codit

White paper

IOT Field Gateways

Table of Contents

1. About Codit....2
2. In This White Paper....3
3. The IoT Value Chain....4
   - 3.1 Data to value....5
   - 3.2 Time spent versus value created....5
4. The Connectivity Challenge....6
   - 4.1 Direct device connectivity....6
     - 4.1.1 Scenarios....6
   - 4.2 Cloud-to-cloud connectivity....7
     - 4.2.1 Scenarios....7
   - 4.3 Field gateway connectivity....8
     - 4.3.1 Scenarios....8
   - 4.4 The retrofitting problem....8
5. What Is a Field Gateway....9
   - 5.1 Benefits of using a field gateway....9
   - 5.2 Comparison of the three different connectivity approaches....10
6. Field Gateway Responsabilities....11
   - 6.1 Activation & security....11
   - 6.2 Health monitoring & management....11
   - 6.3 Configuration management....12
   - 6.4 Remote update....12
   - 6.5 Command processing....13
   - 6.6 Edge filtering and aggregation....13
   - 6.7 Extensibility....14
7. Nebulus™ IoT Gateway....15
   - 7.1 The design goals....15
   - 7.2 What is it?....16
8. Conclusion....17
9. References....18

1. About Codit

Large, international companies often struggle to easily exchange data with their subsidiaries, customers, suppliers and other business partners. Many also face challenges with the upcoming trends such as Cloud, SaaS apps, Mobile, Internet of Things, Big Data... Companies not only have to be aware of them, yet have to adopt these technology changes strategically to gain competitive advantage. That is exactly what our expert teams do: we integrate business applications with the newest Microsoft technologies.

Author

Sam Vanhoutte | CTO and Product Manager

Reviewers

I would like to thank Luis Delgado, Technical Director at Codit Switzerland, Sjoerd van Lochem, General Manager at Codit Netherlands and Filip Van Raemdonck and Tom Kerkhove, both Azure Consultant at Codit Belgium for reviewing this white paper.

2. In This White Paper

In this new white paper in our IoT series, we will focus on device connectivity. The white paper describes the different ways to connect machines and devices to a cloud backend and will then focus more on the concepts of an IoT field gateway.

While helping dozens of customers in building their IoT solutions, we have learned that to create value in the cloud, device connectivity is the most important hurdle to take. It is of the utmost importance to have a reliable and solid connection between the deployed devices and the cloud from the beginning, as it can prove to be very difficult to change or fix things after the facts.

By reading this white paper, **you will learn which IoT field gateway scenarios will make your solutions more secure, more reliable, and ready for production faster**. This will allow you to focus on the real value creation by turning data into insights.

3. The IoT Value Chain

A typical IoT solution chain is comprised of the following layers:

- **Devices** are the basis for the IoT solution. They connect to the backend, using one of the three patterns mentioned in chapter "4. The connectivity challenge".
- **Gateways** can be used to help devices to connect with the cloud. Field gateways are the main topic of this white paper.
- The **ingestion layer** is the entry into the IoT backend and is typically the part that must be able to scale out to handle a lot of parallel incoming data streams from the (thousands of) devices.
- The **automation layer** is where business rules, alerting and anomaly detection typically take place. Also, automated actions that will trigger integration in the Report and Act layer are common.
- The **data layer** is where analytics and machine learning typically take place and where all the stored data gets turned into insights and information.
- **Report and Act** is all about turning insights in action, where business events get integrated with the backend systems, or where insights get exposed in reports, apps or open data.
-

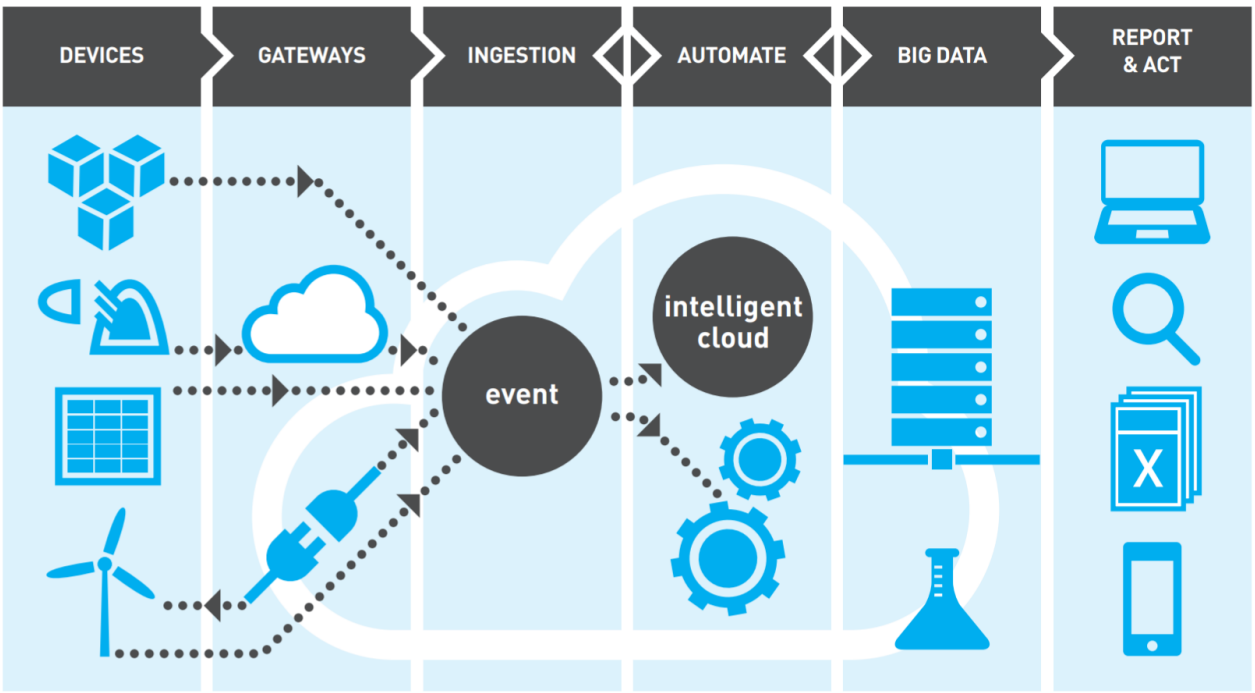


Figure: The IoT value chain; capture, analyze, act.

3.1 Data to value

Many IoT scenarios are driven by data, a lot of data. "Data is the enterprise currency." "Data is the new oil." "From data to information." Marketing departments in the industry are going wild to emphasize the importance of data in today's era.

However, most data intelligence scenarios require a lot of data to be consistent. Predictive maintenance, trend prediction, customer behavior analysis, heatmap scenarios... The more (quality) data companies have available, the better their algorithms and predictions tend to be (no matter if they are driven by traditional data analytics or modern Machine Learning). Therefore, **data cleansing** preparation is also a very common exercise.

Next to that, it is also important to be able to match previous important business events (such as production outages or machine failures) or external events (such as traffic jams, weather data or sports events) with the historical time series of **machine data**. This allows the machine learning software to verify its models. That explains why IoT solutions often require system integration technology, as outlined in our first IoT white paper: Integration of Things (available via codit.eu)

3.2 Time spent versus value created

As mentioned above, the real added value is created in the cloud, using the collected data and telemetry. Real results are achieved by applying data analytics and automating actions based on the analysis findings.

Every IoT project needs results fast. But all too often, a lot of time is lost on the device side. There are huge challenges in building connected devices that are secure, reliable and robust. Yet, most new IoT projects tend to spend a lot of time on just connecting, deploying, maintaining and securing IoT devices. This is called the connectivity challenge. How to jump start the IoT connectivity phase and start generating value with data analytics as quickly as possible? Another issue is that most software engineers don't have enough knowledge of electronics or device programming, while most electronics engineers typically lack the skills needed to build secure and robust cloud connections. Building the right team combining these skills is an important prerequisite to succeed.

Our experience is that customers are spending too much time on **device connectivity** and testing, while they prefer to create value through **data analytics** and **automation**.

Companies don't want to spend time here	They want to create value here
Message handling	Data analytics
Protocol implementation	Device intelligence (edge compute)
Credential management	System integration
Health reporting & diagnostics	Reporting & insights
Secure lifecycle management	Device management & alerting

4. The Connectivity Challenge

Companies want to have their device data in the cloud as soon as possible. They hope to connect as many devices as possible to their cloud backend, quickly. We find that there are three common approaches to enable this: direct device connectivity, cloud-to-cloud connectivity and field gateway connectivity.

4.1 Direct device connectivity

When devices have built-in network connectivity (3G, WiFi...) a good option can be to connect these devices directly to the cloud backend, using one of the common protocols (MQTT, AMQP, HTTPs...). This means that the devices should be able to cope with all computing requirements (hashing/encryption, network buffering, I/O) and security challenges that are common in device connectivity. A good approach is to always rely on the concept of service assisted communication..

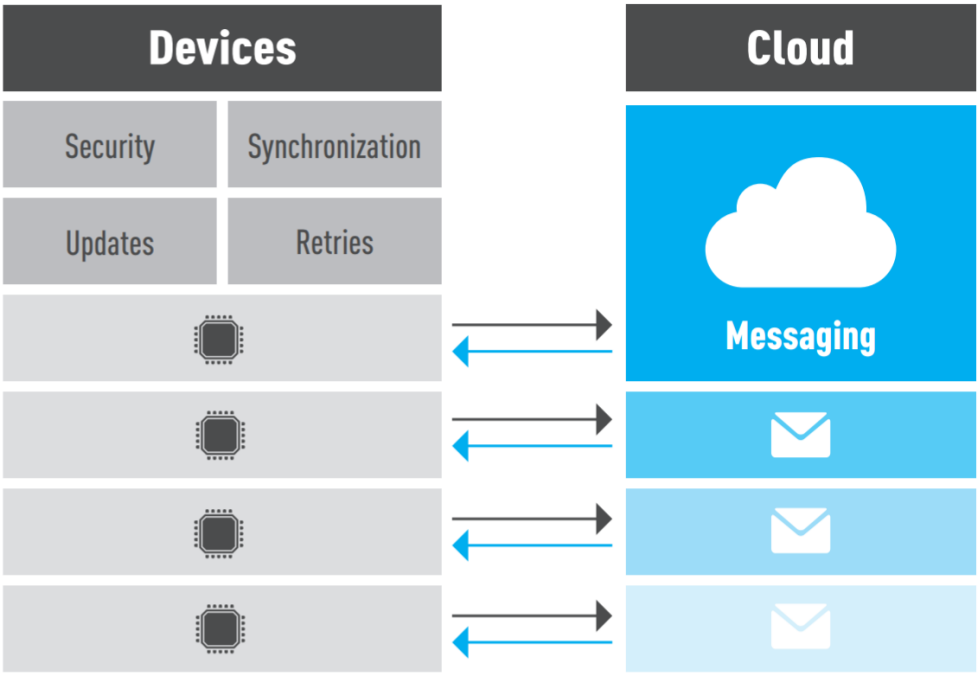


Figure: Direct device connectivity. Devices are responsible for security, synchronization, updates and retries.

4.1.1 Scenarios

For consumer devices such as smart thermostats, this is the de facto standard. As the setup and configuration of these devices should be as easy and solid as possible, it makes sense to have one single-purpose device that has everything built-in to connect to a cloud backend.

It is also common to see that devices of this type don't have the capabilities to support 'from the cloud' updates of the firmware (except for some configuration settings). It is often left up to the user (consumer) to upgrade the smart device through an app acting as a configuration proxy.

4.2 Cloud-to-cloud connectivity

Device makers and sensor manufacturers are also looking to innovate their business. They start to offer addedvalue services, on top of their silicon products. These platforms usually offer integration endpoints where telemetry of the devices for a tenant get forwarded to the endpoint of choice. **Narrow band protocol devices** (such as SigFox and LoRa) almost always come with a direct connection to a backend, as the entire stack from network protocol, security, billing and metering is taken up by these vendors.

When companies decide to take advantage of these off-the-shelf products, they are mostly limited in customizing their device logic, but at least they can get started quickly. The connectivity is mostly set up in a cloud-to-cloud manner, using standard protocols or connections like MQTT, AMQP or HTTPS.

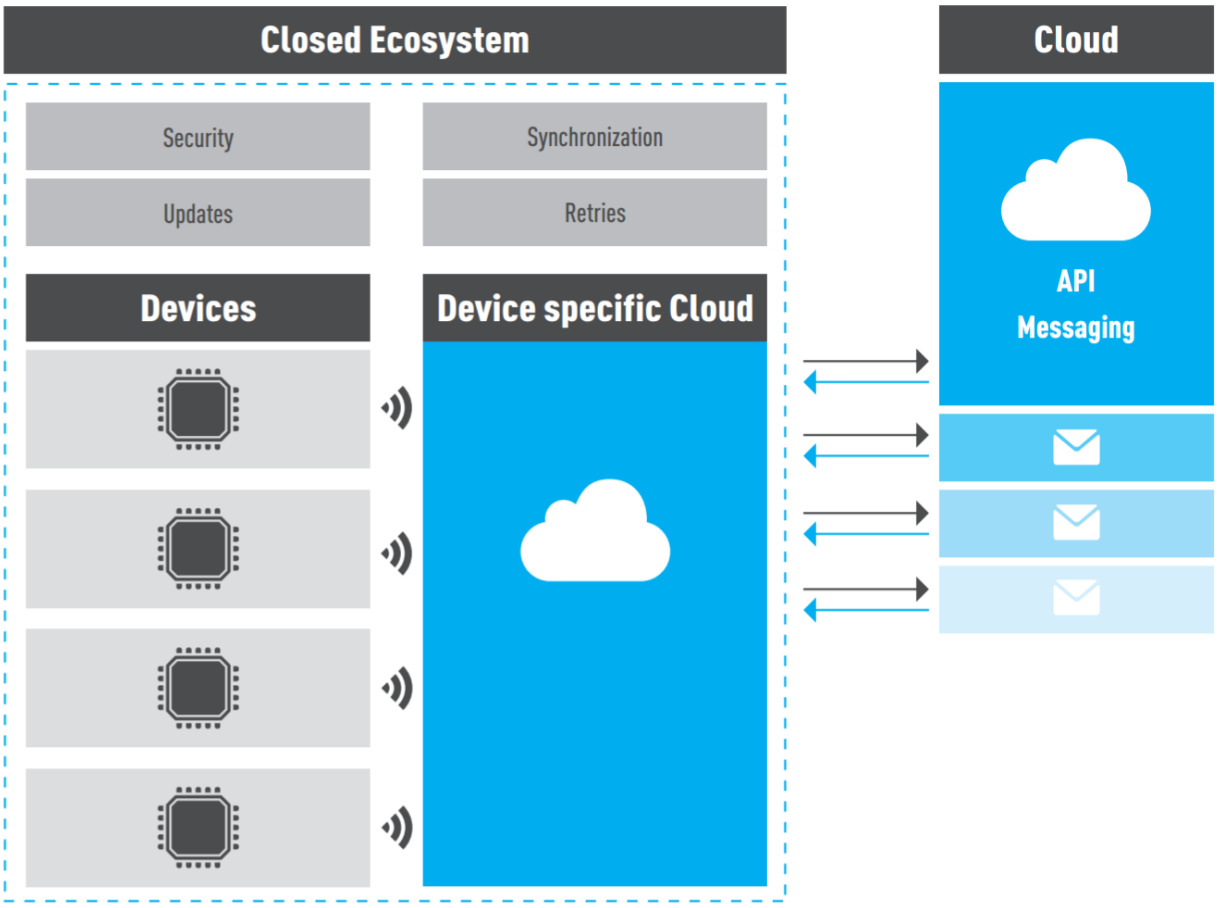


Figure: Device connectivity from cloud-to-cloud in which the ecosystem is responsible for security, synchronization, updates and retries.

4.2.1 Scenarios

This type of device connectivity is often found in low throughput scenarios, known as narrow band IoT, or field scenarios. Think of smart cities or agricultural deployment where it is often impossible to fall back on a reliable network connection. Scenarios with smart energy metering (water/electricity meters) are also often built using this type of connectivity.

4.3 Field gateway connectivity

The third connectivity concept brings us to the core of this white paper: IoT field gateways. Companies can deploy a variety of sensors/devices using different protocols. They want to connect these devices, which are spread across different locations, to a **central cloud backend**. Quite often these devices have existing interfaces (built for local connectivity) that can be consumed on the local network. Instead of upgrading these devices, or building all the required capabilities into them, companies tend to choose for the "Field Gateway approach". This field gateway then acts as an intelligent and secure bridge between the local device protocol and the cloud backend. In this white paper, we will discuss all the flavors and responsibilities such a gateway can have.

Figure: Gateway connectivity. The gateway is responsible for security, synchronization, updates and retries.

4.3.1 Scenarios

Field gateways are mostly used in scenarios where there is a variety of devices and these devices are grouped in deployments that are geographically spread. Industrial IoT (IIoT) solutions are a perfect fit for this. By leveraging a field gateway, its user has more options for devices and the devices typically tend to be cheaper, as they don't need the capabilities described in the next chapter

4.4 The retrofitting problem

As mentioned in an IDC study, an estimated 85% of existing installed devices and sensors are not necessarily able to connect to the internet. However, a lot of these devices hold valuable data and have existing (local) data interfaces, using one of the many protocols that have been used over the past years.

Especially in industrial scenarios, the number of machines and devices that hold such valuable data is huge. Therefore, a field gateway can typically be used to monetize on these devices and previous investments. A gateway makes it possible to connect to the devices, using their existing interface, and securely the bridge to the IoT backend where data analytics and reporting can be performed.

5. What Is a Field Gateway

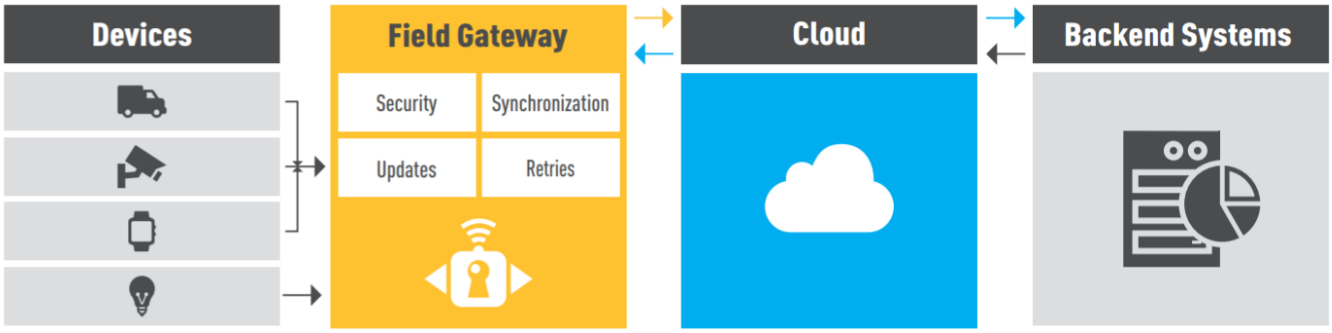


Figure: Basic figure of the role of a field gateway

A field gateway acts as a bridge between your local devices and the IoT backend. It connects and listens to different devices and/or sensors, using a variety of (mostly local) protocols while sending all device events and telemetry messages to its configured cloud backend, using a standard cloud protocol (such as AMQP, MQTT or HTTPs). This is commonly called **protocol translation**.

In downstream messages, the field gateway will also listen to (device and gateway) commands from the cloud and forward them to the right connected device (or handle them in case of gateway commands).

A field gateway can come as a hardware device that has built in software to have gateway functionalities. It can also come as a software package that runs device independent

5.1 Benefits of using a field gateway

A field gateway accelerates companies in connecting their devices to their cloud backend system. Using different available connectors (or by building their own), connecting systems with the cloud becomes a matter of configuration instead of custom development.

A gateway provides for all required functionality to cope with connectivity issues, cloud based configuration management and security challenges.

Using a gateway also allows device makers (especially in the industry) to fully isolate their machine logic (and versioning/testing) from the actual internet connectivity. This has advantages on stability, agility, cost and security.

Other benefits:

Faster time to value (accelerating device connectivity)	✔
Can bridge legacy systems to an industrial standard interface (retrofitting)	✔
Provide core capabilities out of the box, while leaving room for customization & extensibility	✔
Cloud connected by design	✔
Centrally manageable	✔

5.2 Comparison of the three different connectivity approaches

	Direct	**Cloud-Cloud**	**Field gateway**
Time to value	Slow	Fast	Fast
Device responsibilities	Complex	Complex	Minimal
Cloud-based updates	Complex	Mostly not provided	Native, built in
Installation complexity	Simple	Simple	More complex
Device compute needs	Higher	Very low	Low
Edge capabilities	Limited	Very limited	High

6. Field Gateway Responsabilities

This chapter describes common gateway features and responsibilities that robust and well-featured gateways should have. **One of the most important things that companies should consider when selecting a field gateway is to see if the gateway comes as an open solution, allowing them to extend and customize where needed**. And it should also be possible to use the backend of choice for the telemetry ingestion.

6.1 Activation & security

Gateways that come with a cloud backend typically need to be provisioned on that backend. During this step, basic information such as serial number, activation code, metadata, and device type is provided. At that point, the gateway exists virtually and is ready to be activated from a real, physical gateway device. The device cannot start communicating in this state.

After installation, the gateway should be activated to enable it to start sending telemetry and receiving commands. There are a several patterns available for the activation logic, depending on the gateway that is being used:

Pre-activated gateways	Bootstrapped gateways	Local only gateways
These gateways come with their activation/security credentials pre-configured and will just start connecting to the cloud on startup. Typically, this requires the credentials to be installed on a security/encryption module, such as a Trusted Platform Module (TPM).	These gateways come with a minimal 'bootstrap' startup firmware and a 'call-home' address. When connecting to that bootstrap endpoint, a security validation can occur and configuration (that could be pre-configured) can be retrieved. Optionally, extra required libraries can be downloaded onto the gateway system.	Local gateways are limited in functionality but are still commonly used. Typically, they need to be installed, updated and configured through a local web interface hosted on the gateway system (much like your local network router). When the gateway sends data to a cloud backend, this is typically configured locally. This white paper will not focus much on local only gateways, as we believe that cloud-managed gateways offer much more value.

Upon successful activation, a gateway connects to the cloud backend using its dedicated identity (often retrieved in the activation process). **Gateway connectivity** is often setup bidirectionally. This allows commands and actions to be sent from the cloud to the gateway (and devices) as well as to stream telemetry information from devices to the cloud

6.2 Health monitoring & management

When field gateways are being used in a **distributed environment** (which is obviously very common in IoT scenarios), it is extremely valuable to have a centralized overview of all the deployed gateways. Ideally this should be linked to the topology / hierarchy in which they are deployed.

Cloud-connected field gateways mostly come with a cloud backend that allows users to provision, monitor, manage and update the gateways that are deployed in their infrastructure. The actual device metadata (such as customer, location, hierarchy, et cetera) is typically only configured after activation.

Automated event integration, alerting and rule configuration can add extra features here. This would allow notification, either through e-mail or text message or by subscribing to gateway events. Notifications can also be integrated in existing management systems.

6.3 Configuration management

The configuration for a gateway typically contains a set of **configured components or modules** that define the path that messages (telemetry or commands) should travel between the devices and the actual backend. Other common functionalities that are configured for a field gateway are message formatting, parsing, enrichment, batching and filtering.

Cloud-managed gateways typically only cache their configuration locally, but have the real master data living in their 'digital twin' in the cloud backend. Through this setup, it is possible to update the configuration of one or more gateways simultaneously in the cloud (typically through an API or portal). The gateways will get notified of their new configuration and will apply the configuration locally. In scenarios like this, it is very important that the gateways report back to their backend (digital twin), so that administrators (or automated processes) can get notified of gateways that are out of sync
Microsoft offers a variety of services in Azure that can be used to gain intelligence from your data.

6.4 Remote update

Gateways can be deployed in widely distributed and isolated environments. They typically also require a lifespan of multiple years.

During this time, it is highly likely that updates should be rolled out to gateways that are deployed in the field. Reasons include security patches, bug fixes, updates of functionality or breaking changes to the back end (which should be prevented).

The last thing a company wants to do is to have field engineers travel to all these remote locations to deploy the new version of the gateway software. Therefore, a real IoT field gateway should support **cloud-driven updates**. This allows organizations to fix security issues, deploy new features and to push new modules from a central location, in an orchestrated way.

6.5 Command processing

Reading telemetry from devices is the first step in almost all IoT projects. But sending commands to devices is something that follows quite often. Security is crucial in this pattern, as it is very common that device commands trigger a physical change in the field (turning on/off lights, setting temperature, increasing speed). The impact of commands is mostly much bigger than the impact of telemetry. This makes it extremely important that commands are handled in a very secure and robust manner. Also, commands should have a certain lifespan. If a command is not received by the device in a certain time, it should expire, and the backend should monitor (and act) upon this.

Very often, device commands are being relayed over a gateway to the actual destination device. The gateway can take responsibility to handle the security, validity and feedback of the commands.

6.6 Edge filtering and aggregation

While there are cases where companies are sending all data events into the cloud, we see most Codit customers opting for intelligent filtering 'on the edge'.

- **Filtering** served to only send events that are considered relevant. It is very common to only send a data field when its value has been changed. Or to only send a value if it is outside of a threshold.
- **Aggregation** takes sums (i.e. energy consumption) or averages (i.e., temperature values) over a certain period. In this way, it is still possible to get accurate enough data.
- **Batching** is another common approach where several data items are being combined into one cloud message.
- **Compression** (such as Avro) is frequently used to save bandwidth.

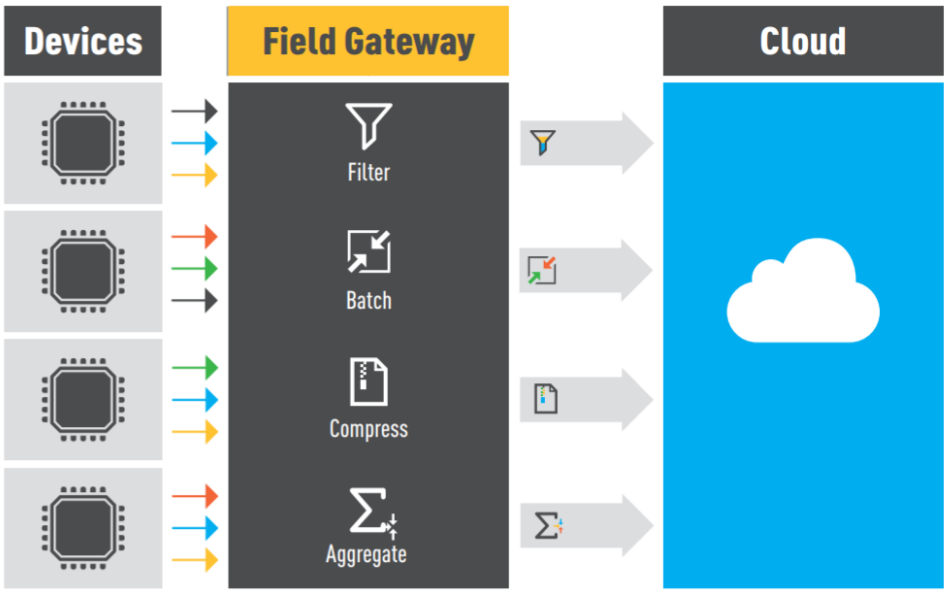


Figure: Applying edge filtering and aggregation techniques on a field gateway

By applying edge filtering and aggregation techniques on a gateway, companies can save a lot of precious bandwidth (and money). Extra features are also possible, such as local actions or execution of Machine Learning models against the collected data.

6.7 Extensibility

Due to the nature of the solutions, a gateway should be extensible from the ground up. Companies that want an **off-the-shelf gateway** will probably require customization along the way of the project. Existing protocols, custom message formats and custom activation logic all require extensibility. Companies either want to be able to do this themselves or have this done by an implementation partner.

We believe that, when looking for an IoT field gateway, extensibility should be one of the top requirements.

7. Nebulus™ IoT Gateway

In all the IoT projects that Codit has implemented, we noticed that our customers very frequently needed a field gateway to tackle the connectivity and retrofitting challenges. After considering several off-the-shelf hardware gateways, we found they all lacked the flexibility and openness we wanted to offer our customers. Most of those gateways were also locally managed and did not come with a decent cloud backend.

In close collaboration with the Microsoft Azure IoT engineering team (Codit is an Azure IoT elite partner), we decided to build a software gateway on top of the Microsoft Azure IoT Gateway SDK (now Azure IoT Edge SDK). The Nebulus™ IoT Gateway was born. It is a robust and cloud-managed IoT field gateway, built and sold by Codit.

The system Nebulus™ IoT Gateway is built on top of the Azure IoT Edge SDK and can run as a standalone software gateway or as an Azure IoT Edge module.

7.1 The design goals

We decided to build the gateway with the following design goals in mind:

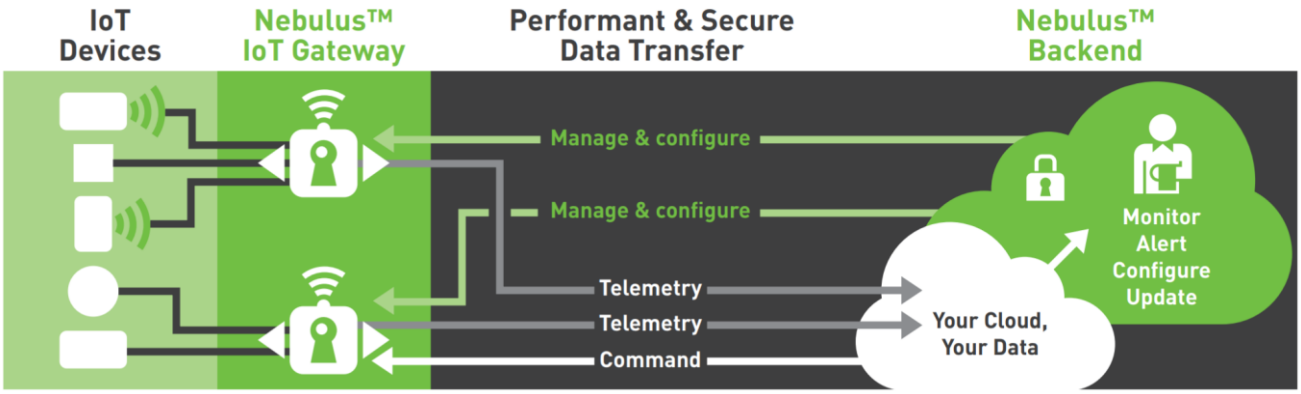
- The gateway is to be used in mission critical systems and can be leveraged by Codit Managed Services, providing up to 24/7 support on the platform.
- The possibility of extending the gateway yourself (using custom modules and plugins) and managing the gateway lifecycle from internal applications and processes (using our secured Web API).
- The gateway is easy to maintain because everything is managed and updated from the cloud.
- The gateway can be installed through zero-touch provisioning. This requires minimal to zero field intervention from a service technician, thereby drastically reducing operational costs.
- Because we are building on top of the Azure IoT Edge platform (as Azure IoT elite partner), the system is
- future proof and will benefit from the evolutions and enhancements that Microsoft keeps bringing.
- It is possible to fully integrate the solution with existing backend and business processes, leveraging all the integration points (such as our event pipelines and API).

- Capabilities are built natively on Microsoft Azure as we are convinced that this platform offers the best functionality for customers.
- Offering a cloud business model where customers pay per number of installed gateways. Most customers start with a small number of gateways, allowing for decreased connectivity costs as compared to a custombuilt solution.
- The gateway should be allowed to run on embedded or larger devices, on different operating systems and even in the cloud.

7.1 What is it?

Nebulus™ IoT Gateway provides all the features that have been listed in this white paper, including the following:

- Extensible and open software gateway
- Cross platform (Linux & Windows)
- Out-of-the-box modules and connectors
- OPC-UA compliant
- Comes with a managed, reliable cloud backend
- Dynamic configuration
- Cloud-based device models
- Remote updates
- Device and sensor management
- Secure gateway activation
- Extensible logging and diagnostics



The graphic highlights the setup of the Nebulus™ IoT Gateway.

- All device data is streamed to the subscription of the customer, guaranteeing **privacy and data ownership**. This is with the white and grey arrows linking to the white cloud.
- The Nebulus™ cloud backend handles all provisioning, configuration, updating, monitoring and alerting workloads and is used to **manage and configure** the actual Nebulus™ IoT Gateway agents in the field.
- The actual Nebulus™ IoT Gateway **connects to the local devices**, using the settings and configuration, defined in the Nebulus IoT configuration portal.

8. Conclusion

An IoT field gateway accelerates the implementation of many IoT projects and reduces ROI even more drastically in case existing disconnected devices can be internet-enabled through a retrofitting approach.

Adding **edge-capabilities** (such as filtering, aggregation or local rules) on a gateway will further reduce the cost of an IoT project and enhance the local decisions and actions.

Therefore, IoT field gateways are frequently used in **Industrial IoT solutions**. In such projects, it is very common to have existing devices using a plethora of different protocols that have not been designed for internet connectivity. **Leveraging a field gateway will speed up the implementation of a project and will bring intelligence from the factory into the cloud**.

The Nebulus™ IoT Gateway is a software field gateway that has been built following the principles outlined in this white paper. It is guaranteed to accelerate your IoT projects, allowing you to focus quickly on building added value that sets your solution apart from the competition.

9. References

Reference material for further reading:

- "**Integration of Things**" by Tom Kerkhove: https://www.codit.eu/about-us/download-center/13185/integration-of-things/
- "**Service Assisted Communication**" for Connected Devices by Clemens Vasters: https://blogs.msdn.microsoft.com/clemensv/2014/02/09/service-assisted-communication-for-connecteddevices/
- "**The Race For Industrial (IOT) Standardization: Collaboration And Open Source Initiatives**" by Shivakumar Movva: https://www.iotone.com/guide/the-race-for-industrial-iot-standardization-collaboration-and-open-sourceinitiatives/g493
- **Machine learning**: https://azure.microsoft.com/en-us/services/machine-learning/
- **Introducing IoT Edge**: https://azure.microsoft.com/en-us/campaigns/iot-edge/
- **Nebulus™ IoT Gateway**: www.nebulus-iot.com