.1.3 Data Categorisation for Storage:

Services, business processes and business intelligence use data. Valid, useful and relevant data can be categorised into three categories for storage—data alone, data as well as results of processing, only the results of data analytics are stored. Following are three cases for storage:

1. Data which needs to be repeatedly processed, referenced or audited in future, and therefore, data alone needs to be stored.

2. Data which needs processing only once, and the results are used at a later time using the analytics, and both the data and results of processing and analytics are stored. Advantages of this case are quick visualisation and reports generation without reprocessing. Also the data is available for reference or auditing in future.

3. Online, real-time or streaming data need to be processed and the results of this processing and analysis need storage.

Data from large number of devices and sources categorises into a fourth category called Big data. Data is stored in databases at a server or in a data warehouse or on a Cloud as Big data.

4.2 ACCESS CONTROL:

Three FCs(functional components) in a security FG(function group) for ensuring security and privacy are:

- Authentications
- Authorisation
- Key exchange and management

4.2.1 Authentication:

ID establishment and authentication are essential elements of access control. A hash function or MD5 gives the irreversible result after many operations on that and the operations are just one way. The algorithm generates a fixed size, say, 128 or 256-bit hash or digest value using authentication data and secret key. Only the hash or digest value communicates. The receiver-end receives the value, and compares that with a stored value. If both are equal then the sender is authenticated.

Hash function characteristic features are pre-image resistance, hash function should not alter, before or after communication and should be as per the previous image (original message), second pre-image resistance: hash function should not be altered by an inbetween entity (called eavesdropper), should remain the same as one for the previous image (original message) should be collision-resistance and should not be the same for any form of altered message.

Example

Problem Show and list the steps for the key generation and exchange for authentication and authorisations followed by secure communication of an application/service message to the device/gateway.

Solution below Figure shows the steps of using FCs. The steps are for key exchanges and management, authentication and authorisations. The steps follow secure communication of an application or service message to the gateway and device. Steps for designing use case for key exchanges and encrypting and decrypting the messages are:

1. 1 and 2: Device/gateway D generates secret key K
2. 3 and 4: Application/service A generates secret key K
3. 5 and 6: D exchanges key K and K'
4. 7 and 8: Authentication and authorisation of D and A,
5. 9: Message given to encryption algorithm
6. 10: Encrypted message using K' to D
7. 11: Decryption algorithm decrypts encrypted message using K' for D
8. 12: Message M retrieves at D

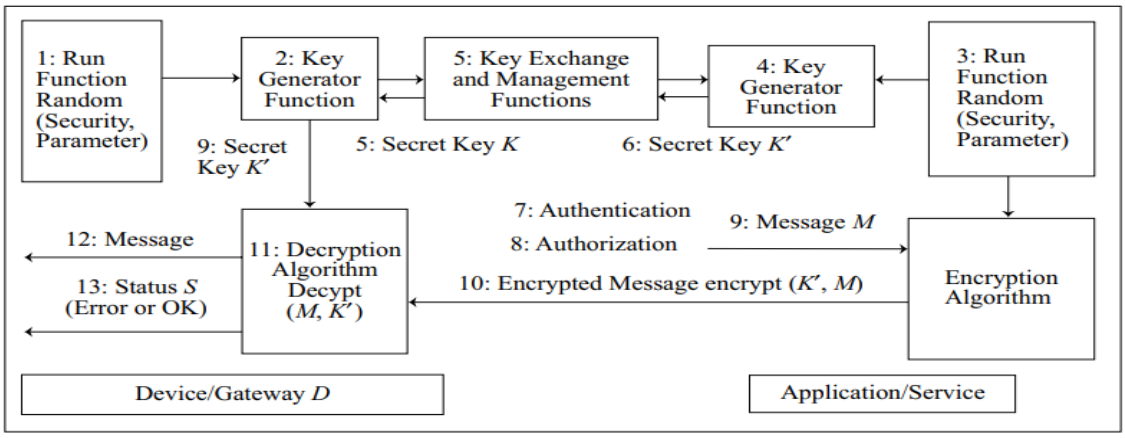


Figure Steps during key exchanges and management, authentication and authorisations followed by secure communication of application/service message to the device/gateway

9. 13: Status code sends 'Error' or 'OK'. As per the status of data exchanges for authentication, authorisations and message communication. Similarly, message encrypts at D using K and decrypts at A using K.

4.2.2 Authorisation:

Access control allows only an authorised device or application/service access to a resource, such as web API input, IoT device, sensor or actuator data or URL. Authorisation model is an essential element of secure access control. The standard authorisation models are as follows:

- Access Control List (ACL) for coarse-grain access control
- Role-Based Access Control (RBAC) for fine-grain access control
- Attribute-Based Access Control (ABAC) or other capability-based fine grain access control

An access control server and data communication gateway can be centrally used to control accesses between application/service and IoT devices. The server central control can be on a cloud server. Each device can access the server and communicate data to another server. Alternatively, a distributed architecture enables:

- Each device to request access to the server and the server grants application/service access token
- Each application/service to request access to the server and the server grants device access token for the device.

Example

Problem Show and list the steps for the key generation and exchange for authentication and authorisations followed by secure communication of an application/service message to the device/gateway.

Solution below Figure shows the steps of using FCs. The steps are for key exchanges and management, authentication and authorisations. The steps follow secure communication of an application or service message to the gateway and device. Steps for designing use case for key exchanges and encrypting and decrypting the messages are:

1. 1 and 2: Device/gateway D generates secret key K
2. 3 and 4: Application/service A generates secret key K
3. 5 and 6: D exchanges key K and K'
4. 7 and 8: Authentication and authorisation of D and A,
5. 9: Message given to encryption algorithm
6. 10: Encrypted message using K' to D
7. 11: Decryption algorithm decrypts encrypted message using K' for D
8. 12: Message M retrieves at D

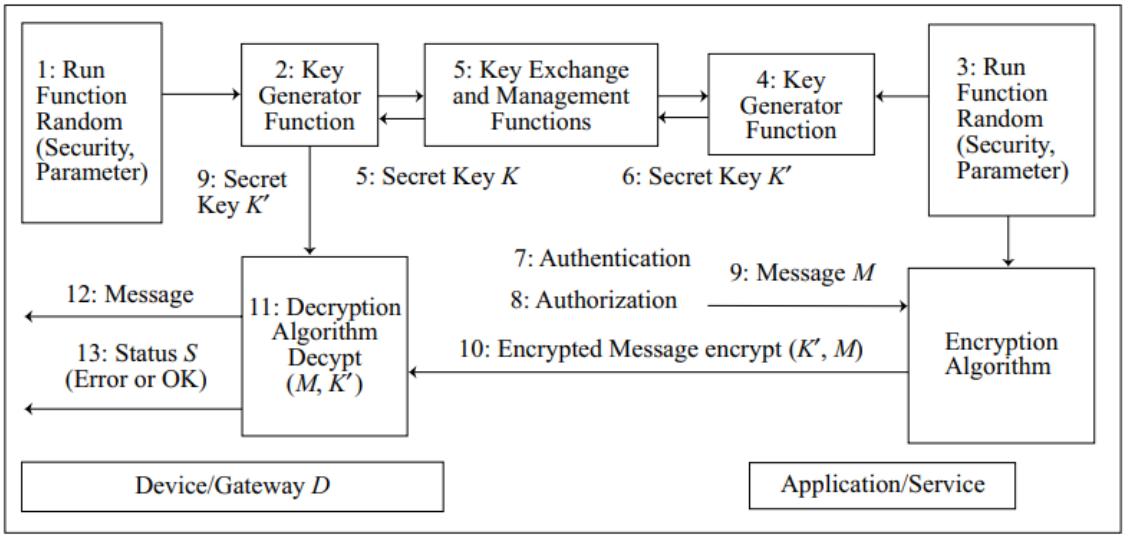


Figure Steps during key exchanges and management, authentication and authorisations followed by secure communication of application/service message to the device/gateway

9. 13: Status code sends ‘Error’ or ‘OK’. As per the status of data exchanges for authentication, authorisations and message communication. Similarly, message encrypts at D using K and decrypts at A using K.

4.2.3 Key Exchange and Management:

Key of sender messages needs to be known to receiver for accessing the received data. Key of respondent of messages needs to be known to sender for accessing the responses. The keys, therefore, need to be exchanged before the communication of authentication code, authorisation commands and encrypted messages. Since each application/service component and device data application or service may need unique and distinct keys, an FC provisions for the functions of key management and exchanges. Below Example gives the steps for key exchanges and encryption and decryption.

Example

Problem Show and list the steps for the key generation and exchange for authentication and authorisations followed by secure communication of an application/service message to the device/gateway.

Solution below Figure shows the steps of using FCs. The steps are for key exchanges and management, authentication and authorisations. The steps follow secure communication of an application or service message to the gateway and device. Steps for designing use case for key exchanges and encrypting and decrypting the messages are:

1. 1 and 2: Device/gateway D generates secret key K
2. 3 and 4: Application/service A generates secret key K
3. 5 and 6: D exchanges key K and K’
4. 7 and 8: Authentication and authorisation of D and A,
5. 9: Message given to encryption algorithm
6. 10: Encrypted message using K’ to D
7. 11: Decryption algorithm decrypts encrypted message using K’ for D
8. 12: Message M retrieves at D

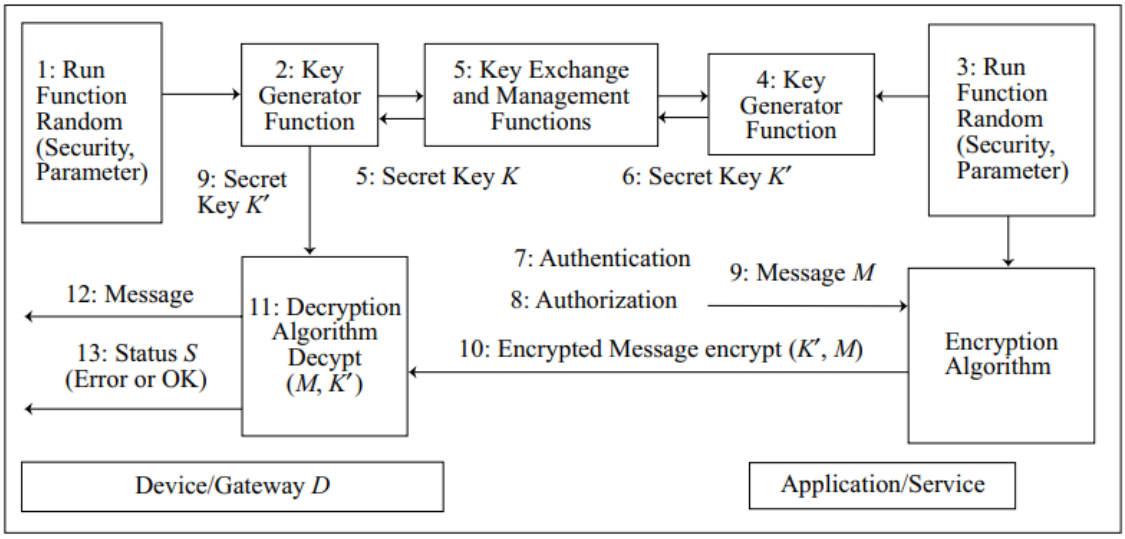


Figure Steps during key exchanges and management, authentication and authorisations followed by secure communication of application/service message to the device/gateway

9. 13: Status code sends ‘Error’ or ‘OK’. As per the status of data exchanges for authentication, authorisations and message communication. Similarly, message encrypts at D using K and decrypts at A using K.

4.3 What Is Structured Data?

Structured data has a well-defined schema for the information it holds. To give an extremely simple definition, any data that can be presented in a spreadsheet program like Google Sheets or Microsoft Excel is structured data.

In this example, data can be represented as rows and columns. Each column represents a different attribute, while each row will have the data associated with the attribute for a single instance. Rows and columns form a table that can be referenced easily.

Different tables can be connected—that is, they can be said to be related by the common column present in both tables.

If multiple tables are related in succession and combination, this creates a relational database. For instance, the customer, sales, and inventory data of a department store can be considered structured data stored as a relational database.

- Each customer will have a customer ID, as well as fields for their name, contact number, credit card information, address, etc.
- The database of customers can be connected to the database of sales, with attributes including the time of purchase, item codes purchased, total amount spent, customer ID, etc. Both the tables will be connected with the common attribute of customer ID.
- Finally, the sales database can be connected to the database of inventory using the common attribute of item code, effectively interconnecting all three tables into a relational database.

Structured data like this is generally stored in relational database management systems (RDBMSes). Databases can be written, read, and manipulated using Structured Query Language (SQL), a language that was developed by IBM in the 1970s to support its mainframe databases (though it was initially known as Sequence English Query Language or SEQUEL). It was so named since it reads pretty much like the English language. SQL in its current form was popularized by Relational Software, Inc. (now called Oracle).

SQL: (To store structured Data)

SQL stands for Structured Query Language. It is a language for viewing or changing (update, insert or append or delete) databases. It is a language for data querying, updating, inserting, appending and deleting the databases. It is a language for data access control, schema creation and modifications. It is also a language for managing the RDBMS.

SQL was originally based upon the tuple relational calculus and relational algebra. SQL can embed within other languages using SQL modules, libraries and pre-compilers. SQL features are as follows:

● Create Schema is a structure that contains descriptions of objects created by a user (base tables, views, constraints). The user can describe and define the data for a database.

● Create Catalog consists of a set of schemas that constitute the description of the database.

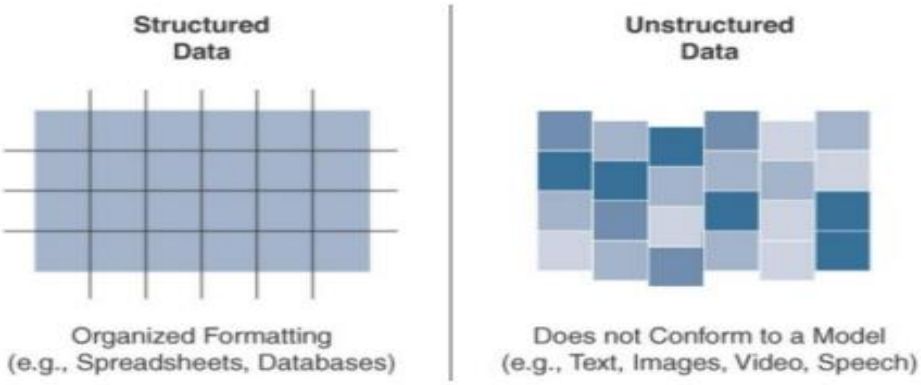
● Use Data Definition Language (DDL) for the commands that depict a database, including creating, altering and dropping tables and establishing constraints. The user can create and drop databases and tables, establish foreign keys, create view, stored procedure, functions in a database.

● Use Data Manipulation Language (DML) for commands that maintain and query a database. The user can manipulate (INSERT, UPDATE or SELECT the data and access data in relational database management systems.

● Use Data Control Language (DCL) for commands that control a database, including administering privileges and committing data. The user can set (grant or add or revoke) permissions on tables, procedures, and views

4.4 *Structured vs unstructured data*

Structured data and unstructured data are important classifications as they typically require different toolsets from a data analytics perspective. Below Figure provides a highlevel comparison of structured data and unstructured data. Structured data means that the data follows a model or schema that defines how the data is represented or organized, meaning it fits well with a traditional relational database management system (RDBMS). Inmany cases you will find structured data in a simple tabular form—for example, a spreadsheet where data occupies a specific cell and can be explicitly defined and referenced. Structured data can be found in most computing systems and includes everything from banking transaction and invoices to computer log files and router configurations.



IoT sensor data often uses structured values, such as temperature, pressure, humidity, and so on, which are all sent in a known format. Structured data is easily formatted, stored, queried, and processed; for these reasons, it has been the core type of data used for making business decisions. Because of the highly organizational format of structured data, a wide array of data analytics tools is readily available for processing this type of data. From custom scripts to commercial software like Microsoft Excel and Tableau, most people are familiar and comfortable with working with structured data. Unstructured data lacks a logical schema for understanding and decoding the data through traditional programming means. Examples of this data type include text, speech, images, and video. As a general rule, any data that does not fit neatly into a predefined data model is classified as unstructured data. According to some estimates, around 80% of a business's data is unstructured.

Because of this fact, data analytics methods that can be applied to unstructured data, such as cognitive computing and machine learning, are deservedly garnering a lot of attention. With machine learning applications, such as natural language processing (NLP), you can decode speech. With image/facial recognition applications, you can extract critical information from still images and video. The handling of unstructured IoT data employing machine learning techniques is covered in more depth later. Smart objects in IoT networks generate both structured and unstructured data. Structured data is more easily managed and processed due to its well-defined organization. On the other hand, unstructured data can be harder to deal with and typically requires very different analytics tools for processing the data. Being familiar with both of these data classifications is important because knowing which data classification you are working with makes integrating with the appropriate data analytics solution much easier.

Characteristic	Structured Data	Unstructured Data
Nature of data	Usually quantitative	Usually qualitative
Data model	Pre-defined; once it is defined and some data stored, it is difficult to change the model	No particular schema is involved in unstructured data; the data model is very flexible
Data format	A limited number of data formats are available	A huge variety of data formats are available for unstructured data

Database	SQL-based relational databases are used	NoSQL databases with no specific schema are used
Search	Very easy to search and find data within the database or data set	Very difficult to search for particular data due to its unstructured nature
Analysis	Very easy to analyze, given the quantitative nature of data	Very difficult to analyze, even with existing software tools
Storage method	Data warehouses are used for structured data	Data lakes are used to store unstructured data

4.5 UNSTRUCTURED DATA STORAGE ON CLOUD/LOCAL SERVER

Different methods of data collection, storage and computing are shown in below Figure. The figure shows (i) Devices or sensor networks data collection at the device web server, (ii) Local files, (iii) Dedicated data store at coordinating node, (iii) Local node in a distributed DBMS, (iv) Internet-connected data centre, (v) Internet-connected server, (vi) Internet-connected distributed DBMS nodes, and (vii) Cloud infrastructure and services.

Cloud computing paradigm is a great evolution in Information and Communications Technology (ICT). The new paradigm uses XAAS at the Internet connected clouds for collection, storage and computing.

Following are the key terms and their meanings, which need to be understood before learning about the cloud computing platform.

Resource refers to one that can be read (used), written (created of changed) or executed (processed). A path specification is also a resource. The resource is atomic (not-further divisible) information, which is usable during computations. A resource may have multiple instances or just a single instance. The data point, pointer, data, object, data store or method can also be a resource

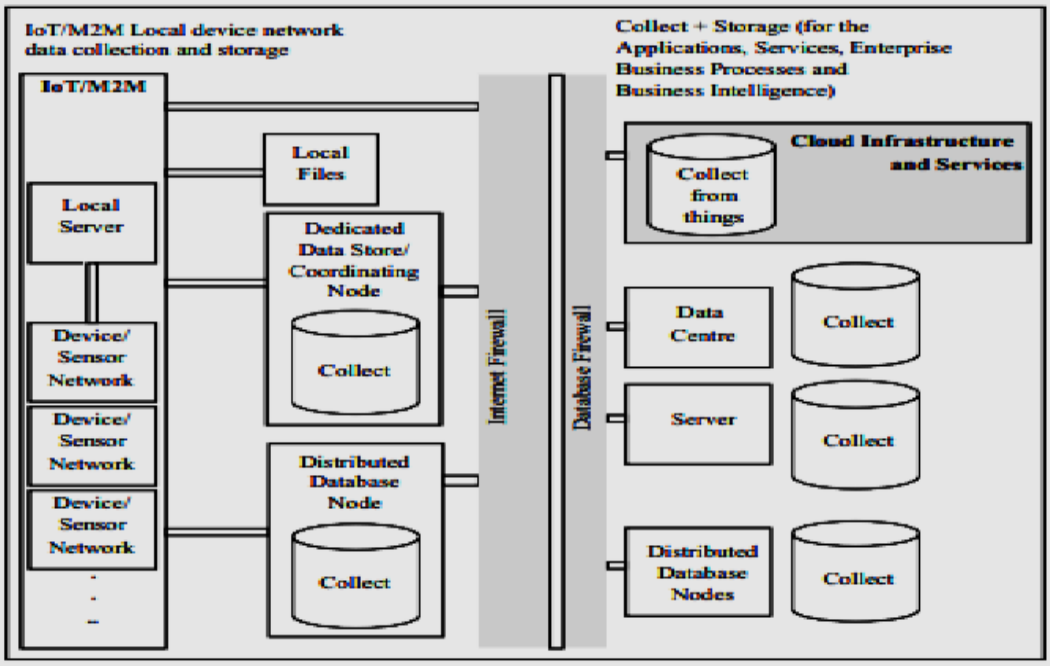


Figure Devices or sensors network data collection at a device local-server, local files, dedicated data store, at a coordinating node, a local node of a distributed DBMS, Internet-connected server of data centre, server or distributed database nodes or a cloud infrastructure

What Is Unstructured Data?

Every piece of data that is not structured data can be classified as unstructured data. It's estimated that by 2025, 80% of the data we encounter will be unstructured data in the form of text, audio, image, or video[1].

In short, unstructured data is modern data. It's often:

- Born digital and unpredictable
- Always being created and on the move
- Blended, multimodal, and interoperable
- Geo-distributed for better protection

Unstructured data can have some associated metadata that can, in turn, have a structure. For example, a video can have metadata of video resolution, bit rate, frames per second (FPS), owner of the video, etc. But the video itself is unstructured. When there's some structured metadata associated with unstructured data, it's occasionally referred to as semi-structured data.

Looking more closely at the example of a YouTube video, some metadata is present, such as the time of upload, date of upload, number of views (partial or full), number of likes and dislikes, etc. But the content inside the video title, the video description, and the video itself is unstructured. It has a qualitative aspect that cannot be captured purely by numbers.

The most commonly used database for unstructured data is NoSQL. NoSQL stands for "not only SQL," indicating that the database can handle a wider range of data beyond the capabilities of SQL databases. There's no schema or tabular structure for NoSQL databases; it's just a collection of data grouped together.

NOSQL: (To store Unstructured Data)

NOSQL stands for No-SQL or Not Only SQL that does not integrate with applications that are based on SQL. NOSQL is used in cloud data store. NOSQL may consist of the following:

● A class of non-relational data storage systems, flexible data models and multiple schemas

● Class consisting of uninterpreted key and value or 'the big hash table'. For example in [Dynamo (Amazon S3)]

● Class consisting of unordered keys and using the JSON. For example in PNUTS

● Class consisting of ordered keys and semi-structured data storage systems. For examples in the BigTable, Hbase and Cassandra (used in Facebook and Apache)

● Class consisting of JSON (Section 2.3). For example in MongoDb6 which is widely used for NOSQL)

● Class consisting of name and value in the text. For example in CouchDB

● May not require a fixed table schema NOSQL systems do not use the concept of joins (in distributed data storage systems). Data written at one node replicates to multiple nodes, therefore identical and distributed system can be fault-tolerant, and can have partitioning tolerance. CAP theorem is applicable. The system offers relaxation in one or more of the ACID and CAP properties. Out of the three properties (consistency, availability and partitions), two are at least present for an application.

● Consistency means all copies have same value like in traditional DBs.

● Availability means at least one copy available in case a partition becomes inactive or fails. For example, in web applications, the other copy in other partition is available.

● Partition means parts which are active but may not cooperate as in distributed databases