MODAPTO [101091996]: Modular Manufacturing and Distributed Control via Interoperable Digital Twins

12.1.1 Understanding Smart Service Manifest structure and components

2025-08-11

The Smart Service Manifest structure in MODAPTO follows a carefully designed hierarchy that ensures comprehensive service documentation while maintaining clarity and usability. Understanding this structure is crucial for both service providers who create manifests and service consumers who need to interpret them for integration purposes.

The manifest begins with **Service Identification** elements that uniquely identify and classify the service within the Catalogue. The service name provides a human-readable identifier that should be descriptive and meaningful within the manufacturing context. The unique identifier (UUID) ensures that each service can be referenced unambiguously across distributed systems. Version information follows semantic versioning principles (major.minor.patch), enabling precise tracking of service evolution and compatibility. The service type designation (FMU, External, Internal, or Orchestrated) immediately indicates the service's architectural pattern and deployment requirements.

Service Capabilities form the functional heart of the manifest, describing what the service can accomplish within the manufacturing environment. This section goes beyond simple feature lists to provide contextualized descriptions of service functions. For example, a predictive maintenance service would detail its analytical capabilities, supported equipment types, prediction horizons, and accuracy metrics. The capabilities section uses structured descriptions that can be parsed programmatically while remaining understandable to human readers. This includes operational modes, processing capacities, and any limitations or constraints that affect service utilization.

The **Interface Specifications** section provides complete technical details necessary for service integration. For REST-based services, this includes endpoint URLs, HTTP methods (GET, POST, PUT, DELETE), required headers, authentication mechanisms, and payload formats. The interface documentation must be precise enough to enable automated client generation while providing examples and clarifications for manual integration. Special attention is given to error handling specifications, timeout configurations, and retry policies that ensure robust service consumption.

Input and Output Definitions leverage the Asset Administration Shell (AAS) standard to provide semantic clarity and interoperability. Each input parameter is defined as an AAS SubmodelElement with specific attributes: modelType indicates the element type (Property, File, Entity, etc.); idShort provides a unique identifier within the service context; valueType specifies the data type (string, integer, float, boolean, etc.); and additional constraints such as ranges, patterns, or enumerations. Output definitions follow the same structure but are organized as arrays to accommodate multiple return values. This standardized approach enables automatic validation, type checking, and data transformation between services.

The **Dependencies and Requirements** section documents all prerequisites for service operation. This includes technical dependencies such as required libraries, external services, or specific software versions. System requirements specify minimum and recommended hardware resources including CPU cores, memory, storage, and network bandwidth. Environmental requirements might include operating system specifications, container runtime versions, or specific configuration parameters. For FMU services, this section details the FMI version compliance and any co-simulation platform requirements.

12.1.1 Understanding Smart Service Manifest structure and components

Metadata and Supplementary Information enriches the manifest with business and operational context. Service ownership information includes the providing organization, responsible team, and contact details for support. Licensing terms clarify usage rights and any commercial considerations. Quality of Service (QoS) parameters define expected performance characteristics such as response times, throughput rates, and availability targets. Documentation references link to detailed user guides, API documentation, or training materials. Keywords and tags facilitate service discovery through Catalogue search functions.

1. References
2. MODAPTO Consortium. (2024). "D2.3 MODAPTO Architecture v2." MODAPTO Project Deliverable.
3. Platform Industry 4.0. (2023). "Details of the Asset Administration Shell – Part 1: The exchange of information between partners in the value chain of Industry 4.0." Federal Ministry for Economic Affairs and Energy (BMWi).
4. IEC 61499-1:2012. "Function blocks – Part 1: Architecture." International Electrotechnical Commission.
5. Modelica Association. (2023). "Functional Mock-up Interface 3.0 – Interface Specification." Available at: https://fmi-standard.org/
6. OpenAPI Initiative. (2023). "OpenAPI Specification v3.1.0." Available at: https://spec.openapis.org/oas/latest.html
7. JSON Schema. (2023). "JSON Schema: A Media Type for Describing JSON Documents." Available at: https://json-schema.org/latest/json-schema-core.html