OPEN MANUFACTURING
PLATFORM

Edge Computing in the Context of Open Manufacturing

01.07.2021

LEGAL DISCLAIMERS

THESE MATERIALS ARE PROVIDED ”AS IS.” The parties expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to the materials. The entire risk as to implementing or otherwise using the materials is assumed by the implementer and user. IN NO EVENT WILL THE PARTIES BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS DELIVERABLE OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER MEMBER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

ACKNOWLEDGMENTS

This document is a work product of the Open Manufacturing Platform – IoT Connectivity Working Group, chaired by Sebastian Buckel (BMW Group) and co-chaired by Dr. Veit Hammerstingl (BMW Group).

AUTHORS

Anhalt, Christopher, Dr.	(Softing)
Buckel, Sebastian	(BMW Group)
Hammerstingl, Veit, Dr.	(BMW Group)
Köpke, Alexander	(Microsoft)
Müller, Michael	(Capgemini)
Muth, Manfred	(Red Hat)
Ridl, Jethro	(Reply)
Rummel, Thomas	(Softing)
Weber Martins, Thiago, Dr.	(SAP)

FURTHER CONTRIBUTION BY

Attrey, Kapil	(Cognizant)
Kramer, Michael	(ZF)
Krapp, Chiara	(BMW Group)
Krebs, Jeremy	(Microsoft)
McGrath, Daniel	(Cognizant)

Title Image by Possessed Photography from unsplash.com

Contents

1 Introduction: The Importance of Edge Computing

Current approaches to implement IoT solutions focus on registering, managing, and connecting IoT devices directly to the cloud. The main challenge of these solutions has typically been the transmission of raw data from a device to the cloud where the data was processed, filtered, or aggregated to generate business value.

In our last white paper [OMP IOTCON 2020][1], we described how this direct connectivity approach could not cover manufacturing requirements, creating the need for edge computing. For example, industrial devices typically generate significantly higher volumes of data than conventional IoT devices resulting in increased costs and latency times for the data transmission to the cloud. Particularly in manufacturing, reaction times to critical events must be minimized and special security considerations must be met.

By moving processes down from the cloud to the edge layer at the plant site, edge computing can help mitigate these issues and bridge the gap between data generation, storage, and processing. This helps to improve response times and save bandwidth. This means that some data processing and storage are shifted from the cloud to the edge level in the industrial context, resulting in a tailored combination at both levels to realize Industrial IoT use cases.

Edge computing has recently attracted a lot of interest; however, there is no consensus on a standardized definition and architecture for edge computing. Therefore, this publication approaches the topic of edge computing from a manufacturing use case perspective with three different views on it: an *infrastructural*, an *application*, and an *operational* view. Based on this approach, an edge computing framework's core characteristics and components are identified and described. The main contribution of this paper is to outline edge computing in a manufacturing setting and start moving the sector towards a common understanding.

[1] See https://github.com/OpenManufacturingPlatform/iotcon-connectivity-handbook and https://open-manufacturing.org/blog/2020/12/09/industrial-iot-white-paper/

2 Definition of Edge Computing in the Context of Open Manufacturing

In the domain of manufacturing, we define edge computing as follows:

> In the context of manufacturing, edge computing describes a system of decentralized edge nodes, which reside near the physical origin of the data. Edge nodes must be able to run arbitrary containers and are managed centrally. An edge node connects to both the cloud level and the production asset level and can temporarily run offline.

3 Reference Use Case

To analyze the different aspects of edge computing, we define a reference use case used in the subsequent chapters to describe different technological viewpoints.

In our use case, a company wants to analyze the energy data of a drive and send anomalies to a cloud dashboard. Also, the analysis results are stored in a cloud storage service.

On the production asset level, a PLC[2] is connected to the drive through a real-time ethernet bus. The PLC monitors the high-frequency energy data and is able to publish them for other systems.

Since the amount of data generated is too large to be sent to the cloud without aggregation, a preprocessing must be done. This is executed on the *edge level*. An edge node consumes the energy data from the PLC via a modern protocol (e.g., OPC UA[3]).

The edge node provides two key application features:

1. *Communication*: The edge node acts as a two-way gateway. It establishes a secure connection southbound to the PLC and northbound to the cloud.

[2] programmable logic controller
[3] https://opcfoundation.org/

2. *Parameter analysis*: The edge node has a custom-defined business logic to recognize energy consumption-related anomalies of the drive. The analysis results are pushed to the communication module and sent to the cloud for visualization and further processing. The analysis logic can be a simple threshold detection or a more sophisticated machine learning model.

The edge node can be a device residing near the production floor or a virtual edge node in a plant data center.

On the *cloud level*, the aggregated data is distributed via an enterprise bus to a dashboard for user visualization purposes (hot path). In addition, it is stored in long-term storage (cold path). A process specialist interprets the results from the visualization and sets new parameters for the drive to reduce occurring energy spikes. These are published on the enterprise bus and received by the communication software on the edge device. Afterward, the parameters are set on the PLC and transferred to the drive.

As data accumulates in cloud storage, it is applied to train the machine learning model. New models are pushed down to the edge device at regular intervals to update the parameter analysis.

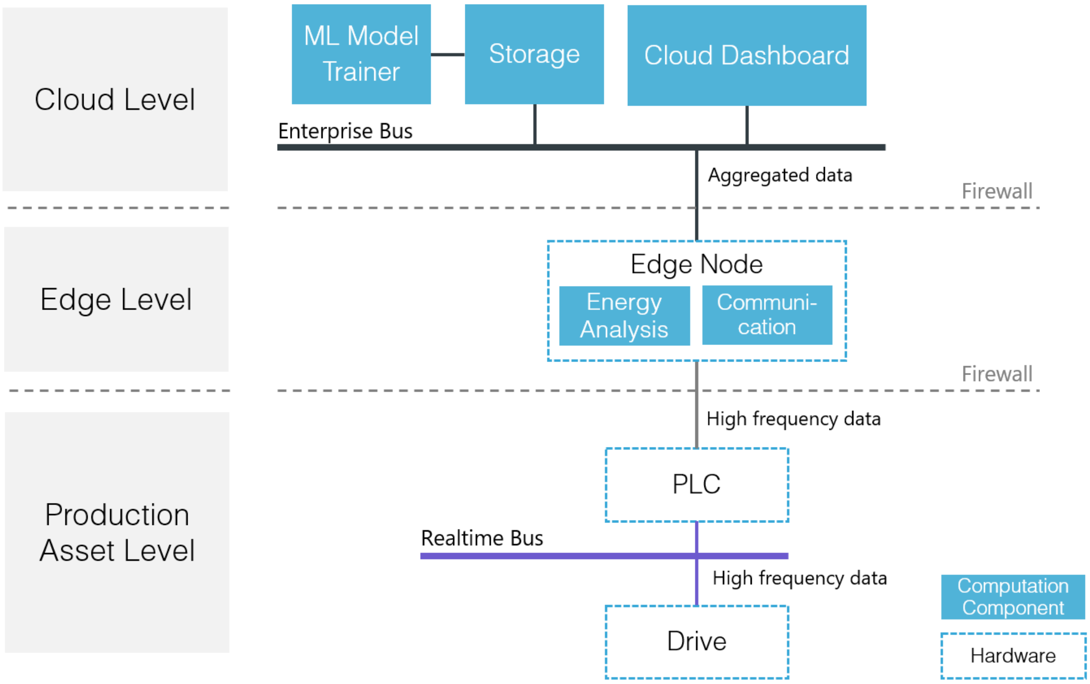

Figure 1: *Reference use case architecture based on specifications in* [OMP IOTCON 2020][1]

4 Views on Edge Computing

This chapter will have three different views on edge computing: an *infrastructural*, an *application*, and an *operational* view.

4.1 Infrastructural View

In general, an edge node consists of a compute node and a guest system. Part of the compute node is the physical hardware, operating system, and hypervisor. It allows running independent guest systems. A guest system consists of the container runtime as well as the containerized software itself.

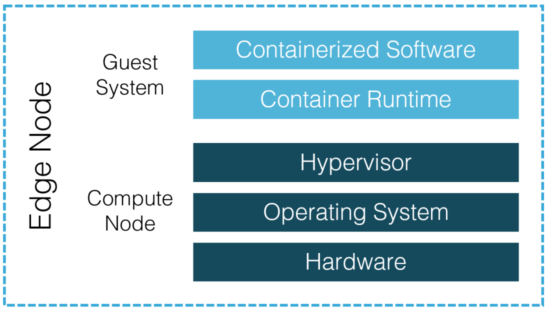

Figure 2: *Infrastructure building blocks of an edge node*

The first defining characteristic for edge nodes, from an infrastructural perspective, is the capability to run and manage containerized software. Therefore, the edge container runtime provides lightweight virtualization, i.e., no virtual machines with their own operating systems.

The second defining characteristic is the ability to connect bidirectional the edge node to the cloud level and production asset level. This means, that a compliant edge infrastructure needs to handle direct or indirect connectivity to the cloud, e.g., through different network layers, and must deal with temporary offline situations. It is essential to point out that cloud connectivity needs to be achieved in a secure manner. Rather than undermining the security architecture that relies on strict network separation (aka perimeter), future platforms need to be aware of the users, data, services, and devices that are each a vital part of manufacturing platforms.

In general, there are two major hosting solutions for edge nodes: virtualized in the plant data center or physical as an edge device on the shop floor level. Datacenter deployments predominantly use container orchestration and have the following benefits over edge devices:

- Higher availability
- Better scalability
- Lower costs for operations (for large scale)
- No additional hardware and wiring in the cell required

Nevertheless, there are also many scenarios where physical edge devices are needed. Examples are:

- Data preprocessing (e.g., filtering, aggregation) in the production cell because network load needs to be reduced
- Run AI-based control loops to achieve lower latency
- The connection of additional peripheral hardware is required
- Offline scenarios (without data center connectivity)
- Translation of non-secure protocol to secured ones
- Low costs and time to usage for initial proof of concept implementations

Possible edge node realizations can be Industrial PCs, Edge Gateways or PLCs.

As a general rule, the hosting level (cloud, data center, device) should be chosen as "high" as possible and as "low" as required by the use case and desired functionality.

Looking at the reference use case, a defining feature that can drive the hosting decision is the large amount of data being pre-processed. If raw data from multiple drives is published on the network, this can lead to bandwidth issues. Therefore local preprocessing on the edge device is beneficial.

On the other hand, the processing power and scalability of the solution must also be considered when an excessive amount of resources are needed for analyzing the data. This may lead to an edge node hosted in the data center and potentially require specific network requirements for throughput optimization.

4.2 Application View

The application view describes functional components deployed on an edge node that are required to fulfill the OT and IT application requirements. The functional components come as containerized software to match architecture and infrastructure requirements as described in the requisite sections.

Typical questions from an application perspective are:

- How can standardized communication within a distributed edge environment up to the cloud be enabled and managed?
- How can diverse data sources be standardized and transformed to move the entire solution towards a common semantic modeling approach?
- How can custom business logic components be integrated with standardized connectivity modules?
- How can data storage between distributed edge nodes be managed?
- Where does business logic reside, and how is it managed between the edge and cloud environment?

Hence, the application view needs to include multiple components with a range of different capabilities:

- Cloud connectivity
- Production asset connectivity
- Components for communication within an edge node and between distributed edge nodes
- Data (pre-)processing
- Data aggregation
- Semantic enrichment
- Components with specific business logic (e.g., edge analytics, machine learning)

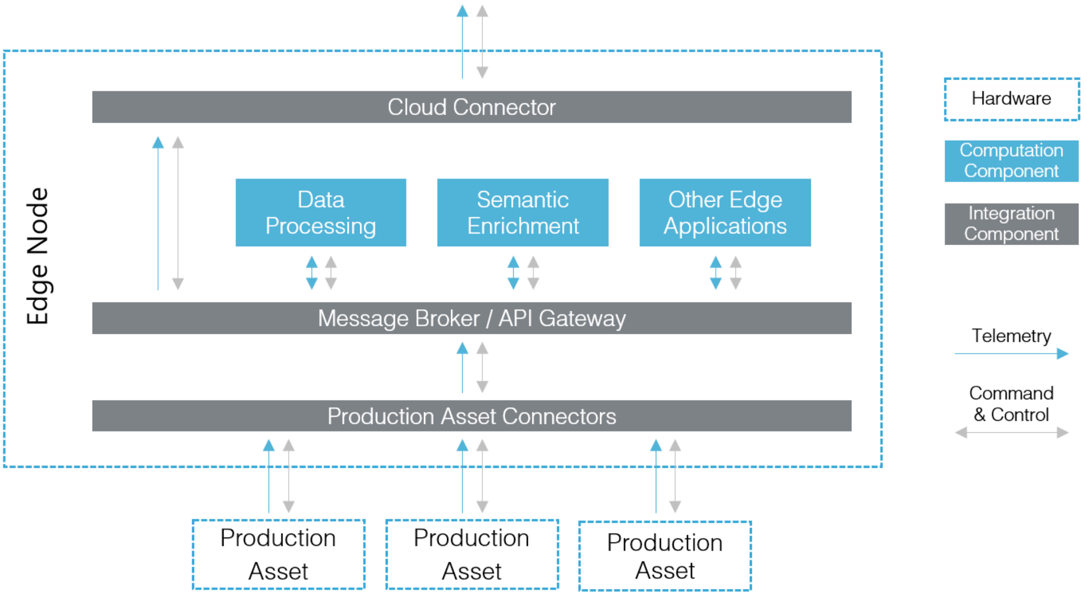

Figure 3: *Application view on an edge node (exemplary)*

Basic connectivity and communication are fundamental for any edge application. The need to implement data pre-processing, aggregation and semantic enrichment functionalities typically scales with the size and complexity of the entire IoT solution (i.e., number and types of data sources, data volume, number of IoT applications, etc.).

The following needs to be considered to give an appropriate answer to the questions listed above:

- Message broker or API gateway to create loosely coupled architectures between edge applications and between edge nodes
- Synchronous and asynchronous communication patterns, depending on the use case
- Defined, flexible payload formats for telemetry and command data
- Information models for semantic enrichment of the data
- Offline scenarios and data buffering capability
- Training of complex machine learning models where computing power is abundant (mostly cloud) and utilized at a level with sufficient access to the data flow while also taking bandwidth constraints into consideration (mostly edge)

In the reference use case, the Product Asset Connector retrieves the energy data from the PLC and forwards it to a message broker. The message broker ensures the exchange of messages between the different edge applications and connectors. The energy analysis component implements a custom-built business logic to recognize anomalies in the collected data. Therefore, the application receives the energy data from the message broker, performs the analyses, and publishes the result on the message broker. The cloud connector transfers the analysis results to the enterprise bus. The reference use case describes that a parameter is also changed on the PLC. The cloud connector and production asset connector transmit the new parameter from the enterprise bus to the PLC via the message broker.

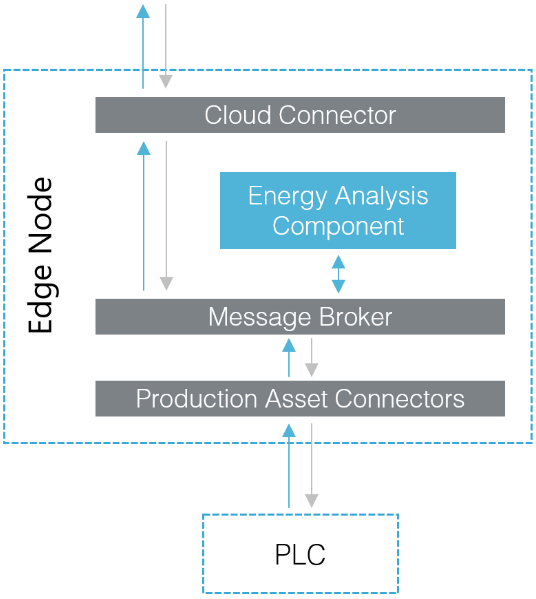

Figure 4: *Application view of the reference use case*

4.3 Operational View

Since applications in the manufacturing sector have special requirements in terms of stability and automation, an integrated operational approach for the edge nodes is essential. This chapter focuses on the functions that are important to manage the edge node through its lifecycle.

As described in chapter 4.1, it is possible to host the edge node either in the plant data center or on a computing unit on the shop floor next to the production asset. From an operational point of view, both options have their own challenges.

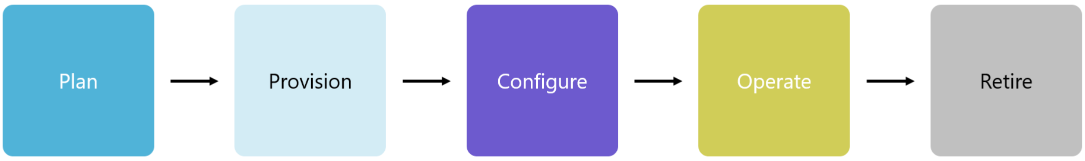

Figure 5: *Device lifecycle phases in [OMP IOTCON 2020]*[1]

An edge node also goes through the characteristic device lifecycle phases described in [OMP IOTCON 2020][1]. Each lifecycle phase has specific requirements for successful execution. These requirements are generally met with additional cloud services. The predominant supporting cloud services are the Edge Node Management & Onboarding Service, the Monitoring Service, the Security Service, and the Edge Node Twin.

Management & Onboarding Service: Generally, this cloud service has the task of fetching the current or setting the desired state of connected edge nodes.

In the *provision phase*, the service onboards new edge nodes. The first step is to create digital identities. The digital identity can originate from existing asset management systems. The service can also save and manage target configurations of edge nodes.

The Management & Onboarding Service provides an endpoint where edge nodes initiate a connection. For this, a basic setup of the edge node must be performed. Example actions are installing the operating system, provisioning the agents, and providing security credentials.

After the initial connection is complete, the edge node offers basic self-describing characteristics (state). A few examples are the firmware version and node settings. The node state is compared with the target configuration from the service. If deviations occur, a state update is sent down to the edge node. State updates can be security or policy updates, changed configurations, and application versions (containers). In our reference use case, a configuration would be the thresholds to identify anomalous energy consumption patterns.

State updates resulting in edge node downtime that interfere with production are not possible at any time. Therefore, careful consideration should be given to scheduling the updates at an appropriate time.

Customized parameters are required to adapt the installed application to the specific use case. These parameters are pushed to the edge node and applied to the application in the *configuration phase*. An example can be a configuration file that is loaded onto the storage (e.g., hard drive) and fetched by the containers on startup. This is the same when updates occur in the *operations phase* (see Edge Node Twin).

The final phase is *retirement*. On *physical* edge nodes, a retirement is initiated by a hardware failure or an upgrade cycle. The hardware is exchanged, and the existing identity with its state is transferred to the new device. To reduce downtime in manufacturing scenarios, this relocation must be done by the service as seamlessly (i.e., automatically) as possible. On *virtual* edge nodes, a failure results in starting a new edge node and in the transferal of the digital identity.

If the edge node can be fully retired, the digital identity is removed from the edge node management service to prevent further access, both in physical or virtual edge node cases.

The **Cloud Monitoring Service** collects all log and metric information from edge nodes. Information can originate from the host system as well as from the running container applications. Therefore, the edge runtime and the applications must support this mechanism.

Via the service, it is possible to create alerting mechanisms (e.g., on allocated memory) which are applied as stream analysis on the incoming data. In the *provisioning phase*, the edge node establishes a connection to the monitoring service. Afterwards, alerts, as well as logs and metrics, can be used throughout the following *configuration* and *operations phase*. They are used to determine the system's health and perform incident traceability.

To obtain meaningful results, the log messages must have a defined format. Common components are message origin, severity, content, UTC timestamp, and correlation id to ensure better traceability.

The **Security Service** is responsible for managing the secure entities of the edge node and its applications. In the *provisioning phase*, the edge node connects to the cloud service the first time, and they exchange their trust entities. In a scenario of large-scale installation of physical edge nodes, a default certificate could be provided, which is valid for the first connection. After the connection to the service, it gets exchanged by the security service. For this, the service needs access to the relevant certificate authorities (CAs).

A second task is policy enforcement. This is done by a comparison of a target state with the current device state. Examples are updating the host system and the application of security rules, like disabling ports.

In the *configuration phase*, the security service supports the edge application configuration by handling security-related tasks, e.g., installing certificates to connect to third-party services. Also, edge applications can manage their secrets in a secure manner through this service. In the *operations phase*, continuous monitoring of vulnerability databases is performed. Any findings can result in potential updates of the target states. In the *retirement phase*, the revocation of the security entities of an edge node takes place.

Another relevant service is the **Edge Node Twin.** Its job is to synchronize application parameters between the cloud service and the edge nodes. Therefore, its main use is in the *operations phase*. Possible configuration parameters are stored in an "edge node twin" representation in the service. Changes made on the twin's parameters are synchronized via the Device Management Service. A possible example is the adoption of a threshold value inside an application.

The parameter is generally reflected as hierarchically organized key-value-pairs. As a prerequisite, the edge container runtime has to be able to receive these parameters and provide them to the container applications. The container applications themselves need to be able to interpret them and change dynamically based on the given values from the twin service.

In our case, the process specialist analyzes the visualized data in the cloud dashboard and then sets new values in the Edge Node Twin to reduce the energy consumption.

5 Outlook

In this paper, different views were taken on the topic of edge computing in a manufacturing context. Each of the views, the infrastructural, the applicational, and the operational view, had a look at basic services and processes needed in different lifecycle phases. Looking ahead, the following topics are the main drivers for future digitized production scenarios:

Device compatibility: Devices complying with the above-described services and architectures are a success factor for scalable and stable rollout scenarios. This will get increasingly important in the future.

Data semantic: Data insights play a central role for most IIoT applications, and the availability of semantic information is a critical component to enable such insights. Semantic information unified across an enterprise is important for further developing of edge solutions for the Industrial IoT and the interaction between edge and central platforms.

Openness: Orchestrating the increasing number of different standards and services will be an important ongoing challenge. In order to create tailored business value, it is important that standards are open in terms of specifications, interfaces, and implementation.

Data sovereignty: Edge computing is a fundamental part of industrial cloud infrastructures to ensure the sovereign usage and processing of data that can be distributed across multiple platforms and vendors. The different usage scenarios and requirements fulfilling sovereignty must be taken into consideration.

The working group will address these in an open manner. If you want to participate, please feel free to contribute to our public GitHub space[4].

[4] See https://github.com/OpenManufacturingPlatform/iotcon-connectivity-handbook