**15**

**The Internet of Things Architecture**

As internet connectivity is increasing, there is an increasing number of small devices everywhere with small memory and compute capacities. These sensors connect various physical entities, such as your home alarm, thermal sensors, and car, and the data from millions of these connected devices needs to be collected and analyzed. For example, weather data collected from multiple sensors can be utilized to forecast weather for wind energy and farming. There are billions of connected devices in homes, factories, oil wells, hospitals, cars, and thousands of other places that are fueling digital transformation, generating huge volumes of data and growing exponentially.

The **Internet of Things** (**IoT**) is much more than just a collection of data from sensors and devices. Comprehensive IoT solutions include devices, local data collection and analysis, and cloud services to collect, store, and analyze data. You need integration with devices to operate even without internet connectivity, and when connections are possible, they can connect to report data and status.

IoT solutions require device connectivity and messaging services, fleet management, and device update services. They need to ensure device and fleet security, including audit and anomaly detection; on top of that, you should perform analytics functionality for noisy and intermittent IoT data. To get full business value in real time, you should build machine learning models and then deploy them back to devices to optimize processes and outcomes.

In this chapter, you will learn about the following topics to handle and manage your connected IoT devices:

* What is the Internet of Things?
* Components of IoT architecture
* Managing IoT devices
* Connecting and controlling IoT devices
* Performing analytics on IoT data
* IoT in the cloud
* Building an industrial IoT solution
* Implementing a digital twin

By the end of this chapter, you will know how to design IoT solution architecture. You will learn about the various component of IoT solutions, including device management, device connectivity, device security, and getting insights from IoT data through analytics and machine learning.

**What is the Internet of Things?**

Imagine if you knew the state of everything and could reason on top of that data; what problems would you solve? IoT is all about telling the state of everything, everywhere. IoT refers to a network ecosystem of physical devices with an IP address and connected to the internet. While IoT devices are multiplying in number, the complexity of leveraging your IoT devices currently is growing with them.

You need to ingest data from IoT sensors, store it for analysis via streaming, and provide results quickly. There are large numbers of devices in homes, healthcare facilities, factories, cars, and many other places. You increasingly need solutions to connect with these devices and to collect, store, and analyze device data to improve operational efficiency.

IoT solutions provide you with the ability to collect data from all those deceives and acquire insights. IoT is a critical enabler for emerging technologies like AI/ML, robotics, video analytics, mobile, and voice. IoT is at the heart of these emerging technologies as access to device data is crucial to train machine learning models, deliver intelligence, and drive business efficiency. There are multiple industrial use cases that enterprises are solving using IoT, such as:

* **Optimize manufacturing**: By capturing machine performance data, you can improve the performance and productivity of industrial processes. You can gain insight into machine performance and replace parts before they become broken with predictive maintenance.
* **Healthcare**: With IoT, you can bring healthcare to patients anytime and everywhere. Doctors can remotely monitor patient health and take action in case of any health alert. Now everyone is using wellness applications embedded in wearable devices such as the Apple Watch and Fitbit to be better aware of their health and share health data directly with their primary care.
* **Inventory tracking**: IoT helps to maintain just-in-time inventory and optimize warehouse operation costs. You can track inventory levels and auto-place replenishment orders. With IoT sensors, you can pretty much automate entire warehouse operations from receiving to replenishment, packing, and shipping.
* **Connected homes**: Using IoT-enabled devices such as smart switches, smart thermostats, and smart cameras, you can enhance user experiences in homes, buildings, and cities. You can operate the entire facility and optimize capacity using intelligent devices or equip homes with smart security devices to help you monitor your home from anywhere, anytime.
* **Agriculture**: This is an important area for human survival, and IoT sensors for humidity, weather, and temperature are helping to grow healthier crops with greater efficiency. You can determine when to water the crop by combining data from humidity sensors and weather forecasts.
* **Energy efficiency**: You can manage energy resources more efficiently with IoT, such as monitoring wind farm and solar farm energy production in real time, and plan maintenance.
* **Transforming transportation**: IoT is helping shape the future of transportation with connected and autonomous vehicles. The most popular use case you can see is Tesla cars, equipped with IoT sensors and powering full self-drive features by collecting thousands of data points in real time.
* **Enhancing safety**: IoT devices help improve safety in the home, the office, and the factory floor by continuously monitoring and providing alerts ahead of any equipment malfunction or security incident to take immediate action.

IoT strategies empower the enterprise with the intelligence needed to build new services and improve products and services over time, enjoying better relationships with their customers. Data-driven discipline results in making faster, more intelligent decisions that lead to revenue growth and greater operational efficiency. With IoT, organizations either become more efficient and lower costs or build entirely new services and products, driving new business. These are some of the most critical challenges to any IoT device architecture. You need to ensure the security and management of devices in addition to data collection and performing insights on top of it. Let's learn more about the various attributes of IoT architecture.

**Components of IoT architecture**

In the use cases mentioned in the previous section, organizations have many devices across multiple product lines. They needed an architecture to ingest various telemetry measures and attributes to support real-time consumption by users and service applications. At a high level, IoT architecture consists of three components, as shown in the following diagram:

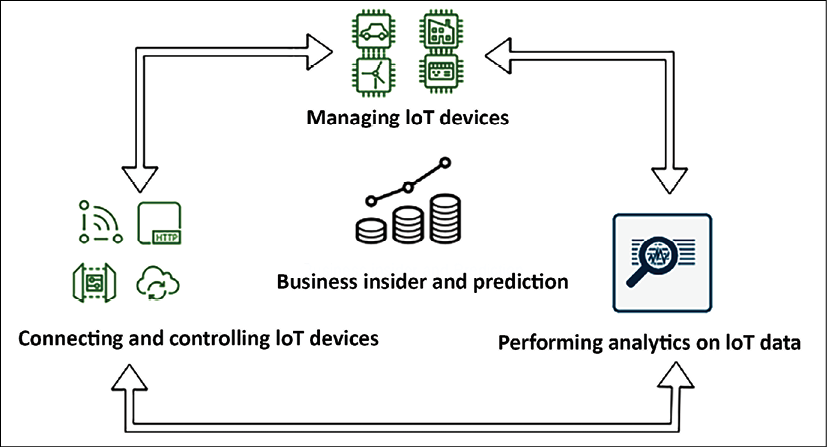


Figure 15.1: IoT architecture cycle

As shown in the preceding diagram, the IoT architecture cycle has three elements:

1. **Managing IoT device software**: To build IoT solutions, you need to deploy a large number of devices, ranging from thousands to millions. These devices should have the ability to generate the data required for the use case and perform an operation at the edge as required. The first thing you need to figure out is, *how can I build devices that operate at the edge?*
2. **Handling IoT device connectivity and control**: Managing millions of devices and securing them is a tedious task. On top of that, you want them to be updated with the latest software/firmware or they may need to be maintained in different versions. You also need to group for ease of management and make sure they are securely connected. The second thing you need to understand is how to control, manage, and secure your devices at scale.
3. **Performing IoT analytics services**: Once your devices are deployed and connected, you need to collect a large volume of data arriving at high velocity and gain insights to derive business value.

In the final action, you need to design how to make sense of your IoT data and take the appropriate actions.

So now, to build an IoT architecture, you have three questions to answer from the architecture layers mentioned above. While deep diving into the architecture, you want to embed examples from available technology choices in each of the preceding elements. Let's explore the first element of IoT architecture in more detail to learn IoT device software management in greater depth.

**Managing IoT devices**

When it comes to IoT devices, you want to understand how to build and operate intelligent device software at the edge. With the increased availability of the internet, you can find almost every device equipped with support for either a microcontroller or microprocessor. **A microcontroller unit** (**MCU**) is a single chip containing a simple processor with memory; it is used in devices such as industrial sensors, thermostats, smart switches, and lightbulbs. Microcontrollers make up more than 80% of all connected and connectable devices. A **microprocessor unit** (**MPU**) extends compute and processing power on edge devices. It has memory and I/O components connected externally; these are more powerful devices, such as your laptop, computer, cameras, and routers. Let's now learn about microcontroller device management and connectivity.

**Microcontroller device management**

When it comes to microcontroller-based devices, FreeRTOS is one of the most popular **real-time operating systems** (**RTOSes**). FreeRTOS includes a kernel and set of IoT libraries suitable for use across all industry sectors that make it easy to securely connect your small, low-power devices to more powerful edge devices and gateways. FreeRTOS help to easily program connected microcontroller-based devices and collect data to send for analysis and scale IoT applications across millions of devices. It helps to keep edge devices secure with security credentials and key management, and it also keeps your data secure with transport layer encryption.

MCU devices can connect to AWS IoT Core using MQTT Pub/Sub messaging or HTTPS-based file downloads from cloud or uploads to cloud storage for cloud connectivity to ingest and analyze data. MQTT is an OASIS standard messaging protocol for IoT. It is a lightweight publish/subscribe messaging transport ideal for connecting remote devices with a small code footprint and minimal bandwidth. FreeRTOS extends cellular LTE and the Wi-Fi abstraction layer, which helps to continue communication, collecting data, and taking actions without a cloud connection.

FreeRTOS provides an AWS IoT Device Defender library for device security, making it easy to report device-side metrics to detect anomalies when these metrics deviate from expected behavior. Device Defender also continuously audits the IoT configurations associated with your FreeRTOS devices to ensure that they comply with security best practices, such as auditing and monitoring devices, reporting TCP connections, and detecting anomalies.

FreeRTOS is fully supported and integrated with the AWS cloud. AWS supports a fully integrated firmware update service for FreeRTOS with integrated code signing using AWS IoT Device Management and supports **over-the-air** (**OTA**) software updates. OTA is a critical piece of the IoT value proposition and a vital part of the end-to-end security solution.

As MCU devices are more powerful and provide the ability to extend data analytics capabilities to the edge, AWS provides IoT Greengrass to connect these devices to the AWS cloud. Also, FreeRTOS provides convenient APIs that make it easy to connect to AWS Greengrass devices. Suppose the Greengrass Core device loses connection to the cloud. In that case, FreeRTOS devices in the Greengrass group can continue to communicate with each other over the local network, so your applications continue to run even with intermittent connectivity. Let's learn about MPU device management and building data collection capability near the data source.

**Microprocessor device management**

It's not always applicable to collect IoT data in a central place and perform analytics to gain some insights. You need to collect data at the edge and perform analytics locally in scenarios where internet connectivity is impossible, such as airplanes, cruise ships, or remote areas. It is also applicable when you cannot store data in other locations due to compliance or need ultra-fast latency, such as managing the robot fleet on the factory floor. In such cases, your devices need to reduce latency, reduce costs, and improve regulatory compliance at the edge location. Often, MPU devices are preferred for such use cases as they are much more powerful than MCU devices. They can also work as a gateway and manage multiple MCU devices at the edge.

AWS provides IoT Greengrass, which helps extend AWS services onto your devices to act locally on the data they generate to get immediate data insight and take action. With Greengrass, devices don't need to send your data to a distant cloud; data is stored locally, saving time when milliseconds matter. Also, it provides a choice of sending only the data you need to the cloud, which lowers costs. Greengrass-enabled devices continue to route local messages when data needs to stay local for data sovereignty laws, ensuring that data is secure and kept local.

AWS IoT Greengrass consists of two parts—an IoT edge runtime and cloud service. Using the Greengrass edge runtime on a device helps customers add device intelligence through local processing, data management, and ML inference and seamlessly connect it to AWS cloud services.

The Greengrass cloud service allows customers to deploy and manage IoT applications across their device fleets remotely. The following is a high-level overview of the pre-integrated analytics and ML services with IoT Greengrass in the AWS cloud:

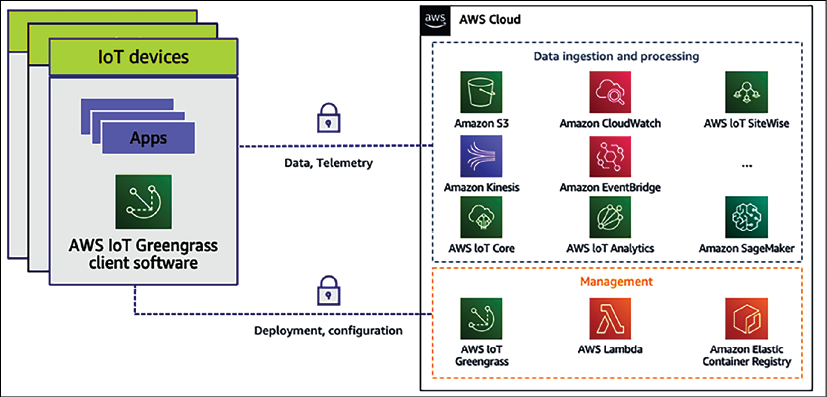


Figure 15.2: AWS IoT Greengrass pre-integrated AWS cloud services

Sometimes, IoT devices are not directly connected to the cloud. They communicate with hubs or gateways locally, which then connect to the cloud. For such use cases, the Greengrass edge runtime can be installed on hubs or gateways and helps device developers to build, deploy, and manage IoT edge applications on their gateways. Furthermore, the gateway enables the provision of intelligence for all devices connected locally to the gateway.

AWS Greengrass embeds local lambda compute, local messaging, local device shadows for data and state synchronization, and communication security in connected devices. OTAmakes installing updated versions of Greengrass Core easier to access new features, bug fixes, and security enhancements using AWS IoT Device Management.

Finally, you want to make sure your device is connected and working with cloud IoT services. AWS provides IoT Device Tester, a test automation application that lets you test FreeRTOS or AWS IoT Greengrass on your choice of devices. You can test whether FreeRTOS cloud connectivity, OTA, and security libraries function correctly on top of microcontroller board device drivers for MCU devices. For MPU devices, you can test if the combination of the device's CPU architecture, Linux kernel configuration, and drivers works with AWS Greengrass.

Now, let's learn more about the second component of architecture device connectivity and control to connect, manage, and secure your device.

**Connecting and controlling IoT devices**

Nowadays, you have millions of devices generating gigabytes and terabytes of data every second. So the next question is, *how I can connect my data securely and handle the data they generate at scale?* This is not only about data ingestion. You need to consider other factors, such as the following:

* **Identity service**: You need to identify services to manage the authorization of devices and provision unique identities at scale. IoT Core provides you with the capability to bring your root CA and client certificates or let the IoT platform generate certificates for you. IoT platforms need to support SigV4, X.509, and custom authentication, while providing fine-grained access control with IoT policies down to the MQTT topic level.
* **Device gateway:** This securely connects devices to the data center or cloud. The data gateway should automatically provision large fleets of devices with unique identities on their first connection using fleet provisioning and facilitate automatic device registration with just-in-time registration. The data gateway should securely connect devices to the cloud or data center and other devices at scale. For connection reliability, the data gateway needs to establish long-lived connections for bi-directional communication over MQTT, WebSocket, or HTTP, and secure communications over TLS 1.2 mutual authentication.
* **Message broker**: This processes and routes data messages to the data center or cloud. The message broker needs to route data from IoT devices with scalable, low-latency, and reliable message routing. It needs to provide publish/subscribe for decoupled devices and applications and facilitate two-way message streaming. The IoT message broker helps to understand and control the status of your device at any time and retain messages for offline devices and extends support for **Quality of Service** (**QoS**) messaging for MQTT:
  + QoS level 0 means, at most, one-time delivery of messages. It is also called **fire and forget**, where there is no guarantee of delivery as the recipient does not acknowledge receipt.
  + QoS level 1 guarantees that a message is delivered at least once to the receiver, but the message can also get duplicated due to the re-delivery of the same message.
  + QoS level 2 is the highest level of service in MQTT. It guarantees that each message is received only once; however, it is the slowest method and needs a four-part handshake between the sender and the receiver.
* **Rules engine**—This triggers actions on your devices as per business needs. A rules engine ingests large amounts of data, pre-processes it, and makes it available to other analytics, reporting, and visualization services. Rules engines need to have built-in functions for math, string manipulation, dates, and so on for data transformation and provide the capability to filter data before routing to other services for ML and analytics.
* **Device Shadow**—This enables applications to interact with devices even when they are offline and help to understand and control the status of your device at any time. Device Shadow should represent the device state by maintaining the last known state for offline devices, such as *the last known color of the light bulb is red*. Device Shadow should make real-time state changes based on action from the application and control devices via changes of state, for example, changing the light bulb's color to blue. Once a device's connectivity gets established, it automatically synchronizes.
* **Device registry**—This enables automatic device registration and helps to manage them. A device registry defines and catalogs devices for easy use by performing simple searches such as *which devices were made in 2016?*, or by defining thing types; for example, BMW and Audi are of the *Car* thing type to enable standardization of attributes and policies across devices. To simplify further, you can define groups such as sensors in a wind turbine to enable simpler management for running jobs and setting policies. It is better to use a managed IoT platform such as AWS IoT Core, which provides all of the above services. AWS IoT Core allows the secure connection of any number of devices to the cloud and other devices without requiring you to provision servers. You can route, process, and act upon data from connected devices and enable your applications to interact with devices even when they are offline. IoT Core provides AWS services to reason on top of the data through analytics, AI, and ML as part of cloud ecosystems.
* **AWS IoT Device Management**(**DM**) helps to register, organize, monitor, and remotely manage a growing fleet of connected devices. It registers many devices using bulk registration, organizes devices into groups, performs OTA firmware updates, and facilitates end-to-end management of all IoT devices with a fully managed web application.
* **AWS IoT Device Defender** (**DD**) is a fully managed IoT security service that facilitates the securing of a fleet of connected devices on an ongoing basis. It monitors IoT resources associated with appliances and the entire device fleet for abnormal behavior that might indicate a potential security issue. Device Defender sends alerts if something doesn't look right, such as traffic from devices to an unauthorized IP address or spikes in outbound traffic that might indicate that a device is participating in a DDoS attack. Finally, through its integration with IoT Device Management, IoT Device Defender lets you take corrective action to secure your devices.

AWS provides IoT Core, Device Managment , and Device Defender, collectively known as IoT connectivity and control services, which provide the ability to connect, manage, and secure devices. As you are collecting data from millions of IoT devices, getting insights from your data becomes very important. Let's learn more details about techniques for performing analytics on IoT data.

**Performing analytics on IoT data**

IoT data is challenging to analyze because it isn't the highly structured data usually processed by analytics tools designed for business intelligence and web analytics. Instead, IoT data comes from sensors attached to moving machinery with intermittent connections, controllers with poor Wi-Fi or wireless coverage, or lots of other places where signals get lost or weakened. The data from these devices can frequently have significant gaps and false readings. Also, IoT data is often only meaningful in the context of other data from external sources. For example, farmers need to enrich humidity sensor data with expected rainfall on the field to determine when to water their crops.

The incoming real-world IoT data needs to be enriched by combining it with other data such as time, location, and additional information, which creates challenges for businesses. To make their applications perform well, they frequently need to design custom logic to clean false readings, fill in gaps in the data, and enrich it with contextual information. They also need to store the process data appropriately before even crunching the data for their application. This requires custom code that takes time to build, test, and maintain and adds processing costs to IoT applications.

You can see that there are a lot of common data management and analytics tasks across IoT applications, including processing and enriching data, provisioning and partitioning databases, and writing complex queries. All data processing needs to be constantly developed as devices evolve, fleet sizes change, and new analytics requirements emerge. A company like C3 IoT provides sophisticated analytics, and cloud vendors such as AWS have created services such as **AWS IoT Analytics** to perform IoT data analysis at scale.

AWS IoT Analytics is a managed service that collects, pre-processes, enriches, stores, analyzes, and visualizes IoT device data at scale. It allows you to gather only the data you want to store and convert raw data to meaningful information.

Most of the IoT data is timestamp bound, so AWS IoT Analytics stores device data in a time series data store for analysis to get a deeper insight into the health and performance of assets and visualize your IoT datasets.

Overall, to design your IoT architecture, you should choose the right device software for your IoT project based on your hardware choice, software environment, and use case. If you employ highly constrained devices, typically, microcontrollers recommend using FreeRTOS and the IoT Device SDK. You can use AWS IoT Greengrass if you have microprocessor-powered IoT devices. Greengrass accelerates your device applications' development using pre-built processing and connectivity capabilities and remotely deploys and manages device software at the edge.

Once you are ready with your device, you can use AWS IoT Core, Device Management, and Device Defender to connect and control your device and perform data insights on top of collected data using AWS IoT data analytics. As the cloud is becoming a go-to place for data collection and analytics on a large scale, we will take examples from IoT services offered by one of the leading cloud providers, Amazon Web Services. Now, let's learn more about IoT in the cloud.

**IoT in the cloud**

Because IoT solutions can be complex and multidimensional, you need to remove the complexity of implementing IoT in the business and securely help customers connect any number of devices to the central server. When it comes to IoT, cloud providers have managed service offerings to achieve scalability to millions of devices. Some of the most popular cloud IoT platforms are Google Cloud IoT, AWS IoT Core, Azure IoT Hub, IBM Watson IoT, and Oracle IoT. Let's look into AWS IoT offerings to understand the workings of IoT systems, and the other cloud providers such as GCP and Azure, who have IoT offerings along the same lines.

The AWS cloud helps in processing and acting upon device data and reading and setting device states at any time. AWS provides the infrastructure to scale as needed, so organizations can gain insights into their IoT data, build IoT applications and services that better serve their customers, and help move their businesses toward full IoT exploitation.

The following diagram illustrates the components of AWS IoT:

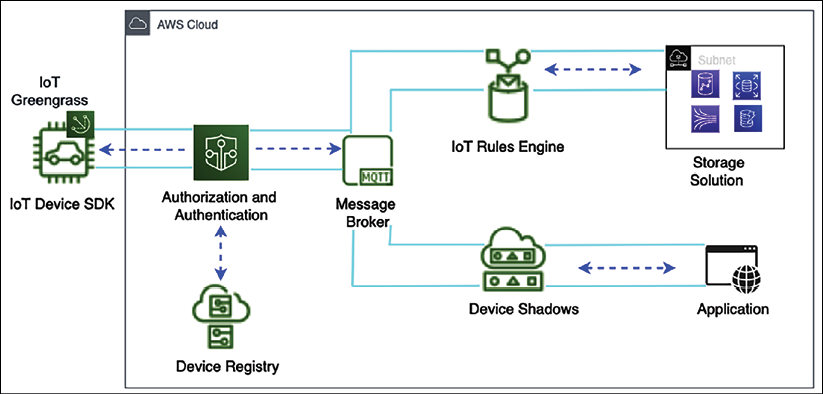


Figure 15.3: IoT architecture on an AWS platform

The following are the details of each IoT component and how they are connected in the preceding diagram:

* **IoT Greengrass**: AWS IoT Greengrass is installed on edge devices and helps to send IoT messages to the AWS cloud.
* **The IoT Device SDK**: The AWS IoT Device SDK helps to connect IoT devices to your application. The IoT Device SDK provides an API to connect and authenticate devices to the application. It helps to exchange messages between devices and AWS IoT cloud services using the MQTT or HTTP protocols. The IoT Device SDK supports C, Arduino, and JavaScript.
* **Authentication and authorization**: AWS IoT facilitates mutual authentication and encryption to exchange data with only authorized devices. AWS IoT uses authentication mechanisms such as SigV4 and X.509 certificates. You can attach authentication to all connected devices by attaching a certificate and handling authorization remotely.
* **IoT message broker**: The message broker supports the MQTT and HTTP protocols and establishes secure communication between IoT devices and cloud services such as the AWS IoT rules engine, Device Shadow, and other AWS services.
* **The IoT rules engine**: The IoT rules engine helps to manage data pipelines for IoT data processing and analytics. The rules engine looks at IoT data to perform streaming analytics and connect to other AWS storage services such as Amazon S3, DynamoDB, and Elasticsearch.
* **Device Shadow service**: The Device Shadow service helps you maintain a device's status when it is offline due to the loss of network connectivity in a remote area. As soon as the device comes online, it can resume its state from Device Shadow. Any application connected to the device can continue working by reading data from the shadow using a RESTful API.
* **Device registry**: The device registry helps identify IoT devices and helps to manage millions of devices at scale. The registry stores device metadata such as the version, manufacturer, and reading method (Fahrenheit versus Celsius, for example).

As of now, you are familiar with IoT service offerings in the cloud. As IoT is becoming very common in the manufacturing industry for handling machine data and optimizing production, it developed the concept of **Industrial IoT** (**IIoT**). Let's learn more about this now.

**Building an industrial IoT solution**

Industrial customers seek to gain insights into their industrial data and achieve outcomes such as lower energy costs, detecting and fixing equipment issues, spotting inefficiencies in manufacturing lines, improving product quality, and improving production output. These customers are looking for visibility into **operational technolog**y (**OT**) data from machines and **product life cycles** (**PLCs**) systems for performing **Root Cause Analysis** (**RCA**) when a production line or a machine goes down. Furthermore, IoT improves production throughput without compromising product quality by understanding micro-stoppages of machinery in real time.

Data collection and organization across multiple sources, sites, or factories are challenging to build and maintain. Organizations need a consistent representation of all their assets that can be easily shared with users and used to build applications, at a plant, across plants, and at a company level. Data collected and organized using on-premises servers is isolated to one plant. Most data collected on-premises is never analyzed and thrown away due to a lack of open and accessible data.

The best practice is to extract data from databases commonly found in industrial facilities, transfer it to the centralized storage in the data center or cloud, and structure it to be easily searchable by users and applications. On top of that data, you can derive common industrial performance metrics such as **Overall Equipment Efficiency** (**OEE**), monitor operations across multiple industrial facilities, and build applications to analyze industrial equipment data, prevent costly equipment issues, and reduce gaps in production. To design industrial IoT architecture, you need to perform the following steps:

* Ingest data from industrial equipment, data servers, and historian databases
* Collect, organize, and analyze industrial data at scale
* Read data from onsite equipment using industrial protocols and standards such as OPC-UA, Modbus, and Ethernet/IP
* Create visual representations of physical assets, process equipment data streams, and compute industrial performance metrics
* Access local dashboards to view real-time and historical equipment data, even when temporarily disconnected from the internet
* Consume asset data to create local or cloud applications that optimize factory output quality, maximize asset utilization, and identify equipment maintenance issues

To address the growing need for IIOT, a leading cloud provider such as AWS provides a managed service, **AWS IoT SiteWise**, which collects data from the plant floor with a local gateway, structures and labels that data, and generates real-time KPIs and metrics to make better data-driven decisions.

Data is collected from equipment across all sites during ingestion and sent to AWS IoT SiteWise from AWS IoT Core. Then it creates model assets that are virtual representations of physical assets. SiteWise helps to digitize, contextualize, and model entire production environments without customers having to maintain their infrastructure. Customers can represent complex equipment hierarchies using rich information modeling.

Event management is critical for detecting changes across complex industrial systems. There is a need to continuously monitor data from your equipment to identify their state, detect changes, and trigger the appropriate responses when changes occur. AWS provides AWS IoT Events, which builds simple logic to evaluate incoming telemetry data to detect stateful changes in equipment or a process. It detects events from data across thousands of sensors and triggers responses to optimize operations. Instead of relying on the manual inspection of parts, machines, or products, it can notify maintenance teams of issues more quickly or instruct a device to shut down. For example, you could alert a technical support rep when equipment runs abnormally or apply the fix.

Because you configure AWS IoT Events using logical expressions rather than complex code, adjusting to changes such as the addition of new equipment is easy. You can scale across thousands of devices in your fleet. AWS IoT Events integrates with other AWS services that handle and analyze IoT data, such as AWS IoT Core and AWS IoT Analytics. Customers can identify and ingest the data they need to evaluate right from the AWS IoT Events console. IoT Events can trigger actions in AWS Lambda, SQS, SNS, Kinesis Firehose, IoT Core, and so on. Let's look at an IIoT reference architecture to bring all pieces together.

**Connected Factory IoT architecture**

The **Connected Factory** (**CF**) solution is designed to bring together capabilities to transform manufacturing operations. CF makes it easy for customers to unblock data from their legacy systems, visualize the data in near real time, perform deeper analytics to optimize operations, and improve productivity and asset availability. The key focus for the CF offering is to commoditize industrial data collection and develop repeatability. Let's look at the following diagram, which demonstrates IoT architecture for implementing connected factory solutions on the AWS cloud platform.

Diagram

Description automatically generated

Figure 15.4: Connected Factory architecture in the AWS cloud

As shown in the preceding diagram, AWS IoT Greengrass is deployed at the edge of the factory floor to collect equipment data and other data ingested from servers at the facility. Data lands in the AWS cloud through IoT Core, and IoT SiteWide helps to build a model of physical devices. Data from various facilities is stored in Amazon S3 to build a manufacturing data lake that can be further loaded in Redshift for data warehousing and processed through the ETL pipeline using AWS Glue and ad hoc queries performed using Amazon Athena. Finally, you can use QuickSight to visualize data for business users.

Streaming data is transformed and processed through Amazon Kinesis and provides input back to product equipment or shipment information to the vehicle. You can also see ML components perform production forecasting and post that data in ERP and PLM systems to optimize production efficiency. Amazon SageMaker performs ML at the edge to understand and alert on equipment health to reduce downtime.

When it comes to training staff on equipment and creating a simulation, adding a visualization layer makes sense. This is now possible with the availability of **AR/VR**(**Augmented Reality/Virtual Reality**). That's where digital twins come into the picture. Let's learn more details about digital twins.

**Implementing a digital twin**

A digital twin is a digital replica of a physical machine. In a digital twin, you build a virtual representation of the machine using AR/VR to visualize a real-time data overlay. It helps to see real-time operational and health data, combined with machine learning; you can draw insights from real-world behavior such as performing a proactive maintenance model. A digital twin can be handy for simulating what-if scenarios to determine the optimum KPI for the machine, and building immersive education and training to handle the equipment.

A digital twin continuously collects real-time data using IoT and can control the machine's operation from a digital replica. It provides an immersive experience of the living model of a machine and helps with early warning, prediction, and optimization. A digital twin performs the following tasks, as shown in the following diagram:

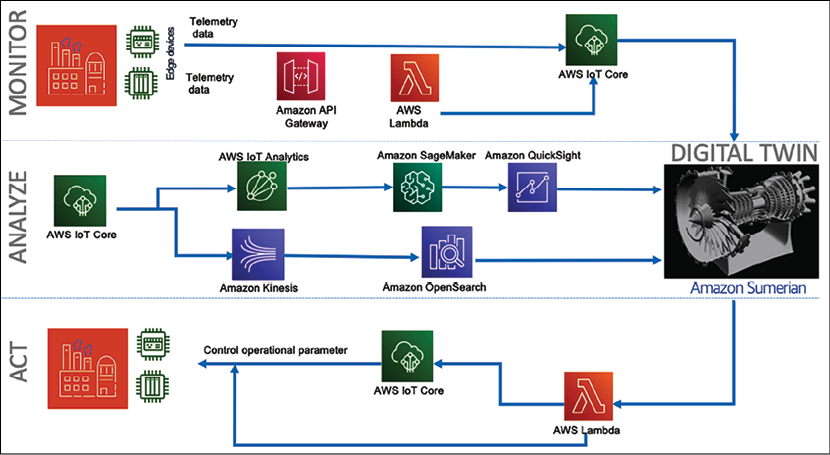


Figure 15.5: Modeling the mind of the machine with a digital twin

As shown in the preceding diagram, a digital twin manages a machine by:

* **Monitoring**: The digital twin collects and analyzes the data by replicating a digital copy in the virtual world. Machine telemetry data can come from sensors ingested by the cloud, such as AWS, using AWS IoT Core. The factory floor data can be ingested by building an API wrapper around the on-premises applications.
* **Analyzing**: To build a digital replica, you can use popular AR/VR technology such as Microsoft HoloLens, Amazon Sumerian, or Oculus. You can create a data overlay on top of the digital replica to show how data flows from various sensors. Further analytics can be performed using AWS IoT Analytics. To build data visualization and search capabilities, you can use Amazon OpenSearch and QuickSight. The digital twin can be controlled over voice using AI-powered services such as Amazon Alexa. ML capability can be implemented using Amazon SageMaker to train, tune, and deploy the ML model.
* **Acting**: As you get data insights and predictions, you can take the required action by sending messages back to the operations team. You can use AWS IoT Events and AWS Lambda to notify operational applications by creating automated maintenance tickets for staff on the ground. AWS IoT Core can take your message and apply direct operations to the machine. If a cooling fan is running abnormally or getting hotter than expected, you can stop the machine directly from the digital twin.

Let's take a reference architecture of a digital twin for an aircraft jet engine, as shown in the following diagram. Here, you collect engine temperature and speed data in real time using IoT sensors and show data overlays in a digital engine replica to gain insights and take action.

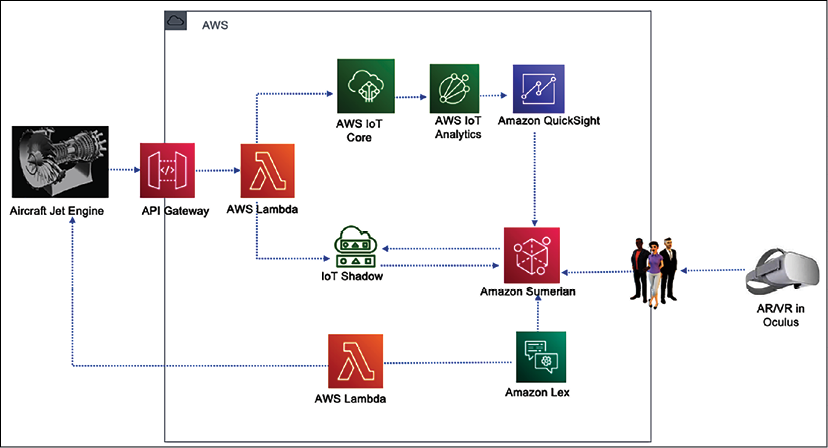


Figure 15.6: Digital twin architecture for an aircraft jet engine

As shown in the preceding diagram, temperature and engine speed data were sent from a jet engine to the AWS cloud through IoT Core. AWS IoT Analytics performed data processing to collect insights into the sensor data collected and visualized information in the Amazon QuickSight dashboard. The jet engine's current state is maintained using Device Shadow, so if sensors go offline, you can still perform simulations. Here, a jet engine digital replica was created using the Amazon Sumerian virtual reality platform and deployed in Oculus. Using the Amazon Lex AI service, you can start/stop the actual engine with your voice or a message.

IoT is a very vast topic that warrants an entire book. In this chapter, you learned about the various components of IoT architecture that have industrial uses.

**Summary**

There are millions of small devices connected to the internet, referred to collectively as the IoT. In this chapter, you learned about IoT and the components of IoT architecture. IoT is all about processing unstructured telemetry data from sensors and machines in high volumes and at speed. To handle such data, you need to have a scalable system, and you learned about IoT in the cloud with examples from AWS IoT services.

Devices are one of the central points when it comes to building IoT solutions. You learned about two major types of device software, including an MCU and an MPU, and methods to ingest these devices' data. You learned about the various techniques for controlling IoT devices, including device authentication, a device registry, and device management at scale, as well as the AWS IoT Device Management service. Security is the most pressing job, and it applies to IoT devices as well; you learned about various mechanisms for managing and securing IoT devices with AWS IoT Device Defender.

Once you collect and store data, you need to transform data to get insights into that data and visualize your business requirements. You learned about the various components available in the cloud to collect, process, and analyze IoT data to produce meaningful insights. Industrial IoT is becoming very popular for optimizing production and reducing operational downtime. You learned about IIoT and how AWS IoT SiteWide helps to address IIoT operations at scale. You also learned about Connect Factory IoT architecture and its functioning in detail.

Combining AR/VR technology with IoT provides an immersive experience. You learned about the digital twin concept, where a virtual replica of a physical machine gets created with real-time data overlay. You learned about a jet engine digital twin architecture with different components to monitor, analyze, and act using the digital twin model.

Until now, we have relied on supercomputers with significant numbers of GPUs and CPUs to solve the majority of problems. But with the increased use of technology, supercomputers are becoming slow in cases where millions and billions of combinations are required to solve a problem in complex use cases such as molecular analysis and building a financial risk model. For such kinds of use cases, quantum computing may be the ideal technology. We are still in the early stages of the quantum evolution, but organizations have started experimenting with it. In the next chapter you will learn about quantum computing, its use cases, and the options available.