

# IoT Platforms: AWS vs Azure vs Google vs IBM vs Cisco

Farnaz Golnam  
NPU MSCS



# Table of Content

1. Introduction: What is IOT
2. IOT Architecture Layers
3. Comparing Different IOT Platforms
4. Conclusion: How to Decide on Platforms
5. References

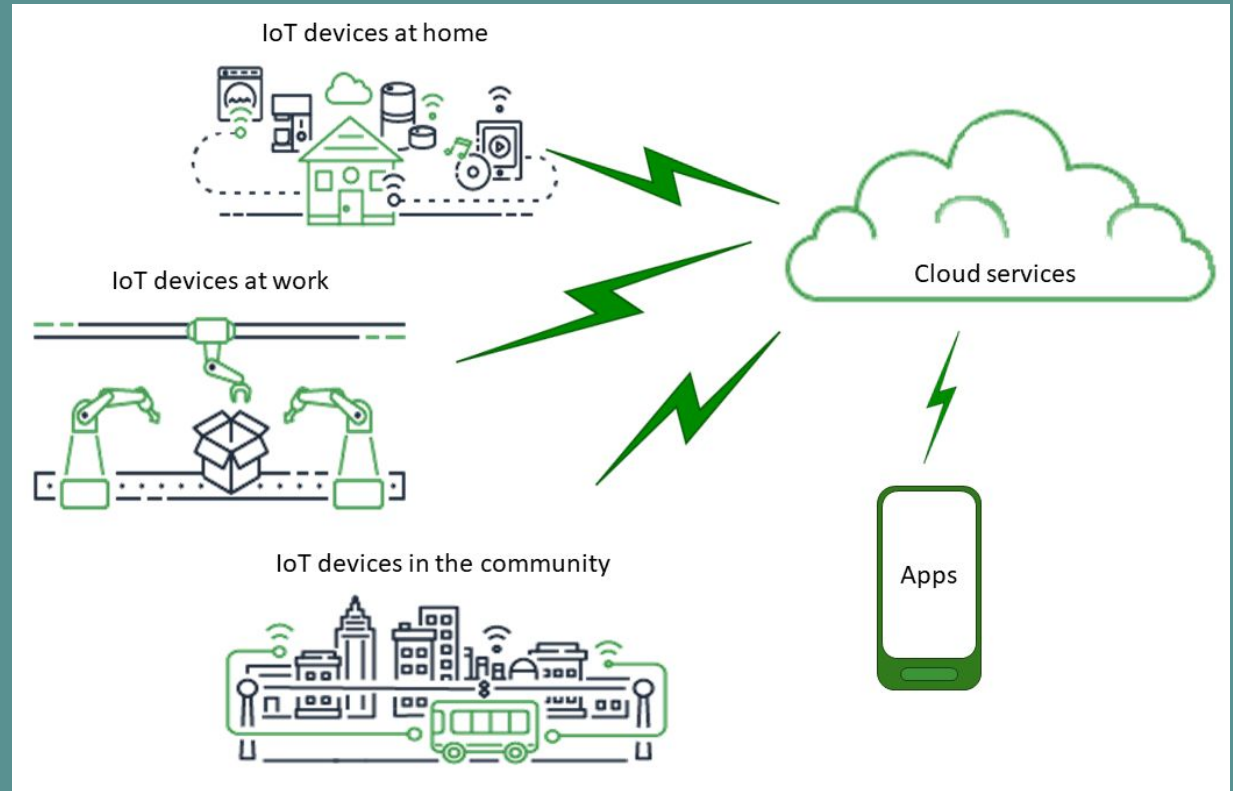


# Introduction: What is IoT

Internet of things(IOT) is a collective term for the physical objects that are connected to the internet (directly or indirectly) and can exchange data without user involvement.

The Internet of Things (IoT) consists of the key components shown in this diagram.

(image source [AWS website](#) )



## Apps

Apps give end users access to IoT devices and the features provided by the cloud services to which those devices are connected.

## Cloud services

Cloud services are distributed, large-scale data storage and processing services that are connected to the internet. Examples include:

- IoT connection and management services for example AWS IoT, Cisco IOT, Microsoft Azure IOT, IBM Watson IOT, Google Cloud IOT.
- Compute services, such as AWS Lambda.
- Database services, such as Amazon DynamoDB

## Communications

Devices communicate with cloud services by using various technologies and protocols.

Examples include:

- Wi-Fi/Broadband internet
- Broadband cellular data
- Narrow-band cellular data
- Long-range Wide Area Network (LoRaWAN)
- Proprietary RF communications



## Devices

A device is a type of hardware that manages interfaces and communications. Devices are usually located in close proximity to the real-world interfaces they monitor and control. Devices can include computing and storage resources, such as microcontrollers, CPU, memory. Examples include:

- Raspberry Pi
- Arduino
- Voice-interface assistants
- LoRaWAN and devices
- Amazon Sidewalk devices
- Custom IoT devices

**Interfaces,** An interface is a component that connects a device to the physical world.

- **User interfaces:** Components that allow devices and users to communicate with each other.

- Input interfaces: Enable a user to communicate with a device, Examples: keypad, button
- Output interfaces: Enable a device to communicate with a user,

Examples: Alpha-numeric display, graphical display, indicator light, alarm bell

- **Actuators:** Output components that the device can use to control something in the outside world. Examples include:
  - Stepper motors (convert electric signals to movement)
  - Relays (control high electric voltages and currents)

- **Sensors:** input components that measure or sense something in the outside world in a way that a device understands. Examples:

- Temperature sensor (converts temperature to an analog voltage)
- Humidity sensor (converts relative humidity to an analog voltage)
- Analog to digital convertor (converts an analog voltage to a numeric value)
- Ultrasonic distance measuring unit (converts a distance to a numeric value)
- Optical sensor (converts a light level to a numeric value)
- Camera (converts image data to digital data)



## **2. IOT Architecture Layers**



# IOT Architecture Layers: Three-layer architecture

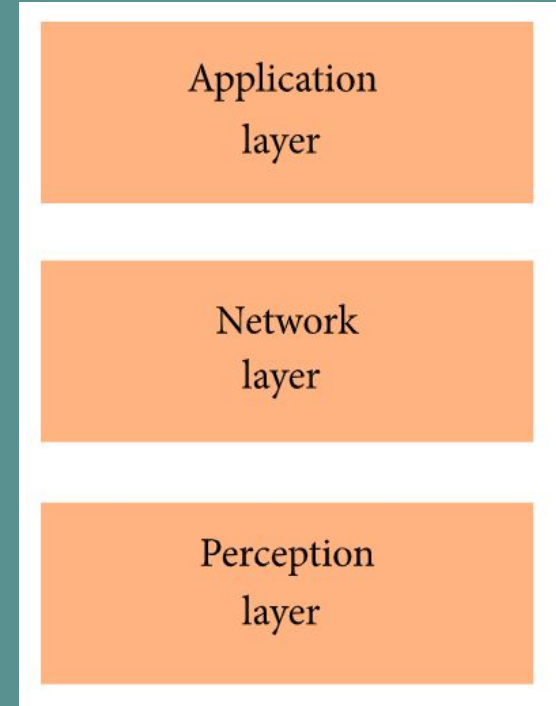
The most basic architecture is a **three-layer architecture** which was introduced in the early stages of IOT research.

**Perception layer** is the physical layer, which has sensors for sensing and gathering information from the environment. It senses physical parameters or identifies other smart objects in the environment.

**Network layer** transmit and process sensor data and is responsible for connecting to other smart things, network devices, and servers.

**Application layer** is responsible for delivering application specific services to the user. It defines various applications in which the Internet of Things can be deployed, for example, smart homes, smart cities, and smart health.

The three-layer architecture defines the main idea of the Internet of Things, but it is not enough for research on IoT and does not cover the finer aspects of the Internet of Things. So, more layered architectures proposed one of them is the five-layer architecture.



# IOT Architecture Layers: Five-layer architecture

The **five-layer architecture** consists of perception, transport, processing, application, and business layers. The role of the perception and application layers is the same as the architecture with three layers.

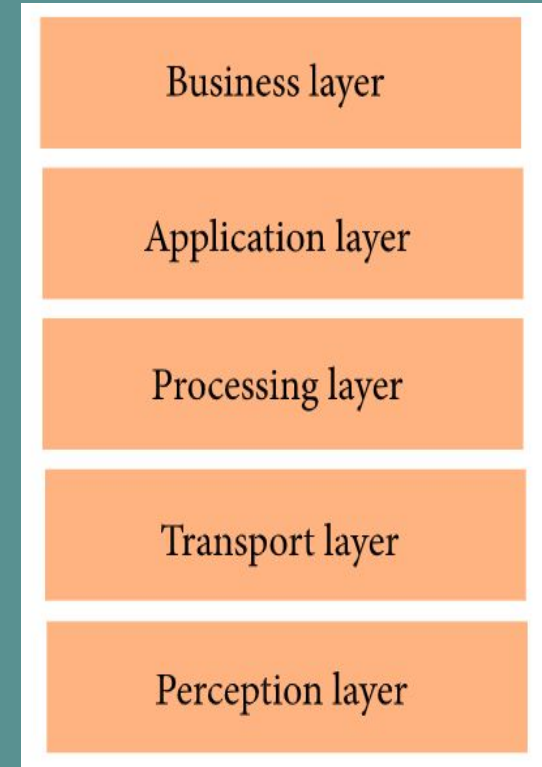
**Perception layer** is the physical layer, which has sensors for sensing and gathering information from the environment. It senses physical parameters or identifies other smart objects in the environment.

**Transport layer** transfers the sensor data from the perception layer to the processing layer and vice versa through networks such as wireless, 3G, LAN, Bluetooth, RFID, and NFC

**Processing layer** is also known as the middleware layer. It stores, analyzes, and processes huge amounts of data that comes from the transport layer. It applies many technologies such as databases, cloud computing, and big data processing modules.

**Application layer** is responsible for delivering application specific services to the user. It defines various applications in which the Internet of Things can be deployed, for example, smart homes, smart cities, and smart health.

**Business layer** manages the whole IoT system, including applications, business and profit models, and users' privacy.





## Perception Layer

**Perception layer (The hardware or “things”)** includes the following gears that work with signals from the physical world:

**Electronic sensors** capture signals from the physical world, convert them into digital form, and feed them to the IoT system. You can monitor and manage sensors remotely, using a special application.

**Actuators** receive signals from the IoT system and translate them into physical actions manipulating equipment. Similar to sensors, actuators can be configured from remote computers.

**Devices** are connected to sensors or even have them embedded as an integral part. On the other side, devices link to a gateway (in transport layer) or directly to an IoT platform (in processing layer). These hardware components cache and preprocess real-time data, reducing the burden on central storages and main processors.

# Transport Layer, Gateway and Protocols

**Transport layer (Networks and gateways)** encompasses **wired or wireless networks and a gateway** (a hardware or software module that consolidates data from devices, analyzes it, performs translation between different **protocols**, and forwards information to the cloud (processing layer). As a rule, the gateway converts all information into MQTT messages) the lightweight protocol most widely used in the IoT.

A **gateway** is a piece of networking hardware or software that acts as a "gate" between two networks. It may be a router, firewall, server, or other device that enables traffic to flow in and out of the network. The gateway node is considered to be on the "edge" of the network as all data must flow through it before coming in or going out of the network. It may also translate data received from outside networks into a format or protocol recognized by devices within the internal network.

A **router** is a common type of gateway used in home networks. It allows computers within the local network to send and receive data over the Internet.

A **firewall** is a more advanced type of gateway, which filters inbound and outbound traffic, disallowing incoming data from suspicious or unauthorized sources.

A **proxy server** is another type of gateway that uses a combination of hardware and software to filter traffic between two networks. For example, a proxy server may only allow local computers to access a list of authorized websites.

**Gateway** manages traffic between networks that use different protocols. A gateway is responsible for protocol translation and other interoperability tasks. An IoT gateway device is sometimes employed to provide the connection and translation between devices and the cloud. Because some devices don't contain the network stack required for Internet connectivity, a gateway device acts as a proxy, receiving data from devices and packaging it for transmission over TCP/IP.

A gateway device might be a requirement if devices in the deployment:

- Don't have routable connectivity to the Internet, for example, Bluetooth devices.
- Don't have processing capability needed for transport-layer security (TLS), and as such can't communicate with Google APIs.
- Don't have the electrical power to perform required network transmission.

A **gateway device** might be used even when the participating devices are capable of communicating without one. In this scenario, the gateway adds value because it provides processing of the data across multiple devices before it is sent to the cloud. In that case, the direct inputs would be other devices, not individual sensors. The following tasks would likely be relegated to a gateway device:

- Condensing data to maximize the amount that can be sent to the cloud over a single link
- Storing data in a local database, and then forwarding it on when the connection to cloud is intermittent
- Providing a real-time clock, with a battery backup, used to provide a consistent timestamp for devices that can't manage timestamps well or keep them well synchronized
- Performing IPV6 to IPV4 translation
- Ingesting and uploading other flat-file-based data from the local network that is relevant and associated with the IoT data
- Acting as a local cache for firmware updates

A **protocol** is a standard set of rules that allow electronic devices to communicate with each other. These rules include what type of data may be transmitted, what commands are used to send and receive data, and how data transfers are confirmed. You can think of a protocol as a spoken language. If two hardware devices support the same protocol, they can communicate with each other, regardless of the manufacturer or type of device. For example, an Apple iPhone can send an email to an Android device using a standard mail protocol. The Internet protocol suite, which is used for transmitting data over the Internet, contains dozens of protocols. These protocols may be broken up into four categories:

1. **Link layer** - PPP, DSL, Wi-Fi, etc.
2. **Internet layer** - IPv4, IPv6, etc.
3. **Transport layer** - TCP, UDP, etc.
4. **Application layer** - HTTP, IMAP, FTP, etc.

**Link layer** protocols establish communication between devices at a hardware level. In order to transmit data from one device to another, each device's hardware must support the same link layer protocol.

**Internet layer** protocols are used to initiate data transfers and route them over the Internet. Transport layer protocols define how packets are sent, received, and confirmed.

**Application layer** protocols contain commands for specific applications. For example, a web browser uses HTTPS to securely download the contents of a webpage from a web server. An email client uses SMTP to send email messages through a mail server. Protocols are a fundamental aspect of digital communication. In most cases, protocols operate in the background, so it is not necessary for typical users to know how each protocol works. Still, it may be helpful to familiarize yourself with some common protocols so you can better understand settings in software programs, such as web browsers and email clients.

# Transport Layer, Edge/Fog Computing

Often the terms “fog computing” and “edge computing” are used interchangeably. The latter term predates the former and is construed to be more generic, envisions adding smart data preprocessing capabilities to physical devices such as motors, pumps, or lights. The aim is to do as much of preprocessing of data as possible in these devices, which are termed to be at the *edge* of the network.

In Fog computing, the sensors and network gateways do a part of the data processing and analytics. A **fog architecture** presents a layered approach as shown in figure, which **inserts monitoring, preprocessing, storage, and security layers between the physical and transport layers**.

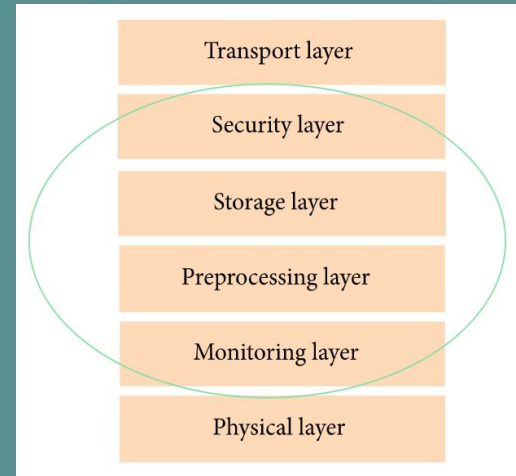
The **monitoring layer** monitors power, resources, responses, and services.

The **preprocessing layer** performs filtering, processing, and analytics of sensor data.

The **temporary storage layer** provides storage functionalities such as data replication, distribution, and storage.

Finally, the **security layer** performs encryption/decryption and ensures data integrity and privacy.

Monitoring and preprocessing are done on the edge of the network before sending data to the cloud.



# Processing layer

**Processing layer (cloud middleware or IoT platforms)** IoT platform or middleware actually drives IoT, enabling you to get all components and data streams connected. On the one side, it links to gateways or devices, and on the other side integrates with third-party applications and systems via APIs. the **processing layer** employing IoT platforms to accumulate and manage all data streams

A fully-fledged **IoT platform** should have the ability to take care of:

- **Connectivity**, ensuring smooth data streaming and interactions between all IoT components;
- **device management**, controls and configures each piece of hardware in the IoT network as well as update software running on devices and gateways;
- **data management**, including data collection, processing, and storage;
- **data analysis** extracting valuable patterns with machine learning, predictive analytics, and other methods;
- **visualization** displaying data findings in the form of charts, graphs, 2D or 3D models;
- **digital twin** creating the virtual representation of a device;
- **IoT app development** platforms provide a workspace with a set of tools and templates to speed up app designing
- **edge / fog computing** the practice of processing and storing data on devices, microcontrollers, gateways, and other IoT nodes to reduce burden for cloud servers.



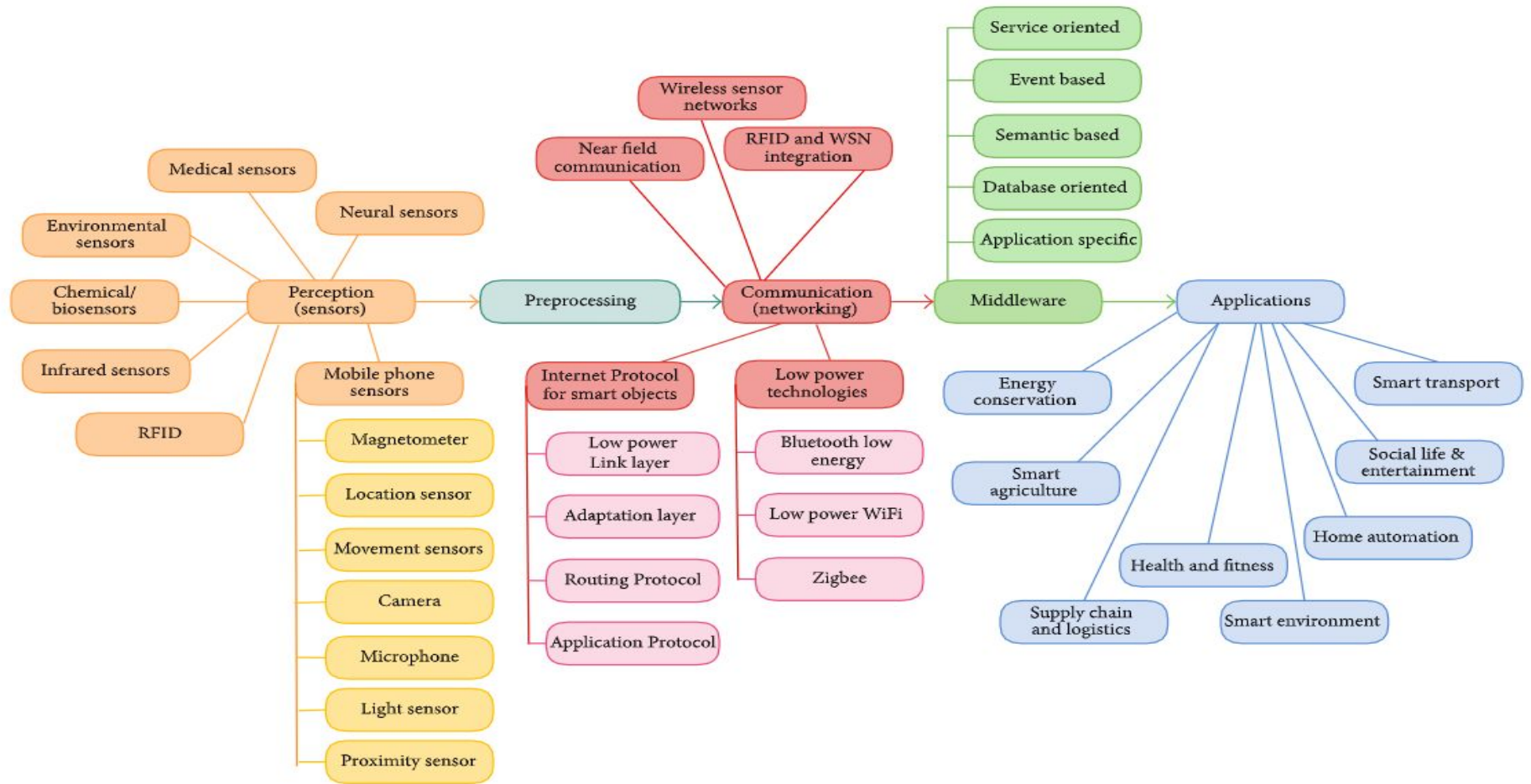


# Application layer

## Application layer (software solutions for users)

IoT software solutions allow end users to gain data insights, monitor and control devices, and, generally, manipulate the physical world through the IoT platform from computers and / or smartphones. Applications can be built on top of the IoT platform or integrate with it through APIs.

the **application layer** delivering solutions like analytics, reporting, and device control to end users.





## **3. Comparing Different IOT Platforms**





There are lots of IoT platforms with half of them focusing on manufacturing and industrial use (IIoT). Other activity areas are energy, mobility, smart cities, and healthcare.

the IoT platform market nonetheless has several major players. The list of the top five, fully-fledged solutions is:

- Amazon Web Service (AWS) IoT platform,
- Cisco IoT,
- Google Cloud IoT,
- IBM Watson IoT platform, and
- Microsoft Azure IoT.

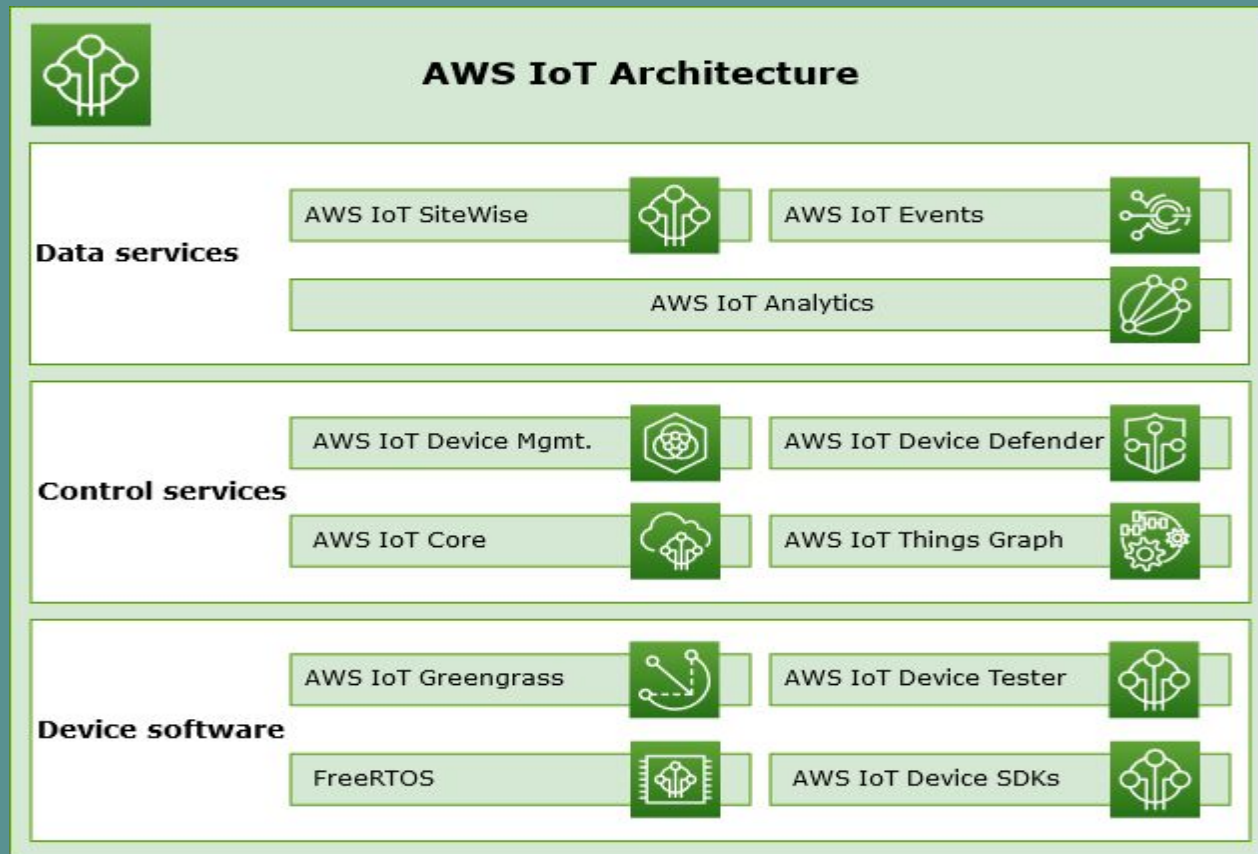
Image source: altexsoft.com

## Key IoT middleware at a glance

	Communication protocols	Key offering and its main functions	Edge computing solutions	Top-3 use cases
	HTTP MQTT WebSockets	AWS IoT Core: ✓ Connectivity ✓ Authentication ✓ Rules engine ✓ Development environment	FreeRTOS edge operating system  IoT GreenGrass edge computing platform	✓ Smart city ✓ Connected home ✓ Agriculture
	MQTT	Cisco IoT Control Center ✓ Mobile connectivity ✓ eSIM as a service ✓ Machine learning to Improve security	Cisco iOX edge development platform  Cisco Edge Intelligence	✓ Connected vehicles ✓ Manufacturing ✓ Smart city
	HTTP MQTT	Google Cloud IoT Core ✓ Connectivity ✓ Device management	Edge TPU chip enabling deployment AI at the edge	✓ Energy ✓ Smart parking ✓ Transportation and logistics
	HTTP MQTT	IBM Watson IoT Platform ✓ Connectivity ✓ Device management ✓ Real-time analytics ✓ Blockchain	IBM Edge Application Manager platform	✓ Manufacturing ✓ Agriculture ✓ Smart buildings
	HTTP MQTT AMQP over WebSockets	Azure IoT Hub ✓ Connectivity ✓ Authentication ✓ Device monitoring ✓ Device management ✓ IoT Edge	IoT Edge as an integral part of IoT Hub	✓ Healthcare ✓ Retail ✓ Manufacturing

# AWS IoT

AWS IoT provides the services that support the devices that interact with the world and the data that passes between them and AWS IoT. AWS IoT is made up of the services that are shown:



# AWS IoT device software

AWS IoT provides this software to support your IoT devices.

**AWS IoT Greengrass** : AWS IoT Greengrass extends AWS IoT to edge devices so they can act locally on the data they generate and use the cloud for management, analytics, and durable storage. With AWS IoT Greengrass, connected devices can run AWS Lambda functions, Docker containers, or both, execute predictions based on machine learning models, keep device data in sync, and communicate with other devices securely – even when they are not connected to the Internet.

**AWS IoT Device Tester** : AWS IoT Device Tester for FreeRTOS and AWS IoT Greengrass is a test automation tool for microcontrollers. AWS IoT Device Tester, test your device to determine if it will run FreeRTOS or AWS IoT Greengrass and interoperate with AWS IoT services.

**AWS IoT Device SDKs** : The AWS IoT Device and Mobile SDKs help you efficiently connect your devices to AWS IoT. The AWS IoT Device and Mobile SDKs include open-source libraries, developer guides with samples, and porting guides so that you can build innovative IoT products or solutions on your choice of hardware platforms.

**FreeRTOS** : FreeRTOS is an open source, real-time operating system for microcontrollers that lets you include small, low-power edge devices in your IoT solution. FreeRTOS includes a kernel and a growing set of software libraries that support many applications. FreeRTOS systems can securely connect your small, low-power devices to AWS IoT and support more powerful edge devices running AWS IoT Greengrass.

# AWS IoT control services

Connect to the following AWS IoT services to manage the devices in your IoT solution.

**AWS IoT Core** : AWS IoT Core is a managed cloud service that enables connected devices to securely interact with cloud applications and other devices. AWS IoT Core can support many devices and messages, and it can process and route those messages to AWS IoT endpoints and other devices. With AWS IoT Core, your applications can interact with all of your devices even when they aren't connected.

**AWS IoT Device Management** : AWS IoT Device Management services help you track, monitor, and manage the plethora of connected devices that make up your devices fleets. AWS IoT Device Management services help you ensure that your IoT devices work properly and securely after they have been deployed. They also provide secure tunneling to access your devices, monitor their health, detect and remotely troubleshoot problems, as well as services to manage device software and firmware updates.

**AWS IoT Device Defender** : AWS IoT Device Defender helps you secure your fleet of IoT devices. AWS IoT Device Defender continuously audits your IoT configurations to make sure that they aren't deviating from security best practices. AWS IoT Device Defender sends an alert when it detects any gaps in your IoT configuration that might create a security risk, such as identity certificates being shared across multiple devices or a device with a revoked identity certificate trying to connect to AWS IoT Core.

**AWS IoT Things Graph** : AWS IoT Things Graph is a service that lets you visually connect different devices and web services to build IoT applications. AWS IoT Things Graph provides a visual drag-and-drop interface for connecting and coordinating interactions between devices and web services, so that you can build IoT applications efficiently.



## AWS IoT data services

Analyze the data from the devices in your IoT solution and take appropriate action by using the following AWS IoT services.

**AWS IoT Analytics :** AWS IoT Analytics lets you efficiently run and operationalize sophisticated analytics on massive volumes of unstructured IoT data. AWS IoT Analytics automates each difficult step that is required to analyze data from IoT devices. AWS IoT Analytics filters, transforms, and enriches IoT data before storing it in a time-series data store for analysis. You can analyze your data by running one-time or scheduled queries using the built-in SQL query engine or machine learning.

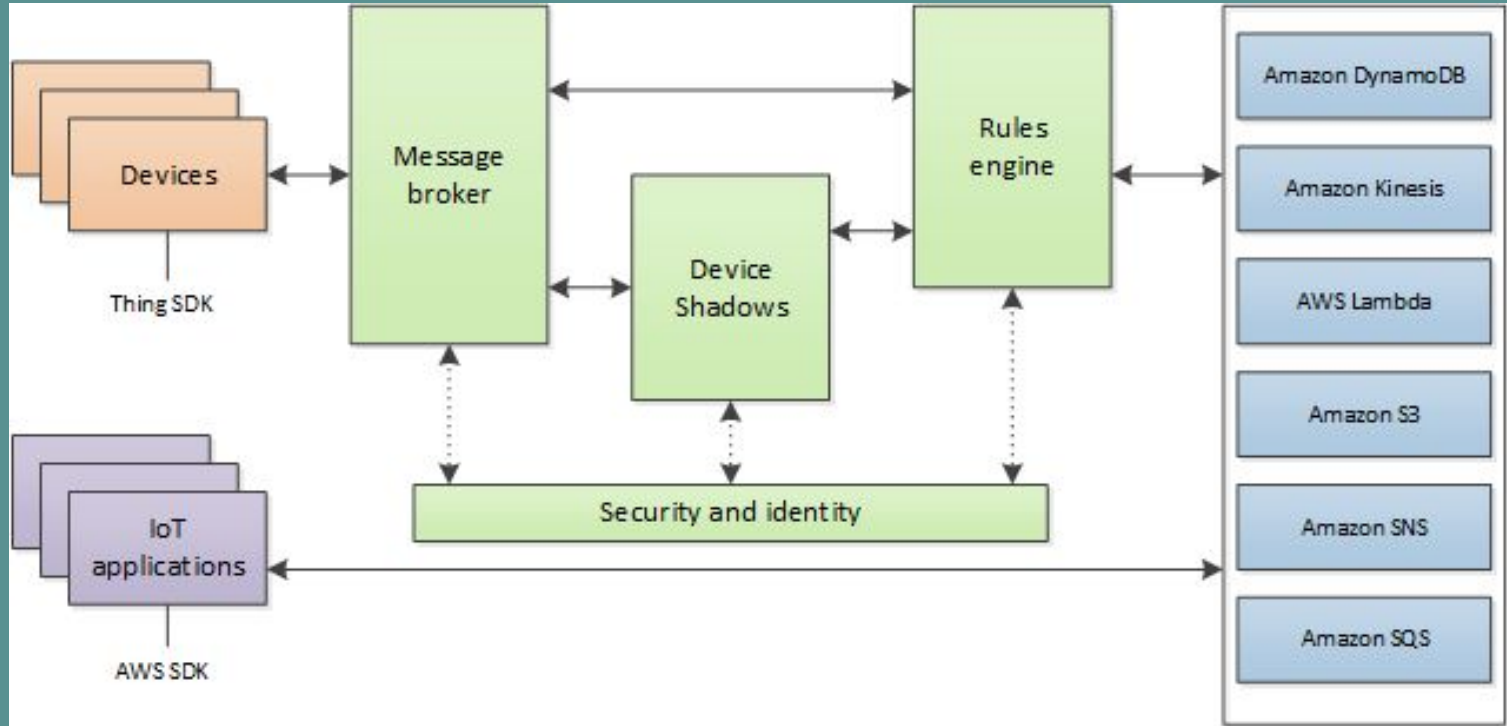
**AWS IoT SiteWise :** AWS IoT SiteWise collects, stores, organizes, and monitors data passed from industrial equipment by MQTT messages or APIs at scale by providing software that runs on a gateway in your facilities. The gateway securely connects to your on-premises data servers and automates the process of collecting and organizing the data and sending it to the AWS Cloud.

**AWS IoT Events :** AWS IoT Events detects and responds to events from IoT sensors and applications. Events are patterns of data that identify more complicated circumstances than expected, such as motion detectors using movement signals to activate lights and security cameras. AWS IoT Events continuously monitors data from multiple IoT sensors and applications, and integrates with other services, such as AWS IoT Core, IoT SiteWise, DynamoDB, and others to enable early detection and unique insights.



# AWS IoT Core services

AWS IoT Core provides the services that connect your IoT devices to the AWS Cloud so that other cloud services and applications can interact with your internet-connected devices.





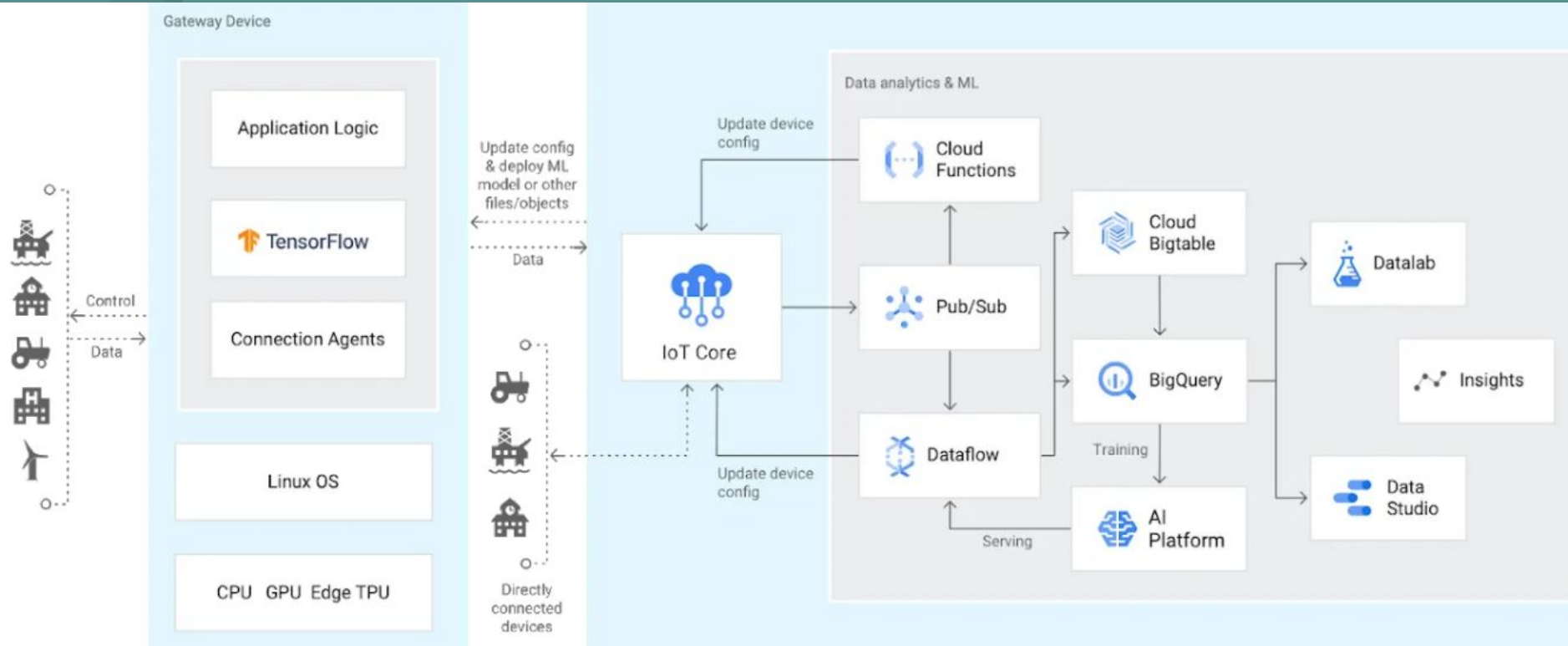
# AWS IoT Core supports these protocols

- MQTT (Message Queuing and Telemetry Transport)
- MQTT over WSS (Websockets Secure)
- HTTPS (Hypertext Transfer Protocol - Secure)
- LoRaWAN (Long Range Wide Area Network)

The AWS IoT Core message broker supports devices and clients that use MQTT and MQTT over WSS protocols to publish and subscribe to messages. It also supports devices and clients that use the HTTPS protocol to publish messages.

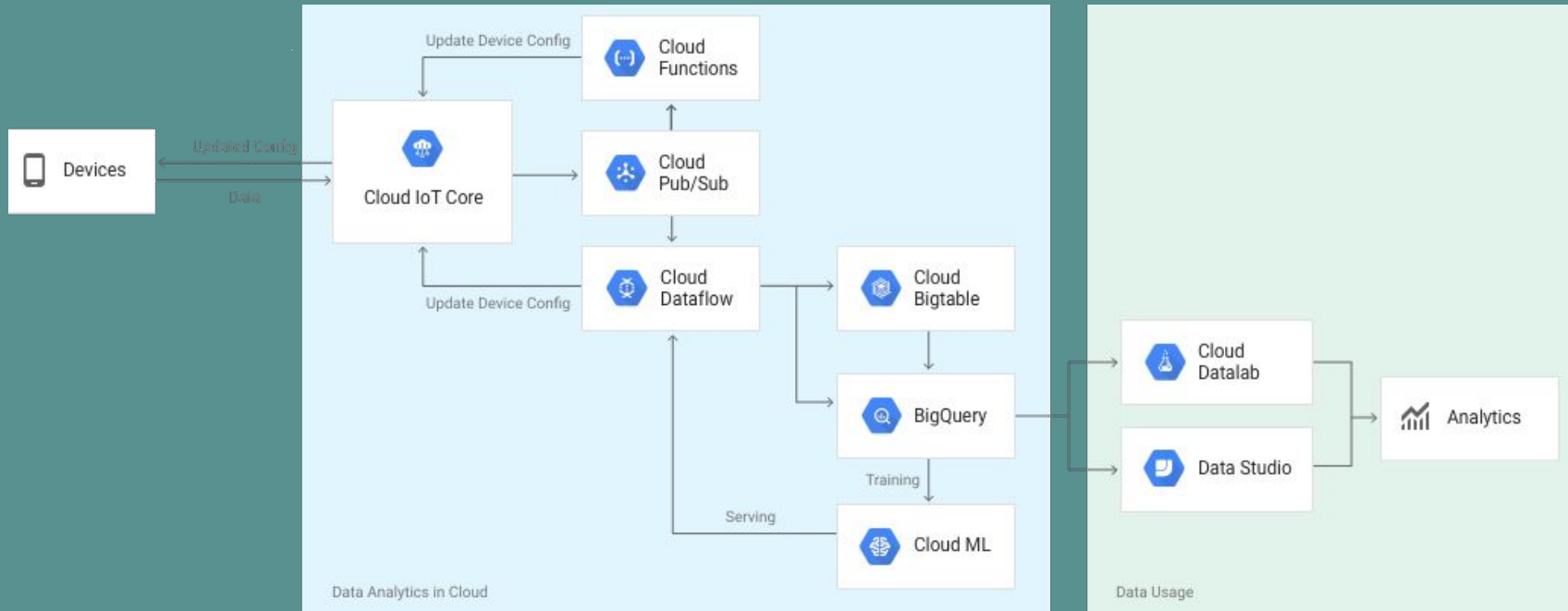
AWS IoT Core for LoRaWAN helps you connect and manage wireless LoRaWAN (low-power long-range Wide Area Network) devices. AWS IoT Core for LoRaWAN replaces the need for you to develop and operate a LoRaWAN Network Server (LNS).

# Google Cloud IoT



# Google Cloud IoT Core

Cloud IoT Core is a fully managed service that allows you to easily and securely connect, manage, and ingest data from millions of globally dispersed devices. Cloud IoT Core, in combination with other services on Google Cloud platform, provides a complete solution for collecting, processing, analyzing, and visualizing IoT data in real time to support improved operational efficiency. This includes registration, authentication, and authorization inside the Google Cloud resource hierarchy as well as device metadata stored in the cloud, and the ability to send device configuration from the service to devices

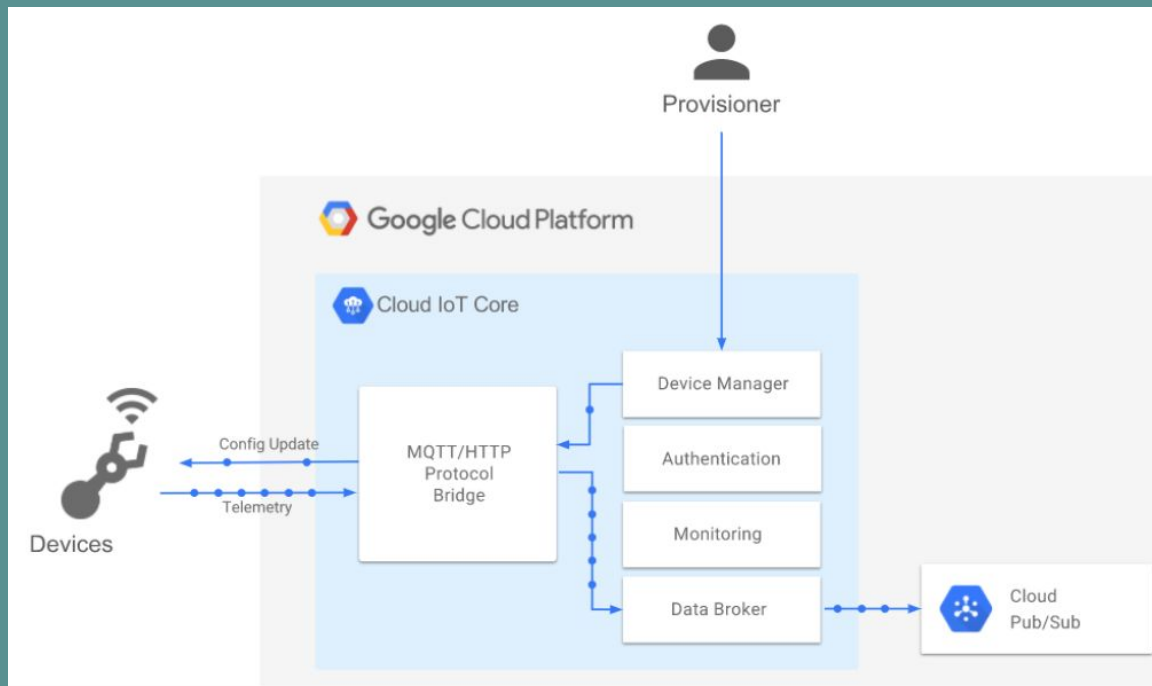


The main components of Cloud IoT Core are the device manager and the protocol bridges:

- A device manager for registering devices with the service, so you can then monitor and configure them
- Two protocol bridges (MQTT and HTTP) that devices can use to connect to Google Cloud Platform

Device telemetry data is forwarded to a Cloud Pub/Sub topic, which can then be used to trigger Cloud Functions. You can also perform streaming analysis with Cloud Dataflow or custom analysis with your own subscribers.

The following diagram summarizes the service components and the flow of data:

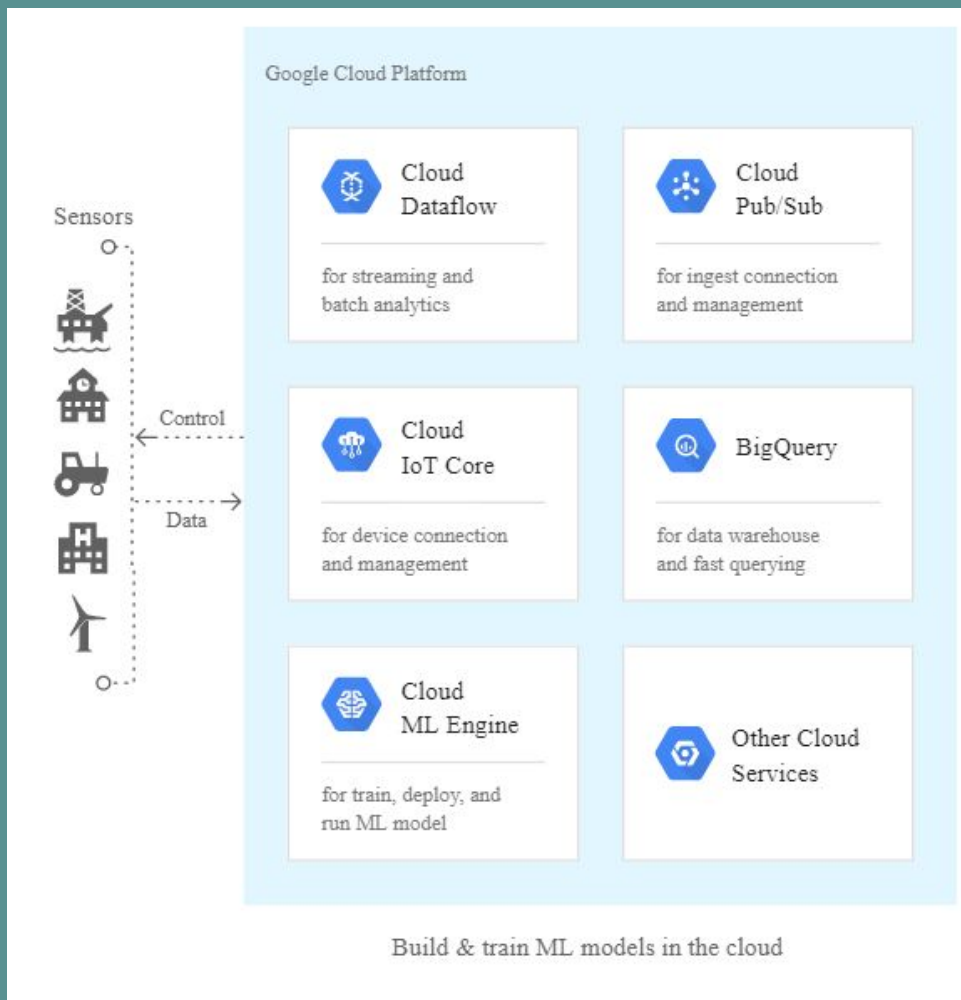


After your IoT project is up and running, many devices will be producing lots of data. You need an efficient, scalable, affordable way to both manage those devices and handle all that information and make it work for you. When it comes to storing, processing, and analyzing data, especially big data, it's hard to beat the cloud.

The following diagram shows the various stages of IoT data management in Google Cloud.

## Ingestion

Ingestion is the process of importing information from devices into Google Cloud services. Google Cloud provides different ingestion services, depending on whether the data is telemetry or operational information about the devices and the IoT infrastructure.





## Pipeline processing tasks

Pipelines manage data after it arrives on Google Cloud, similar to how parts are managed on a factory line. This includes tasks such as:

- **Transforming data.** You can convert the data into another format, for example, converting a captured device signal voltage to a calibrated unit measure of temperature.
- **Aggregating data and computing.** By combining data you can add checks, such as averaging data across multiple devices to avoid acting on a single, spurious, device. You can ensure that you have actionable data if a single device goes offline. By adding computation to your pipeline, you can apply streaming analytics to data while it is still in the processing pipeline.
- **Enriching data.** You can combine the device-generated data with other metadata about the device, or with other datasets, such as weather or traffic data, for use in subsequent analysis.
- **Moving data.** You can store the processed data in one or more final storage locations.

## Pub/Sub

Pub/Sub provides a globally durable message ingestion service. By creating *topics* for streams or channels, you can enable different components of your application to *subscribe* to specific streams of data without needing to construct subscriber-specific channels on each device. Pub/Sub also natively connects to other Google Cloud services, helping you to connect ingestion, data pipelines, and storage systems.

Pub/Sub can act like a shock absorber and rate leveller for both incoming data streams and application architecture changes. Many devices have limited ability to store and retry sending telemetry data. Pub/Sub scales to handle data spikes that can occur when swarms of devices respond to events in the physical world, and buffers these spikes to help isolate them from applications monitoring the data.

## Dataflow

Dataflow provides the open Apache Beam programming model as a managed service for processing data in multiple ways, including batch operations, extract-transform-load (ETL) patterns, and continuous, streaming computation. Dataflow can be particularly useful for managing the high-volume data processing pipelines required for IoT scenarios. Dataflow is also designed to integrate seamlessly with the other Google Cloud services you choose for your pipeline.



## Data storage

Data from the physical world comes in various shapes and sizes. Google Cloud offers a range of storage solutions from unstructured blobs of data, such as images or video streams, to structured entity storage of devices or transactions, and high-performance key-value databases for event and telemetry data.

## Storing application data in Datastore and Firebase

When you need to make state or telemetry data available to mobile or web apps, you can store processed or raw data in structured but schemaless databases, such as Datastore and Firebase Realtime Database, where you can represent IoT device data as domain or application level objects.

## Storing state in IoT Core

Some device state might be directly connected to the hardware. For example, when you check the state of a digital GPIO pin, it reads as HIGH or LOW, based on the physical reading of the voltage on the pin.

Other device state might exist at the application layer. For example, recording-audio might have a state condition of true or false, related to whether the application is sampling from the mic or writing to disk. At the hardware level, the mic itself might be left on.

From the software perspective, the application code running on the device maintains the source of truth. It is often valuable, even required, for other software to read the device's last known state. Given that IoT devices can spend some time in low-power sleep mode and might exist on particularly unreliable networks, it's often useful to store some of a device's state in the cloud. That way, state data can be made available even when the devices themselves are temporarily offline.

The last known device state can be reported and stored in IoT Core for retrieval by applications. State information sent over MQTT or HTTP is persisted in IoT Core and is available in the cloud, even if the device has disconnected or gone off line.

## **Analytics**

Performing analytics on data obtained through IoT sources is often the entire purpose of instrumenting the physical world. After streaming data has been analyzed in a processing pipeline, it will begin to accumulate. Over time, this data provides a rich source of information for looking at trends, and can be combined with other data, including data from sources outside of your IoT devices.

## **Machine Learning**

IoT data is often inherently multi-dimensional and noisy by nature. These attributes can make it hard to extract insight by using conventional analytics techniques. However, this nuance and complexity is often where Deep Neural Networks excel. Tensorflow is a leading open source machine learning framework, and on Google Cloud you can apply Tensorflow in a distributed and managed training service through AI Platform.

## **BigQuery and Datalab**

BigQuery provides a fully managed data warehouse with a familiar SQL interface, so you can store your IoT data alongside any of your other enterprise analytics and logs. The performance and cost of BigQuery means you might keep your valuable data longer, instead of deleting it just to save disk space.

Datalab is an interactive tool for large-scale data exploration, analysis, and visualization. IoT data can end up being useful for multiple use cases, depending on which other data it's combined with. Datalab lets you interactively explore, transform, analyze, and visualize your data using a hosted online data workbench environment based on the open source Jupyter project.



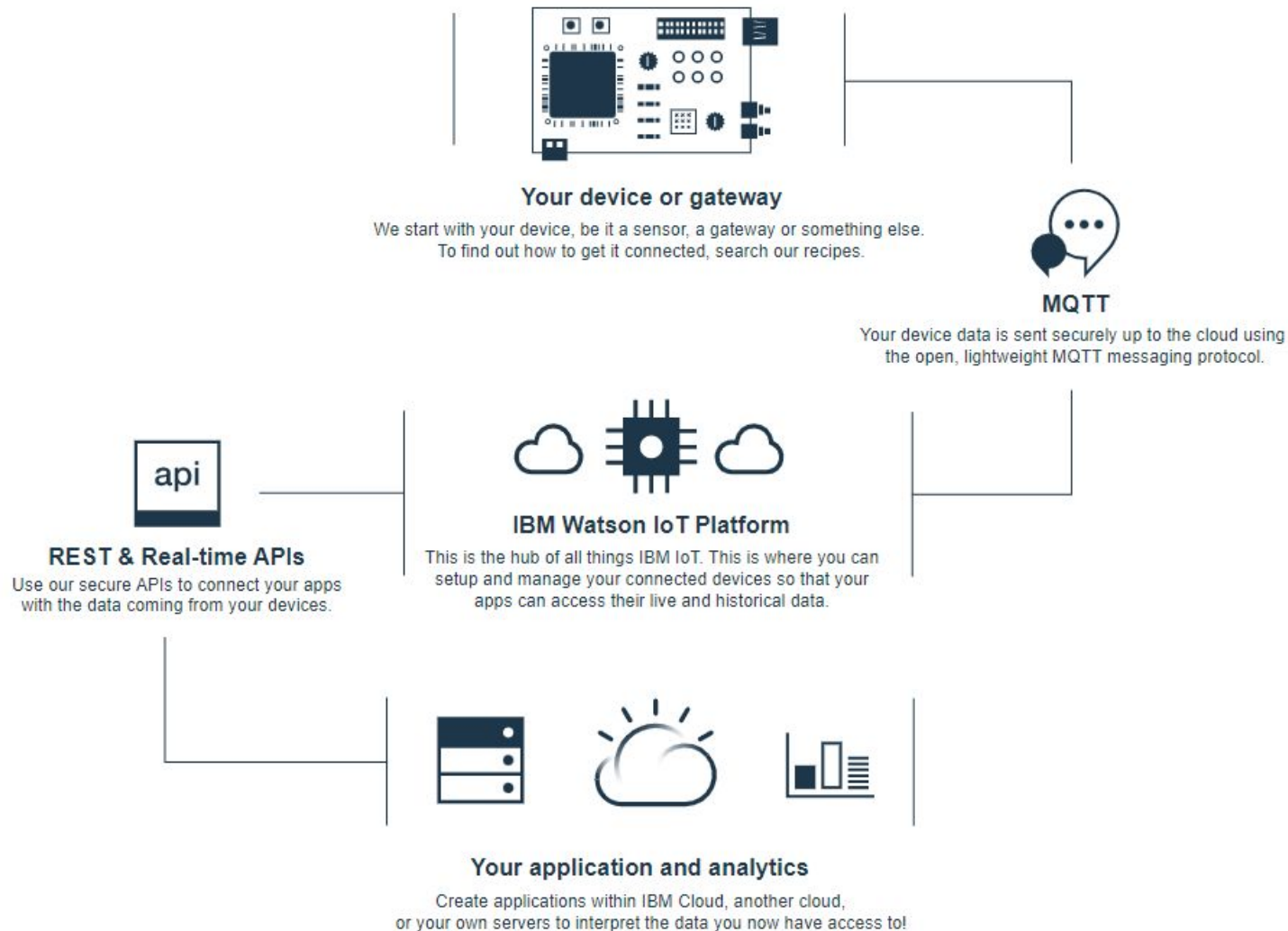
## Time Series dashboards with Cloud Bigtable

Certain types of data need to be quickly sliceable along known indexes and dimensions for updating core visualizations and user interfaces. Bigtable provides a low-latency and high-throughput database for NoSQL data. Bigtable provides a good place to drive heavily used visualizations and queries, where the questions are already well understood and you need to absorb or serve at high volumes.

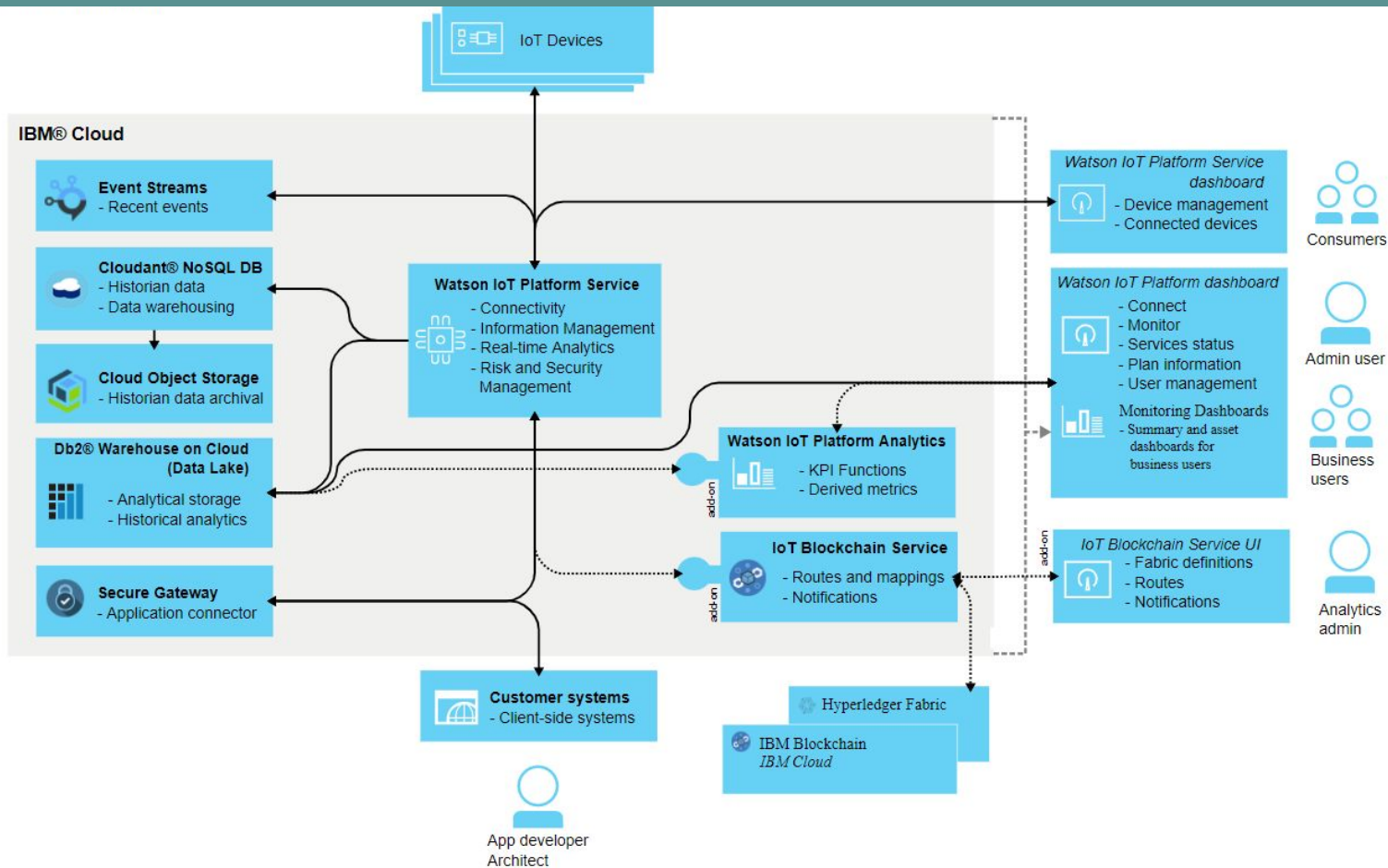
Compared to BigQuery, Bigtable works better for queries that act on rows or groups of consecutive rows, because Bigtable stores data by using a row-based format. Compared to Bigtable, BigQuery is a better choice for queries that require data aggregation.



# IBM Watson IOT



# IBM Watson IoT Platform



## The following table lists the IBM Cloud services that make up the Watson IoT Platform solution.

Service	Details	Access
Platform Service	A secure, smart, scalable platform service that is used as the IoT device message broker for secure device registration, real-time analytics, and more.	Dashboard, Monitoring dashboards, API, App integration, IBM Cloud service integration
Watson IoT Platform on Blockchain (add-on)	<b>Important:</b> IBM® Watson™ IoT Platform on Blockchain is no longer available for purchase, but advancements in cloud and blockchain technologies make it straightforward to send your data to your blockchain by using low-cost Cloud Functions in your IBM Cloud account. To get started, see the following IBM® Developer tutorial: <a href="#">Use cloud functions to send your data to your blockchain</a> . Add-on component of Watson IoT Platform that enables your IoT resources to participate in blockchain business networks.	API, dashboard, Watson IoT Platform on Blockchain UI.
Analytics Service (add-on)	Add-on component that extends the functionality of IBM® Watson™ IoT Platform to include analytics features. Line-of-business users can use Analytics Service to interact with their IoT data and derive essential performance indicators.	Dashboard, API
IoT Registration Service	A service to maintain a registry of IoT appliances and consumers and to seamlessly associate these	Registration UI in Platform Service dashboard, API
Watson IoT Platform starter	A Node.js-based demonstration application that showcases device registration, consumer mapping, device control, and more. The starter includes a sample app, a sample mobile app, and a set of sample Node-RED flows. <b>Important:</b> The Watson IoT Platform starter is only available in a non-production environment. For more information, see the <a href="#">overview</a> topic or contact the named Administrator contact for your IBM service offering.	Starter API, Sample app web console, Mobile app
Cloudbant NoSQL DB	A managed NoSQL database service that captures device measurements and events and stores them for use by real-time applications. A hosted and fully managed database-as-a-service (DBaaS) that handles a variety of data types, such as JSON, full-text, and geospatial. Auto schema generation and device data aggregation. Combine business data with device data.	API, IoT Platform integration

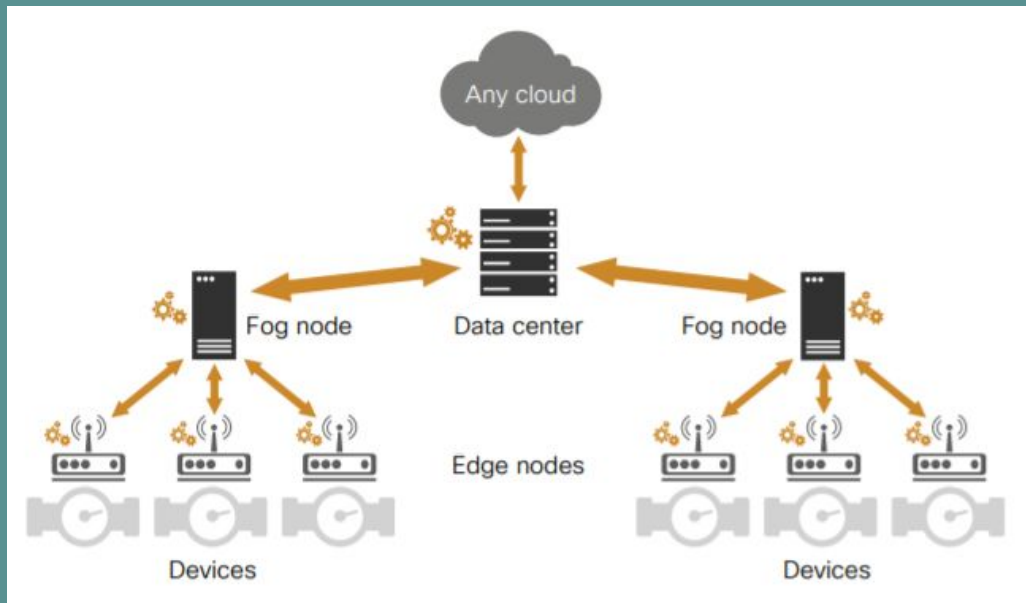




Service	Details	Access
Db2® Warehouse on Cloud	The Watson IoT Platform data lake. Intermediate length storage and analytics of your IoT data. A cloud-based SQL database that includes Netezza® technology for robust analytics capabilities.	Dashboard, Clients, API
IBM® Cloud Object Storage	Unstructured cloud data storage for long time storage. Scalable and fully managed storage. Cross-regional and regionally resilient.	API, Watson IoT Platform Dashboard (Bucket creation)
IBM Cloud App ID	Authentication for mobile and web apps that connect to your Watson IoT Platform solution. Supports Facebook and Google+ authentication.	OAuth server, Access via IBM support.
IBM® Secure Gateway for IBM Cloud	A secure way to access your on-premises or cloud data from your IBM Cloud application by using a secure passage.	Contact IBM support to configure.
IBM® Event Streams for IBM Cloud	A scalable, high-throughput message bus. Wire micro-services together by using open protocols. Connect stream data to analytics to realize powerful insights. Feed event data to multiple applications to react in real time. Bridge to your on-premise messaging infrastructure to create a hybrid cloud messaging solution.	Watson IoT Platform Dashboard (Message creation), Admin API, Kafka REST API

# Cisco IOT

The **Cisco Kinetic platform** is an IoT data fabric, designed to extract data from your connected devices, compute anywhere in a distributed network, and move data to the various applications where it can be used to drive meaningful business outcomes.





## The three key modules of the Cisco Kinetic platform:



**Gateway Management  
Module (GMM)** – to  
provision, monitor, and  
manage gateways at scale



**Edge & Fog Processing  
Module (EFM)** – to enable  
computing on distributed  
nodes of the network



**Data Control Module (DCM)**  
– to enforce policy and get  
the right data to the right  
apps at the right time



## Cisco Kinetic EFM enables you to:

- Extract data from diverse, distributed devices — Connect a broad range of devices and sensors across your organization and get usable data from them.
- Compute data in a distributed IoT environment — Apply rules and logic on data in motion or data at rest. Perform micro-processing where needed, from the edge to the cloud.
- Move data to diverse, distributed applications — Get the right data to the right applications where you can use it to create business value. You can use EFM in both connected environments and industrial environments with no internet connectivity.

# Microsoft Azure IoT

The Azure Internet of Things (IoT) is a collection of Microsoft-managed cloud services that connect, monitor, and control billions of IoT assets. In simpler terms, an IoT solution is made up of one or more IoT devices that communicate with one or more back-end services hosted in the cloud.

Microsoft recommends using Azure IoT Central, which is a fully managed SaaS (software-as-a-service) solution. It abstracts the technical choices and lets you focus on your solution exclusively. This simplicity comes with a tradeoff in being less customizable than a PaaS-based solution.

At a high level, there are two ways to process telemetry data, hot path and cold path. The difference has to do with requirements for latency and data access.

- The hot path analyzes data in near-real-time, as it arrives. In the hot path, telemetry must be processed with very low latency. The hot path is typically implemented using a stream processing engine. The output may trigger an alert, or be written to a structured format that can be queried using analytical tools.
- The cold path performs batch processing at longer intervals (hourly or daily). The cold path typically operates over large volumes of data, but the results don't need to be as timely as the hot path. In the cold path, raw telemetry is captured and then fed into a batch process.

This reference architecture shows a recommended architecture for IoT applications on Azure using PaaS (platform-as-a-service) components.

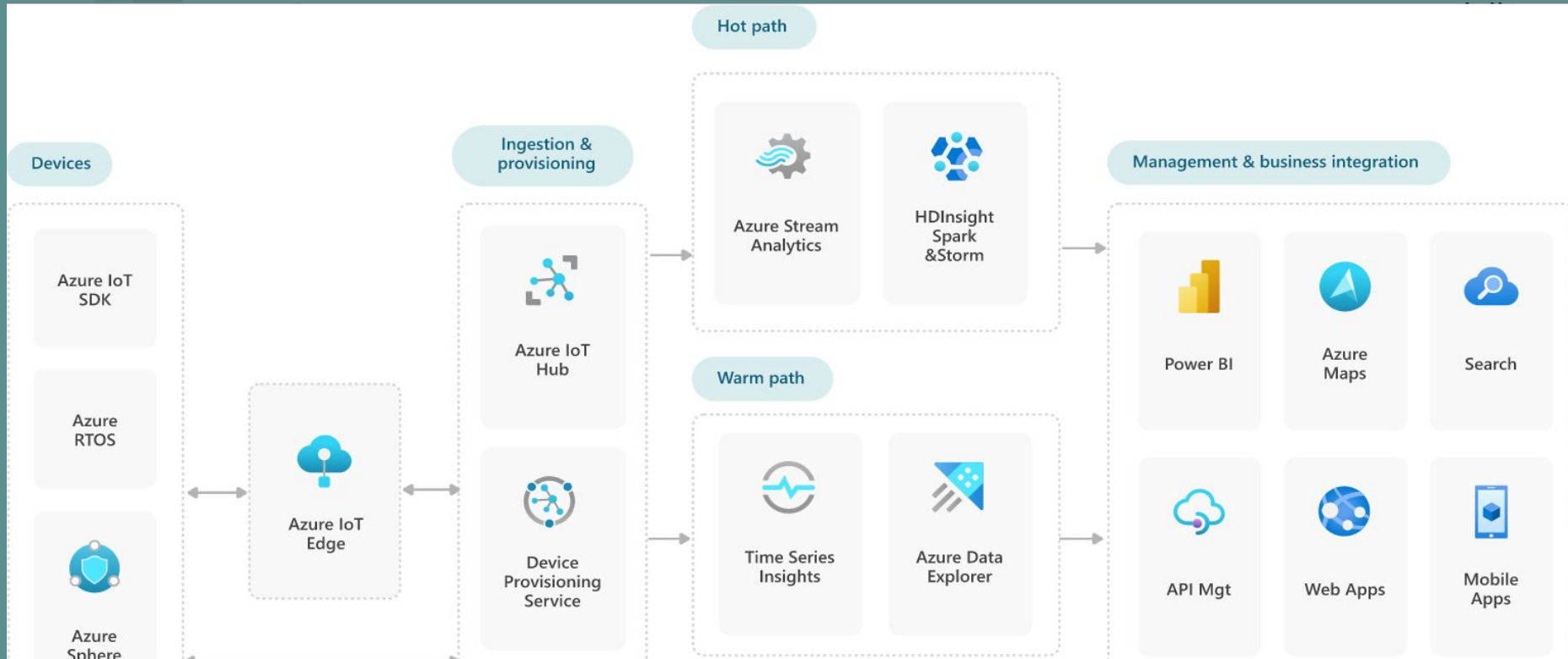


Diagram of the architecture

# This architecture consists of the following components.

Some applications may not require every component listed here.

**IoT devices.** Devices can securely register with the cloud, and can connect to the cloud to send and receive data. Some devices may be edge devices that perform some data processing on the device itself or in a field gateway. We recommend Azure IoT Edge for edge processing.

**Cloud gateway.** A cloud gateway provides a cloud hub for devices to connect securely to the cloud and send data. It also provides device management, capabilities, including command and control of devices. For the cloud gateway, we recommend IoT Hub. IoT Hub is a hosted cloud service that ingests events from devices, acting as a message broker between devices and backend services. IoT Hub provides secure connectivity, event ingestion, bidirectional communication, and device management.

**Device provisioning.** For registering and connecting large sets of devices, we recommend using the IoT Hub Device Provisioning Service (DPS). DPS lets you assign and register devices to specific Azure IoT Hub endpoints at scale.

**Stream processing.** Stream processing analyzes large streams of data records and evaluates rules for those streams. For stream processing, we recommend Azure Stream Analytics. Stream Analytics can execute complex analysis at scale, using time windowing functions, stream aggregations, and external data source joins. Another option is Apache Spark on Azure Databricks.

**Machine learning** allows predictive algorithms to be executed over historical telemetry data, enabling scenarios such as predictive maintenance. For machine learning, we recommend Azure Machine Learning.

**Warm path** storage holds data that must be available immediately from device for reporting and visualization. For warm path storage, we recommend Cosmos DB. Cosmos DB is a globally distributed, multi-model database.

**Cold path** storage holds data that is kept longer-term and is used for batch processing. For cold path storage, we recommend Azure Blob Storage. Data can be archived in Blob storage indefinitely at low cost, and is easily accessible for batch processing.


**Data transformation** manipulates or aggregates the telemetry stream. Examples include protocol transformation, such as converting binary data to JSON, or combining data points. If the data must be transformed before reaching IoT Hub, we recommend using a protocol gateway (not shown). Otherwise, data can be transformed after it reaches IoT Hub. In that case, we recommend using Azure Functions, which has built-in integration with IoT Hub, Cosmos DB, and Blob Storage.

**Business process.** integration performs actions based on insights from the device data. This could include storing informational messages, raising alarms, sending email or SMS messages, or integrating with CRM. We recommend using Azure Logic Apps for business process integration.

**User management** restricts which users or groups can perform actions on devices, such as upgrading firmware. It also defines capabilities for users in applications. We recommend using Azure Active Directory to authenticate and authorize users.

**Security monitoring.** Azure Security Center for IoT provides an end-to-end security solution for IoT workloads and simplifies their protection by delivering unified visibility and control, adaptive threat prevention, and intelligent threat detection and response across workloads from leaf devices through Edge as well as up through the clouds.

**Synapse Link** is the Microsoft preferred solution for analytics on top of Cosmos DB data.



**Azure Synapse Link** for Azure Cosmos DB is a cloud-native hybrid transactional and analytical processing (HTAP) capability that enables you to run near real-time analytics over operational data in Azure Cosmos DB. Azure Synapse Link creates a tight seamless integration between Azure Cosmos DB and Azure Synapse Analytics.

Using Azure Cosmos DB analytical store, a fully isolated column store, Azure Synapse Link enables no Extract-Transform-Load (ETL) analytics in Azure Synapse Analytics against your operational data at scale. Business analysts, data engineers and data scientists can now use Synapse Spark or Synapse SQL interchangeably to run near real-time business intelligence, analytics, and machine learning pipelines. You can achieve this without impacting the performance of your transactional workloads on Azure Cosmos DB.

The following image shows the Azure Synapse Link integration with Azure Cosmos DB and Azure Synapse Analytics:

### Transactional Store

Row store optimized for transactional reads and writes

### Analytical Store

Column store optimized for analytical queries

Operational Data



Auto-Sync



Cloud-Native HTAP



Azure Synapse Link



APACHE  
Spark

SQL

Machine learning

Big data analytics

BI Dashboards



Azure Cosmos DB



Azure Synapse Analytics



The following image shows transactional row store vs. analytical column store in Azure Cosmos DB:

Core (SQL) API

Azure Cosmos DB API  
for Mongo DB

Gremlin

cassandra

Table API

Container

Transactional Store (Row Store)

ID	Product Name	Product Code	Product Category
1	Product1	123	Books
2	Product2	250	Equipment
3	Product3	380	Equipment



Auto-Sync

Analytical Store (Column Store)

ID	Product Name	Product Code	Product Category
1	Product1	123	Books
2	Product2	250	Equipment
3	Product3	380	Equipment



1	Product1	123	Books	2	Product2	250	Equipment	3	Product3	380	Equipment
---	----------	-----	-------	---	----------	-----	-----------	---	----------	-----	-----------

Disk- Logical Representation

1	2	3	Product1	Product2	Product3	123	250	380	Books	Equipment	Equipment
---	---	---	----------	----------	----------	-----	-----	-----	-------	-----------	-----------

Disk- Logical Representation





# Conclusion

# How to choose the best IoT platform

IoT platforms features basically overlap. Along with a rich functionality, all of them offer:

- high scalability, fitting the needs of any business, from startups to enterprises with millions of devices
- built-in security for every layer of an IoT system
- tech support and detailed documentation on their products.

The question is, how do you choose the best option among five equal worth leaders?

There are several things to consider.

## Pricing and free tier

All platforms define a pay-as-you-go model, so the total price will depend on volume of use — or the number of messages or megabytes exchanged, devices connected, actions executed, and so on. To address the pricing model complexity, Amazon, Google, and Microsoft offer their customers a pricing calculator located on their official website. But often the only way to estimate your IoT project cost is to run it for a month and pay the bill. AWS tends to be the most expensive option, followed by Google and Microsoft Azure as the cheapest of the trio, for IBM and Cisco, their pricing details are available on request.

## Hardware compatibility

IoT infrastructure involves numerous devices. You must check whether your existing hardware is compatible with a particular platform. For example, Cisco IoT software is designed to work smoothly with Cisco IoT hardware.

On the other hand, Amazon and Microsoft Azure view themselves as hardware agnostic and support mist of IoT devices.

## Domain expertise

Though all platforms work across several industries, each of them can outperform others in a particular domain. For example running a big enterprise with complex infrastructure and millions of sensors, will lead to choose IBM, boasting exclusive expertise in managing industrial equipment.

AWS is a good fit to implement a smart home scenario.

Cisco will bring most value to businesses dealing with connected vehicles, also sa good start for companies that build their IoT infrastructure from the ground up, offering all necessary hardware, software, and connectivity services.

Microsoft is most efficient in the Internet of Medical Things (IoMT)

Google is on the other hand a good choice when it comes to energy, transportation, and building smart parks.



Platform	Pricing plan	Free tier	Free tier across additional IoT services
Amazon IoT Core	Connectivity: <b>\$0.08</b> per million minutes of connection  Messaging: <b>\$0.7-1</b> per million messages (the more messages the cheaper)  Device shadow and registry: <b>\$1.25</b> per million operations  Rules engine: <b>\$0.15</b> per million rules triggered/actions executed	Available for 12 months  ✓ 2, 250,000 minutes of connection ✓ 500,000 messages ✓ 225,000 registry or Device shadow operations ✓ 225,000 rules triggered and 225, 000 actions executed	Available for 12 months  ✓ Device Management: 50 remote actions per month ✓ AWS Greengrass: 3 devices ✓ AWS IoT Events: 250,000 message evaluations per month ✓ AWS IoT Analytics: 100 MB of data processes and 10 GB of data storage
Cisco IoT Control Center	Details are available at request	No free tier	No free tier
Google Cloud IoT Core	<b>\$0.00045-0.0045</b> per MB of data exchanged (the more data the cheaper)	First 250 MB	✓ 12-month free trial with <b>\$300</b> credit to spend on any Google Cloud Services ✓ The large suite of always free resources
IBM Watson IoT Platform	Starts at <b>\$500</b> per unit/ per month	No free tier	No free tier
Azure IoT Hub	Basic tier: <b>\$10</b> to 500 per unit/per month  Standard tier: <b>\$25</b> to 2500 per unit/per month  The price within the tier depends on the number of messages exchanged per day (up to 400,000, 6 million or 300 million)	Up to 8,000 messages per day and up to 500 registered devices	✓ 12-month free trial of popular Azure services ✓ <b>\$200</b> credit to explore Azure for 30 days ✓ 25+ always free services

## Sources:

<https://www.hindawi.com/journals/jece/2017/9324035/>

<https://www.altexsoft.com/blog/iot-platforms/>

<https://www.techterms.com>

<https://www.forcepoint.com/cyber-edu/osi-model>

<https://docs.aws.amazon.com/iot/latest/developerguide/aws-iot-how-it-works.html>

<https://cloud.google.com/architecture>

<https://cloud.google.com/iot/docs/concepts/overview>

<https://cloud.google.com/architecture/iot-overview>

<https://cloud.google.com/iot-core>

<https://codelabs.developers.google.com/codelabs/cloud-iot-core-overview#0>

<https://www.cisco.com/c/dam/en/us/solutions/collateral/internet-of-things/kinetic-datasheet-efm.pdf>

<https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/iot>

<https://docs.microsoft.com/en-us/azure/cosmos-db/analytical-store-introduction>

<https://docs.microsoft.com/en-us/azure/cosmos-db/synapse-link>

<https://www.ibm.com/docs/en/watson-iot-platform?topic=features-product-architecture>

<https://internetofthings.ibmcloud.com/>

<https://www.forcepoint.com/cyber-edu/osi-model>