UNIT- 4

SOLUTION FRAMEWORK FOR IOT APPLICATIONS

IOT FRAMEWORK:

A framework provides the implementation of the basic infrastructure and a standard way to implement these services. Executives and Services provide a fast way to deliver IoT products.

The IoT framework includes the capabilities to support the cloud and all the other needs that IoT technology has to meet. IOT framework can be used by any manufacturer to design IOT products.

For example, any IoT system will need cloud services, security for data stored in the cloud, supporting protocols, edge devices such as gateways, and supporting software. Amazon Web Services, Microsoft Azure are cloud capacity service providers.

The IoT framework reduces the time it takes to create products that are generally Internet-enabled, speeding up the development cycle. Which in turn shortens product deployment time and helps drive innovation.

Four Basic Components of IoT Framework:

Device Hardware – Requires an idea of the architecture and operation of various microcontrollers as well as various sensors.

Device Software – Requires knowledge of how the API works in microcontrollers and how you can create libraries for programming.

Communication and Cloud Platform – Requires the basics of wired and wireless communication. The cloud itself is an indispensable part of IoT and requires knowledge of how cloud technology works and its IoT integration.

Cloud Application – This is a software program in which local and cloud-based components work together for faster and easier accessibility. It serves to improve our ability to use the system to its fullest potential.

Open Source IoT Frameworks are ThingSpeak ,KAA IoT ,AWS IoT, IBM Bluemix, ZETTA etc by using these frameworks we can design IOT products.

ThingSpeak: is an Internet of Things (IoT) framework that allows you to collect and store data in a cloud and develop IoT applications.

Kaa IoT: is one of the most powerful and richest open-source Internet cloud platforms. Where anyone has a free way to implement their intelligent product concepts.

AWS IoT *:(*Amazon Internet of Things) is an Amazon Web Services platform that collects and analyzes data from devices and sensors connected to the Internet and connects that data to AWS cloud applications.

IBM Bluemix: It allows organizations and developers to quickly and easily create, deploy, and manage applications in the cloud.

Zetta: is an open-source IoT framework into which we can build APIs for device interaction**.**

DEVICE INTEGRATION:

IOT device is the collection of various elements on a single chip and provided with internet. These devices are know as IOT devices and many devices are mounted on a single board, must communicate with each other . for this communication API are used and most popular API is REST API. To provide the communication between various elements that are mounted on the single chip to form iot device. This whole thing is know as device integration.

DATA ACQUISITION AND DATA INTEGRATION**:**

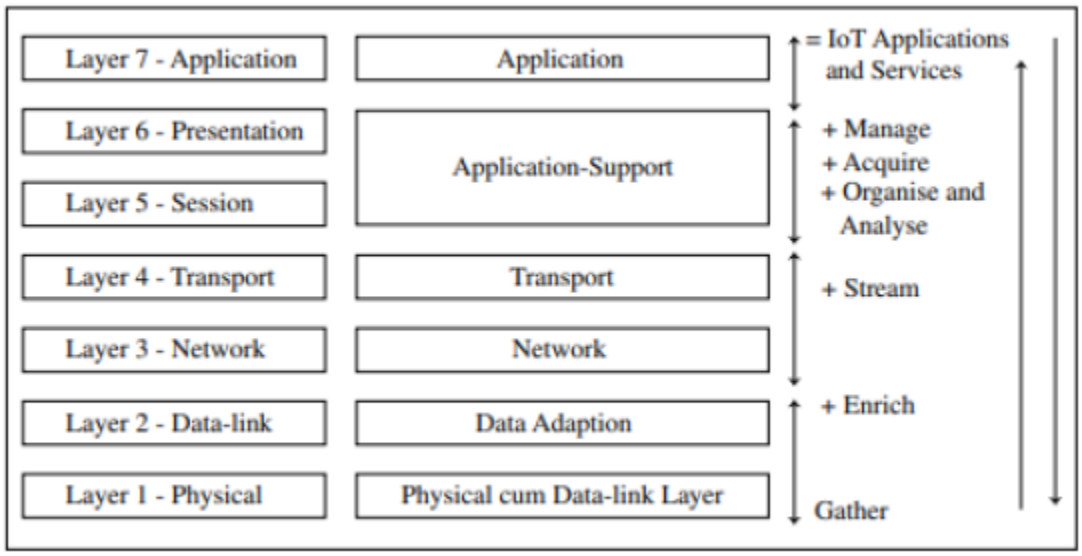
Data acquisition means acquiring data from IoT or M2M devices. i.e collection of data in different form from various devices mounted on IOT devices. The data communicates after the interactions with a data acquisition system (application). The application interacts and communicates with a number of devices 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.

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 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.



The components of data acquisition systems include:

i) Sensors that convert physical parameters to electrical signals.

ii)Signal conditioning circuitry to convert sensor signals into a form that can be converted to digital values.

iii) Analog-to-digital converters, which convert conditioned sensor signals to digital values.

Explanation

Data acquisition is the process of extracting, transforming, and transporting data from the source systems and external data sources to the data processing system to be displayed, analyzed, and stored.

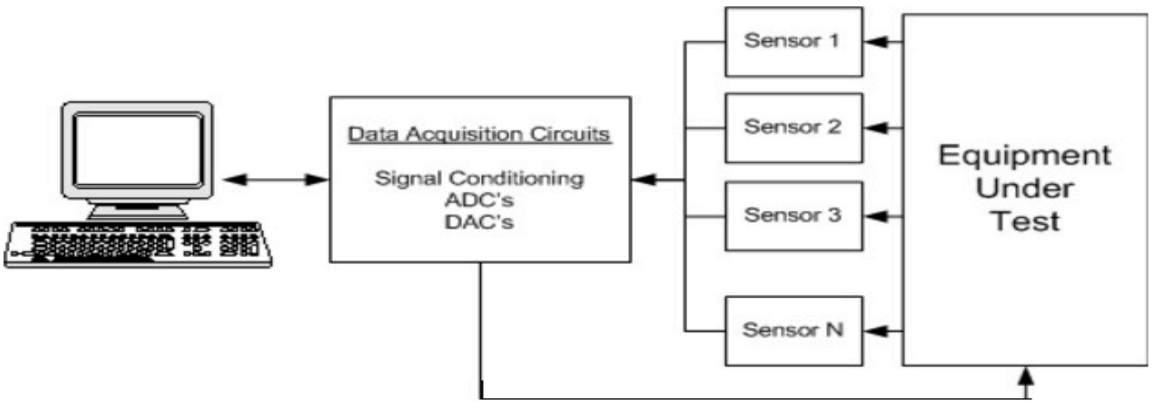
A data acquisition system (DAQ) typically consist of transducers for asserting and measuring electrical signals, signal conditioning logic to perform amplification, isolation, and filtering, and other hardware for receiving analog signals and providing them to a processing system, such as a personal computer.

Data acquisition systems are used to perform a variety of functions, including laboratory research, process monitoring and control, data logging, analytical chemistry, tests and analysis of physical phenomena, and control of mechanical or electrical machinery.

Data recorders are used in a wide variety of applications for imprinting various types of forms, and documents.

Data collection systems or data loggers generally include memory chips or strip charts for electronic recording, probes or sensors which measure product environmental parameters and are connected to the data logger.

Hand-held portable data collection systems permit in field data collection for up-to-date information processing.



DATA INTEGRATION means prepare data collected in various form to be understood by computer. Combination of technical and business process is used to make that data a valid information.

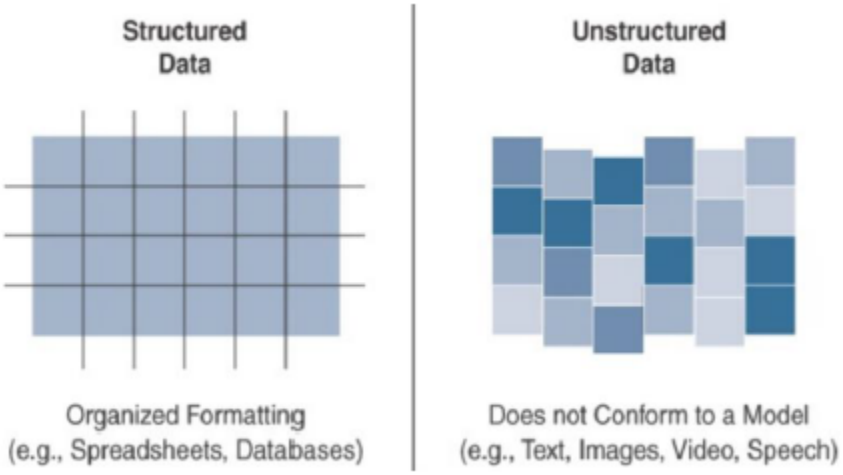
What is Unstructured Data?

Unstructured data is essentially all data that doesn't fall under the purview of relational databases (RDBMS). Unstructured data is not structured via predefined data schema or models. However, it has an internal structure – it can be textual or non-textual, or human- or machine-generated, and can be stored within non-relational databases like NoSQL. Examples of unstructured data include text files, email, mobile data, social media, satellite imagery, sensor or surveillance data, communications such as chats, etc.

Unstructured data is something of a misnomer. Sure, some of this data is difficult to analyze or process, but some of the data have additional features such as metadata, making them semi-structured.

Structured vs unstructured data

Structured data and unstructured data are important classifications as they typically require different toolsets from a data analytics perspective. Figure 2 provides a high-level 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 are 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.

Cloud Storage of Unstructured Data

As we have seen, unstructured data can include pretty much all kinds of information. The file sizes can range from anything to a few bits and bytes to gigabytes or more. Hence, there is no one-size-fits-all approach in terms of data storage. The type of storage where the data sits depends on the capacity as well as the set input/output (I/O) requirements. So, anything from low I/O performance (NAS, cloud instance, object storage) to high-performing, massive files (distributed file, object storage).

Network-attached storage (NAS) used to be associated with single file, siloed storage of data. Not anymore. These days, scale-out NAS is able to manage high-performance, high-capacity storage of the data. But again, object storage has also grown over the years and leads in unstructured data storage. Object storage comes with many advantages, such as having unique IDs for the stored data, being high-performance, highly scalable, and easily accessed with APIs. It is no surprise that most cloud providers opt for object storage.

Cloud providers offer high-performance, scalable storage services to customers, and there is a high demand for these flexible services. Some of them come in subscription-based systems or open source, reducing the overall financial burden to enterprises and organizations.

Storage requirements for unstructured data:

Companies should strategize storing unstructured data during the planning phase of a big data project. The storage infrastructure should be agile, cost-effective, scalable, and cater to a wide range of use cases.

Consider the following requirements for unstructured data storage:

Flexibility

The data model should be flexible to accommodate new fields and data types with minimum impact on existing schema or data, thus requiring no downtime.

Purpose

If your workload is mainly analytics, you need a robust storage system that supports low latency and faster data updates. Cloud storage would be a good option for this purpose as opposed to an on-premise system.

Easy access to archived data

Data archiving prevents data loss, and reduces the cost of primary storage. Data that is old but still required should be stored in such a way that it's easy to retrieve and doesn't increase overall storage cost.

Scalability

The storage system should be horizontally and vertically scalable at all times without any data loss. Modern storage systems like AWS and Azure provide automatic scaling depending on the application requirements.

A NoSQL database is a good approach that satisfies all the above unstructured data storage requirements. To handle scalability and online archiving capabilities as the data continues to grow, cloud-based databases like MongoDB Atlas, a database-as-a-service like MongoDB clusters, and data lakes like MongoDB Atlas Data Lake are excellent options.

STORE UNSTRUCTURED DATA IN COLUD: You can store unstructured data on-premise or in the cloud using a database, data warehouse, or data lake.

While cloud storage *does* offer security, companies might prefer on-premise storage for highly sensitive data.

Non-relational database for storing unstructured data

Non-relational (NoSQL) databases have emerged as a convenient way of storing unstructured big data. They are flexible, scalable, highly available, secure, and help to minimize the unstructured data storage challenges. NoSQL databases make data management more efficient and cost-effective.

There are various types of NoSQL database systems. One type is the document (object) store, which provides a simple query mechanism to quickly retrieve data as the system recognizes the data structure. Documents consist of various attributes with different data types. Document stores are highly scalable and available by design, and can partition, replicate, and persist the data. MongoDB is a document-based NoSQL database that stores data in a BSON (JSON-like format). Such a format is easy to read and traverse. MongoDB is also suitable for handling transactional data.

If you were to store the above information in a relational database, you'd probably need three or more tables and would need to join the tables to see all this information in one view.

MongoDB Atlas, MongoDB's database-as-a-service, utilizes major cloud platforms like AWS, Azure, and Google Cloud for its database servers. This means you don't need to install MongoDB and still get all the benefits of a NoSQL document database in a cloud environment.

Unstructured data storage with data lake

A data lake is a central storage repository that stores data in its native format. It uses flat architecture to store data, usually as object or file storage. Data lakes are vast and store any amount of unstructured, structured, or semi-structured big data. They work on the schema-on-read principle (i.e., do not have a predefined schema).

The data sources can be IoT devices, streaming data, web applications, and many others. Some of the data ingested might be filtered and ready to use as well — the kind of flexibility impossible with relational databases.

Since data lakes are configured on commodity hardware and clusters, they are highly scalable and inexpensive.

Data lakes can be configured on-premise or in the cloud. Again, on-premise data lakes are suitable for highly sensitive and secure data. However, having a cloud data lake reduces the cost of infrastructure and is easier to scale out.

MongoDB Atlas Data Lake is a great solution that provides a single platform for your MongoDB Atlas clusters and allows you to:

- Organize and query across multiple Atlas and AWS S3 clusters.
- Structure the data stored in a data lake.
- Convert MongoDB data into Parquet, CSV, or other formats.
- Natively query, transform, and move data across AWS S3 and MongoDB Atlas clusters.

Data warehouse

A data warehouse is a repository created for analytics and reporting purposes. It usually works on a structured storage (schema-on-write), unlike data lakes. Data warehouses primarily store past and current structured or semi-structured data, which is internal to the organization and available in standard format. Unstructured data (like that from the internet) should be processed and formatted with an ETL step before being ingested into a data warehouse. This makes the data consistent and of high quality—and, therefore, ready for analysis. You can say that a data warehouse is an analytical database used for business intelligence. The schema-based format makes data analysis easier.

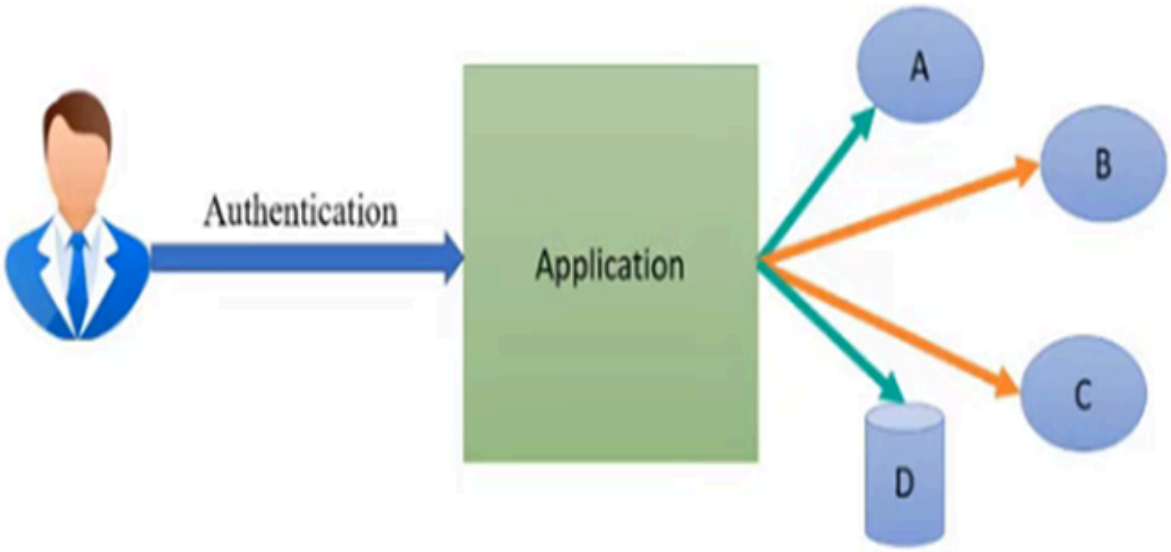
Data warehouses can be on-premise and cloud-based. Cloud data warehouses reduce the cost, deployment process, and infrastructure needs, and can automatically scale based on application needs.

A data mart is a subset of a data warehouse that stores operational data of a particular niche or line of business.

AUTHENTICATION AND AUTHORIZATION:

Definition: Authentication is the process of recognizing a user's identity. Authentication technology provides access control for systems by checking to see if a user's credentials match the credentials in a database of authorized users or in a data authentication server. In doing this, authentication assures secure systems, secure processes and enterprise information security.

Description: The authentication process always runs at the start of the application, before the permission and throttling checks occur, and before any other code is allowed to proceed.



There are several authentication types. For purposes of user identity, users are typically identified with a user ID, and authentication occurs when the user provides credentials such as a password that matches their user ID. The practice of requiring a user ID and password is known as *single-factor authentication* (SFA). In recent years, companies have strengthened authentication by asking for additional authentication factors, such as a unique code that is provided to a user over a mobile device when a sign-on is attempted or a biometric signature, like a facial scan or thumbprint. This is known as *two-factor authentication* (2FA).

Authentication factors can even go further than SFA, which requires a user ID and password, or 2FA, which requires a user ID, password and biometric signature. When three or more identity verification factors are used for authentication -- for example, a user ID and password, biometric signature and perhaps a personal question the user must answer -- it is called *multifactor authentication* (MFA).

Other authentication methods include the following:

- **2FA.** This type of authentication adds an extra layer of protection to the process by requiring users to provide a second authentication factor in addition to the password. 2FA systems often require the user to enter a verification code received via text message on a preregistered mobile phone or mobile device, or a code generated by an authentication application.

- **MFA.** This type of authentication requires users to authenticate with more than one authentication factor, including a biometric factor, such as a fingerprint or facial recognition; a possession factor, like a security key fob; or a token generated by an authenticator app.
- **OTP.** An OTP is an automatically generated numeric or alphanumeric string of characters that authenticates a user. This password is only valid for one login session or transaction and is typically employed for new users or for users who lost their passwords and are given an OTP to log in and change to a new password.
- **Three-factor authentication.** This type of MFA uses three authentication factors -- usually, a knowledge factor, such as a password, combined with a possession factor, such as a security token, and an inherence factor, such as a biometric.

Biometrics. While some authentication systems depend solely on biometric identification, biometrics are usually used as a second or third authentication factor. The more common types of biometric authentication available include fingerprint scans, facial or retina scans, and voice recognition

Why is authentication important in cybersecurity?

Authentication enables organizations to keep their networks secure by permitting only authenticated users or processes to gain access to their protected resources. This may include computer systems, networks, databases, websites and other network-based applications or services.

Once authenticated, a user or process is usually subjected to an authorization process to determine whether the authenticated entity should be permitted access to a specific protected resource or system. A user can be authenticated but not be given access to a specific resource if that user was not granted permission to access it.

The terms *authentication* and *authorization* are often used interchangeably. While they are often implemented together, they are two distinct functions. Authentication is the process of validating the identity of a registered user or process before enabling access to protected networks and systems. Authorization is a more granular process that validates that the authenticated user or

process has been granted permission to gain access to the specific resource that has been requested. The process by which access to those resources is restricted to a certain number of users is called *access control*. The authentication process always comes before the authorization process.

How does authentication work?

During authentication, credentials provided by the user are compared to those on file in a database of authorized users' information either on the local operating system server or through an authentication server. If the credentials entered match those on file and the authenticated entity is authorized to use the resource, the user is granted access. User permissions determine which resources the user gains access to and also any other access rights that are linked to the user, such as during which hours the user can access the resource and how much of the resource the user is allowed to consume.

authentication vs. authorization

Authorization includes the process through which an administrator grants rights to authenticated users, as well as the process of checking user account permissions to verify that the user has been granted access to those resources. The privileges and preferences granted for an authorized account depend on the user's permissions, which are either stored locally or on an authentication server. The settings defined for all these environment variables are established by an administrator.