



SAP-GUIDE

SAP DM Shopfloor Integration

April 2026

Authors	4
List of Figures	4
List of Abbreviations	5
Introduction	7
Delimitation and Scope	7
1 Theoretical Fundamentals	8
1.1 Protocol Descriptions	8
1.2 Data Structures, Data Model, Data Protocols.....	9
2 Use Cases	14
2.1 Results Reporting.....	14
2.2 Process Reporting.....	15
2.3 Process Interlocking.....	17
3 Architectural Overview	19
3.1 Reference Architecture.....	20
3.1.1 Integration Layer (Middle Layer)	21
3.1.2 Edge connectivity software	22
3.1.3 Experiences and Challenges in the Implementation of Edge Solutions.....	23
3.1.4 Examples of Edge Connectivity Software and Providers	24
3.1.5 Recommended Action: Edge Connectivity Software.....	26
3.2 Hardware for Machine Connection	27
4 Scaling	28
4.1 Template Approach	28
4.1.1 Data Structuring in the Unified Namespace	28
4.1.2 Governance and Organizational Integration	29
4.1.3 Infrastructure – Business Continuity and Risk Assessment.....	30
4.2 System Procurement.....	31
4.2.1 Procurement Guidelines and Integration with Operating Processes	31
4.2.2 Standardization of Communication Protocols	32
4.2.3 Standardization of the Data Records to Be Exchanged	32
4.3 Redundancies	35
4.3.1 Redundancy and High Availability	35
4.3.2 Logging and Monitoring	35
4.3.3 Recovery Time Objective (RTO) and Recovery Point Objective (RPO).....	36

4.3.4	Service Levels and Operational Organization.....	36
4.3.5	Roles in the Service Level Model.....	36
5	Operation.....	37
5.1	Backup and Restore.....	37
5.2	Operating Concept.....	38
5.2.1	Incident Occurs (Level 0).....	39
5.2.2	Local Analysis and Processing by Key User and Power User (level 1).....	39
5.2.3	Escalation to Central IT/SAP Support (Level 2).....	39
5.2.4	Rectification and Feedback of the Fault.....	39
5.2.5	Standby Concept to Ensure System Operations.....	39
5.3	Role Concept.....	40
5.3.1	Technical Definition of User Roles.....	40
5.3.2	Matching with SAP DM Standard Roles in SAP BTP.....	40
5.3.3	Technical Design of Roles and User Assignment.....	41
5.3.4	Documentation and Governance.....	41
6	Summary and Next Steps.....	41
6.1	Summary of Key Topics.....	42
6.2	Key Messages.....	42
6.3	Recommendations for the Next Steps.....	43
6.4	Outlook.....	44

Authors

Abbre- viation	Author name	Email	Company name
ABu	Andreas Busse	andreas1.busse@zeiss.com	ZEISS AG
DNo	Dorothea Nochelski	dnochelski@deloitte.de	Deloitte Consulting GmbH
PRe	Peter Reu	peter.reu@zeiss.com	ZEISS AG
MSI	Marc Slonek	marc.slonek@zeiss.com	ZEISS AG
SPi	Sascha Pirke	sascha.pirke@mtu.de	MTU Aero Engines AG
AKr	Andreas Kronier	Fehler! Linkreferenz ungültig.	Schüco International KG
MWa	Martin Walther	Fehler! Linkreferenz ungültig.	Schüco International KG

List of Figures

Figure 1: Data type table under VDMA 40001-1:2025-07	10
Figure 2: Example: Euromap 84 V2.0	13
Figure 3: Derivation of the ISA-95 pyramid to create a Unified Namespace	19
Figure 4: Reference architecture	21
Figure 5: Example of a multi-level support process.....	39

List of Abbreviations

Abbreviation	Meaning
AKS	Azure Kubernetes Service (Kubernetes solution from Microsoft)
BCM	Business Continuity Management
BTP	SAP Business Technology Platform
CE	Conformité Européenne (CE marking, European conformity)
DM / SAP DM	SAP Digital Manufacturing
Docker Swarm	Container orchestration tool from Docker
ERP	Enterprise Resource Planning
EWM	Extended Warehouse Management (SAP module for warehouse management)
IoT, IIoT	Internet of Things, Industrial Internet of Things
IT	Information Technology
JSON	JavaScript Object Notation (data format)
KPI	Key Performance Indicator
Kubernetes	Open-source platform for container orchestration
MQTT	Message Queuing Telemetry Transport

Directories

OData	Open Data Protocol (REST-based protocol)
OEE	Overall Equipment Effectiveness
OPC UA	OPC Unified Architecture
OT	Operational Technology
PoC	Proof of Concept
ProdCon	SAP Production Connector
REST	Representational State Transfer (architectural style for web APIs)
RPO	Recovery Point Objective (maximum tolerated data loss period)
RTO	Recovery Time Objective (maximum tolerated recovery time)
SCADA	Supervisory Control and Data Acquisition
SLA	Service Level Agreement
PLC	Programmable Logic Controller (PLC)
UMATI	Universal Machine Technology Interface (standard for machine communication)
UNS	Unified Namespace (central, standardized dataspace based on a broker)
XML	Extensible Markup Language (data format)

Introduction

The integration of SAP Digital Manufacturing with the existing production landscape is a key factor for the successful digitalization of production. To achieve transparency, controllability, and efficiency games in the long term, SAP DM must be properly embedded in shop floor systems and processes.

This Best Practice guide was developed in collaboration with DSAG member companies and serves as a practical guide for the successful introduction and use of SAP DM on the shop floor. It is based on the experience and knowledge gained from real projects that companies have implemented in their production environments.

Our aim is to provide you with valuable insights based on specific examples and proven methods to help you plan, implement, and use SAP DM. In addition to technical aspects, we also look at organizational and process-related challenges, to provide a holistic perspective.

Delimitation and Scope

This guide focuses on the functional and technical integration of production systems with SAP Digital Manufacturing – in particular on architecture, data structures and models, scaling, templates, operational organization, and procurement aspects.

Security aspects (such as network segmentation, access protection, encryption, and authorization concepts) are intentionally not covered in detail. The applicable company policies and manufacturer-specific security recommendations and standards should be consulted with regard to the specific design of security and compliance measures.

1 Theoretical Fundamentals

1.1 Protocol Descriptions

The integration of SAP Digital Manufacturing (SAP DM) into the production environment requires efficient and stable communication between the IT systems and the shop floor devices, machines, and controllers.

Modern factories are often characterized by heterogeneity: Different machine manufacturers, control systems (such as PLC systems), and data formats require interfaces that are flexible and interoperable. This is where standardized protocols come into play, which enable cross-system communication. A suitable protocol supports both the technical requirements of the machines and the business processes in the IT landscape.

This section of the Best Practice Guide provides an overview of the most important protocols and data communication standards when it comes to integrating SAP DM with the shop floor. It focuses both on classic production protocols and modern IIoT approaches, which are becoming increasingly important.

The relevant protocols include:

- **OPC UA (Open Platform Communications Unified Architecture):** A modern, manufacturer-agnostic standard for communication on the shop floor and the secure transmission of data to SAP DM.
- **MQTT (Message Queuing Telemetry Transport):** A lightweight protocol developed specifically for IIoT applications and the transmission of sensor data.
- **REST and OData-based services:** Important standards for the connection between SAP DM and manufacturing devices, particularly for API-based integration.
- **Proprietary control interfaces** (such as Siemens S7, Mitsubishi MC protocol): Commonly used protocols that can address specific machine controllers.

In addition to these cross-industry protocols, there are also production domain-specific standards (semantics) that are tailored to the specific requirements of individual industries. Examples of these include EUROMAP (plastics industry), Weihenstephan (food and beverage), and SEMI (semiconductor industry).

In this guide, we will focus on overarching protocols and deliberately refrain from detailed examinations of these industry-specific solutions, since their use usually depends heavily on the specific requirements of the industry in question.

The selection of the appropriate protocol depends on a variety of factors, such as compatibility with existing shop floor systems, real-time requirements, data volumes,

and security aspects. The long-term scalability and future-proofing of the chosen standard should also be carefully considered.

1.2 Data Structures, Data Model, Data Protocols

Data model:

Every integration project on the shop floor begins with the development of a data model and a data structure. This step starts with an abstract analysis to determine which information (about machines, sensors, and measured values, for example) is to be recorded and how it these objects interact. The subsequent storage and technical implementation are not yet the top priority at this stage. The aim is to logically map the functional requirements and relationships independently of technical details.

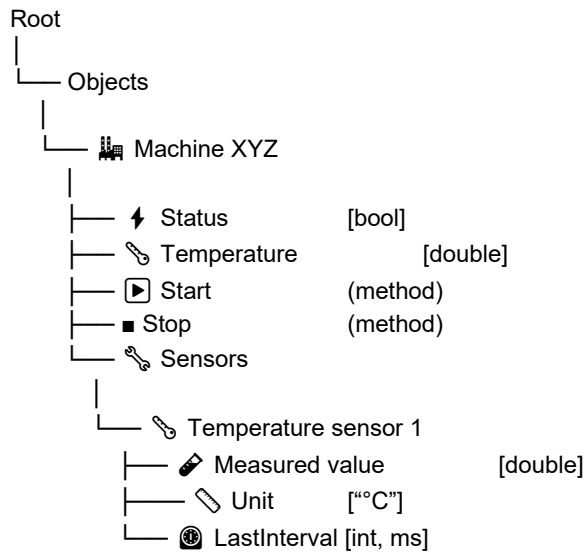
Data structure:

The data structure is usually derived from the data model and describes how this information is technically organized and stored.

An effective data structure is characterized by the following characteristics, which ensure flexibility, comprehensibility, and scalability:

- **Object-oriented structure:**
A hierarchical, object-oriented data model is the focus. Data is mapped in “nodes”, which can be of different types, such as object, variable, method, and data type. These nodes are linked to each other through references, which enable the mapping of complex relationships and structures.
- **Typification and reusability:**
Type definitions (ObjectTypes, VariableTypes) can be used to create reusable models for devices, machines, and processes. For instance, a “temperature sensor” is defined as a type, which means each instance of a sensor can reference this type (see Figure 1: Type definition).
- **Clearly defined metadata:**
Each variable or object can be assigned attributes such as name, description, data type, unit, engineering units, and access restrictions, which promotes interoperability and readability.
- **Namespaces:**
OPC UA uses namespaces to manage extensions and avoid naming conflicts. This enables the integration of custom and industry-specific extensions in a structured manner.
- **Examples of data models**

- **OPC UA** offers an object-oriented structure, for example:



Legend of the illustration:

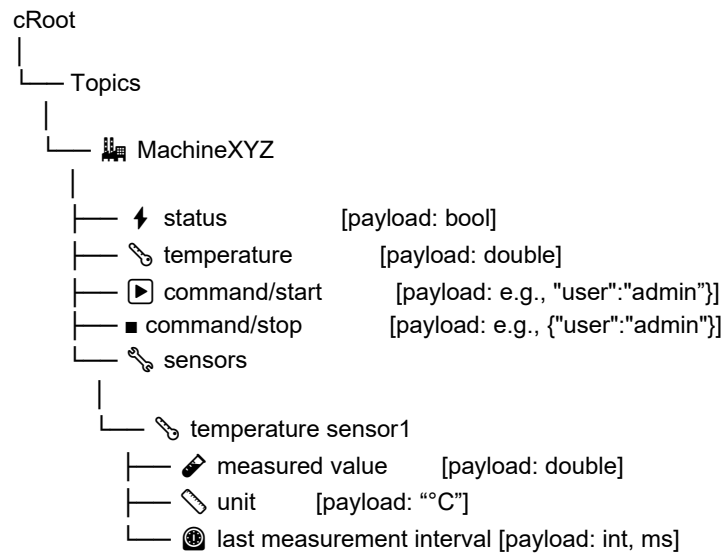
- Objects are displayed as nested structures / tree nodes.
- Variables are labeled as “Variable: Data type”.
- Methods are explicitly identified as such.
- Sub-objects (such as sensors) are located one level below.

Table 2 – Type Definition Table

Attribute	Value				
Attribute name	Attribute value. If it is an optional Attribute that is not set "--" will be used.				
References	NodeClass	BrowseName	Data Type	TypeDefinition	Other
ReferenceType name	NodeClass of the TargetNode.	BrowseName of the target Node.	Data Type of the referenced Node, only applicable for Variables.	TypeDefinition of the referenced Node, only applicable for Variables and Objects.	Additional characteristics of the TargetNode such as the ModellingRule or AccessLevel.
NOTE Notes referencing footnotes of the table content.					
Conformance Units					
Name of ConformanceUnit, one row per ConformanceUnit					

Figure 1: Data type table under VDMA 40001-1:2025-07

- **MQTT** works based on topics. The structure for machine XY could look like this:



Status and temperature values are transmitted as user data (payload) via their respective topics. Commands (such as start/stop) are issued using special command topics, while sensor values are provided separately.

Summary:

A good data structure is based on clear, reusable object and variable types, uses hierarchies, provides meaningful metadata, and uses the namespace mechanism. This keeps the system transparent, expandable, and standardized – for both simple and complex systems.

1.3 Interaction Patterns

This interaction between machines and SAP DM depends on the type of communication and the architecture of the communication participants. The interaction pattern usually describes a cyclical data exchange between the machine and the higher-level SAP DM, triggered by events or commands, which is designed in such a way that smooth production, transparency, and traceability are possible across all production stages. It is generally advisable to implement a standardized structure in the production environment wherever possible. This reduces complexity and helps to simplify administration and safeguard operations.

Typical interaction patterns include:

1. **Request-response, system-driven:** Individual components or machines submit requests (such as material requirements and status reports) to SAP DM. SAP DM responds with releases, instructions, or formulas.
2. **Request-response, MES-driven:** SAP DM sends specific commands to the machines (such as production start and change of formula). The machine confirms the execution or reports errors back.
3. **Event-driven:** Machines or components report status changes, production progress, or disruptions to SAP DM automatically and in real time. The data is processed there and the production data is updated.
4. **Periodic data synchronization (data provision):** Process data is periodically collected and transferred from the machines or components to SAP DM, where it is used for quality assurance, analysis, and visualization.

Standardized interfaces and protocols such as OPC UA, MQTT, and REST APIs are often used to implement these interaction patterns. They ensure reliable, secure, and flexible data exchange between machines (often with controllers such as PLCs) and SAP DM.

Architecture variants

Depending on the structure of the production line and the machines used, there are different communication architectures, as described here with the EUROMAP 84 companion specification, for example:

- **Variante A:** Each machine or component has its own OPC server or MQTT broker and communicates directly with the SAP DM. This enables decentralized data collection, but increases the number of interfaces.

- **Variante B:** A central line controller collects the data from all components and forwards it collectively to SAP DM. Communication with the components can take place via OPC UA, MQTT, fieldbus, or EUROMAP 27.
- **Variante C:** The line controller is integrated into one of the main components (such as the main extruder) and takes over data collection and forwarding to SAP DM from there.

These structures offer a variety of advantages in terms of data integrity, reliability, and administrative effort. The choice of the appropriate architecture often depends on the size and complexity of the line, as well as the desired consistency of the data.

The role of standards

The use of OPC UA companion specifications (such as EUROMAP 84) or MQTT topic conventions facilitates interoperability, because they provide standardized data models and communication methods. This allows SAP DM to communicate efficiently and flexibly with a wide variety of machines and components, for example. The OPC UA companion specification provides a practical overview of the usual structures. The following illustration shows this based on the example of Euromap 84 V2.0.

- Each component has an own OPC server and is connected directly to an MES
- A line control collects all data from the components and forward these to the MES. The exchange between the line control and the components can be realised by OPC UA, but also by other technologies (e.g. field bus, EUROMAP 27)
- The line control is included in an extruder

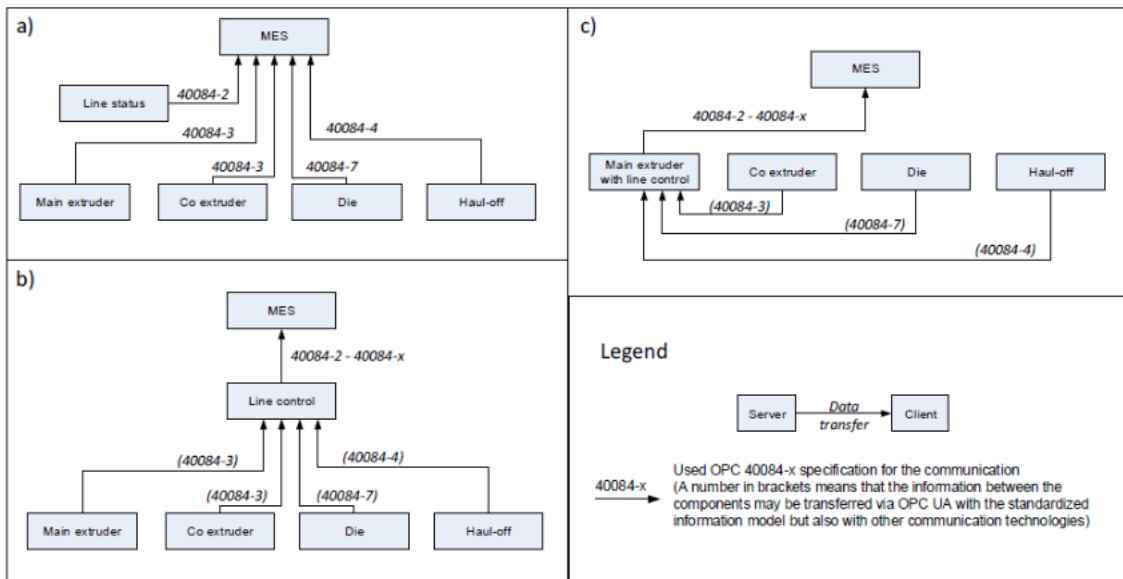


Figure 2: Example: Euromap 84 V2.0

Summary

The clarity of the interaction pattern and the choice of a suitable architecture are crucial for a stable and transparent production process. The variants presented show proven ways of optimally networking machines, line controllers, and SAP DM.

In summary, the established, hierarchical architectures are still justified, but only meet the requirements of modern, highly networked production environments to a limited extent. The Unified Namespace, which is described in detail as a modern architecture variant in Chapter 3, offers a forward-looking answer to this.

2 Use Cases

This chapter deals with the three main use cases in the context of shop floor integration with SAP DM: results reporting, process reporting, and process interlocking. Each of these use cases has its own specific requirements, interaction patterns, and challenges – from the collection of aggregated production KPIs to the combination of real-time data and automated control processes.

The following is described below for each of the three categories:

- What types of data are required to realize the use case
- How this data is modeled, processed, and integrated into the system architecture
- What specific benefits the insights and control options gained have for production

In addition, challenges such as latency times, data quality, and interoperability, all of which must be overcome during implementation, are addressed.

2.1 Results Reporting

Description of the type:

Results reporting involves the collection and aggregation of production process data. This focuses on key figures such as quantity produced, OK/NOK parts (defect rate), capacity utilization of the machines, production progress, and downtimes. It aims to provide an overview of the performance and efficiency of the systems – which is often a central element in the calculation of OEE (overall equipment effectiveness).

Examples of use cases:

Recording the total number of parts produced in a specific period (for instance, per shift, day, number of units per order).

- Recording of OK and NOK parts to evaluate production quality
- Tracking machine running times and downtimes to determine operating time

Goals/business benefits:

- Measure production performance and quality (KPI tracking, especially OEE)
- Get a foundation for reports and dashboards (in SAP DM, for instance)
- Identify inefficiencies and optimization potential in production
- Gain transparency over the production status for management

Required machine data and data model

- Number of parts produced (counter readings, part sensors)
- Classification of OK/NOK parts (for instance based on inspection systems)
- Machine status (ready for production, active, maintenance, downtime)
- Time stamp
- Serial/batch number and order

Amount and relevance of data

- Frequency: Low to medium, depending on cycle/quantity and systems
- Volume per message: Low to medium, depending on the quality parameters to be recorded
- Real-time requirement: Short latency times are helpful, for example, for Andon board, but delayed processing has no immediate impact on the production result

Interaction pattern: Event-driven reporting

Machines regularly send aggregated status data to the edge or middle-layer systems, which is then transferred up to SAP DM.

Typical stumbling blocks:

- Inhomogeneous machine data collection makes uniform aggregation difficult.
- Time stamp synchronization can be complex (due to deviations in machine timers, for instance).
- Data quality (incorrect sensor values or insufficient depth of detail)
- Latencies or interruptions during data transfer; in addition to time delays, this could also lead to inconsistencies during recovery. With regard to interim data storage, see the following chapter

2.2 Process Reporting

Description of the type

Process reporting involves recording and analyzing detailed production data (from sensors and controllers on the shop floor, for instance), to gain insights into the actual

production workflows. For example, data regarding time series sensors can be used to improve processes or analyze problems.

Examples of use cases

- Recording temperature, pressure, vibration, and torque data on machines
- Analyzing time series (such as the processing speed of a conveyor system)
- Identifying process variations (deviating cycle times)
- Training machine learning models for predictive maintenance or process optimization

Goals/business benefits

- Improve process stability and quality by identifying problem areas
- Early detection and prevention of quality problems
- In-depth data analysis for process improvement
- Enable predictive maintenance

Required machine data and data model

- Raw data from sensors (such as temperature, pressure, speed)
- Process parameters (such as lead times, energy consumption)
- Time stamps and close links to serial/batch numbers, orders, and machines

Amount and relevance of data

- Frequency: Medium to very high, depending on the number and recording frequency of the sensors
- Volume per message: Medium to very high, depending on the number of sensors
- Real-time requirement: Short latency times are helpful for immediate intervention in the event of (expected) process deviations; latency only plays a minor role in most use cases

Interaction pattern

- Streaming data: Continuous transmission of sensor data via IIoT systems or middleware (such as MQTT or OPC UA)
- Analysis pipeline: Direct forwarding of data from the edge system to SAP DM or a database/third-party system for storage/analysis

Integration with the Architecture

- With regard to performance aspects, the use of a separate data pipeline and possibly a third-party system for analytics is often recommended when large numbers of data points are involved.

Typical stumbling blocks

- High volumes of sensor data can overload the infrastructure.
- Comprehensible presentation of large amounts of data (for end users, for instance) is often complex.
- Challenges in data preparation and cleansing
- Accuracy and calibration of sensors

2.3 Process Interlocking

Description of the type

Process interlocking describes the direct control of production processes based on specific conditions. The aim is to ensure quality and safety through automatic intervention, for example, when an error occurs. Such interlocking mechanisms are usually time-critical.

Examples of use cases

- Automatic shutdown of a production line when an error is detected (such as excessive temperatures, vibrations)
- Closed loop: Automatic adjustment of production process parameters while the process is running (such as system setpoints). This prevents the risk of scrap in the process and increases quality.
- Safety shutdowns when critical conditions (such as damaged system components) are detected
- Blocking the further processing of defective components

Goals/business benefits

- Increase safety and reduce accidents
- Avoid scrap and damage to machines
- Automated and demand-oriented response to disruptions

Required machine data and data model

- Status data of the machines (such as error codes and alerts)
- Context information, for example, for blocked components

- Process parameters in real time (such as temperature, pressure, amperage), where applicable
- Decisions based on defined limit values or algorithms, where applicable

Amount and relevance of data

- Frequency: Medium to very high, depending on the number and recording frequency of the sensors
- Volume per message: Medium to very high, depending on the number of sensors
- Real-time requirement: Medium to very high. Particularly in the case of hard process interlocking, any latency has a direct impact on the production process. Depending on the lead time, a latency of just 50 ms can mean a significant delay. In other cases (such as lead times of several days/weeks), a normal latency has no significant influence.

Interaction pattern

- Command-driven communication: SAP DM actively sends control commands to the edge system or machine controllers.
- Event-driven actions: Triggering an interlock, for example, via OPC UA to the machine controller.

Integration with the Architecture

- Edge layer, SCADA if necessary: Performs interlocks in real time, because decisions need to be made in the control system (minimizing latency is crucial here)
- SAP DM: Logging and integration with higher-level reporting and analytics processes

Typical stumbling blocks

- Latency in the transmission of signals (real-time requirements!)
- False-positive signals can stop the process unnecessarily
- Susceptibility to errors due to complex logic (for instance, a combination of many event conditions)
- Large implementation effort for integration with different machines
- Lack of backup/redundancy concept with high risk of production downtime

3 Architectural Overview

For a long time, the industrial IT/OT landscape was structured along the ISA-95 automation pyramid. Data flowed in layers from the bottom to the top: Sensors and controllers passed information to SCADA, which in turn passed it to SAP DM, until it finally reached enterprise-level ERP systems. This layered model helped to organize complexity, but also resulted in disadvantages: Data silos, long lead times, costly integration at every level, and high maintenance efforts.

As digitalization in production increased, this model reached its limits. Today, companies need data across all layers – in real time, context-related, and usable for many different systems in parallel. This is where the Unified Namespace architecture (UNS) comes in. Rigid point-to-point connections are replaced by a continuous information backbone that makes all relevant data streams available in a single namespace.

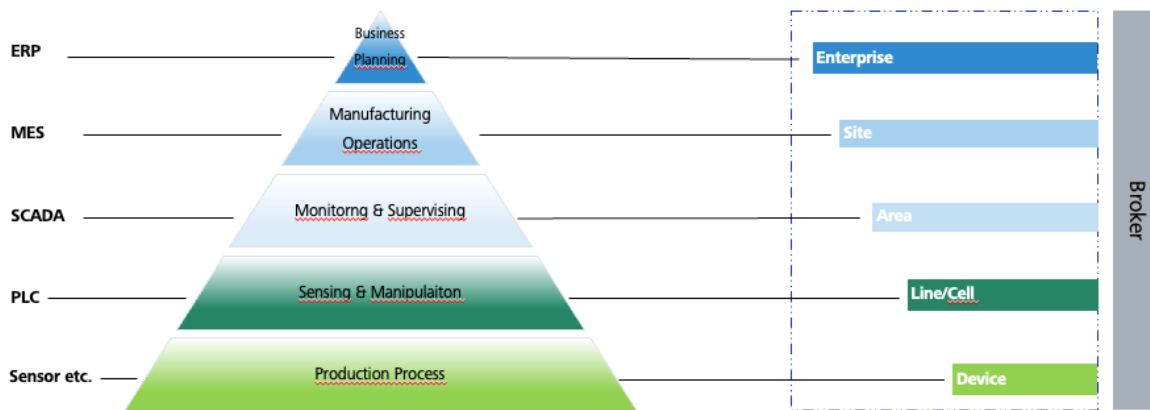


Figure 3: Derivation of the ISA-95 pyramid to create a Unified Namespace

Building on the clear integration patterns and architecture variants presented in Chapter 2, this chapter presents the Unified Namespace, a modern approach that rethinks the networking of machines, line controllers, and SAP DM in a consistent data architecture.

At the heart of this architecture is a broker technology, usually an MQTT broker, which serves as a distributor for all messages. Systems are no longer linked directly with one another, but instead act as publishers or subscribers in the namespace. This creates a flexible, forward-looking model that is seamlessly scalable and “dissolves” the pyramid.

3.1 Reference Architecture

Even though traditional integration based on ISA-95 and point-to-point connections is still possible and works reliably in many environments, this chapter intentionally describes a modern approach based on a Unified Namespace as a forward-looking evolution.

The architecture is divided into three layers that are linked to each other via brokers, according to best practice:

- Enterprise (central aggregation and global IT system connection)
- Site/plant (local contextualization and provision of production-related information)
- Edge (machine and sensor data in real time).

Together, these layers form an end-to-end network that eliminates the classic integration barriers of the ISA-95 pyramid and enables a flexible, standardized data architecture.

The Unified Namespace is therefore best understood as the evolution of the classic ISA-95 pyramid. While the traditional model stratifies data hierarchically and passes it from the machines to the enterprise systems incrementally, the UNS architecture breaks down this strict logic. The enterprise, site/plant, and edge layers remain as orientation, but rigid silos are replaced by an end-to-end communication model.

The connecting element is the broker, which also represents the integration layer (middle layer). Without the broker, there is no UNS, because it is the central component through which all data in the company is published, subscribed to, transported, and contextualized. Broker instances form the foundation at every level: enterprise, site/plant, and edge. They take on the role of the middle layer, mediate between the operational systems, enrich data with context – either themselves or with the help of other connected applications – and make it available to other applications via the namespace.

In this approach, the central enterprise broker acts as a global aggregation point where all site information flows together and is forwarded to ERP, CRM, or analytics platforms. The local brokers at the site or plant level are middle layers themselves, receiving raw data from production, converting it into a semantically usable form, and outputting it to targeted enterprise systems or local applications. Lastly, at the edge level, broker instances are operated directly on machines, controllers, and sensors, to capture data directly where it is generated.

This broker-based architecture transforms the rigid pyramid into a flexible, networked system. Each level no longer communicates exclusively “upwards” or “downwards”, but instead is connected to the entire company network via the broker. The middle layer is therefore not just any additional instance; it is the specific broker level that

brings the Unified Namespace to life and ensures that data does not remain in silos, but instead is available where it is needed.

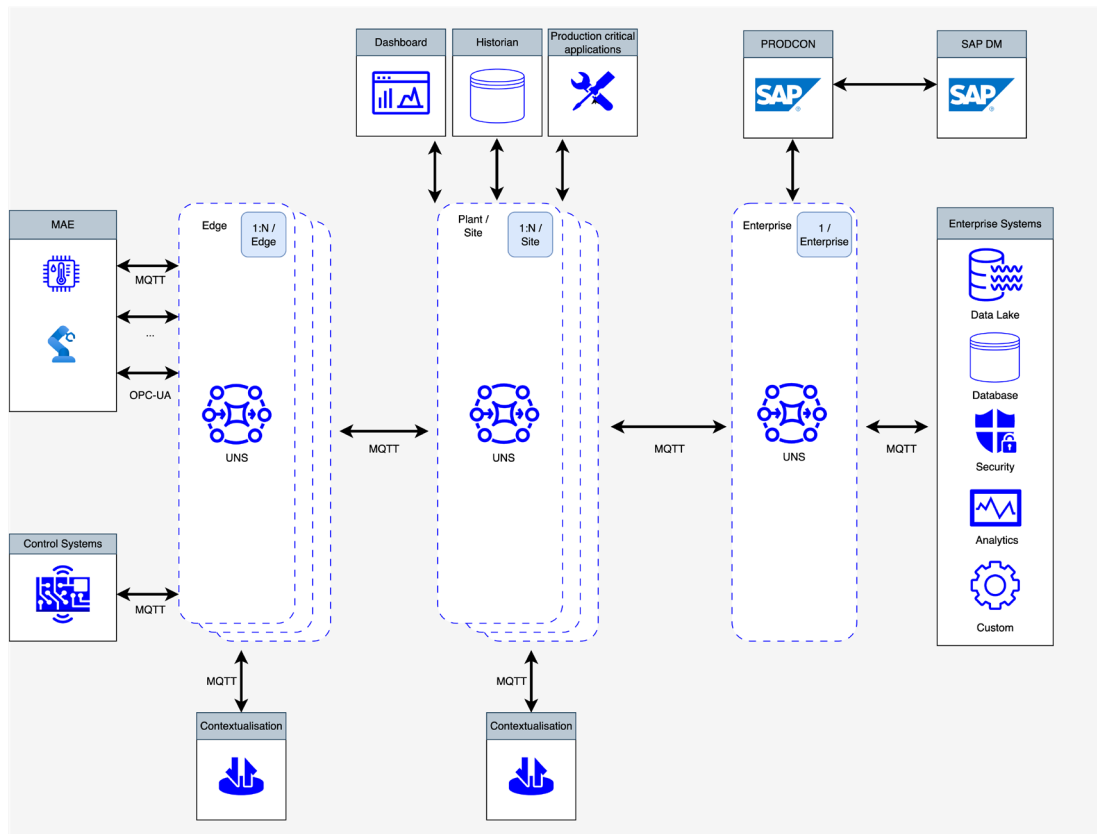


Figure 4: Reference architecture

3.1.1 Integration Layer (Middle Layer)

The middle layer forms the heart of the Unified Namespace architecture – and it is essentially nothing different than the broker platform. The broker is not just a technical tool here; it is the central element that coordinates all data and information flows.

As a broker, the middle layer ensures communication between the IT and OT worlds. On one hand, it integrates shop floor systems such as SCADA, SAP DM, or (local) analysis tools; on the other hand, it connects these systems with IT systems such as ERP, EWM, or quality management. Its role is to abstract the often different protocols, data formats, and requirements of both areas and to provide them with uniform access via topics in the namespace.

The middle layer also contextualizes incoming data. Raw data from production, which often only contains simple measured values or status messages, is enriched with the necessary metadata here – such as version, time stamp, or unit. The data only becomes useful for downstream systems after this processing. As such, the middle

layer not only acts as a transport path, but also as an intelligent translation layer that transforms raw signals into usable information.

The middle layer also fulfills a central filter and relief function. It ensures that only the data that is required and has already been enriched is passed on to the enterprise broker. This conserves bandwidth and simplifies the processing logic in the central IT systems. At the same time, the local broker architecture enables analyses, monitoring, and short-term interventions to be carried out at the site or plant level – even if the connection to the enterprise broker is interrupted.

Its close connection with the broker technology also makes the middle layer highly flexible and scalable. New systems and production lines can be easily integrated by publishing your data in the broker or subscribing to the required topics. Complex point-to-point interfaces are no longer required. This is complemented by fine-grained security and authorization concepts directly at the broker level, ensuring that only authorized users and systems have access to certain data.

This makes it clear that the middle layer is identical to the broker level. It forms the operational hub through which IT and OT are connected, refining raw data and making it usable, and the company dissolves the boundaries of the classic ISA-95 pyramid.

Examples of MQTT Brokers and Providers

In the Unified Namespace architecture, MQTT brokers play a central role as a communication platform between machines, systems, and applications. The following providers and solutions have proven themselves in practice:

- **HiveMQ:** A powerful and scalable MQTT broker developed specifically for industrial applications.
- **EMQX:** An open-source solution known for high scalability and performance.
- **Mosquitto:** A lightweight MQTT broker that is ideal for smaller applications.
- **AWS IoT Core MQTT Broker:** A cloud-based solution from Amazon that enables seamless integration with other AWS services.

3.1.2 Edge connectivity software

Choosing the right edge connectivity software is a key success factor for integrating SAP Digital Manufacturing on the shop floor. These software solutions make it possible to connect machines, controllers, and sensors to IT systems and support a variety of communication protocols. They are essential for the preprocessing, aggregation, and forwarding of data. Solutions such as SAP ProdCon for SAP DM and similar products offer different approaches and functions.

These solutions differ in the following areas in particular:

Supported Protocols

The different software solutions support a variety of communication protocols. Common protocols include OPC UA, MQTT, RESTful APIs, and others. The ability to support multiple protocols is a decisive argument for the flexibility and adaptability of the software. It enables seamless integration with different system landscapes and facilitates communication between machines and applications.

Installation Types

The installation types range from on-premise solutions to cloud-based and hybrid approaches. There are also classic client installations and modern, container-ready applications. Companies should choose the right solution based on their security requirements, infrastructure, and budget. In addition, these edge connectors are often part of comprehensive IIoT platforms, middle-layer software, and MQTT brokers.

Scalability and Customizability

The scalability of the software is a key factor, especially when it comes to implementation, maintenance, and updates. Companies looking to expand their production benefit from customizable solutions that can meet specific requirements and adapt to changing conditions.

Ease of Use

The software should be intuitive to ensure easy implementation and use. A user-friendly interface facilitates training and day-to-day operations, leading to greater acceptance and efficiency.

3.1.3 Experiences and Challenges in the Implementation of Edge Solutions

Supported Protocols

Practice has shown that connecting every exotic interface is not always profitable. It is advisable to modernize shop floor equipment with regard to interfaces and to specify clear standards for new purchases.

Integration with Existing Systems

The integration of edge connectivity software with existing greenfield and brownfield infrastructures can be complex and requires careful planning and implementation. Installations close to the shop floor, such as on IIoT PCs in or on machines, bring new challenges such as loss of warranty or modification of the machines (CE), and physical access is more difficult for IT staff.

Freeware vs. Paid Solutions

When choosing connect software, companies should consider whether they prefer freeware or established providers that offer support, continued development, and scalability.

Data Volumes

Increased data volumes, cycle times, and processing speeds in downstream systems pose challenges, especially when slower protocols (such as REST) are used in the third-party environment.

Shop Floor Data

All relevant data must be carefully recorded on the shop floor. Connect software supports this process by directly retrieving data from machines, calculating, enriching, standardizing, and adding units.

Understanding Roles and Collaboration

A clear understanding of roles and collaboration between maintenance, production staff, the IT department, and data consumers are crucial to the success of the implementation and operation of edge solutions. Team members should have the necessary know-how and understanding to work together effectively. For example, machine downtime for maintenance measures now has different consequences – a rethink is needed on the shop floor.

Use Cases

Initially, a specific use case was identified and implemented, which quickly aroused interest in production for other use cases like energy management, overall equipment effectiveness (OEE), traceability, predictive maintenance, the integration of manufacturing execution systems (MES), condition monitoring, production planning, and process automation. Many of these use cases are aimed at different business applications or only require a subset of the collected data.

3.1.4 Examples of Edge Connectivity Software and Providers

The following providers and solutions have proven themselves in practice and offer different functionalities for edge connectivity.

Edge Connectivity Layer: Typical Components

The edge connectivity layer forms the basis for connecting machines and devices to the IT systems. It ensures that data from machine controllers, sensors, and other shop floor components can be collected, processed and forwarded to the middleware. Typical solutions and providers in this area include:

- **OPC UA Server (machine or gateway):** OPC UA servers are essential for standardized communication between machines and IT systems. They enable the provision of structured data models and interoperability between different devices.
- **Keplware:** A leading industrial connectivity solution that supports a wide range of protocols and enables easy integration with existing systems.
- **Siemens Industrial Edge:** A platform specifically designed to integrate Siemens machines and controllers, allowing seamless data processing at the edge.
- **Unified Automation:** Provides flexible OPC UA-based solutions for machine connectivity and data integration.
- **Codesys OPC UA:** A software platform that enables simple and standardized communication for programmable logic controllers (PLCs).

Protocol Gateways

Protocol gateways play an important role in the translation and mediation between different communication protocols. They are useful in heterogeneous production environments where machines and systems work with different protocols. Common providers include:

- **Anybus Gateways:** Support a variety of protocols and enable the integration of devices with different communication standards.
- **HMS Networks:** Offers a wide range of industrial communication solutions, including protocol translation gateways.
- **Softing EdgeConnector:** A powerful solution for connecting OPC UA and other protocols to IT systems.

Edge Data Collectors

Edge data collector solutions are designed to collect and process data directly at the machine and forward it to higher-level systems. They often offer additional functions such as the preprocessing, filtering, and aggregation of data. Examples of such solutions include:

- **Ignition Edge:** A modular platform that collects and processes data from machines to make it available for analysis and control.
- **Azure IoT Edge:** A cloud-based solution from Microsoft that enables data processing and analysis directly at the edge.
- **AWS IoT Greengrass:** A solution from Amazon Web Services that enables IIoT devices to connect to the cloud while supporting local data processing.

3.1.5 Recommended Action: Edge Connectivity Software

1. Interface standardization

- Define uniform standards (such as OPC UA, MQTT)
- Avoid exotic/proprietary interfaces
- Check the economics of retrofit vs. modernization

2. Select installation type

- Select on-premise, cloud, or hybrid based on security, infrastructure, and budget
- Give preference to container-ready, modular solutions
- Check whether standalone or part of an IIoT platform

3. Ensure scalability

- The solution must grow along with data volume and use cases
- Retain the option of adapting protocols and data models

4. Pay attention to ease of use

- Select an intuitive user interface
- Define training and responsibilities
- Enable updates and maintenance without stopping production

5. Plan integration

- Carry out an inventory of greenfield and brownfield systems
- Consider risks (voiding warranty, CE)
- Close cooperation between production, maintenance, and the IT department

6. Review costs/benefits

- Only use freeware for tests/pilots
- Choose established providers with support for production operations

7. Keep a handle on data volumes

- Dimension the network and back-end performance
- Use efficient protocols (MQTT > REST)
- Preprocess, standardize, and aggregate data at the edge

8. Clarify roles and collaboration

- Define role model between the IT department, production, and maintenance
- Make the effects of maintenance downtimes transparent
- Promote interdisciplinary collaboration

The detailed definition of user roles and their transfer from functional requirements to technical roles and authorizations in the SAP system is described in detail in Chapter 5 – in particular section 5.3.1.

9. Implement use cases in stages

- Start with a clear initial use case (such as OEE or energy management)
- Gradually expand to other applications (traceability, predictive maintenance, MES, condition monitoring)
- Take advantage of the modularity of the software to avoid system discontinuity
- Involve the specialist department from the outset (acceptance)
- Select the use case with the greatest benefit and degree of dissemination

3.2 Hardware for Machine Connection

The integration of SAP DM is limited in terms of interfaces and communication protocols. Practice shows that middleware is required to connect to a wide variety of devices. This practical experience results in the following hardware recommendation:

- 1) OT router as a central data hub on the machine
 - Network segmentation for separation from the business network
 - Remote maintenance via Rendezvous server
- 2) Enlyze Spark
 - a. Collect and standardize process data
 - b. Edge device for process data in the cloud or for SAP DM
- 3) Microsoft AKS / Suse Edge stack (Kubernetes cluster)
 - a. SAP DM Edge
 - b. Middleware for virtualizing the machine connections

To ensure the greatest possible self-sufficiency, the aforementioned components are installed on each machine.

4 Scaling

4.1 Template Approach

The template approach to shop floor integration aims to implement machine connections consistently and efficiently across plants and lines. By defining reusable configuration modules, integration projects can be standardized and accelerated. The foundation consists of interoperable communication structures and standardized data models that enable the flexible connection of a wide variety of machines and control systems. Templates serve as a technical and organizational reference for implementation in various production areas. The structured storage and use of this data in the Unified Namespace supports reusability and scalability.

Things get more complex when it comes to mapping variables, that is, assigning machine values to logical data points in the connector. The required effort here scales directly with the number of machines and can lead to high resource requirements for major rollouts. A well thought-out template approach can significantly reduce the workload here. Once a standardized variable model is created, the configuration for additional machines can be adopted much more quickly. The prerequisite for this is early standardization of the variable names and data structures, tailored to the most important use cases and the connector technologies used. Templates should contain both the technical configuration and the semantic structure of the data points.

4.1.1 Data Structuring in the Unified Namespace

Using a Unified Namespace enables you to organize the configuration of shop floor assets more effectively. All relevant data points – such as status, measured values, conditions, and KPIs – are stored in a consistent, hierarchically structured information space. This enables the standardized, contextual mapping of machine variables, regardless of location or the specific machine instance.

By using naming conventions and semantically clearly defined topics, you can create templates for variable structures that can be reused between plants. For example, an OEE template for a machine class can be maintained centrally and rolled out locally, without having to develop individual mapping logic for each plant. The UNS serves as the technical backbone for data provision and processing in the connector.

A typical example is the integration of an OPC UA server under the UMATI standard. UMATI defines standardized information models for machines such as machine tools. The OPC UA server provides structured data points, such as operating status, quantity, and energy consumption. This information is retrieved via the Production Connector and processed further in SAP Digital Manufacturing. Mapping takes place via browsepaths such as */MachineTool/Production/PartCount*, which are mapped to variables such as *machine.production.count* in the connector. The configuration can be done manually or via mapping files and uses the Unified Namespace for semantic classification.

Another example of the use of a Unified Namespace arises with MQTT-based architectures. Here, data is provided via hierarchically structured topics, such as */acme/munich/assembly/line1/machineA/temperature*. A sensor publishes regularly under this topic and an edge service or SAP DM can subscribe to it. If a defined threshold value is exceeded, such as temperature > 80 °C, the subscriber initiates an event-driven trigger – for example by publishing an alert under */acme/munich/assembly/line1/machineA/alert*. This pattern is asynchronous, decoupled, and highly scalable and is ideal for modern IIoT environments.

REST-based integration can also be embedded in the Unified Namespace. One example is the transmission of production orders from an MES or ERP system to a SCADA system. An HTTP POST request is sent to the SCADA API. The SCADA system processes the data and returns a confirmation.

```
1  {
2    "orderId": "PO-2025-00123",
3    "productCode": "XG-450",
4    "quantity": 100,
5    "startTime": "2025-11-08T08:00:00Z",
6    "endTime": "2025-11-08T16:00:00Z",
7    "machineId": "LINE1-MA01",
8    "parameters": {
9      "temperature": 180,
10     "pressure": 5.2,
11     "speed": 1200
12   },
13   "priority": "high"
14 }
15
```

The combination of template and namespace reduces the effort required for mapping variables and creates the foundation for a scalable, maintainable, and transparent data architecture on the shop floor.

4.1.2 Governance and Organizational Integration

A sustainable template approach requires clear governance structures and close organizational integration. Templates should be developed centrally and have centralized version management, but rolled out and operated locally. They are released via defined processes that also ensure the involvement of relevant stakeholders from the IT department, production, and maintenance. Changes to the template are subject to controlled life cycle management, which ensures transparency and traceability.

A proven model consists of a central, global owner who develops, maintains, and releases the templates, as well as local owners in the respective plants who are responsible for implementation and operation. Regular coordination – for example, in a shop floor integration change board – ensures that changes are coordinated, local requirements are taken into account, and the rollout remains consistent across locations.

Unified Namespace and change management

The Unified Namespace is typically located in OT and therefore in the specialist area. Changes to infrastructure templates – such as new indicators, communication interfaces, or mapping logic – have a direct impact on IT integration. Such changes must therefore be submitted to the IT department through the change board as an official change request. This ensures that all dependent systems and interfaces remain consistent and no integration problems arise.

Every change to the template requires version management and rollout to all existing applications, to avoid inconsistencies. This applies to configuration on edge devices as well as cloud-side modeling. Regression tests must be carried out with each new version to ensure that existing functionalities are not impaired. Ideally, the rollout should follow a clear release plan that also includes fallback strategies and escalation paths.

4.1.3 Infrastructure – Business Continuity and Risk Assessment

The SAP Production Connector can be installed in different ways, where each variant has specific advantages and disadvantages in terms of business continuity management (BCM), maintainability, scalability, and risk management.

The classic variant is local installation on one Windows server per plant. It offers close proximity to the machine and the related low latency times. This solution is particularly attractive in plants with a limited IT infrastructure or for retrofit projects, as it is easy to operate and independent of central systems. From a BCM perspective, it enables a fast recovery strategy in the event of faults, because the systems are isolated locally. At the same time, however, risks arise from inconsistent software versions, increased maintenance costs, and limited scalability.

An alternative is installation on Linux-based systems. This variant is suitable for edge devices with a higher IT maturity close to production. It offers advantages in terms of stability, security, and automation, particularly due to the ability of using script controls to roll out deployments. The main risks here are the higher level of expertise required and possible compatibility problems with proprietary agents and machine protocols.

Containerization via Docker Swarm can be a practical solution for smaller clusters or pilot projects. It allows for quick setup, good isolation of individual instances, and simple recovery mechanisms. However, Docker Swarm is limited in its orchestration capability and is less suitable for mission-critical, scalable production environments.

Installation on a centralized Kubernetes platform is the preferred solution for organizations with multiple plants and the goal of unified governance. Kubernetes offers highly available clusters, self-healing capabilities, and automated rollouts. From a BCM perspective, this enables centralized control of security policies, monitoring, and recovery strategies. At the same time, risks arise due to the complexity of operations, potential latency problems with remote plants, and the need to secure central components such as databases and message brokers separately.

A hybrid structure consisting of more than one of the aforementioned options can make sense if it is clearly delineated and supported organizationally – for example, through dedicated responsibilities, automated test pipelines, and centralized change management. For companies with high standardization plans and limited resources, however, a consolidated architecture is recommended to ensure maintainability, security, and scalability in the long term.

4.2 System Procurement

4.2.1 Procurement Guidelines and Integration with Operating Processes

As part of the procurement process, a binding procurement guideline must be defined to ensure that new systems are requested and delivered in accordance with existing templates. This applies in particular to the communication capability (such as OPC UA with Umati profiles), the data structure, and the openness of the interfaces.

A strategic and financial assessment must be made as to whether machines should be ordered directly with a standardized OPC UA Umati profile – which promises a high level of interoperability and less integration effort, but requires costly adjustments by the manufacturer in the event of subsequent changes. Alternatively, a variable catalog with maximum openness can be promoted, in which the machine provides as many raw data points as possible. Semantic standardization then takes place within the company in the middleware – for example, via a mapping script that maps the variables to internal templates and indicators. This variant offers more flexibility and control, but requires stronger internal governance and technical expertise.

The templates should be closely linked to the operating and procurement processes, especially for retrofit projects and the integration of new machines. Enhancements should ideally be made in line with specific use cases. Responsibilities and escalation paths should be documented in the template, to enable a rapid response in the event of faults.

In addition to the technical configuration, templates should also contain organizational framework conditions such as role models, approval processes, and operating guidelines. This ensures that the solution not only functions technically, but is also sustainable operationally and can be continuously enhanced.

4.2.2 Standardization of Communication Protocols

A key lever for the efficient integration of SAP Digital Manufacturing (SAP DM) with the shop floor is the standardization of hardware connectivity. Although this is not a mandatory prerequisite for implementation, it offers high efficiency gains in data integration and processing. To achieve this, companies should define clear requirements for the communication capabilities of the hardware during the procurement process and implement them consistently.

The minimum goal of defining requirements like this is for all new machines and devices to support standardized communication protocols (such as OPC UA, MQTT, or REST APIs), to simplify integration with the IT landscape.

4.2.3 Standardization of the Data Records to Be Exchanged

In addition to ensuring support for the communication protocols, it is advisable to formulate specifications for standardized data models to ensure that the generated data structures and semantics can be seamlessly integrated with the existing data architecture and SAP DM without the need for complex mapping.

The development of a “Connectivity and data modeling standards guide” for procurement, which clearly defines expectations for data interfaces, recommended protocols and available data formats, can serve as a supporting measure. This not only facilitates communication with suppliers, but also increases long-term interoperability within the machine park. Manufacturers can thus ensure from the outset that their deliveries comply with the company’s connectivity standards.

A consistently implemented, standardized data model ensures that hardware connectivity can be used efficiently and that integration with SAP DM and other digital manufacturing solutions has a solid foundation.

1. Basic Principles of the Data Model

- **Interoperability:** The data model should be compatible with common standards such as OPC UA, ISA-95, or MQTT, to enable easy integration with existing systems.
- **Modularity:** The model should have a modular structure, so that additional data and new devices can be integrated flexibly.
- **Standardized semantics:** Standardized naming and definitions for attributes and parameters facilitate further processing of the data. For example, “temperature” should be defined identically as an attribute in every machine.
- **Scalability:** The model must work just as well in small environments as it does in complex, distributed production systems.
- **Transparency:** All the data that is captured should be traceable, with clear documentation on source, data type, unit, and purpose.

Example Structure of the Data Model

The structure of this kind of data model should cover several levels:

Device/sensor level: Physical data sources

- Identification data: Serial number, manufacturer, model, location.
- Process parameters: Operating status (run/idle/error), temperature, pressure, torque, material consumption.
- Status data: Operating times, alert/error codes, maintenance status, energy measurement.

Shop floor level: Aggregation level

- Machine KPIs: Generation of key performance indicators (KPIs) such as OEE (overall equipment effectiveness), availabilities, utilization, scrap rate.
- Process units: Grouping of devices/machines into logical units (such as a production line or production cell).
- Event logs: Record events such as downtimes, maintenance, and alerts in a standardized structure.

Company IT level: Business processes and decisions

- Production order data: Link to SAP systems (such as SAP ERP or SAP S/4HANA) for production plans, material requirements, and completion confirmations.
- Traceability data: Tracking of individual batches or serial numbers through the entire process chain.
- Quality data: Recording of inspection characteristics and deviations by production process.

Technical Implementation

- Data protocols: OPC UA as a basic protocol to map metadata and content in a hierarchical and standardized way. For IIoT applications, MQTT could be used for real-time data transfers.
- Data formats: Use of universal formats such as JSON or XML for the structured mapping and transfer of data.
- Uniform and metadata standards: Binding units (for instance, ISO units of measurement like °C, bar) and meta information for each data point (such as time, source, unit).

Example of a Data Structure for a Machine (JSON Format)

```
{
  "machine_id": "M12345",
  "manufacturer": "XYZ Manufacturing",
  "model": "ABC1000",
  "location": "Shop floor 1",
  "status": {
    "operational": true,
    "current_state": "RUN",
    "temperature": 85.5,
    "pressure": 3.4,
    "last_maintenance": "2023-10-01"
  },
  "kpis": {
    "oee": 87.2,
    "availability": 92.3,
    "performance": 89.4,
    "quality": 97.6
  },
  "events": [
    {
      "event_id": "E56789",
      "type": "Alert",
      "time stamp": "2023-10-03T14:22:11Z",
      "description": "Overheating detected",
      "severity": "High"
    }
  ]
}
```

4.3 Redundancies

The SAP Production Connector is a production-critical integration module between the shop floor and SAP Digital Manufacturing and must be operated in a fail-safe manner. It runs as a Windows-based application. This chapter describes the conceptual building blocks for fail-safe operation (redundancy/availability, RTO/RPO, logging/monitoring, service levels, and role models).

4.3.1 Redundancy and High Availability

The SAP Production Connector consists of a main service (Windows service) and agent instances that run as independent Windows services in the background and implement the data connection to shop floor sources. Several agent instances can be operated in parallel and independently of each other; their numbers are essentially limited by the capacity of the host. In the event of fatal errors, an agent instance can stop and is then displayed as “interrupted”; the status of the agent instances can be monitored at runtime.

The communication paths are relevant for the availability analysis: Calls from the cloud to the Production Connector are made via the Cloud Connector, while connections from the Production Connector to the cloud are established directly. Communication between the Cloud Connector, Production Connector, and SAP Digital Manufacturing takes place via RESTful web services; internally, communication between the host (cloud services) and agent instances takes place via WebSocket.

Consistent endpoint and certificate parameters must be ensured for redundancy and failover scenarios: The internal host is managed as an HTTPS URL in the format “https://<Fully Qualified Domain Name>:<Port>”, and the FQDN should correspond to the common name (CN) of the server certificate used. In addition, multiple Production Connector systems cannot be defined with the same internal host URL.

For Windows-based workloads, high availability can be implemented via cluster/failover mechanisms at the platform or infrastructure level, in which multiple nodes are operated together and, if one node fails, other nodes take over the workload (failover), while roles are continuously monitored and restarted or migrated if necessary. In addition, recovery settings can be defined for Windows services, to restart services in a controlled manner after faults or to trigger defined recovery actions.

4.3.2 Logging and Monitoring

A reliable logging and monitoring concept is a central component of fail-safety. The SAP Production Connector can be integrated with logging infrastructures such as Elasticsearch to document operating data, error states, and communication events in a traceable manner. In combination with tools such as Kibana or Grafana, dashboards can be created for real-time monitoring that map both IT and OT-relevant metrics.

Monitoring and logging are essential for:

- Early detection of faults and performance problems
- Traceability of communication errors
- Support for restore and recovery processes
- Compliance with SLA specifications through transparent operating data

4.3.3 Recovery Time Objective (RTO) and Recovery Point Objective (RPO)

The RTO describes the maximum period of time that a system may require to be operational again after a failure. In production-related environments with high cycle times, an RTO of less than 2 hours is recommended, especially for central installations or cloud-based scenarios.

The RPO defines the maximum tolerable period of time in which data may be lost. With synchronous data replication, an RPO of less than 5 minutes can be achieved. In decentralized setups with manual backups, a higher RPO is realistic, but should be clearly documented and communicated.

These values can be used as target values for local or hybrid Production Connector installations, but must be adapted to the respective infrastructure and operating strategy.

4.3.4 Service Levels and Operational Organization

A service level agreement (SLA) is a central component of the operating strategy for production-critical systems. It provides binding specifications as to which services, availability, and response times are expected during operations and how to respond in the event of a fault. SLAs create transparency and reliability between internal areas such as IT and OT, as well as with external service providers.

A well-defined SLA enables:

- Reliability: Clear responsibilities and measurable performance metrics.
- Transparency: Everyone involved knows the expectations and processes in the event of a fault.
- Predictability: Resources and processes can be coordinated with defined recovery times.
- Escalation: Defined escalation paths take effect in the event of non-compliance.

4.3.5 Roles in the Service Level Model

In a distributed plant structure, a tiered service organization is needed to implement SLA specifications efficiently:

- Key users (specialist department/production): First point of contact in the event of operational disruptions. Reports problems, documents incidents, and helps to describe faults.
- Local IT/OT: Responsible for first level support. Carries out initial analyses, checks the local infrastructure, and forwards to central departments if necessary.
- Central IT/product owner: Responsible for second level support, template management, system configuration, and coordination with SAP and external service providers. Is responsible for compliance with SLA specifications and the further development of the operating strategy.

5 Operation

5.1 Backup and Restore

The backup and restore function is used to back up and restore the Production Connector configuration in emergency scenarios (such as failure or downtime of the installation). Configuration elements that are defined in SAP Digital Manufacturing and deployed in the Production Connector are backed up. The function is not intended as a generic mechanism to return to any historical system state (“rollback”), but instead to restore a consistent configuration in the Production Connector.

The backups contain the complete configuration of the Production Connector, in particular the relevant connectivity objects and assignments (such as systems, agent instances, shop floor systems, users, and user groups, including role assignments) as part of the connectivity configuration.

To create backups in SAP Digital Manufacturing, the user in question must have the authorization to create backups. The restore may only be carried out by a user authorized as an administrator. The certificate required for cloud-to-production connector communication must also be available for a successful restore.

SAP Digital Manufacturing creates automatic configuration backups. This automatic backup is updated with each new deployment execution and stored as a .pbf file in SAP Digital Manufacturing. The automatic backup serves as the primary restore base to restore the last consistently deployed configuration.

In addition to the automatic backup, a manual backup can be created using the “Configure Production Connectivity” app. To do this, the relevant Production Connector instance is selected in the app and the “Create Backup” function is executed in the “Backup” area. A manually created and downloaded backup is used primarily to support error analysis (issue/support case). It can also be used if a local backup file is required in exceptional cases (for instance, for transfer to support as part of diagnostics).

The current backup is downloaded using “Download Backup”. A password is assigned during the download, because the file contains encrypted content. This password is required when restoring from the downloaded file. The password must be stored reliably, because it is not possible to restore from the downloaded file without the password.

Restore is used when a Production Connector installation fails or the configuration needs to be restored after a disruption event. A restore completely replaces the existing configuration with the restored version. No new backups can be created and no configuration changes can be made during the recovery process.

The restore is initiated in the “Configure Production Connectivity” app: The affected instance is selected in the “Production Connector” tab; “Restore Backup” is executed in the “Backup” area and the restore process is started. If a downloaded or locally created backup file is used, the password assigned during the download must be entered. The preferred approach is to use the backup available in the cloud as the basis for the restore as the preference; local or non-host backups should only be used in justified exceptional cases.

Once the restore has been completed, it must be ensured that the Production Connector instance in SAP Digital Manufacturing is correctly reconnected/assigned and that further deployments are only carried out after successful synchronization and a stable connection has been achieved. Operational capability must then be verified through suitable functional checks (such as the status of the agent instances, connections to relevant shop floor systems, and ability to communicate with the cloud).

Operational processing in the event of a fault (ticket creation, communication, escalation, readiness, and return of solutions) is carried out in accordance with section 5.2. The assignment and governance of the required roles and authorizations (in particular administrator rights for restore) are regulated in the role concept in accordance with section 5.3.

5.2 Operating Concept

The following process stipulates that incidents are always initially recorded and processed locally by specialist staff, key users, and power users (level 0/1), and are only escalated to central IT and SAP support levels and external consulting (level 2/3) when necessary, who rectify the incident and send the solution back to the site.

Delimitation and Scope

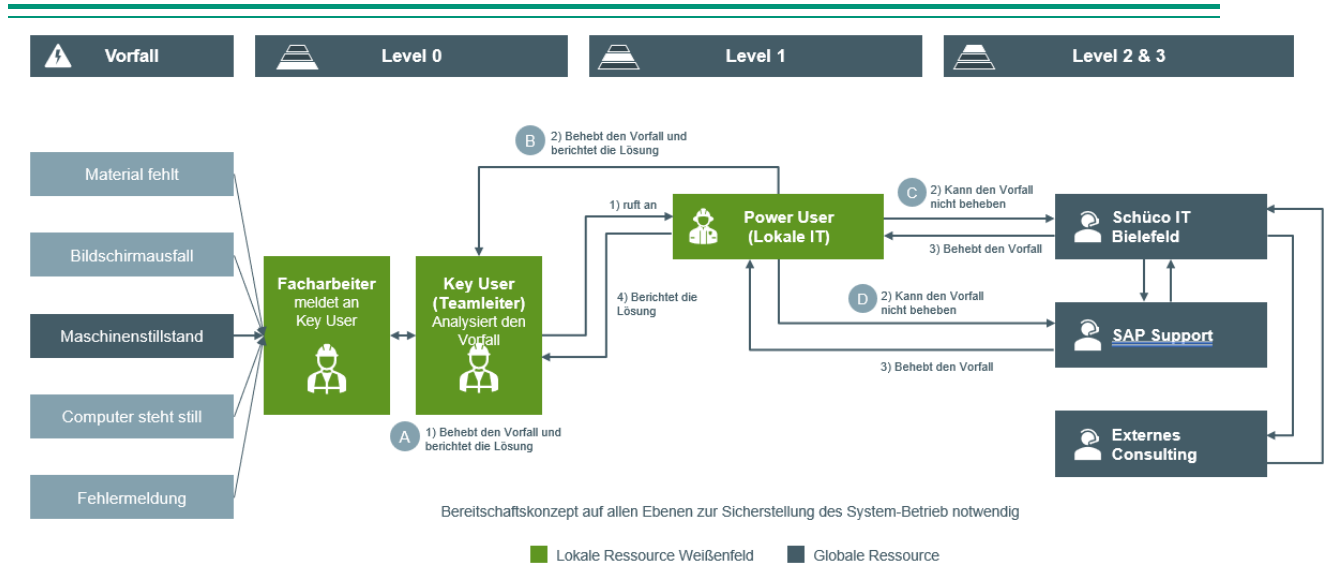


Figure 5: Example of a multi-level support process

The diagram shows a multi-level support process with the following elements:

5.2.1 Incident Occurs (Level 0)

The skilled worker notices a fault on the shop floor (such as machine downtime, missing material, system error). This condition is shown in the diagram as an “incident”. The skilled worker reports the incident to the key user or team leader.

5.2.2 Local Analysis and Processing by Key User and Power User (level 1)

- The key user (team leader) checks the fault.
- If necessary, the power user/local IT is involved.
- This level acts as the first point of contact for support and troubleshooting.
- The aim is to resolve the incident directly on site if possible.

5.2.3 Escalation to Central IT/SAP Support (Level 2)

If the incident cannot be resolved at level 1, it is escalated to central IT/SAP support. If the central IT/SAP expertise is not sufficient, external consulting is called in (for example, specialized SAP or system consultants).

5.2.4 Rectification and Feedback of the Fault

The incident is resolved technically and professionally at level 2 or 3. The developed solution is sent back to the organization from the higher level.

5.2.5 Standby Concept to Ensure System Operations

The diagram clearly shows that:

- local resources (skilled workers, key users, power users) and
- global resources (central IT, SAP support, external consulting)

must work together at all levels (levels 0-3) through a coordinated readiness concept in order to ensure long-term system operations.

5.3 Role Concept

The definition and implementation of the role concept in the SAP project is carried out in a structured, multi-stage process. The aim is to consistently translate functional requirements from the business processes into technical roles and authorizations in the SAP system and thus ensure secure, audit-compliant, and practical use of the solution.

5.3.1 Technical Definition of User Roles

The starting point is the technical view of the business processes in production, logistics, quality assurance, and IT support. Building on this foundation, the required user roles are defined first, for example:

- Production employees
- Picking employees
- Logistics employees
- Production team lead (key user)
- Work scheduling employees
- QA employees
- SAP DM administrators/system administrators (decentralized IT)

The main tasks and activities are initially described for each user role without consideration of the technical details. Among other questions, this includes:

- Which process steps does the role carry out?
- Which postings, completion confirmations, or approvals are carried out?
- Which evaluations or monitoring functions are required?

The result of this step is a functional role overview with clearly described responsibilities and activities.

5.3.2 Matching with SAP DM Standard Roles in SAP BTP

In the next step, the technically defined roles are matched with the existing SAP standard role catalog on SAP BTP. The required single roles are assigned to the respective user roles.

The result is a technically verified list of the required system functions and authorization objects for each user role.

5.3.3 Technical Design of Roles and User Assignment

The technical design of the roles in SAP BTP is based on the functional role descriptions and the documented system objects. For this purpose, the respective roles are assigned to the newly created role collections and updated in the system. The role collections can then be assigned to the respective users.

After the technical implementation, integration tests and, if necessary, production-related tests are carried out:

- Can the user carry out all the steps required for their tasks?
- Have impermissible authorizations (such administrative functions for operational users) been ruled out?
- Any anomalies are used to refine roles iteratively.

5.3.4 Documentation and Governance

The results of the role design are documented in a suitable manner. In particular, this includes:

- Description of the functional user roles
- Assignment to role collections
- Overview of roles by organizational unit/location
- Rules for requesting, assigning, changing, and revoking roles (for instance, in the context of user and authorization management)

This documentation forms the foundation for:

- Future role extensions and adjustments
- Audits and revisions
- Training and support processes

6 Summary and Next Steps

The successful integration of SAP Digital Manufacturing in production is a decisive step towards the digitalization and optimization of the production landscape. This guide has highlighted the key aspects, to help companies with planning, implementation, and scaling. The key findings of this guide are summarized below.

6.1 Summary of Key Topics

Architecture and Data Integration

- The Unified Namespace (UNS) offers a modern, flexible alternative to the traditional ISA-95 pyramid. It enables a consistent, standardized data architecture that breaks down silos and ensures real-time communication.
- The integration layer (middle layer), in particular the broker technology, is the core of the integration and ensures a seamless connection between IT and OT systems.

Use Cases and Benefits

- The three central use cases – results reporting, process reporting, and process interlocking – cover the most important requirements for transparency, efficiency, and security.
- By using standardized data models and protocols such as OPC UA and MQTT, data can be efficiently collected, processed, and analyzed.

Governance and Scaling

- A clearly defined template approach facilitates the reusability and scaling of integration solutions across multiple locations.
- Governance structures and coordinated change management are essential to ensure consistency and transparency.

Operations and Reliability

- A robust operations and support concept with clearly defined roles and escalation paths is crucial to ensure continuous operations.
- Backup-and-restore strategies and redundancy concepts minimize risks and ensure business continuity.

6.2 Key Messages

- **Standardization is the key:** Standardized protocols, data models, and templates reduce complexity and accelerate integration.
- **Unified Namespace is the model for the future:** The UNS provides for a flexible, scalable, forward-looking data architecture.
- **Governance and collaboration:** Clear roles and responsibilities are essential, along with effective change management.

- **Prioritize pilot projects:** Start with a clearly defined use case to achieve initial success and promote acceptance.
- **Operation and security:** Invest in a well thought-out operations and backup concept to minimize downtimes and ensure data integrity.

6.3 Recommendations for the Next Steps

Start a Pilot Project

- Choose a specific use case with high potential benefits (such as OEE reporting or process interlocking).
- Use the guide to plan and implement the integration.

Establish Governance Structures

- Define a centralized and decentralized role model for the operational organization.
- Set up a change management board to coordinate changes to templates and systems.

Assess Technology and Infrastructure

- Check the suitability of your existing shop floor systems for integration with SAP DM.
- Invest in edge connectivity software and middleware that meet your requirements for protocol support and scalability.

Plan for Long-Term Scaling

- Develop templates for recurring use cases and standardize the data structure in the Unified Namespace.
- Consider future requirements such as predictive maintenance and AI-supported analyses.

Promote Training and Acceptance

- Train your employees in the use of SAP DM and the associated systems.
- Promote acceptance by involving the specialist departments in planning and implementation.

6.4 Outlook

The digitalization of production is an ongoing process. By implementing SAP DM and using modern architectures such as Unified Namespace, companies are creating the foundation for future innovations. Topics such as artificial intelligence, machine learning, and predictive analytics will play an increasingly important role in the coming years.

The best practices and recommendations described in this guide provide a solid foundation for taking advantage of these developments and ensuring long-term competitiveness.

About the DSAG

The German-speaking SAP User Group e. V. (DSAG) is one of the most influential user associations in the world. Over 4,000 member companies and more than 70,000 members form a strong network that extends from SMEs to DAX companies and across all economic sectors in Germany, Austria and Switzerland (DACH). Based on this reach, the industry association gains in-depth insights into the digital challenges in the DACH market. DSAG uses this knowledge advantage to represent the interests of SAP users and pave the way to digitalization for its members.

Further information can be found at:

www.dsag.de, www.dsag.at, www.dsag-ev.ch

Imprint

We expressly point out that this document cannot anticipate and cover all regulatory requirements of all DSAG members in all business scenarios. In this respect, the issues and suggestions raised must naturally remain incomplete. The DSAG and the authors involved cannot assume any responsibility with regard to the completeness and suitability of the suggestions.

This publication is protected by copyright.
Unless expressly stated otherwise, all rights are:

German-speaking SAP® User Group e.V.
Altrottstraße 34 a 69190 Walldorf | Germany
Telefon +49 6227 35809-58
Telefax +49 6227 35809-59
E-Mail info@dsag.de
dsag.de

Any unauthorized use is not permitted. This applies in particular to reproduction, processing, distribution, translation or use in electronic systems / digital media.